

JYU DISSERTATIONS 621

---

Joakim Linja

# Advancing Nanomaterials Design using Novel Machine Learning Methods

---



UNIVERSITY OF JYVÄSKYLÄ  
FACULTY OF INFORMATION  
TECHNOLOGY

JYU DISSERTATIONS 621

---

Joakim Linja

# Advancing Nanomaterials Design using Novel Machine Learning Methods

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella  
julkisesti tarkastettavaksi yliopiston Agora-rakennuksen Alfa-salissa  
huhtikuun 13. päivänä 2023 kello 12.

Academic dissertation to be publicly discussed, by permission of  
the Faculty of Information Technology of the University of Jyväskylä,  
in building Agora, Alfa hall, on April 13, 2023 at 12 o'clock noon.



JYVÄSKYLÄN YLIOPISTO  
UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2023

Editors

Marja-Leena Rantalainen

Faculty of Information Technology, University of Jyväskylä

Ville Korkiakangas

Open Science Centre, University of Jyväskylä

Copyright © 2023, by author and University of Jyväskylä

ISBN 978-951-39-9517-1 (PDF)

URN:ISBN:978-951-39-9517-1

ISSN 2489-9003

Permanent link to this publication: <http://urn.fi/URN:ISBN:978-951-39-9517-1>

## ABSTRACT

Linja, Joakim

Advancing Nanomaterials Design using Novel Machine Learning Methods

Jyväskylä: University of Jyväskylä, 2023, 68 p. (+included articles)

(JYU Dissertations

ISSN 2489-9003; 621)

ISBN 978-951-39-9517-1 (PDF)

The rise of machine learning (ML) has revolutionized the usage of data. Researchers continue to develop new ways to use ML and find new targets to apply ML on. One of these areas of application is found in nanoscience. Nanoscience is a constantly expanding field with applications in almost every part of life, such as medicine, materials design, and consumer products. The experimental research of nanoscience is expensive, augmented by computational research. Computational research is, however, also resource-intensive and time-consuming due to the complexity of the simulation models. Machine learning promises to alleviate that strain. This work and the articles presented focus on a family of distance-based machine learning algorithms, Minimal Learning Machine (MLM), and Extreme Minimal Learning Machine (EMLM), in the context of computational nanoscience. Specifically in the context of monolayer protected nanoclusters (MPC).

The distance-based ML methods are studied as surrogates in feature selection and knowledge discovery. A set of benchmark, generated, and molecular dynamics-based datasets were used in the included articles. The performance of MLM was studied by using it as a surrogate, comparing it to other methods, and inspecting the effect of a solver on its function. EMLM was used as the ML model in feature selection and knowledge discovery. A set of scaling-focused benchmark datasets were developed based on the simulation data of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  MPC and a set of synthetic benchmark & development datasets were created to test the performance of a feature selection algorithm. A Mean Absolute Sensitivity (MAS) utilizing distance-based feature selection algorithm, *Distance-based one-shot wrapper*, was developed and then extended to *Feature Importance Detector*. An umbrella review was made to contextualize the one-shot wrapper to feature selection literature. The results prove the viability of distance-based ML methods in the context of computational nanoscience.

Keywords: Machine Learning, Distance-Based Regression, Nanoscience, MLM, EMLM, Hybrid Nanoparticles, Feature Selection, Knowledge discovery



## TIIVISTELMÄ (ABSTRACT IN FINNISH)

Linja, Joakim

Nanomateriaalien suunnittelun edistäminen käyttäen uusia koneoppimismenetelmiä

Jyväskylä: University of Jyväskylä, 2023, 68 s. (+artikkelit)

(JYU Dissertations

ISSN 2489-9003; 621)

ISBN 978-951-39-9517-1 (PDF)

Datan käsittely on mullistunut koneoppimismenetelmien yleistymisen myötä. Koneoppimiselle löydetään jatkuvasti uusia sovelluskohteita ja uusia sovellustapoja. Yksi näistä sovelluskohteista löytyy nanotieteen puolelta. Nanotiede on alati laajeneva tieteenala, jonka vaikutuksia löytää nykyään melkein jokaisesta elämän osa-alueesta, kuten lääketieteestä, materiaalisuunnittelusta ja kuluttajatuotteista. Nanotieteen kokeellinen tutkimus on kuitenkin kallista, mutta tätä voidaan lieventää laskennallisen tieteen keinoja hyödyntäen. Laskennallisen tieteen keinot nanotieteen saralla ovat kuitenkin itsessään raskaita ja aikaavieviä, johtuen tutkimuksen vaatimasta tarkkuustasosta. Laskennallisen tieteen resurssivaadetta voidaan keventää koneoppimisen keinoin. Tässä työssä ja mukaanotetuissa artikkeleissa keskitytään tarkastelemaan etäisyyspohjaisten koneoppimismenetelmien perhettä laskennallisen nanotieteen kontekstissa. Erityisesti yhden kerroksen suojaamien nanoklusterien (monolayer protected cluster, MPC) kontekstissa. Käytettyyn koneoppimismenetelmien perheeseen kuuluvat Minimal Learning Machine (MLM) ja Extreme Minimal Learning Machine (EMLM).

MLM:n ja EMLM:n toimivuutta ja suorituskykyä tutkitaan sijaismalleina, sekä muuttujanvalinnassa että tietämyksen tuottamisessa. Tutkimuksessa käytettiin aineistoja, joihin kuuluu suorituskykymittaukseen käytetyt, generoidut sekä molekyyliidynamiikkasimulaatioon perustuvat aineistot. MLM:ää tutkittiin käyttämällä sitä sijaismallina sekä tutkimalla sen toimintaa eri yhtälönratkaisijoiden avulla. EMLM:ää käytettiin muuttujanvalinnassa sekä tietämyksen tuottamisessa. Tutkimusta varten luotiin skaalausominaisuuksia luotaava,  $Au_{38}(SCH_3)_{24}$  MPC klusteriin perustuva joukko aineistoja sekä joukko synteettisiä aineistoja, joiden tarkoituksena on toimia suorituskykymittauksessa sekä menetelmänkehityksessä muuttujanvalinta-algoritmeille. Tutkimuksessa kehitettiin kaksi Mean Absolute Sensitivity (MAS)-pohjaista muuttujanvalinta-algoritmia: *Distance-based one-shot wrapper* sekä sen laajennos, *Feature Importance Detector*. Etäisyyspohjainen muuttujanvalinta-algoritmi kontekstualisoitiin muuhun lähdekirjallisuuteen laajan koostartikkelien koosteen avulla. Tulokset osoittavat MLM:n ja EMLM:n soveltuvuuden laskennallisen nanotieteen vaatimuksiin.

Avainsanat: Koneoppiminen, Etäisyyspohjainen regressio, nanotiede, MLM, EMLM, Hybridinanopartikkelit, Muuttujanvalinta, Tietämyksen muodostus

**Author**

Joakim Linja  
Faculty of Information Technology  
University of Jyväskylä  
Finland

**Supervisor**

Professor Tommi Kärkkäinen  
Faculty of Information Technology  
University of Jyväskylä  
Finland

Doctor Paavo Nieminen  
Faculty of Information Technology  
University of Jyväskylä  
Finland

Doctor Joonas Hämäläinen  
Faculty of Information Technology  
University of Jyväskylä  
Finland

**Reviewers**

Professor Pranad K Muhuri  
Faculty of Mathematics and Computer Science  
South Asian University  
India

Professor Xiao-Feng Gong  
School of Information and Communication  
Engineering  
Dalian University of Technology  
China

**Opponent**

Professor Tapio Pahikkala  
Department of Computing  
University of Turku  
Finland

## PREFACE

This is the personal voice part of this thesis, relating to experiences gained during the writing of this thesis. I'll be the first to admit that things did not go according to plan. There were unexpected stressors and problems that impacted the writing process. Some to a lesser extent, some to a greater extent. But that is life, isn't it?

One often-said advice is not to let the work become a boogeyman to you. This thesis, however, did manage to turn itself into a boogeyman for me. The extent I managed to turn it back into a thesis remains to be seen. On the same note, there is a saying that something has taken the *blood, sweat, and tears* to get done. Unfortunately for me, this thesis required those literally. I am sure I am not the only one with that fate among those who have written a thesis. Though one would hope that at least the blood part is not a common occurrence in a thesis project.

Throughout the years in the Finnish education system, I've considered myself a bad writer. Any writing assignment was automatically one of the most stressful and time-consuming part of any schoolwork. It is due to my experiences with writing assignments that I knew a project such as this thesis could not be taken lightly. And I didn't. It managed to surprise me anyway.

My roots are in physics, where it has been customary to start each work by somehow referencing the ancient Greeks or by referencing either Schrödinger's equation or Maxwell's equations. I specifically wanted to avoid that, although I kind of didn't due to starting with Feynman. I, however, did want to end this preface with a similar type of reference. Paraphrasing the words of Neil Armstrong back in 1969: *This thesis may have been a small step for science, but it was a giant leap for a man.*

## ACKNOWLEDGEMENTS

First, I would like to thank my supervisor Professor Tommi Kärkkäinen. I do not know how he does it, but he somehow always manages to make any situation or workload appear possible. He is also an endless source of ideas and questions.

I would like to thank my second supervisor, Doctor Paavo Nieminen. He is responsible for most of my ability to move forward in a writing process and for being one of the two people who introduced demoscene to me. He is also the person due to whom I ended up writing this thesis, as he was the first contact I had with the research project I worked on.

I would like to thank Doctor Joonas Hämäläinen, who at first was a roommate at the university and later on my third supervisor. He is also the first I've shared an office with who works on the same project as I did. I would like to thank Professor Hannu Häkkinen for enabling the project this thesis is related to and for the co-leadership with Professor Kärkkäinen throughout this time. I would like to thank Doctor Sami Malola for the guidance through these years and for reminding me that a previous hobby with computer art could still be relevant. I would like to thank Doctor Antti Pihlajamäki, the other doctoral researcher at the time of the project, for the peer support during the project.

I would like to thank the Academy of Finland for providing the funding for my research and the University of Jyväskylä for enabling my PhD studies. I would like to thank my groups of friends, the "gymnasium group", the "physicist group", and the "nano masters" group. Without you, life would be lonely and dull, and I would not be able to stand where I am without you. I would like to thank my family for supporting me, encouraging and for enabling me to exist in the first place.

Finally, I would like to thank my closest friend. I do not believe I would have managed to make it this far without her. She and the three mentioned groups of friends were the most important support I've had in the entire PhD process and the times in university before that. Thank you.

Thank you to all.

March 24, 2023

*Joakim Linja*

## LIST OF FIGURES

FIGURE 1	The number of active journals of the top 79 journals in the field of Nanoscience and Nanotechnology, according to Scimago Journal & Country rank [118] .....	16
FIGURE 2	Visualization of how the included articles are related to the research questions.....	19
FIGURE 3	The Q and T isomers of $\text{Au}_{38}(\text{SCH}_3)_{24}$ .....	22
FIGURE 4	Example of an MBTR descriptor ( $\mathbf{k1+k2+k3}$ ) for Benzene. ....	27

## LIST OF TABLES

TABLE 1	Summary of the research contributions .....	44
---------	---	----

# CONTENTS

ABSTRACT

TIIVISTELMÄ (ABSTRACT IN FINNISH)

PREFACE

ACKNOWLEDGEMENTS

LISTS OF FIGURES AND TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

1	INTRODUCTION .....	15
1.1	Background and motivations .....	15
1.2	Research questions .....	18
1.3	Structure of the thesis.....	18
2	NANOSCIENCE .....	20
2.1	Current nanoscience .....	20
2.2	Monolayer protected nanoclusters .....	21
2.3	Computational nanoscience and -technology .....	22
2.4	Feature extraction using descriptors .....	24
2.4.1	Types of descriptors .....	24
2.4.2	Many-body Tensor Representation .....	25
3	MACHINE LEARNING.....	28
3.1	Current machine learning.....	28
3.2	AI vs. Machine Learning .....	29
3.3	Feature selection.....	30
3.4	Machine learning methods .....	32
3.4.1	Distance-based machine learning methods.....	33
3.4.2	Other machine learning methods .....	34
3.4.3	On the evaluation of methods .....	36
4	SUMMARY OF ARTICLES AND RESEARCH CONTRIBUTION .....	39
4.1	PI: Monte Carlo Simulations of Au <sub>38</sub> (SCH <sub>3</sub> ) <sub>24</sub> Nanocluster Using Distance-Based Machine Learning Methods .....	39
4.2	PII: Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine? .....	40
4.3	PIII: Feature selection for distance-based regression: An umbrella review and a one-shot wrapper.....	41
4.4	PIV: Knowledge Discovery from Atomic Structures using Feature Importances .....	42
4.5	Summary of the research contributions.....	43
5	CONCLUSIONS AND DISCUSSION .....	45

YHTEENVETO (SUMMARY IN FINNISH) .....	49
REFERENCES.....	50
INCLUDED ARTICLES	

## LIST OF INCLUDED ARTICLES

- PI Antti Pihlajamäki, Joonas Hämäläinen, Joakim Linja, Sami Malola, Paavo Nieminen, Tommi Kärkkäinen, Hannu Häkkinen. Monte Carlo Simulations of  $Au_{38}(SCH_3)_{24}$  Nanocluster Using Distance-Based Machine Learning Methods. *Journal of Physical Chemistry A*, Vol 124, 23, 4827–4836, 2020.
- PII Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, Tommi Kärkkäinen. Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine?. *Machine Learning & Knowledge Extraction*, Vol 2, 4, 533–557, 2020.
- PIII Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, Tommi Kärkkäinen. Feature Selection for Distance-Based Regression: An Umbrella Review and a One-Shot Wrapper. *Neurocomputing*, Vol 518, 344–359, 2023.
- PIV Joakim Linja, Joonas Hämäläinen, Antti Pihlajamäki, Paavo Nieminen, Sami Malola, Hannu Häkkinen, Tommi Kärkkäinen. Knowledge Discovery from Atomic Structures using Feature Importances. *Manuscript*, 2023.



# 1 INTRODUCTION

Machine learning and nanoscience are two facets of science from different fields. One is the study of partially automatic, trainable tools, and the other is the study of events, objects, and phenomena in the nanometer scale. The combination of them is akin to computational nanoscience, with the exception that the computational part specifically focuses on machine learning. In the most common case, the term computational would mean computer simulation-based, such as Density Functional Theory (DFT) based molecular dynamics simulations.

This thesis approaches the topic of machine learning-assisted computational nanoscience mainly through the contributions made on the machine learning side of the duo. This chapter discusses the backgrounds and motivations, presents the research questions, and further explains the structure of this thesis.

## 1.1 Background and motivations

In the year 1959, Richard Feynman held his lecture *There's Plenty of Room at the Bottom* [50]. In his lecture, he painted a picture of miniaturizations reaching into atomic scale, delving into nanoscale manufacturing. His lecture can be considered a starting point for the field of nanoscience, but this point has been contested [186]. One could argue that the field of nanoscience started to find the wind under its wings during the 1980s. The advent of the Scanning Tunneling Microscope (STM) in 1981 by Gerd Binnig and Heinrich Rohrer [13] (for which they got Nobel's prize in 1986 [141]) can be considered to be a crucial enabling technology for nanoscience. The creation of STM also led to two other crucial tools in the field of nanotechnology, the Atomic Force Microscope (AFM) and Scanning Probe Microscope (SPM) [12, 92, 9]. Then, in 1986 came the book *Engines of Creation: The Coming Era of Nanotechnology* by Erik Drexler [44], which was the proper popularizer for nanotechnology according to research by Toumey [186].

Two now well-known discoveries gave the field popularizable results, which further cemented the interest towards nanoscience. They were *Buckminsterfullerene* [111]

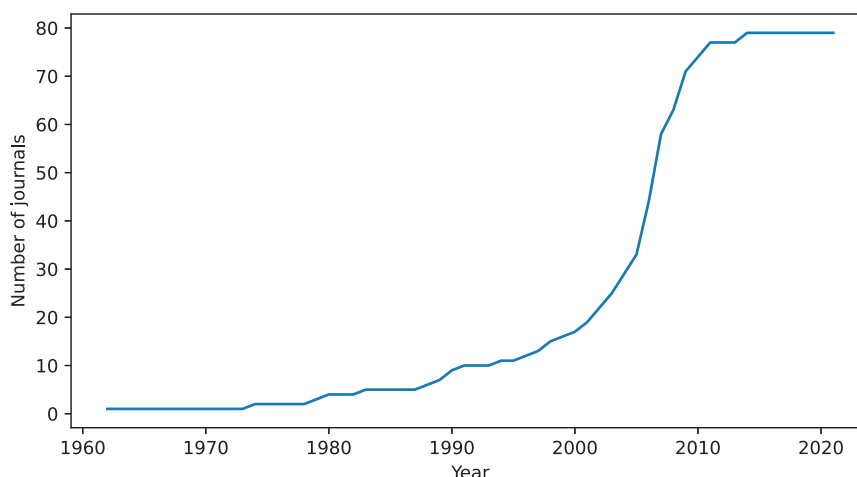


FIGURE 1 The number of active journals of the top 79 journals in the field of Nanoscience and Nanotechnology, according to Scimago Journal & Country rank [118]

and *Carbon nanotubes* [88]. Both were known before their physical structure was found. The years 1985 and 1991 are the years credited for the discovery of their physical structure. The year 1991 also saw another publication by Drexler et al. [43], *Unbounding the Future: the Nanotechnology Revolution*, which led to the term *nanomedicine* according to Bayda et al. [9]. The research on nanoscale carbon led to the creation of carbon dots (C-dots) in 2004 [203], which proved to hold various imaging, delivery applications, and others [9]. Later in 2017, Kinnear et al. published their extensive study on the shape of nanoparticles in the context of nanomedicine, showing the importance nanoscience and nanotechnology has held for medicinal applications [105]. One way to contextualize the progress of nanoscience in recent years is to look at the number of journals that report nanoscience or nanotechnology as one of their subjects. If we take Scimago Lab [118] as a source and plot the number of journals covering nanoscience and -technology as a function of years, we get Figure 1. From the figure, we can see that the field of nanoscience took off during 2000–2010.

Currently, nanoscience and -technology is a well-established field with research marching on and applications and products finding their way into general availability. One area of study in the field of nanoscience is the study of monolayer protected clusters (MPC) (not to be confused with the machine learning term, cluster, or a computation cluster). MPCs have, according to Tsukuda and Häkkinen, been a target of interest for over four decades [187]. The field began with attempts at finding and observing them, trying to figure out where they came from. Later, synthesis became more understood, and new experimental and theoretical discoveries made strides in the field [187]. Tsukuda and Häkkinen also point out that the interesting part about metal nanoclusters is that their properties deviate significantly from their bulk material versions. The protected part into the metal nanoclusters came from a need to passivate the clusters to prevent them from congregating together [187].

Returning to Feynman's talk for a moment:

If they (computers) had millions of times as many elements, they could make judgments. They would have time to calculate what is the best way to make the calculation that they are about to make. They could select the method of analysis which, from their experience, is better than the one that we would give to them [50].

–Feynman, 1959

We can see a hint on the second topic. The idea of a machine capable of making its own decision is by no means new. Science fiction authors have had a field day with the concept, probably for as long as science fiction has been a thing. However, in those cases, the topic is more accurately *artificial intelligence* instead of *machine learning*. The history of machine learning can be considered to begin in 1943 with the publication *A logical calculus of the immanent in nervous activity* by Pitts and McCulloch [130]. Although, one can make a claim that a simple linear regression is machine learning in a way that would take the beginnings of machine learning to a considerably earlier time in human history. In the article, *A logical calculus of the immanent in nervous activity*, Pitts and McCulloch lay the mathematical foundations for neural networks. The term *machine learning* was coined by A. Samuel in 1959 (there are claims that the actual first time for the term was 1952, but no verifiable source for this was found) [169].

An important step was taken in 1965 by Ivakhnenko and Lapa, as they published the first *Multi layer perceptron* and paved the way for *deep learning* later [90]. It was soon followed by the introduction of the *nearest neighbor* algorithm in 1967, which later led to pattern recognition [35]. The 1970's saw *backpropagation* in 1970 by Linnainmaa [125] and *Neocognitron* in 1980 by Fukushima [56]. The former led to automatic differentiation later on, and the latter led to neural networks. From there on out, an important step in reinforcement learning was made by Watkins in 1989 [195]. The 1990s saw the invention of the *Support Vector Machine* by Cortes and Vapnik [34] and an event which made the news, victory by *Deep Blue* against Gasparov in 1997 [28, 23]. A notable invention was published in 2001 by Breiman, the *Random Forest algorithm* [16], utilizing work by Ho [74], Amid & Geman [5] and Ho [75, 48]. The history of machine learning kicked up a gear in the 2000s and 2010s, as the increasing computational capacity and the introduction of GPU computation allowed for increasingly complex and data-heavy models. Perhaps the most notable achievement was made by the company DeepMind Technologies, when their *AlphaGo* won against a professional human Go player Lee Sol in 2016 [179, 10].

The glue between these two, nanoscience and machine learning, is found in computational science. Computational science studies scientific matters through the design, application, and use of mathematical models [113]. The beginning of computational science can be traced to Los Alamos Scientific Laboratory in 1945–1950 in the form of the *Markov chain Monte Carlo* computational method used in the research of nuclear weapons [132, 61, 113]. From there, it became a mainstay as the benefits of simulating an experiment, when possible, became apparent. Simulations are used, for instance, in modern car design. Previously if a

designer wished to optimize the air resistance of a vehicle, they had to construct either a scaled model or a full-sized model of the designed vehicle to test said air resistance. These days, the vehicle is first designed with computer-assisted design tools, and the virtual model is tested for air resistance in a simulated wind tunnel.

In a similar fashion, we have nanoscience. Experiments can be done in a lab, where one usually needs a cleanroom, reagents, measurement devices, and competent people working in the lab. The problems arise when the required equipment and reagents are expensive (and potentially dangerous) with low expected yield. In these cases, it would be better to do the first exploration for potential outcomes computationally. The computational side can then function as a guide to the experimental side so more effective experiments can be made. On the theoretical side, computational experiments may be used to attempt to discover previously unknown interests, which, if found, could then be verified by experimenters in a lab. Of course, computational science requires expensive hardware, so it is not a free lunch either. The simulation of nanoscale events requires a calculator which functions as the ruleset of the simulation. A more accurate ruleset means more realistic simulation results, but this usually comes with the caveat that the more accurate the ruleset, the more computational power it takes. The driving forces behind this thesis are as follows: physically accurate calculations are computationally expensive and time-consuming, and experimental research of hybrid nanoparticles requires expensive reagents and facilities. The project, which this thesis is based on aimed to develop a surrogate for DFT calculations or ML-based tools to be used in the computational research of hybrid nanoparticles to avoid both expensive calculations and experiments.

## 1.2 Research questions

This thesis was made as a part of two Academy of Finland-supported consortium projects: HNP-AI and MLNovCat [116, 143, 117]. The research questions follow the goals of the two projects. The goals are followed from the perspective of machine learning, emphasizing the methods and tools necessary for the realization of the project's goals. The research questions of the dissertation are as follows:

- RQ1** How to improve distance-based machine learning methods, focusing on **a)** scalability and **b)** feature selection
- RQ2** How to utilize the distance-based machine learning methods for nanoscience applications **a)** as surrogate models and **b)** for knowledge discovery

## 1.3 Structure of the thesis

Chapter 2 provides an introduction to nanoscience, hybrid nanoparticles, and descriptors. Chapter 3 presents machine learning and the models used in the in-

cluded articles. Chapter 4 contains a summary of the articles and the contributions given to the articles. Finally, Chapter 5 then concludes the thesis. Figure 2 presents a visualization of the research questions and their relation to the included articles.

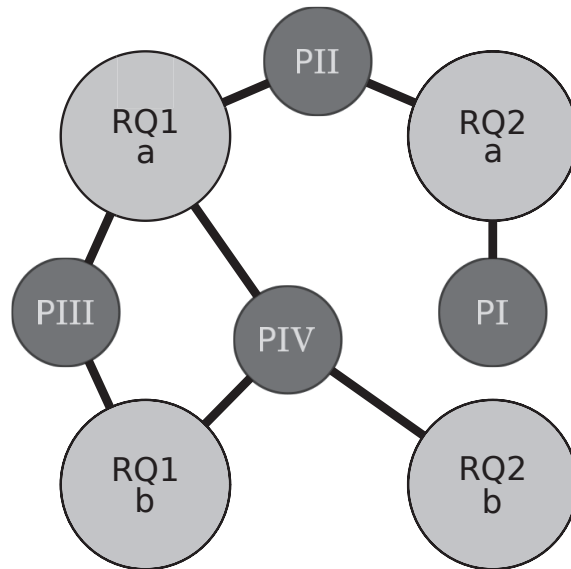


FIGURE 2 Visualization of how the included articles are related to the research questions

## 2 NANOSCIENCE

Nanoscience is the study of objects and phenomena at the nanoscale, which is in the scale of  $10^{-9}$ – $10^{-8}$  meters. It is uniquely multidisciplinary as a nanoscientist can be expected to have at least some level of knowledge on physics, chemistry, and cell- and molecular biology [181]. The scale is important. Rules which apply in one scale do not necessarily apply in another scale. The nanoscale is special in the sense that the rules of atomic physics are in effect at the same time as the physics and chemistry of complex systems [181]. The ultimate goal of nanoscience could be said to be the ability to construct nanoscale objects atom by atom the way the builder sees fit while following the rules of physics and chemistry [50, 44]. In other words, something like the replicator from *Star Trek*. This chapter introduces the reader to the current status of the field and presents the topics of monolayer protected nanoclusters, computational nanoscience, and descriptors.

### 2.1 Current nanoscience

To look into the current nanoscience is to look at surprisingly numerous parts of modern life. Considering the widespread usage of nanoscience and nanotechnology, it would go beyond the scope of this thesis to mention all possible areas. So instead, here are some highlighted ones.

Nanolithography, which pertains to the patterning required for integrated chips or processors, has allowed engineers to develop smaller and more powerful chips, allowing consumer electronics to get smarter and more complicated [157, 176]. The most penetrating outcome is the widespread usage of smartphones. In 2021, Kemp estimated that 66,6% of humans had a smartphone [102]. Smartphones have had significant cultural and societal impacts, which have been studied in, for example [81, 67, 18, 149, 4]. The increase in computational power has also allowed for the leaps and bounds taken on the computational and machine learning side (for more, see Section 2.3 and Chapter 3).

Nanomedicine is one of the most important fields of application due to its

potential to ensure better lives. One of these is the improvement of anticancer methods. Terms such as nanoscale carriers, nanovectors, or nanovehicles have been used to describe designed molecules which can target themselves and either deliver a drug to where it is needed or perform other functions, such as early disease marker or as an aid in imaging techniques [177, 191, 174]. Another way is to use nanoparticles to make cancer cells vulnerable to immunotherapy [129]. Anticancer therapy is not the only area of application. Other examples of medical applications include tissue engineering with nanostructured scaffolds: improved bone regeneration speed [155] and neural tissue engineering [171], as well as a potential aid for inherited blindness [71].

Medical applications are just one of the examples of recent successes. A research team developed a protective layer for spacecraft, which can simultaneously shield the craft from UV radiation and atomic oxygen but also harvest energy [39]. Another team has studied the possibility of using carbon nanotubes as a component in energy storage mediums [137]. Computer technology may also see other avenues of carbon nanotube applications, such as carbon nanotube processors [178]. These CNT-based processors have reached 16-bit operation [72]. An exciting development has also been made in the form of Aquabots, which are said to be ultrasoft liquid robots able to adapt their shape for various tasks [210]. On a more general note, nanoscience and -technology have found their place, for example, in environmental science and technology [206], bioscience [62], food industry [168, 151], materials design [100, 188, 127], catalysis [25, 103], and everyday consumer products [197].

Three examples are also given closer to the topic of this thesis. A software named DNAXiS was developed by Fu et al. in 2022, which allows for the automated design of 3D DNA origami [55]. Self-assembly is a way to partially bypass nanoscale fabrication, and it is a property that DNA holds. The other two are examples of machine learning applied to nanoscience. The first is BOSS, which uses a Bayesian model with DFT calculations to determine optimized results from given building blocks [185]. The second is an example of the autonomous discovery of nanostructures using machine learning [41]. It uses Gaussian process regression method to determine the next step in their experimental loop automatically. These two examples show glimpses of the future of nanoscience and machine learning.

## 2.2 Monolayer protected nanoclusters

Monolayer protected nanoclusters (MPC) or hybrid nanoparticles (HNP) are nanoparticles with a metal core, which then have an outer layer protecting it, passivating them wholly or partially [187]. The reason protective layers are desired for the nanoparticles is that the metal core tends to be reactive, which complicates their experimental study [187].

Regarding this thesis, the special focus is on  $\text{Au}_{38}(\text{SC}_2\text{H}_4\text{Ph})_{24}$  (where  $\text{C}_2\text{H}_4\text{Ph}$  may be shortened to R),  $\text{Au}_{38}$  for short [158]. In other words, it is a gold atom



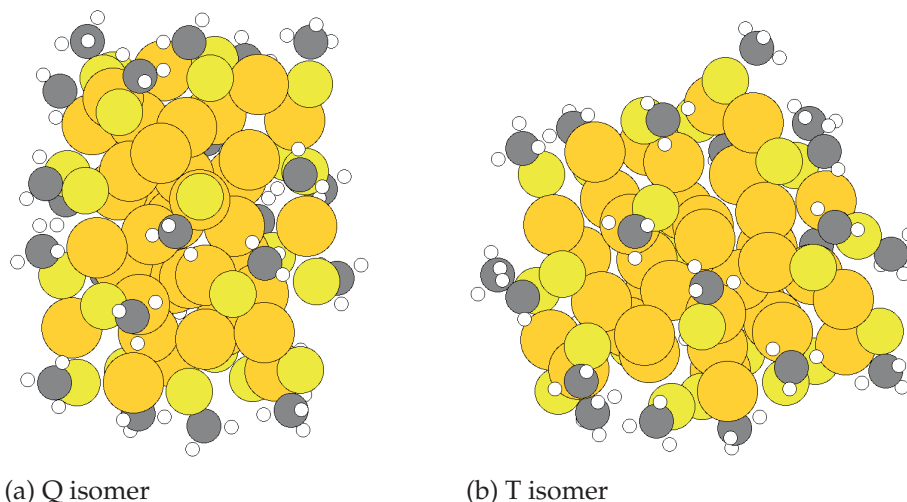


FIGURE 3 The Q and T isomers of  $\text{Au}_{38}(\text{SCH}_3)_{24}$

nanoparticle core protected by 24 thiolate ligands.  $\text{Au}_{38}$  has two experimentally discovered isomers,  $\text{Au}_{38}\text{Q}$  discovered by Qian et al. in 2010 [158] (presented in Figure 3a) and  $\text{Au}_{38}\text{T}$  discovered by Tian et al. in 2015 [183] (presented in Figure 3b).  $\text{Au}_{38}\text{Q}$  is a prolate biicosahedral with quasi- $D_{3h}$  symmetry and  $\text{Au}_{38}\text{T}$  has an oblate structure [97]. The synthesis of the two isomers is described in the respective papers.

The created group of datasets in article PII and the simulation data in article PIV relied on work by Juarez-Mosqueda et al., who did ab initio molecular dynamics simulations using Density Functional Theory (see Sec. 2.3) [97]. An important thing to note is that in the simulations performed by Juarez-Mosqueda et al., they replaced the thiolate ligands  $\text{SC}_2\text{H}_4\text{Ph}$  with computationally cheaper  $\text{SCH}_3$  which, through validation, proved to have affected the total energies by a range of  $-1.5$  to  $2.0$  eV [97]. The simulation data was later augmented by additional simulations by Antti Pihlajamäki and Sami Malola, mainly in high energies or special situations (see article PI).

### 2.3 Computational nanoscience and -technology

Due to this thesis having the combined topics of nanoscience and machine learning, a discussion on computational nanoscience is warranted. The focus will be on the topics present in the included articles.

Neto divided computational nanoscience and -technology into five categories [139]: molecular modeling, nanodevices simulation, high-performance computing, nanoinformatics, and nanotechnology-inspired computing. Of these categories, Neto considered high-performance computing to be just a prerequisite, with the addition that nanotechnology-inspired computing held no actual examples. Neto considered nanoinformatics to be the computational support, information storage and processing, interpretation, manipulation, sharing, and



dissemination of data [139]. Then, nanodevice simulation was determined to be the simulation of specific nanoscale devices, such as LEDs or quantum dots, the function of which was described by mathematical equations. A type of simulation without simulating each individual atom.

The molecular modelling category contains the topic of this thesis. It was further divided into four categories [139]: molecular mechanics, semi-empirical, quantum, and molecular dynamics. In a way, the split is between ways to calculate properties of time-independent structures (molecular mechanics, semi-empirical and quantum) and time-dependent simulations in the form of molecular dynamics.

The time-independent categories can be classified by the mechanics that they use. By the categorization of Neto, molecular mechanics simulate molecules and nanostructures using classical physics. Considering how the laws of physics are very different in the macroscale and nanoscale, this category can be easily pointed out as the most inaccurate one. Semi-empirical category relies on parametrizations and takes only the valence electrons into account. By only considering the valence electrons, semi-empirical methods gain accuracy compared to molecular mechanics without requiring too heavy computations. The last of the three is the quantum mechanics-based one. It produces the most accurate results but is computationally taxing as a tradeoff. The quantum-based solvers deal with the electronic structure of the target nanostructure in one way or another. Examples include the tight-binding approach, plane waves, and orbitals [192]. Probably the most well-known quantum mechanics-based calculator is the Density Functional Theory (DFT) [79, 108, 163]. DFT utilizes constructed potentials that act on the electronic structure and are also determined only by the geometry of the studied nanostructure [65]. One implementation for DFT calculations is given by the projector-augmented wave-based *GPAW* [46]. It is mentioned here since it is the calculator which was used in the molecular dynamics simulations with the  $\text{Au}_{38}$  hybrid nanoparticle present in the articles PI, PII, PIII, and PIV.

The time-dependent molecular dynamics category contains the simulations, which utilize a calculator to determine how each simulated component will move. Molecular dynamics simulations have been said to be one of the most fundamental tools of materials modeling [192]. The basic premise is that for each individual movable object, the next position and velocity are calculated based on where it is at the current timestep. Then when time advances in small steps, the simulated objects move according to the used calculator. The calculators range from the simple Lennard-Jones potential [94, 93, 123] to the quantum-based DFT.

Another application, besides the study of properties and molecular dynamics simulations, is found in local and global structural optimization. In these cases, a structural optimization algorithm, for example, Monte Carlo [132, 110, 61] or Genetic Algorithms [80, 53], is utilized with the addition of using either time-independent methods, time-dependent methods, or a combination of both.

As a final mention to the topic of computational nanoscience and -technology is the topic of this thesis, machine learning. The utilization of machine learning in nanoscience is still a relatively new concept, so all possible ways to apply ML in nanoscience have yet to be seen. However, several pathways have already

been taken. One pathway aims to use ML models as surrogates in the place of computationally heavy quantum-based calculators, such as DFT. The included article PI is an example of this. A second pathway is utilizing ML models in experimental guidance and materials discovery, as in [185] and [41]. Then there is the ML utilization in knowledge discovery, like in article PIV. And many others [17].

## 2.4 Feature extraction using descriptors

In the context of MPCs, the task of a descriptor is to translate information from simulation format into machine learning understandable format. Usually, the simulation format is a .xyz-file, where each individual atom is listed in order. The minimal amount of relevant information is the atomic number and the coordinates in 3D space. This format is machine learning compatible as it is. But not completely. In the "eyes" of a machine learning model, the order the atoms are listed in becomes part of the information. The problem is that the specific order of the atoms in a list is not physical or chemical information, so it is something the machine learning model should not have. The task of a descriptor is to perform feature extraction, i.e., to translate the atomic number and 3D coordinates so that the outcome has only relevant physical and chemical information. It should be chosen carefully, as the descriptor is a crucial component in the prediction ability of the utilized ML method [207]. Himanen et al. listed the properties of an ideal descriptor: invariant to translation and rotation, invariant to the order of the atoms, unique, continuous, compact, and computationally cheap [73].

### 2.4.1 Types of descriptors

There are two types of descriptors, local and global. Local descriptors describe the environment of a single atom, and global descriptors describe all present atoms at once, i.e., the whole molecule or the entire nanoparticle. Some descriptors are presented here. As these descriptors were not used in the included papers, they are left at textual description.

**Coulomb matrix:** represents molecules by using atomic energies and the inter-nuclear Coulomb repulsion operator to form the Coulomb matrix [164]. The issue with the Coulomb matrix is that it cannot represent periodic structures [47]

**Ewald sum matrix:** is an extension to the Coulomb matrix, which allows for periodic structures to be described [47].

**Sine matrix:** is a simplified version of the Ewald sum matrix. In it, the long-range electrostatic interaction is replaced with a simpler-to-compute expression [47].

**Atom-centered Symmetry Functions:** (ACSF) describe the environment of a single atom by the use of symmetry functions, five of which were presented in the paper which proposed ACSF [11].

**Smooth Overlap of Atomic Positions:** (SOAP) is a local descriptor for the environment of a single atom, using radial basis functions and spherical harmonics [8].

**Valle-Oganov descriptor:** is a descriptor specifically designed for periodic crystal structures, going through all atoms in defined unit-cells using a two-body correlation function [189].

**Property-labelled Materials Fragments:** (PLMF) is designed for crystal structures, and it is based on describing a crystal structure as a set of characterized subgraphs [89].

**Classical Force-Field-Inspired Descriptors:** (CFID) is a combined set of radial distribution functions, nearest-neighbor distribution, angle distributions, dihedral distributions as well as a set of chemical descriptors [32].

The included articles PII, PIV focused on the usage of a global descriptor, *Many-body tensor representation* (MBTR), which will be discussed separately next in Section 2.4.2.

## 2.4.2 Many-body Tensor Representation

Many-body tensor representation (MBTR) is a global descriptor for molecules and nanostructures developed by Huo and Rupp originally in 2017 as an arXiv paper, republished later in *Machine Learning: Science and Technology*, 2022 [85]. It has been implemented as a part of the Dscribe-package for Python by Himanen et al. [73]. MBTR primarily functions by constructing density distributions, which allows it to be permutation, translation, and rotation invariant. The included article PII touched on MBTR, and it was further detailed in article PIV, the expression there is mostly repeated here.

The implementation by Himanen et al. has three types of description available: "k1", "k2", and "k3" [73, 85]. Of the three available descriptions, the first is **k1**, or the number of elements in the described nanostructure. More specifically, it is  $F_1^{Z_1}(x)$  which reads as follows:

$$F_1^{Z_1}(x) = \sum_l^{|Z_1|} w_1^l D_1^l(x), \quad (1)$$

where

$$D_1^l(x) = \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(x-g_1(Z_l))^2}{2\sigma_1^2}}, \quad (2)$$

$$g_1(Z_l) = Z_l, \text{ the atomic number.} \quad (3)$$

In Eq. (1),  $x$  is the input for the formed distribution. The number of elements in an array is used to store the distribution of the number of elements in the nanostructure.  $Z_1$  refers to the atomic number,  $w_1^l$  is the weighting function, and  $\sigma_1$  is the standard deviation of the gaussian kernel [73, 85]. The sum in Eq. (1)

goes over all atoms of a type present in the described nanostructure. As Eq. (1) is basically "count the number of atoms of each type", the parameter  $\sigma_1$  with the Eq. (2) is important. If a nanostructure has no changes in the number of atoms, Eq. (2) allows **k1** to produce a smooth descriptor.

The second available description, **k2**, processes the distances between each present atom and then forms distributions from the distances. In other words:

$$F_2^{Z_1, Z_2}(x) = \sum_l^{|Z_1|} \sum_m^{|Z_2|} w_2^{l,m} D_2^{l,m}(x), \quad (4)$$

where

$$D_2^{l,m}(x) = \frac{1}{\sigma_2 \sqrt{2\pi}} e^{-\frac{(x-g_2(\mathbf{R}_l, \mathbf{R}_m))^2}{2\sigma_2^2}}, \quad (5)$$

$$g_2(\mathbf{R}_l, \mathbf{R}_m) = \frac{1}{|\mathbf{R}_l - \mathbf{R}_m|}. \quad (6)$$

The main difference from **k1** is adding another sum term over the atomic number  $Z_2$ .  $F_2^{Z_1, Z_2}$  in Eq. (4) goes through all atoms of two elements in order to form a distance distribution.  $w_2^{l,m}$  and  $\sigma_2$  hold the same function as they held with **k1**. The distance measure  $|\mathbf{R}_l - \mathbf{R}_m|$  is calculated between each individual atom pair, as defined by the sum in Eq. (4). As the inverse distance was used in articles PII and PIV, the distance measure in Eq. (6) is also the inverse version.

The third description, **k3**, adds one more atomic number as a variable and then uses the three observed atoms simultaneously to form an angle distribution. The form is similar to **k1** and **k2**:

$$F_3^{Z_1, Z_2, Z_3}(x) = \sum_l^{|Z_1|} \sum_m^{|Z_2|} \sum_n^{|Z_3|} w_3^{l,m,n} D_3^{l,m,n}(x), \quad (7)$$

where

$$D_3^{l,m,n}(x) = \frac{1}{\sigma_3 \sqrt{2\pi}} e^{-\frac{(x-g_3(\mathbf{R}_l, \mathbf{R}_m, \mathbf{R}_n))^2}{2\sigma_3^2}}, \quad (8)$$

$$g_3(\mathbf{R}_l, \mathbf{R}_m, \mathbf{R}_n) = \cos \angle (\mathbf{R}_l - \mathbf{R}_m, \mathbf{R}_m - \mathbf{R}_n). \quad (9)$$

As can be seen from Eq. (9), each observed group of three atoms produces a vector angle. Going through all atoms of each three types then forms a distribution. The weighting function  $w_3^{l,m,n}$  and standard deviation  $\sigma_3$  keep their meaning from **k1** and **k2**.

The outcome of MBTR with all three **k1**, **k2**, and **k3** is visualized for clarity in Figure 4 for Benzene (from article PIV). The distribution curve in Figure 4 is the mean calculated from the MBTR descriptors of each simulation step present in the source dataset. The molecular dynamics dataset used in the creation of the figure was published by Chmiela et al. and is available as a part of Symmetric Gradient

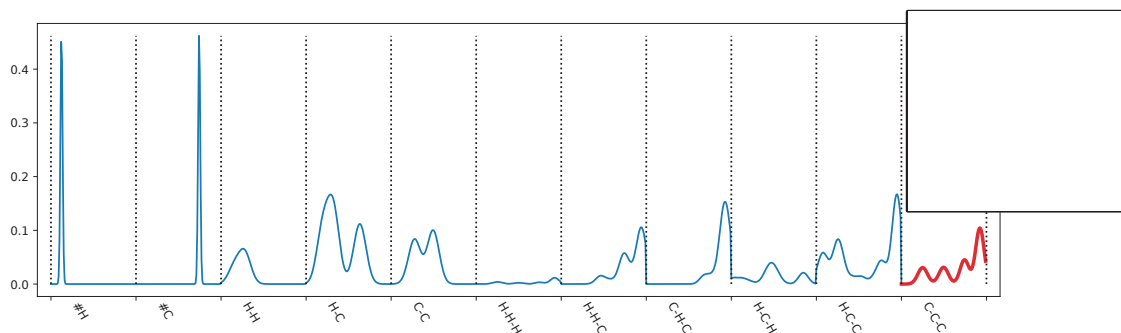


FIGURE 4 Example of an MBTR descriptor ( $\mathbf{k1}+\mathbf{k2}+\mathbf{k3}$ ) for Benzene. Each individual atomic interaction is labeled and given dashed vertical lines as borders. The first two, #H and #C represent  $\mathbf{k1}$ , the second three (H-H, H-C, and C-C) represent  $\mathbf{k2}$ , and the remaining interactions represent  $\mathbf{k3}$ .

Domain Machine Learning (sGDML) datasets [29]. As can be seen in Figure 4, the descriptor forms distribution peaks according to the geometry of the described atomic structure.

In the cases where the described nanostructure has no changes in the number of atoms,  $\mathbf{k1}$  is always unnecessary. However, should one wish to use a single MBTR object to describe multiple nanostructures,  $\mathbf{k1}$  would be almost mandatory. Due to how  $\mathbf{k2}$  and  $\mathbf{k3}$  capture the information of the described nanostructure, either one could be used to reconstruct the original geometry with a suitable autoencoder.

## 3 MACHINE LEARNING

Machine learning research is the study of computer programs that can improve themselves with additional data in relation to a task and the study of the related statistical-computational-information-theoretic laws [95]. The goal of machine learning research could be said to be the creation of a universal approximator. This chapter introduces the reader to the current status of the field, presents a discussion on the differences between artificial intelligence and machine learning, and discusses the machine learning methods used in the included articles.

### 3.1 Current machine learning

Looking at the field of machine learning in January 2023, the most significant thing at the moment are diffusion-based models and text transformers. In a surprisingly short time, these have formed into, at times, scarily good text-to-image generators. Two examples of this are *Dall-E* and *Midjourney* [159, 134]. The latter of which entered into controversial waters by winning an art contest against human artists [162]. A second example of current machine learning ability is the *ChatGPT*, a conversational model [146]. *ChatGPT* has the ability to "remember" the conversation it has had with a user and return to it should a need arise. It should be noted that *ChatGPT* is not able to pass the *Turing test*. For instance, *ChatGPT* has claimed that giraffes have long necks so that they may eat fish (the question was asked in Finnish). It is, however, able to write small programs and summarize encyclopedia entries. One of the well-known applications of machine learning methods are video and board games. The board game *Diplomacy*, characterized by conversations and deal-makings between players, saw an ML model with the capacity to not only play the discussion-based board game but even to win it [109]. Before this, there were specialized models for *Go* [179], *Dota 2* [145], and *Starcraft 2* [194]. These three learned by playing against themselves in order to accumulate levels of experience that would not have been possible vs. a human.

A discussion regarding the current state of machine learning is not com-

plete without the mention of self-driving cars. The concept is attractive to some. Since wouldn't it be nice if the work commute did not actually require attention? However, as of this writing and to the best knowledge of the author of this thesis, the idea of a fully self-driving car has not been achieved yet. This has also been reported by Zhao et al. in 2018 [209] and, more recently, Badue et al. in 2021 [6]. DARPA has organized competitions for self-driving cars. However, even when the competitions saw finalists, the competition environments could not replicate the potential chaos that a typical morning commute might be [6]. Lechner et al. [121] have recently shown a Neural Network model with improved autonomous driving ability using encapsulated input features. It may hold a new beginning for self-driving cars. For a look into the theoretical side of machine learning and as an example of the innovation the field sees, researchers have recently proposed a concept of liquid time-constant networks [69, 68, 138]. These types of neural networks have improved time-series prediction ability and adaptability. In traditional neural networks, the connection between neurons can be expressed as the weight coefficient between them. In these liquid neural networks, the connection is probabilistic and closer to nonlinear [138].

On a final note on current machine learning, is the way the advertising works with it [175, 1]. And that is personal data collection, personal data monetization, and related data mining. Social media, smart devices, websites, and even billboards can collect data. Data such as: "what did you look at", "when did you look at it", "where you looked at it", "how you looked at it" and "what else in addition did you look at" [156, 31]. And many others. This data is collected from each user every time they interact with a collection-capable entity. As there are many humans, it is understandably impossible to process this mass of data manually. So the processing is automated, and the automation is usually combined with machine learning for further ends. The usual further end is to decide what kind of adverts to show to a user in order to maximize advertiser profit.

### 3.2 AI vs. Machine Learning

An important topic of discussion is the distinction between AI and machine learning. Artificial intelligence (a term coined at least as early as 1956 in the Dartmouth workshop [165, p.36]) is a popular term. In fiction, it mainly refers to a sentience with digital existence. In popular media, it is used interchangeably with machine learning and the version in fiction. According to work by Samoili et al., the definition of AI is an ongoing topic, dependent on the presented context [33]. Samoili et al. listed the commonly found features in AI definitions, and they are as follows [33]:

- Perception of environment.
- Information processing.
- Decision making.
- Achievement of specific goals.



These features tell us that for something to be considered an AI, it has to have some level of autonomy in how it achieves the goals given to it. Let us then look at the definition of machine learning. A relatively recent definition for it was given by Mitchell:

A computer program is said to learn from experience  $E$  with respect to some task  $T$  and some performance measure  $P$ , if its performance in  $T$ , as measured by  $P$ , improves with experience  $E$  [135].

–*Mitchell, 1997*

In other words, a computer program that gets better at a task according to a metric should the program be given more data to work on.

A distinction between AI and ML can be made based on the definitions. A ML model is task-specific, it is given the data it needs to work with, and it is told to do the task it is given. An AI, on the other hand, by the common features of the definition, would not be limited to an operated, task-specific tool. A conglomerate of ML models could form it. An AI may appear as a machine learning model, should it be given such a task. It may appear as a sophisticated, almost automatic one but it does contain the necessary components needed for functioning as a machine learning model.

If we approach the topic from the other direction, could a single ML model appear to be an AI? The answer would be maybe it can if the given task  $T$  is "appear to be an AI in a given context  $C$ ". It would be necessary to give the ML model all the data it needs to create a model which appears to be generalized like an AI. In this case, all the needed data refers to all expected environments, all expected information in said environments, every expected decision, and every expected outcome. In practicality, a single ML model with that level of exorbitant information is impractical. One could reasonably expect the accuracy of the model to be on the abysmal side.

On the other hand, should the context  $C$  be well-defined in a limited scope? There should not be a reason why a ML model could not appear to be an AI. And maybe computational capacity increases to the point where the context  $C$  no longer needs to be that tightly limited.

In conclusion, should the context be chosen properly, an AI may appear as a machine learning model, and a machine learning model may appear as an AI. However, by definition, an AI is not a necessary component of a machine learning model, whereas a machine learning model is (most likely) a necessary component of an AI. Therefore, machine learning is a subset of artificial intelligence. The statement of machine learning being a subset of AI is backed by numerous sources, for example [101, 21, 133, 142].

### 3.3 Feature selection

As the included articles PIII and PIV both dealt with feature selection, it would be prudent to discuss the topic here. In machine learning, there is the term *Curse*



of *dimensionality*. The term pertains to several encountered phenomena as the dimension of data increases [193]. The most intuitively understood effect is that as the dimension increases, the required number of samples increases exponentially, should one wish to maintain a constant data sampling density. From the point of view of ML, the increasing number of observations (samples) or features or both causes increasing requirements in the learning task. It is a situation where feature selection may be employed.

The goal of feature selection is to reduce the dimension of the data in such a way that the accuracy of a machine learning model does not suffer in a meaningful manner [126, 59]. It should be noted that the literature regarding feature selection emphasizes classification problems, leaving feature selection in regression a less researched topic (see article PIII). The features in feature selection can be categorized by their relevance to the task. The definitions were given by Kohavi and John [107]: A feature is strongly relevant if its removal will result in the deterioration of ML model performance. A weakly relevant feature is not strongly relevant, but it may contribute to the accuracy of the model in some contexts. Furthermore, should a feature be neither strongly nor weakly relevant, it is categorized as irrelevant. Features can also be of different types. The main differences are between binary features, integer features, and continuous features. Each has its type of effect on the process of feature selection. Another point lies in the distribution of the values that a feature has. The data source can be basically anything, so one cannot assume that the data is, for example, normally distributed. Discussion by Gyuon and Elisseff [59] also points out that features can also be relevant in the presence of other features while being irrelevant on their own.

Feature selection algorithms can be given four categories based on how they utilize the ML model. Filters, wrappers, hybrids and embedded [59, 84, 112]:

- In the filter approach, a learning algorithm independent feature evaluation measure is used. Example evaluation measures include *Spearman R* [200], *Mutual Information* [45], and *Fisher-score* [49].
- The wrapper approach then specifically uses a learning algorithm in its evaluation methods. Examples include *greedy RLS* [147], *Quantum Whale Optimization Algorithm for Feature Selection* [2], *one-shot wrapper* (see article PIII), and *Feature Importance Detector* (see article PIV).
- Hybrid feature selection is a combination of a filter and a wrapper, typically using a filter to reduce the workload of the wrapper. Examples include *smart HGP-FS* [136], *Ensemble Information Theory based binary Butterfly Optimization Algorithm* [166], and *Dynamic Feature Importance based Feature Selection* [196].
- The embedded approach uses independent measures to build feature subsets of different cardinalities, using the learning algorithm to do the final selection form among the generated feature subsets. Examples include *Recursive Feature Addition* [64], *ESFS* [202], and *ODEFS* [27].

Another way to categorize feature selection algorithms is to go by the type of data or domain they are designed for. These are categories such as data type-based FS [124], optimization-based FS [40], application domain targeted FS [122]

and FS for multi-label classification [99]. Data type-based examples include conventional data, structured features, heterogenous data, and streaming data [124]. Optimization-based examples include evolutionary methods for single or multiobjective tasks and swarm intelligence-based methods [40, 140, 36]. There are more potential application domains that can be reasonably listed, but examples include renewable energy [167] and multimedia [122]. Multi-label FS is then feature selection, but the added labels force the inclusion of methodologies from multiobjective optimization [42, 99, 154].

The final presented way to categorize feature selection algorithms is by the search strategy they employ. These are sequential, exhaustive, exponential and random [112, 99]. Sequential search is iterative, feature by feature. Either additive or deductive. Exhaustive search goes through all possible states of the search space and can therefore guarantee the best possible solution. Exponential search strategies then aim to do what exhaustive search does while reducing the search space utilizing a suitable strategy. And finally, random search either bases itself on randomness or utilizes randomness as a part of another search strategy.

### 3.4 Machine learning methods

This section focuses on the machine learning methods used in the included articles. Throughout the history of machine learning, there have been numerous different ways to realize the concept of a trainable model that is then able to utilize what was learned. The most commonly known one is based on how the human brain works, imitating it to a degree as a neural network. As the distance-based models were the main focus with the added comparisons to neural networks, they are the primary focus here. There are three main ways in which a machine learning method may learn [165]:

**Supervised:** The ML model is given a set of inputs and the corresponding set of outputs. The task is to form an approximator that is able to map the input to the outputs in a way that new inputs may be given for new outputs. An example of this would be a set of handwritten numbers, the task of having the ML model recognize the written number.

**Unsupervised:** The ML model is given a set of data from which the model has to learn patterns on its own. An example of this would be a ML model tasked to find which data points in a dataset belong together (clustering).

**Reinforcement:** The ML model is given knowledge on how well it did in a task, good or bad. The model is then responsible for updating itself in order to maximize its desired outcome. An example of this is a model set to play a game. Victory means it did well and loss the opposite.

### 3.4.1 Distance-based machine learning methods

Distance-based methods construct a model where dissimilarity of the observations of data are in one way or another exploited [51, Ch. 8]. The distance is used to differentiate observations from each other and when predicting from previously unseen inputs [198, 51]. Two distance-based models are further elaborated here, Minimal Learning Machine (MLM) and Extreme Minimal Learning Machine (EMLM). Minimal Learning Machine (MLM) is a distance-based feature map machine learning method developed by de Souza et al. [38, 37] and it has been shown to have universal approximation capability [86]. It was primarily used in the included articles PI and PII. MLM is presented as in article PII, and the expression there is mostly reworded here.

The task of MLM is to estimate a continuous mapping  $f : X \rightarrow Y$ , given input  $X = \{\mathbf{x}_i\}_{i=1}^N | \mathbf{x}_i \in \mathbb{R}^D$  and output  $Y = \{\mathbf{y}_i\}_{i=1}^N | \mathbf{y}_i \in \mathbb{R}^S$ , with the multiresponse model

$$\mathbf{Y} = f(\mathbf{X}) + \mathbf{R}, \quad (10)$$

making it a supervised machine learning method [37]. The training of MLM begins by first selecting reference points from the data. The reference points are defined as  $\mathbf{R} = \{\mathbf{r}_k\}_{k=1}^K \subseteq \mathbf{X}$  and  $\mathbf{T} = \{\mathbf{t}_k\}_{k=1}^K \subseteq \mathbf{Y}$  where  $K \leq N$  is the number of selected reference points. A good practice is to select reference points that are as different from each other as possible [86]. The selected reference points are used to form the distance matrices,  $\mathbf{D}_x(i, k) = \|\mathbf{x}_i - \mathbf{r}_k\|_2$ ,  $\mathbf{D}_x \in \mathbb{R}^{N \times K}$  and  $\mathbf{D}_y(i, k) = \|\mathbf{y}_i - \mathbf{t}_k\|_2$ ,  $\mathbf{D}_y \in \mathbb{R}^{N \times K}$ . It should be pointed out that the  $\|\cdot\|_2$  norm can be replaced with other dissimilarity measures [37]. The L2 norm  $\|\cdot\|_2$  was used in the article PII.

These distance matrices then form the core of the MLM [37], by allowing linear mapping to be recovered from

$$\mathbf{B} = \left( \mathbf{D}_x \mathbf{D}_x \right)^{-1} \mathbf{D}_x \mathbf{D}_y. \quad (11)$$

Another thing to be pointed out is that should  $K = N$ , then no reference point selection is necessary and  $\mathbf{B}$  becomes

$$\mathbf{B} = \mathbf{D}_x^{-1} \mathbf{D}_y. \quad (12)$$

Once the coefficient matrix  $\mathbf{B}$  has been obtained, the learning part of MLM has ended and it can then be used to predict new outputs through solving a multilateration problem [37]:

$$\mathcal{J}(\tilde{\mathbf{y}}) = \sum_{i=1}^K \left( \|\tilde{\mathbf{y}} - \mathbf{t}_i\|_2^2 - \delta_i^2 \right)^2, \quad (13)$$

where  $\delta = [\|\tilde{\mathbf{x}} - \mathbf{r}_1\|_2^2, \dots, \|\tilde{\mathbf{x}} - \mathbf{r}_K\|_2^2] \mathbf{B}$ .

A point that was important in article PII, that will also be pointed out here is that due to Eq. (11), the number of features does not have an effect on the

dimensions of the distance matrices of MLM. This makes MLM capable of handling large numbers of features and large target dimensions without scaling issues.

In the same family of machine learning methods as MLM is the Extreme Minimal Learning Machine (EMLM) first proposed by Kärkkäinen in 2018 [114, 115]. It was primarily used in the included articles PIII and PIV as a part of the *Mean Absolute Sensitivity* (MAS, presented in article PIII) based feature ranking algorithm. Like MLM, EMLM is a distance-based supervised machine learning method with only one hyperparameter, the number of reference points. Also, like MLM, EMLM has been shown to have universal approximation capability [115, Remark 1]. The expression of EMLM found in article PIII is mostly repeated here.

EMLM shares its task with MLM and also begins with the selection of reference points  $\mathbf{R} = \{\mathbf{r}_k\}_{k=1}^K \subseteq \mathbf{X}$  from input data  $X = \{\mathbf{x}_i\}_{i=1}^N | \mathbf{x}_i \in \mathbb{R}^{n \times N}$ . The output data is  $Y = \{\mathbf{y}_i\}_{i=1}^N | \mathbf{y}_i \in \mathbb{R}^{p \times N}$ . The selected reference points are then used to form the distance matrix  $\mathbf{H} \in \mathbb{R}^{m \times N}$ :

$$(\mathbf{H})_{ij} = \|\mathbf{r}_i - \mathbf{x}_j\|_2, \quad i = 1, \dots, m; j = 1, \dots, N. \quad (14)$$

While MLM has two distance matrices, EMLM has only one. The distance regression weights  $\mathbf{W} \in \mathbb{R}^{p \times m}$  which are the goal of the training of the EMLM, are then solved from [115]:

$$\mathbf{W} \left( \mathbf{H}\mathbf{H}^T + \frac{\alpha N}{m} \mathbf{I} \right) = \mathbf{Y}\mathbf{H}^T. \quad (15)$$

Once  $\mathbf{W}$  has been computed, the model can be used to predict new outputs  $\tilde{\mathbf{y}}$  with unseen inputs  $\tilde{\mathbf{x}}$  using

$$\tilde{\mathbf{y}} = \mathbf{W}\tilde{\mathbf{H}}, \quad (16)$$

where  $(\tilde{\mathbf{H}})_{i1} = \|\mathbf{r}_i - \tilde{\mathbf{x}}\|_2, \quad i = 1, \dots, m$ .

### 3.4.2 Other machine learning methods

The distance-based MLM and EMLM were not the only ML methods used in the articles. The other ML methods present in the articles are discussed here. It should be noted that the ML methods presented here do not represent all potential ML model structures.

Neural Networks (NN) are a type of ML models that simulate a biological brain with a collection of activation functions, or neurons, which are organized in layers where each neuron is linked to each neuron of adjacent layers (in the case of deep neural networks) [82, 172]. Feedforward neural networks (FNN) were the primary comparison to MLM and EMLM, and FNNs were used in the included articles PII and PIII. FNNs were, however, not discussed in detail as they were added to the papers due to reviewer demands. The FNNs will be discussed more here. They are among the simplest forms of a neural network in the way that the connections between nodes travel only in one direction, as an acyclic graph [172]. In a FNN, the first layer is the input layer of a size  $N_0$ , which is also the number of features present in the input. The input layer is followed by  $k \geq 1$  hidden layers. The size of each hidden layer is determined by the user and the task at hand. The

hidden layers are then followed by the output layer, which has  $N_k$  nodes in it. The nodes in a feedforward neural network determine the way it functions and the way the network handles its inputs. Each node calculates the weighted sum of the outputs of the previous layer through a network function [161]. The outcome of the activation function is then passed onto the neurons of the next layer. A typical value for  $N_k$  would be 1 in a regression task, the number of dimensions in the output. That is not the case in multi-target regression or classification.

The teaching of a neural network is done by minimizing a cost function, which in turn is done by adjusting the weights between the neurons of the network. The teaching is typically done using *ADAM*, a method for stochastic optimization [104]. The training has to be done meticulously, as neural networks are prone to overfitting [26]. The FNNs in the included articles were of a simple form, a shallow one with one neuron layer and a deep one with three hidden layers. One could say that two networks like that do not represent neural networks as a whole. And they would be correct. The so-called FNN-2 and FNN-4 were chosen to have the same "user experience" as the compared MLM and EMLM (see Sec. 3.4.1) would have. It meant that their network configuration was based on a simple feature number-based heuristic, and their hyperparameters were left at their implementation default values.

In addition to the primary comparison FNN, article PIII used the following concisely presented ML methods as a comparison.

**Decision tree:** A method that employs a divide and conquer strategy to divide the given data into a set of attribute nodes, which then have leaves containing the next attribute node or result node [180, 24, 165]. One way to describe a decision tree is a strategically formed set of if-else questions. Decision trees are commonly used in classification, but they can also be used for regression tasks.

**Random forest:** An ensemble type machine learning method; it is a collection of randomly initialized decision trees which together form the final answer [74, 16]. Each tree in the forest is given a random part of the dataset to learn, and high enough numbers of trees have guaranteed levels of accuracy [16]. Breiman also mathematically showed how random forests do not overfit.

**Support Vector Machine:** Proposed by Vapnik and colleagues in the early 1990's [15, 34]. It is a supervised machine learning method that nonlinearly maps input vectors to a high-dimension feature space, where a linear decision surface is constructed [34]. Hearst presented the form that SVM takes in regression tasks [70]:

$$f(\mathbf{x}) = \sum_{i=1}^l v_i \cdot k(\mathbf{x}_i, \mathbf{x}) + b, \quad (17)$$

where  $k(\mathbf{x}_i, \mathbf{x})$  represents the used kernel,  $v_i$  are weights and  $b$  is the bias vector.

**Lasso regression:** A linear model first proposed by Santosi et al. in 1986 and later independently rediscovered by Tibshirani in 1996 [170, 184]. The name comes from the latter publication. Tibshirani put the function of the Lasso

regression in this manner: "Lasso minimizes the residual sum of squares subject to the sum of the absolute value of the coefficients being less than a constant" [184]. More formally, the objective of Lasso is to solve the estimate  $(\hat{\alpha}, \hat{\beta})$  [184]:

$$(\hat{\alpha}, \hat{\beta}) = \arg \min \sum_{i=1}^N y_i - \alpha - \sum_j \beta_j x_{ij} \quad , \quad (18)$$

with subject to

$$\sum_j |\beta_j| \leq t. \quad (19)$$

In (18),  $N$  is the number of observations,  $y_i$  are the targets,  $\alpha$  is the constant coefficient, and  $\hat{\beta}$  is the coefficient vector.

**Ridge regression:** Named by Hoerl et al. in 1970, originates from ridge analysis, also by Hoerl from 1959 [77, 78]. Ridge regression adds a small positive quantity to the diagonals of  $\mathbf{X} \mathbf{X}$  matrix product in order to ensure that all solutions can be obtained [77]. The formal version of ridge regression is [131]:

$$\hat{\beta}(k) = (\mathbf{X} \mathbf{X} + k\mathbf{I})^{-1} \mathbf{X} \mathbf{Y}, \quad k \geq 0, \quad (20)$$

where  $\mathbf{X}$  is the input data,  $k$  is the ridge parameter and  $\mathbf{Y}$  are the targets of the dataset.

### 3.4.3 On the evaluation of methods

In optimization, one uses a "fitness function" to measure the "goodness", the state of the optimization. In ML, the model accuracy holds a similar function. The precise way the accuracy of a ML method can be measured depends on the type of ML and the type of task at hand. As the included articles PI, PII, PIII, and PIV all dealt only with supervised learning regression tasks, it will be the focus here. Evaluation metrics for unsupervised learning are, for example, discussed in [148] and for reinforcement learning in [96, 120].

There are two types of tasks in supervised learning, classification (for categorical targets) and regression (for continuous targets) [19]. An evaluation metric that works for one does not necessarily work for the other. Metrics for classification are discussed, for example, in [20, 83]. Formally, the task of an evaluation metric is to measure how much a predicted output  $\mathbf{Y}^*$  is different from the actual target  $\mathbf{Y}$  [3]. Three such metrics are presented here. Do note that these are not the only three, as, for example, the Scikit-learn library [152] for Python [190] by itself provides 15 regression task metrics. The first of the three is:

**Mean Absolute Error:** (MAE) [199], which is defined as

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |\mathbf{Y}_i^* - \mathbf{Y}_i|, \quad (21)$$

where  $N$  is the number of observations,  $\mathbf{Y}_i^*$  the  $i$ :th predicted output and  $\mathbf{Y}_i$  the  $i$ :th desired output. The second of the three is:



**Mean Square Error:** (MSE) [83], which is defined as

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (\mathbf{Y}_i^* - \mathbf{Y}_i)^2, \quad (22)$$

where  $N$ ,  $\mathbf{Y}_i^*$  and  $\mathbf{Y}_i$  are the same as with MAE in Eq. (21). The final of the three, and the one used in the included articles PI, PII, PIII, and PIV is:

**Root Mean Square Error:** (RSME) [7], which is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\mathbf{Y}^* - \mathbf{Y})^2}, \quad (23)$$

where  $N$ ,  $\mathbf{Y}_i^*$  and  $\mathbf{Y}_i$  are the same as with MAE in Eq. (21).

RMSE is widely used in regression tasks [20]. Willmott and Matura discussed the differences between the three in [199, Table 1] (MAE requires one's own addition of the mean-part into the numbers for  $\sum |e_i|^2$ ). The conclusion Willmott and Matura reach in their paper is that MAE is superior to RMSE. However, this point has been contested by Chai and Draxler [22], with discussion continued in [76] and [98]. Hodson points out in [76] that the discussion of RMSE vs. MAE has been going on for 200 years. Hodson also points out that the discussion is pointless due to both RMSE and MAE having their strengths and weaknesses. Time will tell if a consensus is reached one day.

The use of metrics like RMSE is only one part of the evaluation of a ML method. An example for another part would be  $k$ -fold cross validation [106], utilized in articles PII, PIII, and PIV. The premise of  $k$ -fold cross-validation is that the utilized dataset is split into mutually exclusive  $k$ -folds [106]. These folds are then used as two groups, training dataset, and testing dataset [66, 165], by sequentially using one of the folds as a testing dataset while the others function as a training dataset. The test dataset is used to create test error by attempting to predict it with the model created using the training dataset. In *model selection*, the fold configuration which produced the best test error is selected as the result model [208]. Cross-validation may be used to evaluate the generalizability of the trained ML model, in which case the individual fold errors are combined and used as one (i.e., average) [204]. Additionally, before cross-validation is performed, a part may be taken aside from the available data to be used as a validation dataset. The validation dataset is used as the final check on the success of a ML model, and similarly to test error, it is used to produce validation error [204, 165]. Note that in the literature, the use of the terms test and validation is not consistent. Others call the final check test, and others call it validation. There is at least one more way to utilize cross-validation, and that is in hyperparameter optimization. It is due to the generalization estimate, which can be measured using cross-validation. Usually, methods such as Neural Networks require careful hyperparameter optimization to maximize their potential [205]. A point that was repeated in the articles PII and PIII is that MLM and EMLM are both single hyperparameter methods (the

number of features), thus in their case, allowing cross-validation to be used solely for the evaluation of generalizability.

On the topic of datasets, they can also be used in evaluating the performance of a ML method. The source of the content in a dataset is its own variable. With synthetic datasets, the source is the creator. It allows for purposeful design, which allows the dataset to test a specific capability of a ML method. From the point of view of feature selection, a synthetic dataset also lets one know exactly which features should be selected. There is no guarantee that the same can be said for datasets constructed from real-world data. In literature, the typical examples are benchmark datasets. The primary function of a benchmark is to provide a reference point to be used. In this aspect, benchmark datasets have been created [144, 57, 14, 153, 128] and used [54, 201, 60, 58, 119]. In addition to literacy on the topic, the included articles PII and PIII both contained datasets made for benchmarking and the development of further ML methods. More specifically, in article PII, a set of datasets intended for evaluating the effects of scaling on a ML method was published, and in article PIII, a set of synthetic, purpose-designed datasets were published. The datasets in both articles were also used to compare ML methods to each other.



## 4 SUMMARY OF ARTICLES AND RESEARCH CONTRIBUTION

In this chapter, the included articles are discussed, and the contributions of the author of the thesis are presented. In addition, this chapter presents the contributions of the included articles as a whole. Even though the main topic of this thesis is nanoscience and machine learning, the main topic of the included articles is on the machine learning side.

### 4.1 PI: Monte Carlo Simulations of $\text{Au}_{38}(\text{SCH}_3)_{24}$ Nanocluster Using Distance-Based Machine Learning Methods

Pihlajamäki, A; Hämäläinen, J; Linja, J; Nieminen, P; Malola, S; Kärkkäinen, T; Häkkinen, H. Monte Carlo Simulations of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  Nanocluster Using Distance-Based Machine Learning Methods *Journal Physical Chemistry A*. **2020**, *124*, 4827–4836.

DOI:10.1021/acs.jpca.0c01512

**Research aims** The aim was to apply distance-based machine learning methods with Monte Carlo simulations to show the effectiveness of distance-based machine learning methods in the function of a surrogate model in computing potential energy.

**Data and methods** Simulation data of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  Q and T isomers between 400 and 1200 K, generated by Juarez-Mosqueda et al. [97], was used in conjunction with MBTR to form training data for the distance-based machine learning method. In the process of the study, the source data was extended with additional molecular dynamics simulations in order to combat found unrealistic results.

MLM and EMLM models were then used to predict potential energies of the  $\text{Au}_{38}(\text{SCH}_3)_{24}$  nanocluster in Monte Carlo simulations. The results were analyzed and validated.

**Main results** The article shows that distance-based ML models EMLM and MLM are able to function as DFT surrogate models given a suitable training dataset, allowing the potential energy surface exploration to be run with several orders of magnitude shorter CPU time as would have been needed with DFT. Additionally, the article shows that Monte Carlo is an efficient potential energy landscape exploration strategy.

**Research contributions** The article shows the viability of MLM and EMLM as DFT surrogates and presents a comparison of results between the distance-based models and DFT. The article also shows the viability of utilizing Monte Carlo search strategy in combination with ML methods.

**Author's contributions** The author mainly contributed to the background work, setup of the research environment, and setup of the programming side.

## 4.2 PII: Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine?

Linja, J; Hämäläinen, J; Nieminen, P; Kärkkäinen, T. Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine? *Mach. Learn. Knowl. Extr.* **2020**, *2*, 533–557. DOI:10.3390/make2040029

**Research aims** The main point of article PII was to experiment on a randomized SVD solver with the goal of achieving a speedup in the learning process of *Minimal Learning Machine*. In addition, a comparison between MLM and a shallow as well as a deep feedforward neural network was made while studying the effect of scale on the function of each machine learning method.

**Data and methods** Six of the used datasets were mostly well-known and publically available machine learning datasets. One of the datasets, *Mnist*, is for classification tasks, whereas the rest were for regression tasks. *Mnist* was included due to it being a well-known benchmark dataset with 70000 observations.

In addition to the six datasets, we created nine more, which were based on simulations made by Juarez-Mosqueda et al. [97], provided to us by our research collaborators led by Professor Hannu Häkkinen. These nine were specifically made to probe the effect of scale on the tested machine learning methods in the context of nanoscience.

The main methodology was experimental science, as we ran simulations and measured the performance and time for each tested machine learning method.

**Main results** The main results from article PII were that direct solvers provide the best accuracy when utilizing MLM. However, when the highest possible ac-

curacy is not necessary, randomized solvers are a comparable option. That was also detected with MLM, as the reduction of solver accuracy does not translate through MLM as it is. Giving the result that a speedup is possible to achieve with randomized solvers in the context of MLM.

The other main result is that a distance-based kernel method is able to outperform shallow and deep FNNs in the tested cases. Granted, the FNNs were not finely tuned with time and effort but were taken as “off the shelf” as their comparison, MLM, can be taken.

**Research contributions** This article showed the robustness of a distance-based approach and that it can compete with neural networks. The nanoscience-based nine datasets for a scaling study can be used by others should they desire to test their model on a group of datasets where the information in the data does not essentially change.

**Author’s contributions** The author of this thesis contributed to the practical research work, result analysis, presentation, selection of datasets, selection of tested solvers (with the exception of *SVP update*), programming of the scripts used to run the tests, creation of the scaling datasets and main writing of the article. The author was not the only writer in the article.

### 4.3 PIII: Feature selection for distance-based regression: An umbrella review and a one-shot wrapper

Linja, J; Hämäläinen, J; Nieminen, P; Kärkkäinen, T. Feature Selection for Distance-Based Regression: An Umbrella Review and a One-shot Wrapper. *Neurocomputing*. 2023, 518, 344–359. DOI:10.1016/j.neucom.2022.11.023

**Research aims** After the speedup experiments of article PII, the next step was to find a way to reduce the number of features required by a machine learning method. Simultaneously, the aim was to study feature selection with distance-based ML models, contextualize the research among feature selection research in general, and develop and propose a feature selection algorithm as well as compare it to other available feature selection algorithms.

**Data and methods** Two sets of datasets were used in this study. The first set is a set of created toy datasets that were primarily designed for feature selection algorithm development. Of these, the first six were created by us, and the latter four were created by Sun et al. [182]. Inspired by the nine scaling datasets in article PII and in conjunction with feature selection competition datasets where the correct result remains unknown, we wanted datasets where the correct features were known from the start. The second set is a group of 13 publically available and well-known machine learning datasets, some intended for classification and

others for regression.

The methodology was the experimental measurement of the performance of the developed feature selection algorithm in comparison to other known algorithms. In addition, we compared the function of our feature selection algorithm when the feature ranking algorithm was switched to be a FNN-based one.

**Main results** We made an extensive umbrella review to contextualize distance-based regression feature selection to the feature selection research in general. We developed and proposed a synthetic dataset and the one-shot wrapper feature selection algorithm. We evaluated the performance of our developed feature selection algorithm by utilizing the developed set of benchmark datasets and by extensive comparison to other feature selection algorithms. We also tested the function of the algorithm should the EMLM core be switched to a Neural Network-based one. The result of each comparison was that, in general, the EMLM-MAS combination held the best results.

**Research contributions** This article had three main contributions. First is the extensive umbrella review on feature selection methods for regression datasets. The second is the development of the one-shot wrapper feature selection algorithm and the extensive comparisons to other feature selection methods. The third contribution is the development of the toy datasets to be used in feature selection development and benchmarking.

**Author's contributions** The author of this thesis contributed to the practical research work, result analysis, presentation, selection of datasets, creation of synthetic datasets, selection of tested algorithms, programming of the scripts used to run the tests, the final form of the feature selection algorithm, and main writing of the article (with the exception of the umbrella review). The author was not the only writer in the article.

#### 4.4 PIV: Knowledge Discovery from Atomic Structures using Feature Importances

Linja, J; Hämäläinen, J; Nieminen, P; Pihlajamäki, A; Malola, S; Häkkinen, H; Kärkkäinen, T. Knowledge Discovery from Atomic Structures using Feature Importances. *Manuscript*.

**Research aims** In a continuation from article PIII, the aim was to use feature selection on datasets based on molecular dynamics simulations and to analyze the results using domain knowledge.

**Data and methods** Two types of molecular dynamics datasets were used in this study. The gold nanocluster data also used in the articles PI and PII, as well as 11

organic molecule simulation datasets from sGDML (symmetric Gradient Domain Machine Learning) [30, 173, 29].

The main methodology was to use the datasets and domain knowledge to verify the function of the updated version of the feature selection algorithm presented in article PIII and to analyze results and present a way for knowledge discovery.

**Main results** We extended the feature selection algorithm presented in article PIII with a new rule. It was then applied to molecular dynamics-based datasets. The selected features for all 12 datasets were then presented and analyzed with the aid of domain knowledge. The outcome of the analysis is that the combination of molecular dynamics simulation data, distance-based machine learning, descriptor, and feature selection can infer knowledge from first principles simulations. In addition, a way to present the chemical information was shown and used.

**Research contributions** A previously proposed feature selection algorithm was extended and applied to molecular dynamics simulation data. In doing so, we extended the comparative assessment of the MAS-based feature ranking method. We provided a proof-of-concept on how the developed feature scoring and ranking method can be used for knowledge discovery.

**Author's contributions** The author of this thesis contributed to the practical research work, main result analysis, presentation, selection of datasets, development of the extension to the feature selection algorithm, development of the interaction relevance metric, programming of the scripts used to run the tests, and main writing of the article. The author was not the only writer in the article.

## 4.5 Summary of the research contributions

This thesis studied the viability of distance-based methods in the context of computational nanoscience. In article PI, the viability of using MLM and EMLM as surrogate models was proven by comparing them to DFT calculations. In article PII, the performance of MLM was studied and compared to two FNN implementations, showing that in the studied case, MLM proves itself. The scalability of MLM was also studied through the use of the created scaling-based set of datasets. The results there correspond with previous findings [115, 52, 86, 63] that MLM does not seem to overlearn. Article PII also showed that MLM is resistant to the effects of unnecessary features. In article PIII, the use of EMLM as the ML method in feature selection was studied. The distance-based one-shot wrapper was proposed and compared to other feature selection methods. Additionally, a set of purposefully designed synthetic datasets were proposed and used in the study of the proposed feature selection algorithm. The function of the Mean Absolute Sensitivity based one-shot wrapper was studied by switching the EMLM at its core to a feedforward

neural network. The results show that EMLM and the one-shot wrapper hold their ground and that they are viable in feature selection. The study was also contextualized to feature selection literature through an umbrella review. The feature selection saw a continuation in article PIV, as the developed one-shot wrapper was extended to Feature Importance Detector (FID) and exclusively used to study and analyze the information that can be gleaned from molecular dynamics simulation datasets. The summary of the research contributions is presented in Table 1 along with the articles' relation to the research questions (given in Section 1.2).

TABLE 1 Summary of the research contributions

<b>P</b>	<b>RQ</b>	<b>Research contributions</b>
PI	RQ2a	<i>i</i> ) Showed that distance-based ML models work as DFT surrogates.
PII	RQ1a, RQ2a	<i>i</i> ) Applied MLM to molecular dynamics simulation based scaling datasets, <i>ii</i> ) studied the effects of scaling on the function of MLM.
PIII	RQ1a, RQ1b	<i>i</i> ) Used datasets of various sizes while comparing to other methods, <i>ii</i> ) developed a feature selection algorithm, <i>iii</i> ) developed a set of synthetic datasets for use in feature selection benchmarking and development.,
PIV	RQ1a, RQ1b, RQ2b	<i>i</i> ) Used datasets with increasingly large numbers of observations with 1100-5400 features, <i>ii</i> ) developed an extension to previously developed feature selection algorithm, <i>iii</i> ) extensively analyzed feature selection results, <i>iv</i> ) showed proof-of-concept for knowledge extraction from simulation data.

## 5 CONCLUSIONS AND DISCUSSION

In this thesis, the topic of hybrid nanoparticles through the use of machine learning was approached from the machine learning side. Specifically with the idea of focusing on tools and improvements to methods. The first question to ask is: **What was before?**

Before this thesis, in this context, there were three fields of science, one of nanoscience, one of machine learning, and one of computational nanoscience through the use of machine learning. Each of the three was taken a look at, and each three fields are constantly advancing forward. In addition, two two-part research questions needed an answer. The second question to ask is: **What is now?**

This thesis included four articles, article PI more from the side of nanoscience and the other three from the machine learning side. In article PI, the viability of distance-based ML models MLM and EMLM as DFT surrogates was studied. A Monte Carlo-type potential energy surface exploration strategy was used and found effective. Finally, the results were compared to DFT ground truth. The results of article PI show that MLM and EMLM are competitive alternatives to DFT. Additionally, they show that the Monte Carlo-based search strategy is effective in combination with ML surrogate models. The results also show that the ML surrogates produce results that match the DFT ground truth. The article gives the primary answer to **RQ2a**. DFT is important in nanoscience, and for a good reason. But improvements can always be made. Article PI picks the path of surrogate models in search of said improvement and finds a foothold. The reliance on DFT is still there, as the training set for a ML method has to come from somewhere. Perhaps later on, DFT will be generally used together with a ML method, where DFT lays the groundwork, and the ML method handles the interpolation.

