Hydrostaattinen tasapainoyhtälö yleisessä suhteellisuusteoriassa valkoisen kääpiön sovelluksessa

Kandidaatintutkielma, 27.1.2023

Tekijä:

Antero Voutilainen

Ohjaaja:

Sami Nurmi

JYVÄSKYLÄN YLIOPISTO
FYSIIKAN LAITOS

Tiivistelmä

Voutilainen, Antero
Hydrostaattinen tasapainoyhtälö yleisessä suhteellisuusteoriassa valkoisen kääpiön sovelluksessa
LuK-tutkielma
Fysiikan laitos, Jyväskylän yliopisto, 2023, 64 sivua

Tässä kandidaatin tutkielmassa johdattelen lukijan tutustumaan yleisen suhteellisuusteorian käsitteisiin ja notaatioihin. Tutkielmassa tutustutaan muun muassa tensoreihin, Einsteinin yhtälöihin sekä aika-avaruuden kaarevuuteen ja sitä kuvaavaan metriikkaan. Yleiseen suhteellisuusteoriaan tutustumisen lisäksi johdan TOV (Tolmann-Oppenheimer-Volkoff) - yhtälön ratkaisemalla Einstein yhtälöt yleisessä aikainvariantissa, pallosymmetrisessä metriikassa.

Sovellan TOV-yhtälöä myös valkoisten kääpiöiden ja degeneroituneen elektronikaasun tapauksessa ja ratkaisen sen numeerisesti Python-koodilla. Ratkaisuna TOV-yhtälöön saadaan valkoisen kääpiön rakenne, ja useaa tähteä mallintamalla saadaan valkoisen kääpiön massa-säde relaatio, josta pystytään lukemaan valkoisen kääpiön maksimaalinen mahdollinen massa Chandrasekharin rajalla (noin 1.4 $M_\odot$, jossa $M_\odot$ tarkoittaa Auringon massa). Tämän lisäksi vertailen relativistisen ja epärelativistisen hydrostaattisen tasapainoyhtälön eroa massa-säde relaation avulla sekä laskemalla eri teorioiden tiheysprofiilien välisen suhteen.

Avainsanat: opinnäyte, astrofysiikka, suhteellisuusteoria, kosmologia, TOV-yhtälö, valkoinen kääpiö

Abstract

Voutilainen, Antero
Hydrostatic equilibrium equation in general relativity in the white dwarf application
Bachelor's thesis
Department of Physics, University of Jyväskylä, 2023, 64 pages.

In this bachelor's thesis, I introduce the reader to the concepts and notations of general relativity. In the thesis, I will present, among other things, tensors, Einstein's equations and the curvature of space-time and the metric that describes it. In addition to familiarizing myself with the general theory of relativity, I derive the TOV (Tolmann-Oppenheimer-Volkoff) equation by solving the Einstein equations in the general time-invariant, spherically symmetric metric.

I apply the TOV equation also in the case of white dwarfs and degenerate electron gas and solve it numerically with a Python code. As a solution to the TOV equation, the mass, pressure and density profile of the white dwarf is obtained. By modeling several stars, the mass-radius relation of the white dwarf is obtained, from which the maximum possible mass of the white dwarf at the Chandrasekhar limit (approximately 1.4 $M_{\odot}$, where $M_{\odot}$ is the Solar mass) can be read. In addition to this, the difference between the relativistic and non-relativistic hydrostatic equilibrium equations are compared using the mass-radius relation and by calculating the ratio between the density profiles of different theories.

Keywords: thesis, astrophysics, theory of relativity, cosmology, TOV-equation, white dwarf

Esipuhe

Haluan kiittää perhettäni, sekä ystäviäni yliopistolla ja muualla saamastani tuesta sekä kuuntelevien korvien lainaamisesta tämän kandidaatin tutkielman opinnäytetyön loppuunsaattamiseksi. Erityisesti haluan myös kiittää Olli Väisästä, joka osasi antaa arvokkaita vinkkejä numeriikan onnistumisen kannalta sekä ohjaajaani Sami Nurmea kärsivällisyydestä ja loistavasta ohjauksesta työn hitaasta etenemisestä huolimatta!