In article PII, the possibility of utilizing a randomized solver in the context of distance-based machine learning was studied. In addition, a comparison was made between a deep and a shallow feedforward neural network and Minimal Learning Machine. Finally, a group of datasets based on the molecular simulation data of MPC Au<sub>38</sub> was created. The randomized solvers were deemed to be able to achieve speedups compared to direct solvers in the context of MLM. The MLM was also found able to hold its own against the two FNN models.



The results of article PII show that a distance-based model does not necessarily need full accuracy from each of its components in order to function and that a researcher should spend time thinking on which of the available machine learning methods are best suited to the task at hand. The article PII primarily answers **RQ1a** and gives a partial answer to **RQ2a**. The results of article PII open a discussion on how exact does a component of a ML method need be to still produce desired results. An example of a probabilistic neural network [121] was presented in Section 3.1. The point being in that in the paper, the researchers managed to make an autonomous driving ML model which relies upon probabilistic components to work [121]. Other applications in a similar manner could follow, where the ML model relies on, for example, a probabilistic component.

In article PIII, the Mean Absolute Sensitivity and EMLM-based feature selection algorithm were proposed and compared to other feature selection algorithms as well as a FNN-version of itself. An extensive umbrella review on the topic of feature selection was made, taking part in ensuring that the observed lack of discussion regarding feature selection in regression tasks was brought to light. A group of synthetic datasets were proposed, intended for the development and benchmarking of feature selection algorithms.

The results of article PIII show that EMLM comes out on top against FNNs as the machine learning model within the Mean Absolute Sensitivity ranking algorithm. The article also shows that in the field of feature selection, datasets which have known answers are helpful when performing a sanity check on a feature selection algorithm in development. The need to do the extensive umbrella review stemmed from the literature itself, pointing out that the field of feature selection is not looking in the direction of distance-based regression. Article PIII held many components, partially answering **RQ1a** and giving the first half of the answer for **RQ1b**. For a developer working on creating a feature selection algorithm, article PIII presents the utility of synthetic datasets where the correct features are known. The knowledge of correct features gives easily processed feedback on how well the development of an algorithm is going. Synthetic datasets also allow one to have a dataset of any size should one be desired. The MAS-based one-shot wrapper proved capable. Considering how EMLM is at its core and how EMLM shares properties with MLM, it would be interesting to see some of the results recomputed when the calculations are done on a GPU instead of a CPU.

In the article PIV, the feature selection algorithm from article PIII was further developed and used exclusively on molecular dynamics simulation data of a group of organic molecules as well as the MPC Au<sub>38</sub>. The article focuses on analyzing the descriptor formed by Many-body Tensor Representation and what kind of information may be gained by such analysis.

The results of article PIV show that chemical information can be extracted from the data, even though the chemists already knew the answers in the case of the used datasets. It also gives a proof-of-concept for knowledge discovery from first principles simulation data. The article also proposes the extension to the one-shot wrapper proposed in article PIII, in the form of FID. The function of FID was verified utilizing the aid of chemical domain knowledge. Finally, an interaction



relevance metric was proposed, with potential applications in determining the priority of parametrizations and others. The dataset sizes in article PIV finalize the answer to **RQ1a**, give the second half of the answer for **RQ1b**, and answer **RQ2b** in full. The knowledge discovery proof-of-concept shown in article PIV may potentially see use in further computational nanoscience applications.

The outcome of this thesis is then the study of the properties of solvers in the context of distance-based machine learning, two feature selection algorithms (with one being a continuation from the other), two groups of datasets, a way to use feature selection to infer chemical knowledge and an extensive umbrella review on feature selection. The final question to ask is: **What will be in the future?**

Under no circumstances should one think that this thesis was expected to rock the boat, so to say. And that is okay, for that is how the vast majority of scientific research is nowadays concluded [150]. However, all that is asked is that something is provided, which others may then build upon. And that is provided here. The two groups of datasets may help someone to develop new algorithms, and the feature selection algorithms may help someone in their task. The analysis of the effect of solvers may lead to machine learning methods which might even rely upon the increased randomness in their working. The provided umbrella review may give someone the context they need to move forward. The work here has mostly been with a certain set of tools while mainly focusing on the positions of the atoms.

A clear possible path for the future is, for instance, in the study of other descriptors, local and global, as well as taking more of the atomic properties into account. Local structural descriptors have been shown to, in some cases, provide a better global descriptor through local descriptors than global descriptors [91]. So naturally, a part of future work is to study local descriptors as they are and in combination with global descriptors. In the same vein, the knowledge discovery approach from article PIV could be applied to other MPCs and heterogeneous nanostructures in conjunction with utilizing other types of descriptors. It would be interesting to see if the knowledge discovery approach manages to unearth something previously unthought of. In this work and the included articles, the direction of a descriptor has been from a 3D structure to a ML input vector. In order to fulfill the goal of automatically locating new MPCs and other nanostructures, work has to be made for the reversed direction: from a descriptor to a 3D structure. GPU computation has been in the news in recent years (see Section 3.1). It is quite easy to put the application of GPU computation to the todo-list. Additionally, the ML model part could be rethought, and the model could be split into multiple parts [87]. The feature selection could also be applied to the first phase of MLM in order to see if the feature selection could be applied mid-training. The MAS-based approach to feature selection showed promise in regression tasks and should also be applied to classification tasks. The distance-based methods should also be applied to different types of datasets with different types of dissimilarity measures, for example, to text data with edit distance [160] or graphs and the related distance measures.

The research work itself is not done. As it rarely is. The topic of simulated

MPCs using machine learning is not solved. It may someday be. It is difficult to estimate where the fields will go. Nanoparticles have such vast potential that keeping track of the new applications and discoveries would be a topic of its own. Machine learning has, in recent years, taken rapid steps, and the waves caused by said steps are still rocking the boat.

## YHTEENVETO (SUMMARY IN FINNISH)

### Nanomateriaalien suunnittelun edistäminen käyttäen uusia koneoppimismenetelmiä

Tässä väitöskirjassa tutkittiin koneoppimisen keinojen käyttämistä nanotieteen tutkimuksen edistämiseksi, keskittyen erityisesti hybridinanoklustereihin. Väitöskirja on kaksijakoinen yhdistelmä nanotiedettä ja koneoppimista. Aihetta pohjustetaan käsittelemällä molemmat aiheet erikseen, sekä tuomalla aiheet lähemmäs toisiaan kummankin näkökulmasta. Nanotieteessä tutkitaan nanoskaalan tapahtumia ja asioita. Tieteenalan kehitys on ollut nopeaa ja nykyään on helpompi nimetä elämän osa-alueita, joihin nanotiede ja sen tuotokset eivät ole tavalla tai toisella koskeneet. Koneoppimisessa tutkitaan osittain automaattisia, opetettavissa olevia työkaluja. Koneoppiminen on datankäsittelykykynsä ja jatkuvasti uusien sovellustapojensa ansiosta noussut tärkeäksi osaksi teknologista ja tieteellistä kehitystä. Yhdessä nämä kaksi tieteenalaa muodostavat koneoppimis pohjaisen laskennallisen nanotieteen. Väitöskirjan sisällössä tämä tarkoittaa tiheysfunktionaaliteorian (DFT) korvaamista etäisyyspohjaisilla koneoppimismenetelmillä sekä menetelmien käyttämisen tietämyksen tuottamisessa. DFT on fysikaalisesti tarkka laskin, mutta samalla laskennallisesti hyvin vaativa. Laskennallinen nanotieteen tutkimus DFT:tä käyttäen on siten myös laskennallisesti vaativaa. Koneoppimisella tuotettu sijaismalli kykenee poistamaan suuren osan DFT:n laskentavaatimuksesta, jolloin laskentojen vaativuustasokin helpottuu.

Väitöskirjan tavoitteena on vastata seuraaviin tutkimuskysymyksiin:

- RQ1** Miten etäisyyspohjaisia koneoppimismenetelmiä voi parantaa keskittyen **a)** menetelmien skaalautuvuuteen ja **b)** muuttujanvalintaan
- RQ2** Miten etäisyyspohjaisia koneoppimismenetelmiä voidaan hyödyntää nanotieteen sovelluksissa **a)** sijaismalleina ja **b)** tietämyksen tuottamiseen

Tutkimuskysymyksiin vastataan väitöskirjaan liitetyn neljän artikkelin avulla. Ensimmäisessä artikkelissa käsiteltiin etäisyyspohjaisten koneoppimismenetelmien soveltuvuutta ja toimivuutta hybridinanopartikkelien potentiaalienergian ennustamisessa. Toisessa artikkelissa käsiteltiin etäisyyspohjaisten menetelmien toiminnan parantamista, sekä suoritettiin skaalautuvuudesta käyttäen kultaytimeistä ja suojaavasta ligandikerroksesta koostuvan  $Au_{38}(SCH_3)_{24}$  klusterin simulaatiodatasta muodostettua skaalautuvuusaineistojoukkoa. Kolmannessa artikkelissa kehitettiin etäisyyspohjainen muuttujanvalintamenetelmä sekä pohjustettiin työlaajalla koostartikkelien koosteen avulla. Neljännessä artikkelissa laajennettiin kolmosartikkelissa kehitettyä muuttujanvalintamenetelmää ja hyödynnettiin sitä tietämyksen löytämisessä  $Au_{38}(SCH_3)_{24}$  klusterin kanssa sekä 11:sta muun molekyyli simulaatiodatsetin kanssa.

## REFERENCES

- [1] I. W. Advertising. Five benefits of machine learning in advertising, 2021. URL <https://www.ibm.com/watson-advertising/thought-leadership/benefits-of-machine-learning-in-advertising>. Referenced 16.1.2023.
- [2] R. Agrawal, B. Kaur, and S. Sharma. Quantum based Whale Optimization Algorithm for wrapper feature selection. *Applied Soft Computing*, 89:106092, 2020. .
- [3] E. Alpaydin. *Introduction to machine learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts ; London, England, third edition edition, 2014. URL <https://ebookcentral.proquest.com/lib/jyvaskyla-ebooks/detail.action?docID=3339851>. Includes index.
- [4] E. F. Amankwaa and K. B. Blay. Cities at risk? Exploring the synergies between smartphones and everyday vulnerabilities. *Cities*, 83:129–139, 2018. URL <https://www.sciencedirect.com/science/article/pii/S0264275117307874>.
- [5] Y. Amit and D. Geman. Shape Quantization and Recognition with Randomized Trees. *Neural Computation*, 9(7):1545–1588, 1997. .
- [6] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixão, F. Mutz, L. de Paula Veronese, T. Oliveira-Santos, and A. F. De Souza. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021. . URL <https://www.sciencedirect.com/science/article/pii/S095741742030628X>.
- [7] A. G. Barnston. Correspondence among the Correlation, RMSE, and Heidke Forecast Verification Measures; Refinement of the Heidke Score. *Weather and Forecasting*, 7(4):699 – 709, 1992. .
- [8] A. P. Bartók, R. Kondor, and G. Csányi. On representing chemical environments. *Phys. Rev. B*, 87:184115, 2013. .
- [9] S. Bayda, M. Adeel, T. Tuccinardi, M. Cordani, and F. Rizzolio. The History of Nanoscience and Nanotechnology: From Chemical–Physical Applications to Nanomedicine. *Molecules*, 25(1), 2020. . URL <https://www.mdpi.com/1420-3049/25/1/112>.
- [10] BBC. Google’s AI wins final Go challenge, 2016. URL <https://www.bbc.com/news/technology-35810133>.
- [11] J. Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *The Journal of Chemical Physics*, 134(7):074106, 2011. .

- [12] G. Binnig, C. F. Quate, and C. Gerber. Atomic Force Microscope. *Phys. Rev. Lett.*, 56:930–933, 1986. .
- [13] G. Binnig and H. Rohrer. Scanning Tunneling Microscopy. *Surface Science*, 126(1):236–244, 1983. . URL <https://www.sciencedirect.com/science/article/pii/0039602883907161>.
- [14] V. Bolón-Canedo, N. Sánchez-Marroño, and A. Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, 34(3):483–519, 2013. .
- [15] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92*, 144–152. Association for Computing Machinery, New York, NY, USA, 1992. . URL <https://doi.org/10.1145/130385.130401>.
- [16] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. .
- [17] K. A. Brown, S. Brittman, N. Maccaferri, D. Jariwala, and U. Celano. Machine Learning in Nanoscience: Big Data at Small Scales. *Nano Letters*, 20(1):2–10, 2020. .
- [18] X. Carbonell, A. Chamarro, U. Oberst, B. Rodrigo, and M. Prades. Problematic Use of the Internet and Smartphones in University Students: 2006–2017. *International Journal of Environmental Research and Public Health*, 15(3), 2018. . URL <https://www.mdpi.com/1660-4601/15/3/475>.
- [19] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová. Machine learning and the physical sciences. *Rev. Mod. Phys.*, 91:045002, 2019. .
- [20] R. Caruana and A. Niculescu-Mizil. Data Mining in Metric Space: An Empirical Analysis of Supervised Learning Performance Criteria. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04*, 69–78. Association for Computing Machinery, New York, NY, USA, 2004. .
- [21] C. Cavnar. Is machine learning a subset of AI?, 2022. URL <https://capacity.com/intelligent-document-processing/faqs/is-machine-learning-a-subset-of-ai/>. Referenced 15.1.2023.
- [22] T. Chai and R. R. Draxler. Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, 7(3):1247–1250, 2014. . URL <https://gmd.copernicus.org/articles/7/1247/2014/>.
- [23] N. Chandross. Deep Blue Earns Celebrity Rating, 2000. URL <https://abcnews.go.com/Entertainment/story?id=116123&page=1>. Referenced 13.1.2023.

- [24] B. Charbuty and A. Abdulazeez. Classification Based on Decision Tree Algorithm for Machine Learning. *Journal of Applied Science and Technology Trends*, 2(01):20 – 28, 2021. . URL <https://jastt.org/index.php/jasttpath/article/view/65>.
- [25] S. Chaturvedi, P. N. Dave, and N. Shah. Applications of nano-catalyst in new era. *Journal of Saudi Chemical Society*, 16(3):307–325, 2012. .
- [26] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS 2016*, 7–10. Association for Computing Machinery, New York, NY, USA, 2016. .
- [27] L. Cheng, Y. Wang, X. Liu, and B. Li. Outlier Detection Ensemble with Embedded Feature Selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3503–3512, 2020. . URL <https://ojs.aaai.org/index.php/AAAI/article/view/5755>.
- [28] Chessgames.com. Garry Kasparov vs. Deep Blue, 12 games, 1996-1997. URL <https://www.chessgames.com/perl/chess.pl?pid=29912&pid2=15940>. Referenced 13.1.2023.
- [29] S. Chmiela, H. E. Sauceda, K.-R. Müller, and A. Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature Communications*, 9(1):3887, 2018. .
- [30] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*, 3(5):e1603015, 2017. .
- [31] J.-A. Choi and K. Lim. Identifying machine learning techniques for classification of target advertising. *ICT Express*, 6(3):175–180, 2020. . URL <https://www.sciencedirect.com/science/article/pii/S2405959520301090>.
- [32] K. Choudhary, B. DeCost, and F. Tavazza. Machine learning with force-field-inspired descriptors for materials: Fast screening and mapping energy landscape. *Phys. Rev. Mater.*, 2:083801, 2018. .
- [33] E. Commission, J. R. Centre, S. Samoili, M. López Cobo, E. Gómez, G. De Prato, F. Martínez-Plumed, and B. Delipetrev. *AI watch : defining artificial intelligence : towards an operational definition and taxonomy of artificial intelligence*. Number KJ-NA-30117-EN-N (online) in JRC Technical Reports. Publications Office of the European Union, Luxembourg (Luxembourg), 2020. .
- [34] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. . URL <https://doi.org/10.1007/BF00994018>.



- [35] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. .
- [36] R. Das, R. Nath, A. K. Shukla, and P. K. Muhuri. Multi-objective Optimization Based Feature Selection Using Correlation. In W. Chen, L. Yao, T. Cai, S. Pan, T. Shen, and X. Li, editors, *Advanced Data Mining and Applications*, 325–336. Springer Nature Switzerland, Cham, 2022.
- [37] A. H. de Souza, F. Corona, G. A. Barreto, Y. Miche, and A. Lendasse. Minimal Learning Machine: A novel supervised distance-based approach for regression and classification. *Neurocomputing*, 164:34–44, 2015. . URL <https://www.sciencedirect.com/science/article/pii/S0925231215003021>.
- [38] A. H. de Souza Junior, F. Corona, Y. Miche, A. Lendasse, G. A. Barreto, and O. Simula. Minimal Learning Machine: A New Distance-Based Method for Supervised Learning. In I. Rojas, G. Joya, and J. Gabestany, editors, *Advances in Computational Intelligence*, 408–416. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [39] M. Delkowski, J. V. Anguita, C. T. G. Smith, and S. R. P. Silva. Multifunctional Nanostructures with Controllable Band Gap Giving Highly Stable Infrared Emissivity for Smart Thermal Management. *ACS Nano*, 17(2):1335–1343, 2023. .
- [40] R. Diao and Q. Shen. Nature inspired feature selection meta-heuristics. *Artificial Intelligence Review*, 44(3):311–340, 2015. .
- [41] G. S. Doerk, A. Stein, S. Bae, M. M. Noack, M. Fukuto, and K. G. Yager. Autonomous discovery of emergent morphologies in directed self-assembly of block copolymer blends. *Science Advances*, 9(2):eadd3687, 2023. .
- [42] G. Doquire and M. Verleysen. Feature selection for multi-label classification problems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6691 LNCS(PART 1):9 – 16, 2011. . URL [https://www.scopus.com/inward/record.uri?eid=2-s2.0-79957967497&doi=10.1007%2f978-3-642-21501-8\\_2&partnerID=40&md5=f7aed29e42b38bfa359fbd040a607ba3](https://www.scopus.com/inward/record.uri?eid=2-s2.0-79957967497&doi=10.1007%2f978-3-642-21501-8_2&partnerID=40&md5=f7aed29e42b38bfa359fbd040a607ba3). Cited by: 76.
- [43] E. K. Drexler, C. Peterson, and G. Pergamit. *Unbounding the Future: The Nanotechnology Revolution; William Morrow and Company*. William Morrow and Company, Inc., 1991.
- [44] K. E. Drexler. *Engines of creation: the coming era of nanotechnology*. Anchor Books, Garden City, New York, 1986.
- [45] T. E. Duncan. On the Calculation of Mutual Information. *SIAM Journal on Applied Mathematics*, 19(1):215–220, 1970. .



- [46] J. Enkovaara, N. A. Romero, S. Shende, and J. J. Mortensen. GPAW - massively parallel electronic structure calculations with Python-based software. *Procedia Computer Science*, 4:17–25, 2011. . URL <https://www.sciencedirect.com/science/article/pii/S1877050911000615>. Proceedings of the International Conference on Computational Science, ICCS 2011.
- [47] F. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento. Crystal structure representations for machine learning models of formation energies. *International Journal of Quantum Chemistry*, 115(16):1094–1101, 2015. .
- [48] K. Fawagreh, M. M. Gaber, and E. Elyan. Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1):602–609, 2014. .
- [49] B. Fetsje, J. Marianne, and V. Aad van der. *An Introduction to Mathematical Statistics*. Amsterdam University Press, 2017. URL <https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=1593127&site=ehost-live>.
- [50] R. P. Feynman. There’s plenty of room at the bottom [data storage]. *Journal of Microelectromechanical Systems*, 1(1):60–66, 1992. .
- [51] P. Flach. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*, chapter 8. Distance-based models, 231–261. Cambridge University Press, Cambridge, 2012. . URL <https://www.cambridge.org/core/books/machine-learning/distancebased-models/1C8BC8913F732E8EDF2DBE37EE396A28>.
- [52] J. A. Florêncio, S. A. Oliveira, J. P. Gomes, and A. R. R. Neto. A new perspective for Minimal Learning Machines: A lightweight approach. *Neurocomputing*, 401:308–319, 2020. .
- [53] D. B. Fogel. *Evolutionary Computation: The Fossil Record*. Wiley-IEEE Press, 1998. URL <https://ieeexplore.ieee.org/servlet/opac?bknumber=5263042>.
- [54] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In L. Saitta, editor, *ICML’96: Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.
- [55] D. Fu, R. P. Narayanan, A. Prasad, F. Zhang, D. Williams, J. S. Schreck, H. Yan, and J. Reif. Automated design of 3D DNA origami with non-rasterized 2D curvature. *Science Advances*, 8(51):eade4455, 2022. .
- [56] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980. .

- [57] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection.net: A new change detection benchmark dataset. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 1–8. 2012. .
- [58] I. Guyon, C. Aliferis, G. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A. Statnikov. Design and Analysis of the Causation and Prediction Challenge. In I. Guyon, C. Aliferis, G. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A. Statnikov, editors, *Proceedings of the Workshop on the Causation and Prediction Challenge at WCCI 2008*, volume 3 of *Proceedings of Machine Learning Research*, 1–33. PMLR, Hong Kong, 2008. URL <http://proceedings.mlr.press/v3/guyon08a.html>.
- [59] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003. URL <https://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf?ref=driverlayer.com/web>.
- [60] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction, Foundations and Applications*. Studies in Fuzzyness and Soft Computing. Springer, 2006.
- [61] T. Haigh, M. Priestley, and C. Rope. Los Alamos Bets on ENIAC: Nuclear Monte Carlo Simulations, 1947-1948. *IEEE Annals of the History of Computing*, 36:42–63, 2014.
- [62] K. Hakeem, M. Kamli, J. Sabir, and H. Alharby. *Diverse Applications of Nanotechnology in the Biological Sciences: An Essential Tool in Agri-Business and Health Care Systems*. Apple Academic Press, New York, USA, 1 edition, 2022. .
- [63] J. Hämmäläinen and T. Kärkkäinen. *Newton Method for Minimal Learning Machine*, 97–108. Springer International Publishing, Cham, 2022. .
- [64] T. Hamed, R. Dara, and S. C. Kremer. An Accurate, Fast Embedded Feature Selection for SVMs. In *2014 13th International Conference on Machine Learning and Applications*, 135–140. 2014. .
- [65] D. A. H. Hanaor, M. H. N. Assadi, S. Li, A. Yu, and C. C. Sorrell. Ab initio study of phase stability in doped TiO<sub>2</sub>. *Computational Mechanics*, 50(2):185–194, 2012. .
- [66] G. S. Handelman, H. K. Kok, R. V. Chandra, A. H. Razavi, S. Huang, M. Brooks, M. J. Lee, and H. Asadi. Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods. *American Journal of Roentgenology*, 212(1):38–43, 2019. . PMID: 30332290.

- [67] E. Harmon and M. Mazmanian. Stories of the Smartphone in Everyday Discourse: Conflict, Tension & Instability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '13*, 1051–1060. Association for Computing Machinery, New York, NY, USA, 2013. .
- [68] R. Hasani, M. Lechner, A. Amini, L. Liebenwein, A. Ray, M. Tschaikowski, G. Teschl, and D. Rus. Closed-form continuous-time neural networks. *Nature Machine Intelligence*, 4(11):992–1003, 2022. .
- [69] R. Hasani, M. Lechner, A. Amini, D. Rus, and R. Grosu. Liquid Time-constant Networks, 2020. .
- [70] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998. .
- [71] M. Herrera-Barrera, R. C. Ryals, M. Gautam, A. Jozic, M. Landry, T. Korzun, M. Gupta, C. Acosta, J. Stoddard, R. Reynaga, W. Tschetter, N. Jacomino, O. Taratula, C. Sun, A. K. Lauer, M. Neuringer, and G. Sahay. Peptide-guided lipid nanoparticles deliver mRNA to the neural retina of rodents and nonhuman primates. *Science Advances*, 9(2):eadd4623, 2023. .
- [72] G. Hills, C. Lau, A. Wright, S. Fuller, M. D. Bishop, T. Srimani, P. Kanhaiya, R. Ho, A. Amer, Y. Stein, D. Murphy, Arvind, A. Chandrakasan, and M. M. Shulaker. Modern microprocessor built from complementary carbon nanotube transistors. *Nature*, 572(7771):595–602, 2019. .
- [73] L. Himanen, M. O. Jäger, E. V. Morooka, F. Federici Canova, Y. S. Ranawat, D. Z. Gao, P. Rinke, and A. S. Foster. DDescribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020. . URL <https://www.sciencedirect.com/science/article/pii/S0010465519303042>.
- [74] T. K. Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, 278–282 vol.1. 1995. .
- [75] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. .
- [76] T. O. Hodson. Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not. *Geoscientific Model Development*, 15(14):5481–5487, 2022. . URL <https://gmd.copernicus.org/articles/15/5481/2022/>.
- [77] A. E. Hoerl and R. W. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67, 1970. .
- [78] R. W. Hoerl. Ridge Analysis 25 Years Later. *The American Statistician*, 39(3):186–192, 1985. .

- [79] P. Hohenberg and W. Kohn. Inhomogeneous Electron Gas. *Phys. Rev.*, 136:B864–B871, 1964. .
- [80] J. H. Holland. Genetic Algorithms. *Scientific American*, 267(1):66–73, 1992. URL <http://www.jstor.org/stable/24939139>. Full publication date: JULY 1992.
- [81] D. Hooi Ting, S. Fong Lim, T. Siuly Patanmacia, C. Gie Low, and G. Chuan Ker. Dependency on smartphone and the impact on purchase behaviour. *Young Consumers*, 12(3):193–203, 2011. .
- [82] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. .
- [83] M. Hossin and M. Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2), 2015. .
- [84] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu. Hybrid feature selection by combining filters and wrappers. *Expert Systems with Applications*, 38(7):8144–8150, 2011. . URL <https://www.sciencedirect.com/science/article/pii/S0957417410015198>.
- [85] H. Huo and M. Rupp. Unified representation of molecules and crystals for machine learning. *Machine Learning: Science and Technology*, 3(4):045017, 2022. .
- [86] J. Hämmäläinen, A. S. C. Alencar, T. Kärkkäinen, C. L. C. Mattos, A. H. S. Júnior, and J. a. P. P. Gomes. Minimal Learning Machine: Theoretical Results and Clustering-Based Reference Point Selection. *J. Mach. Learn. Res.*, 21(1), 2022.
- [87] J. Hämmäläinen and T. Kärkkäinen. Instance-Based Multi-Label Classification via Multi-Target Distance Regression. In *ESANN 2021 : Proceedings of the 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning Online event (Bruges, Belgium), October 06 - 08*, 653–658. ESANN, 2021. .
- [88] S. Iijima. Helical microtubules of graphitic carbon. *Nature*, 354(6348):56–58, 1991. .
- [89] O. Isayev, C. Oses, C. Toher, E. Gossett, S. Curtarolo, and A. Tropsha. Universal fragment descriptors for predicting properties of inorganic crystals. *Nature Communications*, 8(1):15679, 2017. .
- [90] A. G. Ivakhnenko and V. G. Lapa. First working Deep Learners with many layers, learning internal representations. Technical report, CCM Information Corporation, 1965.

- [91] M. O. J. Jäger, E. V. Morooka, F. Federici Canova, L. Himanen, and A. S. Foster. Machine learning hydrogen adsorption on nanoclusters through structural descriptors. *npj Computational Materials*, 4(1):37, 2018. .
- [92] J. Jahanmir, B. Hagggar, and J. Hayes. The scanning probe microscope. *Scanning microscopy*, 6(3):625–660, 1992.
- [93] J. E. Jones and S. Chapman. On the determination of molecular fields.—II. From the equation of state of a gas. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 106(738):463–477, 1924. .
- [94] J. E. Jones and S. Chapman. On the determination of molecular fields.—I. From the variation of the viscosity of a gas with temperature. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 106(738):441–462, 1924. .
- [95] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015. .
- [96] S. Jordan, Y. Chandak, D. Cohen, M. Zhang, and P. Thomas. Evaluating the Performance of Reinforcement Learning Algorithms. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 4962–4973. PMLR, 2020. URL <https://proceedings.mlr.press/v119/jordan20a.html>.
- [97] R. Juarez-Mosqueda, S. Malola, and H. Häkkinen. Ab initio molecular dynamics studies of Au<sub>38</sub>(SR)<sub>24</sub> isomers under heating. *The European Physical Journal D*, 73(3):62, 2019. .
- [98] D. S. K. Karunasingha. Root mean square error or mean absolute error? Use their ratio as well. *Information Sciences*, 585:609–629, 2022. . URL <https://www.sciencedirect.com/science/article/pii/S0020025521011567>.
- [99] S. Kashef, H. Nezamabadi-pour, and B. Nikpour. Multilabel feature selection: A comprehensive review and guiding experiments. *WIREs Data Mining and Knowledge Discovery*, 8(2):e1240, 2018. .
- [100] H. Katayama-Yoshida, K. Sato, T. Fukushima, M. Toyoda, H. Kizaki, V. Dinh, and P. Dederichs. Computational nano-materials design for high-TC ferromagnetism in wide-gap magnetic semiconductors. *Journal of Magnetism and Magnetic Materials*, 310(2, Part 3):2070–2077, 2007. . URL <https://www.sciencedirect.com/science/article/pii/S0304885306022372>. Proceedings of the 17th International Conference on Magnetism.
- [101] E. Kavlakoglu. AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What’s the Difference?, 2020. URL <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>. Referenced 15.1.2023.



- [102] S. Kemp. Digital 2021: Global Overview Report, 2021. URL <https://datareportal.com/reports/digital-2021-global-overview-report>. Referenced 13.1.2023.
- [103] J. H. Kim, J. K. Kim, J. Liu, A. Curcio, J.-S. Jang, I.-D. Kim, F. Ciucci, and W. Jung. Nanoparticle Ex-solution for Supported Catalysts: Materials Design, Mechanism and Future Perspectives. *ACS Nano*, 15(1):81–110, 2021.
- [104] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, 2014.
- [105] C. Kinnear, T. L. Moore, L. Rodriguez-Lorenzo, B. Rothen-Rutishauser, and A. Petri-Fink. Form Follows Function: Nanoparticle Shape and Its Implications for Nanomedicine. *Chemical Reviews*, 117(17):11476–11521, 2017.
- [106] R. Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'95*, 1137–1143. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995.
- [107] R. Kohavi and G. H. John. The Wrapper Approach. In H. Liu and H. Motoda, editors, *Feature Extraction, Construction and Selection: A Data Mining Perspective*, 33–50. Springer US, Boston, MA, 1998.
- [108] W. Kohn and L. J. Sham. Self-Consistent Equations Including Exchange and Correlation Effects. *Phys. Rev.*, 140:A1133–A1138, 1965.
- [109] J. Kramár, T. Eccles, I. Gemp, A. Tacchetti, K. R. McKee, M. Malinowski, T. Graepel, and Y. Bachrach. Negotiation and honesty in artificial intelligence methods for the board game of Diplomacy. *Nature Communications*, 13(1):7214, 2022.
- [110] D. P. Kroese and R. Y. Rubinstein. Monte Carlo methods. *WIREs Computational Statistics*, 4(1):48–58, 2012.
- [111] H. W. Kroto, J. R. Heath, S. C. O'Brien, R. F. Curl, and R. E. Smalley. C60: Buckminsterfullerene. *Nature*, 318(6042):162–163, 1985.
- [112] V. Kumar and S. Minz. Feature Selection: A literature Review. *Smart Comput. Rev.*, 4(3):211–229, 2014.
- [113] V. Kunwar Singh. *Computational Science*. Arcler Press, Ashland, 2020. URL <https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=2725236&site=ehost-live>.

- [114] T. Kärkkäinen. Extreme Minimal Learning Machine. In *ESANN 2018 : Proceedings of the 26th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 237–242. ESANN, 2018. URL <https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2018-72.pdf>.
- [115] T. Kärkkäinen. Extreme minimal learning machine: Ridge regression with distance-based basis. *Neurocomputing*, 342:33–48, 2019. . URL <https://www.sciencedirect.com/science/article/pii/S0925231219301432>. Advances in artificial neural networks, machine learning and computational intelligence.
- [116] T. Kärkkäinen and H. Häkkinen. Structure Prediction of Hybrid Nanoparticles Via Artificial Intelligence, 2018. URL <https://www.jyu.fi/it/en/research/research-projects/academy-of-finland/hybridipartikkelit>. Referenced 17.2.2023.
- [117] T. Kärkkäinen, H. Häkkinen, and K. Honkala. High performing machine learning for novel catalyst design (MLNovCat), 2022. URL <https://converis.jyu.fi/converis/portal/detail/Project/102431943>. Referenced 12.2.2023.
- [118] S. Lab. List of Journals with the subject "Nanoscience and Nanotechnology", 2022. URL [https://www.scimagojr.com/journalrank.php?category=2509&page=1&total\\_size=79](https://www.scimagojr.com/journalrank.php?category=2509&page=1&total_size=79).
- [119] A. Lavin and S. Ahmad. Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 38–44. 2015. .
- [120] C. Le Lan, M. G. Bellemare, and P. S. Castro. Metrics and Continuity in Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(9):8261–8269, 2021. . URL <https://ojs.aaai.org/index.php/AAAI/article/view/17005>.
- [121] M. Lechner, R. Hasani, A. Amini, T. A. Henzinger, D. Rus, and R. Grosu. Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, 2(10):642–652, 2020. .
- [122] P. Y. Lee, W. P. Loh, and J. F. Chin. Feature selection in multimedia: The state-of-the-art review. *Image and Vision Computing*, 67:29–42, 2017. URL <https://www.sciencedirect.com/science/article/pii/S0262885617301518>.
- [123] J. E. Lennard-Jones. Cohesion. *Proceedings of the Physical Society*, 43(5):461, 1931. .
- [124] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu. Feature Selection: A Data Perspective. *ACM Comput. Surv.*, 50(6), 2017. .
- [125] S. Linnainmaa. *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors (orig. title: Algoritmin Kumulatiivinen Pyöristysvirhe Yksittäisten Pyöristysvirheiden Taylor-*



- sarjakehitelmänä*). Master's thesis, University of Helsinki, Tapiola, 1970. URL <https://people.idsia.ch/~juergen/linnainmaa1970thesis.pdf>.
- [126] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer, Norwell, MA, 1998.
- [127] Y. Liu, G. Zhou, K. Liu, and Y. Cui. Design of Complex Nanomaterials for Energy Storage: Past Success and Future Opportunity. *Accounts of Chemical Research*, 50(12):2895–2905, 2017. .
- [128] S. Lu, D. Guo, S. Ren, J. Huang, A. Svyatkovskiy, A. Blanco, C. Clement, D. Drain, D. Jiang, D. Tang, G. Li, L. Zhou, L. Shou, L. Zhou, M. Tufano, M. Gong, M. Zhou, N. Duan, N. Sundaresan, S. K. Deng, S. Fu, and S. Liu. CodeXGLUE: A Machine Learning Benchmark Dataset for Code Understanding and Generation, 2021. .
- [129] Y. Lu, K. Huntoon, D. Lee, Y. Wang, J. Ha, Y. Qie, X. Li, B. R. Schrank, S. Dong, T. D. Gallup, M. Kang, H. Zhao, Y. An, Z. Yang, J. Li, B. Y. S. Kim, and W. Jiang. Immunological conversion of solid tumours using a bispecific nanobioconjugate for cancer immunotherapy. *Nature Nanotechnology*, 17(12):1332–1341, 2022. .
- [130] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943. .
- [131] G. C. McDonald. Ridge regression. *WIREs Computational Statistics*, 1(1):93–100, 2009. .
- [132] N. Metropolis. The Beginning of the Monte Carlo Method. *Los Alamos Science*, 15, 1987. URL <https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-88-9067>. Special issue.
- [133] Microsoft. What is machine learning?, 2023. URL <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-machine-learning-platform/>. Referenced 15.1.2023.
- [134] Midjourney. Midjourney, 2022. URL <https://www.midjourney.com/home>. Referenced 12.1.2023.
- [135] T. M. Mitchell. *Machine learning*, volume 1. McGraw-hill New York, 1997.
- [136] F. Moslehi and A. Haeri. A novel hybrid wrapper–filter approach based on genetic algorithm, particle swarm optimization for feature subset selection. *Journal of Ambient Intelligence and Humanized Computing*, 11(3):1105–1127, 2020. .
- [137] K. Moyer-Vanderburgh, M. C. Ma, S. J. Park, M. L. Jue, S. F. Buchsbaum, K. J. Wu, M. Wood, J. Ye, and F. Fornasiero. Growth and Performance of High-Quality SWCNT Forests on Inconel Foils as Lithium-Ion Battery Anodes. *ACS Applied Materials & Interfaces*, 14(49):54981–54991, 2022. .

- [138] S. Nadis. Researchers Discover a More Flexible Approach to Machine Learning, 2023. URL <https://www.quantamagazine.org/researchers-discover-a-more-flexible-approach-to-machine-learning-20230207/#0>. Referenced 16.2.2023.
- [139] O. P. V. Neto. Intelligent Computational Nanotechnology: The Role of Computational Intelligence in the Development of Nanoscience and Nanotechnology. *Journal of Computational and Theoretical Nanoscience*, 11(4):928–944, 2014. . URL <https://www.ingentaconnect.com/content/asp/jctn/2014/00000011/00000004/art00002>.
- [140] B. H. Nguyen, B. Xue, and M. Zhang. A survey on swarm intelligence approaches to feature selection in data mining. *Swarm and Evolutionary Computation*, 54:100663, 2020. .
- [141] NobelPrize.org. Press release, 1986. URL <https://www.nobelprize.org/prizes/physics/1986/press-release/>.
- [142] T. F. F. S. of Engineering and A. S. of the Columbia University. Artificial Intelligence (AI) vs. Machine Learning, 2023. URL <https://ai.engineering.columbia.edu/ai-vs-machine-learning/>. Referenced 15.1.2023.
- [143] A. of Finland. Novel Applications of Artificial Intelligence in Physical Sciences and Engineering Research (AIPSE). URL <https://www.aka.fi/en/research-funding/programmes-and-other-funding-schemes/academy-programmes/alasivut/academy-programmes-under-evaluation/novel-applications-of-artificial-intelligence-in-physical-sciences-and-engineering-research-aipse/>. Referenced 17.2.2023.
- [144] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Fong, A. Roy-Chowdhury, and M. Desai. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR 2011*, 3153–3160. 2011. .
- [145] OpenAI. OpenAI Five. <https://blog.openai.com/openai-five/>, 2018.
- [146] OpenAI. ChatGPT, 2022. URL <https://openai.com/blog/chatgpt/>. Referenced 12.1.2023.
- [147] T. Pahikkala, S. Okser, A. Airola, T. Salakoski, and T. Aittokallio. Wrapper-based selection of genetic features in genome-wide association studies through fast matrix operations. *Algorithms for Molecular Biology*, 7(1):11, 2012. .
- [148] J.-O. Palacio-Niño and F. Berzal. Evaluation Metrics for Unsupervised Learning Algorithms, 2019. .

- [149] T. Panova, X. Carbonell, A. Chamarro, and D. X. Puerta-Cortés. Specific smartphone uses and how they relate to anxiety and depression in university students: a cross-cultural perspective. *Behaviour & Information Technology*, 39(9):944–956, 2020. . URL <https://doi.org/10.1080/0144929X.2019.1633405>.
- [150] M. Park, E. Leahey, and R. J. Funk. Papers and patents are becoming less disruptive over time. *Nature*, 613(7942):138–144, 2023. .
- [151] V. R. Pasupuleti. Chapter 40 - Nanoscience and nanotechnology advances in food industry. In R. Bhat, editor, *Future Foods*, 721–732. Academic Press, 2022. . URL <https://www.sciencedirect.com/science/article/pii/B9780323910019000116>.
- [152] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. URL [https://scikit-learn.org/stable/modules/model\\_evaluation.html#regression-metrics](https://scikit-learn.org/stable/modules/model_evaluation.html#regression-metrics).
- [153] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
- [154] R. B. Pereira, A. Plastino, B. Zadrozny, and L. H. C. Merschmann. Categorizing feature selection methods for multi-label classification. *Artificial Intelligence Review*, 49(1):57 – 78, 2018. . URL <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84988719261&doi=10.1007%2fs10462-016-9516-4&partnerID=40&md5=edd82616e88ddc0ec61f0238d199e2eb>. Cited by: 97.
- [155] J. E. Perez, B. Bajaber, N. Alsharif, A. I. Martínez-Banderas, N. Patel, A. Sharip, E. Di Fabrizio, J. Merzaban, and J. Kosel. Modulated nanowire scaffold for highly efficient differentiation of mesenchymal stem cells. *Journal of Nanobiotechnology*, 20(1):282, 2022. .
- [156] C. Perlich, B. Dalessandro, T. Raeder, O. Stitelman, and F. Provost. Machine learning for targeted display advertising: transfer learning in action. *Machine Learning*, 95(1):103–127, 2014. .
- [157] A. Pimpin and W. Srituravanich. Review on Micro- and Nanolithography Techniques and Their Applications. *Engineering Journal*, 16(1):37–56, 2011. .
- [158] H. Qian, W. T. Eckenhoff, Y. Zhu, T. Pintauer, and R. Jin. Total Structure Determination of Thiolate-Protected Au<sub>38</sub> Nanoparticles. *Journal of the American Chemical Society*, 132(24):8280–8281, 2010.

- [159] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-Shot Text-to-Image Generation. *CoRR*, abs/2102.12092, 2021. URL <https://arxiv.org/abs/2102.12092>. 2102.12092.
- [160] E. Ristad and P. Yianilos. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532, 1998. .
- [161] P. Rojas. *Neural Networks: A Systematic Introduction*. Springer Berlin / Heidelberg, 1 edition, 1996. Ebook ISBN: 9783642610684.
- [162] K. Roose. An A.I.-Generated Picture Won an Art Prize. Artists Aren't Happy., 2022. URL <https://www.nytimes.com/2022/09/02/technology/ai-artificial-intelligence-artists.html>. Referenced 12.1.2023.
- [163] E. Runge and E. K. U. Gross. Density-Functional Theory for Time-Dependent Systems. *Phys. Rev. Lett.*, 52:997–1000, 1984. .
- [164] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning. *Phys. Rev. Lett.*, 108:058301, 2012. .
- [165] S. J. Russell, M.-W. Chang, J. Devlin, A. Dragan, D. Forsyth, I. Goodfellow, J. M. Malik, V. Mansinghka, P. Norvig, J. Pearl, and M. Wooldridge. *Artificial intelligence : a modern approach*. Pearson series in artificial intelligence. Pearson, Harlow, Fourth edition. Global edition, 2022. URL <https://resolver.vitalsource.com/9781292401171>.
- [166] Z. Sadeghian, E. Akbari, and H. Nematzadeh. A hybrid feature selection method based on information theory and binary butterfly optimization algorithm. *Engineering Applications of Artificial Intelligence*, 97:104079, 2021. .
- [167] S. Salcedo-Sanz, L. Cornejo-Bueno, L. Prieto, D. Paredes, and R. García-Herrera. Feature selection in machine learning prediction systems for renewable energy applications. *Renewable and Sustainable Energy Reviews*, 90:728–741, 2018. .
- [168] R. Sami, E. Khojah, A. Elhakem, N. Benajiba, M. Helal, N. Alhuthal, S. A. Alzahrani, M. Alharbi, and M. Chavali. Performance Study of Nano/SiO<sub>2</sub> Films and the Antimicrobial Application on Cantaloupe Fruit Shelf-Life. *Applied Sciences*, 11(9), 2021. . URL <https://www.mdpi.com/2076-3417/11/9/3879>.
- [169] A. L. Samuel. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959. .
- [170] F. Santosa and W. W. Symes. Linear Inversion of Band-Limited Reflection Seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330, 1986. .

- [171] G. A. A. Saracino, D. Cigognini, D. Silva, A. Caprini, and F. Gelain. Nanomaterials design and tests for neural tissue engineering. *Chem. Soc. Rev.*, 42:225–262, 2013. .
- [172] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. . URL <https://www.sciencedirect.com/science/article/pii/S0893608014002135>.
- [173] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8(1):13890, 2017. .
- [174] L. Shabani, M. Abbasi, M. Amini, A. M. Amani, and A. Vaez. The brilliance of nanoscience over cancer therapy: Novel promising nanotechnology-based methods for eradicating glioblastoma. *Journal of the Neurological Sciences*, 440:120316, 2022. . URL <https://www.sciencedirect.com/science/article/pii/S0022510X22001782>.
- [175] N. Shah, S. Engineer, N. Bhagat, H. Chauhan, and M. Shah. Research Trends on the Usage of Machine Learning and Artificial Intelligence in Advertising. *Augmented Human Research*, 5(1):19, 2020. .
- [176] E. Sharma, R. Rathi, J. Misharwal, B. Sinhmar, S. Kumari, J. Dalal, and A. Kumar. Evolution in Lithography Techniques: Microlithography to Nanolithography. *Nanomaterials*, 12(16), 2022. . URL <https://www.mdpi.com/2079-4991/12/16/2754>.
- [177] C. Shi, D. Guo, K. Xiao, X. Wang, L. Wang, and J. Luo. A drug-specific nanocarrier design for efficient anticancer therapy. *Nature Communications*, 6(1):7449, 2015. .
- [178] M. M. Shulaker, G. Hills, N. Patil, H. Wei, H.-Y. Chen, H.-S. P. Wong, and S. Mitra. Carbon nanotube computer. *Nature*, 501(7468):526–530, 2013. .
- [179] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. .
- [180] G. Stein, B. Chen, A. S. Wu, and K. A. Hua. Decision Tree Classifier for Network Intrusion Detection with GA-Based Feature Selection. In *Proceedings of the 43rd Annual Southeast Regional Conference - Volume 2*, ACM-SE 43, 136–141. Association for Computing Machinery, New York, NY, USA, 2005. .
- [181] L. Stuart. *Introduction to Nanoscience*. OUP Oxford, 2010. URL <https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=304000&site=ehost-live>.



- [182] Y. Sun, J. Yao, and S. Goodison. Feature Selection for Nonlinear Regression and its Application to Cancer Research. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*, 73–81. Society for Industrial and Applied Mathematics, Pinnacle Vancouver Harbourfront Hotel Vancouver, British Columbia, Canada, 2015. .
- [183] S. Tian, Y.-Z. Li, M.-B. Li, J. Yuan, J. Yang, Z. Wu, and R. Jin. Structural isomerism in gold nanoparticles revealed by X-ray crystallography. *Nature Communications*, 6(1):8667, 2015. .
- [184] R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. URL <http://www.jstor.org/stable/2346178>. Full publication date: 1996.
- [185] M. Todorović, M. U. Gutmann, J. Corander, and P. Rinke. Bayesian inference of atomistic structure in functional materials. *npj Computational Materials*, 5(1):35, 2019. .
- [186] C. Toumey. Reading Feynman Into Nanotechnology: A Text for a New Science. *Techné*, 13(3):133–168, 2008. URL <https://scholar.lib.vt.edu/ejournals/SPT/v12n3/pdf/toumey.pdf>.
- [187] T. Tsukuda and H. Häkkinen. *Protected metal clusters: from fundamentals to applications*, volume 9 of *Frontiers of Nanoscience*. Elsevier, 1 edition, 2015. Ebook ISBN: 9780444635020.
- [188] R. V. Ulijn and A. M. Smith. Designing peptide based nanomaterials. *Chem. Soc. Rev.*, 37:664–675, 2008. .
- [189] M. Valle and A. R. Oganov. Crystal fingerprint space – a novel paradigm for studying crystal-structure sets. *Acta Crystallographica Section A*, 66(5):507–517, 2010. .
- [190] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.
- [191] A. Varadharajan, P. Kothari, and D. Gupta. Nanoscience in Cancer Treatment. *PRAXIS UNDERGRADUATE MEDICAL RESEARCH JOURNAL*, 3, 2020. URL [https://www.researchgate.net/profile/Ashvin-Varadharajan/publication/362117910\\_Nanoscience\\_in\\_cancer\\_treatment/links/62d75698441ed55f843cd9ce/Nanoscience-in-cancer-treatment.pdf](https://www.researchgate.net/profile/Ashvin-Varadharajan/publication/362117910_Nanoscience_in_cancer_treatment/links/62d75698441ed55f843cd9ce/Nanoscience-in-cancer-treatment.pdf).
- [192] K. Varga and J. A. Driscoll. *Computational Nanoscience : Applications for Molecules, Clusters, and Solids*. Cambridge University Press, Cambridge, UK, 2011. URL <https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=366307&site=ehost-live>.
- [193] M. Verleysen and D. François. The Curse of Dimensionality in Data Mining and Time Series Prediction. In J. Cabestany, A. Prieto, and F. Sandoval,

- editors, *Computational Intelligence and Bioinspired Systems*, 758–770. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
- [194] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, W. Czarnecki, A. Dudzik, A. Huang, P. Georgiev, R. Powell, T. Ewalds, D. Horgan, M. Kroiss, I. Danihelka, J. Agapiou, J. Oh, V. Dalibard, D. Choi, L. Sifre, Y. Sulsky, S. Vezhnevets, J. Molloy, T. Cai, D. Budden, T. Paine, C. Gulcehre, Z. Wang, T. Pfaff, T. Pohlen, D. Yogatama, J. Cohen, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, C. Apps, K. Kavukcuoglu, D. Hassabis, and D. Silver. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. <https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/>, 2019.
- [195] C. J. C. H. Watkins. *Learning from delayed rewards*. Ph.D. thesis, King’s College, Cambridge United Kingdom, 1989.
- [196] G. Wei, J. Zhao, Y. Feng, A. He, and J. Yu. A novel hybrid feature selection method based on dynamic feature importance. *Applied Soft Computing*, 93:106337, 2020. .
- [197] Y. Wei and B. Yan. Nano products in daily life: to know what we do not know. *National Science Review*, 3(4):414–415, 2016. . <https://academic.oup.com/nsr/article-pdf/3/4/414/31566092/nww073.pdf>.
- [198] D. Wettschereck. *A Study of Distance-Based Machine Learning Algorithms*. Ph.D. thesis, Oregon State University, 1994.
- [199] C. J. Willmott and K. Matsuura. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, 30(1):79–82, 2005. . URL <https://www.int-res.com/abstracts/cr/v30/n1/p79-82/>.
- [200] C. Wissler. The Spearman Correlation Formula. *Science*, 22(558):309–311, 1905. .
- [201] N. . workshop. NIPS 2003 workshop on feature extraction, 2003. URL <http://www.clopinet.com/isabelle/Projects/NIPS2003/>. References 19.2.2023.
- [202] Z. Xiao, E. Dellandrea, W. Dou, and L. Chen. ESFS: A new embedded feature selection method based on SFS. Research report, Ecole Centrale Lyon ; Université de Lyon ; LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/École Centrale de Lyon ; Tsinghua University, Beijing, China, 2008. URL <https://hal.science/hal-01984705>.
- [203] X. Xu, R. Ray, Y. Gu, H. J. Ploehn, L. Gearheart, K. Raker, and W. A. Scrivens. Electrophoretic Analysis and Purification of Fluorescent Single-Walled Carbon Nanotube Fragments. *Journal of the American Chemical Society*, 126(40):12736–12737, 2004. .



- [204] Y. Xu and R. Goodacre. On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. *Journal of Analysis and Testing*, 2(3):249–262, 2018. .
- [205] L. Yang and A. Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020. . URL <https://www.sciencedirect.com/science/article/pii/S0925231220311693>.
- [206] I. V. Yentekakis. The 10th Anniversary of Nanomaterials – Recent Advances in Environmental Nanoscience and Nanotechnology. *Nanomaterials*, 12(6), 2022. . URL <https://www.mdpi.com/2079-4991/12/6/915>.
- [207] S. S. Young, F. Yuan, and M. Zhu. Chemical Descriptors Are More Important Than Learning Algorithms for Modelling. *Molecular Informatics*, 31(10):707–710, 2012. .
- [208] Y. Zhang and Y. Yang. Cross-validation for selecting a model selection procedure. *Journal of Econometrics*, 187(1):95–112, 2015. . URL <https://www.sciencedirect.com/science/article/pii/S0304407615000305>.
- [209] J. Zhao, B. Liang, and Q. Chen. The key technology toward the self-driving car. *International Journal of Intelligent Unmanned Systems*, 6(1):2–20, 2018. .
- [210] S. Zhu, G. Xie, H. Cui, Q. Li, J. Forth, S. Yuan, J. Tian, Y. Pan, W. Guo, Y. Chai, Y. Zhang, Z. Yang, R. W. H. Yu, Y. Yu, S. Liu, Y. Chao, Y. Shen, S. Zhao, T. P. Russell, and H. C. Shum. Aquabots. *ACS Nano*, 16(9):13761–13770, 2022. .



## ORIGINAL PAPERS

PI

### MONTE CARLO SIMULATIONS OF $AU_{38}(SCH_3)_{24}$ NANOCLUSTER USING DISTANCE-BASED MACHINE LEARNING METHODS

by

Antti Pihlajamäki, Joonas Hämäläinen, Joakim Linja, Sami Malola, Paavo Nieminen,  
Tommi Kärkkäinen, Hannu Häkkinen 2020

Journal of Physical Chemistry A, Vol 124, 23, 4827–4836

<https://doi.org/10.1021/acs.jpca.0c01512>

Reproduced with kind permission of 2020 American Chemical Society.

# Monte Carlo Simulations of Au<sub>38</sub>(SCH<sub>3</sub>)<sub>24</sub> Nanocluster Using Distance-Based Machine Learning Methods

Published as part of *The Journal of Physical Chemistry virtual special issue "Machine Learning in Physical Chemistry"*.

Antti Pihlajamäki, Joonas Hamalainen, Joakim Linja, Paavo Nieminen, Sami Malola, Tommi Karkkainen, and Hannu Hakkinen\*



Cite This: *J. Phys. Chem. A* 2020, 124, 4827–4836



Read Online

ACCESS |



Metrics & More

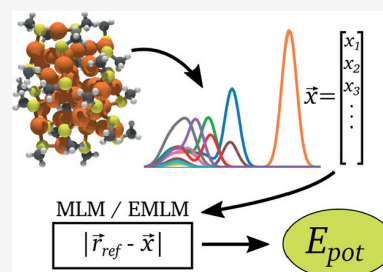


Article Recommendations



Supporting Information

**ABSTRACT:** We present an implementation of distance-based machine learning (ML) methods to create a realistic atomistic interaction potential to be used in Monte Carlo simulations of thermal dynamics of thiolate (SR) protected gold nanoclusters. The ML potential is trained for Au<sub>38</sub>(SR)<sub>24</sub> by using previously published, density functional theory (DFT) based, molecular dynamics (MD) simulation data on two experimentally characterized structural isomers of the cluster and validated against independent DFT MD simulations. This method opens a door to efficient probing of the configuration space for further investigations of thermal-dependent electronic and optical properties of Au<sub>38</sub>(SR)<sub>24</sub>. Our ML implementation strategy allows for generalization and accuracy control of distance-based ML models for complex nanostructures having several chemical elements and interactions of varying strength.



## INTRODUCTION

Monolayer-protected clusters (MPCs) are small metal nanoparticles that have a metal core with size ranging from a few atoms to a few hundred atoms and a protecting surface layer of organic molecules such as thiols, phosphines, alkynyls, or carbenes.<sup>1</sup> MPCs are synthesized via wet chemistry by reducing metal salts in the presence of the protecting molecules. A variety of synthesis recipes and combination of metals and protecting molecules yields a rich chemistry and a large array of products in terms of size, shape, and composition of metal cores and the molecular overlayer. The wide range of synthetic parameters gives a unique possibility to study the fundamental structure–stability–property relations and to engineer the properties for applications such as catalysis, plasmonics, biosensing, and drug delivery.

The first crystallographically resolved MPCs were reported already over 50 years ago (such as the so-called undecagold Au<sub>11</sub> cluster protected by phosphines<sup>2</sup>), and first advances in synthesis and structural characterization produced a series of mostly noble metal clusters protected by L-type (such as phosphine) and mixed L–X type (X being an electronegative ligand such as halide or thiolate) ligands. The largest such known cluster was the phosphine–halide protected Au<sub>39</sub>, reported in 1992.<sup>3</sup>

Considerable steps forward were taken when Brust and co-workers<sup>4</sup> reported a synthesis that produced all-thiolate protected gold clusters for an average size of two nanometers. Several new chemical compositions of both organo-soluble and

water-soluble clusters were reported soon after,<sup>5–8</sup> culminating to the breakthroughs of the first crystal structure of a large water-soluble all-thiol protected cluster, Au<sub>102</sub>(pMBA)<sub>44</sub> (pMBA = *p*-mercaptobenzoic acid) by the Kornberg group in 2007<sup>9</sup> as well as the organo-soluble Au<sub>25</sub>(PET)<sub>18</sub><sup>–10–12</sup> in 2008 and Au<sub>38</sub>(PET)<sub>24</sub> (PET = phenyl ethyl thiolate)<sup>13,14</sup> clusters in 2008–2010. Up to date, atomic structures of at least 150 different compounds are crystallographically known, which facilitates detailed theoretical computations and dynamical simulations of the properties of MPCs and greatly helps to correlate structures to measured properties in experimental data.

Density functional theory (DFT) methods are the cornerstone for all computations that need to deal with details of the electronic structure, such as studies of optical absorption, optical excitation, fluorescence, and magnetism. However, while giving the most accurate and detailed information, DFT methods are also numerically the most demanding. DFT computations of some of the largest structurally known MPCs like the thiolate-protected Ag<sub>374</sub><sup>15,16</sup> have to deal with up to 13 000 valence electrons, and even a single-point DFT energy

Received: February 21, 2020

Revised: April 30, 2020

Published: May 15, 2020

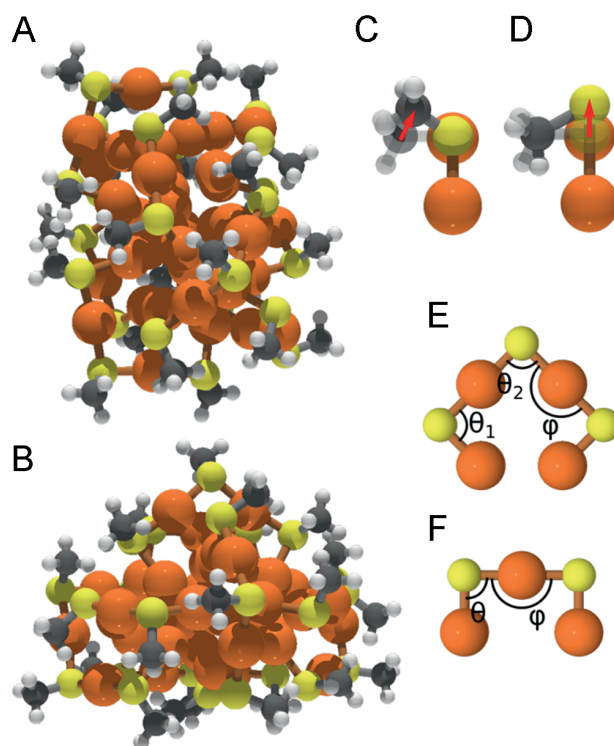


calculation can take minutes and use hundreds or even thousands of CPU cores in a supercomputer. Force fields describing gold–thiolate MPCs have been developed to be used in molecular dynamics (MD) simulations, e.g., in the context of ReaxFF<sup>17</sup> and AMBER-GROMACS.<sup>18</sup> Effective but reliable methods to simulate the atomic dynamics of MPCs are needed, for instance, to study interactions of the clusters with the environment in the solvent phase, or with biomolecules and biological materials (viruses, proteins, lipid layers etc.).<sup>19–21</sup> However, developing such force fields may be time-consuming, system- or problem-specific, and suffer from poor transferability. Finally, understanding of nucleation processes in formation reactions of MPCs or reactions between two different MPCs are fundamental unsolved issues that are currently out of reach of any usable simulation method.

Machine learning (ML) and data-driven methods are emerging as a promising alternative to analyze structure–property correlations and make systematic predictions of physicochemical properties in materials science.<sup>22,23</sup> So far, ML has been applied to relatively small systems such as molecules with up to a few tens of atoms or systems where degrees of freedom can be limited such as binding of an atom to the surface.<sup>24–28</sup> A few homogeneous systems such as bulk water<sup>29,30</sup> or pure metal nanoparticles<sup>31,32</sup> have been studied as well. There has been very few studies of applying ML to MPCs. Recently deep neural networks and support vector machines were applied successfully to predict formation of MPCs in varying synthesis conditions.<sup>33,34</sup>

Systems with diverse chemical environments, such as MPCs, possess a large number of degrees of freedom, a range of chemical interactions of varying strength, and may require large training sets in order to cover the chemical space thoroughly enough. The most popular ML methods include neural networks, kernel ridge regression and Gaussian processes.<sup>35</sup> Neural networks have a great potential to learn very complicated data, because of their large number of parameters, weights, and network shapes to be adjusted during training. On the other hand, this flexibility also makes the method prone to overfitting. Kernel ridge regression and Gaussian processes are versatile tools, since one can define different kernel functions suiting a problem at hand. These kernels can easily transform the method to a complex one.

Here we demonstrate that even simple distance-based methods are applicable to complex systems such as MPCs. We use two methods, the so-called Minimal Learning Machine (MLM)<sup>36</sup> and the Extreme Minimal Learning Machine (EMLM)<sup>37</sup> and create a ML potential for a gold–thiolate  $\text{Au}_{38}(\text{SR})_{24}$  cluster. We utilize our previously published extensive DFT MD simulation data<sup>38</sup> based on two known structural isomers of  $\text{Au}_{38}(\text{PET})_{24}$ <sup>13,39</sup> (Figure 1A,B) as the initial training set. We test the ML potential by performing Monte Carlo simulations up to 300 K and compare the cluster dynamics to that from DFT MD simulations. To our knowledge, this work reports the first successful demonstration of a ML potential for MPCs, suitable for fast explorations of the configurational space. An immediate application could be to combine the MLM/EMLM potential with the recently published algorithm<sup>40</sup> designed to build complete nanoparticle structures based only on information about the metal core, in order to accelerate structural discovery. Alternatively, the efficient probing of the configuration space at a desired temperature can be utilized to generate realistic cluster



**Figure 1.** Initial structures of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  are visualized for Q and T isomers in parts A and B, respectively. While moving sulfur atoms and methyls the orientation of the S–C bond has to be preserved. Part C shows how alignment is preserved if methyl is moved. Part D shows the same when sulfur atom is moved. A long protecting unit is visualized in part E) and a short unit in part F. In parts E and F, methyls are omitted for the sake of clarity. Key: orange, gold; yellow, sulfur; gray, carbon; white, hydrogen.

structures for further investigations of thermal-dependent electronic and optical properties of  $\text{Au}_{38}(\text{SR})_{24}$ .

## THEORETICAL METHODS

Here we discuss the necessary components of the development of the ML method to deal with dynamical simulations of thiolate protected gold nanoclusters. We introduce the used descriptor for the cluster structures, the general principles of the distance-based machine learning, and the Monte Carlo method to probe the configuration space.

**Many-Body Tensor Representation.** The Cartesian coordinates of atomic positions include the whole structural information about a single nanostructure, however one cannot use them to describe the system for a machine-learning method. If even a small rotation or translation is applied to the system, the coordinates would change, but physically, the situation is still the same. In order to overcome this problem, one needs to use a suitable structural descriptor, which are required to be invariant to translation, rotation, and permutation. Cartesian coordinates are not fulfilling any of these requirements. In addition to these requirements it is desirable that description would be continuous, unique in the sense of description–property correlation, and fast to be computed.<sup>41</sup> There have been several different approaches with a varying level of complexity to describe nanostructures for machine-learning methods. Frequently used descriptors in the field are atom-centered symmetry functions,<sup>42</sup> Coulomb

matrices,<sup>43</sup> Ewald sum and sine matrices,<sup>44</sup> bag of bonds,<sup>45</sup> Zernike functions,<sup>46,47</sup> and smooth overlap of atomic positions (SOAP),<sup>48</sup> to name a few. These descriptors can be divided to local and global ones depending on whether they describe the environment around a single atom or the whole system as relationships between atoms. In this study, we used a global descriptor called many-body tensor representation (MBTR),<sup>41</sup> which is implemented in the Dscribe package.<sup>49</sup> We chose to use a global descriptor instead of a local one, because it gives a straightforward and fast way to describe the system. It gives a single representation for a single configuration. A local descriptor, on the other hand, would have to be evaluated several times in order to describe every atom in the system. Since our system is quite large and has many different chemical interactions, a global descriptor such as the MBTR keeps the process simple and transparent.

The basic idea of the MBTR is based on a bag of bonds description. There, the system is first divided into the contributions of different element pairs and then described with pairwise distances between the atoms belonging to the elements of interest. Huo and Rupp used this as a starting point and formalized the basis of MBTR.<sup>41</sup> Afterward Jager et al. simplified the theoretical presentation<sup>50</sup> and Himanen et al. implemented it into the Dscribe package.<sup>49</sup> The backbone of the description is

$$f_k(x, z_1, z_2) = \sum_{i=1}^{N_{atoms,1}} \sum_{j=1}^{N_{atoms,2}} w_k(i, j) D(x, g_k(i, j)) \quad (1)$$

where

$$D(x, g) = (\sigma\sqrt{2\pi})^{-1} \exp\left(-\frac{(x-g)^2}{2\sigma^2}\right) \quad (2)$$

In eq 1, summations are going through atoms with atomic (element) numbers of  $z_1$  and  $z_2$ . Function  $D(x, g)$  introduces broadening, which can be controlled by changing the parameter  $\sigma$ . Here  $x$  is sweeping variable, which probes the values produced by the function  $g_k(i, j)$ . Parameter  $k$  is the one defining the properties that are used to describe the system. In the theory, there is no limits for  $k$ ; therefore, in principle, one can freely define a suitable property. Usually choices are  $k = 1$  for atomic numbers,  $k = 2$  for pairwise atomic distances (or the inverse of the distance), and  $k = 3$  for angles formed by three different atoms. In this study, we chose to set  $k = 2$  in order to use pairwise distances, therefore the weights are  $w_2(i, j) = \exp(-dR_{ij})$  and the property measure is defined as  $g_2(i, j) = R_{ij}^{-1}$ . Here  $d$  is a parameter, which is used to define the amount of weight for the contributions of atoms  $i$  and  $j$  if they are  $R_{ij}$  apart from each other.

As the name suggest, MBTR is a tensor with dimensions of  $N_{elements} \times N_{elements} \times n_x$  when  $k = 2$ .  $N_{elements}$  is the number of different elements in the system and  $n_x$  is the number of points that variable  $x$  can probe. Every element pair is described with their own summation but all pairs are using the same set of parameters. We list parameters as sets of {min, max,  $n_x$ ,  $\sigma$ ,  $d$ , cutoff}. First there are minimum and maximum values of the variable  $x$ .  $n_x$  is the number of points for  $x$ . As mentioned earlier,  $\sigma$  controls the broadening and  $d$  is used in weighting. Dscribe package has also its own parameter to define cutoff. Only the values of the eq 1, which are greater than the cutoff, are used in summation for every value of  $x$ . This affects the sensitivity of the descriptor and also the speed of

computations. A small cutoff value allows a large number of values to be included into the summation increasing the time spent for every element pair. On the other hand, a small cutoff would allow smaller changes in the structure to be visible in the description than a large cutoff. Using small cutoff values makes the descriptor sensitive but also very system-specific. Thus, there is a trade-off between accuracy and transferability.

**Distance-Based Machine Learning Methods.** *Minimal Learning Machine MLM.* Here we briefly introduce the theoretical background of the utilized distance-based machine-learning methods. First we go through the Minimal Learning Machine (MLM) formalized by de Souza Junior et al.<sup>36</sup> In general, we assume that a set of  $N_d$  input points  $X = \{\mathbf{x}_i\}_{i=1}^{N_d}$ ,  $\mathbf{x}_i \in \mathbb{R}^n$ , are given with the corresponding output points  $Y = \{\mathbf{y}_i\}_{i=1}^{N_d}$ ,  $\mathbf{y}_i \in \mathbb{R}^p$ , to be predicted. We restrict here to univariate (nonlinear) regression problems. In supervised machine learning, one usually trains a model to map input points to certain output directly or through some kernel space. In that case the mapping  $f: X \rightarrow Y$  between input and output spaces would be used to make the regression model as

$$\mathbf{Y} = f(\mathbf{X}) + \mathbf{E} \quad (3)$$

where  $\mathbf{E}$  denotes residuals. MLM, on the other hand, determines the Euclidean distances between input and reference points and then uses these distances to construct a linear regression model to predict the Euclidean distances in the output space. These predicted distances with respect to the output space reference points form a multilateration problem from which the actual output is computed.

Reference points are defined as  $M = \{\mathbf{m}_k\}_{k=1}^K$  with  $M \subseteq X$  and corresponding outputs are naturally  $T = \{\mathbf{t}_k\}_{k=1}^K$  with  $T \subseteq Y$ . Then input space distances  $d(\mathbf{x}_i, \mathbf{m}_k) = |\mathbf{x}_i - \mathbf{m}_k|$  are forming the distance matrix  $\mathbf{D}_x \in \mathbb{R}^{N_d \times K}$ . Analogously output space distances  $\delta(\mathbf{y}_i, \mathbf{t}_k) = |\mathbf{y}_i - \mathbf{t}_k|$  are presented in a matrix  $\Delta_y \in \mathbb{R}^{N_d \times K}$ . In the notation, Greek letters are used for output space distances in order to distinguish them from input space notations. Next the mapping  $g$  is used to create regression model between distances in input and output spaces as

$$\Delta_y = g(\mathbf{D}_x) + \mathbf{E} \quad (4)$$

Next, de Souza Junior et al. assume that the mapping  $g$  has a linear structure for each response. The model simplifies into a matrix product<sup>36</sup>

$$\Delta_y = \mathbf{D}_x \mathbf{B} + \mathbf{E} \quad (5)$$

In order to get the matrix  $\mathbf{B}$  containing the coefficients for the  $K$  responses some approximations are needed.  $\mathbf{B}$  is estimated from training data through minimizing the multivariate residual sum of squares. This provides a least-squares estimate of the matrix

$$\hat{\mathbf{B}} = (\mathbf{D}_x^T \mathbf{D}_x)^{-1} \mathbf{D}_x^T \Delta_y \quad (6)$$

Solving the  $\mathbf{B}$  corresponds to training of the model.

Now the last task is the multilateration problem in the output space. There is no single definite way to approach this problem, but many approaches can be applied.<sup>51</sup> The idea is to minimize the objective function of single output regression problem



$$J(y) = \sum_{k=1}^K ((y - t_k)^2 - \sum \mathbf{d}(\tilde{\mathbf{x}}, M) \hat{\mathbf{B}}_k)^2 \quad (7)$$

where  $\mathbf{d}(\tilde{\mathbf{x}}, M) \in \mathbb{R}^{1 \times K}$  is a vector containing distances between a new input  $\tilde{\mathbf{x}}$  and all reference points  $M$ . The task is to find suitable output  $y$ , which minimizes the objective function. In our case we adopted cubic equation introduced by Mesquita et al.<sup>52</sup> The minimum or minima are found where the derivative equals zero. Differentiation yields

$$\begin{aligned} Ky^3 - 3 \sum_{k=1}^K t_k y^2 + \sum_{k=1}^K (3t_k^2 - (\mathbf{d}(\tilde{\mathbf{x}}, M) \hat{\mathbf{B}}_k)^2) y \\ + \sum_{k=1}^K ((\mathbf{d}(\tilde{\mathbf{x}}, M) \hat{\mathbf{B}}_k)^2 - t_k^3) \\ = 0. \end{aligned} \quad (8)$$

This can be thought as a cubic equation  $ay^3 + by^2 + cy + d = 0$ . From three possible roots, we choose the one that yields the smallest value of the objective function.

**Extreme Minimal Learning Machine EMLM.** Another distance-based machine-learning method, which was used in this study, is the Extreme Minimal Learning Machine (EMLM). The origin of the method lies in the so-called Extreme Learning Machine (ELM), which are single-layer perceptrons with special training and optimization methods.<sup>53–57</sup> When their training methods are combined with the Euclidean distance basis of MLMs, one gets EMLM.<sup>37</sup>

The first step is again to collect  $N_d$  input points into a matrix  $\mathbf{X} \in \mathbb{R}^{n \times N_d}$ . Corresponding outputs are in a matrix  $\mathbf{Y} \in \mathbb{R}^{p \times N_d}$ . Here  $n$  and  $p$  are the lengths of single input and output vectors  $\mathbf{x}_i$  and  $\mathbf{y}_i$ . Input points  $\mathbf{x}_i$  are first operated with a kernel function  $\mathbf{h}(\cdot)$  forming new inputs  $\mathbf{H} \in \mathbb{R}^{K \times N_d}$ . Here  $\mathbf{h}(\cdot)$  is a vector valued function, which is used to calculate the input vector in a kernel space. Due to the fact that we are using distance-based method,  $K$  is the number of reference points; therefore, the elements of  $\mathbf{H}$  are defined as

$$\mathbf{H}_{i,j} = (\mathbf{h}(\mathbf{x}_j))_i = |\mathbf{m}_i - \mathbf{x}_j| \quad (9)$$

This is just the Euclidean distance between a reference point and an input point. We simplify the notation by writing  $\mathbf{h}_i \equiv \mathbf{h}(\mathbf{x}_j)$ . Now  $\mathbf{h}_i \in \mathbb{R}^{K \times 1}$  and  $\mathbf{H} \in \mathbb{R}^{K \times N_d}$ . Then as Karkkainen states, the training of the model is done through regularized least-squares (RLS) optimization problem<sup>37</sup>

$$\min_{\mathbf{V} \in \mathbb{R}^{p \times K}} \frac{1}{2N_d} \sum_{i=1}^{N_d} |\mathbf{V} \mathbf{h}_i - \mathbf{y}_i|^2 + \sum_{i=1}^p \frac{\alpha}{2K} \sum_{j=1}^K |\mathbf{V}_{ij}|^2 \quad (10)$$

The parameter  $\alpha$  is a small positive real number (square root of machine  $\epsilon$  by default) used for regularization.  $\mathbf{V}$  is a matrix containing the coefficients used for the actual regression and  $\mathbf{V} \in \mathbb{R}^{p \times K}$ . One could say, that  $\mathbf{V}$  and reference points together form the actual machine-learning model. The minimum of the optimization problem lies on the zero point of the matrix derivative. The optimal solution  $\mathbf{W} \equiv \mathbf{V}_{\text{optimal}}$  satisfies

$$\frac{1}{N_d} (\mathbf{W} \mathbf{H} - \mathbf{Y}) \mathbf{H}^T + \sum_{i=1}^p \frac{\alpha}{K} \mathbf{I} = 0 \quad (11)$$

After getting the optimal solution for the RLS problem, one can use  $\mathbf{W}$  to predict the output for a new arbitrary input  $\tilde{\mathbf{x}}$ . This is done as

$$f(\tilde{\mathbf{x}}) = \mathbf{W} \mathbf{h}(\tilde{\mathbf{x}}) \quad (12)$$

where  $\mathbf{h}$  is the same vector valued kernel function as before. With input vector  $\tilde{\mathbf{x}}$ , it yields a  $K \times 1$  vector. The elements of this vector are defined to be Euclidean distances as  $|\mathbf{m}_i - \tilde{\mathbf{x}}|$ .

We can see that the EMLM framework is fundamentally a kernel ridge regression with the Euclidean distance basis as a kernel. Because of the structural similarity to the linear radial basis function network, the EMLM model possesses the universal approximation capability.<sup>58–60</sup> MLM and EMLM have just one hyperparameter, which is the number of reference points. Overfitting is rarely an issue for distance based ML methods, therefore we can use all data points as reference points in training without worrying about overfitting.<sup>37,61</sup> There is no need for optimization of hyper- or metaparameters. This is a significant difference compared to the artificial neural networks, support vector machines, Gaussian processes or other popular ML methods. These methods require hyper- or metaparameter optimization through, for example, cross-validation.

**Monte Carlo.** We used Monte Carlo to simulate the dynamics of the  $\text{Au}_{38}(\text{SCH}_3)_{24}$  clusters with simplified methyl ligands. Clusters are divided to three different moving parts: gold, sulfur and methyl. Gold atoms are moved into a random direction according to the step size. Sulfur is moved in a similar fashion, but in order to preserve the orientation of sulfur-carbon bond, the methyl group is rotated making it to face the sulfur atom. The same principle is applied for the movement of the methyl groups. When methyl is moved according to the step size, the S-C bond orientation is preserved. In addition to this we allowed methyl group to rotate around the sulfur-carbon bond. The way how the alignment of sulfur-carbon bond is preserved is visualized in Figure 1, parts C and D. The stretching of carbon-hydrogen bond does not have a significant contribution to the total potential energy of the system; therefore, we decided to fix these bonds.

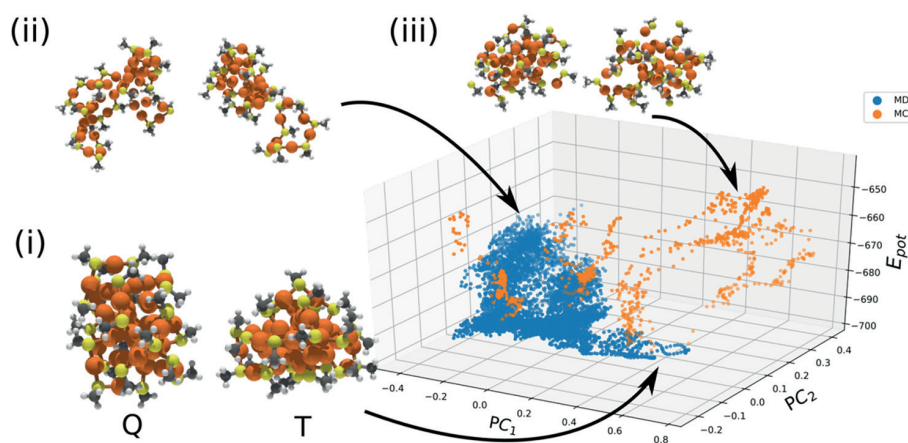
The acceptance of every move is decided according to the Metropolis question. The probability of the move to be accepted is defined as

$$P = \min \left\{ 1, \exp \left( \frac{-(E_{i+1} - E_i)}{k_B T} \right) \right\} \quad (13)$$

$E_i$  is the potential energy of the  $i$ th configuration and  $E_{i+1}$  is the potential energy of the configuration after a proposed move. Going downhill in energy landscape is always permitted but going uphill is accepted with certain probability defined by the energy difference and simulation temperature  $T$ . In the exponent  $k_B$  is the Boltzmann constant. The step size of a single move is adjusted during the simulations so that the acceptance of the moves is between 40% and 60%. This step size is the same throughout the whole cluster and it is not affected by the type of the moved block. During a MC step, all moving parts are sampled randomly, and every one of them has an opportunity to move. This means that one MC step consists of  $38 + 24 + 24 = 86$  trial moves.

## RESULTS AND DISCUSSION

**Generating Training Data and Training the Models.** The training data from the  $\text{Au}_{38}(\text{SCH}_3)_{24}$  clusters were



**Figure 2.** PCA visualization of MBTR descriptors of the training data. For the sake of clarity only 25% of the points are present in the graph. (i) the initial structures and (ii) high-temperature structures of the original MD simulations<sup>38</sup> (iii) snapshots from Monte Carlo simulations, where S–Au bonds have been broken. In parts ii and iii, left/right structures originate from Q/T isomers. Key: orange, gold; yellow, sulfur; gray, carbon; white, hydrogen.

generated using density functional theory (DFT) run with GPAW code.<sup>62,63</sup> The major training data were published earlier by Juarez-Mosqueda et al.<sup>38</sup> In that work, Born–Oppenheimer *NVT* molecular dynamics simulations were run for the so-called Q<sup>13</sup> and T<sup>39</sup> isomers of Au<sub>38</sub>(SCH<sub>3</sub>)<sub>24</sub> at various temperatures between 400 and 1200 K. To be consistent with the training data we used same level of theory (the Perdew–Burke–Ernzerhof (PBE) exchange–correlation functional<sup>64</sup>). The DFT MD simulation trajectories of Juarez-Mosqueda et al.<sup>38</sup> contained 12413 configurations for the Q isomer and 12647 for the T isomer.

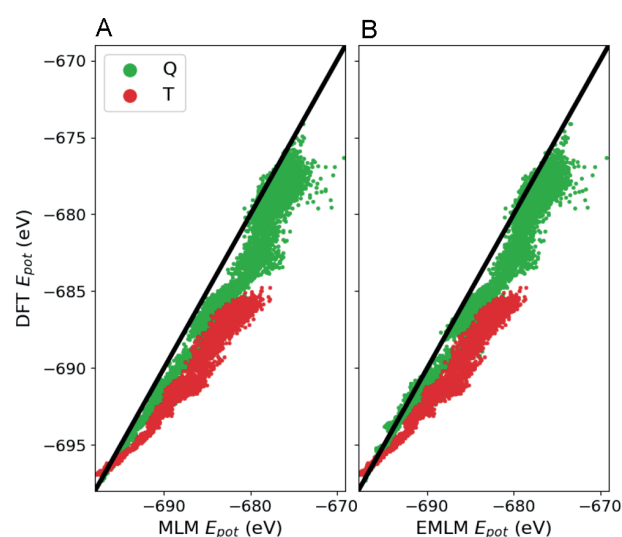
We used two different sets of MBTR parameters {min, max,  $n_x$ ,  $\sigma$ ,  $d$ , cutoff}. The first set was {0, 1.4, 100, 0.1, 0.5,  $10^{-3}$ } and the second set was {0, 1.2, 100, 0.045, 0.8,  $10^{-5}$ } (for a discussion on choosing the parameters, see the Supporting Information text and Figures S1 and S2). In the beginning, we trained MLM for the MBTR data corresponding to the first set of parameters. Minmax scaling was applied to the training data, so that descriptor values belonged to interval [0, 1]. As we mentioned earlier in the Theory section, overfitting is rarely an issue for MLM and EMLM. Therefore, we used the Full MLM and EMLM variants meaning that all data points were selected as reference points. We used MLM to predict potential energies during the Monte Carlo simulations in various simulation temperatures and with different starting structures taken from the training data. Monte Carlo frequently found the outer boundaries of the reference points pushing itself out of the working range of MLM. This resulted in erroneous potential energy values and nonphysical structures. In the Supporting Information text and Figure S3, we show that the MLM, which was trained only with the initial MD data,<sup>38</sup> is not able to handle configurations produced by the Monte Carlo. However, it can still find clear structure–energy correlation within the training data.

To cope with the erroneous behavior, we expanded the MLM training set including the MC-generated “unrealistic” configurations and their energies from DFT. The training set was expanded with 1580 new configurations for the Q and 2124 for the T isomer. After this we used the second set of MBTR parameters, which had improved descriptive possibilities (see Supporting Information). With the expanded training set and improved descriptor we trained both MLM and

EMLM. In Figure 2, the principal component analysis (PCA) of the MBTR shows that the training set contains a large variety of configurations of both isomers spanning a large area of the feature space. Due to the fact that MLM/EMLM methods are using the Euclidean distances to measure the similarity of input point it is educative to visualize how the data points are arranged in the feature space.

#### Validation: Potential Energy MLM/EMLM vs DFT-MD.

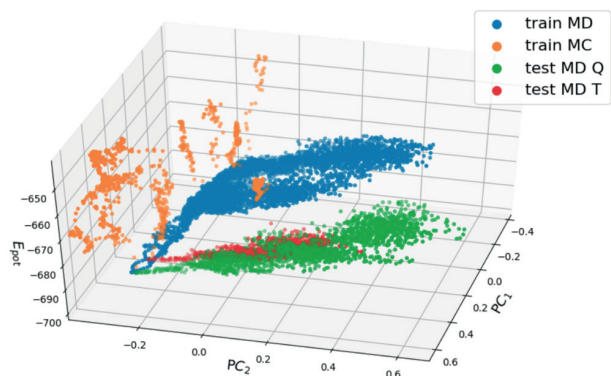
For validation, we created new independent DFT MD reference data sets for both Q and T isomers. For the Q isomer, we ran 2000 steps at 269 K, 2000 steps at 475 K, and 3653 steps at 795 K. For the T isomer we ran 2000 steps at 273 K and 2049 steps at 486 K. Potential energies were predicted for every configuration using both MLM and EMLM and compared to the actual DFT values from the MD run. The performance is seen in Figure 3. Generally, the predicted values correlate clearly with the DFT values, with the root-mean-



**Figure 3.** Correlation between the predicted potential energy from (A) MLM and (B) EMLM to the DFT energy from the MD calculations for Q and T isomers.



squared error (RMSE) being 2.98 eV for MLM and 2.67 eV for EMLM. The corresponding average relative errors are only 0.38% and 0.33%, respectively. The predicted energies are somewhat higher (less negative) than those from DFT. Our training set contains a lot of high energy configurations of  $\text{Au}_{38}(\text{SCH}_3)_{24}$ ; therefore, the set might be biased. The visualization of PCA in Figure 4 indicates that the new MD



**Figure 4.** Visualization for PCA from training data and test MD data. Potential energies on  $z$  axis are computed with DFT. The graph is rotated with respect to Figure 2. In order to keep visualization clear, only 25% of the points are included.

simulations are rather far away from the points in the original training set. However, they are not outside of the working region of the MLM and EMLM like the first Monte Carlo simulations, which were used to expand the training set. This enables distance-based methods to predict well the potential energy values.

**MC Simulations with EMLM-Predicted Energies.** As the most stringent test, we performed MC simulations of both Q and T isomers at temperatures of 200, 250, and 300 K, using the EMLM-predicted potential energy in the Metropolis criterion while advancing the dynamics. Typical simulations were run for 9000 to 10000 MC steps, one MC step consisting of 86 independent trial moves of the atoms (hence 86 EMLM energy evaluations per MC step). PCA of the runs at 300 K is shown in Figure 5A, indicating that the MC dynamics of both

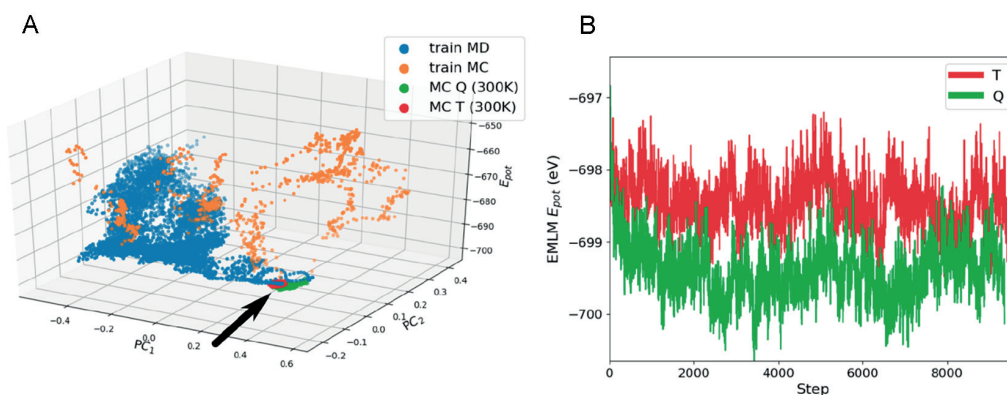
isomers are concentrated on a quite small region close to the  $T = 0$  K local potential energy minimum, as expected for this rather low temperature. Figure 5(B) shows the evolution of the potential energy of both isomers at 300 K indicating that the potential energy of the Q isomer is consistently lower by about 1 eV than that of the T isomer. This result is consistent with the energetics known from DFT.

We analyzed the statistics of selected bond distances and bond angles for both isomers from the MC runs at 200, 250, and 300 K. The last 500 MC steps from each simulations were used for the analysis. Figure 6 shows the statistics for the nearest neighbor Au–Au bonds in the metal core as well as for the S–Au and S–C bonds, and compares them to the statistics obtained from DFT MD runs at 268 and 474 K for Q isomer and 272 and 486 K for T isomer. We observe that the EMLM-MC runs generally slightly overestimate the Au–Au bonds in both isomers as compared to DFT MD. The peaks of the distributions are at 2.862 Å (MC) and 2.805 Å (MD) for Q isomer, and 2.845 Å (MC) and 2.805 Å (MD) for T isomer. For S–Au and S–C bonds, EMLM-MC and DFT-MD produce very similar distributions both regarding the peak position and width. This analysis shows that the EMLM-MC runs indeed are able to simulate the bond dynamics of the atoms in the harmonic vibration regime.

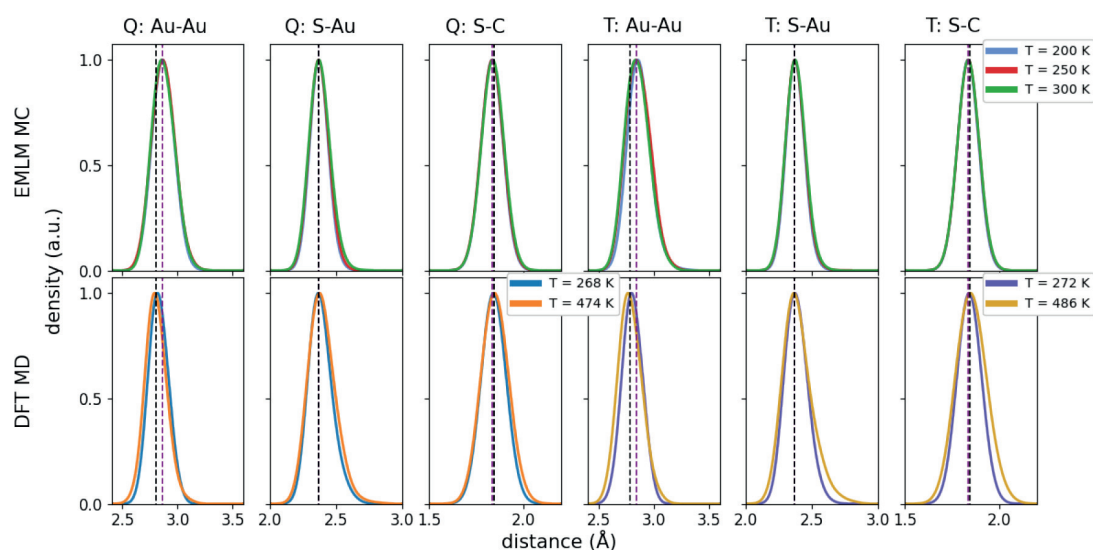
Figure 7 shows the corresponding comparison between EMLM-MC and DFT-MD data for Au–S–Au and S–Au–S angles. In the crystal structures of these isomers the Au–S–Au angle is close to  $90^\circ$  and S–Au–S angle close to  $170^\circ$  (Figure 1). We observe that the maxima of Au–S–Au angles produced by EMLM-MC are slightly smaller than  $90^\circ$ , with a small side peak around  $130^\circ$  for the T isomer. We see a wider scatter in describing the S–Au–S angles in EMLM-MC as compared to DFT-MD, with the distributions having a maximum around  $150^\circ$  and tail extending close to  $100^\circ$ . MD simulations shows distributions peaked around  $170^\circ$ . We assign these slight discrepancies to the  $k_2$  description of the MBTR which does not take into account any angular information.

## CONCLUSION

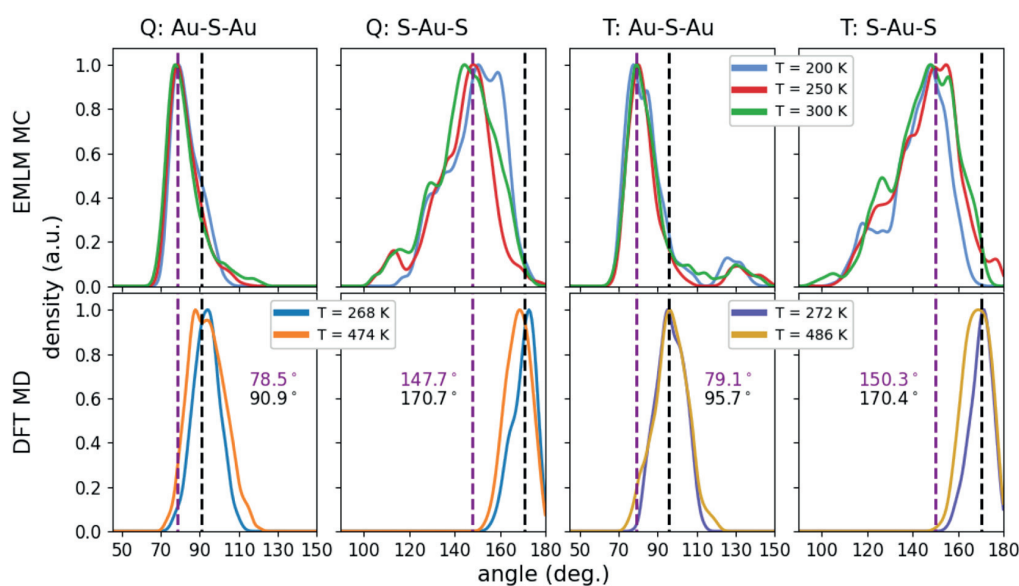
Distance-based machine-learning methods discussed in this study are conceptually straightforward and very simple to implement. We have shown here that they are suitable to



**Figure 5.** (A) Same as Figure 2, but including also the PCA analysis of EMLM MC runs at 300 K for isomers Q and T. The arrow highlights the region of the MC data. The analysis indicates that both of the isomers are vibrating close to their minima. Only 25% of the points are included into the Figure and PC1 values are multiplied with  $-1$  to produce a comparable graph. (B) Evolution of potential energies of both isomers predicted by EMLM during MC.



**Figure 6.** Top row: bond distance distributions from EMLM MC simulations at the indicated temperatures. Bottom row: the same data from DFT MD simulations at indicated temperatures. Labels on the top indicate the isomer and bond type. The vertical dashed lines indicate the average peak positions for every angle distribution in both MC and MD cases for every column (purple, MC; black, MD). Most of them are overlapping, and only black lines are visible. The statistics are summed from Gaussian-smoothened ( $\sigma = 0.05$  Å) data points.



**Figure 7.** Top row: Selected bond angles distributions from EMLM MC simulations at the indicated temperatures. Bottom row: the same data from DFT MD simulations at indicated temperatures. Labels on the top indicate the isomer and type of the angle. The vertical dashed lines indicate the average peak positions for every angle distribution in both MC and MD cases for every column (purple, MC; black, MD). The colored numbers show the averages. The statistics are summed from Gaussian-smoothened ( $\sigma = 1.75^\circ$ ) data points.

simulate complex systems such as MPCs that have a number of chemical interactions with varying strength, while resulting in significantly reduced computational cost as compared to DFT. The CPU time to predict the energy by using MLM or EMLM with MBTR k2-level descriptors for the atomic structure is several magnitudes smaller than for DFT. For a comparison, MLM/EMLM energy predictions were run on a single core of Intel Xeon CPU E5-2680 v3 @ 2.50 GHz with 8GB memory. Computing MBTR k2 with our parameters took about 0.07 s for one atomic structure. Prediction of the potential energy using MBTR k2 took about 0.05 s with EMLM and 0.56 s with MLM. The order-of-magnitude difference between MLM and

EMLM arises from the fact that the EMLM needs reference points only in the input space and is ready to give an output estimate from matrix and vector multiplication, while the MLM is predicting distances in the output space and solving a multilateration problem.

Excluding all angular information and using only pairwise distances to describe atomic structures with MBTR k2-level further helps to make these methods computationally light. The lack of angular information in MBTR k2 description does not mean, that our methods would not be able to reproduce reasonable bond angles. As shown in the [Supporting Information](#), we could improve the description of the angles

of protecting  $\text{RS}(\text{AuSR})_{n=1,2}$  units by tuning the parameters, although the MC simulations showed that the energy landscape produced by EMLM slightly differed from the one that DFT would yield.

Monte Carlo was shown to be an efficient strategy to study the energy landscape learned by MLM and EMLM. The method is not bound by any assumptions; therefore, it freely explores the feature space and gives useful insight of possible weaknesses of the machine-learning method. An important lesson learned in this work was that the initial MC simulations showed that our initial DFT-MD training set<sup>38</sup> was not extensive enough to train a comprehensive machine-learning method, since the DFT-MD produced atomistic configurations that were all “physical”. By enlarging the training data with the structures corresponding to the DFT energies of the “unphysical” configurations predicted by MLM/EMLM-MC back to the training data, we were able to teach the methods to avoid the unphysical regions of the configurational phase space.

Our future work involves further development of the models and descriptors for MPCs and other heterogeneous nanostructures. Here we used a global descriptor and predicted the potential energy of the system as a property of a whole system. Dividing the potential energy into atomic or molecular contributions creates in principle a way to get spatial insight into the energetics.<sup>26</sup> Fabrizio et al. have pointed out that it is reasonable to use global description when predicting global properties but it might cause size-dependence, which sometimes can be overcome with usage of local descriptions.<sup>65</sup> Our method is currently trained solely for  $\text{Au}_{38}(\text{SCH}_3)_{24}$  with the goal to demonstrate that distance-based machine-learning methods can be used to handle complex systems such as MPCs. We aim to generalize the methods by including other MPCs (other metals and ligands) and other sizes of gold–thiolate clusters in the training set.

## ■ ASSOCIATED CONTENT


### ● Supporting Information

The Supporting Information is available free of charge at <https://pubs.acs.org/doi/10.1021/acs.jpca.0c01512>.

Additional discussion on testing of the MBTR parameters (text and Figures S1 and S2) and performance of the MLM with the initial MD training data (Figure S3) (PDF)


## ■ AUTHOR INFORMATION


### Corresponding Author

**Hannu Hakkinen** – Department of Physics, Nanoscience Center and Department of Chemistry, Nanoscience Center, University of Jyväskylä, FI-40014 Jyväskylä, Finland;  [orcid.org/0000-0002-8558-5436](https://orcid.org/0000-0002-8558-5436); Email: [hannu.j.hakkinen@jyu.fi](mailto:hannu.j.hakkinen@jyu.fi)

### Authors

**Antti Pihlajamäki** – Department of Physics, Nanoscience Center, University of Jyväskylä, FI-40014 Jyväskylä, Finland

**Joonas Hamalainen** – Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland;  [orcid.org/0000-0002-8466-9232](https://orcid.org/0000-0002-8466-9232)

**Joakim Linja** – Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland;  [orcid.org/0000-0003-2573-1240](https://orcid.org/0000-0003-2573-1240)

**Paavo Nieminen** – Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland  
**Sami Malola** – Department of Physics, Nanoscience Center, University of Jyväskylä, FI-40014 Jyväskylä, Finland  
**Tommi Karkkainen** – Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland

Complete contact information is available at:  
<https://pubs.acs.org/10.1021/acs.jpca.0c01512>

### Notes

The authors declare no competing financial interest.

## ■ ACKNOWLEDGMENTS

This work was supported by the Academy of Finland through the AIPSE research program with Grant 315549 to H.H. and Grant 315550 to T.K., through the Universities Profiling Actions with Grant 311877 to T.K., and through H.H.’s Academy Professorship. MC simulations were done at the FGCI node at the University of Jyväskylä (persistent identifier: urn:nbn:fi:research-infras-2016072533) and the DFT MD simulations were run as part of PRACE Project 2018194723 at the Barcelona Supercomputing Centre.

## ■ REFERENCES

- (1) Tsukuda, T.; Hakkinen, H. *Protected metal clusters: from fundamentals to applications*; Elsevier: Amsterdam, The Netherlands, 2015.
- (2) McPartlin, M.; Mason, R.; Malatesta, L. Novel cluster complexes of gold(0)–gold(I). *J. Chem. Soc. D* **1969**, *0*, 334–334.
- (3) Teo, B. K.; Shi, X.; Zhang, H. Pure gold cluster of 1:9:9:1:9:9:1 layered structure: a novel 39-metal-atom cluster [(Ph3P)14Au39Cl6]Cl2 with an interstitial gold atom in a hexagonal antiprismatic cage. *J. Am. Chem. Soc.* **1992**, *114*, 2743–2745.
- (4) Brust, M.; Walker, M.; Bethell, D.; Schiffrin, D. J.; Whyman, R. Synthesis of thiol-derivatised gold nanoparticles in a two-phase liquid-liquid system. *J. Chem. Soc., Chem. Commun.* **1994**, *0*, 801–802.
- (5) Schaafl, T. G.; Whetten, R. L. Giant gold–glutathione cluster compounds: intense optical activity in metal-based transitions. *J. Phys. Chem. B* **2000**, *104*, 2630–2641.
- (6) Schaafl, T. G.; Shafiqullin, M. N.; Khoury, J. T.; Vezmar, I.; Whetten, R. L. Properties of a ubiquitous 29 kDa Au:SR cluster compound. *J. Phys. Chem. B* **2001**, *105*, 8785–8796.
- (7) Templeton, A. C.; Wuelfing, W. P.; Murray, R. W. Monolayer-protected cluster molecules. *Acc. Chem. Res.* **2000**, *33*, 27–36.
- (8) Negishi, Y.; Nobusada, K.; Tsukuda, T. Glutathione-protected gold clusters revisited: bridging the gap between gold(I)–thiolate complexes and thiolate-protected gold nanocrystals. *J. Am. Chem. Soc.* **2005**, *127*, 5261–5270.
- (9) Jazdzinsky, P. D.; Calero, G.; Ackerson, C. J.; Bushnell, D. A.; Kornberg, R. D. Structure of a thiol monolayer-protected gold nanoparticle at 1.1 Å resolution. *Science* **2007**, *318*, 430–433.
- (10) Heaven, M. W.; Dass, A.; White, P. S.; Holt, K. M.; Murray, R. W. Crystal structure of the gold nanoparticle [N(C8H17)4]–[Au25(SCH2CH2Ph)18]. *J. Am. Chem. Soc.* **2008**, *130*, 3754–3755.
- (11) Zhu, M.; Aikens, C. M.; Hollander, F. J.; Schatz, G. C.; Jin, R. Correlating the crystal structure of a thiol-protected Au25 cluster and optical properties. *J. Am. Chem. Soc.* **2008**, *130*, 5883–5885.
- (12) Akola, J.; Walter, M.; Whetten, R. L.; Hakkinen, H.; Gronbeck, H. On the structure of thiolate-protected Au25. *J. Am. Chem. Soc.* **2008**, *130*, 3756–3757.
- (13) Qian, H.; Eckenhoff, W. T.; Zhu, Y.; Pintauer, T.; Jin, R. Total structure determination of thiolate-protected Au38 nanoparticles. *J. Am. Chem. Soc.* **2010**, *132*, 8280–8281.
- (14) Lopez-Acevedo, O.; Tsunoyama, H.; Tsukuda, T.; Hakkinen, H.; Aikens, C. M. Chirality and electronic structure of the thiolate-protected Au38 nanocluster. *J. Am. Chem. Soc.* **2010**, *132*, 8210–8218.



- (15) Yang, H.; Wang, Y.; Chen, X.; Zhao, X.; Gu, L.; Huang, H.; Yan, J.; Xu, C.; Li, G.; Wu, J.; et al. Plasmonic twinned silver nanoparticles with molecular precision. *Nat. Commun.* **2016**, *7*, 12809.
- (16) Zhou, Q.; Kaappa, S.; Malola, S.; Lu, H.; Guan, D.; Li, Y.; Wang, H.; Xie, Z.; Ma, Z.; Hakkinen, H.; et al. Real-space imaging with pattern recognition of a ligand-protected Ag<sub>374</sub> nanocluster at sub-molecular resolution. *Nat. Commun.* **2018**, *9*, 2948.
- (17) Bae, G.-T.; Aikens, C. M. Improved ReaxFF force field parameters for Au-S-C-H systems. *J. Phys. Chem. A* **2013**, *117*, 10438–10446.
- (18) Pohjolainen, E.; Chen, X.; Malola, S.; Groenhof, G.; Hakkinen, H. A unified AMBER-compatible molecular mechanics force field for thiolate-protected gold nanoclusters. *J. Chem. Theory Comput.* **2016**, *12*, 1342–1350.
- (19) Marjomaki, V.; Lahtinen, T.; Martikainen, M.; Koivisto, J.; Malola, S.; Salorinne, K.; Pettersson, M.; Hakkinen, H. Site-specific targeting of enterovirus capsid by functionalized monodisperse gold nanoclusters. *Proc. Natl. Acad. Sci. U. S. A.* **2014**, *111*, 1277–1281.
- (20) Martikainen, M.; Salorinne, K.; Lahtinen, T.; Malola, S.; Permi, P.; Hakkinen, H.; Marjomaki, V. Hydrophobic pocket targeting probes for enteroviruses. *Nanoscale* **2015**, *7*, 17457–17467.
- (21) Pohjolainen, E.; Malola, S.; Groenhof, G.; Hakkinen, H. Exploring strategies for labeling viruses with gold nanoclusters through non-equilibrium molecular dynamics simulations. *Bioconjugate Chem.* **2017**, *28*, 2327–2339.
- (22) Schmidt, J.; Marques, M. R. G.; Botti, S.; Marques, M. A. L. Recent advances and applications of machine learning in solid-state materials science. *npj Comput. Mater.* **2019**, *5*, 83.
- (23) Schleder, G. R.; Padilha, A. C. M.; Acosta, C. M.; Costa, M.; Fazio, A. From DFT to machine learning: recent approaches to materials science—a review. *J. Phys. Materials* **2019**, *2*, 032001.
- (24) Rupp, M.; Tkatchenko, A.; Muller, K.-R.; von Lilienfeld, O. A. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.* **2012**, *108*, 058301.
- (25) Sun, J.; Wu, J.; Song, T.; Hu, L.; Shan, K.; Chen, G. Alternative approach to chemical accuracy: A neural networks-based first-principles method for heat of formation of molecules made of H, C, N, O, F, S, and Cl. *J. Phys. Chem. A* **2014**, *118*, 9120–9131.
- (26) Schutt, K. T.; Arbabzadah, F.; Chmiela, S.; Muller, K. R.; Tkatchenko, A. Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* **2017**, *8*, 13890.
- (27) Chen, X.; Jørgensen, M. S.; Li, J.; Hammer, B. Atomic energies from a convolutional neural network. *J. Chem. Theory Comput.* **2018**, *14*, 3933–3942.
- (28) Kolsbjerg, E. L.; Peterson, A. A.; Hammer, B. Neural-network-enhanced evolutionary algorithm applied to supported metal nanoparticles. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2018**, *97*, 195424.
- (29) Chan, H.; Cherukara, M. J.; Narayanan, B.; Loeffler, T. D.; Benmore, C.; Gray, S. K.; Sankaranarayanan, S. K. Machine learning coarse grained models for water. *Nat. Commun.* **2019**, *10*, 379.
- (30) Patra, T. K.; Loeffler, T. D.; Chan, H.; Cherukara, M. J.; Narayanan, B.; Sankaranarayanan, S. K. R. S. A coarse-grained deep neural network model for liquid water. *Appl. Phys. Lett.* **2019**, *115*, 193101.
- (31) Artrith, N.; Kolpak, A. M. Grand canonical molecular dynamics simulations of Cu-Au nanoalloys in thermal equilibrium using reactive ANN potentials. *Comput. Mater. Sci.* **2015**, *110*, 20–28.
- (32) Zeni, C.; Rossi, K.; Glielmo, A.; Fekete, A.; Gaston, N.; Baletto, F.; De Vita, A. Building machine learning force fields for nanoclusters. *J. Chem. Phys.* **2018**, *148*, 241739.
- (33) Li, J.; Chen, T.; Lim, K.; Chen, L.; Khan, S. A.; Xie, J.; Wang, X. Deep learning accelerated gold nanocluster synthesis. *Adv. Intell. Syst.* **2019**, *1*, 1900029.
- (34) Copp, S. M.; Swasey, S. M.; Gorovits, A.; Bogdanov, P.; Gwinn, E. G. General approach for machine learning-aided design of DNA-stabilized silver clusters. *Chem. Mater.* **2020**, *32*, 430–437.
- (35) Murphy, K. P. *Machine learning: A probabilistic perspective*; MIT Press: Cambridge, MA, 2012.
- (36) de Souza Junior, A. H.; Corona, F.; Barreto, G. A.; Miche, Y.; Lendasse, A. Minimal Learning Machine: A novel supervised distance-based approach for regression and classification. *Neurocomputing* **2015**, *164*, 34–44.
- (37) Karkkainen, T. Extreme minimal learning machine: Ridge regression with distance-based basis. *Neurocomputing* **2019**, *342*, 33–48.
- (38) Juarez-Mosqueda, R.; Malola, S.; Hakkinen, H. Ab initio molecular dynamics studies of Au<sub>38</sub>(SR)<sub>24</sub> isomers under heating. *Eur. Phys. J. D* **2019**, *73*, 62.
- (39) Tian, S.; Li, Y.-Z.; Li, M.-B.; Yuan, J.; Yang, J.; Wu, Z.; Jin, R. Structural isomerism in gold nanoparticles revealed by x-ray crystallography. *Nat. Commun.* **2015**, *6*, 8667.
- (40) Malola, S.; Nieminen, P.; Pihlajamaki, A.; Hamalainen, J.; Karkkainen, T.; Hakkinen, H. A method for structure prediction of metal-ligand interfaces of hybrid nanoparticles. *Nat. Commun.* **2019**, *10*, 3973.
- (41) Huo, H.; Rupp, M. Unified representation of molecules and crystals for machine learning. *arXiv* 2017, 1704.06439v3 [physics.chem-ph].
- (42) Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.* **2011**, *134*, 074106.
- (43) Rupp, M.; Tkatchenko, A.; Muller, K.-R.; von Lilienfeld, O. A. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.* **2012**, *108*, 058301.
- (44) Faber, F.; Lindmaa, A.; von Lilienfeld, O. A.; Armiento, R. Crystal structure representations for machine learning models of formation energies. *Int. J. Quantum Chem.* **2015**, *115*, 1094–1101.
- (45) Hansen, K.; Biegler, F.; Ramakrishnan, R.; Pronobis, W.; von Lilienfeld, O. A.; Muller, K.-R.; Tkatchenko, A. Machine learning predictions of molecular properties: accurate many-body potentials and nonlocality in chemical space. *J. Phys. Chem. Lett.* **2015**, *6*, 2326–2331.
- (46) Canterakis, N. 3D Zernike moments and Zernike affine invariants for 3D image analysis and recognition. *11th Scandinavian Conference on Image Analysis*; 1999; pp 85–93.
- (47) Novotni, M.; Klein, R. Shape retrieval using 3D Zernike descriptors. *Computer-Aided Design* **2004**, *36*, 1047–1062.
- (48) Bartok, A. P.; Kondor, R.; Csanyi, G. On representing chemical environments. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2013**, *87*, 184115.
- (49) Himanen, L.; Jager, M. O. J.; Morooka, E. V.; Federici Canova, F.; Ranawat, Y. S.; Gao, D. Z.; Rinke, P.; Foster, A. S. Dscribe: Library of descriptors for machine learning in materials science. *Comput. Phys. Commun.* **2020**, *247*, 106949.
- (50) Jager, M. O. J.; Morooka, E. V.; Federici Canova, F.; Himanen, L.; Foster, A. S. Machine learning hydrogen adsorption on nanoclusters through structural descriptors. *npj Comput. Mater.* **2018**, *4*, 37.
- (51) Navidi, W.; Murphy, W. S., Jr.; Hereman, W. Statistical methods in surveying by trilateration. *Comput. Stat. Data Anal.* **1998**, *27*, 209–227.
- (52) Mesquita, D. P. P.; Gomes, J. P. P.; Souza Junior, A. H. Ensemble of efficient minimal learning machines for classification and regression. *Neural Process Lett.* **2017**, *46*, 751–766.
- (53) Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: A new learning scheme of feedforward neural networks. *Proc. IEEE Int. Joint Conf. Neural Netw.* **2004**, 985–990.
- (54) Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501.
- (55) Huang, G.-B.; Zhou, H.; Ding, X.; Zhang, R. Extreme learning machine for regression and multiclass classification. *IEEE Trans. Syst., Man, Cybern. B, Cybern.* **2012**, *42*, 513–529.
- (56) Cambria, E.; Huang, G.-B.; Kasun, L. L. C.; Zhou, H.; Vong, C. M.; Lin, J.; Yin, J.; Cai, Z.; Liu, Q.; Li, K.; et al. Extreme learning machines [trends & controversies]. *IEEE Intelligent Systems* **2013**, *28*, 30–59.

- (57) Akusok, A.; Bjork, K.-M.; Miche, Y.; Lendasse, A. High-performance extreme learning machines: A complete toolbox for big data applications. *IEEE Access* **2015**, *3*, 1011–1025.
- (58) Poggio, T.; Girosi, F. Networks for approximation and learning. *Proc. IEEE* **1990**, *78*, 1481–1497.
- (59) Park, J.; Sandberg, I. W. Universal approximation using radial-basis-function networks. *Neural Comput* **1991**, *3*, 246–257.
- (60) Liao, Y.; Fang, S.-C.; Nuttle, H. L. Relaxed conditions for radial-basis function networks to be universal approximators. *Neural Networks* **2003**, *16*, 1019–1028.
- (61) Hamalainen, J.; Alencar, A. S. C.; Karkkainen, T.; Mattos, C. L. C.; Souza Junior, A. H.; Gomes, J. P. P. Minimal Learning Machine: Theoretical results and clustering-based reference point selection. *arXiv* 2019, 1909.09978v1 [cs.LG].
- (62) Enkovaara, J.; Rostgaard, C.; Mortensen, J. J.; Chen, J.; Dulak, M.; Ferrighi, L.; Gavnholt, J.; Glinsvad, C.; Haikola, V.; Hansen, H. A.; et al. Electronic structure calculations with GPAW: a real-space implementation of the projector augmented-wave method. *J. Phys.: Condens. Matter* **2010**, *22*, 253202.
- (63) Mortensen, J. J.; Hansen, L. B.; Jacobsen, K. W. Real-space grid implementation of the projector augmented wave method. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2005**, *71*, 035109.
- (64) Perdew, J. P.; Burke, K.; Ernzerhof, M. Generalized gradient approximation made simple. *Phys. Rev. Lett.* **1996**, *77*, 3865.
- (65) Fabrizio, A.; Grisafi, A.; Meyer, B.; Ceriotti, M.; Corminboeuf, C. Electron density learning of non-covalent systems. *Chem. Sci.* **2019**, *10*, 9424–9432.

# Supporting Information for: Monte Carlo Simulations of $\text{Au}_{38}(\text{SCH}_3)_{24}$ Nanocluster Using Distance-Based Machine Learning Methods

Antti Pihlajamäki,<sup>†</sup> Joonas Hämäläinen,<sup>‡</sup> Joakim Linja,<sup>‡</sup> Paavo Nieminen,<sup>‡</sup> Sami  
Malola,<sup>†</sup> Tommi Kärkkäinen,<sup>‡</sup> and Hannu Häkkinen<sup>\*,†,¶</sup>

<sup>†</sup>*Department of Physics, Nanoscience Center, University of Jyväskylä, FI-40014 Jyväskylä,  
Finland*

<sup>‡</sup>*Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland*

<sup>¶</sup>*Department of Chemistry, Nanoscience Center, University of Jyväskylä, FI-40014  
Jyväskylä, Finland*

E-mail: hannu.j.hakkinen@jyu.fi



# 1 Effects of MBTR k2 parameters to Monte Carlo simulation

The theory behind the MBTR descriptor is explained in the main article. It has a few parameters, which affect its descriptive effectiveness. One can define minimum and maximum values of the sweeping variable  $x$ , adjust Gaussian broadening with  $\sigma$ , increase or decrease the effect of long distance terms with  $d$  and adjust the summation of distributions with cut-off.<sup>1,2</sup> We used the MBTR descriptor with  $k = 2$  (MBTR k2), which contains only pairwise distances in the description. However, it does not totally neglect angular information. The parametrization actually affects the angles in protecting units during Monte Carlo simulations.

We present parameters as sets of  $\{\text{min,max},n_x,\sigma,d,\text{cutoff}\}$ . The first used set was  $\{0, 1.4, 100, 0.1, 0.5, 10^{-3}\}$  and the second one was  $\{0, 1.2, 100, 0.045, 0.8, 10^{-5}\}$ . The MBTR is visualized in the top row of Figures S1 and S2. The most significant pairwise term is S-Au, which is drawn with thick red line in the Figures. When the first parameter set is used, the S-Au curve is dominated by the peak at about  $x \approx 0.2$ . This corresponds to the distance of 5.0 Å. This is not the bond distance between the closest neighboring S and Au atoms. On the other hand, the MBTR shows two separate peaks when the second parameter set is used. One is at  $x \approx 0.2$  and another one at  $x \approx 0.4$ . The second peak corresponds to the region close to 2.5 Å, which is close to the bond length of S-Au bond. This shows clearly that using the second parameter set descriptor can distinguish closest and second closest S and Au neighbors.

We trained EMLMs and ran Monte Carlo simulations at 200 K, 250 K and 300 K using both parameter sets. The mechanics of Monte Carlo are explained in the main article. Simulations were run for 9000 to 10000 steps and the last 500 were used for the analysis. In the Figures S1 and S2, the angle distributions are presented for the corresponding MBTR parameters. It is clear that the distributions are much more well defined, when the second

parameter set is used. Especially the S-Au-S angle improves when the method uses the latter parameter set.

## 2 Pitfalls of using molecular dynamics as a training data

Molecular dynamics (MD) simulations are always deterministic. They create a path in a configuration space as a function of time or simulation steps. This creates a pitfall for machine learning methods, especially for those whose construction relies on the actual observations, like the reference points with the distance-based methods.

In the beginning we used the first set of MBTR parameters to describe the structures of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  clusters. The structures were from the publication of Juarez-Mosqueda *et al.*<sup>3</sup> The data set contained 25060 configurations in total. This data was then used to train the Minimal Learning Machine (MLM). From the whole data set 80% was randomly chosen to the training set. All training data points were selected as reference points during the training process. Finally the remaining 20% of the MD data was used for testing. The test results can be seen in Figure S3. It seems that the predicted potential energies of MLM follow accurately the results of DFT. In other words, the MLM could find a clear structure-property correlation from the data set. Unfortunately the MLM is greatly restricted to the path that MD had made. It is not a difficult task to predict the property (in our case potential energy) between two similar data points along the path but predicting what is outside the path is much more difficult. This was seen in Monte Carlo simulations. They frequently broke the structures and made non-physical configurations, when this MLM was used to predict potential energies. In the main article principal component analysis (PCA) of the MBTR descriptors in Figure 2 shows how much the predicted structures differ from the original training data. In order to improve the generalization capability of the method we used configurations from these Monte Carlo simulations to expand the training set.

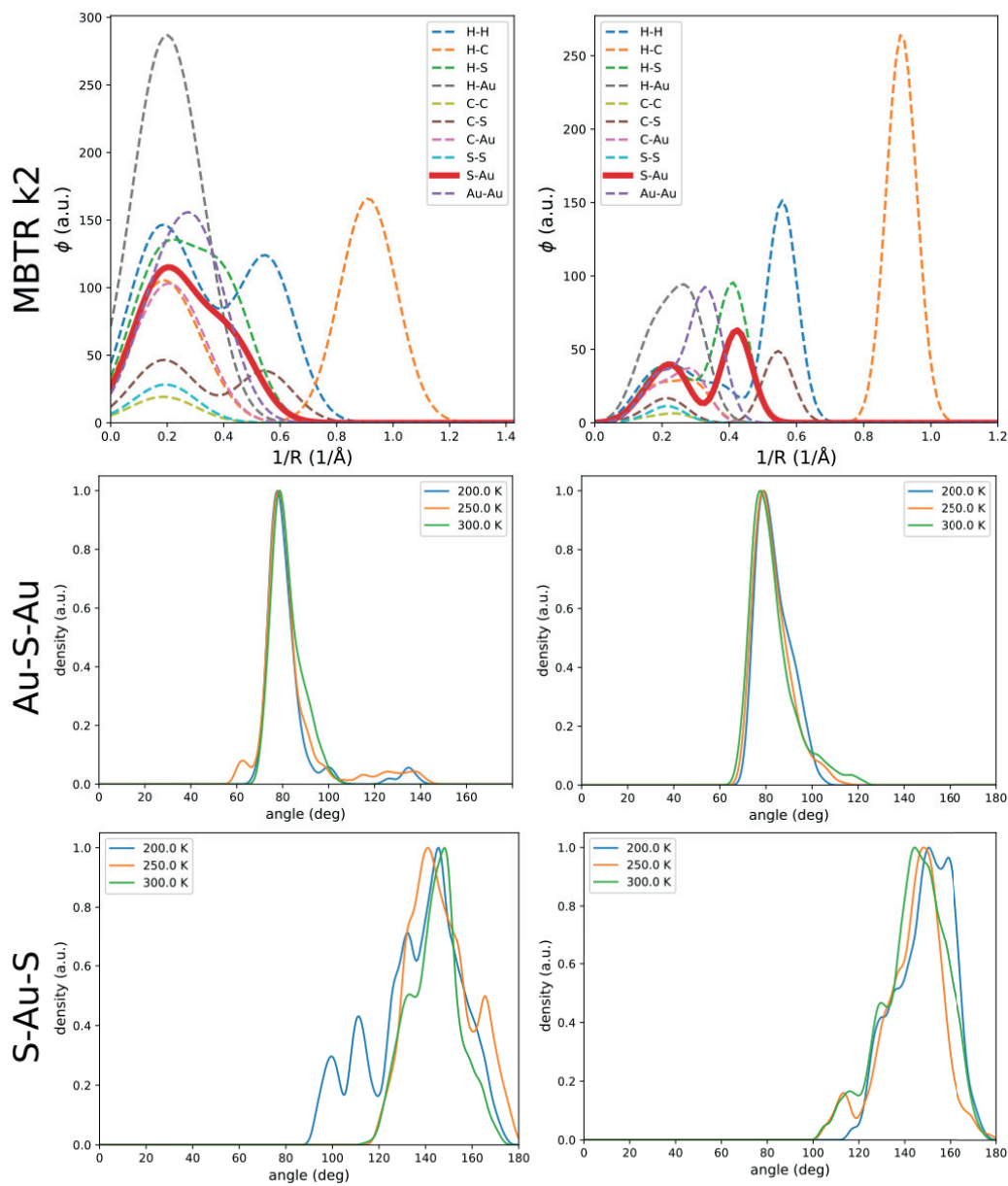


Figure S1: Here we visualize the effect of MBTR k2 to the angles of protecting units during the Monte Carlo simulations of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  Q. In the top row MBTR k2 is shown for different element pairs. Left side shows the results for the first parameter set and right side for the second set. The statistics of angles are summed from gaussian-smoothened ( $\sigma = 1.75^\circ$ ) data points.

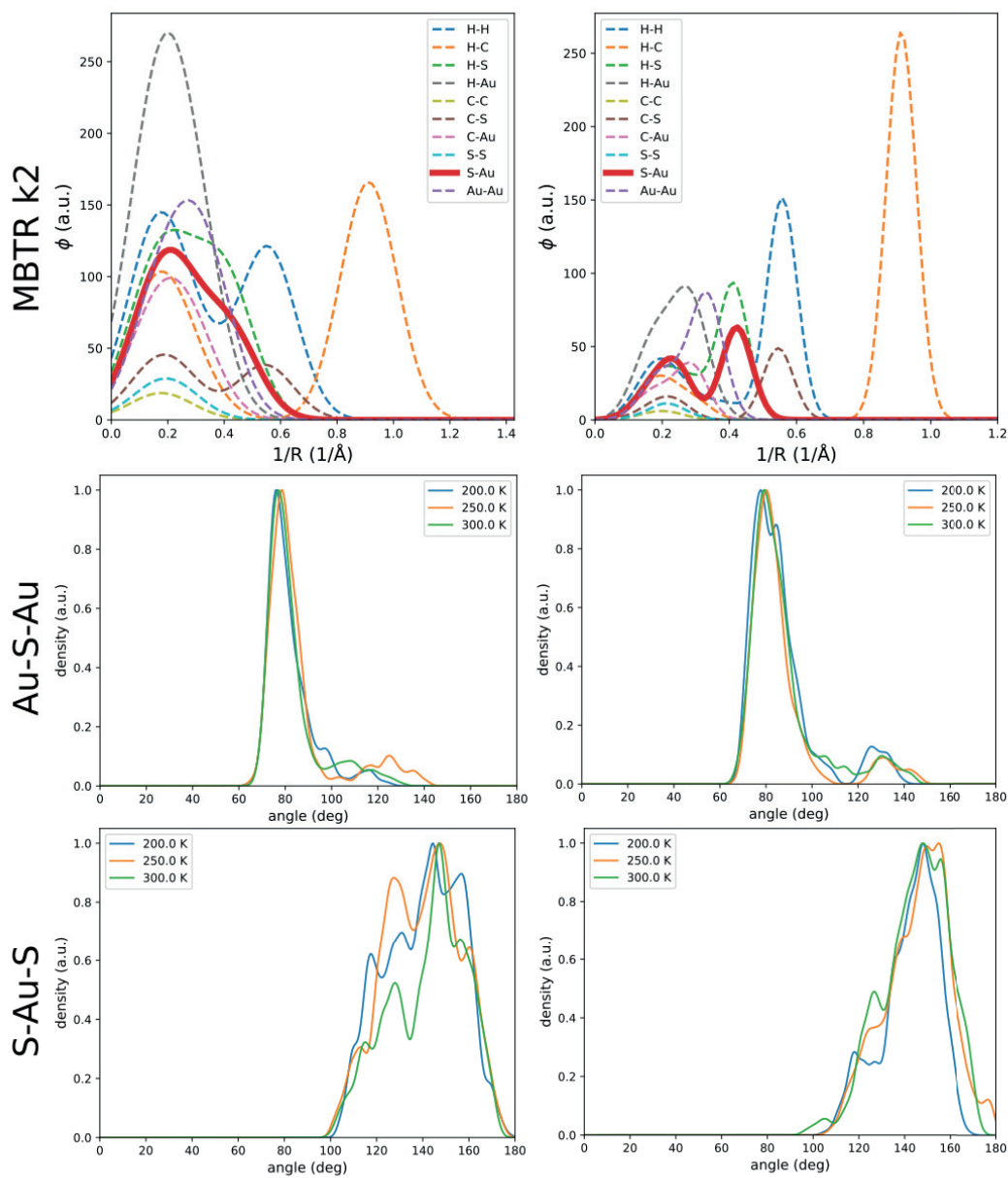


Figure S2: Here we visualize the effect of MBTR k2 to the angles of protecting units during the Monte Carlo simulations of  $\text{Au}_{38}(\text{SCH}_3)_{24}$  T. In the top row MBTR k2 is shown for different element pairs. Left side shows the results for the first parameter set and right side for the second set. The statistics of angles are summed from gaussian-smoothened ( $\sigma = 1.75^\circ$ ) data points.

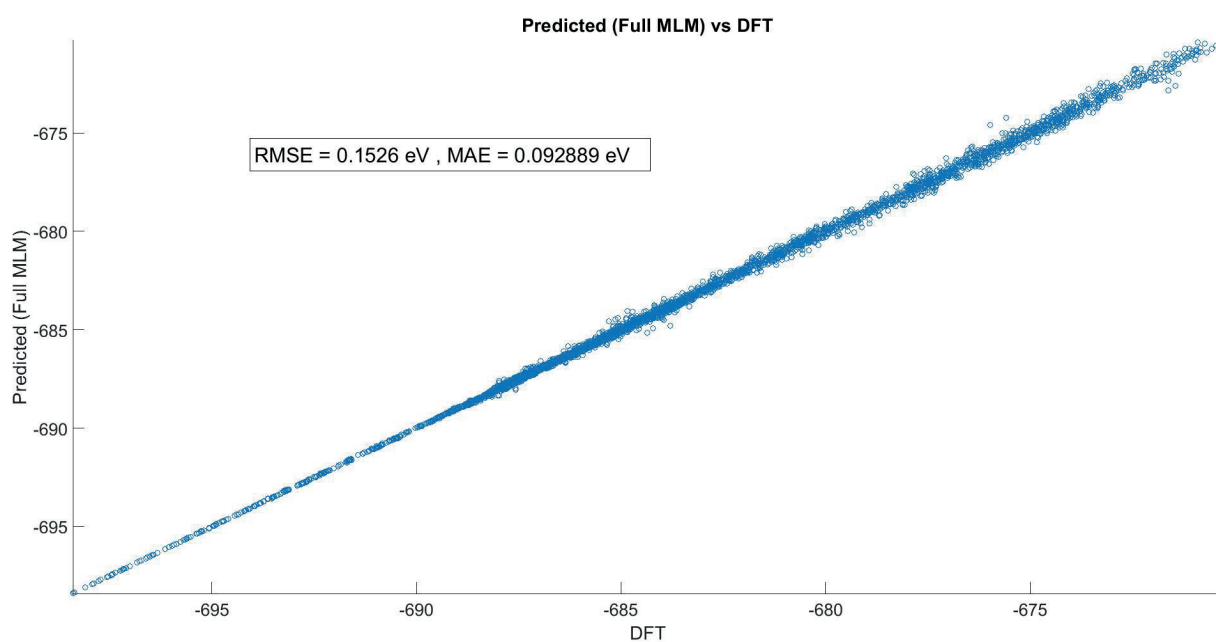


Figure S3: Here the potential energies predicted by MLM are shown as a function of real DFT level potential energies. Data set contains both Q and T isomers of  $\text{Au}_{38}(\text{SCH}_3)_{24}$ . From the data 80% is used as reference structures and the remaining 20% are used for testing. When MLM is interpolating within the set its predictive power is excellent. RMSE = root mean squared error, MAE = mean absolute error

## References

- (1) Huo, H.; Rupp, M. Unified Representation of Molecules and Crystals for Machine Learning, arXiv.org/1704.06439v3. **2017**,
- (2) Himanen, L.; Jäger, M. O. J.; Morooka, E. V.; Federici Canova, F.; Ranawat, Y. S.; Gao, D. Z.; Rinke, P.; Foster, A. S. Dscribe: Library of descriptors for machine learning in materials science. *Comput. Phys. Commun.* **2020**, *247*, 106949.
- (3) Juarez-Mosqueda, R.; Malola, S.; Häkkinen, H. Ab initio molecular dynamics studies of Au<sub>38</sub>(SR)<sub>24</sub> isomers under heating. *Eur. Phys. J. D.* **2019**, *73*, 62.





**PII**

**DO RANDOMIZED ALGORITHMS IMPROVE THE  
EFFICIENCY OF MINIMAL LEARNING MACHINE?**

by

Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, Tommi Kärkkäinen 2020

Machine Learning & Knowledge Extraction, Vol 2, 4, 533–557

<https://doi.org/10.3390/make2040029>

Reproduced with kind permission of 2020 MDPI.



*machine learning &  
knowledge extraction*



Article

---

# Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine?

---

Joakim Linja Joonas Hämäläinen Paavo Nieminen and Tommi Kärkkäinen



<https://doi.org/10.3390/make2040029>



Article

# Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine?

Joakim Linja <sup>\*</sup>, Joonas Hämäläinen , Paavo Nieminen and Tommi Kärkkäinen

Faculty of Information Technology, University of Jyväskylä, P.O. Box 35, University of Jyväskylä, FI-40014 Jyväskylä, Finland; joonas.k.hamalainen@jyu.fi (J.H.); paavo.j.nieminen@jyu.fi (P.N.); tommi.p.karkkainen@jyu.fi (T.K.)

\* Correspondence: joakim.j.linja@jyu.fi

Received: 29 September 2020; Accepted: 11 November 2020; Published: 13 November 2020



**Abstract:** Minimal Learning Machine (MLM) is a recently popularized supervised learning method, which is composed of distance-regression and multilateration steps. The computational complexity of MLM is dominated by the solution of an ordinary least-squares problem. Several different solvers can be applied to the resulting linear problem. In this paper, a thorough comparison of possible and recently proposed, especially randomized, algorithms is carried out for this problem with a representative set of regression datasets. In addition, we compare MLM with shallow and deep feedforward neural network models and study the effects of the number of observations and the number of features with a special dataset. To our knowledge, this is the first time that both scalability and accuracy of such a distance-regression model are being compared to this extent. We expect our results to be useful on shedding light on the capabilities of MLM and in assessing what solution algorithms can improve the efficiency of MLM. We conclude that (i) randomized solvers are an attractive option when the computing time or resources are limited and (ii) MLM can be used as an out-of-the-box tool especially for high-dimensional problems.

**Keywords:** machine learning; supervised learning; distance-based regression; minimal learning machine; approximate algorithms; ordinary least-squares; singular value decomposition; random projection

## 1. Introduction

Minimal Learning Machine (MLM) [1,2] is a supervised learning method that is based on a linear multi-output regression model between the input and output space distance matrices. The distance matrices are computed with respect to a subset of data points referred to as reference points. Although the mapping between the distance matrices is conducted linearly, the kernel-based construction using pairwise distances to the reference points enables the MLM to learn nonlinear relations from data for classification and regression problems. Promising results were obtained with the MLM in several applications [3–5].

The original formulation of the MLM was proposed in [1] and fully formalized in [2]. However, closely related proximity-based machine learning methods were proposed already from the start of the 21st century in [6–10]. Although the proposed methods in [6,9] are also based on using distance kernel in supervised learning, they are very different from the MLM. In the MLM, distances are used directly as features, and the outputs of the distance regression model are only used to define the multilateration problem which can be solved efficiently to obtain the actual prediction. The distance kernel-based construction of the MLM provides advantages compared to the more popular supervised methods such as Neural Networks and Support Vector Machines: The MLM has only one hyperparameter—the number of reference points—and over-learning seems not to occur in multidimensional input spaces [3,11–14]. Moreover, compared to currently popular feedforward

deep learning techniques [15] with nonlinear optimization of multiple layers of weights whose dimensions depend on the data dimension, the number of unknowns in the linear problem for the MLM is independent of the number of features (see Section 2.1). Hence, the simple formulation and straightforward training can improve the efficiency and reliability of the machine learning framework where the MLM is being applied. Practitioners do not need to tune various metaparameters related to the structure of the model and the way in which it is trained, so that more efforts can be dedicated to the way in which a particular application is being presented for supervised learning (see, e.g., [3]).

Lately, it was shown that the MLM possesses basic theoretical guarantees [12]: interpolation and universal approximation capability. To reduce the complexity of the distance regression model, reference point selection methods for the MLM were proposed for classification [16,17] and regression [12]. In Ref. [13], use of a data independent regularization matrix in the MLM training was proposed to reduce the MLM model's complexity similarly to reference point selection. The prediction of the MLM is obtained by solving a multilateration optimization problem determined by the predicted distances of the distance matrix mapping. In general, methods such as Levenberg-Marquardt [1], Newton [14], or Localization Linear System [12] (LLS) can be used to solve this problem efficiently. However, in a single-output regression and in classification, iterative algorithms can be completely avoided by using analytic formulae and nearest neighbor classification [11,18]. Hence, computationally the most demanding step of the MLM is to recover the linear distance matrix mapping from the ordinary least-squares (OLS) problem during training. If cross-validation is used to optimize the MLM's only hyperparameter, the OLS problem has to be solved several times [12].

One possibility to reduce the computational burden in the linear distance regression is to use the iterative Singular Value Decomposition (SVD) update (pp. 101–102, [19]), which tries to reduce the total computational complexity by using the previously computed regression model. A more straightforward approach to speed up the computation is to use randomized SVD (pp. 49–50, [20]) or randomized LU decomposition (pp. 251–252, [21]) where the main idea is to compute the decomposition in a lower dimensional space approximately and then expand the result to the original dimensions. Comparisons between randomized and non-randomized solvers were reported previously, e.g., in [20], but to the best of our knowledge, such comparisons have not been performed when the overall quality of a machine learning framework, here the MLM, is being assessed. We also present a comparison of the MLM with shallow and deep feedforward neural network (FNN) models, experimenting on the effects of the number of features and the number of observations with both methods.

Let us briefly summarize our main findings here. Direct solvers, especially *Cholesky decomposition* hold their position as the go-to solver if accuracy of the MLM model is the main objective. They can be also extended to GPUs [22]. Randomized solvers are a valid option if time is an issue or the accuracy of the model can be slightly relaxed. In addition, differently from feedforward networks, the MLM remains both efficient and accurate in a high-dimensional situation where even 90% of the given features are unnecessary. Therefore, based on the comparison, the MLM should be generally preferred over FNN for high-dimensional datasets.

The contents of the paper are as follows: First, in Section 2, we introduce the basic MLM formulation and solvers for the MLM's distance matrix mapping. In Section 3, we show experimental results for the solver comparison with several real datasets. The expanded result tables are presented in Supplementary Material. Discussion of the results is given in Section 4. Finally, Section 5 concludes the paper.

## 2. Methods

Next, we summarize the basic MLM method [2,11], the reference point selection algorithm RS-maximin [11,12], and different choices of the linear system solvers.

### 2.1. MLM in a Nutshell

Given a set of inputs  $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ , where  $\mathbf{x}_i \in \mathbb{R}^M$ , and the corresponding outputs  $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$ , where  $\mathbf{y}_i \in \mathbb{R}^L$ , the MLM originates from the distance matrices  $\mathbf{D}_x \in \mathbb{R}^{N \times K}$  and  $\mathbf{D}_y \in \mathbb{R}^{N \times K}$ . These are computed with respect to sets of reference points  $\mathbf{R} = \{\mathbf{r}_k\}_{k=1}^K \subseteq \mathbf{X}$  and  $\mathbf{T} = \{\mathbf{t}_k\}_{k=1}^K \subseteq \mathbf{Y}$ , where  $\mathbf{D}_x(i, k) = \|\mathbf{x}_i - \mathbf{r}_k\|$ ,  $\mathbf{D}_y(i, k) = \|\mathbf{y}_i - \mathbf{t}_k\|$ , and  $K \leq N$ . Here  $\|\cdot\|$  denotes the originally proposed euclidean distance [2], although the MLM method is applicable with any dissimilarity metric.

The linear mapping between the distance matrices can be recovered from the Ordinary Least-Squares (OLS) estimate (p. 36, [2])

$$\mathbf{B} = (\mathbf{D}_x \mathbf{D}_x)^{-1} \mathbf{D}_x \mathbf{D}_y. \quad (1)$$

For the special case when  $N = K$  we have a single solution if the matrix  $\mathbf{D}_x$  is of full-rank (p. 36, [2]). Then the mapping can be solved simply with

$$\mathbf{B} = \mathbf{D}_x^{-1} \mathbf{D}_y. \quad (2)$$

Equations (1) and (2) suggest that the inversion of the input distance matrix  $\mathbf{D}_x$  has a central role in the computational efficiency of the MLM. It is a dense matrix with non-negative components because with distinct inputs only the distance from each reference point to itself ( $\|\mathbf{x}_i - \mathbf{x}_i\|$ ) is zero.

The original suggestion in [2] was to select the reference points, i.e., locations of the distance-based basis functions, randomly. However, this does not ensure proper exploration and representation of the convex hull of the data. For this purpose, the RS-maximin algorithm was later suggested (p. 11, [12]), (pp. 36–37, [11]) for the reference point selection. RS-maximin is based on maximin clustering method by Gonzales [23]. It selects new reference points sequentially from the input space starting from the mean of data by maximizing the distance to the already selected reference points. This selection method makes the MLM approach fully deterministic and enforces possible duplicate observations in the input data to be the last ones to be selected.

The training phase of the MLM simply consists of the creation of distance matrices and the recovery of the coefficients  $\mathbf{B}$  of the linear distance regression. To predict the output  $\tilde{\mathbf{y}}$  of a new, unseen input  $\tilde{\mathbf{x}}$  with the MLM, one needs to solve a multilateration problem [2]

$$\mathcal{J}(\tilde{\mathbf{y}}) = \sum_{i=1}^K \left( \|\tilde{\mathbf{y}} - \mathbf{t}_i\|^2 - \mathbf{ff}_i^2 \right)^2, \quad (3)$$

where  $\mathbf{ff}_i = [\|\tilde{\mathbf{x}} - \mathbf{r}_1\|, \dots, \|\tilde{\mathbf{x}} - \mathbf{r}_K\|] \mathbf{B}$ . As explained in Section 1, in single-output regression and multi-class classification, the second step can be solved efficiently by using direct methods [11,18].

As the training phase of the MLM is based on learning a multi-target regression model for the distance matrices, the computational costs and scalability of the method are solely determined by the number of observations also for the high-dimensional input and output data. More precisely, the number of features do not affect the dimensions of  $\mathbf{B}$  as it is determined by the number of observations. It does affect the computation time of  $\mathbf{B}$  as vectors need to be computed for  $\mathbf{B}$ . This is a significant difference, e.g., compared to the feedforward randomized [24] or deep [15] learning techniques.

### 2.2. OLS Calculation Methods

A total of nine different approaches for solving the matrix inverse in Equation (1) in the context of the MLM were chosen for the comparison. While matrix inversion is a thoroughly explored subject, the effect of a solver on the performance of the MLM, from both computational efficiency and prediction accuracy perspectives, has not been presented before. Iterative, inexact stochastic gradient-based methods such as Adam [25] are typically used in deep learning without rigorous control on the solution

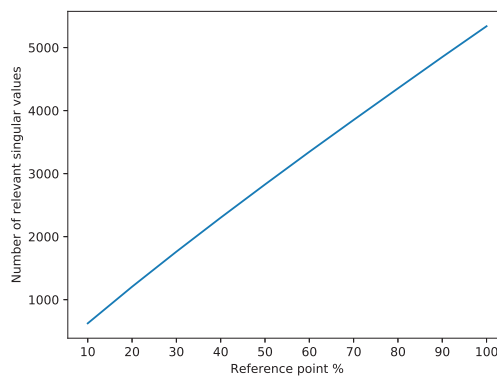
accuracy of the nonlinear optimization problem during training, so here we test how randomized approximate solution affects the prediction capability of the MLM.

The methods were selected on the basis of covering different solution techniques and being available. The selected approaches are summarized in Table 1 along with their abbreviations used in the plots. Please note that the recently proposed randomized LU decomposition method in [21] was omitted from the comparison because it is based on a randomized SVD template which is readily covered by the other techniques.

**Table 1.** Methods compared.

Name	Abbreviation	Source
np.lstsq	Lstsq_X	[26]
np.inverse	np_inv_X	[26]
np.solve	np_sol_X	[26]
Cholesky decomposition	Cho.Dec	[26,27]
np.svd	np_svd	[26]
sp.svd	sp_svd	[27]
rank K random SVD	rKrSVD_XX	[20]
sk rank random SVD	sk_rrSVD_XX	[28]
SVD update	SVD-upd	[19]

More of the practical details of the methods compared are given in Section 3.1. It is also relevant to note that when using RS-maximin as the reference point selection algorithm for the MLM, the rank of matrix  $D_x$  increases as a function of the number of reference points as shown in Figure 1. We show the mean of the number of relevant singular values in Figure 1 as each simulation was repeated 40 times for each dataset and reference point percentage. This is explained further in Section 3.



**Figure 1.** Mean of the number of relevant singular values in matrix  $D_x$  as the function of reference points with dataset Computer Activity.

The singular value decomposition [29] is a factorization method of a matrix  $A \in \mathbb{R}^{N \times K}$  in which it can be expressed as the product of three matrices: unitary  $U \in \mathbb{R}^{N \times N}$ , diagonal  $S \in \mathbb{R}^{N \times K}$  and unitary  $V \in \mathbb{R}^{K \times K}$  in the way of  $A = USV$ . The randomized SVD [20,28] methods are based on a random projection to a lower matrix rank by constructing a desired rank randomized orthogonal matrix and using it with the original matrix to construct the singular value decomposition. Their goal is to approximate the input matrix to a high enough degree to be useful in practical situations while providing speedup in computational times by momentarily moving the calculation to a lower dimension.



The use of randomized SVD's in solving  $\mathbf{B}$  with Equation (1) is based on the singular value decomposition of the input space distance matrix  $\mathbf{D}_x = \mathbf{U}\mathbf{S}\mathbf{V}$  which allows us to avoid computing the matrix inverse. By substituting  $\mathbf{D}_x = \mathbf{U}\mathbf{S}\mathbf{V}$  into Equation (1) we get

$$\begin{aligned}\mathbf{B} &= (\mathbf{V}\mathbf{S}\mathbf{U}\mathbf{U}\mathbf{S}\mathbf{V})^{-1}\mathbf{V}\mathbf{S}\mathbf{U}\mathbf{D}_y \\ &= (\mathbf{V}\mathbf{S}^2\mathbf{V})^{-1}\mathbf{V}\mathbf{S}\mathbf{U}\mathbf{D}_y \\ &= (\mathbf{V})^{-1}\mathbf{S}^{-2}\mathbf{V}^{-1}\mathbf{V}\mathbf{S}\mathbf{U}\mathbf{D}_y \\ &= \mathbf{V}\mathbf{S}^{-1}\mathbf{U}\mathbf{D}_y.\end{aligned}\tag{4}$$

Equation (4) is useful since the SVD decomposition can be used as it is and there is no need to reconstruct  $\mathbf{D}_x$  from the decomposition or to compute heavy matrix inversions. In this paper, we use Equation (4) in the place of Equation (1) when using SVD based methods.

### 3. Experiments and Results

In this section we describe the experimental setup and show the experimental results.

#### 3.1. Details on the Methods

Of the nine tested unique methods, five were tested multiple times with small differences. Methods *Lstsq*, *np.inverse* and *np.solve* had two versions, one with Tikhonov regularization [30] and one without. Methods *rank K random SVD* and *sk rank random SVD* are based on random projection to a lower matrix rank and were tested with five different settings, one for each remaining rank percentage (20%, 40%, 60%, 80%, and 100%). Thus, the total number of tested methods and their versions was 20.

Of the methods, *np.lstsq*, *np.inverse*, *np.solve*, *Cholesky decomposition*, *np.svd*, *sp.svd* and *sk rank random SVD* were readily implemented in commonly available Python packages [26–28]. Of these, *np.inverse* and *np.solve* require the input matrix to be square and invertible. *Cholesky decomposition* requires the input matrix to be a Hermitian positive-definite matrix. Matrix  $\mathbf{D}_x$  in (1) is not guaranteed to satisfy these assumptions. As the readily implemented methods are readily available with documentation and source code, we will not go into detailed explanations here. As their names imply, *np.lstsq* uses a least squares solver to compute the approximate solution, *np.inverse* solves the inverse of the given matrix and *np.solve* computes the solution to a linear equation. Decomposition methods *Cholesky decomposition*, *np.svd*, *sp.svd* and *sk rank random SVD* compute their respective decomposition and SVD based ones make use of Equation (4). The *Cholesky decomposition* solves the inverse of a matrix by computing  $\mathbf{L}$  and using a specialized solver made for triangular groups of equations.

Martinsson et al. [20] proposed a rank- $k$  random SVD algorithm, for which we could not find any available implementation. Our implementation is based on algorithms 4.3 and 5.1 by (pp. 25–28, [31]). Moreover, the SVD update by (pp. 101–102, [19]) was readily available for Python 2.7 (link in source for original) and had to be modified to use Python 3.

Finally, the so-called industry standard least squares solver with Tikhonov regularization (*Lstsq*) was chosen as the baseline in the statistical analysis of the MLM model's generalization capability (i.e., cross-validation error) using the Kruskal–Wallis H test [32].

#### 3.2. Datasets

A representative set of publically available machine learning datasets were chosen for the tests, as presented in Table 2. Of the public datasets, the first five are actual regression datasets and, therefore, primarily used in this study. They were previously used in the MLM research and they were chosen to enable us to probe the scalability of the OLS computation methods (p. 13, [12]). The largest dataset, *Mnist*, was created by combining the training set and the test set. Because of our special research interest in nanotechnology [3], the density functional theory (DFT) oriented nanostructures

as gathered by Juarez-Mosqueda et al. [33] were used to create nine different special datasets. These different feature representations of the  $Au_{38Q}$  hybrid nanostructures allow us to assess the effect of the number of features and the number of observations. The details on how these nine instances of data were formed are given in Appendix C.

**Table 2.** Datasets used.

Dataset name	Acronym	# Obser.	# Feat.	# Uniq. Targets	Source
Breast Cancer Progn. (Wisc.)	BC	194	32	94	[34]
Boston Housing	BH	506	13	229	[34]
Airplane Companies Stocks	AC	950	9	203	[35]
Computer Activity (CPU)	CA	8192	21	56	[36]
Census (House 8L)	CE	22,784	8	2045	[36]
Mnist	MN	70,000	784	10	[37]
$Au_{38Q}$	AuNx-yk	12,413	4000	12,413	[33]

Preprocessing each dataset included removal of incomplete observations and constant features, and normalization of the input and output ranges into  $[0,1]$  with minmax-scaling. Duplicate observations were not removed. This only affects the *Census* dataset due to it being the only one with duplicate observations (one duplicate). The duplicate was present in the simulations when the portion of the reference points increased but it did not affect the results because of the use of RS-maximin as the reference point selection algorithm (see Section 2.1).

The first two features (measurement identification number and class-based outcome variable) were removed from the Breast Cancer dataset and the feature “disease-free time or recurrence time” was used as the target output. For the other datasets, the last feature was used as the output feature. Table 2 presents the dimensions of the input matrices for the datasets. The sources of the datasets include a link to where we obtained the datasets.

### 3.3. Experimental Setup

Experiments were carried out using Python 3.7.3 with two Intel Xeon E5-2680 v3 CPUs (for a total of 24 threads at 2.50 GHz) and 188 GB RAM. The main packages were NumPy (version 1.16.4) [26], Scikit-learn (version 0.20.3) [28] and SciPy (version 1.3.0) [27]. Source code for the implemented  $rKrSVD\_XX$  is provided in the supplementary files along with MLM and RS-maximin. Each dataset was randomly split into a training set (80%) and a test set (20%) 100 times. We measured the root mean square error (RMSE) value of the MLM test error and the process time taken to compute Equation (1) when the portion of the selected reference points increased according to  $[10\%, 20\%, \dots, 100\%]$ . The measurements were performed on all methods for all reference points before a new randomly generated dataset split was created. This was to ensure that the results from each iteration would be comparable.

Exceptions had to be made with the *Census* dataset and *SVD update* due to time constraints. The dataset *Census* was randomly split 40 times instead of 100, a smaller set of methods were tested and with reference points  $[10\%, 20\%, \dots, 70\%]$ . *SVD update* had to be measured differently, due to its iterative nature. The method was used to evaluate Equation (1) at the same reference points as the other methods. In the required intermediary update iterations, only the internal update was called to avoid reconstructing the SVD decomposition which the *SVD update* holds within itself. For *SVD update*, each provided process time value includes the time taken to evaluate Equation (1) and the time that was taken to get to that evaluation. The *SVD update* in total is the slowest of the selected methods and it was only experimented with the *Breast Cancer*, *Boston Housing* and *Airplane Companies Stocks* datasets.

Based on the results, we wished to study lower remaining rank percentages and chose to do this with *Census* and *Mnist*. The reasons are further expanded upon in Section 4. We selected *Cholesky Decomposition* as the baseline and compared it to *sk rank random SVD* with remaining rank percentages  $[1\%, 10\%]$  at reference points  $[10\%, 20\%, \dots, 100\%]$ . The measurements were repeated 30 times at

each reference point by randomly selecting the training and test sets. We repeated the process time and the model accuracy measurements 40 times. Each with randomly selected training and test set. The full MLM case of Equation (2) is not used in the experiments. The results as a function of different reference point percentages need to be comparable to each other, which removes the possibility of using Equation (2).

It is also interesting to compare the performance of MLM with a shallow and a deep feedforward neural network (FNN) models due to their popularity [38]. The experiments with FNNs were carried out using the popular Tensorflow\_cpu 2.1 for Python 3 framework [39]. Both a shallow model with one hidden layer as well as a deep network architecture with three hidden layers, and four transformation layers altogether, were tested. We applied systematic, gradual decrease strategy to fix the sizes of the hidden layers. More precisely:

#### Deep feedforward neural network (FNN-4)

1. Input layer of size  $M$ ,
2. Feedforward layer with sigmoid activation function of size  $M/2$ ,
3. Feedforward layer with sigmoid activation function of size  $M/4$ ,
4. Feedforward layer with sigmoid activation function of size  $M/8$ ,
5. Feedforward layer with sigmoid activation function of size 1,

#### Shallow feedforward neural network (FNN-2)

1. Input layer of size  $M$ ,
2. Feedforward layer with sigmoid activation function of size  $M/2$ ,
3. Feedforward layer with sigmoid activation function of size 1,

FNNs were trained using maximum of 150 epochs using a batch size 10 with an early stopping criteria that followed the RMSE test error. I.E. the default settings for Adam-solver and sequential dense layer model given by Tensorflow 2.1 plus the early stopping criteria. Because of random initialization, the training process of FNN is not deterministic, so we repeated the training 10 times and selected the result based on the minimal training error. From the 40 results of the random splits into training/test datasets, we then calculated the mean and standard deviation for the test errors. The random splits into training/test datasets were the same as with the MLM, limiting to the first 40 in the case of smaller datasets (where total of 100 splits were used with the MLM). We measured the training time of the FNNs and compared them to Equation (1) computation time on the same hardware. The reported process times for the FNNs are the average of the sum of 10 trainings made for each training/test split.

Finally, we performed experimentation on the  $Au_{38Q}$  dataset. The dataset and its variants allowed us to study the effects of the number of observations and the number of features on the MLM by doing a method comparison between *Cholesky decomposition* and *random SVD* with remaining rank percentages [1%, 10%]. We also computed results with the FNNs for the  $Au_{38Q}$  dataset.

The MLM measurements were performed as previously, we had *Cholesky Decomposition* and variants of *sk\_rrSVD\_XX* (remaining rank percentages [1, 10]) as the solvers. Each dataset was randomly split into 40 different training and test datasets with 80/20 ratio and the reported normalized RMSE value is measured from the test set as well as the process time taken to solve Equation (1). FNN measurements were also carried out in the previous way with the 40 random 80/20 splits and 10 repeats at each random split.

### 3.4. Results

We were interested in measuring the MLM test error (RMSE) and process time (time taken to compute Equation (1)) and we measured them as a function of reference point percentages [10%, 100%] for each dataset. We report the numerical values in tables and of these tables, we show the RMSE table for reference point percentage 70% in Table 3 here. Due to the number of tables, we opted to present the rest in Supplementary Material (Tables S1–S19). In Table 3, we report the mean  $\bar{x}$  and

standard deviation  $\sigma$  of each measured method, dataset and reference point percentage combination. For rigorous statistical analysis, [40], we also show the result of Kruskal–Wallis H test (significance level 0.05) by bolding the text of the mean values that are considered to be different from the mean result. In the data tables, below the symbol of either mean  $\bar{x}$  or standard deviation  $\sigma$  is the order of magnitude of the numbers in the same column.

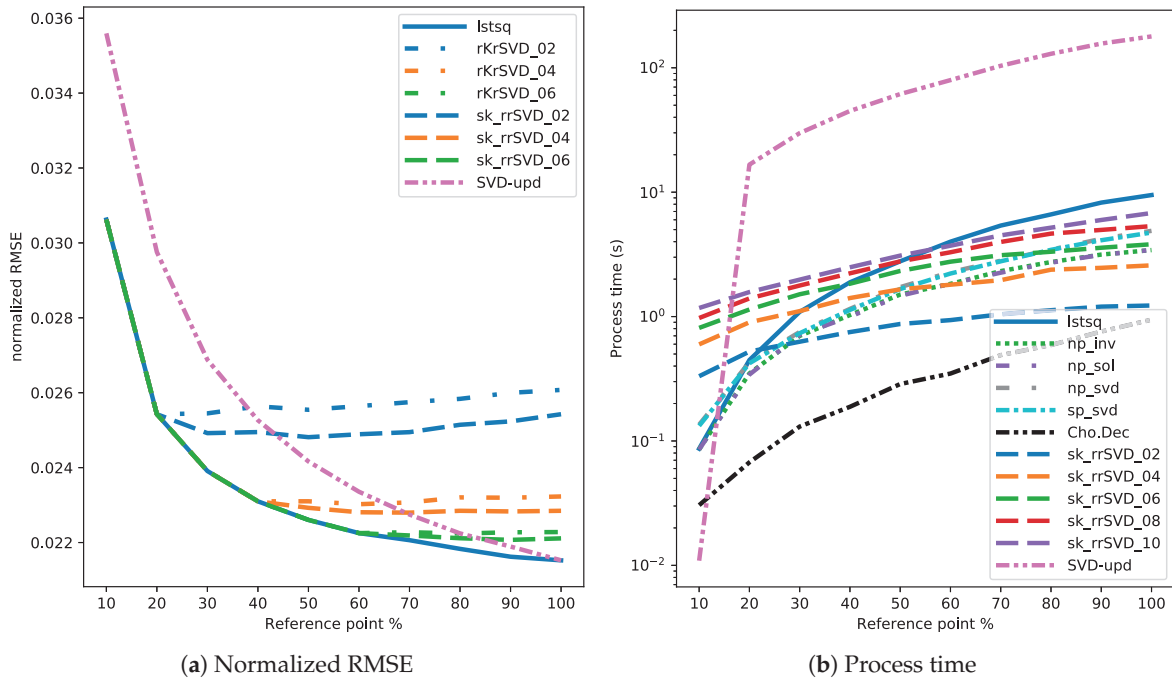
**Table 3.** RMSE for  $K = 70$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
Lstsq	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	6.28	2.17
Lstsq (nr)	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
np.inv	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	6.28	2.17
np.inv (nr)	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
np.solve	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	6.28	2.17
np.solve (nr)	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
np.svd	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	6.28	2.17
sp.svd	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
Cho.Dec	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	6.28	2.17
rKrSVD02	<b>2.67</b>	2.33	<b>7.93</b>	1.38	<b>2.57</b>	1.47	<b>2.48</b>	8.17	6.21	2.12
rKrSVD04	2.72	2.27	7.1	1.36	<b>2.31</b>	1.48	2.45	8.27	6.24	2.18
rKrSVD06	2.75	2.37	6.86	1.36	2.23	1.46	2.44	8.39	6.27	2.18
rKrSVD08	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
rKrSVD10	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
skrSVD02	<b>2.65</b>	2.32	<b>7.8</b>	1.47	<b>2.49</b>	1.63	<b>2.46</b>	8.38	<b>6.18</b>	2.2
skrSVD04	2.72	2.35	6.98	1.38	<b>2.28</b>	1.46	2.44	8.3	6.24	2.2
skrSVD06	2.75	2.37	6.84	1.36	2.22	1.45	2.43	8.39	6.27	2.17
skrSVD08	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
skrSVD10	2.77	2.34	6.79	1.35	2.21	1.47	2.43	8.36	-	-
SVDU	2.74	2.36	7.07	1.47	<b>2.27</b>	1.62	-	-	-	-

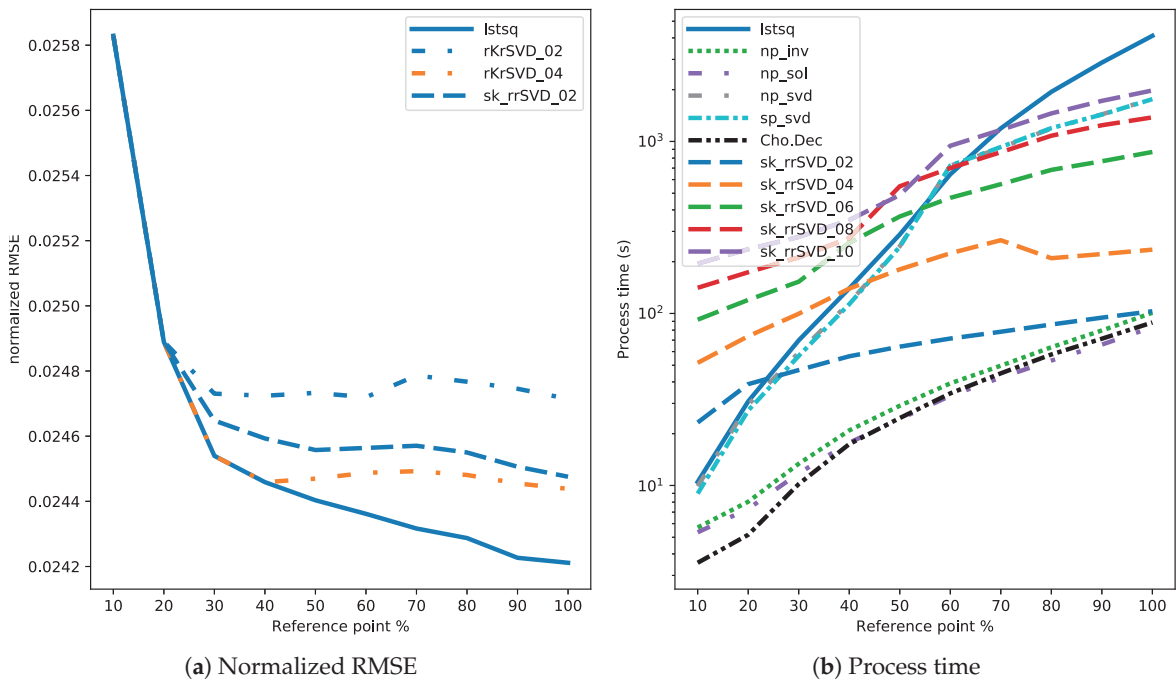
We show plots of the RMSE and process time behaviors as a function of reference point percentages for *Airplane Companies Stocks* in Figure 2, for *Computer Activity* in Figure 3 and for *Census* in Figure 4. In the figures we show the mean value of either RMSE or process time and plot only the lines that are either from *Lstsq* or differ statistically from it using the Kruskal–Wallis H test with 0.05 significance level. As is shown in Table 3, we also calculated the standard deviation for each datapoint and as stated previously, we presented similar tables corresponding reference point percentages [10–60%, 80–100%] for RMSE and [10%, 100%] for process time in Supplementary Material. Non-Tikhonov-regularized versions of methods are not shown in the figures in order to improve readability. Since the two random SVD methods have nearly identical performance, only the readily implemented version, *rank K random SVD* is shown in the figures.

Please note that the process time in Figure 2b is not truly comparable between *SVD update* and the other methods because of the iterative nature of the *SVD update*. The process time values for *SVD update* include the iterative update steps that are necessary for the method. As the other methods do not need iterative steps, they naturally have faster computation times in the type of point testing that we performed.

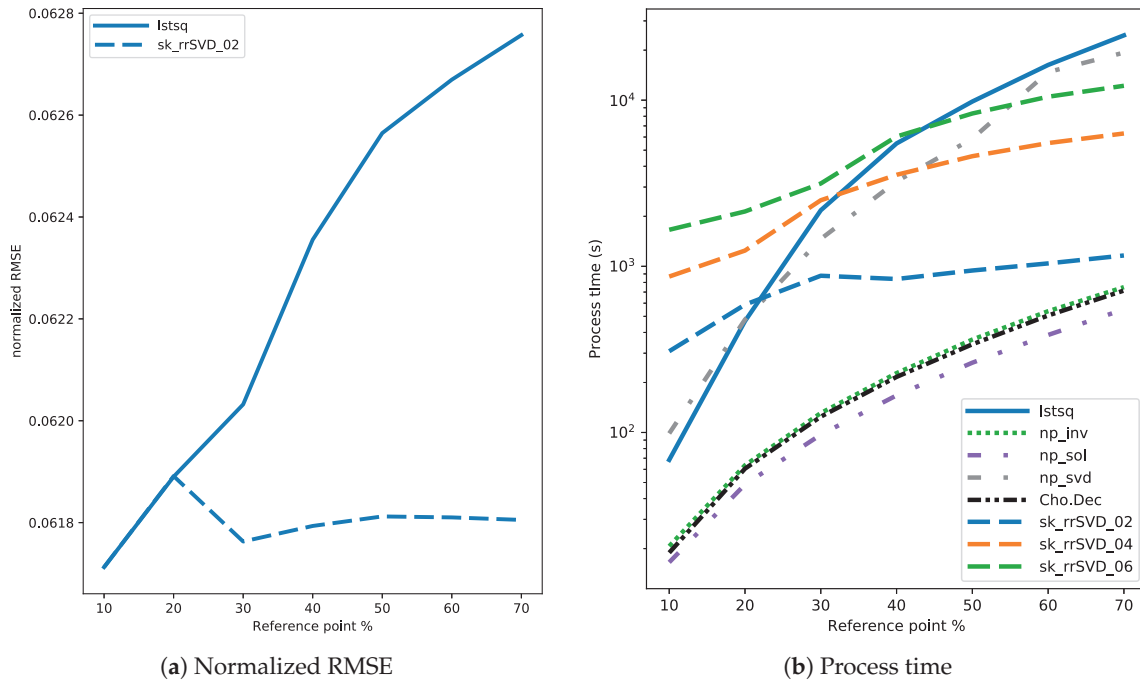
After our initial simulations we wanted to see an even lower rank reduction percentage. To that end, we chose to use *Cholesky decomposition* as the baseline method due to its performance in the initial simulations. We compared *sk\_rrSVD\_XX* with remaining rank percentages [1%, 10%] to the selected baseline with reference point percentages [10%, 100%]. As with previous simulations, we repeated the calculations 40 times in order to use statistical testing. The result for RMSE values and process times are shown in Figure 5 and the numerical values and standard deviations can be found in the Supplementary Material in Tables S20 and S21. Please note that unlike in the initial simulations, duplicates were removed from *Census* since the MLM requires all observations to be unique and *Census* as it contains a duplicate observation.



**Figure 2.** Normalized RMSE values (a) and process times (b) as a function of reference points for the *Airplane Companies Stocks* dataset. In (a), only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.



**Figure 3.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Computer Activity* dataset. In (a), only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.



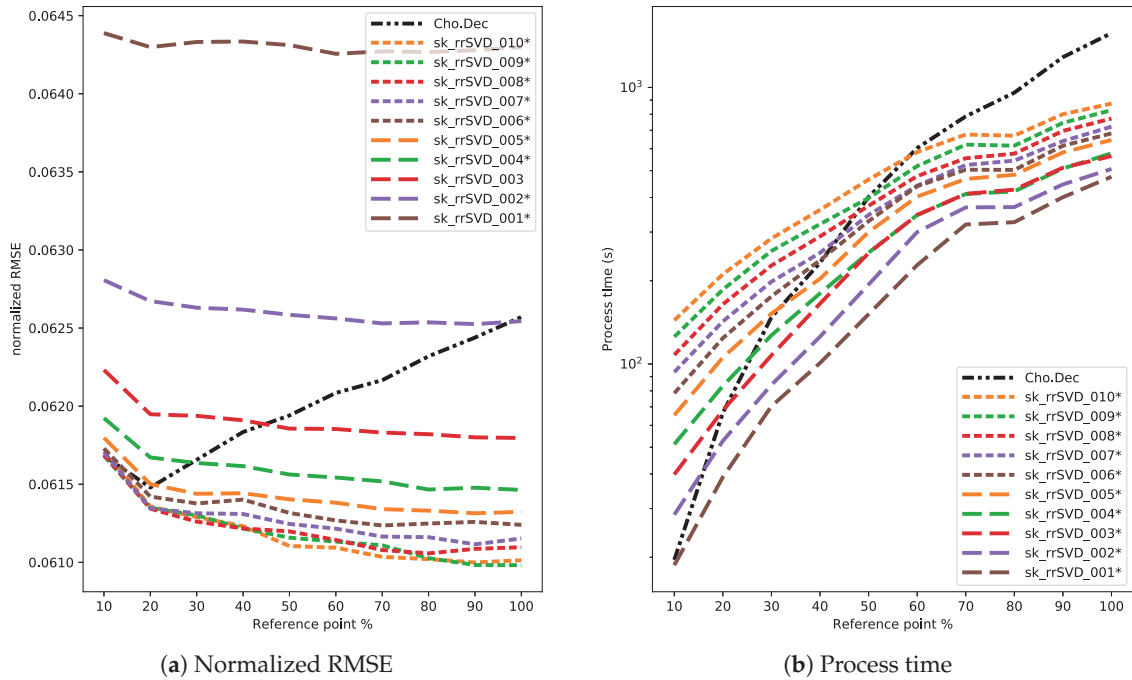
**Figure 4.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Census* dataset. In (a), only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.

After the results shown in Figure 5, we wished to see the behavior on an even larger dataset. For this, we chose *Mnist* since it is readily available and sufficiently larger than *Census*. *Mnist* is a classification dataset but here it is considered to be an integer regression dataset. *Cholesky decomposition* continued to function as a baseline method to the compared *sk\_rrSVD\_XX* at reference point percentages [10%, 100%]. We used remaining rank percentages [1%, 10%] with *sk\_rrSVD\_XX* and repeated each point calculation 40 times with randomized selection of training and testing sets. The result is shown in Figure 6 and the numerical values along with computed standard deviations are presented in Supplementary Material in Tables S22 and S23. Please note that *Mnist* had duplicate removal as a preprocessing step unlike the datasets in the initial simulations.

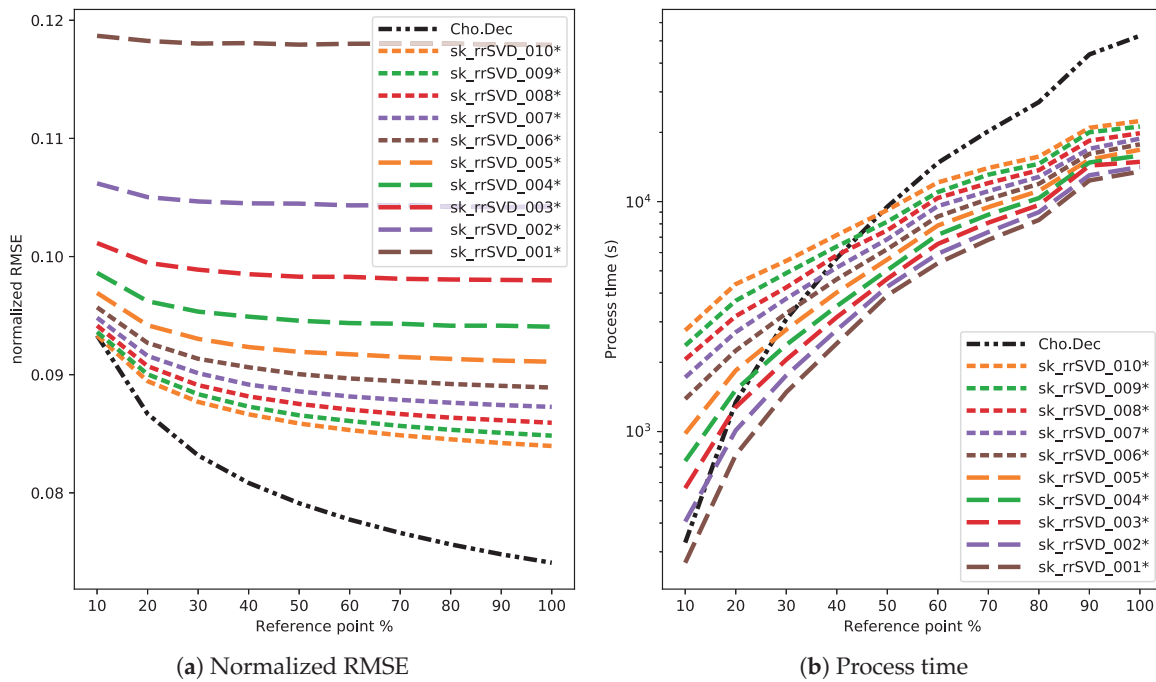
We opted to include result figures with the FNNs in Appendix A for datasets *Breast Cancer* (Figure A1), *Boston Housing* (Figure A2), *Airplane Companies Stocks* (Figure A3), *Census* (Figure A5) and *Mnist* (Figure A6). The raw data is provided in the supplementary as Tables S24 and S25. The figures in Appendix A contain the calculated results for the FNNs and the corresponding results for the MLM. Figures A1 and A2 are the normalized RMSE and process time used for the MLM and the FNNs for datasets *Breast Cancer* and *Boston Housing*. Figure 2 corresponds to Figure A3, where the difference is the addition of the FNN results. Similarly, Figures 3, 5 and 6 respectively correspond to Figures A4, A5 and A6. The figures presented in Appendix A have the item “FNN-4” representing the results relating to the deep neural network, or the neural network with four layers and “FNN-2” representing the one with two layers. For the figures with RMSE values, we also included the standard deviation and represented it as a see-through filled area.

Results for the dataset *Au<sub>38Q</sub>* are shown in Appendix B in Figures A7 and A8. The raw data is provided in the supplementary as Tables S26–S45. It is important to note that in the name of readability, we included another version of Figure A7 in Figure A9. The difference being the removed plot of *FNN-2* due to its behaviour in the 4000 feature column. In addition, Figures A8 and A9 also have equalized Y-axes to better show the behaviour of MLM.





**Figure 5.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Census* dataset. Variants of *sk\_rrSVD\_XX* that differ statistically from *Cholesky decomposition* are marked with \*.



**Figure 6.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Mnist* dataset. Variants of *sk\_rrSVD\_XX* that differ statistically from *Cholesky decomposition* are marked with \*.

#### 4. Discussion

Various methods to recover the weight matrix in the distance-regression and their effects to the accuracy of the whole MLM were studied with fifteen different instances of seven datasets. In addition to this, we also compared a smaller subset of MLM methods with two feedforward neural network

configurations. We will first provide discussion related to the direct and randomized solvers and then discuss the comparison between MLM and FNNs. We wished also to see the performance of the novel incremental SVD update algorithm *SVDU* but the computational resources it required forced us to stop using it on bigger datasets than *Airplane Companies Stocks* (see Figure 2b).

#### 4.1. Comparison of Direct and Randomized Solvers for MLM

In the experiments, we wanted to see if the industry standard solver *Lstsq* could be improved upon by randomization. It was noted that the readily implemented solvers generally had the same accuracy with minor differences in progress times. Considering this, we do our comparisons of *random SVD* to either *Lstsq* due to its status as the industry standard or to *Cholesky decomposition* due to it generally outperforming all other readily implemented methods. We presented summary of the comparison in Table 4. In short, Table 4 tells us that randomized solvers beat the accuracy of *Cholesky decomposition* in two datasets and that the training time with randomized solvers is faster in 11 datasets. Especially in the largest two and in the *Au<sub>38Q</sub>* dataset. We can summarize Figures 2a, 3a and 4a by noting that readily implemented methods match each other in the accuracy they provide to the MLM and that randomized solvers start falling behind at 30% reference points and higher. It is to be expected that the introduction of randomness in the solver causes the accuracy of the MLM to change relative to the amount of introduced randomness. The differences between the methods are mostly found in the process time Figures 2b, 3b and 4b. Based on the process time behavior in the figures, it is clear that a specialized solver such as *Cholesky decomposition* is superior to *Lstsq* in the MLM context. This led us to leave *Lstsq* out of further experiments and use *Cholesky decomposition* as the default method in this comparison. The results also show the expected behaviour that the degree of randomness in the *random SVD* solver has a direct effect on the process time taken.

The process time figures gave us the idea to use even lower remaining rank percentages with the *random SVD* and to use it on a larger dataset. This led us to use remaining rank percentages [1%, 2%, . . . , 10%] which were reported in Figures 5 and 6. Figures 5a and 6a, in conjunction with Figures 2a, 3a and 4a, show that the *random SVD* has a tendency to plateau around 30–50% reference point mark. This is to say that the *random SVD* does not provide the MLM with meaningful increases in accuracy at higher reference point percentages. The same 30–50% reference point mark is the area where low remaining rank percentage *random SVD* overtakes *Cholesky decomposition* in process time as is seen in Figures 5b and 6b. We do not know why in Figure 5a the *Cholesky decomposition* shows a non-improving accuracy score.

**Table 4.** Summary of comparison between *Cholesky decomposition* and randomized solvers for the MLM. Symbols + and – indicate that the randomized solver is better/worse than the *Cholesky decomposition*.

MLM ( <i>Cholesky Decomposition</i> )	Accuracy	Training Time
Dataset	rnd. Solver	rnd. Solver
BreastCancer	+	–
BostonHousing	–	–
AirplaneCompanies	–	–
ComputerActivity	–	–
Census	+	+
Mnist	–	+
AuN2-4k	–	+
AuN2-8k	–	+
AuN2-12k	–	+
AuN10-4k	–	+
AuN10-8k	–	+
AuN10-12k	–	+
AuN100-4k	–	+
AuN100-8k	–	+
AuN100-12k	–	+

The results of the MLM in the case of the  $Au_{38Q}$  dataset are shown in Figures A7, A8 and A9. Figure A8 depicts the process times and Figure A9 the accuracy most clearly. An immediate observation from Figure A9 is that the subfigure columns for 400 features and 4000 features appear identical.

Figure A10 shows that the outlook of the dataset has not changed in any meaningful way when the number of features increases from 400 to 4000. The reason for that is found in the original nanocluster trajectory data for  $Au_{38Q}$  where the entire dataset is composed of nanocluster configurations where each adjacent nanocluster differs from each other in miniscule fashion. This is to say that the nanocluster configurations in  $Au_{38Q}$  are similar enough to each other that 4000 features do not provide additional information that could be used when compared to 400 features. Should there be more variance in the dataset (such as nanoclusters from another isomer) we would expect to see 4000 features provide information that 400 features cannot provide. This is at the same time a good argument for feature selection. Regarding *random SVD* and *Cholesky decomposition*, we are comfortable in noting that *Cholesky decomposition* is a clear winner in the accuracy it provides to the MLM with the  $Au_{38Q}$  dataset. Although we do suspect that the result would not be so cut an dry with a dataset of more variance. Considering the properties of the MLM in the case of  $Au_{38Q}$  in Figure A7 we can say the following: The increasing number of observations improves the accuracy of the MLM model and the increasing number of features do not improve the result if the increased number of features do not provide meaningful information. Also, the increasing number of features does not make the result worse either. Process time wise (Figure A8), the MLM behaved as expected. The process time taken increased as the number of observations increased and it changed only slightly by the changes in the number of features.

#### 4.2. Comparison of MLM to Shallow and Deep FNNs

We begin the discussion about the FNN results with a short summary of them. In general, the deep FNN architecture, with higher computational costs, provided better accuracy than the shallow one. However, the variation of accuracy for both FNN models was rather extensive in the *Breast Cancer*, *Boston Housing*, *Computer Activity* and *Census* datasets. One could expect that the larger network *FNN-4* would have higher accuracy than the *FNN-2*, with longer training times as a drawback. One could also expect that the *FNN-2* models for datasets with low number of features could be competitive against the more complex *FNN-4*. Both assumptions are backed up by our results, although the process time for *Mnist* in Figure A6b was smaller for the *FNN-4* compared to *FNN-2*.

The results for the MLM and FNN comparison are summarized in Table 5. In terms of the overall performance, i.e., by considering both accuracy and training time, MLM with *Cholesky decomposition* was better than FNN in nine of fifteen (60%) of the cases. The training times were on a same level only with *Mnist*, otherwise MLM training was more efficient with a clear margin, altogether in 93% (14/15) of the cases. The *FNN-4* model's accuracy was comparable to the MLM in five cases, and better than MLM only for the *Mnist* dataset. This outcome is probably due to the form of *Mnist* data, which is given by small  $28 \times 28$  images whose processing through multiple higher-level feature generating layers is known to be advantageous, especially, compared to a direct distance computation between the gray scale values as with the MLM.

Regarding FNN and the  $Au_{38Q}$  dataset, one can immediately see that *FNN-2* is not competitive in a situation with 4000 features (Figure A7). This was fully expected, as *FNN-2* was not meant for datasets with high number of features (even if it adapts to the number of features in the dataset). In addition, *FNN-2* can at best only match *FNN-4* with the  $Au_{38Q}$  dataset. Observing *FNN-4* in Figure A9 reveals a trend. At first, the increasing number of features to 400 is beneficiary for both MLM and *FNN-4*. However, a further increase in the number of features to 4000 causes *FNN-4* to lose accuracy while MLM stays the same. In addition, the increase in the number of observations causes *FNN-4* to gain accuracy with 80 and 400 features whereas it loses accuracy in the case of 4000 features. This indicates that *FNN-4* cannot naturally handle an increased number of either observations or features that do not provide new information. Something that MLM shows itself to be capable of.

**Table 5.** Summary of comparison of MLM with *Cholesky decomposition* to a shallow (FNN-2) and a deep (FNN-4) neural network results. Symbols +,  $\approx$ , and – indicate that the FNN results are better, no different, or worse than the MLM results, respectively. Symbol / and the symbols with it are used to signify that the FNN results are not properly in either category.

MLM ( <i>Cholesky Decomposition</i> ) Dataset	Accuracy		Training Time	
	FNN-4	FNN-2	FNN-4	FNN-2
BreastCancer	$\approx$	$\approx$	–	–
BostonHousing	–	–	–	–
AirplaneCompanies	–	–	–	–
ComputerActivity	$\approx$	–	–	–
Census	–	–	–	–
Mnist	+	$\approx$	– / $\approx$	– / $\approx$
AuN2-4k	–	–	–	–
AuN2-8k	–	–	–	–
AuN2-12k	–	–	–	–
AuN10-4k	– / $\approx$	– / $\approx$	–	–
AuN10-8k	– / $\approx$	– / $\approx$	–	–
AuN10-12k	– / $\approx$	–	–	–
AuN100-4k	–	–	–	–
AuN100-8k	–	–	–	–
AuN100-12k	–	–	–	–

In order to improve the FNN models' accuracies, we would need to manually tailor the models for each dataset. This brings an important point in the comparison of the MLM to FNN models. The MLM has a clear advantage compared to the deep FNN models. Achieving a better result with the MLM can be conducted straightforwardly since MLM has a only one hyperparameter. Increasing the number of reference points increases the accuracy of the regression model with a cost of higher training time. In contrast, the FNN models are known to require careful hyperparameter selection and a well-designed structuring of the hidden layers. Zhou and Feng [41] argue that the training of deep neural network models resembles more an art than science or engineering.

## 5. Conclusions and Future Work

Summarizing our work, we showed results for the accuracy of the MLM and the process time taken to solve Equation (1) for randomized SVD algorithms with varying level of randomness and for direct solvers that are well-known to research community. In addition to this, we compared the accuracy of the MLM to deep and shallow FNN models by means of the number of observations and the number of features. To begin, we started with *Lstsq* as the standard and ended up switching it to *Cholesky decomposition* as it clearly outperformed *Lstsq* as well as other tested direct solvers. The only datasets where *Cholesky decomposition* was not the clear winner in accuracy provided to the MLM were *Breast Cancer* (see Tables 3, Tables S5–S9) and *Census* (see Figure 5a and Table S20). From there we can make the conclusion that if the accuracy of the MLM is the only objective, then randomized solvers do not provide an advantage if the dataset is preprocessed so that there are no duplicates. The randomized solvers come in to play if the training speed is an issue and one is handling a large number of observations. The *random SVD* follows the general principle that the higher the randomness the lower the accuracy and faster the training speed. However, in the context of the MLM the reduction of accuracy in the solver does not translate to as great of a loss in model accuracy. Meaning that one might expect to have results roughly somewhere in the ballpark if 99% of the information is replaced with randomness but in practice the results with *random SVD* at low remaining rank percentage are in the same order of magnitude. We can conclude that a researcher may choose to sacrifice some accuracy and gain speed in the training phase by choosing a randomized solver with a low remaining rank percentage.

As a whole, we demonstrated that:

1. If accuracy is the only objective in a prepared situation, one should choose to use a direct solver with the MLM.
2. If not, randomized solvers provide comparable performance to direct solvers while providing a speedup.
3. Utilization of linear training technique with a distance-based kernel is computational more efficient than nonlinear optimizers needed for the FNN models. In general, the MLM outperformed the shallow and deep FNNs with fixed architecture, with one exception. Especially, in cases of thousands of features, as with the experiments for the  $Au_{38Q}$  dataset here, the MLM is the recommended technique from both training time and model accuracy perspectives.

Therefore, the use of randomized algorithms appear to be a promising avenue for improving the efficiency of the MLM and other distance-based regression techniques. In addition to this, we believe that our comparison of the MLM with currently popular neural network techniques showed that the MLM is a viable option as the general method for nonlinear regression problems.

While this work focused on randomized algorithms, we find it an important topic of future work to expand the comparison also to iterative solution methods such as the Chebyshev method [42] and the block-iterative methods [43]. Especially the classical block Jacobi method with successive overrelaxation can be easily parallelized using data parallelism. It would also be interesting to see the performance of the MLM when it is set to use a GPU in the calculations.

**Supplementary Materials:** The following are available online at <http://www.mdpi.com/2504-4990/2/4/29/s1>, Table S1: RMSE for  $K = 10$ , Table S2: RMSE for  $K = 20$ , Table S3: RMSE for  $K = 30$ , Table S4: RMSE for  $K = 40$ , Table S5: RMSE for  $K = 50$ , Table S6: RMSE for  $K = 60$ , Table S7: RMSE for  $K = 80$ , Table S8: RMSE for  $K = 90$ , Table S9: RMSE for  $K = 100$ , Table S10: Process time (s) for  $K = 10$ , Table S11: Process time (s) for  $K = 20$ , Table S12: Process time (s) for  $K = 30$ , Table S13: Process time (s) for  $K = 40$ , Table S14: Process time (s) for  $K = 50$ , Table S15: Process time (s) for  $K = 60$ , Table S16: Process time (s) for  $K = 70$ , Table S17: Process time (s) for  $K = 80$ , Table S18: Process time (s) for  $K = 90$ , Table S19: Process time (s) for  $K = 100$ , Table S20: RMSE for *Census* at reference point percentages [10%, 100%], Table S21: Process time in seconds for *Census* at reference point percentages [10%, 100%], Table S22: RMSE for *Mnist* at reference point percentages [10%, 100%], Table S23: Process time in seconds for *Mnist* at reference point percentages [10%, 100%], Table S24: RMSE for neural networks, Table S25: Process time (s) for neural networks, Table S26: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 10$ , Table S27: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 20$ , Table S28: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 30$ , Table S29: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 40$ , Table S30: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 50$ , Table S31: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 60$ , Table S32: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 70$ , Table S33: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 80$ , Table S34: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 90$ , Table S35: Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 100$ , Table S36: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 10$ , Table S37: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 20$ , Table S38: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 30$ , Table S39: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 40$ , Table S40: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 50$ , Table S41: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 60$ , Table S42: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 70$ , Table S43: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 80$ , Table S44: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 90$ , Table S45: Process time (s) mean, standard deviation and best result for  $Au_{38Q}$   $K = 100$ , Source codes for the Equation (1), source codes for the FNN models and source code for the MLM. The  $Au_{38Q}$  dataset is available at [10.5281/zenodo.4268064](https://zenodo.org/record/4268064).

**Author Contributions:** Conceptualization, J.L., J.H., P.N., T.K.; methodology, J.L.; software, J.L. and J.H.; formal analysis, J.L.; investigation, J.L.; data curation, J.L.; writing—original draft preparation, J.L. and J.H.; writing—review and editing, J.L., J.H., P.N., T.K.; visualization, J.L., J.H., P.N., T.K.; supervision, P.N. and T.K.; project administration, T.K.; funding acquisition, T.K. All authors have read and agreed to the published version of the manuscript.

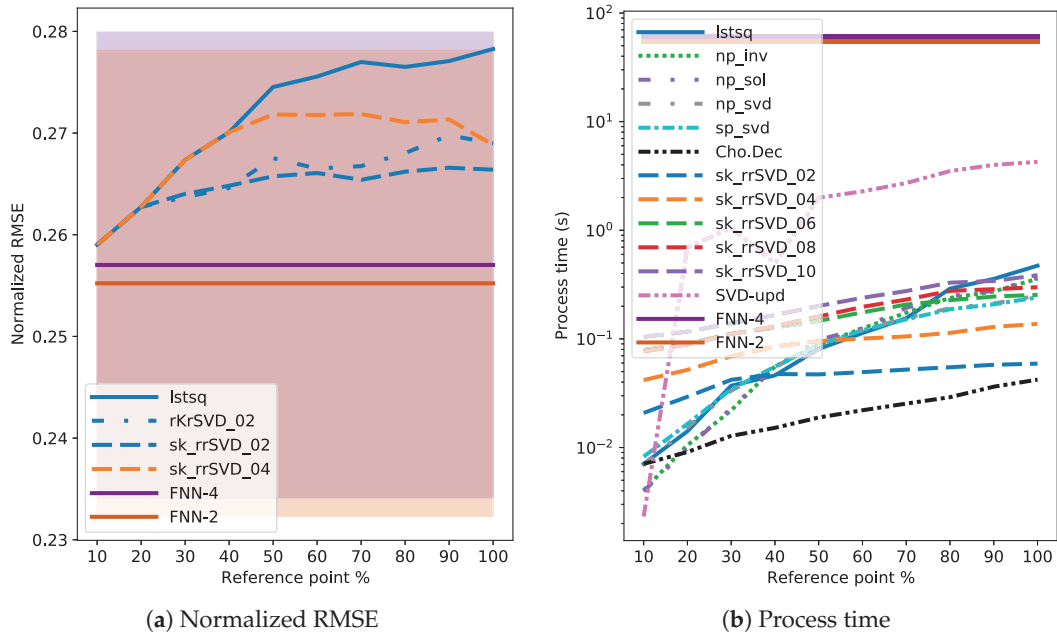
**Funding:** This work was supported by the Academy of Finland from the projects 311877 (Demo) and 315550 (HNP-AI).

**Acknowledgments:** We acknowledge grants of computer capacity from the Finnish Grid and Cloud Infrastructure (persistent identifier urn:nbn:fi:research-infras-2016072533).

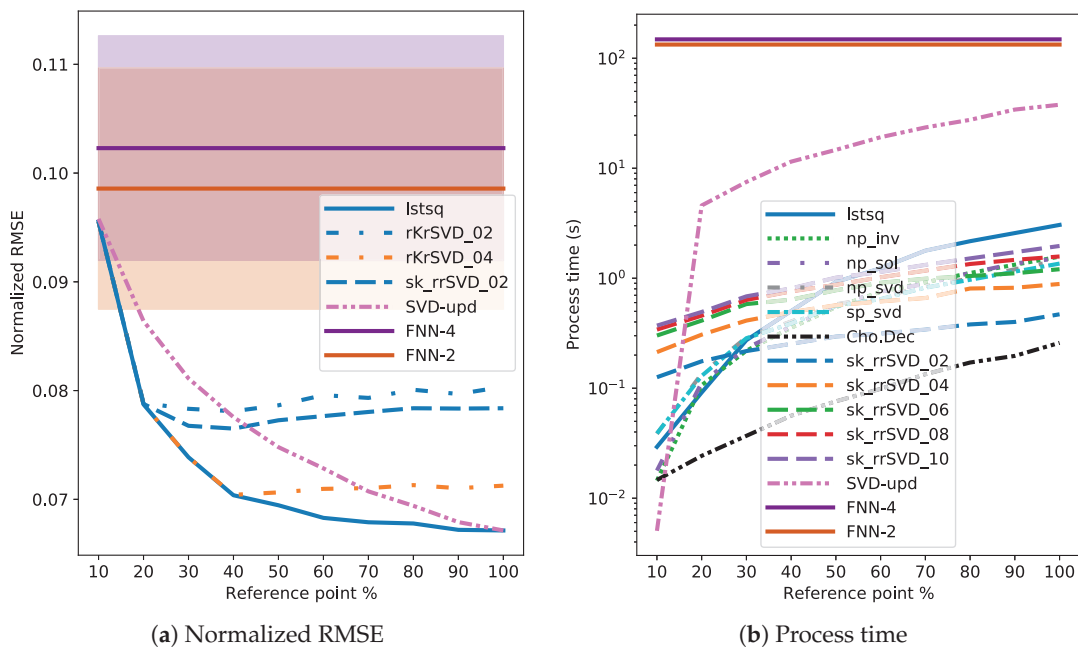


**Conflicts of Interest:** The authors declare no conflict of interest.

**Appendix A. MLM vs. FNN**

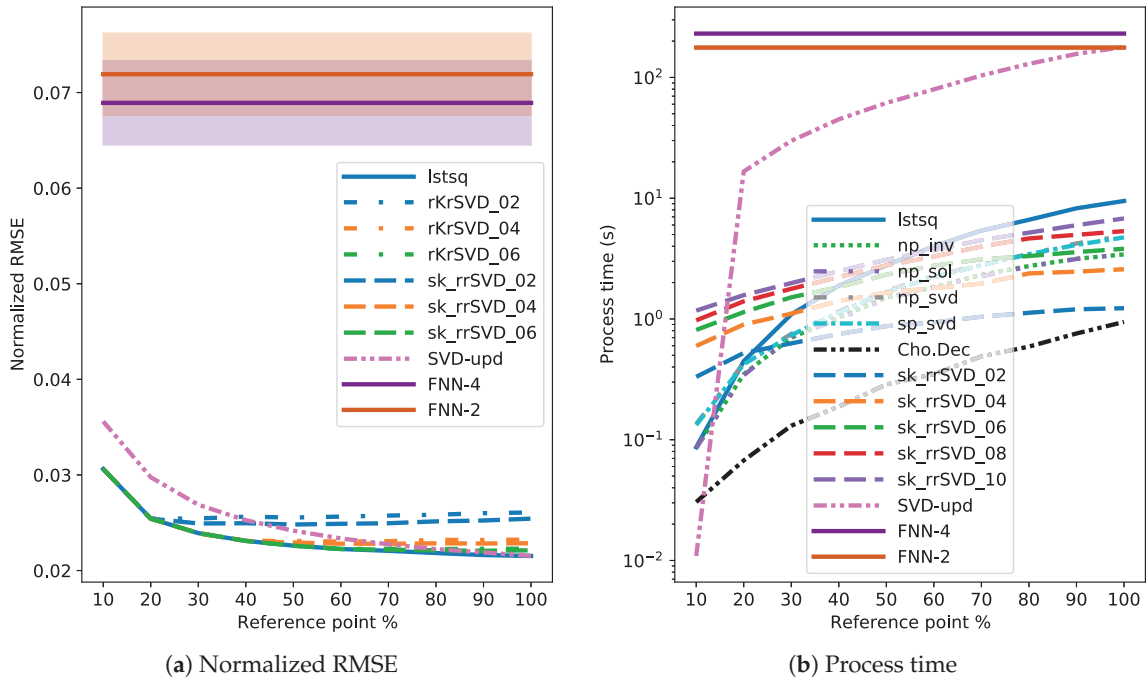


**Figure A1.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Breast Cancer* dataset. In (a) for MLM, only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.

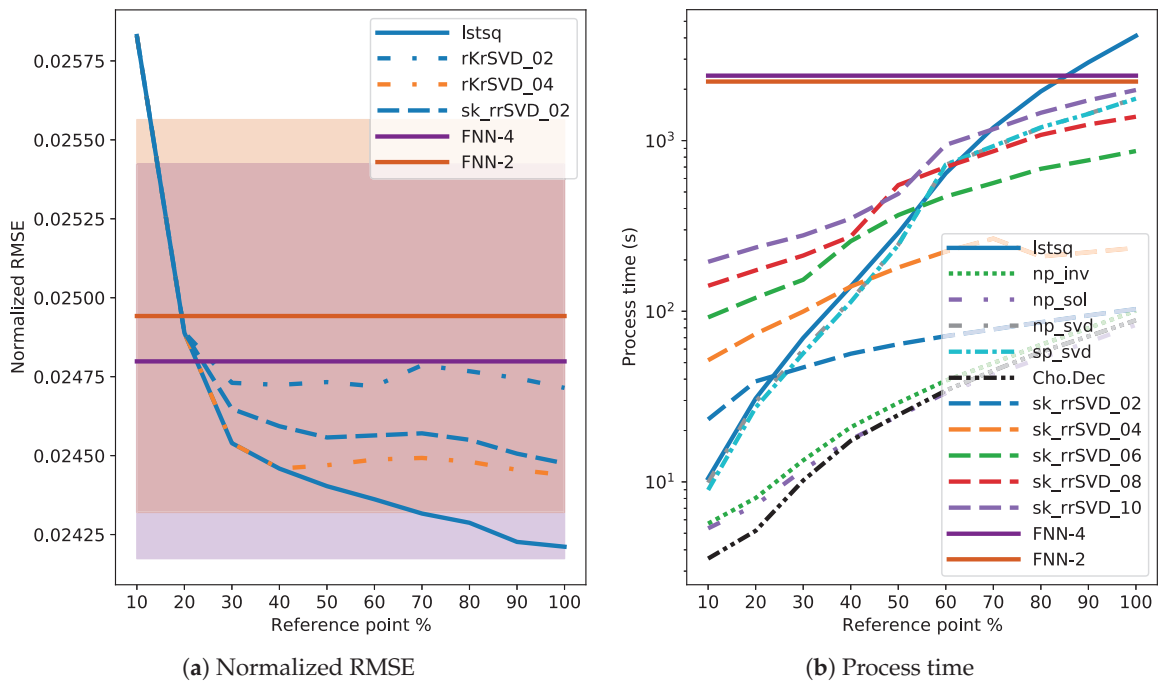


**Figure A2.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Boston Housing* dataset. In (a) for MLM, only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.

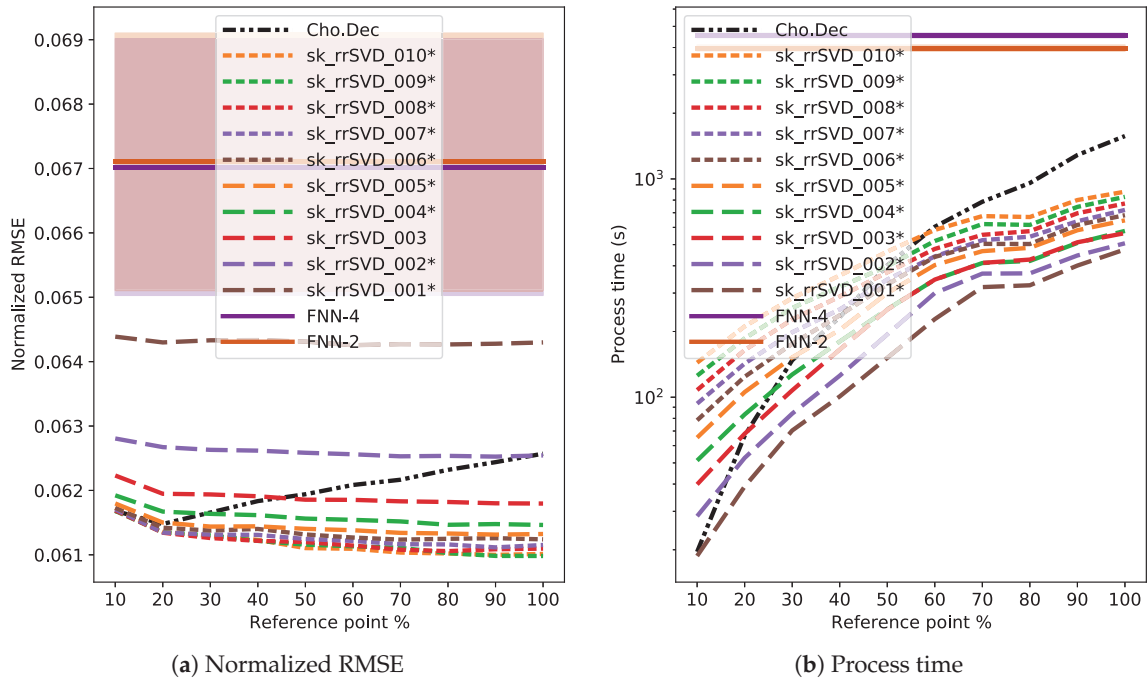




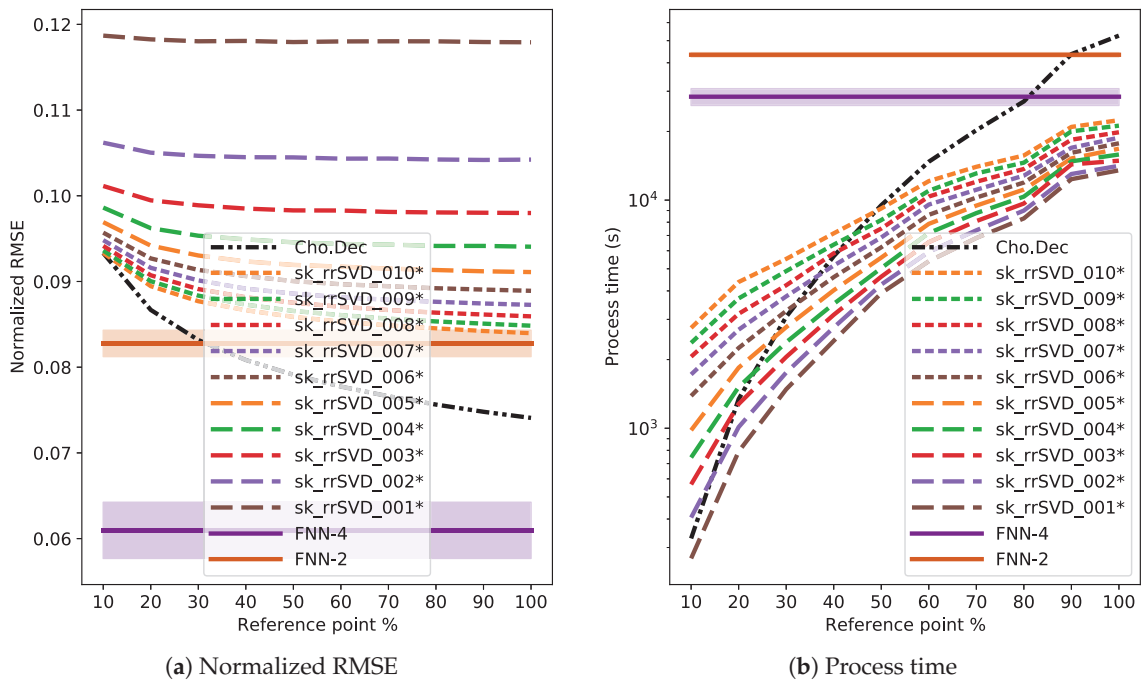
**Figure A3.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Airplane Companies Stocks* dataset. In (a) for MLM, only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.



**Figure A4.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Computer Activity* dataset. In (a) for MLM, only methods that at any reference point percentage differ from *Lstsq* with Kruskal–Wallis H test are shown along *Lstsq*.

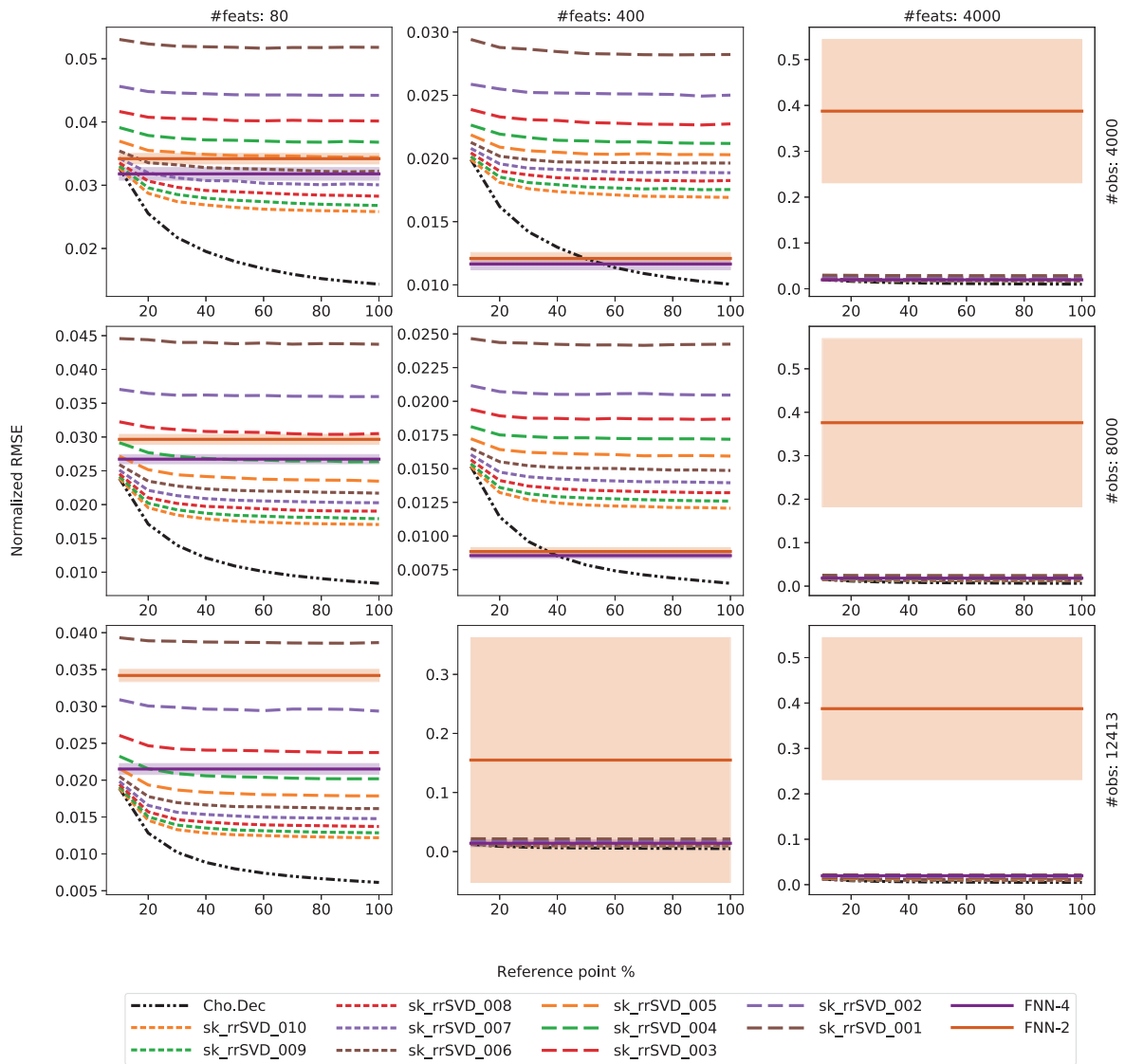


**Figure A5.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Census* dataset. For MLM, variants of *sk\_rrSVD\_XX* that differ statistically from *Cholesky decomposition* are marked with \*.

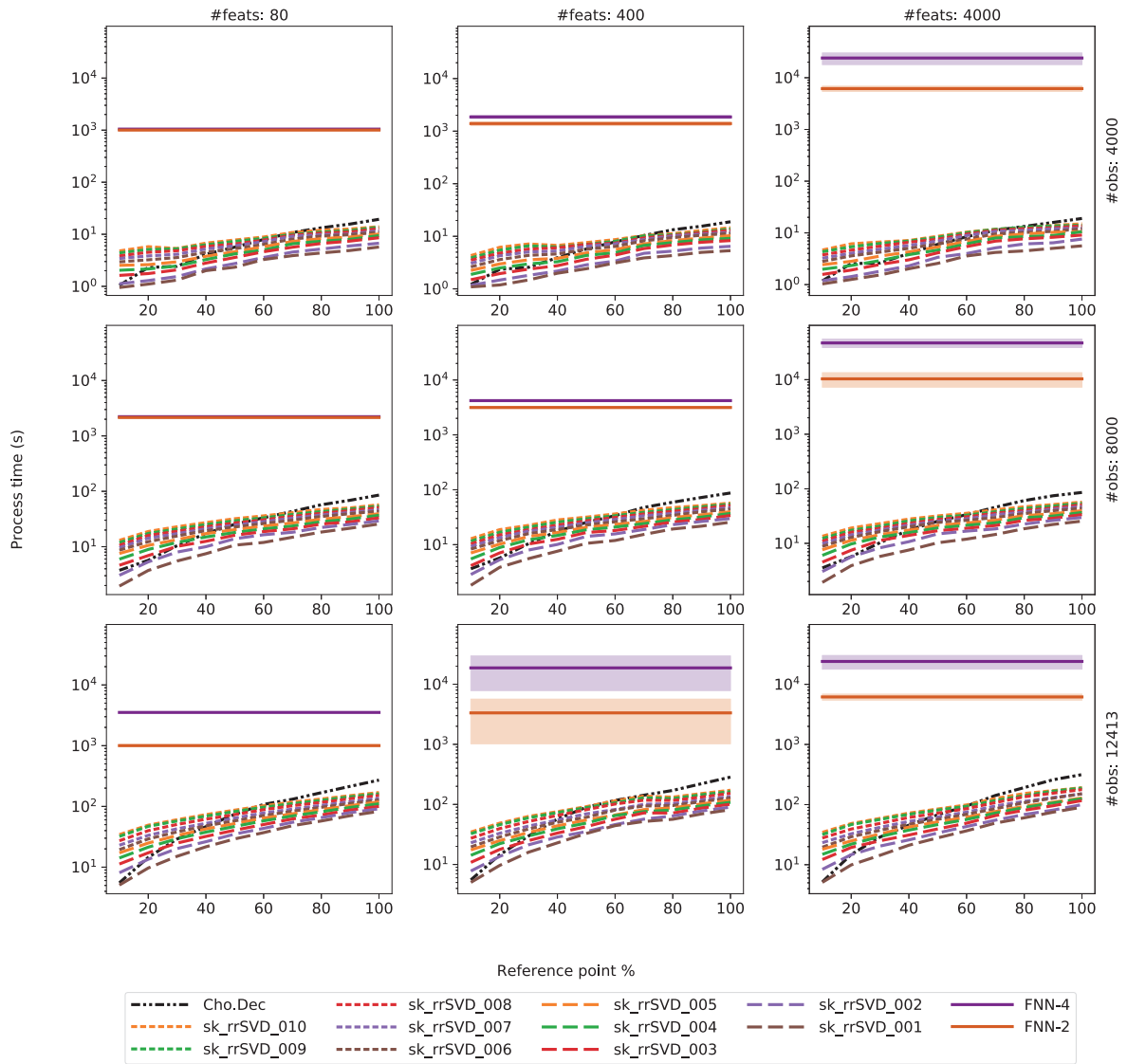


**Figure A6.** Normalized RMSE values (a) and process time in seconds taken to compute Equation (1) (b) as a function of reference point percentage for the *Mnist* dataset. For MLM, variants of *sk\_rrSVD\_XX* that differ statistically from *Cholesky decomposition* are marked with \*.

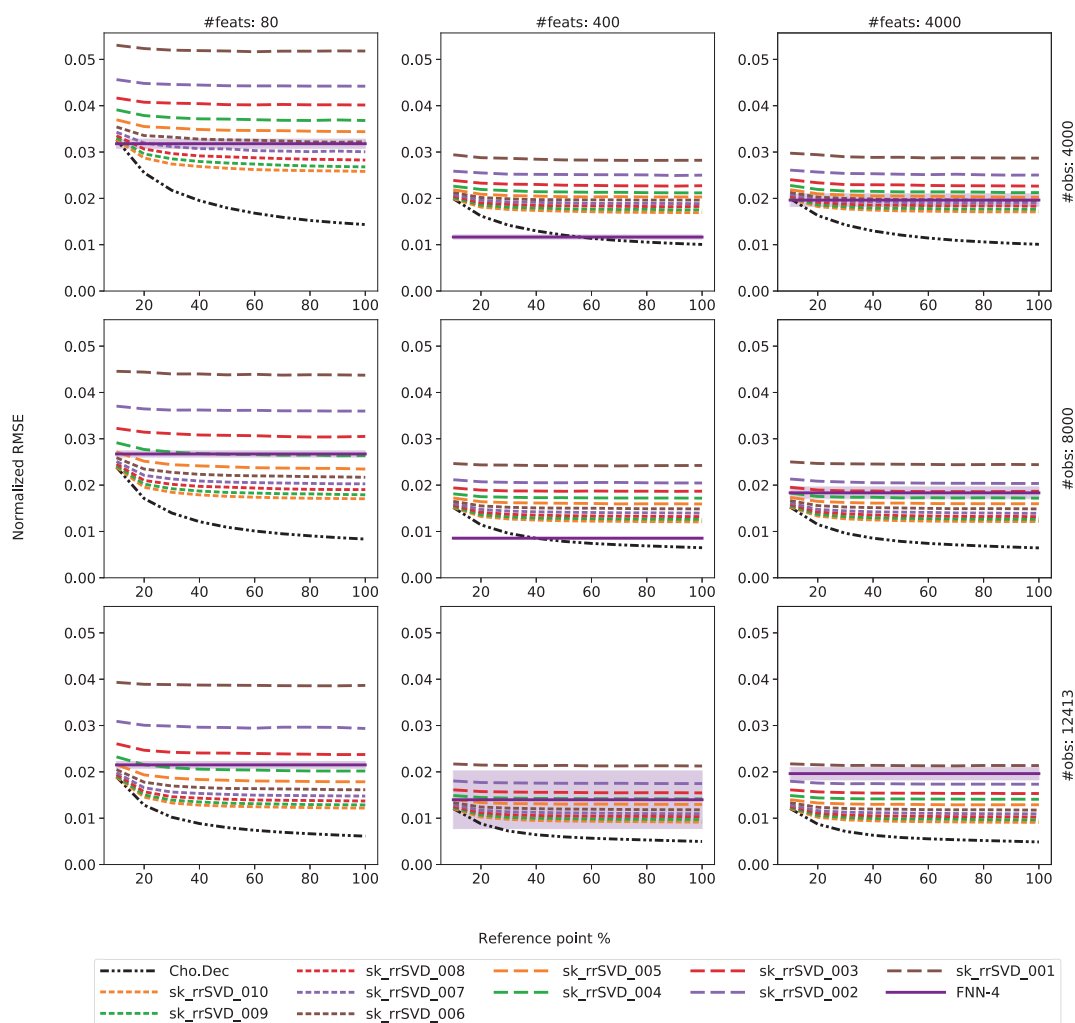
Appendix B. Au<sub>38Q</sub>



**Figure A7.** Normalized RMSE for Au<sub>38Q</sub>. Each subfigure has reference point percentage on the horizontal axis and normalized RMSE on the vertical axis. The top row corresponds to dataset variants with 4000 observations, middle row represents variants with 8000 observations and the bottom row is for the full dataset. The left column represents dataset variants with 80 features, middle represents 400 features and the right column is for the full feature set. Please note that the Y-axis is different in each column.



**Figure A8.** Process time for  $Au_{38Q}$ . Each subfigure has reference point percentage on the horizontal axis and normalized RMSE on the vertical axis. The top row corresponds to dataset variants with 4000 observations, middle row represents variants with 8000 observations and the bottom row is for the full dataset. The left column represents dataset variants with 80 features, middle represents 400 features and the right column is for the full feature set. Please note that the Y-axis is equalized in the subfigures.



**Figure A9.** Same as Figure A7 but, for readability, without FNN-2. In addition, the Y-axis scale is same in each subfigure.

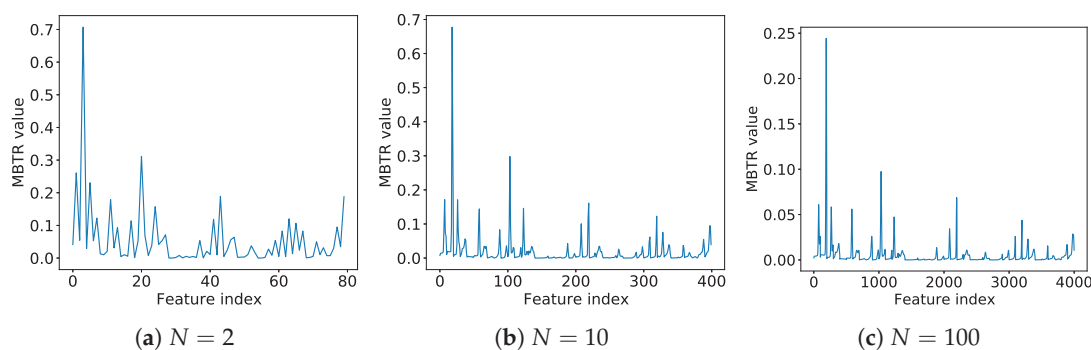
### Appendix C. Au<sub>38</sub>Q MBTR-K3 Dataset and Its Variants

The Au<sub>38</sub>Q dataset has its roots in a hybrid nanostructure reported by Qian et al. [44]. The Au<sub>38</sub>Q is a monolayer ligand protected nanoparticle, or a hybrid nanoparticle that is from here on referred to as a nanostructure (referring more to the physical structure than to the scientific concept, also to avoid confusion with the machine learning meaning of a cluster). The nanostructure was studied with molecular dynamics simulations under heating by Juarez-Mosqueda et al. [33] which produced a trajectory file (a data format containing one or more nanostructure configurations and metadata) for Au<sub>38</sub>Q that contained 12,413 nanostructures formed during a molecular dynamics simulation from a start point to an end point. Each nanostructure in the trajectory is akin to a log of a time step during the molecular dynamics simulation. Each nanostructure is also paired with a value called potential energy, which can be used to describe the stability of a nanostructure. The potential energy was calculated by Juarez-Mosqueda et al. [33] using a density functional theory (DFT) calculator GPAW. However, a raw nanostructure configuration (the cartesian coordinates of the atoms) is problematic as it is for machine learning since there is no way to deterministically order the atoms so that the order in which they are given has no effect on the machine learning model. A set of cartesian coordinates are also not translationally and rotationally invariant, which is a major issue since the orientation and the exact location of a nanostructure are not allowed to have an effect. To solve this problem one has to use a concept called a descriptor. A descriptor is very literal in its function. We opted to use one called many

body tensor representation (MBTR) [45] which combines several descriptors to form its own. The use of MBTR in the context of nanostructures is also discussed in [3,45]. In our case, the description of a nanostructure is a vector, computed using a package known as Dscribe [46] (version 0.4.0). In short, the input values in our  $Au_{38Q}$  datasets are the description vectors and the output values are the potential energy values of each nanostructure.

We opted to use a setting denoted as  $K = 3$  which computes a distribution of angles between groups of three elements and then concatenates those distributions to form a description vector. The  $Au_{38Q}$  is formed of four elements, hydrogen (H), carbon (C), sulfur (S) and gold (Au). The name  $Au_{38Q}$  refers to the nanostructure having 38 gold atoms as its core. For example, one of the distributions is formed from H-Au-C, meaning that the angle between vectors formed from H-Au and Au-C are computed. Repeated angles are of no interest, in our case there are 40 different three element angle groups [H-H-H, H-H-C, ..., Au-S-Au, Au-Au-Au] that are calculated to form the MBTR  $K = 3$  description vector. We used mainly the default settings of the MBTR in Dscribe: l2 normalization (the length of a description vector was normalized to 1), cosine was used as the angle calculation function, measurement grid was set as  $[-1, 1]$  and  $\sigma$  was 0.1. We used exponential weighting function with a scale of 0.5 and a cutoff of  $1 \times 10^{-3}$ . As we were describing a single nanostructure in a vacuum at a time, the descriptor is not periodic.

We mentioned that there are variants of  $Au_{38Q}$  dataset. In total we used nine variants by having two parameters with three values in each. One of the varied parameters was the number of features through the grid accuracy parameter  $N$  for the MBTR, which defines the number of points in a single angle distribution component. In other words, a grid accuracy parameter  $N = 100$  results in 40 vectors of 100 length to form a description vector with 4000 features. The grid accuracy parameters that we used were  $N \in [2, 10, 100]$ . The number two is the lowest that can be used since a single number is not enough to describe a distribution. The other varied parameter was the number of observations. The full variant is denoted as 12 k, even though it uses the full 12,413 observations. Other two variants are 4 k and 8 k, which use 4000 observations and 8000 observations, respectively. The subsets of observations are taken from the full set with the use of the reference point selection algorithm RS-maximin. Since RS-maximin is deterministic, the first 4000 observations in 8 k are the same as all the observations in the 4 k subset. We denoted the variants of  $Au_{38Q}$  as  $AuNx-yk$  in the tables, where  $x \in [2, 10, 100]$  and  $y \in [4, 8, 12]$ . As  $Au_{38Q}$  is not a commonly used dataset we included a figure showing the mean of all 12,413 MBTR description vectors with the three grid accuracy values in Figure A10 to aid the reader in understanding the way that a description vector might look like. The data files are available at [10.5281/zenodo.4268064](https://doi.org/10.5281/zenodo.4268064).



**Figure A10.** Means of 12,413 MBTR-K3 descriptors for grid accuracy  $N$  values 2 (a), 10 (b) and 100 (c). Features represent  $N$  consecutive numbers representing the angle distribution measured between groups of three atoms. The nanostructure  $Au_{38Q}$  is composed of four elements, hydrogen (H), carbon (C), sulfur (S) and gold (Au). The first  $N$  features are H-H-H angles, second  $N$  are H-H-C and so on. Ending with Au-Au-Au angle. There are no repeated angles and so there are a total of 40 distributions laid consecutively to form the resulting MBTR-K3 vector.



## References

1. De Souza Junior, A.H.; Corona, F.; Miche, Y.; Lendasse, A.; Barreto, G.A.; Simula, O. Minimal Learning Machine: A New Distance-Based Method for supervised Learning. In *International Work-Conference on Artificial Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 408–416.
2. De Souza Junior, A.H.; Corona, F.; Barreto, G.A.; Miche, Y.; Lendasse, A. Minimal Learning Machine: A novel supervised distance-based approach for regression and classification. *Neurocomputing* **2015**, *164*, 34–44. [[CrossRef](#)]
3. Pihlajamäki, A.; Hämäläinen, J.; Linja, J.; Nieminen, P.; Malola, S.; Kärkkäinen, T.; Häkkinen, H. Monte Carlo Simulations of Au<sub>38</sub>(SCH<sub>3</sub>)<sub>24</sub> Nanocluster Using Distance-Based Machine Learning Methods. *J. Phys. Chem.* **2020**, *124*, 23. [[CrossRef](#)]
4. Marinho, L.B.; Almeida, J.S.; Souza, J.W.M.; Albuquerque, V.H.C.; Rebouças Filho, P.P. A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images. *Expert Syst. Appl.* **2017**, *72*, 1–17. [[CrossRef](#)]
5. Coelho, D.N.; Barreto, G.A.; Medeiros, C.M.S.; Santos, J.D.A. Performance comparison of classifiers in the detection of short circuit incipient fault in a three-phase induction motor. In Proceedings of the 2014 IEEE Symposium on Computational Intelligence for Engineering Solutions (CIES), Orlando, FL, USA, 9–12 December 2014; pp. 42–48.
6. Pekalska, E.; Duin, R.P. Automatic pattern recognition by similarity representations. *Electron. Lett.* **2001**, *37*, 159–160. [[CrossRef](#)]
7. Pekalska, E.; Paclik, P.; Duin, R.P. A generalized kernel approach to dissimilarity-based classification. *J. Mach. Learn. Res.* **2001**, *2*, 175–211.
8. Balcan, M.F.; Blum, A.; Srebro, N. A theory of learning with similarity functions. *Mach. Learn.* **2008**, *72*, 89–112. [[CrossRef](#)]
9. Zorzucha, P.; Daszykowski, M.; Walczak, B. Dissimilarity partial least squares applied to non-linear modeling problems. *Chemom. Intell. Lab. Syst.* **2012**, *110*, 156–162. [[CrossRef](#)]
10. Sanchez, J.D.; Rêgo, L.C.; Ospina, R. Prediction by Empirical Similarity via Categorical Regressors. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 38. [[CrossRef](#)]
11. Kärkkäinen, T. Extreme minimal learning machine: Ridge regression with distance-based basis. *Neurocomputing* **2019**, *342*, 33–48. [[CrossRef](#)]
12. Hämäläinen, J.; Alencar, A.S.; Kärkkäinen, T.; Mattos, C.L.; Júnior, A.H.S.; Gomes, J.P. Minimal Learning Machine: Theoretical Results and Clustering-Based Reference Point Selection. *arXiv* **2019**, arXiv:1909.09978v2. To appear.
13. Florêncio, J.A.; Oliveira, S.A.; Gomes, J.P.; Neto, A.R.R. A new perspective for Minimal Learning Machines: A lightweight approach. *Neurocomputing* **2020**, *401*, 308–319. [[CrossRef](#)]
14. Hämäläinen, J.; Kärkkäinen, T. Newton's Method for Minimal Learning Machine. In *Computational Sciences and Artificial Intelligence in Industry—New Digital Technologies for Solving Future Societal and Economical Challenges*; Springer Nature: Cham, Switzerland, 2020.
15. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [[CrossRef](#)] [[PubMed](#)]
16. Dias, M.L.D.; de Souza, L.S.; da Rocha Neto, A.R.; de Souza Junior, A.H. Opposite neighborhood: A new method to select reference points of minimal learning machines. In Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning—ESANN, Bruges, Belgium, 25–27 April 2018; pp. 201–206.
17. Florêncio, J.A.V.; Dias, M.L.D.; da Rocha Neto, A.R.; de Souza Júnior, A.H. A Fuzzy C-Means-Based Approach for Selecting Reference Points in Minimal Learning Machines; Barreto, G.A., Coelho, R., Eds.; Springer: Cham, Switzerland, 2018; pp. 398–407.
18. Mesquita, D.P.P.; Gomes, J.P.P.; Souza Junior, A.H. Ensemble of Efficient Minimal Learning Machines for Classification and Regression. *Neural Process. Lett.* **2017**, *46*, 751–766. [[CrossRef](#)]
19. Grigorievskiy, A.; Miche, Y.; Käpylä, M.; Lendasse, A. Singular value decomposition update and its application to (Inc)-OP-ELM. *Neurocomputing* **2016**, *174*, 99–108. [[CrossRef](#)]

20. Martinsson, P.G.; Rokhlin, V.; Tygert, M. A randomized algorithm for the decomposition of matrices. *Appl. Comput. Harmon. Anal.* **2011**, *30*, 47–68. [CrossRef]
21. Shabat, G.; Shmueli, Y.; Aizenbud, Y.; Averbuch, A. Randomized LU decomposition. *Appl. Comput. Harmon. Anal.* **2018**, *44*, 246–272. [CrossRef]
22. Abdelfattah, S.; Haidar, A.; Tomov, S.; Dongarra, J. Analysis and Design Techniques towards High-Performance and Energy-Efficient Dense Linear Solvers on GPUs. *IEEE Trans. Parallel Distrib. Syst.* **2018**, *29*, 2700–2712. [CrossRef]
23. Gonzalez, T.F. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* **1985**, *38*, 293–306. [CrossRef]
24. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: A new learning scheme of feedforward neural networks. In Proceedings of the 2004 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 25–29 July 2004; pp. 985–990.
25. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
26. Oliphant, T.E. *A Guide to NumPy*; Trelgol Publishing: USA, 2006.
27. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0—Fundamental Algorithms for Scientific Computing in Python. *arXiv* **2019**, arXiv:1907.10121.
28. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
29. Stewart, G.W. On the Early History of the Singular Value Decomposition. *SIAM Rev.* **1993**, *35*, 551–566. [CrossRef]
30. Tikhonov, A.N.; Arsenin, V.J. *Solution of Ill-Posed Problems*; Winston&Sons: New York, NY, USA, 1977.
31. Halko, N.; Martinsson, P.G.; Tropp, J.A. Finding Structure with Randomness: Stochastic Algorithms for Constructing Approximate matrix Decompositions. *ACM Tech. Rep.* **2009**. [CrossRef]
32. Kruskal, W.H.; Wallis, W.A. Use of ranks in one-criterion variance analysis. *J. Am. Stat. Assoc.* **1952**, *47*, 583–621. [CrossRef]
33. Juarez-Mosqueda, R.; Sami, M.; Hannu, H. Ab initio molecular dynamics studies of Au<sub>38</sub>(SR)<sub>24</sub> isomers under heating. *Eur. Phys. J.* **2019**, *73*. [CrossRef]
34. Dua, D.; Graff, C. UCI Machine Learning Repository. California, USA 2017. Available online: <http://archive.ics.uci.edu/ml> (accessed on 13 November 2020).
35. Torgo, L. *Airplane Companies Stocks*; Faculdade de Ciências da Universidade do Porto: Porto, Portugal, 1991.
36. University of Toronto. *Delve Datasets*; University of Toronto: Toronto, ON, Canada, 1996.
37. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2323. [CrossRef]
38. Khamparia, A.; Singh, K.M. A systematic review on deep learning architectures and applications. *Expert Syst.* **2019**, *36*, e12400. [CrossRef]
39. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. *arXiv* **2015**, arXiv:1603.04467.
40. Emmert-Streib, F.; Dehmer, M. Understanding Statistical Hypothesis Testing: The Logic of Statistical Inference. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 54. [CrossRef]
41. Zhou, Z.H.; Feng, J. Deep forest. *Natl. Sci. Rev.* **2018**, *6*, 74–86. [CrossRef]
42. Li, H.B.; Huang, T.Z.; Zhang, Y.; Liu, X.P.; Gu, T.X. Chebyshev-type methods and preconditioning techniques. *Appl. Math. Comput.* **2011**, *218*, 260–270. [CrossRef]
43. Hadjidimos, A. Accelerated overrelaxation method. *Math. Comput.* **1978**, *32*, 149–157. [CrossRef]
44. Qian, H.; Eckenhoff, W.T.; Zhu, Y.; Pintauer, T.; Jin, R. Total Structure Determination of Thiolate-Protected Au<sub>38</sub> Nanoparticles. *J. Am. Chem. Soc.* **2010**, *132*, 8280–8281. [CrossRef] [PubMed]

45. Huo, H.; Rupp, M. Unified Representation of Molecules and Crystals for Machine Learning. *arXiv* **2017**, arXiv:1704.06439.
46. Himanen, L.; Jäger, M.O.J.; Morooka, E.V.; Federici Canova, F.; Ranawat, Y.S.; Gao, D.Z.; Rinke, P.; Foster, A.S. Dscribe: Library of descriptors for machine learning in materials science. *Comput. Phys. Commun.* **2020**, *247*, 106949. [[CrossRef](#)]

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

# Supplementary Materials: Do Randomized Algorithms Improve the Efficiency of Minimal Learning Machine?

Joakim Linja <sup>1\*</sup>, Joonas Hämäläinen <sup>1</sup>, Paavo Nieminen <sup>1</sup> and Tommi Kärkkäinen <sup>1</sup>

## 1. RMSE tables

- 2 Tables S1–S9 report the mean and standard deviation of MLM test error, measured as RMSE value.
- 3 The data table with  $K = 70$  for RMSE is shown in the main text and is not repeated here due to that.

**Table S1.** RMSE for  $K = 10$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
<b>Method / coef.</b>										
Lstsq	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
Lstsq (nr)	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
np.inv	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
np.inv (nr)	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
np.solve	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
np.solve (nr)	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
np.svd	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
sp.svd	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
Cho.Dec	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
rKrSVD02	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
rKrSVD04	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
rKrSVD06	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
rKrSVD08	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
rKrSVD10	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
skrSVD02	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
skrSVD04	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
skrSVD06	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	6.17	2.24
skrSVD08	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
skrSVD10	2.59	2.16	9.55	1.46	3.06	1.94	2.58	7.53	-	-
SVDU	2.62	2.16	9.58	1.63	<b>3.56</b>	3.14	-	-	-	-

**Table S2.** RMSE for  $K = 20$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
<b>Method / coef.</b>										
Lstsq	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
Lstsq (nr)	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
np.inv	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
np.inv (nr)	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
np.solve	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
np.solve (nr)	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
np.svd	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
sp.svd	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
Cho.Dec	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
rKrSVD02	2.63	2.28	7.88	1.48	2.54	1.5	2.49	7.99	6.19	2.22
rKrSVD04	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
rKrSVD06	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
rKrSVD08	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
rKrSVD10	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
skrSVD02	2.63	2.28	7.88	1.48	2.54	1.5	2.49	7.99	6.19	2.22
skrSVD04	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
skrSVD06	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	6.19	2.22
skrSVD08	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
skrSVD10	2.63	2.28	7.87	1.48	2.54	1.5	2.49	7.99	-	-
SVDU	2.64	2.33	<b>8.64</b>	1.56	<b>2.98</b>	2.43	-	-	-	-

**Table S3.** RMSE for  $K = 30$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
<b>Method / coef.</b>										
Lstsq	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
Lstsq (nr)	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
np.inv	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
np.inv (nr)	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
np.solve	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
np.solve (nr)	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
np.svd	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
sp.svd	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
Cho.Dec	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
rKrSVD02	2.64	2.17	<b>7.83</b>	1.45	<b>2.54</b>	1.52	2.47	8.09	6.18	2.18
rKrSVD04	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
rKrSVD06	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
rKrSVD08	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
rKrSVD10	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
skrSVD02	2.64	2.22	7.68	1.46	<b>2.49</b>	1.55	2.46	8.08	6.18	2.22
skrSVD04	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
skrSVD06	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	6.2	2.21
skrSVD08	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
skrSVD10	2.67	2.19	7.39	1.46	2.39	1.47	2.45	8.14	-	-
SVDU	2.66	2.39	<b>8.11</b>	1.56	<b>2.69</b>	2.23	-	-	-	-

**Table S4.** RMSE for  $K = 40$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
<b>Method / coef.</b>										
Lstsq	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
Lstsq (nr)	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
np.inv	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
np.inv (nr)	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
np.solve	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
np.solve (nr)	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
np.svd	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
sp.svd	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
Cho.Dec	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
rKrSVD02	2.65	2.29	<b>7.81</b>	1.35	<b>2.56</b>	1.47	<b>2.47</b>	8.2	6.19	2.22
rKrSVD04	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
rKrSVD06	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
rKrSVD08	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
rKrSVD10	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
skrSVD02	2.65	2.27	<b>7.65</b>	1.41	<b>2.49</b>	1.53	2.46	8.13	6.18	2.2
skrSVD04	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.19
skrSVD06	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	6.24	2.2
skrSVD08	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
skrSVD10	2.7	2.24	7.04	1.33	2.31	1.46	2.45	8.33	-	-
SVDU	2.69	2.42	<b>7.76</b>	1.54	<b>2.53</b>	1.96	-	-	-	-

**Table S5.** RMSE for  $K = 50$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$
<b>Method / coef.</b>										
Lstsq	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
Lstsq (nr)	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
np.inv	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
np.inv (nr)	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
np.solve	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
np.solve (nr)	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
np.svd	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
sp.svd	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
Cho.Dec	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
rKrSVD02	<b>2.68</b>	2.23	<b>7.86</b>	1.4	<b>2.55</b>	1.54	<b>2.47</b>	8.13	6.2	2.19
rKrSVD04	2.72	2.25	7.06	1.35	<b>2.31</b>	1.45	2.45	8.24	6.24	2.2
rKrSVD06	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
rKrSVD08	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
rKrSVD10	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
skrSVD02	<b>2.66</b>	2.25	<b>7.73</b>	1.47	<b>2.48</b>	1.54	2.46	8.21	6.18	2.23
skrSVD04	2.72	2.25	6.98	1.36	2.29	1.46	2.44	8.3	6.24	2.18
skrSVD06	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	6.26	2.2
skrSVD08	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
skrSVD10	2.75	2.29	6.95	1.35	2.26	1.45	2.44	8.34	-	-
SVDU	2.71	2.45	<b>7.48</b>	1.51	<b>2.42</b>	1.82	-	-	-	-



**Table S6.** RMSE for  $K = 60$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>
Lstsq	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
Lstsq (nr)	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
np.inv	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
np.inv (nr)	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
np.solve	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
np.solve (nr)	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
np.svd	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
sp.svd	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
Cho.Dec	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
rKrSVD02	<b>2.67</b>	2.22	<b>7.96</b>	1.45	<b>2.56</b>	1.57	<b>2.47</b>	8.64	6.21	2.24
rKrSVD04	2.72	2.3	7.1	1.37	<b>2.3</b>	1.44	2.45	8.2	6.24	2.22
rKrSVD06	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.27	2.19
rKrSVD08	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
rKrSVD10	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
skrSVD02	<b>2.66</b>	2.28	<b>7.77</b>	1.46	<b>2.49</b>	1.63	2.46	8.3	<b>6.18</b>	2.21
skrSVD04	2.72	2.28	6.97	1.37	<b>2.28</b>	1.43	2.44	8.26	6.24	2.2
skrSVD06	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	6.29	2.22
skrSVD08	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
skrSVD10	2.76	2.26	6.83	1.36	2.23	1.45	2.44	8.41	-	-
SVDU	2.71	2.41	<b>7.28</b>	1.54	<b>2.34</b>	1.75	-	-	-	-

**Table S7.** RMSE for  $K = 80$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	$\bar{x}$	$\sigma$
Lstsq	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
Lstsq (nr)	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
np.inv	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
np.inv (nr)	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
np.solve	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
np.solve (nr)	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
np.svd	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
sp.svd	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
Cho.Dec	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
rKrSVD02	<b>2.68</b>	2.41	<b>8.01</b>	1.39	<b>2.58</b>	1.53	<b>2.48</b>	8.32	-	-
rKrSVD04	2.74	2.3	7.13	1.36	<b>2.32</b>	1.43	2.45	8.33	-	-
rKrSVD06	2.73	2.37	6.88	1.36	<b>2.22</b>	1.45	2.44	8.47	-	-
rKrSVD08	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
rKrSVD10	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
skrSVD02	<b>2.66</b>	2.32	<b>7.84</b>	1.46	<b>2.51</b>	1.65	<b>2.45</b>	8.43	-	-
skrSVD04	2.71	2.34	6.99	1.37	<b>2.28</b>	1.42	2.44	8.34	-	-
skrSVD06	2.74	2.36	6.84	1.36	2.21	1.46	2.43	8.44	-	-
skrSVD08	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
skrSVD10	2.77	2.39	6.78	1.35	2.18	1.48	2.43	8.4	-	-
SVDU	2.75	2.41	6.94	1.41	<b>2.22</b>	1.55	-	-	-	-

**Table S8.** RMSE for  $K = 90$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
Method / coef.	$1 \times 10^{-1}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-3}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	-	-
Lstsq	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
Lstsq (nr)	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
np.inv	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
np.inv (nr)	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
np.solve	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
np.solve (nr)	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
np.svd	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
sp.svd	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
Cho.Dec	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
rKrSVD02	<b>2.7</b>	2.39	<b>7.97</b>	1.44	<b>2.6</b>	1.53	<b>2.47</b>	8.5	-	-
rKrSVD04	2.73	2.4	<b>7.1</b>	1.35	<b>2.32</b>	1.45	<b>2.45</b>	8.39	-	-
rKrSVD06	2.73	2.35	6.85	1.36	<b>2.23</b>	1.49	2.44	8.55	-	-
rKrSVD08	2.76	2.44	6.75	1.35	2.18	1.47	2.43	8.45	-	-
rKrSVD10	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
skrSVD02	<b>2.67</b>	2.32	<b>7.83</b>	1.47	<b>2.52</b>	1.59	<b>2.45</b>	8.37	-	-
skrSVD04	2.71	2.26	6.96	1.38	<b>2.28</b>	1.44	2.43	8.38	-	-
skrSVD06	2.73	2.37	6.81	1.37	<b>2.21</b>	1.46	2.43	8.47	-	-
skrSVD08	2.76	2.46	6.74	1.35	2.17	1.48	2.43	8.44	-	-
skrSVD10	2.77	2.45	6.72	1.36	2.16	1.49	2.42	8.48	-	-
SVDU	2.77	2.38	6.79	1.36	2.19	1.49	-	-	-	-

**Table S9.** RMSE for  $K = 100$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
Method / coef.	$1 \times 10^{-1}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-3}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	-	-
Lstsq	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
Lstsq (nr)	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
np.inv	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
np.inv (nr)	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
np.solve	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
np.solve (nr)	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
np.svd	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
sp.svd	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
Cho.Dec	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
rKrSVD02	<b>2.69</b>	2.35	<b>8.03</b>	1.4	<b>2.61</b>	1.66	<b>2.47</b>	8.51	-	-
rKrSVD04	2.73	2.36	<b>7.13</b>	1.33	<b>2.32</b>	1.45	2.44	8.58	-	-
rKrSVD06	2.75	2.41	6.87	1.35	<b>2.23</b>	1.45	2.43	8.62	-	-
rKrSVD08	2.76	2.32	6.79	1.35	2.18	1.52	2.43	8.46	-	-
rKrSVD10	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
skrSVD02	<b>2.66</b>	2.31	<b>7.84</b>	1.43	<b>2.54</b>	1.63	<b>2.45</b>	8.38	-	-
skrSVD04	<b>2.69</b>	2.39	6.99	1.37	<b>2.28</b>	1.45	2.43	8.46	-	-
skrSVD06	2.73	2.34	6.82	1.37	<b>2.21</b>	1.44	2.43	8.52	-	-
skrSVD08	2.77	2.35	6.76	1.35	2.17	1.51	2.42	8.5	-	-
skrSVD10	2.78	2.43	6.71	1.36	2.15	1.5	2.42	8.48	-	-
SVDU	2.78	2.43	6.71	1.36	2.15	1.5	-	-	-	-

4 **2. Process time tables**

- 5 Tables [S10–S19](#) report the mean and standard deviation of process time taken in OLS evaluation.

**Table S10.** Process time (s) for  $K = 10$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-1}$	$\bar{x}$ $1 \times 10^2$	$\sigma$ $1 \times 10^1$
<b>Method / coef.</b>										
Lstsq	0.71	3.46	2.93	0.75	0.87	0.18	1.05	3.19	0.68	0.09
Lstsq (nr)	<b>0.61</b>	3.26	<b>2.67</b>	0.7	<b>0.8</b>	0.14	<b>1.01</b>	3.19	-	-
np.inv	<b>0.4</b>	3.84	<b>1.44</b>	0.66	<b>0.86</b>	2.88	<b>0.57</b>	2.37	<b>0.21</b>	0.17
np.inv (nr)	<b>0.33</b>	6.36	<b>1.42</b>	1.02	<b>0.89</b>	3.46	<b>0.52</b>	3.81	-	-
np.solve	<b>0.4</b>	6.03	<b>1.78</b>	45.32	<b>0.85</b>	5.36	<b>0.53</b>	1.74	<b>0.17</b>	0.06
np.solve (nr)	<b>0.31</b>	3.62	<b>1.54</b>	26.85	<b>0.79</b>	1.85	<b>0.53</b>	1.5	-	-
np.svd	<b>0.69</b>	4.93	<b>3.86</b>	1.02	<b>1.35</b>	0.33	<b>0.99</b>	1.71	<b>0.99</b>	0.53
sp.svd	<b>0.83</b>	4.86	<b>3.88</b>	2.58	<b>1.32</b>	0.36	<b>0.9</b>	1.48	-	-
Cho.Dec	0.71	4.82	<b>1.46</b>	0.4	<b>0.3</b>	0.06	<b>0.36</b>	1.57	<b>0.19</b>	0.05
rKrSVD02	<b>1.38</b>	7.81	<b>8.18</b>	4.46	<b>3.02</b>	3.01	<b>2.02</b>	3.11	<b>2.75</b>	3.08
rKrSVD04	<b>2.66</b>	24.14	<b>18.28</b>	5.84	<b>5.41</b>	3.46	<b>4.89</b>	46.35	<b>7.98</b>	12.9
rKrSVD06	<b>5.86</b>	175.01	<b>27.47</b>	11.01	<b>7.68</b>	6.67	<b>8.82</b>	138.23	<b>15.73</b>	28.32
rKrSVD08	<b>9.75</b>	31.69	<b>31.5</b>	13.87	<b>10.12</b>	3.88	<b>13.63</b>	246.86	-	-
rKrSVD10	<b>9.3</b>	23.94	<b>36.57</b>	24.79	<b>11.61</b>	8.62	<b>18.98</b>	367.6	-	-
skrSVD02	<b>2.08</b>	4.19	<b>12.58</b>	19.79	<b>3.31</b>	0.85	<b>2.32</b>	3.86	<b>3.08</b>	3.69
skrSVD04	<b>4.18</b>	8.45	<b>21.29</b>	6.05	<b>5.98</b>	4.55	<b>5.16</b>	51.83	<b>8.68</b>	14.9
skrSVD06	<b>7.95</b>	19.95	<b>30.11</b>	11.45	<b>8.11</b>	8.98	<b>9.21</b>	139.95	<b>16.59</b>	31.63
skrSVD08	<b>7.69</b>	58.72	<b>34.12</b>	28.07	<b>9.74</b>	2.79	<b>14.1</b>	250.61	-	-
skrSVD10	<b>10.43</b>	78.93	<b>37.07</b>	23.16	<b>11.72</b>	3.65	<b>19.49</b>	371.08	-	-
SVDU	<b>0.23</b>	0.86	<b>0.5</b>	0.29	<b>0.11</b>	0.06	-	-	-	-

**Table S11.** Process time (s) for  $K = 20$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>1</sup>	$\sigma$ 1x10 <sup>-1</sup>	$\bar{x}$ 1x10 <sup>2</sup>	$\sigma$ 1x10 <sup>1</sup>
Lstsq	1.41	4.43	0.91	0.17	4.47	3.55	3.08	5.12	4.68	2.74
Lstsq (nr)	<b>1.29</b>	4.04	<b>0.87</b>	0.14	<b>4.33</b>	3.26	<b>2.95</b>	4.61	-	-
np.inv	<b>1.05</b>	7.34	<b>1.06</b>	7.99	<b>3.49</b>	8.57	<b>0.8</b>	2.74	<b>0.63</b>	0.41
np.inv (nr)	<b>0.96</b>	5.6	<b>0.91</b>	0.34	<b>3.38</b>	5.06	<b>0.79</b>	3.68	-	-
np.solve	<b>0.99</b>	5.77	1.12	9.15	<b>3.42</b>	5.89	<b>0.73</b>	3.05	<b>0.49</b>	0.18
np.solve (nr)	<b>0.91</b>	5.81	<b>0.91</b>	0.51	<b>3.44</b>	9.17	<b>0.71</b>	2.67	-	-
np.svd	<b>1.53</b>	4.54	<b>1.35</b>	0.52	<b>4.3</b>	2.1	<b>2.95</b>	4.31	4.78	5.58
sp.svd	<b>1.65</b>	3.57	<b>1.29</b>	0.36	<b>4.21</b>	1.46	<b>2.73</b>	4.03	-	-
Cho.Dec	<b>0.91</b>	4.61	<b>0.24</b>	0.05	<b>0.67</b>	0.13	<b>0.52</b>	0.93	<b>0.61</b>	0.32
rKrSVD02	<b>2.09</b>	4.03	<b>1.44</b>	0.38	<b>4.74</b>	3.3	<b>3.34</b>	4.66	<b>5.19</b>	5.1
rKrSVD04	<b>3.58</b>	16.01	<b>2.68</b>	0.73	<b>8.29</b>	2.49	<b>6.81</b>	52.26	<b>11.24</b>	16.8
rKrSVD06	<b>6.93</b>	17.37	<b>3.77</b>	4.25	<b>10.87</b>	7.11	<b>11.04</b>	144.73	<b>19.5</b>	33.04
rKrSVD08	<b>10.79</b>	31.3	<b>4.25</b>	3.81	<b>13.31</b>	3.89	<b>16.18</b>	256.66	-	-
rKrSVD10	<b>10.6</b>	118.74	<b>4.77</b>	3.96	<b>14.81</b>	4.42	<b>22.12</b>	384.69	-	-
skrSVD02	<b>2.93</b>	4.98	<b>1.76</b>	0.56	<b>5.23</b>	2.41	<b>3.88</b>	5.42	<b>5.88</b>	6.58
skrSVD04	<b>5.19</b>	41.87	<b>3.07</b>	1.92	<b>9.00</b>	6.62	<b>7.37</b>	59.56	<b>12.44</b>	19.19
skrSVD06	<b>9.02</b>	18.26	<b>4.12</b>	2.81	<b>11.4</b>	5.04	<b>11.99</b>	153.22	<b>21.39</b>	37.31
skrSVD08	<b>8.87</b>	16.33	<b>4.57</b>	4.16	<b>13.99</b>	4.12	<b>17.37</b>	269.13	-	-
skrSVD10	<b>11.68</b>	37.99	<b>4.93</b>	4.56	<b>15.75</b>	4.59	<b>23.63</b>	404.46	-	-
SVDU	<b>69.47</b>	415.75	<b>45.84</b>	73.05	<b>166.61</b>	242.93	-	-	-	-

**Table S12.** Process time (s) for  $K = 30$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>1</sup>	$\sigma$ 1x10 <sup>1</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>2</sup>
Lstsq	3.71	1.00	2.74	0.54	1.09	7.48	6.98	0.82	2.17	2.33
Lstsq (nr)	<b>3.54</b>	1.01	<b>2.65</b>	0.6	<b>1.06</b>	7.67	<b>6.79</b>	0.92	-	-
np.inv	<b>2.19</b>	1.03	<b>2.2</b>	4.91	<b>0.7</b>	11.07	<b>1.34</b>	0.4	<b>0.13</b>	0.07
np.inv (nr)	<b>2.64</b>	37.42	<b>2.17</b>	3.04	<b>0.7</b>	9.26	<b>1.28</b>	0.47	-	-
np.solve	<b>2.3</b>	17.7	<b>2.31</b>	7.95	<b>0.71</b>	12.19	<b>1.18</b>	0.42	<b>0.1</b>	0.05
np.solve (nr)	<b>2.54</b>	37.45	<b>2.14</b>	2.07	<b>0.69</b>	11.2	<b>1.14</b>	0.41	-	-
np.svd	<b>3.33</b>	0.89	<b>2.9</b>	0.9	<b>0.75</b>	5.67	<b>5.92</b>	1.97	<b>1.46</b>	2.05
sp.svd	<b>3.43</b>	0.87	<b>2.85</b>	1.67	<b>0.73</b>	10.34	<b>5.67</b>	2.47	-	-
Cho.Dec	<b>1.28</b>	0.63	<b>0.37</b>	0.17	<b>0.13</b>	0.47	<b>1.02</b>	0.27	<b>0.12</b>	0.05
rKrSVD02	<b>2.37</b>	0.43	<b>1.68</b>	1.77	<b>0.58</b>	5.12	<b>4.07</b>	0.69	<b>0.8</b>	0.94
rKrSVD04	<b>5.2</b>	3.98	<b>3.71</b>	0.92	<b>1.02</b>	8.73	<b>9.06</b>	5.44	<b>2.34</b>	3.6
rKrSVD06	<b>8.84</b>	1.75	<b>5.52</b>	3.44	<b>1.44</b>	15.73	<b>13.95</b>	14.31	<b>2.94</b>	4.96
rKrSVD08	<b>12.8</b>	3.13	<b>6.00</b>	1.43	<b>1.7</b>	4.86	<b>19.48</b>	25.75	-	-
rKrSVD10	<b>12.6</b>	2.84	<b>6.61</b>	1.84	<b>1.89</b>	7.39	<b>25.86</b>	39.04	-	-
skrSVD02	<b>4.2</b>	2.91	<b>2.18</b>	0.41	<b>0.63</b>	5.53	<b>4.69</b>	0.81	<b>0.88</b>	1.15
skrSVD04	<b>6.9</b>	4.89	<b>4.12</b>	0.93	<b>1.11</b>	9.29	<b>9.96</b>	6.84	<b>2.5</b>	3.99
skrSVD06	<b>11.17</b>	8.82	<b>5.82</b>	2.83	<b>1.51</b>	6.59	<b>15.3</b>	16.58	<b>3.15</b>	5.52
skrSVD08	<b>11.09</b>	1.96	<b>6.34</b>	4.52	<b>1.78</b>	4.73	<b>21.21</b>	29.17	-	-
skrSVD10	<b>13.91</b>	2.73	<b>6.84</b>	3.52	<b>1.98</b>	5.32	<b>27.85</b>	43.78	-	-
SVDU	<b>104.84</b>	45.74	<b>75.01</b>	116.65	<b>29.85</b>	178.95	-	-	-	-

**Table S13.** Process time (s) for  $K = 40$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^2$	$\sigma$ $1 \times 10^1$	$\bar{x}$ $1 \times 10^3$	$\sigma$ $1 \times 10^2$
<b>Method / coef.</b>										
Lstsq	4.62	3.25	5.03	2.36	1.89	14.76	1.4	2.86	5.49	7.26
Lstsq (nr)	<b>4.46</b>	4.3	<b>4.91</b>	2.41	<b>1.82</b>	5.5	<b>1.38</b>	2.88	-	-
np.inv	<b>5.57</b>	48.52	<b>3.54</b>	5.7	<b>1.03</b>	14.65	<b>0.21</b>	0.7	<b>0.23</b>	0.14
np.inv (nr)	<b>5.29</b>	27.8	<b>3.49</b>	8.37	<b>1.00</b>	6.74	<b>0.2</b>	0.75	-	-
np.solve	<b>5.49</b>	37.29	<b>3.64</b>	9.38	<b>1.00</b>	11.00	<b>0.18</b>	0.39	<b>0.17</b>	0.09
np.solve (nr)	<b>5.71</b>	66.33	<b>3.57</b>	8.18	<b>1.00</b>	13.42	<b>0.17</b>	0.55	-	-
np.svd	<b>5.5</b>	1.52	<b>4.04</b>	1.08	<b>1.15</b>	13.44	<b>1.15</b>	9.97	<b>3.25</b>	5.00
sp.svd	<b>5.57</b>	1.43	<b>3.87</b>	0.98	<b>1.13</b>	4.29	<b>1.13</b>	10.69	-	-
Cho.Dec	<b>1.52</b>	0.61	<b>0.56</b>	0.12	<b>0.19</b>	0.32	<b>0.17</b>	0.3	<b>0.22</b>	0.08
rKrSVD02	<b>2.56</b>	0.52	<b>2.05</b>	0.9	<b>0.73</b>	6.3	<b>0.49</b>	0.85	<b>0.72</b>	0.63
rKrSVD04	<b>6.52</b>	1.48	<b>4.47</b>	2.43	<b>1.29</b>	5.7	<b>1.27</b>	8.4	<b>3.28</b>	4.54
rKrSVD06	<b>10.23</b>	2.03	<b>6.04</b>	3.58	<b>1.75</b>	12.19	<b>2.41</b>	29.93	5.72	8.46
rKrSVD08	<b>14.88</b>	20.39	<b>7.19</b>	4.75	<b>2.17</b>	5.4	<b>2.54</b>	32.29	-	-
rKrSVD10	<b>15.19</b>	6.86	<b>8.06</b>	7.06	<b>2.4</b>	7.93	<b>3.24</b>	46.71	-	-
skrSVD02	<b>4.75</b>	26.04	<b>2.53</b>	0.49	<b>0.75</b>	4.98	<b>0.56</b>	1.12	<b>0.84</b>	0.88
skrSVD04	<b>8.52</b>	9.27	<b>4.92</b>	3.4	<b>1.41</b>	4.16	<b>1.4</b>	11.11	<b>3.55</b>	5.19
skrSVD06	<b>12.7</b>	9.88	<b>6.38</b>	3.92	<b>1.84</b>	4.06	<b>2.57</b>	33.22	<b>6.04</b>	9.89
skrSVD08	<b>13.03</b>	2.55	<b>7.59</b>	6.37	<b>2.23</b>	5.55	<b>2.74</b>	36.77	-	-
skrSVD10	<b>16.61</b>	7.5	<b>8.08</b>	1.96	<b>2.49</b>	21.2	<b>3.49</b>	53.82	-	-
SVDU	<b>49.86</b>	22.8	<b>114.78</b>	142.93	<b>44.98</b>	389.7	-	-	-	-

**Table S14.** Process time (s) for  $K = 50$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^2$	$\sigma$ $1 \times 10^1$	$\bar{x}$ $1 \times 10^3$	$\sigma$ $1 \times 10^2$
<b>Method / coef.</b>										
Lstsq	8.08	5.85	9.25	12.04	2.78	6.92	2.88	1.63	9.81	12.28
Lstsq (nr)	<b>7.74</b>	1.57	<b>8.99</b>	7.82	<b>2.75</b>	10.53	2.88	1.77	-	-
np.inv	8.8	57.52	<b>5.46</b>	8.24	<b>1.5</b>	15.93	<b>0.29</b>	0.11	<b>0.36</b>	0.18
np.inv (nr)	<b>8.48</b>	47.64	<b>5.27</b>	3.89	<b>1.47</b>	13.35	<b>0.28</b>	0.1	-	-
np.solve	<b>9.83</b>	106.89	<b>5.65</b>	12.11	<b>1.47</b>	19.87	<b>0.24</b>	0.06	<b>0.26</b>	0.17
np.solve (nr)	<b>9.56</b>	91.57	<b>5.43</b>	10.09	<b>1.44</b>	21.34	<b>0.23</b>	0.07	-	-
np.svd	<b>8.75</b>	2.76	<b>5.79</b>	4.22	<b>1.72</b>	15.16	<b>2.46</b>	2.89	<b>5.87</b>	10.18
sp.svd	<b>8.85</b>	9.66	<b>5.6</b>	4.26	<b>1.67</b>	5.67	<b>2.44</b>	2.93	-	-
Cho.Dec	<b>1.89</b>	0.59	<b>0.76</b>	0.21	<b>0.29</b>	0.43	<b>0.25</b>	0.06	<b>0.34</b>	0.15
rKrSVD02	<b>2.68</b>	0.54	<b>2.32</b>	1.09	<b>0.82</b>	5.99	<b>0.54</b>	0.08	<b>0.8</b>	0.64
rKrSVD04	<b>6.93</b>	1.74	<b>5.07</b>	5.29	<b>1.51</b>	14.08	<b>1.64</b>	1.24	<b>4.28</b>	5.93
rKrSVD06	<b>12.14</b>	3.19	<b>7.42</b>	6.47	<b>2.18</b>	5.86	<b>3.43</b>	4.25	<b>8.08</b>	12.9
rKrSVD08	<b>17.44</b>	21.88	<b>8.38</b>	2.85	<b>2.72</b>	9.11	<b>5.18</b>	7.79	-	-
rKrSVD10	<b>18.96</b>	22.02	<b>10.05</b>	2.32	<b>3.07</b>	8.59	<b>4.57</b>	6.54	-	-
skrSVD02	<b>4.71</b>	0.95	<b>2.95</b>	3.29	<b>0.87</b>	2.57	<b>0.64</b>	0.11	<b>0.94</b>	1.06
skrSVD04	<b>9.56</b>	2.00	<b>5.68</b>	3.08	<b>1.66</b>	7.34	<b>1.8</b>	1.61	<b>4.6</b>	7.26
skrSVD06	<b>14.66</b>	4.14	<b>7.76</b>	5.55	<b>2.33</b>	5.17	<b>3.66</b>	4.92	<b>8.31</b>	13.98
skrSVD08	<b>16.06</b>	3.53	<b>8.71</b>	1.87	<b>2.78</b>	6.87	<b>5.49</b>	8.53	-	-
skrSVD10	<b>20.1</b>	4.7	<b>10.12</b>	5.4	<b>3.09</b>	13.74	<b>4.88</b>	7.3	-	-
SVDU	<b>199.23</b>	35.27	<b>146.85</b>	180.49	<b>61.48</b>	503.14	-	-	-	-

**Table S15.** Process time (s) for  $K = 60$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
Method / coef.	$1 \times 10^{-1}$	$1 \times 10^{-3}$	$1 \times 10^{-1}$	$1 \times 10^{-2}$	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^2$	$1 \times 10^1$	$1 \times 10^3$	$1 \times 10^2$
Lstsq	1.13	2.35	12.32	2.64	4.00	0.97	6.44	5.73	16.29	20.62
Lstsq (nr)	<b>1.1</b>	2.52	<b>12.17</b>	5.76	4.00	1.31	6.47	5.93	-	-
np.inv	<b>1.25</b>	27.89	<b>7.52</b>	13.18	<b>1.83</b>	1.21	<b>0.39</b>	0.15	<b>0.54</b>	0.27
np.inv (nr)	<b>1.27</b>	52.01	<b>7.2</b>	7.32	<b>1.83</b>	1.65	<b>0.38</b>	0.09	-	-
np.solve	<b>1.25</b>	15.13	<b>7.55</b>	13.1	<b>1.84</b>	1.63	<b>0.33</b>	0.08	<b>0.39</b>	0.27
np.solve (nr)	<b>1.36</b>	85.89	<b>7.49</b>	12.9	<b>1.8</b>	1.37	<b>0.32</b>	0.07	-	-
np.svd	<b>1.2</b>	4.31	<b>6.72</b>	6.18	<b>2.28</b>	1.7	<b>7.22</b>	12.24	<b>14.76</b>	20.75
sp.svd	<b>1.2</b>	3.7	<b>6.46</b>	2.16	<b>2.22</b>	1.04	<b>7.24</b>	12.59	-	-
Cho.Dec	<b>0.22</b>	0.72	<b>0.99</b>	0.7	<b>0.35</b>	0.09	<b>0.34</b>	0.09	<b>0.51</b>	0.24
rKrSVD02	<b>0.29</b>	1.45	<b>2.55</b>	0.68	<b>0.91</b>	1.04	<b>0.6</b>	0.1	<b>0.91</b>	0.65
rKrSVD04	<b>0.73</b>	5.64	<b>5.37</b>	1.19	<b>1.62</b>	1.15	<b>2.05</b>	1.7	<b>5.33</b>	7.02
rKrSVD06	<b>1.44</b>	3.42	<b>8.38</b>	2.4	<b>2.59</b>	0.78	<b>4.48</b>	5.72	<b>10.02</b>	14.59
rKrSVD08	<b>2.06</b>	5.59	<b>10.03</b>	14.34	<b>3.24</b>	1.15	6.76	10.6	-	-
rKrSVD10	<b>2.23</b>	14.4	<b>11.4</b>	7.67	<b>3.67</b>	0.84	<b>9.18</b>	15.63	-	-
skrSVD02	<b>0.49</b>	1.00	<b>3.14</b>	0.69	<b>0.94</b>	0.65	<b>0.71</b>	0.12	<b>1.04</b>	0.85
skrSVD04	<b>1.01</b>	2.13	<b>6.15</b>	2.02	<b>1.8</b>	0.77	<b>2.24</b>	2.23	<b>5.52</b>	7.67
skrSVD06	<b>1.76</b>	3.98	<b>9.09</b>	9.73	<b>2.76</b>	1.2	<b>4.7</b>	6.64	<b>10.49</b>	16.59
skrSVD08	<b>1.97</b>	4.4	<b>10.2</b>	10.9	<b>3.29</b>	1.21	<b>7.00</b>	11.43	-	-
skrSVD10	<b>2.39</b>	5.66	<b>11.41</b>	7.82	<b>3.71</b>	0.87	<b>9.44</b>	17.01	-	-
SVDU	<b>22.81</b>	26.09	<b>191.38</b>	218.18	<b>79.62</b>	39.45	-	-	-	-

**Table S16.** Process time (s) for  $K = 70$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
Method / coef.	$1 \times 10^{-1}$	$1 \times 10^{-2}$	$1 \times 10^{-1}$	$1 \times 10^{-2}$	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^2$	$1 \times 10^1$	$1 \times 10^3$	$1 \times 10^2$
Lstsq	1.55	0.51	17.81	10.81	5.39	1.7	11.92	13.41	24.55	31.52
Lstsq (nr)	<b>1.52</b>	0.31	17.65	3.89	<b>5.34</b>	1.38	11.9	13.17	-	-
np.inv	<b>1.74</b>	7.97	<b>9.17</b>	14.43	<b>2.32</b>	1.83	<b>0.5</b>	0.18	<b>0.75</b>	0.33
np.inv (nr)	<b>1.69</b>	6.2	<b>8.91</b>	10.69	<b>2.32</b>	1.69	<b>0.48</b>	0.18	-	-
np.solve	<b>1.93</b>	10.79	<b>8.9</b>	9.21	<b>2.25</b>	1.3	<b>0.42</b>	0.11	<b>0.55</b>	0.35
np.solve (nr)	1.6	1.52	<b>8.95</b>	10.73	<b>2.25</b>	1.49	<b>0.41</b>	0.11	-	-
np.svd	1.56	1.52	<b>8.48</b>	6.57	<b>2.81</b>	0.54	<b>9.2</b>	15.63	<b>19.32</b>	21.7
sp.svd	<b>1.51</b>	1.36	<b>8.21</b>	7.04	<b>2.78</b>	0.57	<b>9.28</b>	15.47	-	-
Cho.Dec	<b>0.25</b>	0.1	<b>1.34</b>	0.38	<b>0.49</b>	0.12	<b>0.45</b>	0.28	<b>0.71</b>	0.31
rKrSVD02	<b>0.31</b>	0.29	<b>2.78</b>	0.77	<b>0.95</b>	0.7	<b>0.66</b>	0.09	<b>1.01</b>	0.75
rKrSVD04	<b>0.77</b>	0.87	<b>5.92</b>	7.53	<b>1.85</b>	1.3	<b>2.49</b>	2.24	<b>6.21</b>	9.00
rKrSVD06	<b>1.63</b>	1.96	<b>9.17</b>	5.75	<b>2.93</b>	1.23	<b>5.39</b>	6.9	<b>11.45</b>	16.65
rKrSVD08	<b>2.41</b>	2.95	<b>11.63</b>	5.26	<b>3.89</b>	2.25	<b>8.5</b>	13.00	-	-
rKrSVD10	<b>2.63</b>	2.75	<b>13.65</b>	6.34	<b>4.48</b>	0.99	<b>11.4</b>	18.04	-	-
skrSVD02	<b>0.52</b>	0.13	<b>3.41</b>	0.65	<b>1.04</b>	0.84	<b>0.78</b>	0.13	<b>1.16</b>	1.08
skrSVD04	<b>1.05</b>	0.23	<b>6.61</b>	1.9	<b>1.96</b>	0.38	<b>2.67</b>	2.47	<b>6.3</b>	9.28
skrSVD06	<b>2.07</b>	0.6	<b>9.9</b>	6.16	<b>3.11</b>	1.71	<b>5.64</b>	7.34	<b>12.2</b>	19.83
skrSVD08	<b>2.3</b>	0.72	<b>11.71</b>	2.4	<b>3.98</b>	1.47	<b>8.66</b>	12.95	-	-
skrSVD10	<b>2.76</b>	0.98	<b>13.21</b>	2.75	<b>4.5</b>	0.87	<b>11.67</b>	18.74	-	-
SVDU	<b>27.22</b>	2.78	<b>234.85</b>	154.78	<b>103.73</b>	74.91	-	-	-	-



**Table S17.** Process time (s) for  $K = 80$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-1}$	$\bar{x}$ $1 \times 10^2$	$\sigma$ $1 \times 10^1$	$\bar{x}$	$\sigma$
<b>Method / coef.</b>									-	-
Lstsq	2.9	5.12	2.17	13.28	6.63	1.82	19.42	23.55	-	-
Lstsq (nr)	<b>2.81</b>	3.25	2.17	22.53	<b>7.54</b>	5.2	19.4	25.48	-	-
np.inv	<b>2.38</b>	7.07	<b>1.12</b>	15.77	<b>2.74</b>	1.56	<b>0.64</b>	0.24	-	-
np.inv (nr)	<b>2.34</b>	6.05	<b>1.09</b>	11.02	<b>2.72</b>	1.43	<b>0.62</b>	0.31	-	-
np.solve	<b>2.48</b>	8.07	<b>1.11</b>	13.21	<b>2.71</b>	1.49	<b>0.53</b>	0.19	-	-
np.solve (nr)	<b>2.44</b>	7.84	<b>1.13</b>	19.49	<b>2.71</b>	1.58	<b>0.52</b>	0.23	-	-
np.svd	<b>1.91</b>	1.05	<b>0.99</b>	4.23	<b>3.45</b>	1.13	<b>11.89</b>	19.1	-	-
sp.svd	<b>1.87</b>	0.51	<b>0.97</b>	8.00	<b>3.44</b>	0.9	<b>11.96</b>	19.96	-	-
Cho.Dec	<b>0.29</b>	0.28	<b>0.17</b>	1.12	<b>0.59</b>	0.5	<b>0.58</b>	0.37	-	-
rKrSVD02	<b>0.32</b>	0.18	<b>0.3</b>	1.89	<b>1.04</b>	0.58	<b>0.73</b>	0.12	-	-
rKrSVD04	<b>0.82</b>	0.21	<b>0.69</b>	8.89	<b>2.14</b>	0.83	<b>1.88</b>	0.95	-	-
rKrSVD06	<b>1.77</b>	1.51	<b>0.97</b>	1.73	<b>3.1</b>	0.61	<b>6.44</b>	8.85	-	-
rKrSVD08	<b>2.77</b>	2.35	<b>1.32</b>	4.1	<b>4.48</b>	1.19	<b>10.38</b>	15.63	-	-
rKrSVD10	<b>3.03</b>	0.94	<b>1.61</b>	3.38	<b>5.32</b>	1.43	<b>14.11</b>	22.47	-	-
skrSVD02	<b>0.55</b>	0.43	<b>0.38</b>	2.65	<b>1.12</b>	0.27	<b>0.86</b>	0.18	-	-
skrSVD04	<b>1.14</b>	0.95	<b>0.81</b>	10.46	<b>2.38</b>	0.67	<b>2.1</b>	1.26	-	-
skrSVD06	<b>2.27</b>	2.27	<b>1.05</b>	1.98	<b>3.32</b>	1.1	<b>6.83</b>	9.38	-	-
skrSVD08	<b>2.77</b>	2.79	<b>1.35</b>	3.41	<b>4.64</b>	2.22	<b>10.8</b>	16.22	-	-
skrSVD10	<b>3.29</b>	5.16	<b>1.51</b>	5.56	<b>5.19</b>	0.91	<b>14.56</b>	24.14	-	-
SVDU	<b>35.08</b>	6.2	<b>27.53</b>	163.82	<b>129.56</b>	55.15	-	-	-	-

**Table S18.** Process time (s) for  $K = 90$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ $1 \times 10^{-1}$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^1$	$\sigma$ $1 \times 10^{-1}$	$\bar{x}$ $1 \times 10^2$	$\sigma$ $1 \times 10^1$	$\bar{x}$	$\sigma$
<b>Method / coef.</b>									-	-
Lstsq	3.57	2.6	2.57	14.37	8.24	2.01	28.74	37.33	-	-
Lstsq (nr)	3.58	5.09	<b>2.52</b>	5.86	8.25	3.67	29.04	40.28	-	-
np.inv	<b>2.73</b>	5.36	<b>1.33</b>	9.29	<b>3.15</b>	1.94	<b>0.8</b>	0.4	-	-
np.inv (nr)	<b>2.61</b>	0.81	<b>1.31</b>	10.47	<b>3.11</b>	1.8	<b>0.78</b>	0.45	-	-
np.solve	<b>2.77</b>	3.57	<b>1.36</b>	20.7	<b>3.14</b>	1.89	<b>0.66</b>	0.37	-	-
np.solve (nr)	<b>2.79</b>	8.19	<b>1.31</b>	12.11	<b>3.08</b>	1.91	<b>0.64</b>	0.36	-	-
np.svd	<b>2.12</b>	1.52	<b>1.21</b>	12.51	<b>4.22</b>	1.42	<b>14.37</b>	24.44	-	-
sp.svd	<b>2.08</b>	2.17	<b>1.15</b>	11.00	<b>4.11</b>	2.00	<b>14.33</b>	23.87	-	-
Cho.Dec	<b>0.36</b>	0.17	<b>0.2</b>	0.76	<b>0.76</b>	0.11	<b>0.71</b>	0.46	-	-
rKrSVD02	<b>0.33</b>	0.23	<b>0.33</b>	1.37	<b>1.13</b>	0.24	<b>0.8</b>	0.24	-	-
rKrSVD04	<b>0.87</b>	0.73	<b>0.73</b>	2.3	<b>2.3</b>	0.94	<b>1.98</b>	1.06	-	-
rKrSVD06	<b>1.89</b>	0.49	<b>1.04</b>	6.45	<b>3.4</b>	1.16	<b>7.27</b>	9.87	-	-
rKrSVD08	<b>2.97</b>	2.74	<b>1.42</b>	16.97	<b>4.85</b>	1.13	<b>11.75</b>	18.25	-	-
rKrSVD10	<b>3.33</b>	4.54	<b>1.77</b>	9.05	<b>5.98</b>	1.43	<b>16.58</b>	26.35	-	-
skrSVD02	<b>0.58</b>	0.75	<b>0.4</b>	2.18	<b>1.2</b>	0.73	<b>0.95</b>	0.27	-	-
skrSVD04	<b>1.29</b>	1.5	<b>0.82</b>	1.49	<b>2.47</b>	1.08	<b>2.22</b>	1.37	-	-
skrSVD06	<b>2.45</b>	3.05	<b>1.11</b>	5.12	<b>3.58</b>	0.77	<b>7.67</b>	10.3	-	-
skrSVD08	<b>2.87</b>	4.24	<b>1.47</b>	2.41	<b>4.97</b>	0.87	<b>12.42</b>	19.99	-	-
skrSVD10	<b>3.42</b>	3.04	<b>1.73</b>	3.02	<b>5.98</b>	1.56	<b>17.22</b>	28.6	-	-
SVDU	<b>39.98</b>	4.21	<b>34.13</b>	214.64	<b>156.52</b>	43.84	-	-	-	-

**Table S19.** Process time (s) for  $K = 100$ . Bolded numbers differ statistically from *Lstsq*.

Dataset	BC		BH		AC		CA		CE	
	$\bar{x}$ 1x10 <sup>-1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>1</sup>	$\sigma$ 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>1</sup>	$\sigma$ 1x10 <sup>-1</sup>	$\bar{x}$ 1x10 <sup>2</sup>	$\sigma$ 1x10 <sup>2</sup>	$\bar{x}$	$\sigma$
Lstsq	4.72	3.07	3.05	6.98	9.48	2.51	41.14	6.4	-	-
Lstsq (nr)	<b>4.75</b>	6.44	3.06	6.89	9.44	2.34	40.96	6.03	-	-
np.inv	<b>3.58</b>	8.24	<b>1.57</b>	17.92	<b>3.42</b>	2.04	<b>1.01</b>	0.07	-	-
np.inv (nr)	<b>3.47</b>	8.22	<b>1.55</b>	17.07	<b>3.45</b>	3.02	<b>0.98</b>	0.07	-	-
np.solve	<b>3.65</b>	9.79	<b>1.52</b>	11.55	<b>3.43</b>	2.05	<b>0.83</b>	0.06	-	-
np.solve (nr)	<b>3.62</b>	10.12	<b>1.51</b>	15.6	<b>3.36</b>	2.15	<b>0.81</b>	0.06	-	-
np.svd	<b>2.5</b>	1.63	<b>1.46</b>	4.52	<b>4.89</b>	1.15	<b>17.71</b>	3.17	-	-
sp.svd	<b>2.41</b>	0.61	<b>1.36</b>	7.84	<b>4.74</b>	1.93	<b>17.62</b>	3.27	-	-
Cho.Dec	<b>0.42</b>	0.34	<b>0.26</b>	0.42	<b>0.94</b>	0.38	<b>0.89</b>	0.04	-	-
rKrSVD02	<b>0.35</b>	0.09	<b>0.4</b>	6.79	<b>1.15</b>	0.58	<b>0.88</b>	0.02	-	-
rKrSVD04	<b>0.95</b>	0.28	<b>0.77</b>	2.17	<b>2.42</b>	1.9	<b>2.11</b>	0.13	-	-
rKrSVD06	<b>2.00</b>	0.51	<b>1.15</b>	14.89	<b>3.57</b>	0.84	<b>8.42</b>	1.28	-	-
rKrSVD08	<b>3.1</b>	1.41	<b>1.5</b>	3.82	<b>5.18</b>	0.89	<b>13.52</b>	2.23	-	-
rKrSVD10	<b>3.68</b>	1.82	<b>2.01</b>	6.03	<b>6.66</b>	1.23	<b>19.4</b>	3.29	-	-
skrSVD02	<b>0.59</b>	0.45	<b>0.47</b>	6.26	<b>1.23</b>	0.28	<b>1.03</b>	0.03	-	-
skrSVD04	<b>1.38</b>	1.1	<b>0.89</b>	2.17	<b>2.58</b>	0.88	<b>2.35</b>	0.17	-	-
skrSVD06	<b>2.55</b>	0.8	<b>1.21</b>	8.18	<b>3.81</b>	1.11	<b>8.69</b>	1.33	-	-
skrSVD08	<b>2.99</b>	1.43	<b>1.58</b>	3.19	<b>5.33</b>	0.94	<b>13.81</b>	2.4	-	-
skrSVD10	<b>3.86</b>	3.13	<b>1.96</b>	4.61	<b>6.8</b>	1.85	<b>19.79</b>	3.41	-	-
SVDU	<b>42.67</b>	5.79	<b>37.78</b>	184.8	<b>178.72</b>	43.06	-	-	-	-

6 **3. RMSE and process time tables for additional simulations**

7 Tables Table S20 and Table S21 report the mean and standard deviation of MLM test error measured as  
 8 RMSE value and process time taken in OLS evaluation for the additional simulations with *Census* and  
 9 tables Table S22, Table S23 for *Mnist*. Contrary to other data tables, Table S20, Table S21, Table S22 and  
 10 Table S23 are for one dataset only.

**Table S20.** RMSE for *Census* at reference point percentages [10%, 100%] for *Cholesky Decomposition* (CD) and for random SVD with remaining rank percentages [1%, 10%]. Bolded numbers differ statistically from *Cholesky Decomposition*.

sk_rank_svd_xxx												
Method	CD		001		002		003		004		005	
RefP / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>
10	6.17	3.98	<b>6.44</b>	3.86	<b>6.28</b>	3.81	6.22	3.79	6.19	4.01	6.18	4.07
20	6.15	4.19	<b>6.43</b>	3.89	<b>6.27</b>	3.61	6.19	3.7	6.17	3.88	6.15	3.98
30	6.17	4.41	<b>6.43</b>	3.87	6.26	3.74	6.19	3.8	6.16	3.91	6.14	4.06
40	6.18	4.49	<b>6.43</b>	3.85	6.26	3.72	6.19	3.72	6.16	4.01	6.14	3.96
50	6.19	4.39	<b>6.43</b>	3.98	6.26	3.83	6.19	3.75	6.16	4.02	6.14	3.94
60	6.21	4.41	<b>6.43</b>	3.97	6.26	3.73	6.19	3.73	6.15	4.01	6.14	4.00
70	6.22	4.18	<b>6.43</b>	3.79	6.25	3.78	6.18	3.84	6.15	4.06	6.13	4.05
80	6.23	4.09	<b>6.43</b>	3.89	6.25	3.73	6.18	3.79	6.15	3.97	<b>6.13</b>	3.98
90	6.24	4.03	<b>6.43</b>	4.01	6.25	3.77	6.18	3.91	<b>6.15</b>	4.07	<b>6.13</b>	4.09
100	6.26	4.04	<b>6.43</b>	3.8	6.25	3.76	6.18	3.9	<b>6.15</b>	4.08	<b>6.13</b>	4.08

sk_rank_svd_xxx												
Method	CD		006		007		008		009		010	
RefP / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-6</sup>
10	6.17	3.98	6.17	3.98	6.17	4.01	6.17	3.99	6.17	3.99	6.17	4.1
20	6.15	4.19	6.14	4.07	6.13	4.18	6.13	4.18	6.13	4.27	6.14	4.24
30	6.17	4.41	6.14	4.19	6.13	4.21	6.13	4.28	6.13	4.39	6.13	4.31
40	6.18	4.49	6.14	4.21	6.13	4.21	6.12	4.35	6.12	4.39	6.12	4.44
50	6.19	4.39	6.13	4.09	6.12	4.13	6.12	4.3	6.12	4.26	6.11	4.29
60	6.21	4.41	6.13	4.2	6.12	4.22	6.11	4.37	6.11	4.35	6.11	4.22
70	6.22	4.18	<b>6.12</b>	4.14	<b>6.12</b>	4.27	<b>6.11</b>	4.44	<b>6.11</b>	4.3	<b>6.1</b>	4.44
80	6.23	4.09	<b>6.12</b>	4.05	<b>6.12</b>	4.11	<b>6.11</b>	4.31	<b>6.1</b>	4.35	<b>6.1</b>	4.28
90	6.24	4.03	<b>6.13</b>	4.07	<b>6.11</b>	4.26	<b>6.11</b>	4.32	<b>6.1</b>	4.33	<b>6.1</b>	4.3
100	6.26	4.04	<b>6.12</b>	4.11	<b>6.12</b>	4.13	<b>6.11</b>	4.41	<b>6.1</b>	4.18	<b>6.1</b>	4.32

**Table S21.** Process time in seconds for *Census* at reference point percentages [10%, 100%] for *Cholesky Decomposition* (CD) and for random SVD with remaining rank percentages [1%, 10%]. Bolded numbers differ statistically from *Cholesky Decomposition*.

sk_rank_svd_xxx												
Method	CD		001		002		003		004		005	
RefP / coef.	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^3$
10	0.2	0.003	<b>0.19</b>	0.03	<b>0.28</b>	0.01	<b>0.4</b>	0.001	<b>0.51</b>	0.01	<b>0.65</b>	0.01
20	0.67	0.08	<b>0.39</b>	0.08	<b>0.53</b>	0.01	0.68	0.02	<b>0.83</b>	0.69	<b>1.06</b>	0.15
30	1.48	0.77	<b>0.7</b>	1.14	<b>0.84</b>	0.06	<b>1.08</b>	0.2	<b>1.27</b>	3.45	1.52	0.46
40	2.33	0.67	<b>1.01</b>	2.02	<b>1.25</b>	0.32	<b>1.65</b>	0.72	<b>1.8</b>	11.58	<b>2.03</b>	1.32
50	4.02	2.73	<b>1.51</b>	6.99	<b>1.94</b>	1.58	<b>2.52</b>	1.07	<b>2.53</b>	23.02	<b>3.00</b>	2.56
60	6.04	2.82	<b>2.28</b>	22.99	<b>3.00</b>	1.84	<b>3.46</b>	1.75	<b>3.46</b>	31.76	<b>4.03</b>	2.98
70	7.86	10.91	<b>3.19</b>	72.58	<b>3.68</b>	8.36	<b>4.12</b>	7.32	<b>4.12</b>	82.01	<b>4.67</b>	8.93
80	9.58	3.44	<b>3.26</b>	12.22	<b>3.69</b>	1.85	<b>4.27</b>	2.56	<b>4.21</b>	9.97	<b>4.83</b>	1.91
90	12.87	5.73	<b>4.01</b>	3.44	<b>4.46</b>	1.54	<b>5.13</b>	2.1	<b>5.09</b>	2.31	<b>5.82</b>	2.04
100	15.72	8.17	<b>4.75</b>	21.91	<b>5.07</b>	1.45	<b>5.65</b>	1.83	<b>5.78</b>	2.39	<b>6.46</b>	1.61
sk_rank_svd_xxx												
Method	CD		006		007		008		009		010	
RefP / coef.	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^3$	$1 \times 10^2$	$1 \times 10^3$
10	0.2	0.003	<b>0.78</b>	0.03	<b>0.93</b>	0.74	<b>1.08</b>	0.16	<b>1.25</b>	0.25	<b>1.44</b>	0.45
20	0.67	0.08	<b>1.24</b>	0.28	<b>1.43</b>	4.02	<b>1.64</b>	0.64	<b>1.86</b>	1.03	<b>2.11</b>	1.27
30	1.48	0.77	<b>1.76</b>	0.82	<b>1.98</b>	9.53	<b>2.27</b>	1.55	<b>2.56</b>	2.24	<b>2.84</b>	3.06
40	2.33	0.67	2.37	2.42	<b>2.52</b>	19.17	<b>2.89</b>	2.97	<b>3.21</b>	4.12	<b>3.61</b>	5.66
50	4.02	2.73	<b>3.28</b>	3.03	<b>3.46</b>	31.41	<b>3.73</b>	4.37	3.98	4.64	<b>4.65</b>	6.47
60	6.04	2.82	<b>4.39</b>	4.36	<b>4.41</b>	53.18	<b>4.78</b>	6.23	<b>5.2</b>	6.49	5.84	6.27
70	7.86	10.91	<b>5.04</b>	10.24	<b>5.24</b>	93.42	<b>5.54</b>	9.31	<b>6.21</b>	13.84	<b>6.76</b>	19.2
80	9.58	3.44	<b>5.03</b>	1.13	<b>5.44</b>	12.35	<b>5.76</b>	1.24	<b>6.16</b>	2.4	<b>6.69</b>	2.29
90	12.87	5.73	<b>6.15</b>	2.15	<b>6.4</b>	5.86	<b>6.96</b>	1.67	<b>7.45</b>	2.22	<b>8.00</b>	2.83
100	15.72	8.17	<b>6.82</b>	1.17	<b>7.21</b>	5.36	<b>7.72</b>	0.87	<b>8.28</b>	1.78	<b>8.74</b>	1.79

**Table S22.** RMSE for *Mnist* at reference point percentages [10%, 100%] for *Cholesky Decomposition* (CD) and for random SVD with remaining rank percentages [1%, 10%]. Bolded numbers differ statistically from *Cholesky Decomposition*.

sk_rank_svd_xxx												
Method	CD		001		002		003		004		005	
RefP / coef.	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-1}$	$1 \times 10^{-6}$	$1 \times 10^{-1}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$
10	9.33	1.6	<b>1.19</b>	1.64	<b>1.06</b>	1.56	<b>10.11</b>	1.65	<b>9.86</b>	1.57	<b>9.69</b>	1.59
20	8.67	1.62	<b>1.18</b>	1.78	<b>1.05</b>	1.65	<b>9.95</b>	1.49	<b>9.62</b>	1.51	<b>9.42</b>	1.55
30	8.32	1.61	<b>1.18</b>	1.54	<b>1.05</b>	1.38	<b>9.89</b>	1.62	<b>9.53</b>	1.66	<b>9.3</b>	1.61
40	8.08	1.63	<b>1.18</b>	1.48	<b>1.05</b>	1.47	<b>9.85</b>	1.54	<b>9.49</b>	1.77	<b>9.23</b>	1.52
50	7.91	1.67	<b>1.18</b>	1.71	<b>1.04</b>	1.55	<b>9.83</b>	1.66	<b>9.46</b>	1.69	<b>9.19</b>	1.73
60	7.77	1.75	<b>1.18</b>	1.57	<b>1.04</b>	1.49	<b>9.83</b>	1.56	<b>9.44</b>	1.71	<b>9.17</b>	1.68
70	7.66	1.76	<b>1.18</b>	1.61	<b>1.04</b>	1.65	<b>9.81</b>	1.59	<b>9.43</b>	1.58	<b>9.15</b>	1.6
80	7.56	1.69	<b>1.18</b>	1.48	<b>1.04</b>	1.68	<b>9.81</b>	1.53	<b>9.41</b>	1.71	<b>9.13</b>	1.55
90	7.48	1.73	<b>1.18</b>	1.62	<b>1.04</b>	1.73	<b>9.8</b>	1.56	<b>9.41</b>	1.46	<b>9.12</b>	1.59
100	7.41	1.71	<b>1.18</b>	1.61	<b>1.04</b>	1.59	<b>9.8</b>	1.71	<b>9.41</b>	1.69	<b>9.11</b>	1.67
sk_rank_svd_xxx												
Method	CD		006		007		008		009		010	
RefP / coef.	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$	$\bar{x}$	$\sigma$
	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$	$1 \times 10^{-2}$	$1 \times 10^{-6}$
10	9.33	1.6	<b>9.57</b>	1.6	<b>9.48</b>	1.59	<b>9.42</b>	1.63	9.36	1.62	9.33	1.6
20	8.67	1.62	<b>9.27</b>	1.55	<b>9.16</b>	1.57	<b>9.07</b>	1.52	<b>9.00</b>	1.52	<b>8.95</b>	1.6
30	8.32	1.61	<b>9.13</b>	1.6	<b>9.01</b>	1.59	<b>8.91</b>	1.63	<b>8.84</b>	1.56	<b>8.77</b>	1.59
40	8.08	1.63	<b>9.06</b>	1.46	<b>8.92</b>	1.54	<b>8.82</b>	1.55	<b>8.73</b>	1.52	<b>8.66</b>	1.48
50	7.91	1.67	<b>9.00</b>	1.68	<b>8.86</b>	1.58	<b>8.75</b>	1.6	<b>8.66</b>	1.64	<b>8.59</b>	1.59
60	7.77	1.75	<b>8.97</b>	1.73	<b>8.82</b>	1.55	<b>8.7</b>	1.61	<b>8.61</b>	1.65	<b>8.53</b>	1.59
70	7.66	1.76	<b>8.94</b>	1.63	<b>8.79</b>	1.56	<b>8.67</b>	1.63	<b>8.57</b>	1.63	<b>8.49</b>	1.57
80	7.56	1.69	<b>8.92</b>	1.56	<b>8.76</b>	1.55	<b>8.64</b>	1.55	<b>8.53</b>	1.6	<b>8.45</b>	1.51
90	7.48	1.73	<b>8.91</b>	1.54	<b>8.74</b>	1.5	<b>8.61</b>	1.59	<b>8.51</b>	1.54	<b>8.42</b>	1.52
100	7.41	1.71	<b>8.89</b>	1.56	<b>8.73</b>	1.51	<b>8.59</b>	1.58	<b>8.48</b>	1.56	<b>8.4</b>	1.42

**Table S23.** Process time in seconds for *Mnist* at reference point percentages [10%, 100%] for *Cholesky Decomposition* (CD) and for random SVD with remaining rank percentages [1%, 10%]. Bolded numbers differ statistically from *Cholesky Decomposition*.

sk_rank_svd_xxx												
Method	CD		001		002		003		004		005	
RefP / coef.	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>6</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>4</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>4</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>4</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>4</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>4</sup>
10	0.33	0.00001	<b>0.27</b>	0.002	<b>0.41</b>	0.01	<b>0.57</b>	0.03	<b>0.74</b>	0.09	<b>0.98</b>	0.24
20	1.34	0.003	<b>0.79</b>	1.05	<b>1.01</b>	1.12	<b>1.28</b>	2.04	<b>1.52</b>	3.33	<b>1.84</b>	6.28
30	3.08	0.01	<b>1.48</b>	1.82	<b>1.75</b>	1.23	<b>2.05</b>	1.41	<b>2.38</b>	3.64	<b>2.77</b>	5.88
40	5.68	0.03	<b>2.42</b>	3.18	<b>2.76</b>	3.56	<b>3.14</b>	2.48	<b>3.51</b>	4.44	<b>4.03</b>	8.6
50	9.5	1.41	<b>3.91</b>	51.79	<b>4.26</b>	36.16	<b>4.62</b>	24.57	<b>5.03</b>	14.91	<b>5.6</b>	16.25
60	14.73	3.66	<b>5.4</b>	52.39	<b>5.93</b>	64.04	<b>6.54</b>	89.37	<b>7.17</b>	117.21	<b>7.87</b>	135.29
70	20.25	3.81	<b>6.82</b>	49.46	<b>7.37</b>	30.29	<b>8.09</b>	84.25	<b>8.77</b>	120.31	<b>9.46</b>	113.56
80	27.08	2.96	<b>8.32</b>	0.34	<b>9.00</b>	0.53	<b>9.66</b>	1.23	<b>10.36</b>	2.03	<b>11.12</b>	3.58
90	43.64	131.52	<b>12.36</b>	1167.41	<b>13.02</b>	1133.69	<b>14.34</b>	1660.08	<b>14.79</b>	1646.39	<b>15.23</b>	1575.86
100	52.6	138.29	<b>13.51</b>	225.74	<b>14.14</b>	147.14	<b>14.89</b>	182.74	<b>15.84</b>	230.71	<b>16.81</b>	219.76
sk_rank_svd_xxx												
Method	CD		006		007		008		009		010	
RefP / coef.	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>6</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>5</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>5</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>5</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>5</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>5</sup>
10	0.33	0.00001	<b>1.39</b>	0.06	<b>1.72</b>	0.15	<b>2.06</b>	0.2	<b>2.37</b>	0.28	<b>2.75</b>	0.5
20	1.34	0.003	<b>2.25</b>	1.25	<b>2.7</b>	1.47	<b>3.18</b>	2.26	<b>3.71</b>	2.75	<b>4.38</b>	4.74
30	3.08	0.01	<b>3.27</b>	1.15	<b>3.79</b>	1.33	<b>4.23</b>	1.82	<b>4.89</b>	2.49	<b>5.52</b>	3.49
40	5.68	0.03	<b>4.6</b>	1.61	<b>5.17</b>	1.57	<b>5.85</b>	2.96	<b>6.38</b>	3.54	<b>7.15</b>	5.29
50	9.5	1.41	<b>6.21</b>	4.53	<b>6.86</b>	6.12	<b>7.5</b>	9.34	<b>8.18</b>	8.19	<b>9.19</b>	16.37
60	14.73	3.66	<b>8.62</b>	22.9	<b>9.56</b>	30.78	<b>10.37</b>	37.53	<b>10.99</b>	39.53	<b>12.11</b>	50.69
70	20.25	3.81	<b>10.26</b>	15.76	<b>11.1</b>	15.99	<b>12.04</b>	31.91	<b>13.09</b>	43.39	<b>14.00</b>	34.44
80	27.08	2.96	<b>11.94</b>	0.82	<b>12.81</b>	1.15	<b>13.69</b>	1.32	<b>14.59</b>	1.49	<b>15.69</b>	1.75
90	43.64	131.52	<b>16.12</b>	150.63	<b>16.96</b>	107.47	<b>18.4</b>	162.85	<b>20.02</b>	184.62	<b>20.93</b>	136.39
100	52.6	138.29	<b>17.74</b>	32.43	<b>18.78</b>	20.1	<b>19.83</b>	20.77	<b>21.2</b>	28.97	<b>22.41</b>	29.42



#### 11 4. Deep neural network results

12 Tables S24 and S25 report the mean, standard deviation and the best (minimum) RMSE and process  
 13 time taken of the used DNN on the studied datasets.

**Table S24.** RMSE for neural networks.

Dataset / coef.	FNN-4		FNN-2	
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>
BreastCancer	25.7	22.88	25.52	22.9
BostonHousing	10.23	10.33	9.86	11.09
AirplaneCompanies	6.89	4.4	7.19	4.29
ComputerActivity	2.48	0.62	2.49	0.62
Census	6.7	1.99	6.71	2.00
Mnist	6.1	3.28	8.28	1.54
AuN2-4k	3.18	1.00	3.42	0.83
AuN2-8k	2.67	0.68	2.97	0.73
AuN2-12k	2.15	0.73	3.42	0.83
AuN10-4k	1.17	0.45	1.21	0.47
AuN10-8k	0.85	0.21	0.88	0.29
AuN10-12k	1.4	6.22	15.49	207.22
AuN100-4k	1.96	1.35	38.75	156.31
AuN100-8k	1.83	1.28	37.6	193.52
AuN100-12k	1.96	1.35	38.75	156.31

**Table S25.** Process time (s) for neural networks.

Dataset / coef.	FNN-4		FNN-2	
	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>2</sup>	$\bar{x}$ 1x10 <sup>3</sup>	$\sigma$ 1x10 <sup>2</sup>
BreastCancer	0.06	0.01	0.06	0.01
BostonHousing	0.15	0.02	0.13	0.03
AirplaneCompanies	0.23	0.02	0.18	0.06
ComputerActivity	2.4	0.51	2.22	0.21
Census	4.54	1.08	3.97	1.3
Mnist	28.4	23.94	43.36	9.47
AuN2-4k	1.05	0.36	1.00	0.4
AuN2-8k	2.22	0.51	2.15	0.34
AuN2-12k	3.53	0.44	1.00	0.4
AuN10-4k	1.86	1.24	1.39	1.22
AuN10-8k	4.21	1.87	3.16	1.41
AuN10-12k	18.74	108.95	3.34	23.19
AuN100-4k	24.03	61.34	6.16	7.29
AuN100-8k	47.11	82.71	10.35	30.74
AuN100-12k	24.03	61.34	6.17	7.13

#### 14 5. Au<sub>38Q</sub> RMSE and process time tables

15 Tables S26–S35 report the normalized test RMSE of MLM on the variants of Au<sub>38Q</sub> dataset. Similarly

16 Tables S36–S45 report the process time taken in OLS evaluation.

**Table S26.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 10$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	3.27	1.54	2.97	2.39	7.67	2.23	1.89	6.22	1.76
sk_rrSVD_001	<b>5.3</b>	1.98	4.9	<b>4.46</b>	7.15	4.32	<b>3.93</b>	9.44	3.71
sk_rrSVD_002	<b>4.56</b>	1.62	4.08	<b>3.7</b>	7.61	3.5	<b>3.09</b>	7.59	2.96
sk_rrSVD_003	<b>4.17</b>	1.49	3.79	<b>3.22</b>	8.77	2.97	<b>2.61</b>	6.96	2.45
sk_rrSVD_004	<b>3.91</b>	1.55	3.5	<b>2.92</b>	8.44	2.64	<b>2.32</b>	7.5	2.15
sk_rrSVD_005	<b>3.7</b>	1.5	3.33	<b>2.72</b>	8.2	2.5	<b>2.16</b>	7.04	1.98
sk_rrSVD_006	<b>3.54</b>	1.52	3.19	<b>2.59</b>	7.65	2.4	<b>2.05</b>	7.02	1.89
sk_rrSVD_007	<b>3.43</b>	1.56	3.09	<b>2.51</b>	8.04	2.33	<b>1.98</b>	6.48	1.85
sk_rrSVD_008	<b>3.35</b>	1.61	3.04	<b>2.45</b>	7.81	2.28	<b>1.94</b>	6.37	1.8
sk_rrSVD_009	3.3	1.55	3.00	2.42	7.95	2.24	1.91	6.38	1.78
sk_rrSVD_010	3.27	1.54	2.97	2.39	7.67	2.23	1.89	6.25	1.76
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.99	7.85	1.85	1.52	5.48	1.41	1.21	3.2	1.13
sk_rrSVD_001	<b>2.94</b>	9.79	2.75	<b>2.47</b>	5.92	2.34	<b>2.17</b>	4.22	2.1
sk_rrSVD_002	<b>2.59</b>	9.94	2.41	<b>2.12</b>	5.14	2.00	<b>1.8</b>	3.17	1.71
sk_rrSVD_003	<b>2.39</b>	9.04	2.24	<b>1.94</b>	4.65	1.85	<b>1.61</b>	3.03	1.55
sk_rrSVD_004	<b>2.26</b>	8.76	2.13	<b>1.81</b>	4.91	1.74	<b>1.49</b>	2.98	1.39
sk_rrSVD_005	<b>2.19</b>	8.61	2.01	<b>1.72</b>	5.47	1.6	<b>1.4</b>	3.15	1.32
sk_rrSVD_006	<b>2.13</b>	8.36	1.96	<b>1.65</b>	5.16	1.56	<b>1.34</b>	2.95	1.27
sk_rrSVD_007	<b>2.08</b>	8.26	1.94	<b>1.6</b>	5.51	1.49	<b>1.29</b>	3.15	1.22
sk_rrSVD_008	<b>2.04</b>	7.81	1.91	<b>1.57</b>	5.41	1.45	<b>1.25</b>	3.24	1.18
sk_rrSVD_009	2.01	7.92	1.88	1.54	5.5	1.42	<b>1.22</b>	3.18	1.15
sk_rrSVD_010	1.99	7.85	1.85	1.52	5.48	1.41	1.21	3.2	1.13
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	2.00	7.5	1.83	1.53	5.15	1.45	1.21	3.56	1.15
sk_rrSVD_001	<b>2.98</b>	8.82	2.8	<b>2.5</b>	6.08	2.37	<b>2.17</b>	4.83	2.08
sk_rrSVD_002	<b>2.61</b>	8.22	2.44	<b>2.13</b>	5.57	2.01	<b>1.8</b>	4.32	1.68
sk_rrSVD_003	<b>2.4</b>	7.94	2.26	<b>1.95</b>	5.27	1.85	<b>1.61</b>	3.62	1.52
sk_rrSVD_004	<b>2.28</b>	7.44	2.13	<b>1.83</b>	4.45	1.74	<b>1.49</b>	3.54	1.41
sk_rrSVD_005	<b>2.19</b>	7.55	2.03	<b>1.73</b>	4.79	1.66	<b>1.4</b>	3.51	1.31
sk_rrSVD_006	<b>2.13</b>	7.72	1.96	<b>1.67</b>	4.94	1.58	<b>1.33</b>	3.58	1.25
sk_rrSVD_007	<b>2.08</b>	7.33	1.92	<b>1.62</b>	5.2	1.53	<b>1.28</b>	3.55	1.22
sk_rrSVD_008	<b>2.05</b>	7.38	1.9	<b>1.58</b>	5.02	1.51	<b>1.25</b>	3.52	1.19
sk_rrSVD_009	2.02	7.71	1.85	1.55	5.21	1.47	1.22	3.56	1.16
sk_rrSVD_010	2.00	7.5	1.83	1.53	5.15	1.45	1.21	3.56	1.15

**Table S27.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 20$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	2.55	1.17	2.31	1.71	4.92	1.63	1.28	3.75	1.21
sk_rrSVD_001	<b>5.23</b>	1.83	4.81	<b>4.44</b>	8.2	4.26	<b>3.89</b>	8.89	3.68
sk_rrSVD_002	<b>4.48</b>	1.23	4.2	<b>3.65</b>	7.35	3.49	<b>3.01</b>	7.63	2.83
sk_rrSVD_003	<b>4.08</b>	1.25	3.78	<b>3.14</b>	8.54	2.94	<b>2.47</b>	6.73	2.34
sk_rrSVD_004	<b>3.79</b>	1.33	3.52	<b>2.77</b>	7.18	2.59	<b>2.15</b>	5.06	2.05
sk_rrSVD_005	<b>3.55</b>	1.47	3.19	<b>2.52</b>	6.62	2.39	<b>1.93</b>	4.8	1.85
sk_rrSVD_006	<b>3.36</b>	1.28	3.11	<b>2.35</b>	5.95	2.25	<b>1.78</b>	4.81	1.68
sk_rrSVD_007	<b>3.19</b>	1.38	2.93	<b>2.21</b>	6.06	2.09	<b>1.66</b>	4.12	1.56
sk_rrSVD_008	<b>3.07</b>	1.21	2.84	<b>2.1</b>	5.78	1.99	<b>1.57</b>	4.08	1.48
sk_rrSVD_009	<b>2.96</b>	1.27	2.69	<b>2.02</b>	5.41	1.91	<b>1.5</b>	4.27	1.41
sk_rrSVD_010	<b>2.87</b>	1.35	2.63	<b>1.95</b>	5.92	1.85	<b>1.45</b>	4.05	1.38
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.62	6.15	1.51	1.14	3.87	1.05	0.88	2.33	0.83
sk_rrSVD_001	<b>2.88</b>	9.65	2.69	<b>2.44</b>	5.49	2.34	<b>2.15</b>	3.48	2.06
sk_rrSVD_002	<b>2.55</b>	8.9	2.37	<b>2.07</b>	5.04	1.97	<b>1.77</b>	3.21	1.69
sk_rrSVD_003	<b>2.33</b>	6.85	2.22	<b>1.89</b>	4.00	1.82	<b>1.57</b>	2.41	1.53
sk_rrSVD_004	<b>2.19</b>	6.72	2.01	<b>1.75</b>	3.94	1.67	<b>1.45</b>	2.83	1.39
sk_rrSVD_005	<b>2.09</b>	6.7	1.95	<b>1.64</b>	3.73	1.53	<b>1.34</b>	2.78	1.26
sk_rrSVD_006	<b>2.02</b>	6.44	1.91	<b>1.55</b>	3.65	1.43	<b>1.24</b>	2.75	1.17
sk_rrSVD_007	<b>1.96</b>	6.71	1.83	<b>1.47</b>	3.47	1.37	<b>1.17</b>	2.8	1.08
sk_rrSVD_008	<b>1.9</b>	6.51	1.77	<b>1.41</b>	3.24	1.33	<b>1.11</b>	2.44	1.05
sk_rrSVD_009	<b>1.85</b>	6.42	1.73	<b>1.36</b>	3.09	1.28	<b>1.07</b>	2.42	1.01
sk_rrSVD_010	<b>1.81</b>	6.6	1.68	<b>1.32</b>	3.15	1.25	<b>1.02</b>	2.14	0.98
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.63	6.16	1.53	1.15	4.07	1.03	0.87	2.96	0.82
sk_rrSVD_001	<b>2.94</b>	8.98	2.73	<b>2.47</b>	5.68	2.35	<b>2.15</b>	4.89	2.06
sk_rrSVD_002	<b>2.57</b>	7.74	2.42	<b>2.08</b>	5.15	1.97	<b>1.76</b>	3.15	1.72
sk_rrSVD_003	<b>2.34</b>	7.37	2.24	<b>1.89</b>	4.83	1.8	<b>1.57</b>	3.2	1.51
sk_rrSVD_004	<b>2.19</b>	6.01	2.07	<b>1.76</b>	3.96	1.67	<b>1.44</b>	3.23	1.37
sk_rrSVD_005	<b>2.1</b>	5.57	1.98	<b>1.65</b>	4.12	1.57	<b>1.33</b>	3.52	1.26
sk_rrSVD_006	<b>2.03</b>	6.29	1.89	<b>1.56</b>	4.17	1.49	<b>1.24</b>	3.51	1.17
sk_rrSVD_007	<b>1.97</b>	5.98	1.84	<b>1.47</b>	3.96	1.39	<b>1.16</b>	2.99	1.11
sk_rrSVD_008	<b>1.91</b>	6.37	1.78	<b>1.42</b>	3.66	1.35	<b>1.11</b>	3.06	1.05
sk_rrSVD_009	<b>1.86</b>	6.26	1.76	<b>1.37</b>	3.66	1.26	<b>1.06</b>	2.92	0.99
sk_rrSVD_010	<b>1.82</b>	6.07	1.71	<b>1.33</b>	3.64	1.23	<b>1.02</b>	2.81	0.96

**Table S28.** Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 30$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-3}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$
cho_dec	2.17	0.93	1.99	1.4	4.73	1.29	1.02	3.4	0.94
sk_rrSVD_001	<b>5.2</b>	1.97	4.73	<b>4.4</b>	8.09	4.27	<b>3.88</b>	8.11	3.7
sk_rrSVD_002	<b>4.46</b>	1.33	4.14	<b>3.62</b>	6.94	3.52	<b>2.99</b>	8.03	2.78
sk_rrSVD_003	<b>4.06</b>	1.43	3.74	<b>3.11</b>	7.1	2.97	<b>2.42</b>	6.16	2.32
sk_rrSVD_004	<b>3.74</b>	1.18	3.51	<b>2.71</b>	6.88	2.54	<b>2.09</b>	5.49	1.96
sk_rrSVD_005	<b>3.52</b>	1.16	3.26	<b>2.44</b>	5.66	2.33	<b>1.87</b>	5.13	1.76
sk_rrSVD_006	<b>3.32</b>	1.12	3.11	<b>2.28</b>	5.73	2.15	<b>1.7</b>	4.22	1.6
sk_rrSVD_007	<b>3.12</b>	1.19	2.83	<b>2.13</b>	5.24	2.00	<b>1.56</b>	4.36	1.48
sk_rrSVD_008	<b>2.97</b>	1.16	2.7	<b>2.02</b>	3.98	1.93	<b>1.46</b>	3.58	1.39
sk_rrSVD_009	<b>2.85</b>	1.14	2.6	<b>1.92</b>	5.27	1.8	<b>1.39</b>	3.31	1.33
sk_rrSVD_010	<b>2.74</b>	1.08	2.48	<b>1.84</b>	4.75	1.74	<b>1.33</b>	3.51	1.26
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$
cho_dec	1.42	4.34	1.34	0.96	2.91	0.9	0.72	1.66	0.68
sk_rrSVD_001	<b>2.86</b>	9.84	2.67	<b>2.43</b>	4.53	2.35	<b>2.14</b>	3.99	2.05
sk_rrSVD_002	<b>2.52</b>	7.88	2.4	<b>2.06</b>	4.6	1.96	<b>1.76</b>	3.01	1.7
sk_rrSVD_003	<b>2.31</b>	6.66	2.19	<b>1.87</b>	4.31	1.79	<b>1.57</b>	2.67	1.51
sk_rrSVD_004	<b>2.17</b>	6.33	2.07	<b>1.74</b>	4.14	1.64	<b>1.43</b>	2.81	1.37
sk_rrSVD_005	<b>2.06</b>	6.52	1.96	<b>1.62</b>	3.25	1.53	<b>1.32</b>	2.57	1.25
sk_rrSVD_006	<b>1.99</b>	5.95	1.89	<b>1.52</b>	3.2	1.45	<b>1.22</b>	2.68	1.13
sk_rrSVD_007	<b>1.92</b>	6.79	1.82	<b>1.44</b>	2.51	1.37	<b>1.14</b>	2.21	1.06
sk_rrSVD_008	<b>1.87</b>	6.31	1.74	<b>1.37</b>	2.76	1.31	<b>1.08</b>	2.23	1.02
sk_rrSVD_009	<b>1.81</b>	5.81	1.68	<b>1.31</b>	2.33	1.26	<b>1.02</b>	1.93	0.96
sk_rrSVD_010	<b>1.76</b>	6.23	1.65	<b>1.27</b>	2.23	1.22	<b>0.97</b>	1.75	0.92
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$	$\bar{x}$ $1 \times 10^{-2}$	$\sigma$ $1 \times 10^{-4}$	$\min(x)$ $1 \times 10^{-2}$
cho_dec	1.43	5.27	1.34	0.96	2.64	0.88	0.72	2.16	0.68
sk_rrSVD_001	<b>2.9</b>	7.49	2.69	<b>2.46</b>	5.57	2.34	<b>2.14</b>	4.28	2.05
sk_rrSVD_002	<b>2.54</b>	7.74	2.38	<b>2.07</b>	4.5	2.00	<b>1.74</b>	3.66	1.69
sk_rrSVD_003	<b>2.3</b>	6.01	2.21	<b>1.88</b>	3.41	1.81	<b>1.55</b>	3.11	1.49
sk_rrSVD_004	<b>2.16</b>	5.44	2.04	<b>1.74</b>	3.58	1.66	<b>1.42</b>	3.31	1.36
sk_rrSVD_005	<b>2.07</b>	5.31	1.97	<b>1.63</b>	3.27	1.56	<b>1.31</b>	3.03	1.24
sk_rrSVD_006	<b>2.00</b>	4.8	1.91	<b>1.53</b>	2.8	1.48	<b>1.21</b>	2.93	1.15
sk_rrSVD_007	<b>1.93</b>	4.97	1.85	<b>1.44</b>	3.24	1.39	<b>1.13</b>	2.47	1.09
sk_rrSVD_008	<b>1.87</b>	5.68	1.77	<b>1.38</b>	2.92	1.32	<b>1.07</b>	2.39	1.02
sk_rrSVD_009	<b>1.82</b>	5.62	1.73	<b>1.32</b>	2.46	1.26	<b>1.01</b>	2.34	0.96
sk_rrSVD_010	<b>1.77</b>	5.66	1.68	<b>1.27</b>	2.46	1.22	<b>0.96</b>	2.37	0.9

**Table S29.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 40$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.95	0.82	1.78	1.21	3.67	1.14	0.88	2.4	0.83
sk_rrSVD_001	<b>5.19</b>	1.74	4.76	<b>4.4</b>	7.99	4.24	<b>3.87</b>	8.84	3.69
sk_rrSVD_002	<b>4.45</b>	1.23	4.2	<b>3.62</b>	6.91	3.49	<b>2.96</b>	6.92	2.81
sk_rrSVD_003	<b>4.04</b>	1.33	3.71	<b>3.08</b>	6.85	2.94	<b>2.41</b>	5.83	2.24
sk_rrSVD_004	<b>3.72</b>	1.24	3.5	<b>2.68</b>	7.86	2.51	<b>2.06</b>	4.71	1.96
sk_rrSVD_005	<b>3.49</b>	1.08	3.23	<b>2.42</b>	6.03	2.32	<b>1.84</b>	4.66	1.73
sk_rrSVD_006	<b>3.28</b>	1.19	3.04	<b>2.23</b>	5.36	2.1	<b>1.66</b>	3.92	1.59
sk_rrSVD_007	<b>3.07</b>	1.17	2.77	<b>2.09</b>	4.33	2.01	<b>1.53</b>	3.74	1.44
sk_rrSVD_008	<b>2.92</b>	1.11	2.66	<b>1.97</b>	4.68	1.84	<b>1.43</b>	3.41	1.38
sk_rrSVD_009	<b>2.8</b>	1.05	2.57	<b>1.87</b>	4.57	1.78	<b>1.35</b>	2.95	1.3
sk_rrSVD_010	<b>2.69</b>	1.07	2.44	<b>1.79</b>	5.00	1.69	<b>1.28</b>	2.86	1.23
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.3	4.45	1.22	0.85	2.54	0.8	0.64	1.47	0.61
sk_rrSVD_001	<b>2.85</b>	8.26	2.69	<b>2.42</b>	4.43	2.33	<b>2.13</b>	3.82	2.07
sk_rrSVD_002	<b>2.52</b>	7.51	2.38	<b>2.05</b>	4.51	1.96	<b>1.76</b>	3.09	1.7
sk_rrSVD_003	<b>2.3</b>	5.88	2.19	<b>1.87</b>	3.93	1.79	<b>1.56</b>	2.47	1.5
sk_rrSVD_004	<b>2.14</b>	5.18	2.06	<b>1.73</b>	3.54	1.62	<b>1.43</b>	2.43	1.38
sk_rrSVD_005	<b>2.05</b>	5.75	1.94	<b>1.61</b>	3.18	1.55	<b>1.31</b>	2.71	1.25
sk_rrSVD_006	<b>1.97</b>	5.65	1.87	<b>1.51</b>	3.49	1.43	<b>1.2</b>	2.37	1.15
sk_rrSVD_007	<b>1.91</b>	5.39	1.81	<b>1.42</b>	3.08	1.36	<b>1.13</b>	1.99	1.07
sk_rrSVD_008	<b>1.85</b>	5.88	1.69	<b>1.35</b>	2.55	1.31	<b>1.06</b>	1.88	1.01
sk_rrSVD_009	<b>1.79</b>	5.61	1.68	<b>1.29</b>	2.23	1.26	<b>1.00</b>	2.02	0.96
sk_rrSVD_010	<b>1.74</b>	5.71	1.62	<b>1.25</b>	2.2	1.2	<b>0.95</b>	2.01	0.91
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.3	4.35	1.23	0.85	2.24	0.79	0.63	1.78	0.59
sk_rrSVD_001	<b>2.88</b>	7.24	2.72	<b>2.46</b>	6.01	2.33	<b>2.14</b>	4.46	2.05
sk_rrSVD_002	<b>2.53</b>	6.57	2.41	<b>2.05</b>	4.89	1.93	<b>1.75</b>	3.3	1.69
sk_rrSVD_003	<b>2.29</b>	6.05	2.17	<b>1.87</b>	4.29	1.81	<b>1.54</b>	3.47	1.47
sk_rrSVD_004	<b>2.15</b>	5.5	2.03	<b>1.74</b>	3.7	1.67	<b>1.42</b>	2.85	1.35
sk_rrSVD_005	<b>2.05</b>	5.55	1.96	<b>1.61</b>	3.35	1.56	<b>1.3</b>	2.78	1.22
sk_rrSVD_006	<b>1.99</b>	4.88	1.91	<b>1.51</b>	2.87	1.45	<b>1.2</b>	2.41	1.15
sk_rrSVD_007	<b>1.92</b>	4.89	1.84	<b>1.42</b>	3.27	1.37	<b>1.12</b>	2.23	1.07
sk_rrSVD_008	<b>1.85</b>	5.35	1.76	<b>1.36</b>	2.75	1.32	<b>1.05</b>	2.31	1.01
sk_rrSVD_009	<b>1.79</b>	5.2	1.71	<b>1.3</b>	2.6	1.26	<b>0.99</b>	2.26	0.95
sk_rrSVD_010	<b>1.75</b>	5.11	1.67	<b>1.25</b>	2.68	1.2	<b>0.94</b>	2.24	0.9

**Table S30.** Normalized RMSE mean, standard deviation and best result for  $Au_{38Q}$   $K = 50$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-3}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.79	0.71	1.67	1.09	3.2	1.04	0.8	2.07	0.76
sk_rrSVD_001	<b>5.18</b>	1.71	4.76	<b>4.38</b>	7.36	4.2	<b>3.87</b>	7.28	3.71
sk_rrSVD_002	<b>4.43</b>	1.44	4.1	<b>3.61</b>	6.63	3.49	<b>2.96</b>	7.24	2.81
sk_rrSVD_003	<b>4.02</b>	1.32	3.67	<b>3.07</b>	6.98	2.94	<b>2.4</b>	5.94	2.26
sk_rrSVD_004	<b>3.71</b>	1.32	3.46	<b>2.66</b>	6.07	2.5	<b>2.05</b>	4.3	1.93
sk_rrSVD_005	<b>3.47</b>	1.17	3.18	<b>2.4</b>	6.05	2.25	<b>1.82</b>	3.95	1.75
sk_rrSVD_006	<b>3.26</b>	1.23	2.95	<b>2.21</b>	5.64	2.06	<b>1.64</b>	3.14	1.59
sk_rrSVD_007	<b>3.07</b>	1.11	2.78	<b>2.07</b>	4.45	1.96	<b>1.51</b>	3.05	1.44
sk_rrSVD_008	<b>2.9</b>	1.07	2.6	<b>1.96</b>	3.81	1.87	<b>1.41</b>	2.67	1.34
sk_rrSVD_009	<b>2.76</b>	0.99	2.53	<b>1.84</b>	3.96	1.76	<b>1.32</b>	2.56	1.27
sk_rrSVD_010	<b>2.65</b>	1.02	2.39	<b>1.76</b>	3.68	1.66	<b>1.26</b>	2.71	1.21
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.21	4.34	1.11	0.78	2.14	0.75	0.6	1.29	0.58
sk_rrSVD_001	<b>2.83</b>	7.67	2.68	<b>2.42</b>	4.46	2.32	<b>2.14</b>	3.76	2.06
sk_rrSVD_002	<b>2.52</b>	6.51	2.37	<b>2.05</b>	4.92	1.96	<b>1.75</b>	3.18	1.7
sk_rrSVD_003	<b>2.28</b>	6.11	2.18	<b>1.87</b>	3.94	1.79	<b>1.56</b>	2.55	1.5
sk_rrSVD_004	<b>2.14</b>	5.36	2.05	<b>1.73</b>	3.71	1.64	<b>1.42</b>	2.08	1.38
sk_rrSVD_005	<b>2.04</b>	4.99	1.93	<b>1.61</b>	2.88	1.54	<b>1.3</b>	2.62	1.24
sk_rrSVD_006	<b>1.97</b>	5.46	1.86	<b>1.5</b>	3.25	1.43	<b>1.2</b>	2.55	1.14
sk_rrSVD_007	<b>1.9</b>	5.4	1.78	<b>1.41</b>	2.84	1.35	<b>1.11</b>	2.11	1.07
sk_rrSVD_008	<b>1.84</b>	5.65	1.73	<b>1.34</b>	2.52	1.29	<b>1.05</b>	2.07	1.00
sk_rrSVD_009	<b>1.77</b>	5.64	1.65	<b>1.28</b>	2.07	1.24	<b>0.99</b>	2.28	0.93
sk_rrSVD_010	<b>1.72</b>	4.93	1.62	<b>1.23</b>	1.93	1.19	<b>0.94</b>	1.89	0.9
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.21	3.84	1.15	0.79	2.26	0.73	0.58	1.61	0.54
sk_rrSVD_001	<b>2.89</b>	8.37	2.71	<b>2.45</b>	5.12	2.36	<b>2.14</b>	3.78	2.06
sk_rrSVD_002	<b>2.52</b>	6.48	2.4	<b>2.05</b>	5.03	1.95	<b>1.74</b>	2.83	1.69
sk_rrSVD_003	<b>2.29</b>	6.04	2.13	<b>1.86</b>	3.93	1.8	<b>1.54</b>	3.08	1.48
sk_rrSVD_004	<b>2.14</b>	4.69	2.03	<b>1.72</b>	3.86	1.64	<b>1.42</b>	2.69	1.37
sk_rrSVD_005	<b>2.05</b>	5.29	1.96	<b>1.61</b>	3.79	1.56	<b>1.3</b>	2.39	1.25
sk_rrSVD_006	<b>1.97</b>	4.54	1.88	<b>1.51</b>	3.01	1.45	<b>1.19</b>	2.6	1.14
sk_rrSVD_007	<b>1.91</b>	5.09	1.79	<b>1.42</b>	3.07	1.35	<b>1.11</b>	2.14	1.06
sk_rrSVD_008	<b>1.85</b>	4.62	1.75	<b>1.34</b>	2.83	1.3	<b>1.05</b>	2.15	1.00
sk_rrSVD_009	<b>1.78</b>	4.41	1.71	<b>1.28</b>	2.81	1.22	<b>0.99</b>	2.13	0.94
sk_rrSVD_010	<b>1.73</b>	4.86	1.66	<b>1.23</b>	2.53	1.19	<b>0.93</b>	2.11	0.89



**Table S31.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 60$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.68	0.66	1.58	1.01	3.07	0.96	0.74	2.11	0.69
sk_rrSVD_001	<b>5.17</b>	1.62	4.73	<b>4.39</b>	7.84	4.26	<b>3.87</b>	7.33	3.72
sk_rrSVD_002	<b>4.43</b>	1.33	4.04	<b>3.61</b>	6.4	3.49	<b>2.94</b>	6.38	2.81
sk_rrSVD_003	<b>4.02</b>	1.21	3.71	<b>3.07</b>	5.83	2.97	<b>2.4</b>	5.35	2.27
sk_rrSVD_004	<b>3.7</b>	1.31	3.45	<b>2.66</b>	6.09	2.52	<b>2.04</b>	3.77	1.92
sk_rrSVD_005	<b>3.46</b>	1.12	3.25	<b>2.38</b>	5.71	2.23	<b>1.8</b>	4.23	1.71
sk_rrSVD_006	<b>3.25</b>	1.22	2.96	<b>2.2</b>	5.9	2.03	<b>1.64</b>	3.53	1.55
sk_rrSVD_007	<b>3.03</b>	1.13	2.77	<b>2.05</b>	5.05	1.91	<b>1.5</b>	3.19	1.43
sk_rrSVD_008	<b>2.88</b>	1.15	2.57	<b>1.94</b>	4.41	1.86	<b>1.39</b>	2.52	1.34
sk_rrSVD_009	<b>2.74</b>	0.99	2.48	<b>1.83</b>	3.47	1.75	<b>1.31</b>	2.49	1.25
sk_rrSVD_010	<b>2.62</b>	1.01	2.41	<b>1.74</b>	3.86	1.64	<b>1.25</b>	2.59	1.19
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.14	3.46	1.07	0.74	1.86	0.71	0.57	1.23	0.54
sk_rrSVD_001	<b>2.83</b>	7.82	2.68	<b>2.42</b>	4.34	2.32	<b>2.13</b>	3.76	2.07
sk_rrSVD_002	<b>2.51</b>	6.65	2.36	<b>2.06</b>	4.72	1.97	<b>1.75</b>	2.99	1.69
sk_rrSVD_003	<b>2.28</b>	6.00	2.16	<b>1.87</b>	4.07	1.79	<b>1.55</b>	2.17	1.51
sk_rrSVD_004	<b>2.13</b>	5.3	2.02	<b>1.72</b>	3.69	1.65	<b>1.42</b>	2.36	1.37
sk_rrSVD_005	<b>2.03</b>	4.93	1.94	<b>1.6</b>	3.12	1.53	<b>1.3</b>	2.3	1.26
sk_rrSVD_006	<b>1.97</b>	5.13	1.86	<b>1.5</b>	3.37	1.43	<b>1.19</b>	2.32	1.15
sk_rrSVD_007	<b>1.89</b>	5.66	1.79	<b>1.41</b>	2.81	1.36	<b>1.11</b>	1.76	1.07
sk_rrSVD_008	<b>1.84</b>	5.34	1.74	<b>1.33</b>	2.55	1.28	<b>1.04</b>	1.97	0.99
sk_rrSVD_009	<b>1.77</b>	5.35	1.66	<b>1.28</b>	2.34	1.24	<b>0.98</b>	1.88	0.94
sk_rrSVD_010	<b>1.71</b>	5.17	1.62	<b>1.22</b>	2.15	1.17	<b>0.93</b>	1.89	0.9
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.14	3.46	1.08	0.74	2.1	0.69	0.55	1.6	0.51
sk_rrSVD_001	<b>2.87</b>	8.7	2.72	<b>2.45</b>	4.57	2.37	<b>2.13</b>	3.58	2.06
sk_rrSVD_002	<b>2.51</b>	5.35	2.43	<b>2.04</b>	4.53	1.96	<b>1.74</b>	2.89	1.68
sk_rrSVD_003	<b>2.28</b>	5.66	2.14	<b>1.86</b>	3.95	1.79	<b>1.54</b>	3.37	1.47
sk_rrSVD_004	<b>2.14</b>	4.56	2.04	<b>1.73</b>	3.46	1.65	<b>1.41</b>	2.58	1.35
sk_rrSVD_005	<b>2.05</b>	5.28	1.94	<b>1.6</b>	3.38	1.54	<b>1.29</b>	2.5	1.24
sk_rrSVD_006	<b>1.97</b>	5.24	1.89	<b>1.5</b>	2.73	1.45	<b>1.18</b>	2.21	1.14
sk_rrSVD_007	<b>1.91</b>	4.62	1.82	<b>1.41</b>	3.08	1.36	<b>1.1</b>	2.13	1.06
sk_rrSVD_008	<b>1.84</b>	4.31	1.75	<b>1.33</b>	2.96	1.27	<b>1.04</b>	2.06	0.99
sk_rrSVD_009	<b>1.78</b>	4.78	1.68	<b>1.28</b>	2.86	1.21	<b>0.98</b>	2.14	0.94
sk_rrSVD_010	<b>1.72</b>	4.99	1.64	<b>1.22</b>	2.53	1.17	<b>0.92</b>	1.79	0.89

**Table S32.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 70$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.59	0.62	1.47	0.95	2.86	0.89	0.7	1.78	0.66
sk_rrSVD_001	<b>5.18</b>	1.57	4.9	<b>4.37</b>	7.85	4.16	<b>3.86</b>	6.43	3.71
sk_rrSVD_002	<b>4.43</b>	1.51	4.08	<b>3.6</b>	6.61	3.41	<b>2.96</b>	7.09	2.84
sk_rrSVD_003	<b>4.03</b>	1.17	3.79	<b>3.05</b>	7.3	2.82	<b>2.39</b>	4.83	2.3
sk_rrSVD_004	<b>3.68</b>	1.26	3.44	<b>2.64</b>	5.43	2.51	<b>2.03</b>	4.5	1.91
sk_rrSVD_005	<b>3.46</b>	1.19	3.21	<b>2.37</b>	5.49	2.25	<b>1.8</b>	3.97	1.73
sk_rrSVD_006	<b>3.24</b>	1.23	3.00	<b>2.19</b>	5.6	2.04	<b>1.63</b>	2.95	1.56
sk_rrSVD_007	<b>3.02</b>	1.33	2.72	<b>2.05</b>	4.44	1.96	<b>1.49</b>	3.1	1.42
sk_rrSVD_008	<b>2.86</b>	1.09	2.65	<b>1.92</b>	4.02	1.82	<b>1.39</b>	2.98	1.33
sk_rrSVD_009	<b>2.72</b>	0.99	2.53	<b>1.81</b>	3.63	1.73	<b>1.3</b>	2.31	1.25
sk_rrSVD_010	<b>2.61</b>	0.96	2.41	<b>1.73</b>	3.72	1.65	<b>1.24</b>	2.74	1.18
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.09	3.5	1.03	0.71	1.78	0.68	0.55	1.05	0.53
sk_rrSVD_001	<b>2.82</b>	8.33	2.68	<b>2.42</b>	4.25	2.34	<b>2.13</b>	3.87	2.06
sk_rrSVD_002	<b>2.51</b>	6.51	2.4	<b>2.06</b>	4.5	1.97	<b>1.75</b>	3.01	1.67
sk_rrSVD_003	<b>2.27</b>	5.83	2.17	<b>1.87</b>	3.42	1.8	<b>1.55</b>	2.35	1.5
sk_rrSVD_004	<b>2.13</b>	4.9	2.04	<b>1.72</b>	3.49	1.66	<b>1.42</b>	2.07	1.38
sk_rrSVD_005	<b>2.04</b>	4.92	1.95	<b>1.6</b>	3.37	1.5	<b>1.3</b>	2.04	1.26
sk_rrSVD_006	<b>1.97</b>	5.62	1.86	<b>1.5</b>	3.27	1.42	<b>1.19</b>	2.13	1.14
sk_rrSVD_007	<b>1.89</b>	6.35	1.75	<b>1.4</b>	3.12	1.33	<b>1.1</b>	1.87	1.06
sk_rrSVD_008	<b>1.83</b>	5.33	1.72	<b>1.33</b>	2.83	1.26	<b>1.04</b>	1.79	1.00
sk_rrSVD_009	<b>1.76</b>	5.01	1.66	<b>1.27</b>	2.26	1.23	<b>0.98</b>	1.75	0.95
sk_rrSVD_010	<b>1.7</b>	4.72	1.61	<b>1.22</b>	2.06	1.18	<b>0.93</b>	1.86	0.89
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.09	3.41	1.02	0.71	1.89	0.66	0.53	1.54	0.49
sk_rrSVD_001	<b>2.88</b>	7.6	2.72	<b>2.44</b>	4.74	2.35	<b>2.13</b>	3.58	2.07
sk_rrSVD_002	<b>2.52</b>	5.42	2.44	<b>2.04</b>	4.18	1.97	<b>1.73</b>	2.85	1.65
sk_rrSVD_003	<b>2.27</b>	5.46	2.15	<b>1.86</b>	3.62	1.78	<b>1.53</b>	3.23	1.47
sk_rrSVD_004	<b>2.14</b>	5.22	2.06	<b>1.72</b>	3.34	1.64	<b>1.41</b>	3.03	1.34
sk_rrSVD_005	<b>2.04</b>	4.82	1.95	<b>1.6</b>	3.18	1.54	<b>1.29</b>	2.47	1.24
sk_rrSVD_006	<b>1.97</b>	5.2	1.88	<b>1.49</b>	2.49	1.43	<b>1.18</b>	2.12	1.15
sk_rrSVD_007	<b>1.9</b>	4.23	1.83	<b>1.4</b>	2.54	1.34	<b>1.1</b>	2.06	1.05
sk_rrSVD_008	<b>1.84</b>	4.65	1.75	<b>1.33</b>	2.76	1.27	<b>1.03</b>	2.01	0.99
sk_rrSVD_009	<b>1.77</b>	4.27	1.7	<b>1.27</b>	2.37	1.22	<b>0.98</b>	1.94	0.94
sk_rrSVD_010	<b>1.72</b>	4.79	1.63	<b>1.22</b>	2.58	1.17	<b>0.92</b>	1.85	0.88

**Table S33.** Normalized RMSE mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 80$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-3}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.52	0.57	1.4	0.91	2.86	0.84	0.66	1.53	0.63
sk_rrSVD_001	<b>5.18</b>	1.76	4.83	<b>4.38</b>	5.87	4.27	<b>3.86</b>	8.02	3.69
sk_rrSVD_002	<b>4.42</b>	1.44	4.06	<b>3.6</b>	6.24	3.45	<b>2.96</b>	8.79	2.73
sk_rrSVD_003	<b>4.02</b>	1.3	3.73	<b>3.04</b>	6.04	2.87	<b>2.38</b>	4.95	2.26
sk_rrSVD_004	<b>3.68</b>	1.22	3.41	<b>2.65</b>	6.38	2.49	<b>2.02</b>	4.51	1.93
sk_rrSVD_005	<b>3.45</b>	1.07	3.24	<b>2.36</b>	5.25	2.25	<b>1.79</b>	4.21	1.72
sk_rrSVD_006	<b>3.22</b>	1.19	2.94	<b>2.18</b>	4.75	2.03	<b>1.63</b>	3.48	1.54
sk_rrSVD_007	<b>3.01</b>	1.2	2.69	<b>2.04</b>	4.43	1.92	<b>1.49</b>	3.66	1.41
sk_rrSVD_008	<b>2.84</b>	1.19	2.56	<b>1.91</b>	4.22	1.82	<b>1.38</b>	3.14	1.3
sk_rrSVD_009	<b>2.7</b>	1.00	2.53	<b>1.81</b>	3.7	1.72	<b>1.29</b>	2.61	1.23
sk_rrSVD_010	<b>2.6</b>	0.89	2.39	<b>1.72</b>	3.65	1.63	<b>1.23</b>	2.6	1.17
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.06	3.23	1.00	0.69	1.68	0.66	0.53	1.03	0.51
sk_rrSVD_001	<b>2.82</b>	7.83	2.67	<b>2.42</b>	4.3	2.33	<b>2.13</b>	4.32	2.05
sk_rrSVD_002	<b>2.51</b>	6.83	2.39	<b>2.05</b>	3.94	1.97	<b>1.75</b>	2.99	1.68
sk_rrSVD_003	<b>2.27</b>	6.6	2.16	<b>1.87</b>	3.91	1.79	<b>1.55</b>	2.23	1.5
sk_rrSVD_004	<b>2.12</b>	4.94	2.01	<b>1.72</b>	3.39	1.65	<b>1.42</b>	2.17	1.38
sk_rrSVD_005	<b>2.03</b>	4.46	1.95	<b>1.6</b>	3.1	1.52	<b>1.3</b>	2.29	1.25
sk_rrSVD_006	<b>1.96</b>	5.53	1.84	<b>1.49</b>	3.4	1.4	<b>1.18</b>	2.1	1.13
sk_rrSVD_007	<b>1.89</b>	5.58	1.76	<b>1.4</b>	2.85	1.34	<b>1.1</b>	1.51	1.07
sk_rrSVD_008	<b>1.82</b>	5.25	1.71	<b>1.33</b>	2.83	1.26	<b>1.04</b>	1.71	1.00
sk_rrSVD_009	<b>1.76</b>	5.14	1.68	<b>1.26</b>	2.29	1.22	<b>0.98</b>	1.81	0.93
sk_rrSVD_010	<b>1.7</b>	5.04	1.61	<b>1.21</b>	1.99	1.17	<b>0.93</b>	1.67	0.89
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.06	2.98	1.00	0.69	1.75	0.65	0.52	1.47	0.48
sk_rrSVD_001	<b>2.87</b>	8.03	2.73	<b>2.44</b>	4.84	2.37	<b>2.14</b>	3.5	2.07
sk_rrSVD_002	<b>2.51</b>	5.9	2.39	<b>2.04</b>	4.61	1.97	<b>1.73</b>	2.95	1.66
sk_rrSVD_003	<b>2.27</b>	5.4	2.15	<b>1.86</b>	3.86	1.77	<b>1.54</b>	3.25	1.47
sk_rrSVD_004	<b>2.13</b>	4.71	2.05	<b>1.73</b>	3.27	1.65	<b>1.41</b>	2.71	1.34
sk_rrSVD_005	<b>2.03</b>	5.03	1.93	<b>1.6</b>	3.86	1.54	<b>1.29</b>	2.51	1.23
sk_rrSVD_006	<b>1.96</b>	6.09	1.84	<b>1.49</b>	2.67	1.44	<b>1.18</b>	2.26	1.14
sk_rrSVD_007	<b>1.9</b>	4.62	1.81	<b>1.4</b>	2.83	1.35	<b>1.1</b>	2.19	1.04
sk_rrSVD_008	<b>1.84</b>	4.54	1.74	<b>1.32</b>	2.51	1.27	<b>1.03</b>	1.82	0.99
sk_rrSVD_009	<b>1.77</b>	4.52	1.68	<b>1.27</b>	2.66	1.21	<b>0.97</b>	1.93	0.93
sk_rrSVD_010	<b>1.71</b>	4.7	1.63	<b>1.22</b>	2.21	1.17	<b>0.91</b>	2.06	0.87

**Table S34.** Normalized RMSE mean, standard deviation and best result for Au<sub>38Q</sub>  $K = 90$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-3</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.47	0.6	1.35	0.87	2.68	0.81	0.64	1.55	0.61
sk_rrSVD_001	<b>5.18</b>	1.72	4.83	<b>4.38</b>	7.6	4.13	<b>3.86</b>	8.15	3.64
sk_rrSVD_002	<b>4.42</b>	1.4	4.13	<b>3.6</b>	7.06	3.42	<b>2.96</b>	8.12	2.77
sk_rrSVD_003	<b>4.02</b>	1.22	3.73	<b>3.04</b>	5.78	2.93	<b>2.37</b>	5.35	2.26
sk_rrSVD_004	<b>3.69</b>	1.24	3.4	<b>2.63</b>	5.82	2.47	<b>2.02</b>	4.46	1.89
sk_rrSVD_005	<b>3.44</b>	1.2	3.18	<b>2.36</b>	5.25	2.2	<b>1.79</b>	4.23	1.7
sk_rrSVD_006	<b>3.21</b>	1.13	2.94	<b>2.18</b>	4.32	2.04	<b>1.62</b>	3.37	1.54
sk_rrSVD_007	<b>3.02</b>	1.07	2.81	<b>2.03</b>	4.1	1.93	<b>1.48</b>	3.5	1.39
sk_rrSVD_008	<b>2.84</b>	1.12	2.63	<b>1.9</b>	4.27	1.8	<b>1.38</b>	3.19	1.32
sk_rrSVD_009	<b>2.69</b>	1.04	2.48	<b>1.8</b>	3.98	1.71	<b>1.29</b>	2.35	1.23
sk_rrSVD_010	<b>2.59</b>	0.95	2.4	<b>1.71</b>	3.59	1.63	<b>1.22</b>	2.49	1.18
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.03	3.02	0.97	0.67	1.64	0.64	0.51	0.97	0.5
sk_rrSVD_001	<b>2.82</b>	8.04	2.67	<b>2.42</b>	4.14	2.33	<b>2.13</b>	3.9	2.04
sk_rrSVD_002	<b>2.49</b>	7.29	2.32	<b>2.05</b>	3.98	1.98	<b>1.75</b>	3.01	1.67
sk_rrSVD_003	<b>2.26</b>	5.83	2.18	<b>1.87</b>	3.69	1.8	<b>1.55</b>	2.31	1.49
sk_rrSVD_004	<b>2.12</b>	5.36	2.03	<b>1.72</b>	3.85	1.64	<b>1.42</b>	2.25	1.37
sk_rrSVD_005	<b>2.03</b>	5.08	1.96	<b>1.6</b>	2.91	1.51	<b>1.3</b>	2.1	1.25
sk_rrSVD_006	<b>1.96</b>	5.62	1.86	<b>1.49</b>	3.34	1.41	<b>1.18</b>	2.37	1.14
sk_rrSVD_007	<b>1.89</b>	5.66	1.78	<b>1.4</b>	2.82	1.33	<b>1.1</b>	1.52	1.06
sk_rrSVD_008	<b>1.82</b>	5.37	1.72	<b>1.32</b>	2.97	1.28	<b>1.04</b>	1.56	1.00
sk_rrSVD_009	<b>1.75</b>	4.86	1.66	<b>1.26</b>	2.38	1.22	<b>0.97</b>	1.7	0.94
sk_rrSVD_010	<b>1.69</b>	4.73	1.61	<b>1.21</b>	1.91	1.17	<b>0.92</b>	1.54	0.89
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>	$\bar{x}$ 1x10 <sup>-2</sup>	$\sigma$ 1x10 <sup>-4</sup>	min(x) 1x10 <sup>-2</sup>
cho_dec	1.03	2.88	0.98	0.66	1.85	0.62	0.5	1.43	0.47
sk_rrSVD_001	<b>2.87</b>	7.04	2.7	<b>2.45</b>	5.06	2.36	<b>2.14</b>	3.11	2.07
sk_rrSVD_002	<b>2.5</b>	5.91	2.38	<b>2.04</b>	4.14	1.95	<b>1.73</b>	3.4	1.65
sk_rrSVD_003	<b>2.27</b>	5.29	2.16	<b>1.86</b>	3.45	1.8	<b>1.53</b>	3.16	1.47
sk_rrSVD_004	<b>2.12</b>	4.36	2.06	<b>1.72</b>	3.57	1.64	<b>1.41</b>	2.77	1.34
sk_rrSVD_005	<b>2.02</b>	4.8	1.93	<b>1.6</b>	3.54	1.53	<b>1.29</b>	2.31	1.24
sk_rrSVD_006	<b>1.96</b>	5.39	1.83	<b>1.49</b>	3.14	1.44	<b>1.18</b>	2.08	1.14
sk_rrSVD_007	<b>1.9</b>	4.77	1.8	<b>1.39</b>	3.11	1.34	<b>1.09</b>	2.25	1.04
sk_rrSVD_008	<b>1.84</b>	4.77	1.73	<b>1.32</b>	2.41	1.28	<b>1.03</b>	1.76	1.00
sk_rrSVD_009	<b>1.77</b>	3.91	1.69	<b>1.26</b>	2.31	1.2	<b>0.97</b>	1.79	0.94
sk_rrSVD_010	<b>1.71</b>	4.65	1.63	<b>1.21</b>	2.67	1.17	<b>0.91</b>	1.87	0.88

**Table S35.** Normalized RMSE mean, standard deviation and best result for  $\text{Au}_{38\text{Q}}$   $K = 100$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-3}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.43	0.62	1.32	0.84	2.54	0.78	0.61	1.6	0.59
sk_rrSVD_001	<b>5.18</b>	1.76	4.84	<b>4.37</b>	8.33	4.21	<b>3.87</b>	8.71	3.61
sk_rrSVD_002	<b>4.42</b>	1.45	4.11	<b>3.6</b>	7.29	3.39	<b>2.94</b>	7.38	2.77
sk_rrSVD_003	<b>4.01</b>	1.28	3.76	<b>3.05</b>	5.49	2.91	<b>2.37</b>	4.96	2.26
sk_rrSVD_004	<b>3.68</b>	1.2	3.45	<b>2.64</b>	5.31	2.52	<b>2.02</b>	4.32	1.91
sk_rrSVD_005	<b>3.44</b>	1.32	3.18	<b>2.35</b>	5.58	2.2	<b>1.79</b>	4.23	1.7
sk_rrSVD_006	<b>3.22</b>	1.3	2.88	<b>2.17</b>	4.88	2.03	<b>1.61</b>	3.26	1.55
sk_rrSVD_007	<b>3.01</b>	1.05	2.71	<b>2.03</b>	4.41	1.95	<b>1.48</b>	3.31	1.39
sk_rrSVD_008	<b>2.83</b>	1.16	2.55	<b>1.9</b>	5.05	1.79	<b>1.37</b>	2.95	1.32
sk_rrSVD_009	<b>2.68</b>	1.07	2.47	<b>1.79</b>	3.58	1.71	<b>1.29</b>	2.4	1.23
sk_rrSVD_010	<b>2.58</b>	0.94	2.39	<b>1.71</b>	4.00	1.63	<b>1.22</b>	2.34	1.16
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.01	2.92	0.94	0.65	1.66	0.62	0.5	1.04	0.48
sk_rrSVD_001	<b>2.82</b>	8.02	2.67	<b>2.42</b>	5.27	2.3	<b>2.13</b>	3.25	2.03
sk_rrSVD_002	<b>2.5</b>	6.33	2.37	<b>2.05</b>	4.29	1.96	<b>1.75</b>	2.91	1.69
sk_rrSVD_003	<b>2.27</b>	6.29	2.17	<b>1.87</b>	3.72	1.78	<b>1.55</b>	2.06	1.49
sk_rrSVD_004	<b>2.12</b>	5.82	2.02	<b>1.72</b>	3.37	1.64	<b>1.42</b>	2.11	1.38
sk_rrSVD_005	<b>2.03</b>	4.92	1.94	<b>1.59</b>	2.96	1.54	<b>1.3</b>	1.95	1.25
sk_rrSVD_006	<b>1.96</b>	5.92	1.86	<b>1.49</b>	3.35	1.41	<b>1.18</b>	2.09	1.14
sk_rrSVD_007	<b>1.89</b>	5.38	1.79	<b>1.4</b>	2.89	1.33	<b>1.1</b>	1.73	1.05
sk_rrSVD_008	<b>1.83</b>	5.13	1.72	<b>1.32</b>	2.66	1.25	<b>1.03</b>	1.51	1.00
sk_rrSVD_009	<b>1.75</b>	5.35	1.65	<b>1.26</b>	2.64	1.21	<b>0.97</b>	1.75	0.94
sk_rrSVD_010	<b>1.69</b>	4.21	1.63	<b>1.21</b>	2.23	1.16	<b>0.92</b>	1.47	0.89
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
Method / coef.	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
Method / coef.	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$	$1 \times 10^{-2}$	$1 \times 10^{-4}$	$1 \times 10^{-2}$
cho_dec	1.01	2.85	0.96	0.64	1.82	0.6	0.49	1.45	0.45
sk_rrSVD_001	<b>2.87</b>	7.59	2.72	<b>2.44</b>	4.65	2.34	<b>2.14</b>	3.74	2.07
sk_rrSVD_002	<b>2.5</b>	5.47	2.4	<b>2.04</b>	4.41	1.96	<b>1.73</b>	3.11	1.65
sk_rrSVD_003	<b>2.26</b>	5.00	2.13	<b>1.87</b>	3.62	1.79	<b>1.53</b>	2.8	1.48
sk_rrSVD_004	<b>2.13</b>	4.41	2.05	<b>1.72</b>	3.33	1.65	<b>1.41</b>	2.81	1.34
sk_rrSVD_005	<b>2.02</b>	5.05	1.92	<b>1.6</b>	3.48	1.53	<b>1.29</b>	2.28	1.24
sk_rrSVD_006	<b>1.96</b>	4.81	1.83	<b>1.49</b>	3.16	1.44	<b>1.18</b>	2.35	1.12
sk_rrSVD_007	<b>1.9</b>	5.04	1.81	<b>1.39</b>	3.18	1.34	<b>1.09</b>	1.94	1.05
sk_rrSVD_008	<b>1.83</b>	4.6	1.74	<b>1.32</b>	2.57	1.26	<b>1.03</b>	1.84	0.99
sk_rrSVD_009	<b>1.76</b>	3.81	1.66	<b>1.26</b>	2.58	1.2	<b>0.96</b>	1.61	0.93
sk_rrSVD_010	<b>1.71</b>	4.43	1.62	<b>1.21</b>	2.59	1.17	<b>0.91</b>	1.79	0.88

**Table S36.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 10$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>
cho_dec	1.05	0.38	0.98	3.76	3.49	3.43	0.55	2.39	0.53
sk_rrSVD_001	<b>0.95</b>	1.06	0.88	<b>1.95</b>	3.92	1.64	<b>0.51</b>	7.04	0.46
sk_rrSVD_002	<b>1.12</b>	1.12	1.06	<b>3.06</b>	7.71	2.59	<b>0.81</b>	10.63	0.72
sk_rrSVD_003	<b>1.62</b>	1.47	1.54	<b>4.63</b>	9.72	3.94	<b>1.13</b>	15.28	1.01
sk_rrSVD_004	<b>2.04</b>	1.73	1.97	<b>5.98</b>	12.23	5.18	<b>1.42</b>	14.28	1.3
sk_rrSVD_005	<b>2.53</b>	1.79	2.44	<b>7.56</b>	15.93	6.45	<b>1.72</b>	14.32	1.62
sk_rrSVD_006	<b>2.96</b>	1.13	2.83	<b>8.68</b>	13.3	7.74	<b>1.94</b>	8.11	1.89
sk_rrSVD_007	<b>3.43</b>	1.29	3.32	<b>9.65</b>	11.19	8.89	<b>2.3</b>	6.78	2.25
sk_rrSVD_008	<b>3.85</b>	0.62	3.75	<b>10.6</b>	5.12	10.11	<b>2.72</b>	8.88	2.63
sk_rrSVD_009	<b>4.4</b>	3.15	4.22	<b>11.9</b>	8.65	11.27	<b>3.21</b>	14.08	3.04
sk_rrSVD_010	<b>4.79</b>	3.38	3.00	<b>13.05</b>	5.41	12.53	<b>3.44</b>	4.63	3.38
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>
cho_dec	1.21	3.68	0.82	3.63	2.31	3.36	0.55	2.56	0.53
sk_rrSVD_001	<b>1.09</b>	2.39	0.7	<b>1.82</b>	3.35	1.6	<b>0.5</b>	6.55	0.47
sk_rrSVD_002	1.17	2.59	0.77	<b>2.86</b>	4.00	2.53	<b>0.78</b>	10.06	0.72
sk_rrSVD_003	<b>1.49</b>	3.54	0.92	<b>4.14</b>	6.49	3.7	<b>1.08</b>	14.6	0.99
sk_rrSVD_004	<b>1.88</b>	4.14	1.12	<b>5.48</b>	7.52	5.05	<b>1.42</b>	15.25	1.32
sk_rrSVD_005	<b>2.24</b>	4.31	1.36	<b>7.05</b>	11.77	6.29	<b>1.75</b>	19.59	1.61
sk_rrSVD_006	<b>2.63</b>	4.76	1.63	<b>8.3</b>	12.4	7.55	<b>1.96</b>	9.46	1.89
sk_rrSVD_007	<b>3.07</b>	5.95	1.88	<b>9.3</b>	8.86	8.74	<b>2.3</b>	7.65	2.21
sk_rrSVD_008	<b>3.52</b>	6.37	2.16	<b>10.34</b>	6.25	9.89	<b>2.73</b>	6.17	2.66
sk_rrSVD_009	<b>3.88</b>	6.79	2.44	<b>11.57</b>	8.02	11.08	<b>3.25</b>	18.94	3.01
sk_rrSVD_010	<b>4.3</b>	7.07	2.75	<b>12.8</b>	7.16	12.24	<b>3.47</b>	7.97	3.38
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>-1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>
cho_dec	1.18	1.41	1.05	3.55	0.25	3.24	0.52	0.11	0.5
sk_rrSVD_001	<b>1.03</b>	1.9	0.82	<b>1.92</b>	0.34	1.66	<b>0.51</b>	0.55	0.46
sk_rrSVD_002	1.19	1.56	1.07	<b>3.05</b>	0.66	2.6	<b>0.84</b>	1.62	0.72
sk_rrSVD_003	<b>1.57</b>	0.76	1.52	<b>4.53</b>	0.92	3.82	<b>1.22</b>	2.06	1.01
sk_rrSVD_004	<b>1.98</b>	0.59	1.92	<b>6.02</b>	1.12	5.13	<b>1.49</b>	1.7	1.32
sk_rrSVD_005	<b>2.43</b>	0.61	2.37	<b>7.62</b>	1.64	6.33	<b>1.81</b>	2.02	1.62
sk_rrSVD_006	<b>2.8</b>	0.62	2.74	<b>8.8</b>	1.74	7.5	<b>1.95</b>	0.67	1.86
sk_rrSVD_007	<b>3.25</b>	1.04	3.17	<b>9.87</b>	1.45	8.96	<b>2.32</b>	1.08	2.21
sk_rrSVD_008	<b>3.75</b>	2.16	3.63	<b>10.96</b>	1.14	10.24	<b>2.78</b>	1.34	2.64
sk_rrSVD_009	<b>4.32</b>	1.17	4.12	<b>12.28</b>	1.34	11.46	<b>3.17</b>	1.52	3.01
sk_rrSVD_010	<b>4.66</b>	0.84	4.54	<b>13.49</b>	1.09	12.67	<b>3.45</b>	0.64	3.36



**Table S37.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 20$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$
cho_dec	2.3	1.43	2.09	0.57	0.15	5.41	1.45	6.98	1.38
sk_rrSVD_001	<b>1.1</b>	1.81	0.92	<b>0.37</b>	0.24	3.45	<b>0.97</b>	6.32	0.89
sk_rrSVD_002	<b>1.29</b>	2.57	1.03	<b>0.53</b>	0.47	4.79	<b>1.35</b>	13.02	1.21
sk_rrSVD_003	<b>1.76</b>	4.54	1.29	<b>0.69</b>	0.79	6.25	<b>1.72</b>	15.3	1.6
sk_rrSVD_004	2.16	5.24	1.59	<b>0.89</b>	1.45	7.7	<b>2.13</b>	16.52	1.97
sk_rrSVD_005	2.61	6.61	1.95	<b>1.06</b>	1.63	9.28	<b>2.49</b>	11.52	2.37
sk_rrSVD_006	<b>3.23</b>	8.44	2.39	<b>1.23</b>	1.83	11.03	<b>2.92</b>	7.91	2.78
sk_rrSVD_007	<b>3.84</b>	9.41	2.79	<b>1.43</b>	2.06	12.6	<b>3.34</b>	8.78	3.21
sk_rrSVD_008	<b>4.51</b>	11.55	3.19	<b>1.6</b>	1.91	14.38	<b>3.98</b>	6.23	3.82
sk_rrSVD_009	<b>5.1</b>	12.69	3.61	<b>1.74</b>	1.45	16.13	<b>4.7</b>	15.21	4.42
sk_rrSVD_010	<b>5.78</b>	13.00	4.11	<b>1.89</b>	1.08	18.1	<b>4.91</b>	5.86	4.8
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	2.35	3.04	2.11	0.57	0.34	5.49	1.45	0.82	1.34
sk_rrSVD_001	<b>1.19</b>	1.94	0.94	<b>0.39</b>	0.52	3.47	<b>0.97</b>	0.93	0.85
sk_rrSVD_002	<b>1.47</b>	2.71	1.08	<b>0.53</b>	0.71	4.78	<b>1.36</b>	1.6	1.15
sk_rrSVD_003	<b>1.95</b>	3.71	1.3	<b>0.71</b>	1.13	6.18	<b>1.78</b>	2.02	1.51
sk_rrSVD_004	<b>2.5</b>	5.4	1.6	<b>0.89</b>	1.56	7.56	<b>2.22</b>	2.42	1.93
sk_rrSVD_005	<b>2.98</b>	5.69	1.89	<b>1.02</b>	1.49	9.01	<b>2.49</b>	1.69	2.3
sk_rrSVD_006	<b>3.61</b>	5.5	2.28	<b>1.18</b>	1.76	10.73	<b>2.87</b>	0.93	2.76
sk_rrSVD_007	<b>4.27</b>	7.55	2.68	<b>1.34</b>	1.79	12.16	<b>3.31</b>	1.4	3.17
sk_rrSVD_008	<b>4.78</b>	8.02	3.1	<b>1.5</b>	1.63	13.92	<b>3.96</b>	2.03	3.77
sk_rrSVD_009	<b>5.45</b>	8.95	3.45	<b>1.68</b>	1.59	15.67	<b>4.6</b>	1.95	4.25
sk_rrSVD_010	<b>6.18</b>	10.29	3.9	<b>1.88</b>	1.92	17.6	<b>4.87</b>	1.03	4.74
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	2.55	2.58	2.1	0.57	0.22	5.41	1.48	1.19	1.37
sk_rrSVD_001	<b>1.26</b>	2.51	0.94	<b>0.39</b>	0.5	3.38	<b>0.99</b>	1.09	0.87
sk_rrSVD_002	<b>1.43</b>	2.77	1.02	0.57	1.05	4.55	1.47	1.76	1.2
sk_rrSVD_003	<b>1.91</b>	3.57	1.25	<b>0.75</b>	1.17	5.97	<b>1.93</b>	2.11	1.57
sk_rrSVD_004	2.34	4.29	1.54	<b>0.97</b>	1.61	7.69	<b>2.2</b>	2.46	1.97
sk_rrSVD_005	<b>2.82</b>	5.08	1.84	<b>1.14</b>	1.95	9.18	<b>2.49</b>	1.71	2.32
sk_rrSVD_006	<b>3.59</b>	8.43	2.25	<b>1.28</b>	1.97	10.81	<b>2.91</b>	1.46	2.78
sk_rrSVD_007	<b>4.05</b>	8.19	2.61	<b>1.43</b>	1.96	12.32	<b>3.27</b>	1.09	3.16
sk_rrSVD_008	<b>4.64</b>	9.27	3.03	<b>1.6</b>	2.08	14.02	<b>3.92</b>	0.84	3.78
sk_rrSVD_009	<b>5.38</b>	11.12	3.44	<b>1.74</b>	1.84	15.71	<b>4.65</b>	1.56	4.28
sk_rrSVD_010	<b>6.15</b>	14.37	3.89	<b>1.91</b>	1.9	17.36	<b>4.84</b>	1.18	4.66

**Table S38.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 30$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	2.43	1.64	2.21	1.04	1.06	0.97	2.91	0.88	2.79
sk_rrSVD_001	<b>1.32</b>	0.93	1.22	<b>0.55</b>	0.67	0.5	<b>1.51</b>	1.44	1.38
sk_rrSVD_002	<b>1.54</b>	1.9	1.39	<b>0.8</b>	1.63	0.65	<b>2.02</b>	2.22	1.8
sk_rrSVD_003	<b>2.07</b>	3.8	1.79	1.02	1.91	0.83	<b>2.49</b>	2.52	2.2
sk_rrSVD_004	<b>2.46</b>	4.86	2.08	<b>1.2</b>	1.8	1.00	2.93	2.13	2.71
sk_rrSVD_005	<b>2.96</b>	6.37	2.46	<b>1.34</b>	1.98	1.16	<b>3.32</b>	1.88	3.18
sk_rrSVD_006	<b>3.58</b>	9.08	2.97	<b>1.56</b>	2.2	1.39	<b>3.8</b>	1.47	3.64
sk_rrSVD_007	<b>4.14</b>	9.72	3.41	<b>1.72</b>	1.95	1.55	<b>4.28</b>	1.45	4.09
sk_rrSVD_008	<b>4.83</b>	11.74	3.92	<b>1.9</b>	1.53	1.76	<b>5.05</b>	1.7	4.84
sk_rrSVD_009	<b>5.26</b>	13.11	4.35	<b>2.09</b>	1.74	1.95	<b>5.75</b>	1.9	5.42
sk_rrSVD_010	<b>5.33</b>	8.51	4.88	<b>2.29</b>	1.57	2.17	<b>6.04</b>	1.14	5.86
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	2.57	2.29	2.18	1.04	0.59	1.00	2.98	1.75	2.83
sk_rrSVD_001	<b>1.48</b>	1.13	1.22	<b>0.56</b>	0.56	0.51	<b>1.6</b>	1.78	1.37
sk_rrSVD_002	<b>1.81</b>	1.99	1.47	<b>0.8</b>	1.41	0.66	<b>2.12</b>	2.5	1.76
sk_rrSVD_003	<b>2.37</b>	2.97	1.77	1.03	1.66	0.84	<b>2.5</b>	1.99	2.24
sk_rrSVD_004	<b>2.99</b>	3.73	2.09	<b>1.22</b>	1.64	0.98	2.98	2.44	2.74
sk_rrSVD_005	<b>3.58</b>	4.44	2.45	<b>1.4</b>	1.89	1.13	<b>3.42</b>	2.08	3.19
sk_rrSVD_006	<b>4.34</b>	6.41	2.99	<b>1.58</b>	2.29	1.34	<b>3.9</b>	1.65	3.66
sk_rrSVD_007	<b>5.04</b>	6.65	3.44	<b>1.77</b>	2.32	1.51	<b>4.36</b>	1.51	4.14
sk_rrSVD_008	<b>5.71</b>	7.14	3.91	<b>1.87</b>	1.69	1.73	<b>5.2</b>	1.82	4.86
sk_rrSVD_009	<b>6.59</b>	9.17	4.37	<b>2.08</b>	1.83	1.91	<b>5.97</b>	2.79	5.49
sk_rrSVD_010	<b>7.15</b>	10.85	4.89	<b>2.25</b>	1.52	2.1	<b>6.32</b>	2.59	5.95
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	2.62	2.79	2.18	1.03	0.81	0.97	2.88	0.39	2.81
sk_rrSVD_001	<b>1.54</b>	1.98	1.27	<b>0.58</b>	0.75	0.5	<b>1.43</b>	0.81	1.35
sk_rrSVD_002	<b>1.81</b>	3.46	1.39	<b>0.83</b>	1.47	0.66	<b>2.04</b>	2.44	1.72
sk_rrSVD_003	<b>2.45</b>	5.5	1.7	<b>1.13</b>	1.86	0.84	<b>2.5</b>	2.19	2.23
sk_rrSVD_004	<b>2.91</b>	5.69	2.00	<b>1.3</b>	1.89	0.99	2.94	1.83	2.74
sk_rrSVD_005	<b>3.56</b>	7.07	2.37	<b>1.44</b>	1.87	1.14	<b>3.24</b>	1.22	3.12
sk_rrSVD_006	<b>4.25</b>	7.43	2.9	<b>1.64</b>	2.11	1.36	<b>3.73</b>	1.32	3.58
sk_rrSVD_007	<b>4.83</b>	7.87	3.29	<b>1.75</b>	2.1	1.54	<b>4.19</b>	1.2	4.03
sk_rrSVD_008	<b>5.54</b>	9.28	3.76	<b>1.89</b>	1.69	1.72	<b>4.87</b>	1.29	4.72
sk_rrSVD_009	<b>6.25</b>	11.01	4.36	<b>2.07</b>	1.56	1.9	<b>5.66</b>	1.5	5.32
sk_rrSVD_010	<b>6.66</b>	12.71	4.83	<b>2.3</b>	1.77	2.12	<b>5.9</b>	1.21	5.81

**Table S39.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 40$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	4.14	2.74	3.69	1.74	0.6	1.66	4.64	1.73	4.5
sk_rrSVD_001	<b>2.01</b>	1.94	1.71	<b>0.75</b>	0.87	0.68	<b>2.16</b>	1.57	2.00
sk_rrSVD_002	<b>2.17</b>	2.2	1.92	<b>1.00</b>	1.3	0.89	<b>2.63</b>	1.54	2.48
sk_rrSVD_003	<b>2.8</b>	3.7	2.43	<b>1.25</b>	1.78	1.08	<b>3.21</b>	2.36	3.01
sk_rrSVD_004	<b>3.27</b>	4.85	2.84	<b>1.5</b>	2.27	1.25	<b>3.77</b>	1.52	3.57
sk_rrSVD_005	<b>3.78</b>	6.43	3.24	1.73	2.47	1.5	<b>4.26</b>	1.59	4.05
sk_rrSVD_006	4.53	8.6	3.8	<b>1.96</b>	2.46	1.72	<b>4.78</b>	1.25	4.62
sk_rrSVD_007	<b>5.22</b>	9.41	4.34	<b>2.11</b>	2.02	1.94	<b>5.26</b>	0.98	5.12
sk_rrSVD_008	<b>5.84</b>	11.73	4.9	<b>2.29</b>	1.74	2.14	<b>6.16</b>	1.38	5.93
sk_rrSVD_009	<b>6.28</b>	11.68	5.42	<b>2.51</b>	1.83	2.36	<b>6.92</b>	2.26	6.6
sk_rrSVD_010	<b>6.78</b>	12.33	6.02	<b>2.73</b>	1.3	2.61	<b>7.33</b>	1.84	6.98
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	3.92	3.34	3.48	1.76	0.81	1.68	5.72	9.83	4.54
sk_rrSVD_001	<b>1.98</b>	3.1	1.67	<b>0.76</b>	1.00	0.69	<b>2.29</b>	2.13	2.03
sk_rrSVD_002	<b>2.18</b>	3.67	1.89	<b>1.01</b>	1.45	0.88	<b>2.86</b>	2.76	2.49
sk_rrSVD_003	<b>2.76</b>	5.19	2.33	<b>1.25</b>	1.83	1.07	<b>3.33</b>	2.59	3.00
sk_rrSVD_004	<b>3.28</b>	5.44	2.79	<b>1.47</b>	1.98	1.29	<b>3.93</b>	1.85	3.64
sk_rrSVD_005	<b>3.81</b>	6.72	3.15	<b>1.66</b>	2.03	1.48	<b>4.4</b>	2.1	4.08
sk_rrSVD_006	<b>4.62</b>	8.69	3.76	<b>1.88</b>	2.23	1.7	5.03	3.3	4.64
sk_rrSVD_007	<b>5.25</b>	9.77	4.26	<b>2.07</b>	2.11	1.9	5.51	2.35	5.16
sk_rrSVD_008	<b>6.04</b>	11.66	4.87	<b>2.27</b>	1.35	2.12	<b>6.55</b>	4.07	6.00
sk_rrSVD_009	<b>6.25</b>	12.88	5.3	<b>2.56</b>	2.14	2.37	<b>7.29</b>	3.94	6.67
sk_rrSVD_010	<b>6.71</b>	12.97	5.89	<b>2.82</b>	2.22	2.61	<b>7.58</b>	3.85	7.17
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	4.00	0.32	3.59	1.76	0.38	1.7	4.56	0.72	4.45
sk_rrSVD_001	<b>2.03</b>	0.25	1.72	<b>0.75</b>	0.33	0.7	<b>2.11</b>	0.87	2.01
sk_rrSVD_002	<b>2.31</b>	0.41	1.91	<b>1.07</b>	1.6	0.91	<b>2.58</b>	1.53	2.45
sk_rrSVD_003	<b>3.02</b>	0.75	2.35	<b>1.4</b>	2.01	1.1	<b>3.12</b>	1.3	2.99
sk_rrSVD_004	<b>3.79</b>	1.08	2.83	<b>1.57</b>	2.05	1.3	<b>3.76</b>	1.67	3.57
sk_rrSVD_005	4.28	1.15	3.18	1.75	2.13	1.5	<b>4.23</b>	1.47	4.05
sk_rrSVD_006	<b>5.03</b>	1.1	3.76	<b>2.01</b>	2.48	1.74	<b>4.7</b>	1.24	4.55
sk_rrSVD_007	<b>5.71</b>	1.19	4.29	<b>2.16</b>	2.07	1.94	<b>5.24</b>	1.39	5.07
sk_rrSVD_008	<b>6.59</b>	1.3	4.82	<b>2.33</b>	1.72	2.17	<b>6.1</b>	1.77	5.79
sk_rrSVD_009	<b>6.94</b>	1.41	5.31	<b>2.58</b>	1.69	2.38	<b>6.88</b>	1.88	6.56
sk_rrSVD_010	<b>7.13</b>	1.34	5.77	<b>2.79</b>	1.81	2.64	<b>7.22</b>	1.49	7.04

**Table S40.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 50$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	5.66	2.42	5.37	2.54	0.89	2.43	7.43	7.86	6.62
sk_rrSVD_001	<b>2.33</b>	1.23	2.21	<b>1.07</b>	1.95	0.91	<b>2.92</b>	2.43	2.65
sk_rrSVD_002	<b>2.74</b>	3.44	2.51	<b>1.39</b>	2.22	1.15	<b>3.51</b>	2.35	3.18
sk_rrSVD_003	<b>3.59</b>	4.05	3.18	<b>1.61</b>	2.06	1.37	<b>4.06</b>	2.31	3.8
sk_rrSVD_004	<b>4.2</b>	6.04	3.53	<b>1.81</b>	2.07	1.58	<b>4.68</b>	1.85	4.41
sk_rrSVD_005	<b>4.62</b>	6.89	4.00	<b>2.03</b>	2.01	1.79	<b>5.21</b>	1.98	4.93
sk_rrSVD_006	<b>5.29</b>	8.84	4.44	<b>2.27</b>	2.03	2.08	<b>5.78</b>	1.66	5.56
sk_rrSVD_007	6.03	9.56	5.09	<b>2.45</b>	1.64	2.31	<b>6.44</b>	2.52	6.14
sk_rrSVD_008	<b>6.51</b>	10.00	5.63	<b>2.69</b>	1.17	2.55	7.35	1.87	7.02
sk_rrSVD_009	<b>7.32</b>	13.03	6.22	<b>2.94</b>	1.6	2.81	<b>8.23</b>	2.63	7.74
sk_rrSVD_010	<b>7.76</b>	12.23	6.95	<b>3.2</b>	1.41	3.05	<b>8.69</b>	2.2	8.41
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	5.74	4.82	5.21	2.54	0.69	2.45	8.95	15.29	6.65
sk_rrSVD_001	<b>2.41</b>	3.36	2.15	<b>1.05</b>	1.86	0.89	<b>3.28</b>	5.62	2.68
sk_rrSVD_002	<b>2.87</b>	4.34	2.43	<b>1.41</b>	2.04	1.17	<b>3.52</b>	2.69	3.17
sk_rrSVD_003	<b>3.65</b>	7.86	3.00	<b>1.66</b>	1.78	1.38	<b>4.27</b>	5.82	3.79
sk_rrSVD_004	<b>4.32</b>	9.18	3.42	<b>1.86</b>	2.24	1.57	<b>4.88</b>	3.31	4.42
sk_rrSVD_005	<b>4.83</b>	8.62	3.83	<b>2.07</b>	2.39	1.77	<b>5.49</b>	3.6	5.04
sk_rrSVD_006	5.65	10.22	4.48	<b>2.28</b>	2.17	2.04	<b>6.1</b>	3.63	5.62
sk_rrSVD_007	6.46	11.91	5.01	<b>2.51</b>	2.18	2.28	<b>6.96</b>	4.38	6.23
sk_rrSVD_008	<b>6.98</b>	13.82	5.68	<b>2.73</b>	1.88	2.51	<b>8.17</b>	8.69	7.13
sk_rrSVD_009	<b>7.02</b>	9.81	6.25	<b>2.96</b>	1.85	2.79	8.86	7.59	7.73
sk_rrSVD_010	<b>7.6</b>	11.53	6.81	<b>3.17</b>	1.65	3.04	9.18	6.24	8.41
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	6.01	5.99	5.49	2.55	0.74	2.45	6.76	0.8	6.63
sk_rrSVD_001	<b>2.58</b>	3.66	2.21	<b>1.02</b>	1.4	0.9	<b>2.8</b>	0.92	2.69
sk_rrSVD_002	<b>3.37</b>	6.01	2.66	<b>1.49</b>	2.59	1.13	<b>3.37</b>	1.84	3.16
sk_rrSVD_003	<b>4.12</b>	7.87	3.24	<b>1.7</b>	2.12	1.38	<b>3.94</b>	1.83	3.75
sk_rrSVD_004	<b>4.65</b>	8.63	3.67	<b>1.92</b>	1.74	1.61	<b>4.65</b>	2.00	4.37
sk_rrSVD_005	<b>5.35</b>	9.5	4.16	<b>2.13</b>	2.14	1.85	<b>5.16</b>	1.87	4.93
sk_rrSVD_006	6.15	11.17	4.79	2.42	2.51	2.08	<b>5.69</b>	1.94	5.42
sk_rrSVD_007	<b>6.96</b>	12.09	5.32	2.55	2.23	2.33	<b>6.21</b>	1.27	6.02
sk_rrSVD_008	<b>7.56</b>	12.58	6.06	<b>2.74</b>	1.99	2.53	<b>7.21</b>	1.91	6.91
sk_rrSVD_009	<b>8.21</b>	16.73	6.54	<b>2.99</b>	2.33	2.76	<b>8.1</b>	2.09	7.74
sk_rrSVD_010	<b>8.6</b>	14.9	7.09	<b>3.19</b>	1.3	3.02	<b>8.45</b>	1.33	8.2

**Table S41.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 60$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	7.95	5.44	7.29	3.34	0.85	3.23	10.79	11.17	9.22
sk_rrSVD_001	<b>3.27</b>	1.79	2.79	<b>1.2</b>	1.03	1.09	<b>3.7</b>	1.79	3.52
sk_rrSVD_002	<b>3.59</b>	5.4	3.23	<b>1.64</b>	2.23	1.39	<b>4.35</b>	1.98	4.13
sk_rrSVD_003	<b>4.62</b>	7.08	3.99	<b>1.88</b>	2.1	1.63	<b>5.03</b>	2.07	4.78
sk_rrSVD_004	<b>5.01</b>	8.86	4.4	<b>2.1</b>	2.04	1.86	<b>5.79</b>	1.88	5.5
sk_rrSVD_005	<b>5.76</b>	8.66	4.71	<b>2.36</b>	2.17	2.12	<b>6.41</b>	1.62	6.14
sk_rrSVD_006	<b>6.41</b>	9.51	5.51	<b>2.62</b>	2.07	2.42	<b>7.06</b>	2.02	6.74
sk_rrSVD_007	<b>7.25</b>	9.66	6.06	<b>2.84</b>	1.5	2.7	<b>7.81</b>	2.95	7.39
sk_rrSVD_008	<b>7.74</b>	9.3	6.64	<b>3.06</b>	1.16	2.89	<b>8.83</b>	2.63	8.3
sk_rrSVD_009	<b>8.34</b>	11.04	7.34	3.33	1.39	3.16	<b>9.69</b>	2.78	9.17
sk_rrSVD_010	<b>8.87</b>	10.29	7.99	<b>3.61</b>	1.44	3.46	<b>10.25</b>	3.84	9.65
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^{-1}$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	7.83	5.32	7.31	3.38	0.61	3.28	11.65	1.9	9.37
sk_rrSVD_001	<b>3.12</b>	3.21	2.72	<b>1.19</b>	1.38	1.1	<b>4.43</b>	0.89	3.54
sk_rrSVD_002	<b>3.38</b>	3.78	3.1	<b>1.57</b>	1.62	1.42	<b>4.62</b>	0.52	4.09
sk_rrSVD_003	<b>4.39</b>	5.54	3.8	<b>1.81</b>	1.7	1.67	<b>5.67</b>	0.82	4.75
sk_rrSVD_004	<b>4.87</b>	5.71	4.39	<b>2.09</b>	1.92	1.87	<b>6.51</b>	1.06	5.47
sk_rrSVD_005	<b>5.58</b>	6.94	4.9	<b>2.33</b>	2.26	2.09	<b>6.67</b>	0.57	6.15
sk_rrSVD_006	<b>6.42</b>	10.7	5.42	<b>2.6</b>	2.01	2.41	<b>8.03</b>	1.13	6.75
sk_rrSVD_007	<b>7.19</b>	10.25	5.98	<b>2.87</b>	2.1	2.66	<b>8.22</b>	0.72	7.42
sk_rrSVD_008	<b>7.41</b>	10.74	6.69	<b>3.13</b>	1.44	2.96	<b>10.15</b>	1.39	8.33
sk_rrSVD_009	8.04	10.98	7.19	<b>3.36</b>	1.67	3.18	<b>10.65</b>	1.19	9.37
sk_rrSVD_010	<b>8.65</b>	11.43	7.82	<b>3.62</b>	0.93	3.49	11.48	1.31	9.85
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$	$1 \times 10^1$
cho_dec	8.43	0.93	7.61	3.36	0.67	3.27	9.47	1.91	9.25
sk_rrSVD_001	<b>3.58</b>	0.38	3.21	<b>1.19</b>	1.01	1.11	<b>3.66</b>	1.51	3.49
sk_rrSVD_002	<b>3.98</b>	0.46	3.37	<b>1.68</b>	2.76	1.39	<b>4.32</b>	2.26	4.07
sk_rrSVD_003	<b>5.44</b>	1.00	4.39	<b>1.89</b>	2.31	1.64	<b>4.9</b>	1.97	4.67
sk_rrSVD_004	<b>6.13</b>	1.17	4.57	<b>2.14</b>	2.35	1.86	<b>5.65</b>	1.81	5.39
sk_rrSVD_005	<b>6.83</b>	1.2	5.24	<b>2.36</b>	2.45	2.1	<b>6.15</b>	1.2	6.01
sk_rrSVD_006	<b>7.58</b>	0.87	5.92	<b>2.64</b>	2.05	2.42	<b>6.88</b>	2.1	6.62
sk_rrSVD_007	<b>8.89</b>	1.5	6.57	<b>2.82</b>	1.78	2.59	<b>7.5</b>	1.84	7.28
sk_rrSVD_008	<b>9.49</b>	1.62	7.19	<b>3.06</b>	1.62	2.91	<b>8.54</b>	2.16	8.2
sk_rrSVD_009	<b>10.07</b>	1.87	7.72	<b>3.29</b>	1.27	3.13	9.38	1.87	9.04
sk_rrSVD_010	<b>10.36</b>	1.53	8.36	<b>3.56</b>	1.24	3.4	<b>9.87</b>	1.48	9.53

**Table S42.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 70$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>
cho_dec	10.82	1.14	9.94	4.34	1.25	4.23	13.14	8.68	12.44
sk_rrSVD_001	<b>3.89</b>	0.52	3.5	<b>1.48</b>	1.24	1.39	<b>4.82</b>	3.09	4.48
sk_rrSVD_002	<b>4.56</b>	0.74	3.96	<b>1.82</b>	1.53	1.69	<b>5.48</b>	2.12	5.09
sk_rrSVD_003	<b>5.64</b>	0.8	4.69	<b>2.1</b>	1.4	1.96	<b>6.22</b>	2.8	5.78
sk_rrSVD_004	<b>6.57</b>	1.15	5.46	<b>2.37</b>	1.4	2.24	<b>6.96</b>	2.24	6.55
sk_rrSVD_005	<b>7.29</b>	1.25	5.95	<b>2.68</b>	1.92	2.52	<b>7.55</b>	2.09	7.25
sk_rrSVD_006	<b>8.07</b>	1.14	6.44	<b>3.02</b>	1.94	2.83	<b>8.34</b>	2.36	7.95
sk_rrSVD_007	<b>9.13</b>	1.63	7.07	<b>3.28</b>	1.77	3.06	<b>9.02</b>	2.74	8.67
sk_rrSVD_008	<b>9.59</b>	1.44	7.65	<b>3.55</b>	1.81	3.36	<b>10.26</b>	3.02	9.78
sk_rrSVD_009	<b>9.98</b>	1.65	8.42	<b>3.82</b>	1.6	3.66	<b>11.09</b>	3.5	10.58
sk_rrSVD_010	10.8	1.82	9.03	<b>4.12</b>	1.64	3.94	<b>11.78</b>	5.73	11.3
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	
cho_dec	10.37	0.67	9.57	4.77	5.2	4.26	14.24	1.49	12.39
sk_rrSVD_001	<b>3.88</b>	0.45	3.53	<b>1.54</b>	1.48	1.36	<b>5.2</b>	0.65	4.48
sk_rrSVD_002	<b>4.75</b>	1.05	4.04	<b>1.89</b>	1.49	1.71	<b>5.69</b>	0.34	5.06
sk_rrSVD_003	<b>5.85</b>	1.07	4.87	<b>2.15</b>	1.8	1.97	<b>7.21</b>	1.4	5.89
sk_rrSVD_004	<b>6.69</b>	1.3	5.33	<b>2.41</b>	2.05	2.2	<b>7.59</b>	1.22	6.53
sk_rrSVD_005	<b>7.15</b>	1.12	5.84	<b>2.63</b>	1.37	2.51	<b>8.2</b>	1.11	7.25
sk_rrSVD_006	<b>8.17</b>	1.45	6.49	<b>2.96</b>	1.15	2.82	<b>9.41</b>	1.47	7.94
sk_rrSVD_007	<b>8.68</b>	1.37	7.11	<b>3.35</b>	1.72	3.13	<b>10.04</b>	1.22	8.81
sk_rrSVD_008	<b>9.33</b>	1.57	7.85	<b>3.62</b>	2.26	3.4	<b>11.75</b>	1.76	9.72
sk_rrSVD_009	10.19	1.96	8.42	<b>3.88</b>	1.65	3.71	<b>12.88</b>	1.8	10.7
sk_rrSVD_010	<b>10.35</b>	1.39	9.01	<b>4.23</b>	1.44	4.03	13.73	1.99	11.41
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	
cho_dec	11.28	1.2	9.91	4.59	4.9	4.24	14.18	1.73	12.2
sk_rrSVD_001	<b>4.19</b>	0.6	3.61	<b>1.46</b>	0.73	1.38	<b>4.92</b>	0.68	4.4
sk_rrSVD_002	<b>5.14</b>	0.86	4.14	<b>1.84</b>	1.56	1.67	<b>5.51</b>	0.47	5.02
sk_rrSVD_003	<b>6.93</b>	1.31	4.92	<b>2.12</b>	1.47	1.94	<b>6.56</b>	1.02	5.67
sk_rrSVD_004	<b>7.94</b>	1.34	5.56	<b>2.38</b>	1.44	2.22	<b>7.21</b>	0.96	6.35
sk_rrSVD_005	<b>8.63</b>	1.23	6.05	<b>2.65</b>	1.82	2.49	<b>7.66</b>	0.75	6.85
sk_rrSVD_006	<b>9.52</b>	1.37	6.61	<b>2.97</b>	1.75	2.8	<b>8.53</b>	0.8	7.83
sk_rrSVD_007	10.68	1.68	7.3	<b>3.35</b>	2.07	3.1	<b>9.48</b>	1.04	8.48
sk_rrSVD_008	11.22	1.82	8.04	<b>3.57</b>	2.06	3.37	<b>10.79</b>	1.27	9.63
sk_rrSVD_009	11.71	2.04	8.72	<b>3.79</b>	1.72	3.52	<b>11.67</b>	1.07	10.41
sk_rrSVD_010	11.75	1.92	9.4	<b>4.12</b>	1.7	3.93	<b>12.63</b>	1.56	11.12

**Table S43.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 80$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>
cho_dec	13.32	0.8	12.55	5.72	2.28	5.54	16.84	5.41	16.11
sk_rrSVD_001	<b>4.38</b>	0.44	4.03	<b>1.84</b>	0.88	1.73	<b>5.75</b>	1.86	5.5
sk_rrSVD_002	<b>5.21</b>	0.98	4.53	<b>2.23</b>	1.34	2.05	<b>6.53</b>	1.65	6.25
sk_rrSVD_003	<b>6.66</b>	1.02	5.63	<b>2.57</b>	1.47	2.37	<b>7.24</b>	1.96	6.91
sk_rrSVD_004	<b>7.37</b>	1.14	6.23	<b>2.83</b>	1.57	2.69	<b>8.15</b>	2.24	7.84
sk_rrSVD_005	<b>8.23</b>	1.36	6.87	<b>3.14</b>	1.6	2.99	<b>8.87</b>	2.08	8.57
sk_rrSVD_006	<b>9.22</b>	1.49	7.41	<b>3.51</b>	1.97	3.33	<b>9.67</b>	2.57	9.26
sk_rrSVD_007	<b>9.85</b>	1.43	8.34	<b>3.75</b>	1.59	3.61	<b>10.4</b>	2.26	9.98
sk_rrSVD_008	<b>10.54</b>	1.64	8.81	<b>4.02</b>	1.03	3.9	<b>11.65</b>	2.35	11.19
sk_rrSVD_009	<b>11.04</b>	1.66	9.45	<b>4.31</b>	0.91	4.17	<b>12.58</b>	2.33	12.15
sk_rrSVD_010	<b>12.01</b>	2.02	9.98	<b>4.71</b>	1.39	4.53	<b>13.25</b>	2.17	12.85
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
<b>Method / coef.</b>	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>
cho_dec	13.31	0.83	12.68	5.93	5.85	5.49	17.1	10.55	15.85
sk_rrSVD_001	<b>4.31</b>	0.3	4.04	<b>1.93</b>	2.17	1.72	<b>5.66</b>	1.82	5.41
sk_rrSVD_002	<b>5.23</b>	0.91	4.61	<b>2.31</b>	2.09	2.09	<b>6.43</b>	2.29	6.09
sk_rrSVD_003	<b>6.85</b>	1.57	5.6	<b>2.59</b>	1.93	2.38	<b>7.18</b>	2.7	6.83
sk_rrSVD_004	<b>7.61</b>	1.42	6.11	<b>2.83</b>	1.82	2.65	<b>8.07</b>	2.51	7.7
sk_rrSVD_005	<b>8.3</b>	1.3	6.67	<b>3.12</b>	1.73	2.91	<b>8.76</b>	2.35	8.41
sk_rrSVD_006	<b>9.44</b>	1.69	7.36	<b>3.41</b>	0.94	3.3	<b>9.58</b>	3.18	9.07
sk_rrSVD_007	<b>10.14</b>	1.87	7.98	<b>3.71</b>	1.15	3.57	<b>10.3</b>	2.3	9.88
sk_rrSVD_008	<b>10.37</b>	1.67	8.71	<b>4.06</b>	1.54	3.92	<b>11.65</b>	4.51	11.02
sk_rrSVD_009	<b>10.69</b>	1.69	9.31	<b>4.34</b>	1.35	4.18	<b>12.49</b>	3.22	11.7
sk_rrSVD_010	<b>11.24</b>	1.72	9.97	<b>4.73</b>	1.38	4.48	<b>13.22</b>	5.07	12.61
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
<b>Method / coef.</b>	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>
cho_dec	1.34	0.7	12.8	6.09	7.36	5.48	1.92	2.75	16.05
sk_rrSVD_001	<b>0.45</b>	0.38	4.15	<b>1.86</b>	1.2	1.72	<b>0.6</b>	0.56	5.45
sk_rrSVD_002	<b>0.61</b>	1.65	4.79	<b>2.26</b>	1.63	2.05	<b>0.68</b>	0.85	6.14
sk_rrSVD_003	<b>0.77</b>	1.59	5.72	<b>2.64</b>	1.97	2.39	<b>0.81</b>	1.32	6.94
sk_rrSVD_004	<b>0.84</b>	1.47	6.18	<b>2.95</b>	2.28	2.62	<b>0.89</b>	1.4	7.7
sk_rrSVD_005	<b>0.95</b>	1.73	6.71	<b>3.22</b>	2.35	2.9	<b>0.94</b>	1.17	8.47
sk_rrSVD_006	<b>1.02</b>	1.7	7.57	<b>3.51</b>	2.25	3.24	<b>1.07</b>	1.49	9.31
sk_rrSVD_007	<b>1.12</b>	1.67	8.41	<b>3.74</b>	1.67	3.48	<b>1.13</b>	1.5	10.01
sk_rrSVD_008	<b>1.21</b>	1.82	9.32	<b>4.01</b>	1.73	3.83	<b>1.35</b>	1.85	11.19
sk_rrSVD_009	<b>1.29</b>	2.14	10.06	<b>4.35</b>	1.7	4.11	<b>1.41</b>	1.69	12.29
sk_rrSVD_010	<b>1.26</b>	1.72	10.86	<b>4.68</b>	1.63	4.45	<b>1.53</b>	2.13	12.74



**Table S44.** Process time (s) mean, standard deviation and best result for Au<sub>38</sub>Q  $K = 90$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>
cho_dec	15.73	0.41	15.18	6.9	3.12	6.55	2.14	8.02	2.05
sk_rrSVD_001	<b>4.89</b>	0.38	4.67	<b>2.16</b>	1.93	1.96	<b>0.7</b>	1.74	0.66
sk_rrSVD_002	<b>5.98</b>	1.06	5.24	<b>2.53</b>	1.52	2.32	<b>0.79</b>	2.36	0.75
sk_rrSVD_003	<b>7.4</b>	1.32	6.34	<b>2.81</b>	1.53	2.62	<b>0.87</b>	2.71	0.83
sk_rrSVD_004	<b>8.01</b>	0.79	7.18	<b>3.16</b>	2.29	2.95	<b>0.96</b>	2.8	0.9
sk_rrSVD_005	<b>8.85</b>	1.31	7.65	<b>3.48</b>	2.06	3.26	<b>1.03</b>	1.8	0.99
sk_rrSVD_006	<b>9.82</b>	1.55	8.21	<b>3.95</b>	2.19	3.7	<b>1.12</b>	2.23	1.07
sk_rrSVD_007	<b>10.1</b>	1.29	8.85	<b>4.16</b>	1.46	3.98	<b>1.2</b>	2.1	1.16
sk_rrSVD_008	<b>11.06</b>	1.65	9.58	<b>4.5</b>	2.03	4.26	<b>1.34</b>	3.35	1.28
sk_rrSVD_009	<b>11.78</b>	1.78	10.27	<b>4.79</b>	1.83	4.53	<b>1.44</b>	3.13	1.34
sk_rrSVD_010	<b>12.63</b>	2.27	10.95	<b>5.02</b>	0.93	4.87	<b>1.51</b>	1.95	1.46
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
<b>Method / coef.</b>	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
cho_dec	1.56	1.00	14.73	7.24	8.73	6.61	2.21	17.12	2.02
sk_rrSVD_001	<b>0.5</b>	0.72	4.56	<b>2.17</b>	1.74	1.94	<b>0.69</b>	1.93	0.66
sk_rrSVD_002	<b>0.6</b>	1.13	5.17	<b>2.64</b>	2.34	2.35	<b>0.78</b>	2.25	0.74
sk_rrSVD_003	<b>0.77</b>	1.69	6.3	<b>2.88</b>	1.69	2.66	<b>0.87</b>	4.44	0.82
sk_rrSVD_004	<b>0.86</b>	1.51	6.82	<b>3.18</b>	1.65	2.99	<b>0.94</b>	2.27	0.91
sk_rrSVD_005	<b>0.92</b>	1.62	7.36	<b>3.5</b>	2.2	3.27	<b>1.02</b>	2.45	0.97
sk_rrSVD_006	<b>1.05</b>	1.89	8.25	<b>3.96</b>	2.2	3.67	<b>1.11</b>	3.18	1.05
sk_rrSVD_007	<b>1.1</b>	1.94	8.98	<b>4.23</b>	1.71	4.00	<b>1.19</b>	2.89	1.15
sk_rrSVD_008	<b>1.17</b>	2.18	9.64	<b>4.53</b>	1.78	4.31	<b>1.32</b>	4.05	1.27
sk_rrSVD_009	<b>1.2</b>	1.74	10.37	<b>4.86</b>	1.9	4.62	<b>1.45</b>	6.13	1.37
sk_rrSVD_010	<b>1.29</b>	1.99	11.09	<b>5.15</b>	1.3	4.97	<b>1.52</b>	8.41	1.42
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
<b>Method / coef.</b>	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
cho_dec	1.59	0.72	15.2	7.45	10.6	6.54	2.59	4.12	2.04
sk_rrSVD_001	<b>0.5</b>	0.7	4.6	<b>2.22</b>	2.26	1.96	<b>0.75</b>	0.95	0.67
sk_rrSVD_002	<b>0.64</b>	1.39	5.23	<b>2.63</b>	2.17	2.35	<b>0.81</b>	0.58	0.73
sk_rrSVD_003	<b>0.82</b>	1.78	6.36	<b>2.88</b>	1.62	2.63	<b>0.96</b>	1.56	0.82
sk_rrSVD_004	<b>0.89</b>	1.72	6.92	<b>3.16</b>	1.75	2.97	<b>1.06</b>	1.59	0.92
sk_rrSVD_005	<b>1.03</b>	1.72	7.72	<b>3.49</b>	2.05	3.26	<b>1.1</b>	1.33	0.99
sk_rrSVD_006	<b>1.07</b>	1.47	8.46	<b>3.97</b>	1.98	3.66	<b>1.28</b>	2.17	1.06
sk_rrSVD_007	<b>1.2</b>	1.84	9.1	<b>4.22</b>	2.02	3.98	<b>1.31</b>	1.87	1.13
sk_rrSVD_008	<b>1.26</b>	1.77	9.79	<b>4.44</b>	1.49	4.2	<b>1.55</b>	2.23	1.28
sk_rrSVD_009	<b>1.36</b>	2.15	10.46	<b>4.83</b>	1.63	4.61	<b>1.67</b>	2.24	1.38
sk_rrSVD_010	<b>1.4</b>	1.83	11.17	<b>5.14</b>	2.16	4.84	<b>1.71</b>	2.68	1.45

**Table S45.** Process time (s) mean, standard deviation and best result for Au<sub>38Q</sub>  $K = 100$ . Bolded means  $\bar{x}$  differ statistically from *Cholesky decomposition*.

Dataset	AuN2-4k			AuN2-8k			AuN2-12k		
	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>
cho_dec	1.94	0.91	18.61	8.55	3.65	8.16	2.71	11.00	2.58
sk_rrSVD_001	<b>0.57</b>	0.52	5.28	<b>2.57</b>	1.97	2.37	<b>0.83</b>	1.53	0.8
sk_rrSVD_002	<b>0.67</b>	0.92	5.97	<b>2.93</b>	1.28	2.81	<b>0.91</b>	2.08	0.88
sk_rrSVD_003	<b>0.86</b>	1.48	7.4	<b>3.32</b>	2.21	3.08	<b>1.00</b>	2.37	0.97
sk_rrSVD_004	<b>0.94</b>	1.65	8.11	<b>3.69</b>	1.94	3.51	<b>1.1</b>	2.56	1.06
sk_rrSVD_005	<b>1.01</b>	1.73	8.62	<b>3.94</b>	1.56	3.76	<b>1.19</b>	1.99	1.14
sk_rrSVD_006	<b>1.09</b>	1.93	9.22	<b>4.39</b>	1.78	4.21	<b>1.28</b>	2.56	1.24
sk_rrSVD_007	<b>1.17</b>	1.85	9.94	<b>4.64</b>	1.06	4.48	<b>1.36</b>	1.99	1.33
sk_rrSVD_008	<b>1.29</b>	2.22	10.75	<b>5.11</b>	1.85	4.82	<b>1.51</b>	3.39	1.47
sk_rrSVD_009	<b>1.37</b>	2.22	11.57	<b>5.4</b>	1.72	5.15	<b>1.61</b>	2.39	1.56
sk_rrSVD_010	<b>1.4</b>	1.86	12.4	<b>5.74</b>	1.82	5.47	<b>1.69</b>	4.38	1.65
Dataset	AuN10-4k			AuN10-8k			AuN10-12k		
$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>
cho_dec	1.89	0.6	17.95	8.73	6.14	8.13	2.85	24.1	2.55
sk_rrSVD_001	<b>0.53</b>	0.18	5.06	<b>2.57</b>	2.03	2.37	<b>0.82</b>	2.01	0.79
sk_rrSVD_002	<b>0.64</b>	0.86	5.8	<b>2.99</b>	1.96	2.77	<b>0.9</b>	3.3	0.86
sk_rrSVD_003	<b>0.83</b>	1.61	6.93	<b>3.4</b>	2.32	3.17	<b>1.01</b>	11.4	0.94
sk_rrSVD_004	<b>0.91</b>	1.34	7.64	<b>3.71</b>	2.27	3.49	<b>1.08</b>	3.91	1.04
sk_rrSVD_005	<b>1.03</b>	1.79	8.11	<b>3.98</b>	2.26	3.75	<b>1.18</b>	3.13	1.13
sk_rrSVD_006	<b>1.14</b>	1.8	9.12	<b>4.46</b>	2.01	4.22	<b>1.28</b>	5.59	1.23
sk_rrSVD_007	<b>1.19</b>	1.81	9.8	<b>4.73</b>	1.52	4.53	<b>1.35</b>	2.87	1.3
sk_rrSVD_008	<b>1.29</b>	1.96	10.56	<b>5.2</b>	1.94	4.87	<b>1.53</b>	8.67	1.45
sk_rrSVD_009	<b>1.38</b>	2.08	11.5	<b>5.48</b>	1.76	5.24	<b>1.63</b>	9.23	1.52
sk_rrSVD_010	<b>1.46</b>	2.32	12.21	<b>5.73</b>	1.76	5.48	<b>1.72</b>	10.91	1.62
Dataset	AuN100-4k			AuN100-8k			AuN100-12k		
$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	$\bar{x}$	$\sigma$	min(x)	
<b>Method / coef.</b>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>	1x10 <sup>1</sup>	1x10 <sup>2</sup>
cho_dec	1.9	0.52	18.48	8.57	6.57	8.11	3.15	5.56	2.54
sk_rrSVD_001	<b>0.56</b>	0.45	5.23	<b>2.56</b>	2.06	2.37	<b>0.9</b>	1.5	0.79
sk_rrSVD_002	<b>0.76</b>	1.72	5.92	<b>2.99</b>	1.71	2.78	<b>1.00</b>	1.75	0.87
sk_rrSVD_003	<b>0.91</b>	1.64	7.03	<b>3.36</b>	1.92	3.13	<b>1.17</b>	2.57	0.94
sk_rrSVD_004	<b>1.05</b>	1.87	8.16	<b>3.67</b>	2.19	3.42	<b>1.25</b>	2.35	1.05
sk_rrSVD_005	<b>1.06</b>	1.39	8.58	<b>3.97</b>	2.14	3.75	<b>1.29</b>	1.86	1.13
sk_rrSVD_006	<b>1.2</b>	2.11	9.26	<b>4.41</b>	2.06	4.2	<b>1.51</b>	2.67	1.23
sk_rrSVD_007	<b>1.23</b>	1.78	10.12	<b>4.7</b>	2.06	4.48	<b>1.51</b>	2.4	1.3
sk_rrSVD_008	<b>1.37</b>	2.2	10.83	<b>5.11</b>	1.84	4.79	<b>1.78</b>	3.02	1.44
sk_rrSVD_009	<b>1.41</b>	1.88	11.77	<b>5.42</b>	2.09	5.12	<b>1.88</b>	2.55	1.58
sk_rrSVD_010	<b>1.5</b>	2.2	12.56	<b>5.71</b>	2.3	5.43	<b>1.91</b>	3.03	1.64



**PIII**

**FEATURE SELECTION FOR DISTANCE-BASED REGRESSION:  
AN UMBRELLA REVIEW AND A ONE-SHOT WRAPPER**

by

Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, Tommi Kärkkäinen 2023

Neurocomputing, Vol 518, 344–359

<https://doi.org/10.1016/j.neucom.2022.11.023>

Reproduced with kind permission of 2023 Elsevier.



# Feature selection for distance-based regression: An umbrella review and a one-shot wrapper

Joakim Linja<sup>a,\*</sup>, Joonas Hämäläinen<sup>a</sup>, Paavo Nieminen<sup>a</sup>, Tommi Kärkkäinen<sup>a</sup>

<sup>a</sup> Faculty of Information Technology, University of Jyväskylä, Finland

## ARTICLE INFO

### Article history:

Received 8 March 2022  
 Revised 23 September 2022  
 Accepted 4 November 2022  
 Available online 10 November 2022  
 Communicated by Zidong Wang

### Keywords:

Distance-based method  
 Feature selection  
 Feature saliency  
 Wrapper algorithm  
 EMLM

## ABSTRACT

Feature selection (FS) may improve the performance, cost-efficiency, and understandability of supervised machine learning models. In this paper, FS for the recently introduced distance-based supervised machine learning model is considered for regression problems. The study is contextualized by first providing an umbrella review (review of reviews) of recent development in the research field. We then propose a saliency-based one-shot wrapper algorithm for FS, which is called *MAS-FS*. The algorithm is compared with a set of other popular FS algorithms, using a versatile set of simulated and benchmark datasets. Finally, experimental results underline the usefulness of FS for regression, confirming the utility of certain filter algorithms and particularly the proposed wrapper algorithm.

© 2022 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Dissimilarity has a central role in unsupervised learning, but the increased popularity of using distance-based models for supervised problems (e.g., [1–4]) illustrates how the gap between unsupervised and supervised learning tasks is diminishing. In fact the dissimilarities among a set of prototypical observations can be used as features with any predictive model [5,6]. Further, the history of distance-based supervised models can be traced back to the radial basis function networks [7,8] with a linear kernel [9,10]. The latter references, as noted in [Remark 1] [4], provide proof for the universal approximation capability of the linear distance-regression model. The scope of this article is to consider dimension reduction, specifically feature selection (FS) for this distance-based learning machine [4]. Compared to its sibling method, feature extraction (FE), FS keeps the features as they are while removing those considered unnecessary. FE, on the other hand, aims to reduce the number of features for example, through projections [11]. In this way, FE mixes information of the original features.

The need for FS stems from increasingly complex and demanding datasets where the number of features may become detrimental to the practical operation of a machine learning model [12–14]. FS refers to the identification and selection of a subset of relevant features for a data-based model. Therefore, it is basically a search problem, which can generally be addressed using many techniques

(e.g., forward or backward search, exhaustive search, branch-and-bound, evolutionary approaches) and multiple feature assessment criteria (information, distance, dependency, consistency, and accuracy measures) [15]. The FS process has three main goals [16]: improve the model performance, provide faster and more cost-effective models, and improve the understanding of the data generation process. However, the last goal cannot be fully addressed when working with an already featured, secondary dataset. The classical division of the main types of features is given in [17,18]: *irrelevant*, *weakly relevant*, and *strongly relevant*. The weakly relevant features were further categorized in [19] as *redundant* or *non-redundant*.

The main branches of FS techniques are filter and wrapper approaches [17], depending on whether the intended machine learning model itself is used in the FS process. Usually, this means that a filter approach is faster, and a wrapper approach is more accurate [18]. Filters usually contain two main steps [20]: 1) ranking of features according to importance scoring; and 2) selection of most important features based on step 1). Hybrid [21–24] or embedded methods [25,26] perform FS by using another model or an untrained model to assess feature relevancy. Embedded methods that rank and select features during the construction of the predictive model include decision trees [27] and ensemble learning methods, most prominently, random forests [28,29].

For wrappers, the search phase of the used features means multiple repetitions of training, i.e., the estimation of the model's parameters. Therefore, one aspect of categorizing different wrap-

\* Corresponding author.

pers is given by the number of model trainings needed during the search of the final feature subset. This number can be very large, for instance, when optimization-based methods and metaheuristics are used to search the features (see Section 2.4). Here, we propose a *one-shot wrapper*: rank the features using scores computed from a trained distance-based model with the full feature set and, through a threshold, select the most important ones for the reduced model. The entire endeavor to determine the final model then requires two training rounds – initially with the original feature set and finally with the selected feature set. In between, the full feature set model is used to compute feature importances for the thresholding. It should be noted that if the feature ranking based on the full model is computationally not more expensive than training the model, then this one-shot approach does not add any computational complexity to the (unavoidable) model training.

Feature scoring and ranking is actually a specific technique to quantify and improve the understandability of a model, to explain its behavior [30]. Indeed, interpretable machine learning refers to the ability to understand the work logic of machine learning models and algorithms [31]. A branch of techniques for this purpose uses the saliency of features to rank their explanatory power as a post hoc explainability approach [32,33]. For neural networks (NN), one measure of saliency is the input sensitivity, i.e., the partial derivative of the network’s output with respect to its input. For shallow networks, feature assessment originating from this idea was proposed in [34] and, since then, many similar FS techniques have been considered [35]. Use of a partial derivative method was rediscovered within the context of deep neural networks in [36], where the first-order Taylor’s expansion was used to generate an image-specific saliency map for visual interpretation of a convolutional neural network classifier.

In this work, we provide a wide-ranging overview of earlier FS research, noting that explorations of FS for distance-based regression methods have been scarce. Thus, we aim to fill this gap by proposing and evaluating a new FS algorithm for this learning task. Our main contributions are as follows:

- an umbrella review of recent reviews on FS
- derivation of a one-shot FS approach for distance-based regression
- extensive experimental comparison of filter and wrapper FS algorithms, ensuring the viability of the proposed algorithm and its building blocks.

The rest of the paper is organized as follows. Section 2 presents the umbrella review. Section 3 introduces the proposed FS algorithm and the necessary mathematical foundations. Section 4 describes the used synthetic datasets as well as presents the used open access datasets. In addition, it presents the evaluation criteria used in measuring the gained results. Section 5 details the experiments and results while also presenting the related discussion. The data tables with the results discussed in Section 5 are in the appendix. Finally, Section 6 discusses and concludes the work.

## 2. Umbrella review on FS techniques

FS is a particular instance of model selection for which a considerable number of techniques have been depicted and experimented with over the years [37]. Therefore, the full coverage of the development and current status of the FS field of research from primary studies, after the influential classical works like [38,39,15,16], is out of the scope of this article. However, in order to position our work in the research field, the recent developments of FS research will be summarized through an umbrella review, the

purpose of which is to locate and consider different views and perspectives on a broad area of interest [40]. Here, instead of addressing the primary studies in the field, the already undertaken surveys and reviews and their summarization are considered [41]. Further, the umbrella review is a common practice in the medical research domain where summarizing the vast amount of knowledge from primary articles is a tedious task. However, while this type of review is rarely used in the machine learning field, there was one such recent study related to deep learning [42]. Similar to the deep learning field, there exist many recent reviews for FS, which indicates the need for conducting an umbrella review for FS as well.

For this purpose, we used Google Scholar on December 9–10, 2021, with the search term “feature selection review” and checked the first 500 returned links. Of these links, 32 high-quality reviews and summaries on FS for supervised learning, published since 2013 in journals by leading publishers (IEEE, ACM, Elsevier, Springer, Wiley), were identified to be summarized next. The final paper of the search that was included in the summary, [43], had an entry number of 475. This was the only hit for the last 50 checked papers. The annual number of reviews and summaries identified was as follows: one paper from 2013, three from 2014, two from 2015, three from 2016, three from 2017, seven from 2018, four from 2019, seven from 2020, and two from 2021. The numbers indicate an increasing trend in summarizing the overall research achievements of the research field.

The narrative review of the papers is primarily organized in a chronological order. From the first paper [44] onward, we have not repeated the shared contents but tried to provide only new, relevant information from the chronologically subsequent papers. However, reviews addressing a common topic are presented together, and this defines the subtitle structure used below. The subtitles are ordered according to the publication year of the first review article of the topic, although a review of general FS reviews is presented first. Due to the abundance of abbreviations in the umbrella review, we have included Table 1, in which they are listed.

### 2.1. General reviews

A general review on FS in classification was provided in [43]. Correlation criterion and MI-based criteria were depicted for filter methods. Wrapper methods were classified into Sequential Selection Algorithms (backward search) and Heuristic Search Algorithms (use of a GA) to identify a subset of features. For embedded methods, MI and weights of a classifier were depicted as feature assessment criteria. As classifiers, SVM and RBFN were introduced. Experiments with six datasets (Breast cancer, Diabetes,

**Table 1**  
List of abbreviations used in the umbrella review.

ANN	Artificial Neural Network	Mb	Markov blanked
BNS	Bi-Normal Separation	MB	Markov boundary
CBM	Correlation-Based Methods	MCO	MultiCriteria Optimization
DBM	Deep Boltzmann Machine	MD	Maximum Discrimination
DF	Document Frequency	MH	MetaHeuristic
DT	Decision Tree	MI	Mutual Information
ELM	Extreme Learning Machine	MLM	Minimal Learning Machine
EMLM	Extreme MLM	MLP	MultiLayered Perceptron
FS	Feature Selection	NB	Naive Bayes
GA	Genetic Algorithm	RBFN	Radial Basis Function Network
IG	Information Gain	RF	Random Forest
kNN	k-Nearest Neighbors	SVM	Support Vector Machine
LR	Linear Regression techniques	SVR	Support Vector Regression
LRR	Logistic Ridge Regression	TF-IDF	Term Frequency-Inverse DF

Ionosphere, Liver disorder, Medical, Fault mode) demonstrated the usefulness of FS but did not provide any methodological rankings.

The second general review encountered was [45]. However, after introducing eight different algorithms for FS including maximum variance, Laplacian score, spectral regression, sparsity favoring, etc. approaches, the experiments were only performed for unsupervised FS. Therefore, we did not consider this work further.

In [46], a large and very comprehensive FS review organized from a data perspective, mainly for ranking the features or identifying feature subsets through sparsity-favoring linear learning, was given. The different cases for FS were defined as follows: traditional FS and FS with structured features for conventional data; FS with linked data, multi-source FS, and multi-view FS for heterogenous data; and FS with streaming data or streaming features for streaming data. For conventional data, five similarity-based score criteria (Laplacian Score, SPEC, Fisher Score, Trace Ratio, and ReliefF) were introduced. Then, nine information theory-based methods (IG, MI, Minimum Redundancy Maximum Relevance, Conditional Infomax, Joint MI, Conditional MI, Interaction Capping, Double Input Symmetrical Relevance, and a Correlation-Based Filter) were depicted. Third category of methods for conventional data favored sparsity. Their construction was based on a suitable loss function with nonconvex regularization using the  $\|\cdot\|_p$ -norm for  $0 \leq p \leq 1$  or  $\|\cdot\|_{p,q}$ -norm for  $p > 1$  and  $0 \leq q \leq 1$  with both vector- or matrix-valued unknowns. The Actual formulations for different methods are given in [Section 2.3][46]. In statistical methods, the features with low variance or t-score or Chi-square score or with a high Gini index are eliminated. In Correlation-based FS (CFS), one searches a feature subset which has a strong correlation with class labels but weak inter correlations. FS with structured features included Group Lasso, and its sparse and overlapping variants. For tree structured features, a Tree-guided Group Lasso was introduced, and for graph structure, a Graph-Lasso with two additional variants (GFLasso and GOSCAR) was defined. For linked data, FS using graph regularized least-squares, user-post relationship regularizer, and unsupervised techniques encoding latent and low-rank representations were depicted. Multi-source FS could be addressed using Geometry-Dependent Covariance Analysis and multi-view scenarios, which refer to FS from different feature spaces simultaneously by using linear least-squares with specially constructed desired outputs, constraints, and sparsity favoring regularizers [Section 4.3] [46]. For streaming data with feature streams, assuming a constant number of instances for e.g., Grafting Algorithm (Lasso-like method) and Alpha-Investing Algorithm (a statistical threshold technique) were described. For actual data streams, a particular online FS algorithm and an unsupervised least-squares approach were introduced. Entire sections in the review were dedicated to the evaluation of different FS approaches and open problems in the field. This paper was clearly one of the most comprehensive reviews that was found, even though it should be noted that the linkage between a particular FS technique and the form of data is not a function but a relation: many FS techniques are suitable or modifiable for use with many forms of data.

A general FS review was provided in [47], where the techniques were not directly introduced but discussed through a Problem-Solution-Discussion contents presentation model. Many of its themes coincided with those in [46] as summarized in the previous paragraph, so we only depict the additional topics covered: distributed algorithms for FS (utilizing, e.g., MPI, MapReduce, and peer-to-peer networks), multi-label FS (of which a dedicated review was provided in [48] and summarized in Section 2.6), privacy-preserving FS (where the overall privacy degree of the chosen features is controlled), and adversarial techniques for FS (based

on currently popular adversarial network architectures and attacks on the classification model).

In [49], FS in machine learning was addressed on a general level. However, this high quality review was superseded by the even more extensive exposure in [46] summarized above. Nevertheless, in relation to wrappers, clustering-based unsupervised techniques, GAs, and particle swarm optimization methods were presented, for which a thematic overview is given below in Section 2.4. Semi-supervised FS, which was more thoroughly summarized in [50] as depicted in Section 2.5, was also examined. Future challenges were linked to the properties of data (small, large, imbalanced) and ensemble or online FS techniques.

The FS for ensemble-based machine learning models was reviewed in [51]. The basic relation between ensembles and FS is composed on the triplet of {base learners} {feature subsets} {observation subsets}. Clearly, we can link filters, wrappers, or embedded FS methods to all learners or perform FS individually in the base learners. Interestingly, RF, which is truly a prototypical feature-subset based FS method, was not addressed explicitly in this review. However, a comprehensive depiction of the existing FS tools and techniques that are available on the commonly used software platforms was presented.

In [52], causality-based FS was reviewed, and a new open-source library was proposed and tested. Instead of co-occurrences and correlations within a set of features and targets (usually labels), causality refers to the identification of cause-and-effect relationships typically using graph-based graphical models such as the Bayes/Bayesian Networks with Markov blanket (Mb) or Markov boundary (MB). For MB, which is the minimal set of Mb referring to a subset of variables containing all necessary information to infer a random variable, this review first classified (into five categories of learning) and described 30 different constraint-based FS algorithms. Similarly, three categories with eight algorithms for score-based (scoring or the actual cost function and how it is searched) identification of MB were described. Furthermore, four categories of 18 algorithms to separate features of direct causes (parents) from those with only direct effects (children) were depicted. The new toolbox, CausalFS, was then presented with a list of future challenges regarding form and quality of data (see Section 2.3), causal effect estimation, and causal FS for NNs.

The most recent review in our umbrella review was [53], which provided a clear introduction to search methods and mechanisms in FS. The measures which originated from four categories (statistics, probability, similarity, and sparsity) were a proper subset of those provided in [46]: evolutionary FS algorithms are described in the individual reviews in Section 2.4, and the additional SVM-RFE method mentioned is already part of the first review in here [44]. However, compared to Section 2.5, the body of domains where FS is needed was enlarged to cover natural language processing, emotion recognition, speech processing, sentiment analysis, and biometrics. These and other domains were carefully linked to primary publications, datasets, evaluation measures, and future challenges.

## 2.2. Filter methods

One of the most popular class of methods for FS filters are the information-theoretic methods that utilize MI. These techniques were reviewed in [54]. The basics of information theory and key concepts of filters (relevance, redundancy, and complementarity) were presented. The main results of the work were a unifying framework and a list of open problems without any empirical experiments.

The similarity of the FS techniques, especially the rankings they provided were compared using the Kuncheva index averaged over



all pairwise comparisons in [55]. Five univariate ( $\chi^2$ , IG, Symmetrical Uncertainty, Gain Ratio, and OneR) and three multivariate (ReliefF, SVM-ONE, and SVM-RFE, the latter two both with linear kernel) FS methods were considered for 16 classification datasets with the number of features ranging from 1559 to 10 458. Similar behavior was observed for the three first univariate methods, whereas different behavior of Gain Ratio was witnessed. Difference of multivariate methods compared to the univariate ones was also observed, together with the sensitivity of SVM-related methods on their internal parameters concerning the number/portion of features discarded at each iteration. In this respect, ReliefF had the most stable and consistent behavior.

In [56], structured, sparsity-inducing methods are presented by separating vector-based and matrix-based FS. However, even though the work provides a comprehensive survey, it is superseded by the even more extensive exposure [46] already summarized above.

One of the most commonly used filters, as also demonstrated by this umbrella review, are Relief-based algorithms (RBAs), which were comprehensively depicted and reviewed in [57]. Altogether, 21 variants of RBAs from four general branches were presented, and among them, from our perspective, the most important being the regression-oriented ReliefF with  $\mathcal{O}(N^2n)$  computational costs for the number of observations  $N$  and the number of features  $n$ . Because feature scoring in Relief is based on the feature value differences between a target and its neighboring observations, this filter is particularly relevant to our experiments presented in Sections 4 and 5.

A large comparison of filters for classification, utilizing 16 high-dimensional datasets and a specific R-package *mlr*, was presented in [58]. The 22 filters, also visible in other papers of this umbrella review, originated from statistical tests, feature variance, univariate predictive performance, feature importance with RF, and MI. The classifiers were kNN, LRR, and SVM. In the analysis, similarity of feature scores for ranking was first assessed, by identifying three groups of similar filter methods. The actual comparison for (data, model, filter)-triplets was generally concluded as follows “no filter method is better than all the other methods on all data sets,” and “there is no subset of filter methods that outperforms all other filter methods.” Because of these conclusions, it was recommended to test all filters in a particular context if computational resources suffice.

### 2.3. FS for particular forms of data

The use of synthetic data allows for rigorous comparison between the selected features and accuracies of the reduced feature models. In [44], FS for synthetic data classification was reviewed. Altogether, seven filters (correlation-based method, consistency-based filter, the Interact algorithm, IG, ReliefF-algorithm, the mRMR method, and the  $\mathcal{M}_d$  filter), two embedded methods (SVM-RFE for SVM and FS-P for Perceptron), and two wrappers (Wrapper-C4.5 and Wrapper-SVM using the WrapperSubsetEval algorithm) were applied over eleven synthetic datasets (CorrAL, CorrAL-100, XOR-100, Parity3 + 3, LED-25, LED-100, Monk3, SDI1-3, and Madelon), which included irrelevant and redundant features, noise, and various interaction patterns. Moreover, four classifiers were used (NB and IB1 in addition to C4.5 and SVM). In this work, the challenges were pointed out in the threshold selection for methods that produce feature importance values and, therefore, allow ranking of individual features. They concluded that ReliefF and SVM-RFE with nonlinear kernel were the best methods, and recommended the former because of its independence on the classification model and computational efficiency.

Additionally, the difficulties in comparing and ranking wrapper methods were also noted.

Online settings provide a special context on the availability of data for feature construction. In [59], FS for streaming data classification was considered, with the full or only a partial subset of features being accessible for each arriving new instance (decided by the learner). Next, three novel algorithms, a truncated perceptron, a sparse projection approach, and learning with partial inputs, were presented. The experimental comparison was performed using nine smaller datasets (magic04, svmguide3, german, splice, spambase, a8a, RCV1, and two topic pairs from 20Newsgroup) and five larger datasets (KDDCUP08, ijcnn1, codrna, covtype, and KDDCUP99), focusing on the average number of mistakes made by the algorithms. Further, real word applications of image classification in computer vision and microarray gene expression analysis in bioinformatics were also demonstrated. The overall conclusion was that the proposed online algorithms turned out to be scalable and more efficient than some state-of-the-art batch FS techniques. However, upon taking a closer look, this paper turned out to be a primary study.

A particular focus on online FS with streaming features — a sub-topic also covered in the extensive review [46] summarized in Section 2.1 — was undertaken in [60]. The additional FS techniques compared to [46] contained MI-based SAOLA and Group-SAOLA (Scalable and Accurate OnLine Approach), uncertainty-minimizing GFSSF (Group FS with Streaming Features), and Lasso-oriented OGFS (Online Group FS). Experiments with several (over 10) benchmark data sets did not provide methodological conclusions or rankings but, instead, generated a list of challenges related to multi-label cases (reviewed separately, e.g., in [48] as summarized in Section 2.6), quality of data in real-world applications, and the need to distribute computational efforts.

### 2.4. Optimization-based FS techniques and metaheuristics

A general approach for FS is to cast the problem of identifying a subset of features as an optimization problem. Such an approach needs the definition of a cost function that measures the goodness of a feature subset, typically through the accuracy of a classifier. However, the strict convexity and differentiability of such cost functions might be difficult to establish, so derivative free optimization methods provide a natural family of optimizers in these settings. Clearly, both the necessity of the metalevel fitness and the search that finds arguments of its extremums causes a significant increase in the computing time.

In [61], nature-inspired metaheuristics including GAs and ant colony optimization were reviewed for FS. Further, a taxonomy of such approaches consisting of stochastic algorithms, physical methods, evolutionary approaches, immune systems, and swarm intelligence were also depicted. Then, the elements and basic constituents of population-based approaches, memetic algorithms incorporating local searchers, clonal selection, harmony search, simulated annealing, tabu search, and swarm algorithms (artificial bee colony, ant colony, firefly algorithm, and particle swarm) were given. Experiments with 12 UCI datasets and C4.5 and NB classifiers concluded the capabilities of all tested algorithms in finding good solutions. Similar to [44] as shown above, in some cases, filter-based evaluators had better results as compared to the more complex FS approaches.

Years 2020–2021 were characterized by multiple FS reviews addressing optimization-based techniques. The latest is [62], which focused on nature-inspired MH techniques from both mapping (how much and what kind of publications) and review (what techniques and results) perspectives. Among others, the large number of nature-inspired MH techniques was summarized: 29 were inspired by insects and reptiles, 15 by birds, 13 by animals,



seven by sea creatures, five by plants, six by humans, and 25 other techniques. Altogether, 21 actual FS-MH algorithms were then listed, which were further divided into chaotic (utilizing various forms of randomness) and binary (strict inclusion/exclusion of features) variants. Moreover, the review [62] superseded a few other recent, more specific FS reviews: the grey wolf optimizer treated in [63,64] which considered the Dragonfly algorithm. Moreover, swarm optimization, which was examined in [65], introduced six different algorithms, which were all contained in five categories and 12 instances of "swarm" presented in [62]. Differently from [62], the chaotic category there was replaced with continuous representation of features. The review in [65] was summarized with the observation that MHs are typically applied to identify both the features and predictive model's parameters, and that FS problems with binary presentation need further studies.

Another perspective on optimization-based FS was detailed in [66], where a systematic review on the use of MCO was presented. In all the introduced cases, the multiple criteria were reduced to two basic objectives: minimal number of features with maximal classification performance. Contents of 38 papers were summarized, where the twofold nature noted above meant that most of the papers depicted wrappers and only five filters. Because of the computational costs of MCO algorithms, the mentioned classifiers included rather simple techniques like kNN, NB, DT, and linear SVM, but also SVM, RF, ELM, MLP (i.e., shallow feedforward network), DBM, and Deep NN. Notably, this review also summarized 38 different datasets that were used to evaluate the methods.

### 2.5. FS in particular application domains

Gene expression data, which was focused on in [50], is characterized by a high ratio of the number of features to that of samples: there can be up to hundreds of thousands of features but only a small sample size. In this paper, a thorough introduction to feature evaluation and selection methods was given along with comprehensive summaries of the prediction accuracy vs. number of selected features for dozens of studies with five popular datasets. Expectedly for such problems, the usefulness of filters and the potential of semi-supervised FS methods integrating unsupervised FS from larger unlabelled data with supervised construction of classifiers was concluded. Similarly, the potential of hybrid FS methods combining multiple filter and/or wrapper approaches was emphasized.

FS in the multimedia context, covering, for example, texts, images, videos, audios, animations, etc. as formats and forms of data, was reviewed in [67]. The basics of FS methods and search strategies were depicted with summaries of their use in supervised, semi-supervised, and unsupervised FS techniques for multimedia data based on 70 original papers in 2001–2017. Interestingly, compared to our paper, years 2013 and 2014 were identified as the most active times of publications especially through the emergence of various heuristics. A special emphasis in the current review was given to interactive, active learning -based approaches. However, with both these techniques as well as in the whole research field, several open issues and challenges were identified. Additionally various metrics to evaluate the performance of FS methods with multimedia data were presented.

FS in the application domain of renewable energy was considered in [68]. This was chronologically the first study where regression problems had an explicit role. This was illustrated in the more detailed FS reviews on the following: *i) Wind Energy Prediction* using NN, Gaussian Process, kNN, ELM, SVR, RF, Boosting machine, and Nonlinear Auto-Regressive models, where FS was performed via optimization-based methods (see Section 2.4), and Empirical Mode Decomposition, *ii) Solar Energy Prediction* using correlations,

Lasso, and optimization-based but mainly intrinsic (i.e., domain and data-specific) FS methods for NNs, Deep NNs, SVMs, and ELMs; *iii) Marine Energy Prediction* using rule- and optimization-based FS methods mainly for ELMs, and *iv) Energy-Related Problems* in general using, again, optimization-based FS methods with ELMs and SVMs, RRelief with NNs, and entropy-based filters with, for e.g., RF and NNs. Over half (18) of the 32 reviewed papers used wrappers for FS.

Genomic big data, similar to [50] as shown above, was addressed in the systematic review in [69]. Most of the identified papers proposed new methods, architectures, and tools for processing genomic data, thus overlapping with other reviews mentioned in this paper. A terminological exception was the Integrative FS methods, which depicted multiple hybrid approaches with different datasets and/or FS methods as a preprocessing step before the actual training of a model (with or without FS).

A systematic review on FS for forecasting spatiotemporal traffic data (how much traffic, where) was presented in [70]. From the FS perspective, the categorization of the identified literature followed the normal model except for the division of filters into so-called *Exogenous and Endogenous feature filtering methods*. The latter encapsulated the typical filters like correlation and sparse linear regression-based methods. Whereas the former referred to the use of external data and knowledge to limit possibilities and useful features, such as, knowing that a car is moving in a specific direction at a certain speed. Additionally, optimization-based wrappers, and embedded methods using, for e.g., deep learning techniques were listed. In 211 papers from 1984–2018, a versatile pool of prediction methods were found including the following: Feedforward shallow and deep ANNs; time-delayed, recurrent, long short-term memory, convolutional, autoregressive exogenous ANNs; Deep belief and Bayesian networks; kNN; autoregressive models; Gaussian Process regression; RF and Regression tree; and Tensor decomposition models. It was concluded that urban traffic forecasting in particular needs further empirical FS studies.

Text classification and FS were the scope of the review in [71]. In this application domain, the starting point is the numerical encoding of texts and documents by using, for instance, the classical bag-of-words representation. This is a particularly interesting domain from the point of view of the distance-based methodology because of the key role of similarity of documents especially in unsupervised scenarios. The classifiers summarized in the review are the common ones: kNN method, NB, relevance-based Rocchio, multivariate regression models, DTs, SVMs, NNs, graph partitioning-based approach, and GA-based methods to train the models. From the FS perspective, the text domain is very similar to the genomic data due to the number of features being large when compared to the number of observations in both. Filters in the field result from preprocessing-like techniques such as DF and TF-IDF, as well as more traditional CBM, MI, IG, Term-Relatedness,  $\chi^2$ , MD, LR techniques, BNS along with a few special filters. However, wrappers and embedded methods were only briefly addressed in this review. Interestingly and independently, the review was concluded with summarizing some recent FS categories using almost the same topical division as in our umbrella review.

FS in image analysis was considered in [72]. In this domain, one can distinguish low-level, mid-level, and high-level techniques, where the first refers to pixel-/voxel-level tasks like classification and segmentation, the second to the derivation of features and characteristics from images (typically for low-level tasks), and the last, for e.g., to image annotation, i.e., identification of objects and/or their labels. The actual methods and techniques summarized in the review are mostly the same as those already addressed

in many other papers in this umbrella review, with notable exceptions concerning multiple mentions of fuzzy-rough set FS. Out of c. 50 papers reviewed, more than half referred to the use of filters, 13 to embedded techniques, and 11 to wrappers. This review also depicted the main available datasets for FS and performed a small (four datasets times four methods) experiment with the following overall conclusions: results were dependent on all aspects, the classifier, the FS method, and the dataset, with the recommendation to use the subset FS methods with SVM or RF.

## 2.6. FS in multi-label classification problems

FS in multi-label classification (MLC) problems that are potentially characterized by many simultaneously active labels per instance was presented in [48] using a systematic literature review process. The authors first noted that the use of the straightforward Binary Relevance (BR) method allows for the usage of all single-label FS methods in MLC cases. In the paper, another feature construction method to build binary variables taking into account correlations between multiple labels was depicted. Experiments with 10 MLC datasets, a multi-label extension and the adaptation of kNN-classifier as well as the IG -based filter with BR were presented. The proposed method showed competitive performance with slightly increased computational costs. Finally, the literature review of 99 papers concluded that 70 applied a filter approach in FS.

Another review that focused solely on FS in MLC problems (MLC-FS) was [73]. The MLC-FS was considered from a taxonomy of four perspectives: label, search strategy, interaction with the learning algorithm, and data format. As a whole, this review basically linked the different problem transformation and algorithm adaptation methods of MLC problems with different existing classification models and FS techniques already covered above (for instance, the supervised, semi-supervised, and unsupervised treatment in [67]). In conclusion, the popularity of filters was observed.

Additionally, another review on MLC-FS was undertaken in [74]. Again, the characteristics of addressing MLC problems and a large catalog of existing FS methods were addressed through the analysis of primary publications. The developed taxonomy embeds the known triplet of filter, wrapper, and embedded FS methods into a MLC-specific hierarchy, consisting of direct and transformation-based approaches; the latter was further divided into single and internal/external BR categories.

## 2.7. Summary

Let us briefly summarize our findings. First, the years covered in different reviews varied substantially, naturally depending on when particular techniques (search- and optimization techniques, classifiers, etc.) actually emerged: for e.g., [70] covered years 1984–2018 and [62] 1983–2019, whereas [63] covered 2012–2020.

Next, we did not find any reviews even mentioning distance-based ML models or focusing solely on FS in regression problems, although [68] mainly considered regression tasks. Regarding classification tasks (see, e.g., [45]), as compared to regression problems, the existence of labels opens up possibilities for both filter methods (e.g., statistical tests to assess how strongly features separate the classes) and for embedded and hybrid methods (e.g., using one classification model for FS and another one as the actual classifier [75]).

Interestingly, many papers noted the existence of cases where filter methods performed either equally or even better than the more complex approaches (e.g., [44,61,50]). Further, for filters, the importance of threshold detection was emphasized in [76].

In general, FS using optimization means the generation of a higher-level search process, which inevitably increases the computational complexity. Other forms of filter and wrapper methods can be more direct: if they can provide a ranking on the importance of the given set of features, then the FS problem reduces to finding a rule that identifies the ranks that are large enough to be included and those that should be omitted from the final model. This is the exact method that is proposed next: Through construction and analysis of the feature importances of the predictive model (one feature sensitivity formula) and the direct generation of the inclusion/exclusion rule means no increase in the overall computational complexity and no addition, iterative search procedure.

To conclude this umbrella review – as readily stated in the first included article, [44], and confirmed (for filters) in one of the last reviewed papers [58] – there does not exist one, single “best method” for FS as different methods have their own strengths and weaknesses [61]. Therefore, identifying a good method for a specific problem setting drives the development of the research field, and in this article, our focus is on FS for regression problems. Our umbrella review shows a major research gap in recent years relating to FS for regression when compared to FS for classification. Therefore, our contributions in this paper seem timely and essential towards filling this gap.

## 3. Distance-based one-shot wrapper

In this section, we summarize the essence of the distance-based regression model and derive the one-shot wrapper.

### 3.1. EMLM

EMLM is a supervised distance-based machine learning method. It combines the regularized ridge regression-type learning characteristics of the ELM [77,78] with the distance-based feature map used in the MLM [1,79]. It was proposed by Kärkkäinen [4] and due to its origins, this technique is referred to as EMLM. This model has a structural resemblance to RBFNs with a linear kernel [9,10]. However, the algorithms that select most or even all observations as reference points for the distance-based kernel [4,79] differentiate the overall technique from the RBFNs: reference points for MLM and EMLM are always selected from among observations; not, e.g., as cluster centers. Therefore, the EMLM incorporates only one metaparameter – the number of reference points – and when used with the *RS-maximin* [79] reference point selection algorithm, it provides a deterministic and simple-to-use supervised learning method [4].

The *RS-maximin* method has its origin in the K-means seeding approach [80] known as maximin or the furthest point selection. This seeding approach, in turn, originated from the traveling salesman problem, where it is known as the greedy permutation [81]. The *RS-maximin* approach selects the first reference point as the closest point to the input data mean and then adds the rest of the reference points deterministically with the farthest-first-traversal algorithm. For regression problems, maximizing the input space reference points' pairwise distances is known to improve the MLM's generalization performance [79]. MLM was also found to have the tendency not to overlearn [4,79,82,83].

The training phase of the EMLM is depicted in [Algorithm 3] [4]. Construction of the distance-based regression model starts by computing the distance matrix  $\mathbf{H} \in \mathbb{R}^{m \times N}$  as

$$(\mathbf{H})_{ij} = \|\mathbf{r}_i - \mathbf{x}_j\|_2, \quad i = 1, \dots, m, j = 1, \dots, N, \quad (1)$$

where  $\mathbf{r}_i \in \mathbb{R}^n$  is the  $i$ :th selected reference point and  $\mathbf{x}_j \in \mathbb{R}^n$  denotes the  $j$ :th observation. Here,  $n$  is the number of features,  $m$  denotes the number of selected reference points, and  $N$  specifies

the number of observations in the training set. Distance regression weights  $\mathbf{W} \in \mathbb{R}^{p \times m}$  are then solved from the linear problem

$$\mathbf{W} \mathbf{H} \mathbf{H}^T + \frac{\alpha N}{m} \mathbf{I} = \mathbf{Y} \mathbf{H}^T, \tag{2}$$

where  $\mathbf{Y} \in \mathbb{R}^{p \times N}$  (in the datasets used in this paper,  $p = 1$ ) contains the desired output vectors in its columns, and  $\mathbf{I} \in \mathbb{R}^{m \times m}$  is the identity matrix whose multiplier includes the fixed regularization parameter  $\alpha = \sqrt{\varepsilon}$  corresponding to the square root of machine epsilon  $\varepsilon$ . The predicted output  $\mathbf{y}$  of a trained EMLM for a given input  $\mathbf{x}$  is  $\mathbf{y} = \mathbf{W} \mathbf{H}$ , where  $(\mathbf{H})_{i1} = \|\mathbf{r}_i - \mathbf{x}\|_2$ ,  $i = 1, \dots, m$ . The usage of a trained EMLM model consists of computing the distances between new inputs and fixed reference points as shown in (1) and multiplying by the weight matrix  $\mathbf{W}$  to calculate the predicted output. The dimensions of  $\mathbf{W}$  depend on the number of targets in a dataset. In this paper, we use datasets with single targets, so  $\mathbf{W}$  is a row vector of length  $m$ .

### 3.2. Feature scoring using mean absolute sensitivity

Next, we delineate the wrapper approach for the distance-based model. It should be noted that a sampling-based technique for feature scoring and selection with EMLM, similar to [28,29], was proposed and tested in [84].

One form of the classical Taylor’s formula as given in [Lemma 4.1.5] [85] reads as follows: in the neighborhood of a point  $\mathbf{x}_0 \in \mathbb{R}^n$ , there exists  $\mathbf{z} \in l(\mathbf{x}, \mathbf{x}_0)$  (a line segment connecting the two points) such that for  $\mathbf{y} = \mathbf{x} - \mathbf{x}_0$ ,

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{y} + \frac{1}{2} \mathbf{y}^T \nabla^2 f(\mathbf{z}) \mathbf{y}, \tag{3}$$

where  $\nabla f(\mathbf{x}_0)$  denotes the gradient vector at  $\mathbf{x}_0$  and  $\nabla^2 f(\mathbf{z})$  the Hessian matrix at  $\mathbf{z}$ , respectively. Usually, this formula is used to underlay a linear approximation or second-order optimization algorithm. Analogous to the latter case, we note that a small value of an individual gradient component  $\nabla_j f(\mathbf{x}_0)$  is linked to the weak relevance of the  $i$ th feature in depicting the function’s local behavior. This observation suggests the inclusion of a feature importance criterion  $\mathcal{F} \in \mathbb{R}^n$  in [35,86], which was based on Mean Absolute Sensitivity (MAS) of the training data:

$$\mathcal{F} = \frac{1}{N} \sum_{j=1}^N \frac{\partial \mathcal{M}}{\partial \mathbf{x}_j}, \tag{4}$$

where  $\mathcal{M}$  denotes the output of the distance-based regression model. Note that the use of the Cityblock distance makes  $\mathcal{F}$  both robust and independent between the features (see [87]). The analytic derivative of the output with respect to the input vector  $\mathbf{x}_j$  is straightforward to compute, yielding a penalized expression similar to that of the unsupervised case in [formula (3)] [88]:

$$\frac{\partial \mathcal{M}}{\partial \mathbf{x}_j} = \mathbf{W} \mathbf{d}_j^T,$$

where the  $i$ :th column  $\mathbf{d}_i$  of  $\mathbf{D}$  is defined as

$$\mathbf{d}_i = \frac{\mathbf{r}_i - \mathbf{x}_j}{\max \varepsilon, \|\mathbf{r}_i - \mathbf{x}_j\|}, i = 1, \dots, m.$$

**Remark 1.** It should be noted that it is not completely clear, especially in the context of FS, whether a distance-based feature map would benefit from a separate bias term. The theoretical basis behind EMLM does not require this [Remark 1] [4], and its omission has also been recommended for the ELM [89]. However, a separate bias is known to enforce an unbiased regression

estimate [90], and it could be included in the model simply by enlarging  $\mathbf{H}$  into  $\begin{bmatrix} \mathbf{H} \\ \mathbf{1} \end{bmatrix}$  where  $\mathbf{1}$  denotes the unit matrix of size  $\mathbb{R}^{1 \times N}$ . We conducted a brief experimental pursuit of the question—which has not been reported here—and concluded that a separate bias term added no value. Therefore, the use of the original formulation was confirmed.

**Remark 2.** MAS-formula (4) to quantify feature importance is independent of the model  $\mathcal{M}$ ; one only needs the model’s derivative with respect to features. Basically, this can be obtained using finite differences or automatic differentiation but here we confine ourselves to analytic formulae. A preliminary work with MAS and the analytic derivative of a feedforward neural network (FNN) model, with two transformation layers, was presented in [35]. In order to enable more extensive testing of the MAS-based wrapper approach, we include here the MAS formula for FNNs with any number of layers. The calculus is omitted because it can be performed similarly to [90]. For convenience and building from the previous remark, we assume that the training data has been scaled into  $[-1, 1]$ , the FNN does not contain bias nodes, and that tanh-functions  $\tanh(x) = \frac{2}{1 + \exp(-2x)} - 1$  are used as the activation functions throughout (note that the activation functions need to be differentiable which rules out the use of ReLU). Then, a layerwise formalism for the input–output mapping of any FNN with weights  $\{\mathbf{W}^l\}_{l=1}^L$  (i.e., weights of layers stored in matrices from the first layer  $\mathbf{W}^1$  up to the last layer  $\mathbf{W}^L$ ) can be represented using a diagonal function matrix  $\mathcal{F} = \mathcal{F}(\cdot) = \text{Diag}\{f_i(\cdot)\}_{i=1}^m$ , where  $f_i = \tanh$ , as follows

$$\mathbf{o} = \mathbf{o}^L = \mathcal{M}(\mathbf{x}) = \mathbf{W}^L \mathbf{o}^{(L-1)}, \tag{5}$$

where  $\mathbf{o}^0 = \mathbf{x}$  (a given input vector) and  $\mathbf{o}^l = \mathcal{F}^l = \mathcal{F} \mathbf{W}^l \mathbf{o}^{(l-1)}$  for  $l = 1, \dots, L - 1$ . The analytic derivative of such mapping with respect to the input features reads as

$$\frac{\partial \mathcal{M}}{\partial \mathbf{x}} = \mathbf{W}^L \prod_{l=L-1}^1 (\mathcal{F}^l)' \mathbf{W}^l. \tag{6}$$

---

#### Algorithm 1 Distance-based one-shot wrapper

---

**Input:** Input data  $\mathbf{x}_j \in \mathbb{R}^n | j = 1, \dots, N$ , target data

$$\{y_j \in \mathbb{R} | j = 1, \dots, N\}$$

**Output:** Indices of most important features

- 1: Train EMLM model using (2) with the full set of features
  - 2: Compute  $\mathcal{F}$  using (4)
  - 3: Sort  $\mathcal{F}$
  - 4: Using Kneedle, find kneepoint of sorted  $\mathcal{F}$  at feature index  $k$
  - 5: Keep features that satisfy  $\{i | \mathcal{F}_i \geq \mathcal{F}_k, 1 \leq i \leq n\}$
- 

### 3.3. Threshold selection

Once the features are ranked according to their score, there needs to be a way to decide how many of them are retained and on what basis. Because the scores, sorted according to their rank, define a 1D curve, the classical knee-point could be used to identify a change in the characteristic behavior [91]. A widely used technique for knee-point detection is to maximize the curvature, for which explicit formula is given in [92]. This is realized in a readily

implemented kneepoint detection algorithm, Kneedle [93], which identifies the cutoff point of a smoothed curvature  $\frac{f''(x)}{(1+f'(x)^2)^{1.5}}$ .

The proposed one-shot FS algorithm is detailed in Algorithm 1. Fig. 1 illustrates the use of a kneepoint and Kneedle for FS with the EMLM and the MAS formula (4). In the figure, the mean validation error and the standard deviation for it is shown using blue, while the MAS-values representing each number of features is shown in orange. The simulated dataset for the demonstration is defined in Section 4.1.

#### 4. Experimental setup

In this section, we detail our experimental design related to selected datasets, compared methods, and evaluation metrics. We also compare our proposal <sup>1</sup> with popular FS methods using a representative set of synthetic and benchmark datasets. We utilize the area under the receiver operating characteristic curve as an evaluation metric with the synthetic datasets and the root-mean-square error with the benchmark datasets.

##### 4.1. Datasets

Here, we present the datasets used in the experiments. The use of readily available benchmark datasets is augmented by the use of synthetic data, which is similar to [94].

**Synthetic.** We created a set of synthetic datasets to analyze the goodness of feature importance scoring. We can utilize ranking-based evaluation metrics when the ground truth features are available. Therefore, with the synthetic datasets, we can focus on the primary problem of FS independently from the thresholding of the feature importance score. We used two sets of synthetic datasets: one (denoted  $Y_{Rk}$ ) that has already been used in other studies [94] and a set inspired by the first (denoted  $Y_{Ax}$ ). The functions used to generate the synthetic datasets are presented in Table 2. The last row containing  $Y_{R4}$  consists of two equations forming a spiral equation. The datasets  $Y_{A1}$ ,  $Y_{A3}$ ,  $Y_{A4}$ ,  $Y_{A5}$ , and  $Y_{A6}$  have progressive complexity. The  $Y_{A2}$  dataset is the most challenging, since it mostly represents incoherent noise. However, it can show if a feature ranking algorithm will find results that are not practically there. Thus, it functions as a sanity check. For the  $Y_{A1}$ - $Y_{A6}$  datasets, half of the features are true ones, while for the  $Y_{R1}$ - $Y_{R4}$  datasets, there are only one to four true features. For each synthetic dataset, we generated 1000 observations with 200 features ( $N = 1000, n = 200$ ). For datasets  $Y_{A1}$ - $Y_{A6}$  and  $Y_{R1}$ - $Y_{R3}$ , the features were randomly generated as defined in

$$x_i = \mathcal{U}(0, 1) \quad i \in [0, \dots, n - 1]. \tag{7}$$

For dataset  $Y_{R4}$ ,  $Y = \mathcal{U}(0, 20)$ . After the target of a synthetic dataset has been calculated, the dataset target is augmented with Gaussian noise of zero mean and unit variance, which is augmented by normalizing it with the maximum difference to prevent egregious "measurement errors."

**Benchmark datasets.** A group of openly available datasets, also mentioned in the umbrella review in Section 2, were used to get comparable results. The benchmark datasets' characteristics are presented in Table 3, where the column headers #Obs., #Feat., #Trgt. refer to the numbers of observations, features and targets, respectively. Column #Un.Trgt. refers to the number of unique values found in the target vector, while header T. refers to the dataset type (regression R, classification C) and Src to the source of the dataset. We have used the first target when the dataset had more than one target.

<sup>1</sup> Source codes available at: <https://gitlab.jyu.fi/hnpai-public/extreme-minimal-learning-machine/>

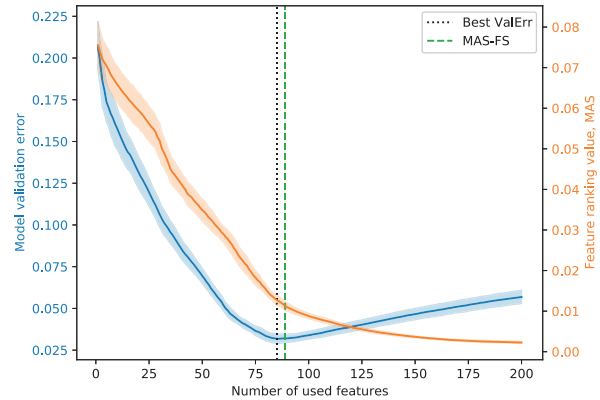


Fig. 1. Example of the cutoff point given by MAS kneepoint (in green) using the synthetic dataset  $Y_{A1}$  and the computed and sorted MAS values. The MAS-kneepoint provided the correct set of features that minimized the validation error.

Table 2  
Synthetic datasets. Column  $F_T$  represents the number of true features, while column  $F_F$  represents the number of false features.

Function	$F_T$	$F_F$
$Y_{A1} = \sum_{i=0}^{99} (99 - i)x_i + \mathcal{N}(0, 1)$	100	100
$Y_{A2} = \sum_{i=0}^{99} \sin(2\pi(99 - i)x_i) + \mathcal{N}(0, 1)$	100	100
$Y_{A3} = \sum_{i=0}^{99} (99 - i)x_i^2 + \mathcal{N}(0, 1)$	100	100
$Y_{A4} = \sum_{i=0}^{99} (99 - i)x_i^6 + \mathcal{N}(0, 1)$	100	100
$Y_{A5} = \sum_{i=0}^{99} (99 - i)e^{x_i} + \mathcal{N}(0, 1)$	100	100
$Y_{A6} = \sum_{i=0}^{99} (99 - i) \log(1 - x_i) + \mathcal{N}(0, 1)$	100	100
$Y_{R1} = -2 \sin(2x_0) + x_1^2 + x_2 + e^{x_3} + \mathcal{N}(0, 1)$	4	196
$Y_{R2} = x_0 e^{2x_1} + x_2^2 + \mathcal{N}(0, 1)$	3	197
$Y_{R3} = \sin(2\pi x_0) + \mathcal{N}(0, 0.1)$	1	199
$\begin{cases} x_0 = Y_{R4} \sin(Y_{R4}) + \mathcal{N}(0, 1) \\ x_1 = Y_{R4} \cos(Y_{R4}) + \mathcal{N}(0, 1) \end{cases}$	2	198

It should be noted that due to a lack of benchmark regression datasets for FS, we also used a set of classification datasets (see Table 3) in a separate experiment. The aim was to observe how our proposed FS algorithm would function with datasets for which it was not designed.

##### 4.2. Compared ranking and FS method

The quality of the MAS score was first compared to the most common ranking algorithms of filters and embedded methods.

Table 3  
Benchmark datasets.

Dataset	#Obs.	#Feat.	#Trgt.	#Un.Trgt.	T.	Src
StudentTest	258	5	1	4	C	[95]
ATP1D	337	411	6	83	R	[96]
COIL	1800	21	1	100	C	[95]
Madelon	2000	500	1	2	C	[95]
Outdoor	2400	21	1	40	C	[95]
ThyroidAnn	3772	21	1	3	C	[95]
OptDigits	3823	64	1	10	C	[95]
SatImage	4435	36	1	6	C	[95]
COIL2000	5822	85	1	2	C	[95]
RF2	7679	576	8	515	R	[96]
ComputerActivity	8192	21	1	56	R	[97]
SCM1D	9803	280	16	1092	R	[96]
Census	22784	8	1	2045	R	[97]



Here, we used *Rrelieff*, *SpearmanR*, *Mutual-Info*, *Fisher-score* and *Mean Absolute Difference* (MAD) as non-model-based comparisons. We also included model-based comparisons, namely *DT* and *RF*. Both model based ones were used only to gain a rank for each feature in the feature ranking experiment.

The quality of the distance-based regression model after feature ranking and one-shot selection was then compared to a selected group of well-known reference models, which have commonly been referenced in the literature. The selected group consisted of linear models, *Linear Regression*, *Ridge Regression*, and *Lasso* as well as tree-based models, *Decision Tree* and *Random Forest* and finally *SVR*. The methods are all readily available in the Python library *Scikit-Learn* [98].

### 4.3. Evaluation criteria

The quality of feature ranking and selection were assessed with multiple evaluation criteria, which are discussed next. The starting point for the evaluation is the existence of a separate validation dataset that can be used to assess FS performance and the resulting data-driven models. Moreover, since the specific true features of the synthetic datasets are known, we can directly count the number of true and false features. However, due to random number generation being involved in data generation, these counts may differ from the intended ground truth in rare occasions, which must be considered when looking at evaluation results. The used evaluation criteria are as follows:

**Root Mean Square Error** (RMSE) is a standard way to compute the validation error of a regression model [99]:

$$RMSE = \sqrt{\frac{\sum_{i=0}^p (y_i - \hat{y}_i)^2}{p}}$$

where  $P$  is the number of observations in the validation set,  $y_i$  is the predicted validation target, and  $y_i$  is the true validation target.

**Kruskal–Wallis** H test is a well-known statistical significance test between two data groups [100]. Because the test is non-parametric, data does not need to be from a normal distribution. We used significance level 0.05 for the H test. Any p-value lower than 0.05 indicates that the two tested groups have significant differences. In our results, we took the best achieved result and compared the other results to it in a pairwise manner.

According to Table 3 of the review by Solorio-Fernández et al. [14], there is no proper consensus on how to verify that a feature ranking algorithm works. Consequently, we have opted to use a couple of different measures for verification purposes. **Area under receiving operating characteristic** (AUROC) is another way to measure and quantify the quality of a feature ranking, when the true features are known for a dataset. Teisseyre [101] defined the receiving operating characteristic curve (ROC) as  $(FPR(k), TPR(k))$ , where  $k = 1, \dots, m$  refers to the top  $k$  selected features,  $FPR$  = false positive rate among the top  $k$  selected features and  $TPR$  = true positive rate among the top  $k$  selected features. When the ROC is paired with the area under curve, we get a single number describing the performance of a feature ranking algorithm. The measurement intuitively explains a performance. The more correctly the features are ranked, the closer the score is to 1. Scores close to 0.5 indicate random selection, while scores close to 0 mean that correct features are specifically not selected.

**Number of features for best validation error**,  $n_{sel}$ , is what the name implies. The number of features in the subset of features that provide the lowest validation error is taken, and the mean is calculated from the achieved numbers of features. This allows the observation of what portion of features are required by a feature ranking algorithm to reach its best-achieved result. Since the goal of FS is to remove as many features as possible, a smaller feature subset is preferable.

### 4.4. Feedforward Neural Networks

In addition to comparing *EMLM*-based *MAS-FS* to the popular feature selection methods, we also performed experiments with the MAS for FNNs (see Remark 2). We used two versions of Sequential model from Tensorflow [102]: A single hidden layer + linear output layer (denoted as *FNN-2*) and three hidden layers + linear output layer (denoted as *FNN-4*). Sizes of the Tensorflow's dense-layers were fixed to:

$$\begin{aligned} FNN_2 &= (\lceil n_s/2 \rceil, 1), \\ FNN_4 &= (\lceil n_s/2 \rceil, \lceil n_s/4 \rceil, \lceil n_s/8 \rceil, 1), \end{aligned}$$

where  $n_s$  is the number of features (after dataset preprocessing (removal of constant features)).

## 5. Experiments and results

In this section, we present the experiments and their results based on the datasets and evaluation criteria depicted in the previous section. The discussion follows the steps of the overall MAS-based FS algorithm (*MAS-FS*) for the distance-based *EMLM* as follows: feature ranking component, testing against commonly known algorithms, and finally testing against *RF*.

The experiments were run on a local computation cluster on a single node (2x Intel Xeon Gold 6148) using Python 3.8. We used the following library versions: *i*) NumPy: 1.20.1 [103]; *ii*) SciPy: 1.6.1 [104]; *iii*) Scikit-learn: 0.24.1 [98]; and *iv*) Knead: 0.7.0 [93].

### 5.1. Assessing the quality of feature scores and rankings

Our first step was to compare the MAS scoring with a representative set of other techniques depicted in Section 4.2. Using the synthetic datasets presented in Table 2, we compared the FS algorithms' ability to generate feature scores and the corresponding ranking to correctly order the features.

We randomly split each synthetic dataset 30 times into training and validation partitions, where 33% of the generated dataset composed the validation set. Each of the 30 training partitions were input to a feature ranking algorithm. The features were scored and ordered, which with the known ground truth allowed us to then use AUROC (presented in Section 4.3) to compute a quality measure.

In addition, we computed the validation error for each subset of the ordered features in each of the 30 splits. Specifically, we used *EMLM* (85% of data used as reference points) to compute the validation error  $E_{val}(1)$  for the rank #1 feature,  $E_{val}(2)$  for rank #1 and #2 features,  $\dots, E_{val}(n)$  for all features. Then, we found the number of features for  $\min(E_{val})$ . Finally, we reported the mean and standard deviation for each AUROC value and each  $\min(E_{val})$ . The results are shown in Table A.4.

The best values, the highest AUROC (denoted in table as AUC), and the lowest  $n_{sel}$ , are in bold for each dataset. In addition, we used  $\blacklozenge$  to show that a value is statistically close to the designated best value (using the Kruskal–Wallis H test).

We will discuss the results presented in Table A.4 in the order of feature ranking algorithms.

*MASvec* achieved the lowest number of features in seven cases out of 10 and the lowest mean validation errors in eight cases out of 10. With these results, it had the best performance among the tested feature ranking algorithms. Further, it is the only algorithm that did not select extra features for datasets A1–A6 from the false feature pool, as it did not return more than 100 features. However, it slightly missed the intended number of features with datasets R1, R2, and R4.

*RreliefF* is a traditional feature ranking algorithm that is often praised. However in our experiments, *RreliefF* failed to impress. Regarding datasets A1–A6 and R3, *RreliefF* performed as if it was a random selector. While the algorithm found better results with datasets R1, R2 and R4, overall its performance was found lacking. Thus, it can be concluded that the algorithm is not suited for datasets where the input features closely resemble each other.

*SpearmanR* performed best in datasets R1–R4 and relatively well in datasets A1–A6, excluding A2. This indicates that it is better suited to datasets where there are relatively few correct features.

*MI* is a popular ranking algorithm. In our experiments, it performed better than *RreliefF*, but was still surprisingly close to a random selector with datasets A1–A6. However, *MI* is one of the three tested algorithms that managed to find the correct number of features with dataset R4. Dataset R4 has spiral-like data, and we expect the form of the dataset to play a role in the performance of *MI*.

Based on the AUROC values, *DT* performed in a similar manner to *MI*: it was surprisingly close to random selection with datasets

A1–A6 and performed significantly better with datasets R1–R4. As *DT* is an iterative method, it may require more training time than what is provided by the default settings to be able to properly handle a situation with more than a few correct features.

*RF* is the third of the three that found the correct number of features with dataset R4, and it performed similar to *MI* and *DT*. We expect that *RF Ranker* has the same needs as *DT* and would require changes to the default settings.

*Fisher-score* was the worst-performing feature ranking algorithm that we tested. Unlike other methods that resembled random selection at their worst, it actively selected wrong features in the A1–A6 datasets. However, the same effect was not present with datasets R1–R4, even though *Fisher-score* was closer to random selection with datasets R1 and R2. Thus, we must conclude that *Fisher-score* is not suitable for datasets like A1–A6 and R1–R4.

*Mean Absolute Difference (MAD)* mostly resembled a random selector, with the exceptions being with datasets R1–R4. Especially of note is the result for dataset R4, since it completely failed

**Table A.4**

Feature-ranking algorithm results for the eight feature ranking algorithms and for the 10 synthetic datasets. The best mean value per dataset has been highlighted in bold. The results where the population median is the same as for the best mean value according to the Kruskal–Wallis H test are marked using  $\blacklozenge$ .

Algorithm		MASvec		RreliefF		SpearmanR		Mutual-Info	
Dataset	Var	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$
A1	AUC	<b>0.93</b>	0.01	0.50	0.05	0.80	0.02	0.57	0.05
	$n_{sel}$	<b>86.67</b>	3.57	198.00	2.21	141.13	18.67	169.63	30.47
A2	AUC	0.49	0.03	0.49	0.03	0.50 $\blacklozenge$	0.04	0.51 $\blacklozenge$	0.05
	$n_{sel}$	78.83 $\blacklozenge$	80.62	51.03 $\blacklozenge$	41.12	54.07 $\blacklozenge$	48.60	49.57 $\blacklozenge$	35.36
A3	AUC	<b>0.91</b>	0.01	0.50	0.04	0.79	0.01	0.58	0.04
	$n_{sel}$	<b>80.20</b>	3.96	198.57	1.94	131.53	20.87	142.77	20.22
A4	AUC	<b>0.87</b>	0.02	0.50	0.03	0.79	0.02	0.59	0.04
	$n_{sel}$	<b>65.53</b>	5.00	197.17	3.53	77.50	17.43	144.93	19.59
A5	AUC	<b>0.93</b>	0.01	0.50	0.04	0.81	0.02	0.55	0.04
	$n_{sel}$	<b>85.27</b>	3.93	197.87	2.80	145.37	27.68	167.93	27.99
A6	AUC	<b>0.86</b>	0.01	0.49	0.04	0.77	0.02	0.57	0.05
	$n_{sel}$	<b>74.87</b>	10.26	197.63	2.66	106.27	20.69	147.90	25.73
R1	AUC	<b>0.99</b>	0.00	0.80	0.10	0.99 $\blacklozenge$	0.00	0.97	0.03
	$n_{sel}$	4.93 $\blacklozenge$	1.06	98.50	57.89	5.00 $\blacklozenge$	0.97	23.00	21.29
R2	AUC	<b>0.99</b>	0.00	0.79	0.11	0.99 $\blacklozenge$	0.00	0.99	0.00
	$n_{sel}$	<b>3.13</b>	0.34	20.13	28.71	3.13 $\blacklozenge$	0.43	3.93	1.39
R3	AUC	<b>0.99</b>	0.00	0.44	0.27	0.99 $\blacklozenge$	0.00	0.99 $\blacklozenge$	0.00
	$n_{sel}$	<b>1.00</b>	0.00	69.80	52.54	1.00 $\blacklozenge$	0.00	1.00 $\blacklozenge$	0.00
R4	AUC	0.95	0.03	0.90	0.09	0.96	0.01	<b>0.99</b>	0.00
	$n_{sel}$	5.37	4.21	7.83	5.77	5.17 $\blacklozenge$	4.89	<b>2.00</b>	0.00
Algorithm		DecisionTree		RandomForest		Fisher-score		MAD	
Dataset	Var	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$	$\bar{x}$	$\delta$
A1	AUC	0.58	0.04	0.74	0.03	0.21	0.05	0.48	0.03
	$n_{sel}$	195.13	8.07	174.00	21.21	197.07	4.87	199.80	0.40
A2	AUC	0.51 $\blacklozenge$	0.04	<b>0.52</b>	0.03	0.19	0.04	0.50	0.02
	$n_{sel}$	62.00 $\blacklozenge$	49.82	61.93 $\blacklozenge$	55.80	57.60 $\blacklozenge$	47.92	<b>42.13</b>	27.41
A3	AUC	0.56	0.04	0.76	0.02	0.20	0.05	0.46	0.02
	$n_{sel}$	197.13	3.19	161.40	21.66	197.67	4.56	198.90	1.14
A4	AUC	0.57	0.03	0.76	0.02	0.22	0.08	0.52	0.02
	$n_{sel}$	195.37	5.38	80.80	18.67	194.53	7.01	197.40	5.09
A5	AUC	0.56	0.03	0.74	0.02	0.19	0.06	0.49	0.02
	$n_{sel}$	196.03	6.13	167.40	22.49	197.70	4.09	198.53	1.61
A6	AUC	0.57	0.04	0.74	0.03	0.20	0.04	0.50	0.02
	$n_{sel}$	195.67	6.19	134.33	34.96	196.30	4.38	197.37	4.59
R1	AUC	0.99 $\blacklozenge$	0.00	0.99 $\blacklozenge$	0.00	0.52	0.05	0.35	0.07
	$n_{sel}$	<b>4.87</b>	0.88	5.00 $\blacklozenge$	1.03	128.53	7.66	179.00	17.40
R2	AUC	0.99 $\blacklozenge$	0.00	0.99 $\blacklozenge$	0.00	0.58	0.07	0.40	0.10
	$n_{sel}$	3.30 $\blacklozenge$	0.64	3.27 $\blacklozenge$	0.51	126.33	16.63	132.53	40.06
R3	AUC	0.99 $\blacklozenge$	0.00	0.99 $\blacklozenge$	0.00	0.86	0.22	0.64	0.24
	$n_{sel}$	1.03 $\blacklozenge$	0.18	1.00 $\blacklozenge$	0.00	27.40	46.68	71.00	50.85
R4	AUC	0.99 $\blacklozenge$	0.00	0.99 $\blacklozenge$	0.00	0.66	0.10	0.00	0.00
	$n_{sel}$	2.00 $\blacklozenge$	0.00	2.00 $\blacklozenge$	0.00	11.00	32.60	52.83	37.17

to find the correct feature. We can conclude that *MAD* does not work well with a spiral-like dataset.

Based on the results, we can conclude that *MAS-FS* ranked first in the test. This confirms its basic utility to be used for feature ranking with the distance-based *EMLM*. Moreover, it should be noted that all ranking methods had problems with the  $Y_{A2}$  dataset. This was expected as it mostly resembles random noise and is the most difficult of the synthetic datasets.

### 5.2. Comparison of Algorithm 1 to reference algorithms with regression datasets

In order to assess the entire FS algorithm given in Algorithm 1, we compared it to other approaches using publicly available regression datasets, which are presented in Section 4.1. Similar to experiments in Section 5.1, each dataset had 30 different training/validation splits (with validation partition using 33% of the whole data). Because the number of reference points has a quadratic effect on the computational effort of *EMLM*, we provide results with two reference point percentages for the results presented in Appendix B, 65% and 85%, and additionally with 100% for the results presented in Appendix C. We were interested in observing how much accuracy might be lost with a reduced number of reference points.

Results of these experiments are given in Table B.5. *MAS-FS* was included with two reference point percentages. In almost all cases, the results between 65% and 85% were close to each other based on the Kruskal–Wallis test, the exception being *ComputerActivity* dataset, where the result for 65% was better than it was for 85%. This indicates that between the two, the higher reference point percentage did not have a meaningful effect on the outcome of the FS. This indicates that for the purposes of FS, *MAS-FS* is robust enough that a lower reference point percentage is recommended. This is also because the lower reference point percentage requires fewer computations. We point out that this conclusion is given in the context of FS: after selecting the final feature set, the portion of reference points for the corresponding *EMLM* model can be selected independently. *MAS-FS* had either the best RMSE value or was close to the best RMSE value based on the Kruskal–Wallis test in four datasets out of five, thus coming out on top of the tested FS methods. Altogether, *MAS-FS* was the best or statistically equally good as the best in 3/5 cases. *RF* had the best value or was statistically similar to the best value in two datasets out of five. *Lasso* did not receive the best RMSE values, but it is noteworthy that *Lasso* had the lowest standard deviation in the RMSE values in four datasets out of five, indicating that it was the most consistent of the tested FS methods. Of the tested methods, no other clear noteworthy results were gained.

### 5.3. Algorithm 1 with FNN on synthetic and regression datasets

Next we conclude the experiments where FNN and the corresponding feature sensitivity formula were used in Algorithm 1. This simply means that the *EMLM* and the  $\mathcal{F}\mathcal{S}$  formula in Steps 1 and 2 of Algorithm 1 were replaced with the FNNs as defined in Section 4.4 and the sensitivity formula given in (6). Each dataset had 30 different training/validation splits (with validation partition using 33% of the whole data). The results of these experiments are given in Tables C.6 and C.7. As can be seen from Table C.6 and Table C.7, the mean RMSE validation errors ( $\bar{e}$  in tables) indicates that the *EMLM*-based *MAS-FS* is able to achieve lower errors than either of the tested FNN version. Another noteworthy observation, although expected, is that the deeper model *FNN-4* achieves better accuracy than *FNN-2* in all cases. We can also point out that the result for *FNN-4* begins to approach the result for *EMLM* with *Census* dataset. From the data-driven model construc-

tion perspective, use of *FNN* and *EMLM* have significant differences. Whereas selection of the portion of reference points is sufficient for *EMLM*, with *FNN* one could tune the number of epochs and the size of batches per epoch in training, the number of hidden layers, the number of neurons in each layer, the activation function in each neuron etc. In addition, *EMLM* is fully deterministic but *FNN* is not and may require multiple training rounds in the hopes of improving the model. Thus, we assume that the *FNN* results could be improved by significantly increasing the amount of time used in hyperparameter optimization and assessing different models. However, these results show that the feature sensitivity based FS can be generalized to completely different models compared to the kernel-like *EMLM* and that the generalization capability of the *EMLM-MAS-FS* algorithm compared to FNN-based versions is promising.

### 5.4. Comparison of Algorithm 1 to reference algorithms with classification datasets

For our last experiment, we compared the *MAS-FS*-based FS to the available implementation of *RF* for a regression task. Similar to experiments in Section 5.1, each dataset had 30 different training/validation splits (with validation partition using 33% of the whole data). Some of the classification datasets came with their own validation dataset. For these, the provided validation dataset was first combined with the rest of the data and then split into train/validation sets as was done with the other datasets.

The results for the mean RMSE validation error are given in Table D.8. The table contains results for three different reference point percentages for *MAS-FS*, as *RF* does not have the same

**Table B.5**

Comparison of *MAS-FS* to reference ML models using regression datasets. Header  $\bar{e}$  denotes the mean validation error calculated with RMSE and header  $\delta$  its standard deviation. Best mean validation error per dataset has been marked with bold text. Results where the population median is the same as for the best mean value (Kruskal–Wallis H test) are marked with  $\blacklozenge$ .

Dataset	ATP1D			RF2		
	$\bar{e}$	$\delta$		$\bar{e}$	$\delta$	
Algorithm						
MAS-FS, 65%	9.42e	2	4.02e 3	7.70e	2 $\blacklozenge$	6.21e 3
MAS-FS, 85%	9.40e	2	4.40e 3	<b>7.68e</b>	<b>2</b>	6.17e 3
DecisionTree	8.65e	2	2.28e 3	1.16e	1	1.45e 2
Lasso	1.06e	1	1.00e 4	1.76e	1	3.62e 3
LinearRegression	5.13e2		2.76e3	4.39e	1	1.07e 1
RandomForest	<b>5.97e</b>	<b>2</b>	1.33e 3	8.10e	2	6.42e 3
Ridge Regression	8.55e	2	5.63e 4	8.80e	2	7.55e 3
SVM	7.67e	2	7.58e 4	9.25e	2	5.06e 3

Dataset	SCM1D			ComputerActivity		
	$\bar{e}$	$\delta$		$\bar{e}$	$\delta$	
Algorithm						
MAS-FS, 65%	3.32e	3 $\blacklozenge$	3.45e 4	2.69e	2	3.41e 3
MAS-FS, 85%	<b>3.29e</b>	<b>3</b>	3.51e 4	3.33e	2	1.28e 2
DecisionTree	1.12e	2	3.62e 3	3.74e	2	2.09e 3
Lasso	2.09e	1	1.16e 4	1.86e	1	3.80e 5
LinearRegression	1.79e	2	1.14e 3	9.80e	2	3.07e 3
RandomForest	7.84e	3	2.26e 3	<b>2.53e</b>	<b>2</b>	9.24e 4
Ridge Regression	2.27e	2	6.58e 4	9.80e	2	2.20e 3
SVM	5.29e	2	6.98e 4	4.76e	2	2.00e 3

Dataset	Census		
	$\bar{e}$	$\delta$	
Algorithm			
MAS-FS, 65%	3.70e	2 $\blacklozenge$	1.27e 3
MAS-FS, 85%	<b>3.64e</b>	<b>2</b>	1.35e 3
DecisionTree	5.64e	2	1.97e 3
Lasso	1.46e	1	1.07e 3
LinearRegression	5.05e	2	1.07e 3
RandomForest	4.00e	2	1.17e 3
Ridge Regression	4.94e	2	1.07e 3
SVM	5.20e	2	1.02e 3



parametrization, it has one result per dataset. Additionally for MAS-FS, we show the remaining number of features after FS as a percentage as well as the standard deviation for it. Of the eight classification datasets, MAS-FS has the best RMSE error in five cases. In general, the three different reference point percentages for MAS-FS produced similar mean validation errors, leading to

the same conclusion as was made with MAS-FS in Section 5.2. The dataset properties as well as the number of observations, features, and unique features do not provide indications on whether there is a pattern to the observed differences in terms of the mean validation errors between MAS-FS and RF. The type of the input (integer/float) did not provide any insight either. Thus,  $\bar{n}_{\%}$  was added

**Table C.6**

Comparison of FNN-based ranking to EMLM-based ranking using regression datasets. Header  $\bar{e}$  denotes the mean validation error calculated with RMSE and header  $\delta$  its standard deviation. Best mean value per dataset has been marked with bold text. Results where the population median is the same as for the best mean value (Kruskal–Wallis H test) are marked with  $\blacklozenge$ .

Dataset	EMLM 85%			FNN-2			FNN-4		
	$\bar{e}$	$\delta$		$\bar{e}$	$\delta$		$\bar{e}$	$\delta$	
A1	<b>4.51e</b> 2	1.20e 2		1.91e 1	2.82e 2		1.10e 1	1.52e 2	
A2	<b>8.48e</b> 2	2.84e 3		1.84e 1	2.16e 2		1.04e 1	1.31e 2	
A3	<b>4.14e</b> 2	8.96e 3		1.84e 1	2.06e 2		1.05e 1	1.52e 2	
A4	<b>5.75e</b> 2	4.49e 3		1.84e 1	2.45e 2		1.08e 1	1.64e 2	
A5	<b>4.52e</b> 2	1.07e 2		1.91e 1	3.20e 2		1.14e 1	1.41e 2	
A6	<b>5.57e</b> 2	7.14e 3		1.93e 1	2.61e 2		1.07e 1	1.43e 2	
R1	<b>7.42e</b> 2	6.45e 3		1.93e 1	2.86e 2		1.06e 1	1.16e 2	
R2	<b>6.63e</b> 2	8.77e 3		1.83e 1	2.33e 2		1.07e 1	1.26e 2	
R3	<b>1.59e</b> 1	3.69e 3		2.26e 1	1.94e 2		1.67e 1	9.23e 3	
R4	<b>1.53e</b> 1	3.03e 3		2.20e 1	1.99e 2		1.65e 1	9.55e 3	

**Table C.7**

Comparison of FNN-based ranking to EMLM-based ranking using regression datasets. Header  $\bar{e}$  denotes the mean validation error calculated with RMSE and header  $\delta$  its standard deviation. Best mean value per dataset has been marked with bold text. Results where the population median is the same as for the best mean value (Kruskal–Wallis H test) are marked with  $\blacklozenge$ .

Dataset	EMLM 85%			FNN-2			FNN-4		
	$\bar{e}$	$\delta$		$\bar{e}$	$\delta$		$\bar{e}$	$\delta$	
ATP1D	<b>3.85e</b> 2	2.98e 3		2.31e 1	8.65e 2		1.52e 1	6.78e 2	
RF2	<b>1.71e</b> 3	2.34e 4		5.06e 2	1.50e 2		2.38e 2	4.32e 3	
SCM1D	<b>1.78e</b> 2	6.22e 4		3.35e 2	3.65e 3		2.95e 2	1.68e 3	
ComputerActivity	<b>1.45e</b> 2	3.78e 3		1.49e 1	3.84e 2		8.99e 2	2.31e 2	
Census	<b>3.56e</b> 2	4.64e 3		5.16e 2	4.81e 3		4.86e 2	4.35e 3	

**Table D.8**

Results of the comparison between MAS-FS and RF. The same notations from the tables above are used with the addition of  $\bar{n}_{\%}$  (percentage of features remaining after FS) and  $\delta n_{\%}$  (standard deviation for  $\bar{n}_{\%}$ ).

Dataset	Algorithm	RefP	MAS-FS				RandomForest	
			$\bar{e}$	$\delta$	$\bar{n}_{\%}$	$\delta n_{\%}$	$\bar{e}$	$\delta$
StudentTest	65	65	6.55e 2 $\blacklozenge$	8.74e 3	47%	9%	7.19e 2	7.44e 3
	85	85	6.56e 2 $\blacklozenge$	8.31e 3	43%	7%	-	-
	100	100	<b>6.48e</b> 2	7.31e 3	41%	5%	-	-
COIL	65	65	7.67e 2	3.10e 3	57%	10%	<b>7.30e</b> 2	3.23e 3
	85	85	7.53e 2	3.37e 3	59%	13%	-	-
	100	100	7.46e 2	3.22e 3	57%	10%	-	-
Madelon	65	65	5.18e 1	4.81e 3	88%	5%	<b>3.94e</b> 1	5.78e 3
	85	85	5.18e 1	5.18e 3	89%	6%	-	-
	100	100	5.19e 1	5.32e 3	87%	6%	-	-
Outdoor	65	65	9.52e 2	8.24e 3	66%	17%	1.06e 1	4.97e 3
	85	85	<b>8.84e</b> 2	8.59e 3	70%	18%	-	-
	100	100	8.90e 2	7.70e 3	65%	20%	-	-
OptDigits	65	65	8.10e 2	1.74e 3	91%	5%	1.05e 1	3.93e 3
	85	85	7.93e 2 $\blacklozenge$	1.89e 3	91%	5%	-	-
	100	100	<b>7.88e</b> 2	1.82e 3	91%	5%	-	-
ThyroidAnn	65	65	1.00e 1	4.69e 3	10%	0%	<b>3.93e</b> 2	6.25e 3
	85	85	1.00e 1	4.68e 3	10%	0%	-	-
	100	100	1.00e 1	4.67e 3	10%	0%	-	-
SatImage	65	65	1.04e 1 $\blacklozenge$	3.10e 3	70%	20%	1.14e 1	3.71e 3
	85	85	1.03e 1 $\blacklozenge$	3.57e 3	70%	19%	-	-
	100	100	<b>1.03e</b> 1	4.00e 3	71%	18%	-	-
COIL2000	65	65	<b>2.58e</b> 1	4.41e 3	83%	26%	2.61e 1	3.32e 3
	85	85	2.65e 1	6.28e 3	77%	31%	-	-
	100	100	2.71e 1	4.83e 3	82%	27%	-	-

in a bid to provide an explanation for the results, but a clear correlation was not found. However, as some of the results indicate (see *Madelon*), the kneepoint detection algorithm *Kneedle* can behave conservatively, leaving a large set of features to the obtained model. This suggests that, in some cases, it could be beneficial to repeat the algorithm to the once-reduced feature set. On the other hand, this works to the strengths of the distance-based *EMLM* since it is robust and is capable of handling extra features without a loss of accuracy.

### 5.5. Summary of the experimental results

Here we discuss the experimental results as a whole. Overall, our proposed FS algorithm performed well. On a more specific note, the feature ranking component in *MAS-FS* was able to determine the feature importance rather accurately, which then allowed the kneepoint detection algorithm to perform the actual FS. In the extensive experimental comparison, it was shown that the proposed method was better than the *RF* with both regression and classification datasets. Moreover, *MAS-FS* with *EMLM* can determine the scores and rankings of features for both the original, full set of features as well as the final, selected feature subset.

Of the synthetic datasets,  $Y_{A2}$  was the most problematic for all tested feature ranking algorithms, which we expected due to how the dataset is formed. Further, the AUROC-score revealed that except for the *Fisher-score*, the features in  $Y_{A2}$  were basically selected randomly. For the  $Y_{A1}$ – $Y_{A6}$  datasets, the AUROC values for the *Fisher-score* are below 0.25, implying that reversing the feature ranking would improve performance.

We included two versions of our algorithm for the regression dataset tests, for which we used two different reference point percentages and three versions for the classification dataset tests. Based on the results, that show that the performance between the reference point percentages was so similar, we recommend using 65% for feature ranking and selection as it is computationally lighter than 85%. After the ranking process, we recommend selecting as high a reference point percentage as possible due to the tendency of *EMLM* to not overlearn [4]. The comparison between *MAS-FS* using *EMLM* vs. *FNN* provided the knowledge that using our *MAS-FS* algorithm performs better with *EMLM* at its core at similar levels of researcher setup.

## 6. Conclusions

In FS, filters are used due to their speed and simplicity even if they often do not possess the best possible accuracy. Meanwhile, wrappers are used for their accuracy, but they require a search component that makes them slow and computationally expensive. A common practice is to combine the two by first applying a filter to quickly reduce the workload and then finishing with a wrapper. Our FS algorithm is a wrapper since it uses the distance-based model of *EMLM*, but it is a wrapper without a search component. This makes our algorithm simple, straightforward, and efficient. Since there is no search component, there is no iterative component either, implying that the feature importance scoring is conducted using a one-shot procedure.

We discovered that regression benchmark datasets for FS (especially with the ground truth features) are rarely available in the literature. Therefore, we presented a group of synthetic datasets ( $Y_{A1}$ – $Y_{A6}$ ), which were designed to have easily understandable relations between the feature importances. This allows them to function as a sanity check for a FS algorithm and as an assurance that the algorithm works properly. Moreover, the availability of ground truth features allows for the usage of feature ranking based performance measures. Indeed, current literature seldom discusses

FS in the regression context and has not discussed it in relation to distance-based ML models. We have positioned our umbrella review to provide a thorough background into the topic of this paper.

This paper proposed a new FS approach for a distance-based supervised machine learning model referred to as the *EMLM*. Subsequently, we evaluated the proposed method with an extensive set of synthetic and real datasets and compared it to popular approaches. In addition, we presented a thorough umbrella review, which is the first, on the topic of FS.

Our experimental results for a representative set of synthetic datasets showed that the regression model sensitivity-based feature importance scoring outperformed other methods in terms of feature ranking quality. Further, the proposed method can identify underlying non-linear, input–output data relations hidden in a large set of noisy features. The experimental results for the real datasets also showed that the proposed one-shot wrapper approach, which straightforwardly utilizes the model's sensitivity-based feature ranking outperformed (although with a slight margin) the popular methods like the *RF*.

In order to adapt the proposed FS method to other machine learning models, we derived a general *MAS-FS* formula for those *FNN* architectures which are differentiable with respect to features. We performed an experimental comparison with two off-the-shelf DL architectures, which demonstrated the adaptability of the proposed FS approach. However, the experimental results showed that the distance-based method with the one-shot wrapper outperformed these DL architectures. These results indicate that the DL architectures require more fine-tuning of parameters and data to obtain the same level of accuracy as this distance-based method.

As for future work, a natural extension of our study would be the application of the FS techniques to reduce features from the distance regression model in the first phase of the *MLM* [1,79]. Similarly, the encouraging initial assessment of *MAS*-based feature scoring and ranking for classification tasks, as given in [86], is to be extended to the full FS framework along the lines of this article. This is a prime example of a multi-output problem [105] where use of the *MAS* technique can produce individual sensitivities for each output variable. This would then allow for the usage of dedicated and different feature subsets for each output.

### CRedit authorship contribution statement

**Joakim Linja:** Methodology, Software, Formal analysis, Investigation, Data curation, Visualization, Validation, Writing – original draft, Writing – review & editing. **Joonas Hämäläinen:** Software, Supervision, Validation, Writing – original draft, Writing – review & editing. **Paavo Nieminen:** Supervision, Validation, Writing – original draft, Writing – review & editing. **Tommi Kärkkäinen:** Software, Conceptualization, Methodology, Supervision, Project administration, Funding acquisition, Validation, Writing – original draft, Writing – review & editing.

### Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### Acknowledgements

This work has been supported by the Academy of Finland through the projects 315550 (HNP-AI) and 351579 (MLNovCat). We acknowledge grants of computer capacity from the Finnish

Grid and Cloud Infrastructure (FCCI; persistent identifier urn:nbn:fi:research-infras-2016072533).

## Appendix A. Comparison of feature ranking algorithms

See Table A.4.

## Appendix B. MAS-FS comparison to reference methods

See Table B.5.

## Appendix C. EMLM-MAS-FS comparison to FNN-MAS-FS

See Tables C.6 and C.7.

## Appendix D. Tests with classification datasets

See Table D.8.

## References

- [1] A.H. de Souza Junior, F. Corona, G.A. Barreto, Y. Miche, A. Lendasse, Minimal Learning Machine: A novel supervised distance-based approach for regression and classification, *Neurocomputing* 164 (2015) 34–44.
- [2] D.P.P. Mesquita, J.P.P. Gomes, A.H. de Souza Junior, Ensemble of efficient minimal learning machines for classification and regression, *Neural Process. Lett.* 46 (3) (2017) 751–766.
- [3] D.P.P. Mesquita, J.P.P. Gomes, A.H. de Souza Junior, J.S. Nobre, Euclidean distance estimation in incomplete datasets, *Neurocomputing* 248 (2017) 11–18.
- [4] T. Kärkkäinen, Extreme minimal learning machine: Ridge regression with distance-based basis, *Neurocomputing* 342 (2019) 33–48.
- [5] E. Pekalska, R.P. Duin, Automatic pattern recognition by similarity representations, *Electron. Lett.* 37 (3) (2001) 159–160.
- [6] Y. Chen, Strategies for similarity-based learning, Ph.D. thesis, University of Washington, Program of Electrical Engineering (2010).
- [7] M.J.D. Powell, Radial basis function for multivariable interpolation: a review, in: *Algorithms for Approximation*, Clarendon Press, Oxford, 1987, pp. 143–167.
- [8] D.S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, *Complex Syst.* 2 (1988) 321–355.
- [9] T. Poggio, F. Girosi, Networks for approximation and learning, *Proc. IEEE* 78 (9) (1990) 1481–1497.
- [10] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, *Neural Comput.* 3 (2) (1991) 246–257.
- [11] Y. Lu, Z. Lai, Y. Xu, X. Li, D. Zhang, C. Yuan, Low-rank preserving projections, *IEEE Trans. Cybern.* 46 (8) (2016) 1900–1913, <https://doi.org/10.1109/TCYB.2015.2457611>.
- [12] Y. Zhai, Y.-S. Ong, I.W. Tsang, The emerging big dimensionality, *IEEE Comput. Intell. Mag.* 9 (3) (2014) 14–26, <https://doi.org/10.1109/MCI.2014.2326099>.
- [13] C.K. Fisher, P. Mehta, Bayesian feature selection for high-dimensional linear regression via the l<sub>1</sub> approximation with applications to genomics, *Bioinformatics* 31 (11) (2015) 1754–1761, <https://doi.org/10.1093/bioinformatics/btv037>.
- [14] S. Solorio-Fernández, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A review of unsupervised feature selection methods, *Artif. Intell. Rev.* 53 (2020) 907–948, <https://doi.org/10.1007/s10462-019-09682-y>.
- [15] H. Liu, H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer, Norwell, MA, 1998.
- [16] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, *J. Mach. Learn. Res.* 3 (2003) 1157–1182.
- [17] G.H. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem, in: *Proceedings of the 11th International Conference on Machine Learning*, 1994, pp. 121–129.
- [18] R. Kohavi, G.H. John, Wrappers for feature subset selection, *Artif. Intell.* 97 (1997) 273–324.
- [19] H. Zare, M. Niazi, Relevant based structure learning for feature selection, *Eng. Appl. Artif. Intell.* 55 (2016) 93–102.
- [20] X. Wu, X. Xu, J. Liu, H. Wang, B. Hu, F. Nie, Supervised feature selection with orthogonal regression and feature weighting, *IEEE Transactions on Neural Networks and Learning Systems*.
- [21] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, *IEEE Trans. Knowl. Data Eng.* 17 (4) (2005) 491–502.
- [22] Z. Xu, I. King, M.R.-T. Lyu, R. Jin, Discriminative semi-supervised feature selection via manifold regularization, *IEEE Trans. Neural Networks* 21 (7) (2010) 1033–1047.
- [23] K. Benabdeslem, M. Hindawi, Efficient semi-supervised feature selection: Constraint, relevance, and redundancy, *IEEE Trans. Knowl. Data Eng.* 26 (5) (2014) 1131–1143, <https://doi.org/10.1109/TKDE.2013.86>.
- [24] X. Zhang, Q. Zhang, M. Chen, Y. Sun, X. Qin, H. Li, A two-stage feature selection and intelligent fault diagnosis method for rotating machinery using hybrid filter and wrapper method, *Neurocomputing* 275 (2018) 2426–2439, <https://doi.org/10.1016/j.neucom.2017.11.016>.
- [25] A.L. Blum, P. Langley, Selection of relevant features and examples in machine learning, *Artif. Intell.* 97 (1997) 245–271.
- [26] J.-X. Peng, S. Ferguson, K. Rafferty, P.D. Kelly, An efficient feature selection method for mobile devices with application to activity recognition, *Neurocomputing* 74 (2011) 3543–3552.
- [27] J.R. Quinlan, Induction of decision trees, *Machine Learning* 1 (1) (1986) 81–106.
- [28] L. Breiman, Random forests, *Machine Learning* 45 (1) (2001) 5–32.
- [29] R. Genuer, J.-M. Poggi, C. Tuleau-Malot, Variable selection using random forests, *Pattern Recogn. Lett.* 31 (14) (2010) 2225–2236.
- [30] M. Wojtas, K. Chen, Feature importance ranking for deep learning, in: *Advances in Neural Information Processing Systems (NeurIPS 2020)*, Vol. 33, 2020, pp. 5105–5114.
- [31] A. Adadi, M. Berrada, Peeking inside the black-box: a survey on explainable artificial intelligence, *XAI, IEEE access* 6 (2018) 52138–52160.
- [32] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al., Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, *Information Fusion* 58 (2020) 82–115.
- [33] N. Burkart, M.F. Huber, A survey on the explainability of supervised machine learning, *Journal of Artificial Intelligence Research* 70 (2021) 245–317.
- [34] Y. Dimopoulos, P. Bourret, S. Lek, Use of some sensitivity criteria for choosing networks with good generalization ability, *Neural Process. Lett.* 2 (6) (1995) 1–4.
- [35] T. Kärkkäinen, Assessment of feature saliency of MLP using analytic sensitivity, in: *European symposium on artificial neural networks, computational intelligence and machine learning-ESANN2015*. Presses universitaires de Louvain, 2015, pp. 273–278.
- [36] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, *arXiv preprint arXiv:1312.6034*.
- [37] J. Ding, V. Tarokh, Y. Yang, Model selection techniques: An overview, *IEEE Signal Process. Mag.* 35 (6) (2018) 16–34.
- [38] M. Dash, H. Liu, Feature selection for classification, *Intelligent data analysis* 1 (1–4) (1997) 131–156.
- [39] R. Kohavi, G.H. John, Wrappers for feature subset selection, *Artificial intelligence* 97 (1–2) (1997) 273–324.
- [40] M.J. Grant, A. Booth, A typology of reviews: an analysis of 14 review types and associated methodologies, *Health information & libraries journal* 26 (2) (2009) 91–108.
- [41] M. Kilpala, T. Kärkkäinen, T. Hämäläinen, *Differential Privacy: An Umbrella review*, Springer Nature (2021) 1–20.
- [42] J. Egger, A. Pepe, C. Gsaxner, Y. Jin, J. Li, R. Kern, Deep learning—a first meta-survey of selected reviews across scientific disciplines, their commonalities, challenges and research impact, *PeerJ Computer Science* 7 (2021).
- [43] G. Chandrashekar, F. Sahin, A survey on feature selection methods, *Computers & Electrical Engineering* 40 (1) (2014) 16–28.
- [44] V. Bolón-Canedo, N. Sánchez-Marroño, A. Alonso-Betanzos, A review of feature selection methods on synthetic data, *Knowl. Inf. Syst.* 34 (3) (2013) 483–519.
- [45] J. Miao, L. Niu, A survey on feature selection, *Procedia Computer Science* 91 (2016) 919–926.
- [46] J. Li, K. Cheng, S. Wang, F. Morstatter, R.P. Trevino, J. Tang, H. Liu, Feature selection: A data perspective, *ACM Computing Surveys (CSUR)* 50 (6) (2018) 94.
- [47] Y. Li, T. Li, H. Liu, Recent advances in feature selection and its applications, *Knowl. Inf. Syst.* 53 (3) (2017) 551–577.
- [48] N. Spolaör, M.C. Monard, G. Tsoumakas, H.D. Lee, A systematic review of multi-label feature selection and a new method based on label construction, *Neurocomputing* 180 (2016) 3–15.
- [49] J. Cai, J. Luo, S. Wang, S. Yang, Feature selection in machine learning: A new perspective, *Neurocomputing* 300 (2018) 70–79.
- [50] J.C. Ang, A. Mirzal, H. Haron, H.N.A. Hamed, Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection, *IEEE/ACM Trans. Comput. Biol. Bioinf.* 13 (5) (2016) 971–989.
- [51] V. Bolón-Canedo, A. Alonso-Betanzos, Ensembles for feature selection: A review and future trends, *Information Fusion* 52 (2019) 1–12.
- [52] K. Yu, X. Guo, L. Liu, J. Li, H. Wang, Z. Ling, X. Wu, Causality-based feature selection: Methods and evaluations, *ACM Computing Surveys (CSUR)* 53 (5) (2020) 1–36.
- [53] P. Dhal, C. Azad, A comprehensive survey on feature selection in the various fields of machine learning, *Applied Intelligence* (2021) 1–39.
- [54] J.R. Vergara, P.A. Estévez, A review of feature selection methods based on mutual information, *Neural Comput. Appl.* 24 (1) (2014) 175–186.
- [55] N. Dessì, B. Pes, Similarity of feature selection methods: An empirical study across data intensive classification tasks, *Expert Syst. Appl.* 42 (10) (2015) 4632–4642.
- [56] J. Gui, Z. Sun, S. Ji, D. Tao, T. Tan, Feature selection based on structured sparsity: A comprehensive study, *IEEE Transactions on Neural Networks and Learning Systems* 28 (7) (2017) 1490–1507.
- [57] R.J. Urbanowicz, M. Meeker, W. La Cava, R.S. Olson, J.H. Moore, Relief-based feature selection: Introduction and review, *J. Biomed. Inform.* 85 (2018) 189–203.

- [58] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, M. Lang, Benchmark for filter methods for feature selection in high-dimensional classification data, *Computational Statistics & Data Analysis* 143 (2020).
- [59] J. Wang, P. Zhao, S.C. Hoi, R. Jin, Online feature selection and its applications, *IEEE Trans. Knowl. Data Eng.* 26 (3) (2014) 698–710.
- [60] X. Hu, P. Zhou, P. Li, J. Wang, X. Wu, A survey on online feature selection with streaming features, *Frontiers of Computer Science* 12 (3) (2018) 479–493.
- [61] R. Diao, Q. Shen, Nature inspired feature selection meta-heuristics, *Artif. Intell. Rev.* 44 (3) (2015) 311–340.
- [62] M. Sharma, P. Kaur, A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem, *Archives of Computational Methods in Engineering* 28 (3).
- [63] Q. Al-Tashi, H.M. Rais, S.J. Abdulkadir, S. Mirjalili, H. Alhussian, A review of grey wolf optimizer-based feature selection methods for classification, *Evolutionary Machine Learning Techniques* (2020) 273–286.
- [64] M. Mafarja, A.A. Heidari, H. Faris, S. Mirjalili, I. Aljarah, Dragonfly algorithm: theory, literature review, and application in feature selection, *Nature-Inspired Optimizers* (2020) 47–67.
- [65] B.H. Nguyen, B. Xue, M. Zhang, A survey on swarm intelligence approaches to feature selection in data mining, *Swarm and Evolutionary Computation* 54 (2020).
- [66] Q. Al-Tashi, S.J. Abdulkadir, H.M. Rais, S. Mirjalili, H. Alhussian, Approaches to multi-objective feature selection: A systematic literature review, *IEEE Access* 8 (2020) 125076–125096.
- [67] P.Y. Lee, W.P. Loh, J.F. Chin, Feature selection in multimedia: The state-of-the-art review, *Image Vis. Comput.* 67 (2017) 29–42.
- [68] S. Salcedo-Sanz, L. Cornejo-Bueno, L. Prieto, D. Paredes, R. García-Herrera, Feature selection in machine learning prediction systems for renewable energy applications, *Renew. Sustain. Energy Rev.* 90 (2018) 728–741.
- [69] K. Tadišt, S. Najah, N.S. Nikolov, F. Mrabti, A. Zahi, Feature selection methods and genomic big data: a systematic review, *Journal of Big Data* 6 (1) (2019) 1–24.
- [70] D. Pavlyuk, Feature selection and extraction in spatiotemporal traffic forecasting: a systematic literature review, *European Transport Research Review* 11 (1) (2019) 1–19.
- [71] X. Deng, Y. Li, J. Weng, J. Zhang, Feature selection for text classification: A review, *Multimedia Tools & Applications* 78 (3).
- [72] V. Bolón-Canedo, B. Remeseiro, Feature selection in image analysis: a survey, *Artif. Intell. Rev.* 53 (4) (2020) 2905–2931.
- [73] S. Kashef, H. Nezamabadi-pour, B. Nikpour, Multilabel feature selection: A comprehensive review and guiding experiments, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8 (2) (2018).
- [74] R.B. Pereira, A. Plastino, B. Zadrozny, L.H. Merschmann, Categorizing feature selection methods for multi-label classification, *Artif. Intell. Rev.* 49 (1) (2018) 57–78.
- [75] P. Raatikainen, J. Hautala, O. Loberg, T. Kärkkäinen, P. Leppänen, P. Nieminen, Detection of developmental dyslexia with machine learning using eye movement data, *Array* 12 (2021).
- [76] M. Cherrington, F. Thabtah, J. Lu, Q. Xu, Feature selection: filter methods performance challenges, in: *2019 International Conference on Computer and Information Sciences (ICIS)*, IEEE, 2019, pp. 1–4.
- [77] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, in: *2009 IEEE Symposium on Computational Intelligence and Data Mining, IEEE 2009* (2009) 389–395.
- [78] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: Theory and applications, *Neurocomputing* 70 (1) (2006) 489–501, <https://doi.org/10.1016/j.neucom.2005.12.126>.
- [79] J. Hämäläinen, A.S.C. Alencar, T. Kärkkäinen, C.L.C. Mattos, A.H. Souza Júnior, J. P.P. Gomes, Minimal Learning Machine: Theoretical results and clustering-based reference point selection, *Journal of Machine Learning Research* 21 (2020) 1–29.
- [80] T.F. Gonzalez, Clustering to minimize the maximum intercluster distance, *Theoret. Comput. Sci.* 38 (1985) 293–306.
- [81] D.J. Rosenkrantz, R.E. Stearns, P.M. Lewis II, An analysis of several heuristics for the traveling salesman problem, *SIAM J. Comput.* 6 (3) (1977) 563–581.
- [82] J. Linja, J. Hämäläinen, P. Nieminen, T. Kärkkäinen, Do randomized algorithms improve the efficiency of minimal learning machine?, *Machine Learning and Knowledge Extraction* 2 (4) (2020) 533–557, <https://doi.org/10.3390/make2040029>.
- [83] A. Pihlajamäki, J. Hämäläinen, J. Linja, P. Nieminen, S. Malola, T. Kärkkäinen, H. Häkkinen, Monte carlo simulations of  $au_{38}(sch)_{24}$  nanocluster using distance-based machine learning methods, *The Journal of Physical Chemistry A* 124 (23) (2020) 4827–4836, <https://doi.org/10.1021/acs.jpca.0c01512>.
- [84] T. Kärkkäinen, Model selection for extreme minimal learning machine using sampling, in: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN, 2019*, pp. 391–396.
- [85] J.E. Dennis Jr, R.B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, vol. 16, SIAM, 1996.
- [86] T. Kärkkäinen, On the role of Taylor's formula in machine learning, *Springer Nature*, 2022, Ch. Impact of scientific computing on science and society, (18 pages, to appear).
- [87] P.J. Huber, *Robust statistics*, vol. 523, John Wiley & Sons, 2004.
- [88] T. Kärkkäinen, S. Äyrämö, On computation of spatial median for robust data mining, in: *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems, EUROGEN, Munich, 2005*, p. 14.
- [89] G.-B. Huang, What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle, *Cognitive Computation* 7 (3) (2015) 263–278.
- [90] T. Kärkkäinen, MLP in layer-wise form with applications to weight decay, *Neural Comput.* 14 (6) (2002) 1451–1480.
- [91] R.L. Thorndike, Who belongs in the family, *Psychometrika* 18 (4) (1953) 267–276.
- [92] R.C. Yates, *A Handbook on Curves and their Properties*, JW Edwards, 1947.
- [93] V. Satopaa, J. Albrecht, D. Irwin, B. Raghavan, Finding a kneedle in a haystack: Detecting knee points in system behavior, in: *2011 31st International Conference on Distributed Computing Systems Workshops, 2011*, pp. 166–171, doi:10.1109/ICDCSW.2011.20.
- [94] Y. Sun, J. Yao, S. Goodison, Feature Selection for Nonlinear Regression and its Application to Cancer Research, 2015, pp. 73–81, arXiv:<https://pubs.siam.org/doi/pdf/10.1137/1.9781611974010.9>, doi:10.1137/1.9781611974010.9, URL:<https://pubs.siam.org/doi/abs/10.1137/1.9781611974010.9>.
- [95] D. Dua, C. Graff, UCI machine learning repository (2017). URL:<http://archive.ics.uci.edu/ml>.
- [96] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, I. Vlahavas, Multi-target regression via input space expansion: treating targets as inputs, *Machine Learning* 104 (1) (2016) 55–98, <https://doi.org/10.1007/s10994-016-5546-z>.
- [97] University of Toronto, Delve datasets (1996). URL:<http://www.cs.toronto.edu/delve/data/datasets.html>.
- [98] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* 12 (2011) 2825–2830.
- [99] A.G. Barnston, Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score, *Weather and Forecasting* 7 (4) (1992) 699–709, [https://doi.org/10.1175/1520-0434\(1992\)007<0699:CATCRA>2.0.CO;2](https://doi.org/10.1175/1520-0434(1992)007<0699:CATCRA>2.0.CO;2), URL: [https://journals.ametsoc.org/view/journals/wefo/7/4/1520-0434\\_1992\\_007\\_0699\\_catcra\\_2\\_0\\_co\\_2.xml](https://journals.ametsoc.org/view/journals/wefo/7/4/1520-0434_1992_007_0699_catcra_2_0_co_2.xml).
- [100] W.H. Kruskal, A nonparametric test for the several sample problem, *Ann. Math. Stat.* 23 (4) (1952) 525–540, URL: <http://www.jstor.org/stable/2236578>.
- [101] P. Teisseyre, Feature ranking for multi-label classification using Markov networks, *Neurocomputing* 205 (2016) 439–454, <https://doi.org/10.1016/j.neucom.2016.04.023>.
- [102] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015). URL: <https://www.tensorflow.org/>.
- [103] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. van Kerkwijk, M. Brett, A. Haldane, J.F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, *Nature* 585 (7825) (2020) 357–362, <https://doi.org/10.1038/s41586-020-2649-2>.
- [104] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S.J. van der Walt, M. Brett, J. Wilson, K.J. Millman, N. Mayorov, A.R.J. Nelson, E. Jones, R. Kern, E. Larson, C.J. Carey, I. Polat, Y. Feng, E.W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E.A. Quintero, C.R. Harris, A.M. Archibald, A.H. Ribeiro, F. Pedregosa, P. van Mulbregt, SciPy 1.0 Contributors, SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, *Nat. Methods* 17 (2020) 261–272, <https://doi.org/10.1038/s41592-019-0686-2>.
- [105] J. Hämäläinen, T. Kärkkäinen, Problem transformation methods with distance-based learning for multi-target regression, in: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN, 2020*, pp. 691–696.



**Joakim Linja** received his M.Sc degree from the Department of Physics in University of Jyväskylä, Finland. He is currently working as a PhD researcher at the Faculty of Information Technology in University of Jyväskylä, Finland. His main fields of interests are machine learning, computational science and nanophysics.





**Joonas Hämäläinen** received the B.S. degree in physics, the M.S. degree in applied physics, and the Ph.D. degree in mathematical information technology from University of Jyväskylä, Jyväskylä, Finland, in 2012, 2013, and 2018. He is currently working as a Postdoctoral Researcher at the Faculty of Information Technology, University of Jyväskylä, Jyväskylä, Finland. His main research interests include data mining and machine learning.



**Paavo Nieminen**, PhD, is currently working as a Senior Lecturer at the Faculty of Information Technology, University of Jyväskylä, where he has received also his own academic education and degrees. Alongside teaching and curriculum work, he collaborates in research and supervision on topics of machine learning and computer science education.



**Tommi Kärkkäinen** (TK) received the Ph.D. degree in Mathematical Information Technology from the University of Jyväskylä (JYU), in 1995. Since 2002 he has been serving as a full professor of Mathematical Information Technology at the Faculty of Information Technology (FIT), JYU. TK has led 50 different R&D projects and has been supervising 60 PhD students. He has published over 190 peer-reviewed articles. TK received the Innovation Prize of JYU in 2010. He has served in many administrative positions at FIT and JYU, leading currently a Research Division and a Research Group on Human and Machine based Intelligence in Learning. The main research interests include data mining, machine learning, learning analytics, and nanotechnology. He is a senior member of the IEEE.



**PIV**

**KNOWLEDGE DISCOVERY FROM ATOMIC STRUCTURES  
USING FEATURE IMPORTANCES**

by

Joakim Linja, Joonas Hämäläinen, Antti Pihlajamäki, Paavo Nieminen, Sami Malola,  
Hannu Häkkinen, Tommi Kärkkäinen 2023

<https://arxiv.org/abs/2303.09453>

Manuscript



# Knowledge Discovery from Atomic Structures using Feature Importances

Joakim Linja    Joonas Hämäläinen    Antti Pihlajamäki  
Paavo Nieminen    Sami Malola    Hannu Häkkinen  
Tommi Kärkkäinen

February 19, 2023

## Abstract

Molecular-level understanding of the interactions between the constituents of an atomic structure is essential for designing novel materials in various applications. This need goes beyond the basic knowledge of the number and types of atoms, their chemical composition, and the character of the chemical interactions. The bigger picture takes place on the quantum level which can be addressed by using the Density-functional theory (DFT). Use of DFT, however, is a computationally taxing process, and its results do not readily provide easily interpretable insight into the atomic interactions which would be useful information in material design. An alternative way to address atomic interactions is to use an interpretable machine learning approach, where a predictive DFT surrogate is constructed and analyzed. The purpose of this paper is to propose such a procedure using a modification of the recently published interpretable distance-based regression method. Our tests with a representative benchmark set of molecules and a complex hybrid nanoparticle confirm the viability and usefulness of the proposed approach.

## 1 Introduction

Machine learning (ML) has a potential to provide novel insights into quantum-mechanical properties of atomic systems [1]. Better availability of ML methods influences natural sciences in general, by improving material design and characterization, and through accelerating *ab initio* simulations [2, 3]. As stated in [4], “the synergistic application of machine learning and traditional atomistic modeling continues to serve as an accelerator of discovery”. Density-functional theory (DFT) is one of the most commonly used methods to simulate the electronic structure of matter. Even if it requires significant amounts of computational power, it is still more efficient than, for example, Hartree-Fock or Quantum Monte Carlo methods. Hence, it is able to generate enough data to enable the use of ML to better understand the properties and dependencies of molecular systems, and

to augment and integrate both simulation-based and experimental studies [5]. Especially in DFT-based computational materials science with applications like materials discovery, drug design, renewable energy, and catalysis, the emergence of the data-driven science paradigm has provided great improvements [6].

DFT provides an accurate physics-based calculator able to solve the electronic structure of a molecule or a nanoparticle. DFT is, however, computationally taxing, which is why there have been numerous studies which attempt to form a surrogate of the DFT calculator using machine learning models [7, 8, 9, 10, 11, 12, 13, 14, 15]. Construction of a data-driven surrogate model is done through the use of a descriptor. Depending on the descriptor, the number of features it provides ranges from tens to thousands [2, 3, 16, 17]. In addition to being mere surrogates to taxing computations, such datasets of descriptive features generated from atomic configurations can provide a fertile ground to explore the possibilities of automated methods that weigh and compare the necessity and relative importance of each of the descriptive features with respect to the physical behavior of materials.

Knowledge discovery, in general, is a stepwise process comprising data generation and selection, model or pattern construction, and interpretation of the constructed model to provide knowledge useful in an application area [18, 19]. In this paper, we examine data of atomic structures generated using DFT and descriptors. We then construct a regression model that predicts DFT energies from the descriptors. Finally, useful knowledge regarding atomic interactions comes from examining feature importance scores that are obtained using our proposed algorithm.

Interpretable machine learning is a field of research, where improved understanding of the working logic of machine learning models and algorithms is addressed and advanced [20]. This can be done in a model-agnostic (MA) or model-specific (MS) way, where the former refers to the use of separate explanation techniques and the latter means direct interpretation of the actual model [21]. An example of a model-agnostic approach to better understand gold nanocluster synthesis was given in [22], where the decision tree technique was used to extract rule-based knowledge from a graph convolutional neural network model. Indeed, the MA approach might be the only possible one if we are using completely black box models and algorithms provided as components or services, whereas the MS approach is viable if we can access and analyze the ML model itself. If applicable, the MS interpretation can be more transparent and accurate, thus providing more potential for novel insight and new discoveries [23].

A popular technique for interpreting an ML surrogate is the post-hoc investigation of the saliency of features, i.e., evaluation of feature importances [21, 24, 25]. This technique is behind the most well-known MA techniques like LIME and SHAP [26, 27]. For the distance-based learning machines proposed in [28, 29, 30], the model-specific feature importance formulation Mean-Absolute-Sensitivity (MAS) was proposed and thoroughly experimented in [31]. Indeed, these distance-based methods can be used to construct deterministic and accurate regression models for general and atomic datasets, which scale to high-dimensional feature and target spaces, are tolerant against noisy and irrelevant features, have theo-

retical guarantees through the universal approximation results, and can provide better generalization results compared to off-the-shelf deep neural networks [12, 29, 30, 31, 32]. Furthermore, as noted in [31], calculation of feature scores to represent their importances can be seen as two sides of the same coin: one to select features and the other to perform knowledge discovery.

In this paper, we examine two different kinds of atomic structures: molecules and a protected metal nanocluster. We study the output of the Many-body Tensor Representation (MBTR) descriptor through the use of a distance-based feature selection algorithm, showing that features can be removed while simultaneously providing information on the studied atomic structures as described by the descriptor.

The purpose of this article is three-fold:

- i*) we extend our earlier feature selection method by proposing a new rule to determine the most important feature scores;
- ii*) we extend the comparative assessment of the MAS-based method from [31] by considering new datasets;
- iii*) we provide a proof-of-concept on how the developed feature scoring and ranking method can be used to extract novel knowledge from atomic structures like molecules and hybrid nanoparticles.

To reach these goals, we employ a representative set of atomic structures arising from benchmark molecular datasets and one particular hybrid nanoparticle dataset, which serve both as validation data of the proposed approach and as the proof-of-concept in the atomic structure realm.

The main contributions of this paper are the proposed feature selection algorithm and the showcase of applying it to data from atomic structures and analysis of the results. The analysis reveals that the algorithm is able to detect the most meaningful features, which agree with chemical intuition, from the vast amount of information encoded into MBTR. These features are directly related to the bonding nature of the systems, thus they could be used to distinguish specific characteristics of the molecules, nanoparticles or other atomic systems.

In Section 2 we describe the theoretical background and present the proposed feature selection algorithm. Section 3 presents the datasets and experimental setup. This is followed by the results in Section 4. Finally, the paper is concluded in Section 5.

## 2 Methods

In this section, we provide the relevant information on the existing and new methods used in the experiments. We consider the regression problem, where it is assumed that, for a set of  $N$  atomic structures  $\{A_i\}_{i=1}^N$ , their potential energies have been calculated using DFT. This desired regression output data for a surrogate is denoted with  $\{y_i\}_{i=1}^N$ .

## 2.1 Descriptors and Many-body Tensor Representation

Descriptors are a way to translate an atomic structure into a format understood by a machine learning model [33]. For a descriptor to be functional, it needs to be invariant to the translation and rotation of the described system as well as to the permutations of atom listing [34]. It also needs to be unique for a given structure and continuous, as the descriptor should be able to capture even the smallest changes of the compounds [34].

Descriptors are divided into two main categories, local descriptors and global descriptors [33]. Local descriptors focus on describing the environment of a single atom, whereas global descriptors simultaneously describe the entire atomic structure. Another way to categorize descriptors is whether or not they are able to handle periodic environments [34].

Musil et al. [33] presented the families of descriptors as atom density fields, potential fields, symmetrized local fields, atom centered distributions, sharp and smooth density correlation features, internal coordinates, atomic symmetry functions, permutation invariant polynomials, distance histograms, sorted distances, sorted eigenvalues and molecular graphs. The focus in this paper is on a global, non-periodic (in this case), distance histogram-family descriptor.

Many Body Tensor Representation (MBTR) by Huo et al. [35] is a global descriptor, which relies on Gaussian broadened geometric properties (distances and angles) grouped according to the atomic numbers. It is invariant with respect to translations, rotations, and permutations, which are crucial properties of a descriptor. Without these properties, for example, the indexing of atoms in an atomic system has an effect on the training of the model.

MBTR has description settings known as "k1", "k2", and "k3" which refer to the level of details taken into account in the atomic interactions. The representation is based on the following density distributions as presented by Himanen et al in [34] (with  $Z$  denoting an atomic number):

- k1: the number of elements,

$$F_1^{Z_1}(x) = \sum_l^{|Z_1|} w_1^l D_1^l(x), \text{ where}$$
$$D_1^l(x) = \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(x-g_1(Z_l))^2}{2\sigma_1^2}},$$
$$g_1(Z_l) = Z_l, \text{ the atomic number,}$$

- k2: inverse distances between each element pair,

$$F_2^{Z_1, Z_2}(x) = \sum_l^{|Z_1|} \sum_m^{|Z_2|} w_2^{l, m} D_2^{l, m}(x), \text{ where}$$

$$D_2^{l, m}(x) = \frac{1}{\sigma_2 \sqrt{2\pi}} e^{-\frac{(x - g_2(\mathbf{R}_l, \mathbf{R}_m))^2}{2\sigma_2^2}},$$

$$g_2(\mathbf{R}_l, \mathbf{R}_m) = \frac{1}{|\mathbf{R}_l - \mathbf{R}_m|}, \text{ and}$$

- k3: angle between each element triple,

$$F_3^{Z_1, Z_2, Z_3}(x) = \sum_l^{|Z_1|} \sum_m^{|Z_2|} \sum_n^{|Z_3|} w_3^{l, m, n} D_3^{l, m, n}(x), \text{ where}$$

$$D_3^{l, m, n}(x) = \frac{1}{\sigma_3 \sqrt{2\pi}} e^{-\frac{(x - g_3(\mathbf{R}_l, \mathbf{R}_m, \mathbf{R}_n))^2}{2\sigma_3^2}},$$

$$g_3(\mathbf{R}_l, \mathbf{R}_m, \mathbf{R}_n) = \cos \angle (\mathbf{R}_l - \mathbf{R}_m, \mathbf{R}_n - \mathbf{R}_m).$$

Here  $\mathbf{R}_i$  denotes the position of an atom  $i$ . The weighting functions  $w_k$  can be used to prioritize features if desired [34]. In the context of this paper,  $w_1 = 1$  and, for  $w_2$  and  $w_3$ , we used the exponential weighting functions which are the default settings given by the DDescribe implementation of MBTR [34]. The described three feature sets can be expressed together as  $\mathbf{F}_{1,2,3} = F_1^{Z_1}, F_2^{Z_1, Z_2}, F_3^{Z_1, Z_2, Z_3}$ . The column vector  $\mathbf{F}_{1,2,3}$  forms the descriptor used in this paper. Put together,  $\mathbf{F}_{1,2,3}$  forms high-dimensional data. Ranging from 1100 features to 5400 features with the used datasets.

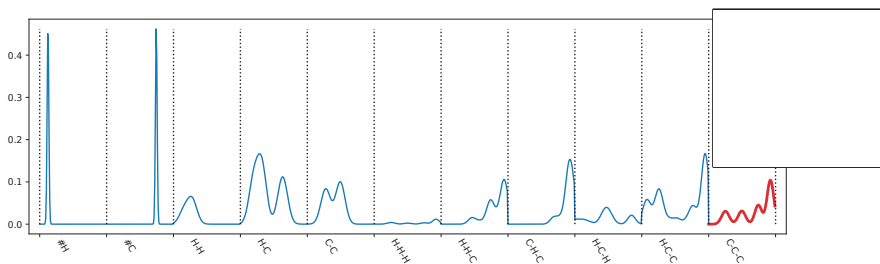


Figure 1: Example of an MBTR descriptor (k1+k2+k3) for Benzene. Each individual atomic interaction is labeled and given dashed vertical lines as borders. The first two, #H and #C, represent k1, the following three (H-H, H-C, and C-C) represent k2, and the remaining interactions represent k3.

Used together,  $\mathbf{F}_{1,2,3}$  describes the number of atoms in an atomic structure, their distances to each other and the angles between them, forming an extensive description of the structure. This description can then be used to study both feature selection and feature importance-based knowledge discovery. In Figure 1, we provide an example of using MBTR with the dataset *Benzene* (presented in Section 3.1), including also a visualization of the molecule. The x-axis in the figure groups together the different interaction types as presented by the MBTR and the y-axis depicts *mean MBTR*: mean over the observations of the whole *Benzene* dataset. The interaction C-C-C has been highlighted in red in Figure 1. This interaction is taken as an example of how the descriptor is formed. All possible angles between carbon atoms have been marked with red lines and angle notations in the inset molecule image of Figure 1. For each descriptor, these angles are calculated for each carbon atom individually, and the combination then forms the C-C-C line highlighted with red. The way to read the notation is that if we were to have an angle A-B-C between atoms A, B and C, it would read as the angle between vectors B-A and B-C.

## 2.2 Extreme Minimal Learning Machine

*Extreme Minimal Learning Machine* (EMLM) is a distance-based machine learning model proposed by Kärkkäinen in 2019 [29]. EMLM is a combination of Extreme Learning Machine (ELM) [36, 37] and Minimal Learning Machine (MLM) [28, 30], combining the distance-based feature map of MLM to the way a regularized least-squares problem is solved in ELM. EMLM works by first constructing a distance matrix

$$(\mathbf{H})_{ij} = \|\mathbf{r}_i - \mathbf{x}_j\|_2, i = 1, \dots, m; j = 1, \dots, N, \quad (1)$$

between observations  $\mathbf{x}_j$  and a set of reference points  $\mathbf{r}_i$ . Here the reference points  $\mathbf{R} = \{\mathbf{r}_i\}_{i=1}^m$  are selected from the given set of  $N$  observations  $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ , i.e.  $\mathbf{R} \subseteq \mathbf{X}$ , and this selection procedure includes the only metaparameter of the method: number of reference points  $m$ . A full EMLM would use all observations as reference points (providing a parameter-free method) but a typical choice would be to apply a strategy, such as RS-maximin, which enables a smaller number of reference points to cover the  $\mathbf{X}$  in a representative manner [38, 30].

In training of EMLM, the weights  $\mathbf{W}$  are calculated using the distance matrix  $\mathbf{H}$ , similarly to the classical ridge regression formula in linear regression:

$$\mathbf{W} \left( \mathbf{H}\mathbf{H}^T + \frac{\alpha N}{m} \mathbf{I} \right) = \mathbf{y}\mathbf{H}^T, \quad (2)$$

where  $\mathbf{y}$  is the vector of DFT-outputs and  $\frac{\alpha N}{m} \mathbf{I}$  the least-squares (ridge regression) regularization term. In the equation,  $\alpha = \sqrt{\epsilon}$  where  $\epsilon$  refers to machine epsilon. After the weights  $\mathbf{W}$  have been obtained from (2), the model’s prediction of a new input vector  $\tilde{\mathbf{x}}$  is calculated by forming the distance matrix between the reference points and  $\tilde{\mathbf{x}}$  according to (1) and then multiplying this with the weights.



The strength of EMLM and other such distance-based models lies in their simplicity of use and robustness. Most importantly, a large and versatile pool of experiments confirm their repeated tendency not to overlearn [29, 30, 32, 12, 31]. From the DFT-surrogate construction perspective, this makes these techniques very different from the popular deep learning methods, which require tedious and extensive selection, tuning, and testing of architecture and metaparameters to manage model complexity and to prevent overlearning [39, 40].

### 2.3 Feature scoring methods

The more complex a measured phenomenon is, the more features are usually required for a surrogate. Events in real world are quite often complex, with many variables. This produces datasets with large numbers of features. However, the assumption is that not all of the measured variables produce features which are actually necessary.

Kohavi and John presented definitions for strong and weak relevance in [41], and noted that the relevance of a feature may not be directly correlated with the optimality of the feature (which clearly is related to what kind of data-driven model is to be constructed). Feature selection algorithms rely on feature scoring in order to assess their relevance or importance. Based on how the scores and/or determination of feature importances is realized, the actual feature selection algorithms can be divided into three main categories: *i*) Filters execute a rule by which they select a feature subset without data-driven model construction. *ii*) Wrappers, which use the machine learning model to assess the importance of features. and *iii*) Hybrids, which are in one way or another a combination of a filter and a wrapper (like decision trees and random forest).

Based on the extensive experiments presented in [31], only the best two feature scoring, ranking, and selection algorithms are used and experimented further in this paper: *Mean Absolute Sensitivity* and *Spearman R*.

#### Feature scoring using Mean Absolute Sensitivity

The partial derivative of a neural network’s output with respect to its input to measure the input sensitivity was proposed in [42]. This technique was enlarged to feature selection in [43]. In order to generate an image-specific saliency map for visual interpretation of a convolutional neural network classifier, the input-output sensitivity was independently rediscovered and proposed in [44].

Formally, as depicted in [45], the derivative of a data-driven model  $M$  with respect to its features  $\frac{\partial M}{\partial \mathbf{x}}$  to assess feature importances emerges from the classical Taylor’s formula. Explicit formulae for the distance-based EMLM and also for a deep, feedforward neural network to calculate a model’s derivatives for an observation were given in [31]. Such calculation provides a pointwise information so that in order to assess the global importance, one can take mean of the

absolute sensitivities (MAS) over the data [43]:

$$MAS = \frac{1}{N} \sum_{i=1}^N \left| \frac{\partial M}{\partial \mathbf{x}_i} \right|. \quad (3)$$

The actual formulae for the EMLM, for  $\tilde{\mathbf{x}} = \mathbf{x}_i$  were given in [31]:

$$\frac{\partial M}{\partial \tilde{\mathbf{x}}} = \mathbf{W}\mathbf{D}^T, \quad \text{where } \mathbf{d}_i = \frac{\mathbf{r}_i - \tilde{\mathbf{x}}}{\max(\alpha, \mathbf{r}_i - \tilde{\mathbf{x}})}, i = 1, \dots, m. \quad (4)$$

MAS can be calculated for any data-driven model which can be differentiated with respect to its features. For regression problems, the calculated MAS value of a feature estimates its importance in the construction of model  $M$  compared to other features, allowing thus MAS to function as a straightforward score for ranking and knowledge discovery purposes.

### Correlation-based feature scoring using Spearman R

The premise of correlation-based feature assessment is that features can be ranked according to their correlation to the output, especially in regression problems [46]. A feature may be positively or negatively correlated, or neutral in respect of the output [47]. In general, features can be categorized into having a strong relevance or a weak relevance [41].

Spearman rank correlation coefficient is a statistical method, which is a non-parametric way to calculate the correlation between two groups of data [48]. In addition, Spearman R does not assume any distributions, which is important since a dataset cannot be expected to have normal distribution. The classical formula, for the correlation coefficient vector  $\rho$ , reads as

$$\rho = \frac{\sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{y}_i - \bar{\mathbf{y}})}{\sqrt{\sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}})^2 \sum_{i=1}^N (\mathbf{y}_i - \bar{\mathbf{y}})^2}}.$$

Note that if no preprocessing is assumed then, similarly to (4), the denominator of the correlation should be safeguarded against the case of a constant feature.

## 2.4 The developed algorithm

The proposed feature scoring, ranking, selection algorithm is an extension on the idea of the one-shot wrapper presented by Linja et al. in [31]. The development was started by the discovery that there are cases when the features of a dataset are ranked and sorted, the curve formed by the ranking values is not suitable for a kneepoint detection. The problematic situation is illustrated in Figure 2.

The proposed algorithm aims to fix the issue by calculating the slope of each line between the highest ranking value and the other ranking values. It then performs a curve fit to the slope values and uses the fitted curve to determine the selected features. The algorithm is presented in Algorithm 1. It's function is illustrated in Figure 3.

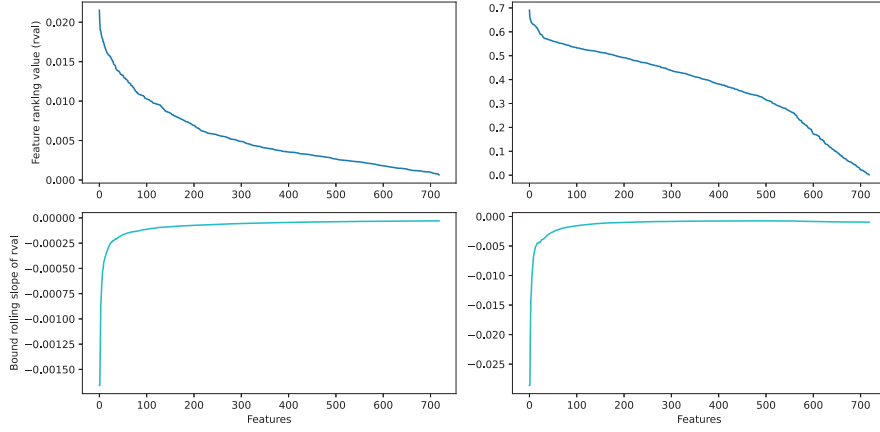


Figure 2: The ranking values of  $Au_{38}$  data calculated with MAS and with Spearman R as well as the next step of the proposed feature selection algorithm to point out that the shape of the ranking values produces similar curve in both cases

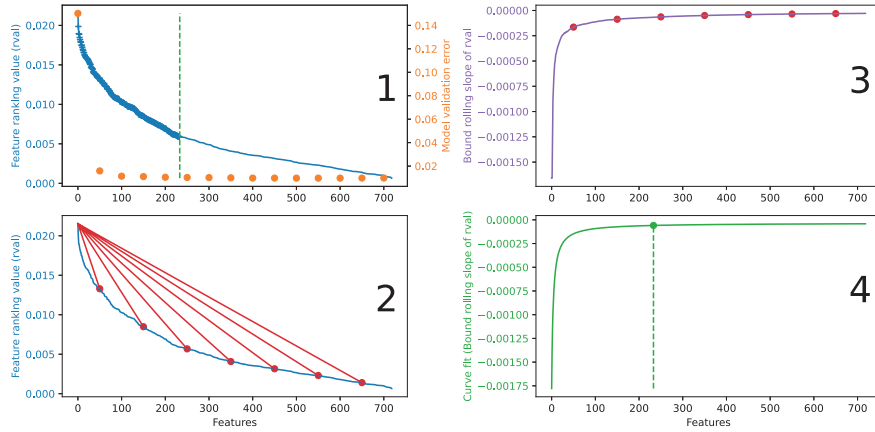


Figure 3: Illustration of the function of the proposed feature selection algorithm

Figure 3 uses the  $Au_{38}$  dataset as an example, showing the ranking values of features of MBTR k2 as calculated by MAS and set to descending order in subfigure 1. It also shows the validation errors of the model as well as the final cutoff point given by the last step in subfigure 4. Subfigure 2 shows the bound rolling slope which forms the curve in subfigure 3. Finally, subfigure 4 shows a curve fit of the curve in subfigure 3 and the cutoff point.

---

**Algorithm 1** Feature Importance Detector - FID

---

**Input:** Input data  $\{\mathbf{x}_j \in \mathbb{R}^n \mid j = 1, \dots, N\}$ , target data  $\{y_j \in \mathbb{R} \mid j = 1, \dots, N\}$ , cutoff value  $\{k \in \mathbb{R} \mid 0 < k < 1\}$

**Output:** Feature score vector  $\mathbf{v}$ , Ranking permutation  $\mathbf{p}$ , and indices of the most important features  $\mathcal{F}$

- 1: Score the  $n$  features of the input data by using a feature scoring algorithm —here Spearman R or MAS
  - 2: Sort the feature scoring values  $\{v_i\}_{i=1}^n$  into descending order and keep permutation function  $p(i)$  which allows to return from sorted indices to original ones
  - 3: Calculate a slope value for each feature:  $s_i = \frac{v_i - v_1}{i - 1}, i = 2, \dots, n$
  - 4: Estimate  $a, b, c, d \in \mathbb{R}$  by fitting the curve  $C(i) = -\frac{a}{bi+c} + d$  to the points  $\{s_i\}$
  - 5: Define total increase of  $C(i)$  denoted as  $C^* = C(n) - C(1)$
  - 6: Return feature set  $\mathcal{F} = \{p(i) \mid C(i) < C(n) - k C^*, i = 1, \dots, n\}$
-

## 3 Experimental Setting

### 3.1 Datasets

In this study, we used the following datasets of the atomic structures: all benchmark data from sGDML (Symmetric Gradient Domain Machine Learning) [49, 1, 50] and the hybrid nanoparticle dataset *Au38QT*, which originates from the simulations by Juarez-Mosqueda et al. [51]. Among the used datasets, *Au38QT* is of special interest because of its higher complexity and due to the interest nanoscience community and machine learning community are giving to nanoclusters, as well as our own interest in it [12, 32]. Its structure was first found experimentally by Qian et al in 2010 and an isomer for it was found by Tian et al in 2015 [52, 53]. As a monolayer protected cluster (MPC), it holds high levels of potential applications, such as biolabeling, catalysis, medicine, solar energy and display panels [54].

The datasets are presented in Table 1. Example images of the molecule or cluster of each dataset is presented in Appendix A. Each dataset is based on either a real-world molecule or a nanoparticle. The observations in each have been gained by simulating the atomic object in DFT-based molecular dynamics simulation. The target variable in each dataset is the potential energy of an observations configuration.

### 3.2 Experimental setup

In the molecular dynamics simulation data, each atom configuration was converted into a set of descriptors using MBTR (k1+k2+k3). Using all three types of MBTR descriptors produces a vector which describes the elements, distances and angles of each atom configuration. Each dataset was split into subsets according to their target variable using distribution-optimal folding, DOP-SCV [55, 56]. The number of subsets was set to be 5 for datasets with less than 250000 observations and 10 for the rest. For each subset, constant features and duplicate observations were removed. In addition, each subset was minmax-scaled to the interval  $[0, 1]$ . Then, the feature importance detection algorithm, Algorithm 1, was performed on each data subset. The following variables were measured:

- baseline validation RMSE of a data subset,
- the same after feature selection has reduced the number of features in the subset and
- whether or not a feature was selected and if removed, the point in the algorithm when it was removed.

In order to focus the experiments on model’s generalization instead of its scalability, We performed cross-validation in reverse fashion, such that one subset was used as a training set and the rest of the subsets as validation sets. The RMSE error was calculated in electron volts (eV).

Table 1: Datasets used in this study. The number of features is determined by the used descriptor, described in Section 2.1 and Section 3.2.

Dataset	# Observations	# Features	# Elements
Au38QT	28761	5400	H72,C24,S24,Au38
Benzene2018	49863	1100	H6,C6
DocosahexaenoicAcid	69753	2700	H32,C22,O2
AlanineTetrapeptide	85109	5400	H22,C12,N4,O4
Azobenzene	99999	2700	H10,C12,N2
Paracetamol	106490	5400	H9,C8,N1,O2
Uracil	133770	5400	H4,C4,N2,O2
Aspirin	211762	2700	H8,C9,O4
Salicylic Acid	320231	2700	H6,C7,O3
Naphthalene	326250	1100	H8,C10
Toluene	442790	1100	H8,C7
Ethanol	555092	2700	H6,C2,O1

We evaluated the feature selection capability of Algorithm 1 using two different feature scoring algorithms, *MAS* and *Spearman R*, and three different cutoff values:  $10^{-1}$ ,  $10^{-2}$  and  $10^{-3}$ . In addition, we extracted information from MBTR regarding the relevance of each atomic interaction via a *term frequency – inverse document frequency*-type measurement [57], denoted here as "interaction relevance"  $\iota$ :

$$\iota(F_k^{Z_1, \dots, Z_k}) = \frac{1}{N_D} \sum_{d \in D} \frac{N_{\text{FS}}(d)}{N_{\text{res}}}, \quad (5)$$

where  $F_k^{Z_1, \dots, Z_k}$  is the atomic interaction (as defined by MBTR, for example, H-C)  $N_D$  is the number of datasets in which  $F_k^{Z_1, \dots, Z_k}$  appears in,  $D$  is a set of datasets in which  $F_k^{Z_1, \dots, Z_k}$  appears in,  $d \in D$  refers to a dataset,  $N_{\text{FS}}$  is the number of features that remains of the described  $F_k^{Z_1, \dots, Z_k}$  after feature selection and  $N_{\text{res}}$  is the resolution parameter of MBTR, i.e., the number of elements used by MBTR to describe each atomic interaction  $F_k^{Z_1, \dots, Z_k}$ . The normalization over datasets is necessary since not all atomic interactions are present in all datasets. Thus,  $\iota$  defines the importance of an atomic interaction normalized over datasets which can then be further used for knowledge discover.

## 4 Results

In this section, we present and summarize the results of the experiments.

### 4.1 Selection of ranking algorithm and cutoff value

We report the generalizability and FS success with figures where the remaining number of features is on the x-axis and on the y-axis is the validation RMSE



after FS divided by the validation RMSE before FS. Figure 4 and Figure 5 show the results for the validation RMSE ratio (error after FS divided by error with a full set of features) as a function of the number of features remaining. This error ratio reflects an improvement in regarding model accuracy when it is less than 1. Higher values than 1 reflect that the reduced model accuracy is degenerated due to selected subset of features.

From these figures, we can observe that the cutoff value 0.1 does not include enough features with either feature scoring algorithm. However, cutoff values 0.01 and 0.001 provide at least as low validation errors as the base method but with a significantly reduced number of features. The second observation is that *MAS* has a more consistent behavior than *Spearman R* when cutoffs 0.01 and 0.001 are considered.

We used *MAS* with the cutoff 0.01 to produce figures for all datasets (Figures 6–17) where the MBTR (k2+k3) has been laid out with the kept features. The k1 part of these figures was removed since, in each case, they were constants that were removed in a preprocessing step. The MBTR figures with the selected features for the datasets (Figures 6–17) allow one to see what the machine learning model considers to be the most important interactions. At the same time, it also allows one to sanity check the function of the feature selection algorithm, provided one has understanding of the involved chemistry.

Figure 6 presents the MBTR for the most complex object of the twelve datasets, a hybrid metal nanoparticle (a monolayer protected nanocluster). The nanoparticle consists of a cluster core composed of gold atoms, a sulfur interface and then 16 hydrocarbon chains in a protective shell around the cluster core. The difference of the importances of the features is immediately visible in Figure 6.

Then, the choice was between *MAS* and *Spearman R*. Observing Figure 4 and Figure 5 we can see that *MAS* produces lower validation errors in a more consistent manner. Due to these findings, the results reported in Section 4.2 and Section 4.3 are given with *MAS* using cutoff of 0.01.

It should be noted, that the validation errors for datasets *Docosahexaenoic Acid* and *AlanineTetrapeptide* presented in Figure 4 have a higher validation error than what the other datasets have. Both *Docosahexaenoic Acid* and *AlanineTetrapeptide* are long molecule chains with many degrees of freedom, which results in a set of complicated movement during molecular dynamics simulation.

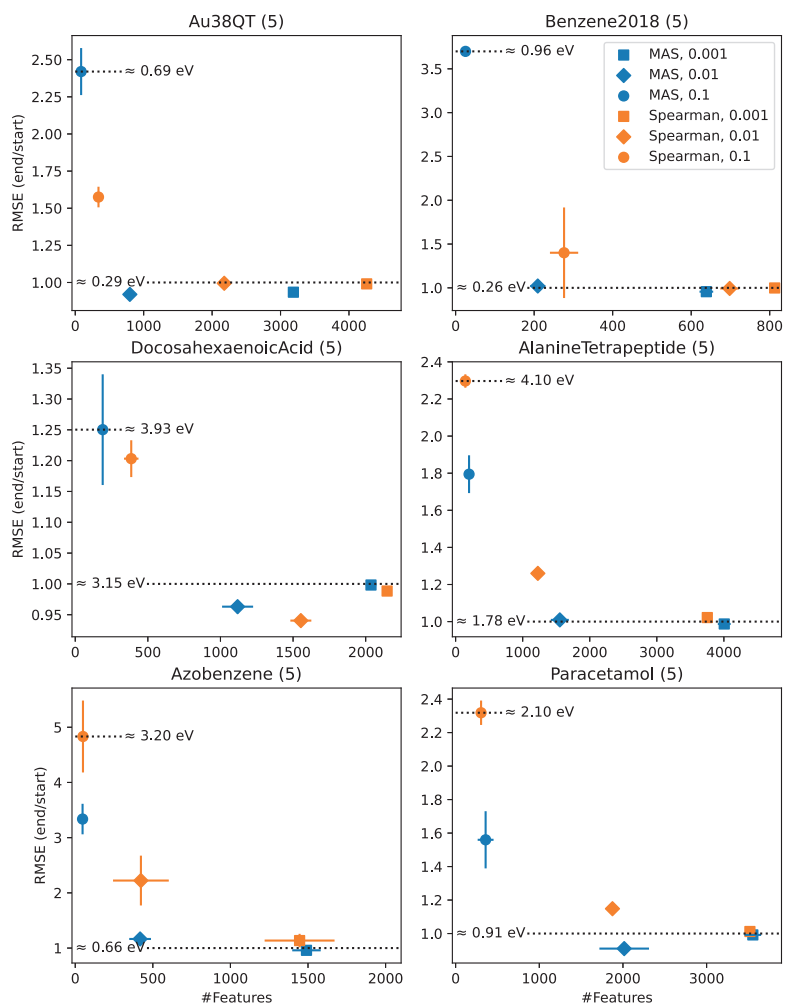


Figure 4: The number of features remaining after feature selection vs. feature selection success measured as model validation RMSE.

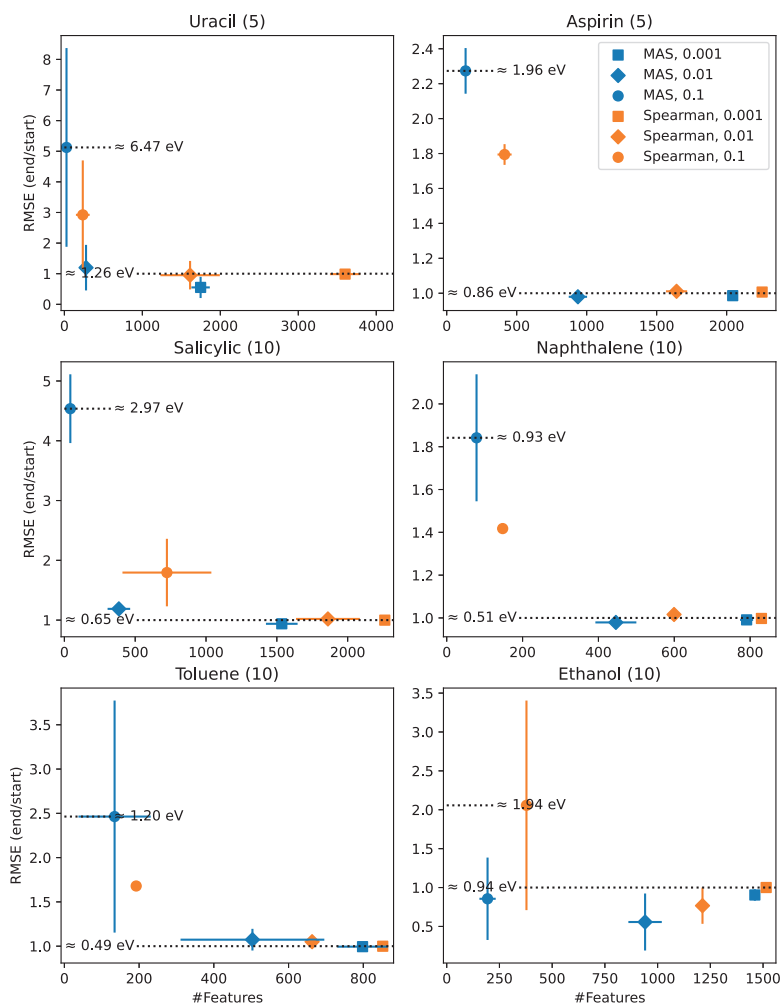


Figure 5: The number of features remaining after feature selection vs. feature selection success measured as model validation RMSE.

## 4.2 Selected features

Figures 6–17 contain the mean MBTR for each dataset, along with the selected features (*MAS* and 0.01) as well as the number of times each selected feature was selected. As a general comment, the results presented here are in agreement with chemical intuition. The chemical bonds and interactions that one would expect to see are present in the figures.

It is interesting that the most complex structure, *Au38QT* does not require the highest number of features. The most plausible cause of this characteristic is the structurally distinguishable three chemical environments: metallic core, metal–ligand interface and protecting ligands. In the simulation data for *Au38QT*, the cluster does break apart at high temperatures but even in those situations, the three main environments of the cluster remain. The outcome is that from the point of view of most individual atoms, the local environment remains the same. This lessens the need for describing features. In addition, the metal–ligand interface of *Au38QT* is one of the defining factors for the stability of the structure [51]. Therefore, it is expected that the three distinct chemical environments remain visible among the selected features, and as is seen in Figure 6, they remain visible.

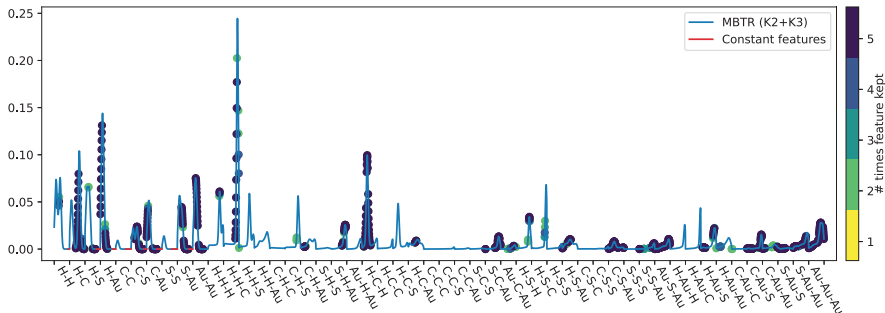


Figure 6: The mean MBTR ( $k_2+k_3$ ) for dataset *Au38QT* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

The rest of the datasets depict organic molecules which are significantly smaller than *Au38QT*. This generates a clear distinction between *Au38QT* and the organic molecules. In the case of the smaller molecules there are fewer atoms in total, which means that each atom has relatively larger contribution to the potential energy of the molecule. In addition to each atom having a relatively larger contribution, the smaller molecules themselves have less moving parts and thus potentially have less movement and higher stability, when compared to *Au38QT*.

The remaining features and the MBTR descriptors of the organic molecules are presented in Figures 7–17. As mentioned earlier in this section, the organic molecules have generally demanded a high number of features. In the case of *Benzene* in dataset *Benzene2018*, very few features are needed since the molecule

is a stable ring. In the cases of *Docosahexaenoic Acid* and *AlanineTetrapeptide* in Figures 8 and 9, both are long chains with multiple and varying interactions. This means that each individual atom sees numerous local environments and that the molecule chain, as it can twist and turn or remain straight or anything in between, creates a complex potential energy surface. The complexity of the potential energy surface then requires a large number of features to work with.

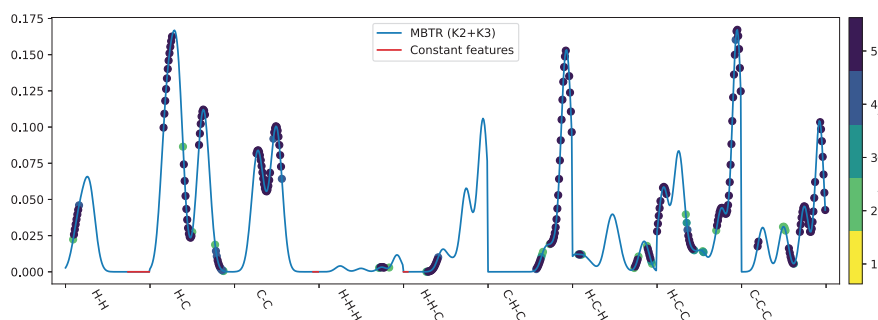


Figure 7: The mean MBTR (k2+k3) for dataset *Benzene2018* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

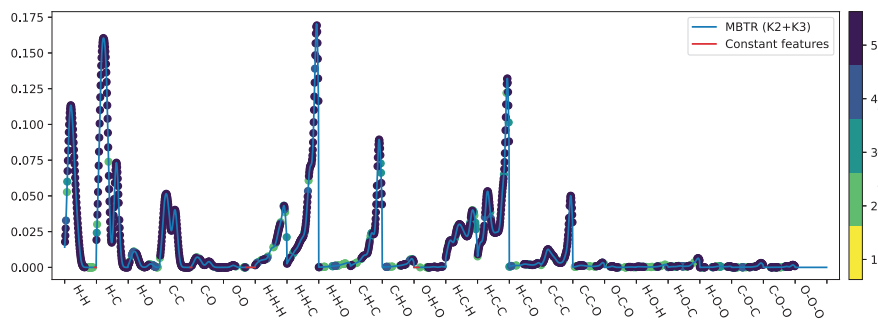


Figure 8: The mean MBTR (k2+k3) for dataset *DocosahexaenoicAcid* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

Among the remaining molecules, *Azobenzene* in Figure 10, *Uracil* in Figure 12, *Salicylic Acid* in Figure 14 and *Toluene* in Figure 16 are similar to *Benzene* in the way that the molecules have stable structures and not a lot of movement. This results in them requiring fewer number of features. On the other hand are molecules *Paracetamol* in Figure 11, *Aspirin* in Figure 13, *Naphthalene* in Figure 15 and *Ethanol* in Figure 17 which required a larger number of features to be kept. Each (except *Naphthalene*) has molecular groups which are able to move, which may be the reason for their need for features. *Naphthalene* is an unexpected case, as the molecule does not have moving parts. The structure of

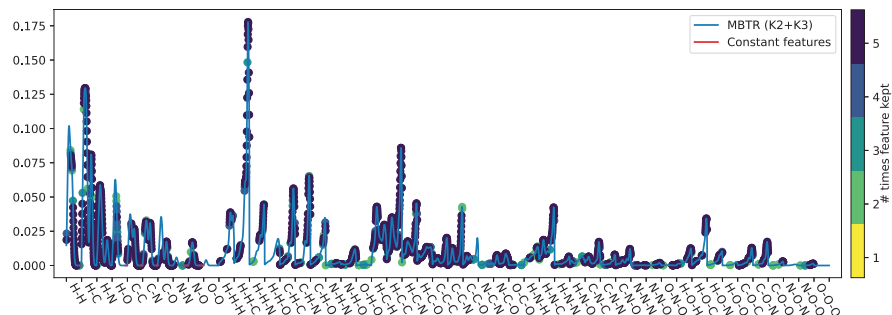


Figure 9: The mean MBTR ( $k_2+k_3$ ) for dataset *AlanineTetrapeptide* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

*Naphthalene* is however clearly visible in the selected features, the distances and angles have remained as expected.

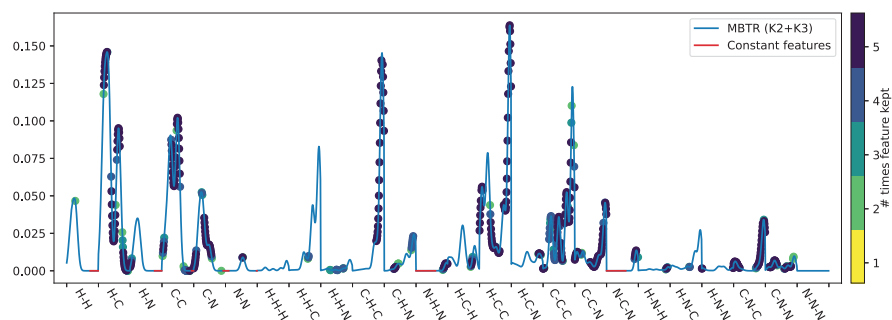


Figure 10: The mean MBTR ( $k_2+k_3$ ) for dataset *Azobenzene* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

The smaller organic molecules surprised with the number of features they required. A potential point of optimization would be to optimize the cutoff parameter to *Au38QT* and the organic molecules separately, instead of optimizing both at the same time. The expectation would be that due to the difference in chemistry, the cutoff points would differ in fully optimized situation. It would then affect the selected number of features.



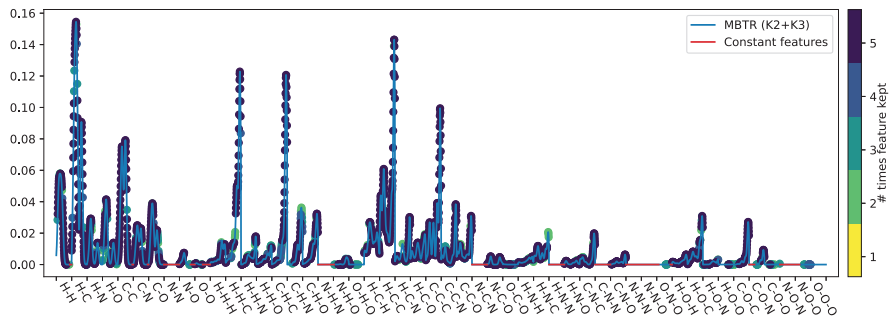


Figure 11: The mean MBTR ( $k_2+k_3$ ) for dataset *Paracetamol* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

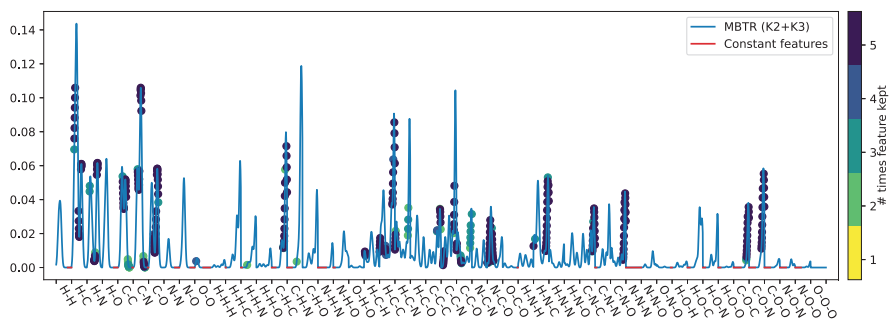


Figure 12: The mean MBTR ( $k_2+k_3$ ) for dataset *Uracil* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

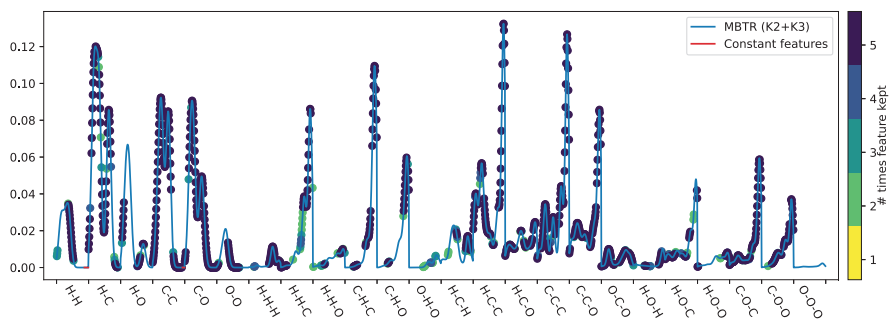


Figure 13: The mean MBTR ( $k_2+k_3$ ) for dataset *Aspirin* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

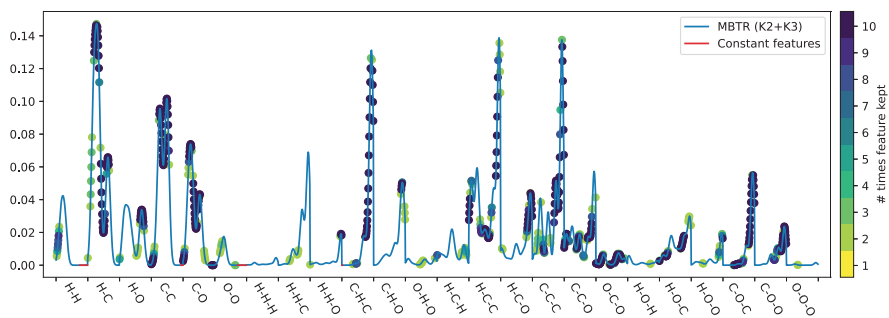


Figure 14: The mean MBTR (k2+k3) for dataset *Salicylic* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

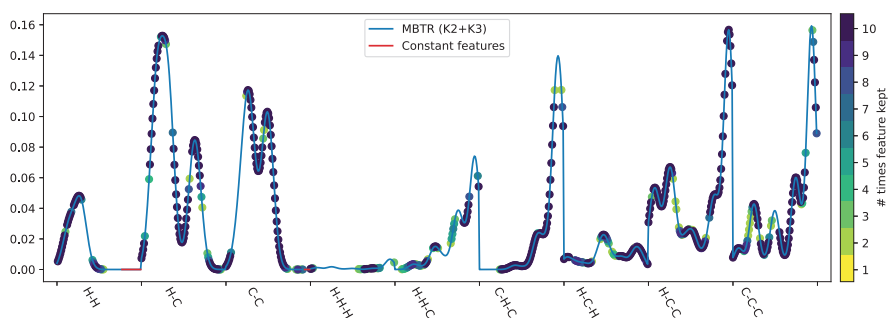


Figure 15: The mean MBTR (k2+k3) for dataset *Naphthalene* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

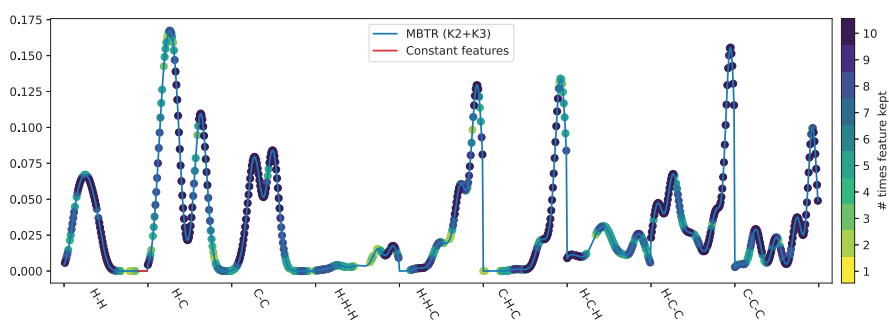


Figure 16: The mean MBTR (k2+k3) for dataset *Toluene* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

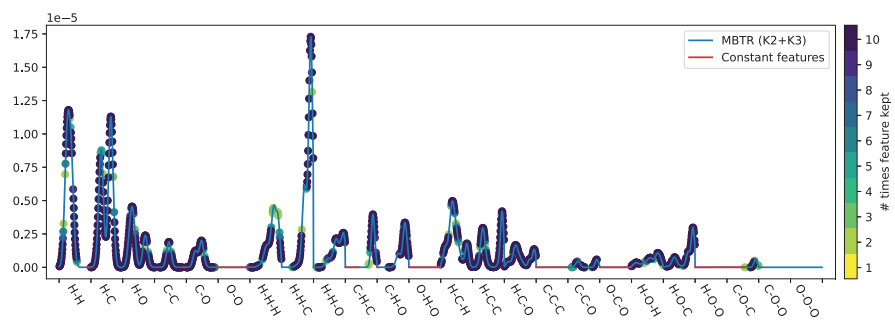


Figure 17: The mean MBTR (k2+k3) for dataset *Ethanol* with features selected by Algorithm 1 using *MAS* and cutoff of 0.01

### 4.3 Interaction relevance

The results here are a summarization of the chemical interactions present in the studied atomic structures. The results for the interaction relevance are presented in Figures 18 and 19 for organic molecules and Figure 20 for *Au38QT*. The results were split into figures this way due to the difference between the chemistry of an organic molecule and the chemistry of a hybrid nanoparticle.

Considering the importance of carbon in organic molecules, it is not a surprise that interactions with carbon were the most important atomic bonds in the datasets in Figure 18. The apparent lack of importance for N-N bond distance is explained by the N-N bond in *Azobenzene* being a double bond and said N-N bond being the only example of N-N bond present in the datasets. It is an especially stable bond, thus in the eyes of feature selection, N-N bonds had little importance to the potential energy. We would expect the N-N bar to have more presence had the datasets contained N-N bonds that were more flexible.

From the angles between atoms presented in Figure 19, the angle H-C-C has the highest relevance. This is again expected based on the nature of organic molecules. The same trend as in Figure 18 is seen in Figure 19, interactions containing carbon are ranked the most relevant.

The results for the *Au38QT* are presented in Figure 20. One can clearly see the dominant effects of gold and sulfur in the interaction relevances. This is again expected based on knowledge of chemistry. An interesting note is that the highest relevances are given to gold-sulfur-gold and carbon-sulfur-gold. In other words, the interface between the gold core and the ligands. Au-S-Au and Au-S-C angles are generally around  $90^\circ$ , hence they have a clear local energy minimum configuration. Deviation from this angle is expected to have visible effect on the potential energy, which explains why they get high relevance.

While the results presented here are mostly a confirmation that the proposed feature selection method works as intended, results like these could be used in the prioritization of atom-atom parametrization. In other word, it is an automated way to print out what the machine learning model in combination with the

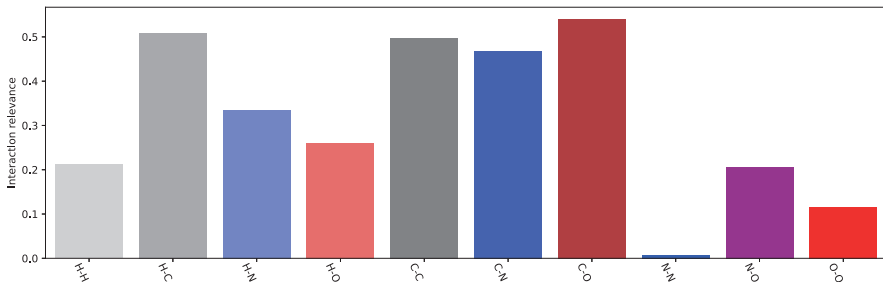


Figure 18: Atomic interaction presence for all organic molecule datasets with MBTR k2 (inverse distances). Interactions with an empty bar were not present in the datasets after feature selection.

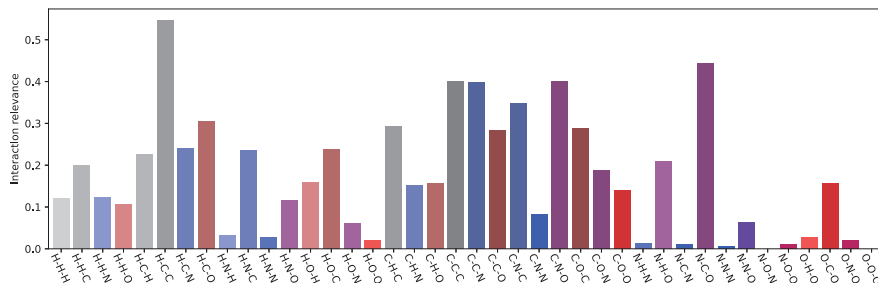


Figure 19: Atomic interaction presence for all organic molecule datasets with MBTR k3 (angle between a triple). Interactions with an empty bar were not present in the datasets after feature selection.

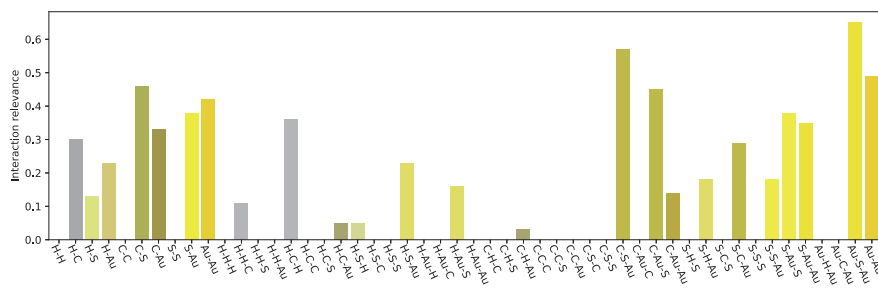


Figure 20: Atomic interaction presence for dataset *Au38QT* with MBTR k2+k3. Interactions with an empty bar were not present in the datasets after feature selection.

feature selection method has seen as the most relevant area of interest.

## 5 Conclusions

The proposed feature selection algorithm was developed and tested with datasets based on molecular dynamics simulations on organic molecules and a hybrid nanoparticle. The Feature Importance Detector requires only a single hyperparameter when used with full EMLM (all of input data selected as reference points) and it was validated by using it with simulation data and the analysis of the selected features. The hyperparameter of FID was first searched by comparing Spearman R and MAS to each other. Other potential feature scoring algorithms were not used here due to the testing made earlier by Linja et al. in [31]. The FID was then used to select features from the used datasets. The results were analyzed with the use of domain knowledge. Finally, interaction relevance score was used to present another way to infer information from the combination of feature selection method, machine learning model, descriptor and molecular dynamics simulation data.

The feature relevance metric was used to summarize the chemical information, as given by the molecular dynamics data, descriptor, EMLM and the proposed feature selection algorithm. It was used to aggregate the information in the organic molecules and single out the *Au38QT*. We conclude that the proposed feature selection algorithm functions as intended, as the results it gave were analyzable and verifiable through domain knowledge.

As this work utilized a single descriptor, one future work would be to compare the result analysis to the result analysis of other descriptors. Jäger et al. [8] showed that there are situations where local descriptors are better than global descriptors. This naturally leads to a future work where local and global descriptors are used simultaneously with FID to see the relative importances of both. An additional avenue of research would be to focus more on different nanoclusters and attempt to analyze whether the features or the interaction relevances have common elements among the various nanoclusters. Another option would be to work on gaining a model from descriptor to 3D structure.

### Acknowledgements

This work has been supported by the Academy of Finland through the project 351579 (MLNovCat). We acknowledge grants of computer capacity from the Finnish Grid and Cloud Infrastructure (**FCCI**; persistent identifier urn:nbn:fi:research-infras-2016072533).

## Conflict of interest

The authors declare no conflict of interest.

## References

- [1] Kristof T. Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R. Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor

- neural networks. *Nature Communications*, 8(1):13890, January 2017.
- [2] Dane Morgan and Ryan Jacobs. Opportunities and challenges for machine learning in materials science. *Annual Review of Materials Research*, 50:71–103, 2020.
  - [3] Sebastian Raschka and Benjamin Kaufman. Machine learning and AI-based approaches for bioactive ligand discovery and GPCR-ligand recognition. *Methods*, 180:89–110, 2020.
  - [4] Michele Ceriotti, Cecilia Clementi, and O Anatole von Lilienfeld. Introduction: machine learning at the atomic scale. *Chemical Reviews*, 121(16):9719–9721, 2021.
  - [5] John A Keith, Valentin Vassilev-Galindo, Bingqing Cheng, Stefan Chmiela, Michael Gastegger, Klaus-Robert Müller, and Alexandre Tkatchenko. Combining machine learning and computational chemistry for predictive insights into chemical systems. *Chemical reviews*, 121(16):9816–9872, 2021.
  - [6] Gabriel R. Schleder, Antonio C.M. Padilha, Carlos Mera Acosta, Marcio Costa, and Adalberto Fazzio. From DFT to machine learning: recent approaches to materials science—a review. *Journal of Physics: Materials*, 2(3):032001, 2019.
  - [7] James Daniel Whitfield, Norbert Schuch, and Frank Verstraete. The computational complexity of density functional theory. In *Many-Electron Approaches in Physics, Chemistry and Mathematics*, pages 245–260. Springer International Publishing, 2014.
  - [8] Marc O. J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Lauri Himanen, and Adam S. Foster. Machine learning hydrogen adsorption on nanoclusters through structural descriptors. *npj Computational Materials*, 4(1):37, July 2018.
  - [9] Claudio Zeni, Kevin Rossi, Aldo Glielmo, *Ádám Fekete*, Nicola Gaston, Francesca Baletto, and Alessandro De Vita. Building machine learning force fields for nanoclusters. *The Journal of Chemical Physics*, 148(24):241739, 2018.
  - [10] Gihan Panapitiya, Guillermo Avendaño-Franco, Pengju Ren, Xiaodong Wen, Yongwang Li, and James P. Lewis. Machine-learning prediction of CO adsorption in thiolated, Ag-alloyed Au nanoclusters. *Journal of the American Chemical Society*, 140(50):17508–17514, December 2018.
  - [11] Alexandre Tkatchenko. Machine learning for chemical discovery. *Nature Communications*, 11(1):4125, August 2020.
  - [12] Antti Pihlajamäki, Joonas Hämäläinen, Joakim Linja, Paavo Nieminen, Sami Malola, Tommi Kärkkäinen, and Hannu Häkkinen. Monte Carlo simulations of Au<sub>38</sub>(SCH<sub>3</sub>)<sub>24</sub> nanocluster using distance-based machine



- learning methods. *The Journal of Physical Chemistry A*, 124(23):4827–4836, 2020. PMID: 32412747.
- [13] Huijie Zhen, Liang Liu, Zezhou Lin, Siyan Gao, Xiaolin Li, and Xi Zhang. Physically compatible machine learning study on the Pt–Ni nanoclusters. *The Journal of Physical Chemistry Letters*, 12(5):1573–1580, February 2021.
- [14] Maddalena D’Amore, Gentoku Takasao, Hiroki Chikuma, Toru Wada, Toshiaki Taniike, Fabien Pascale, and Anna Maria Ferrari. Spectroscopic fingerprints of MgCl<sub>2</sub>/TiCl<sub>4</sub> nanoclusters determined by machine learning and DFT. *The Journal of Physical Chemistry C*, 125(36):20048–20058, September 2021.
- [15] Aishwaryo Ghosh, Soumendu Datta, and Tanusri Saha-Dasgupta. Understanding the trend in core–shell preferences for bimetallic nanoclusters: A machine learning approach. *The Journal of Physical Chemistry C*, 126(15):6847–6853, April 2022.
- [16] Sally R. Ellingson, Brian Davis, and Jonathan Allen. Machine learning and ligand binding predictions: a review of data, methods, and obstacles. *Biochimica et Biophysica Acta (BBA)-General Subjects*, 1864(6):129545, 2020.
- [17] Anthony Yu-Tung Wang, Ryan J. Murdock, Steven K. Kauwe, Anton O. Oliynyk, Aleksander Gurlo, Jakoah Brgoch, Kristin A. Persson, and Taylor D. Sparks. Machine learning for materials scientists: an introductory guide toward best practices. *Chemistry of Materials*, 32(12):4954–4965, 2020.
- [18] Usama Fayyad, Gregory Piatesky-Shapiro, and Padhraic Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34, 1996.
- [19] Anna Rotondo and Fergus Quilligan. Evolution paths for knowledge discovery and data mining process models. *SN Computer Science*, 1(2):1–19, 2020.
- [20] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access*, 6:52138–52160, 2018.
- [21] Nadia Burkart and Marco F. Huber. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317, 2021.
- [22] Jiali Li, Tiankai Chen, Kaizhuo Lim, Lingtong Chen, Saif A. Khan, Jianping Xie, and Xiaonan Wang. Deep learning accelerated gold nanocluster synthesis. *Advanced Intelligent Systems*, 1(3):1900029, 2019.

- [23] Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216, 2020.
- [24] Maksymilian Wojtas and Ke Chen. Feature importance ranking for deep learning. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, volume 33, pages 5105–5114, 2020.
- [25] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020.
- [26] Mirka Saarela and Susanne Jauhiainen. Comparison of feature importance measures as explanations for classification models. *SN Applied Sciences*, 3:1–12, 2021.
- [27] Mirka Saarela, Ville Heilala, Päivikki Jääskelä, Anne Rantakaulio, and Tommi Kärkkäinen. Explainable student agency analytics. *IEEE Access*, 9:137444–137459, 2021.
- [28] Amauri Holanda de Souza, Francesco Corona, Guilherme A. Barreto, Yoan Miche, and Amaury Lendasse. Minimal learning machine: A novel supervised distance-based approach for regression and classification. *Neurocomputing*, 164:34 – 44, 2015.
- [29] Tommi Kärkkäinen. Extreme minimal learning machine: Ridge regression with distance-based basis. *Neurocomputing*, 342:33–48, 2019. Advances in artificial neural networks, machine learning and computational intelligence.
- [30] Joonas Hämäläinen, Alisson S. C. Alencar, Tommi Kärkkäinen, César L. C. Mattos, Amauri H. Souza Júnior, and João P. P. Gomes. Minimal learning machine: Theoretical results and clustering-based reference point selection. *Journal of Machine Learning Research*, 21(239):1–29, 2020.
- [31] Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, and Tommi Kärkkäinen. Feature selection for distance-based regression: An umbrella review and a one-shot wrapper. *Neurocomputing*, 518:344–359, 2023.
- [32] Joakim Linja, Joonas Hämäläinen, Paavo Nieminen, and Tommi Kärkkäinen. Do randomized algorithms improve the efficiency of minimal learning machine? *Machine Learning and Knowledge Extraction*, 2(4):533–557, November 2020.
- [33] Felix Musil, Andrea Grisafi, Albert P. Bartók, Christoph Ortner, Gábor Csányi, and Michele Ceriotti. Physics-inspired structural representations for molecules and materials. *Chemical Reviews*, 121(16):9759–9815, August 2021.

- [34] Lauri Himanen, Marc O.J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. DScribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020.
- [35] Haoyan Huo and Matthias Rupp. Unified representation of molecules and crystals for machine learning. *Machine Learning: Science and Technology*, 3(4):045017, November 2022.
- [36] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, volume 2, pages 985–990 vol.2, 2004.
- [37] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1):489–501, 2006. Neural Networks.
- [38] Joonas Hämäläinen. *Improvements and applications of the elements of prototype-based clustering*. PhD thesis, University of Jyväskylä, 2018.
- [39] Alexios Koutsoukas, Keith J. Monaghan, Xiaoli Li, and Jun Huan. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. *Journal of cheminformatics*, 9(1):1–13, 2017.
- [40] Hongjian Li, Kam-Heung Sze, Gang Lu, and Pedro J. Ballester. Machine-learning scoring functions for structure-based drug lead optimization. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(1):e1465, 2021.
- [41] Ron Kohavi and George H. John. The wrapper approach. In Huan Liu and Hiroshi Motoda, editors, *Feature Extraction, Construction and Selection: A Data Mining Perspective*, pages 33–50. Springer US, Boston, MA, 1998.
- [42] Yannis Dimopoulos, Paul Bourret, and Sovan Lek. Use of some sensitivity criteria for choosing networks with good generalization ability. *Neural Processing Letters*, 2(6):1–4, 1995.
- [43] Tommi Kärkkäinen. Assessment of feature saliency of MLP using analytic sensitivity. In *European symposium on artificial neural networks, computational intelligence and machine learning-ESANN2015. Presses universitaires de Louvain*, pages 273–278, 2015.
- [44] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

- [45] Tommi Kärkkäinen. On the role of Taylor’s formula in machine learning. In *Book title TBA*, chapter Impact of scientific computing on science and society. Springer Nature, 2022. (18 pages, to appear).
- [46] Mark A. Hall and Lloyd A. Smith. Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper. In *Proceedings of the Twelfth International FLAIRS Conference.*, pages 235–240, California, 1999. AAAI Press.
- [47] David A. Bell and Hui Wang. A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41(2):175–195, November 2000.
- [48] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 2023/02/03/1904. Full publication date: Jan., 1904.
- [49] Stefan Chmiela, Alexandre Tkatchenko, Huziel E. Saucedo, Igor Poltavsky, Kristof T. Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*, 3(5):e1603015, 2017.
- [50] Stefan Chmiela, Huziel E. Saucedo, Klaus-Robert Müller, and Alexandre Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature Communications*, 9(1):3887, September 2018.
- [51] Rosalba Juarez-Mosqueda, Sami Malola, and Hannu Häkkinen. Ab initio molecular dynamics studies of Au<sub>38</sub>(SR)<sub>24</sub> isomers under heating. *The European Physical Journal D*, 73(62), 2019.
- [52] Huifeng Qian, William T. Eckenhoff, Yan Zhu, Tomislav Pintauer, and Rongchao Jin. Total structure determination of thiolate-protected Au<sub>38</sub> nanoparticles. *Journal of the American Chemical Society*, 132(24):8280–8281, June 2010.
- [53] Shubo Tian, Yi-Zhi Li, Man-Bo Li, Jinyun Yuan, Jinlong Yang, Zhikun Wu, and Rongchao Jin. Structural isomerism in gold nanoparticles revealed by X-ray crystallography. *Nature Communications*, 6(1):8667, October 2015.
- [54] Tatsuya Tsukuda and Hannu Häkkinen. *Protected metal clusters: from fundamentals to applications*, volume 9 of *Frontiers of Nanoscience*. Elsevier, 1 edition, 2015. Ebook ISBN: 9780444635020.
- [55] Jose García Moreno-Torres, José A Sáez, and Francisco Herrera. Study on the impact of partition-induced dataset shift on  $k$ -fold cross-validation. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1304–1312, 2012.

- [56] Tommi Kärkkäinen. On cross-validation for MLP model evaluation. In *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2014, Joensuu, Finland, August 20-22, 2014. Proceedings*, pages 291–300. Springer, 2014.
- [57] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive data sets*. Cambridge university press, 2020.

## Appendix A

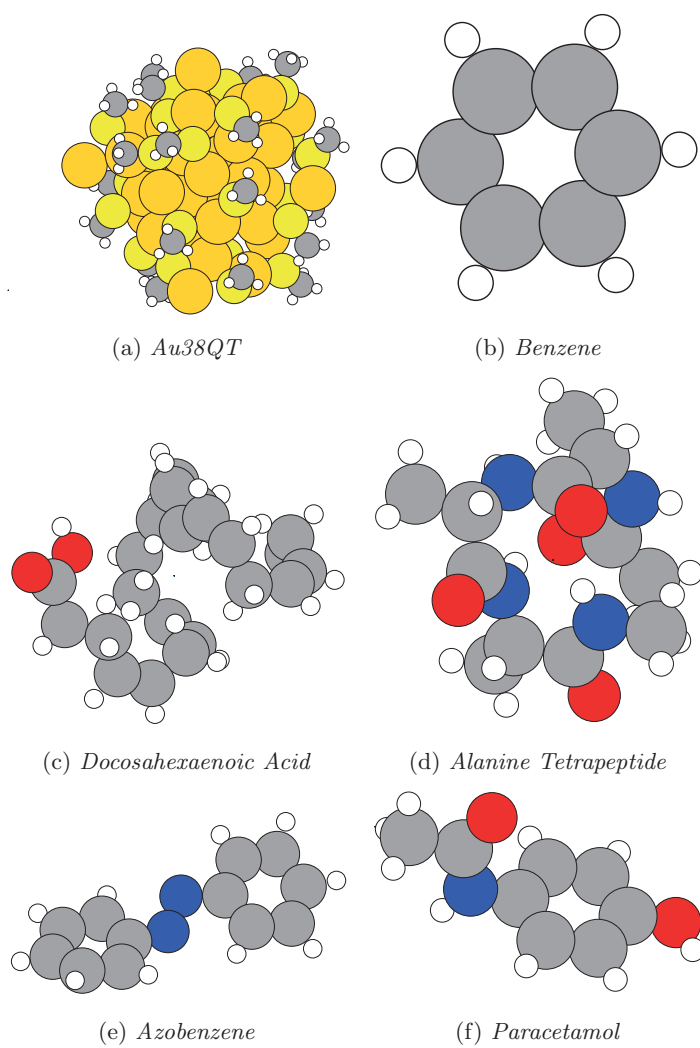


Figure 21: Example figures of the molecular structure of the first six datasets

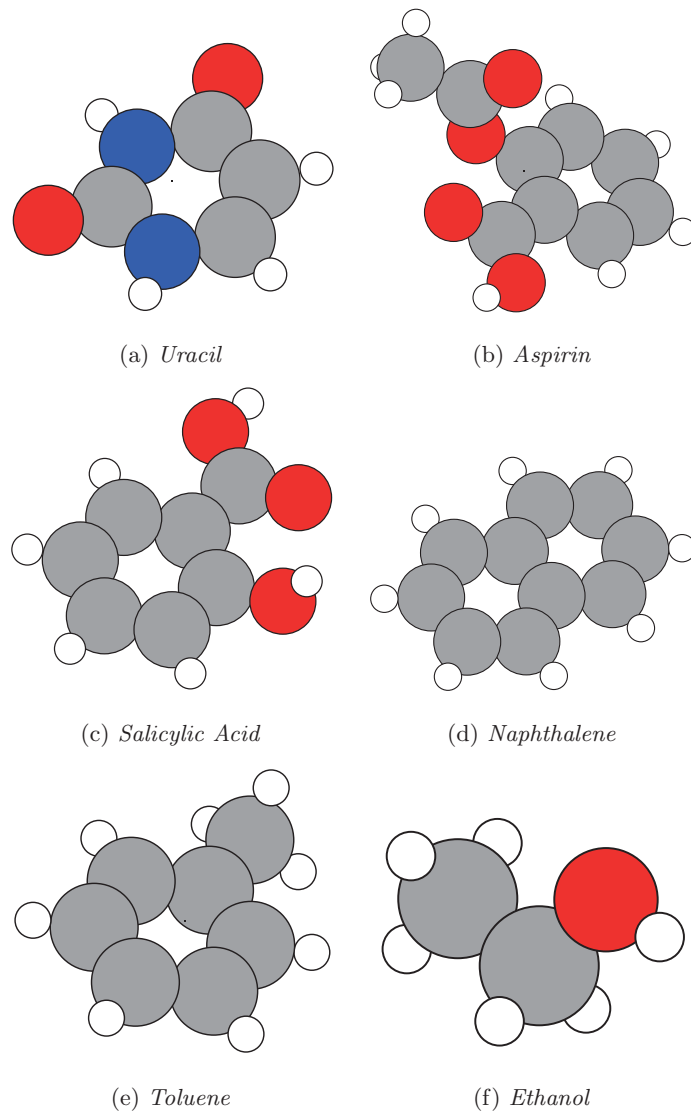


Figure 22: Example figures of the molecular structure of the second six datasets