Jyväskylässä 27.1.2023

Antero Voutilainen

Sisällys

1 Johdanto

Einsteinin vuonna 1905 julkaisema suppea suhteellisuusteoria ja vuonna 1915 julkaisema yleinen suhteellisuusteoria ovat hyvin tunnettuja fysiikan teorioita, jotka perustuvat ns. koko fysiikan keskeisimpään periaatteeseen: suhteellisuusperiaatteeseen. Suppea, toiselta nimeltään erityinen, suhteellisuusteoria käsittelee inertiaalikoordinaatistoja (staattisella nopeudella kulkeva koordinaatisto - ei kiihtyvyyttä) sekä niiden kautta havaittavaa liikettä ja voimia. Fysiikan nykykäsitys ajasta ja paikasta laakeassa avaruudessa perustuu suppeaan suhteellisuusteoriaan. [1]

Tässä tutkielmassa perehdytään kuitenkin syvemmin yleiseen suhteellisuusteoriaan sekä sen sisältämiin notaatioihin, termeihin sekä fysikaaliseen merkitykseen. Yleinen suhteellisuusteoria (eng. General relativity, GR) selittää, kuinka massa kytkeytyy painovoimaan Einsteinin yhtälöiden mukaan ja aiheuttaa neliulotteisen aika-avaruuden kaarevuuden. Yleinen suhteellisuusteoria tunnetaan siis yleisenä gravitaatioteoriana ja se on luonut pohjan nykyajan kosmologialle sekä avaruuden tutkimukselle nykyaikaisin metodein. [2]

Yleinen suhteellisuusteoria sisältää runsaasti matemaattista formalismia [3], jota esittelen tässä tutkielmassa ennen varsinaiseen aiheeseen siirtymistä. Matemaattisen formalismin esittelyn jälkeen siirryn avaruuden kaarevuuteen, metriikkaan, monistoihin, Einsteinin yhtälöihin sekä muihin yleisestä suhteellisuusteoriasta tuttuihin käsitteisiin. Lopulta ratkaisen Einsteinin yhtälöt analyyttisesti yleisessä, aikainvariantissa ja pallosymmetrisessä metriikassa, josta saadaan Einsteinin yhtälöiden ratkaisuksi relativistinen hydrostaattinen tasapainoyhtälö, Tolman-Oppenheimer-Volkof-yhtälö (TOV-yhtälö). Tämän lisäksi palaan hieman takaisin ja johdan myös newtonilaisen teorian hydrostaattiselle tasapainoyhtälölle TOV:n avulla. Hydrostaattista tasapainoyhtälöä voidaan siis soveltaa erinäisiin astrofysikaalisiin kappaleisiin, kuten tähtiin.

Lopuksi sovellan hydrostaattista tasapainoyhtälöä tähtityyppiin nimeltä valkoinen kääpiö. Sovelluksessa ratkaisen TOV:n numeerisesti degeneroituneen elektronikaasun tilanyhtälöllä, josta valkoiset kääpiöt koostuvat, ja josta saadaan ratkaisuna valkoisen kääpiön massa-, paine- ja energiatiheysprofiilit. Energiatiheysprofiilin ratkaisun avulla vertaan relativistista sekä newtonilaista teoriaa keskenään. Yhden tähden mallintamisen lisäksi mallinnan tähtityyppiä usealla eri keskipisteen energiatiheydellä, josta saan ratkaistua massa-säde relaation sekä havainnollistettua valkoisten kääpiöiden luonteen olevan pääosin epärelativistinen ja teorioiden erojen olevan hyvin pieniä, mutta merkityksellisiä tähden maksimimaalisen mahdollisen massan määrittelevän Chandrasekharin rajan (noin 1.4 $M_\odot$, jossa $M_\odot$ tarkoittaa Auringon massa) lähialueella [4]. Valkoisten kääpiöiden energiatiheyden ratkaisuiden suhdetta vertailemalla huomaan myös, ettei Chandrasekharin rajan lähistöllä teorioiden antamien ratkaisuiden suhde ole täysin triviaali.

2 Suhteellisuusteorian perusteet

Tässä osiossa esittelen yleisen suhteellisuusteorian matemaattisen formalismin perustaa sekä yleisen suhteellisuusteorian antamat Einstein-yhtälöt, jotka kuvaavat avaruuden kaarevuutta ja painovoimaa. Osio perustuu vahvasti lähteeseen [3]. Tämän lisäksi syventävää materiaalia löytyy lähteestä [5] sekä yksinkertaistettua teoriaa löytyy lähteestä [6]. Alkuperäinen Einsteinin julkaisu: ”The Foundation of the General Theory of Relativity” on saatavilla lähteestä [7].

2.1 Notaatiot

Mainitsen tässä vaiheessa lyhyesti yleisesti suhteellisuusteoriassa puhuttavasta notaatiosta: Einsteinin notaatiosta. Einsteinin notaatio on lyhennys summamerkinnästä ja tarkoittaa sitä, että mikäli jokin indeksi toistuu, niin sen yli summataan seuraavasti riippuen, onko indeksi kreikkalainen vai latinalainen:

$$
\begin{aligned}
A_{\ 0}B^0 + A_{\ 1}B^1 + A_{\ 2}B^2 + A_{\ 3}B^3 = \sum_{\nu=0}^{3} A_{\ \nu}B^\nu \equiv A_{\ \nu}B^\nu \quad ; \quad \alpha, \beta, \gamma, \ldots = 0, 1, 2, 3 \\
A_{\ 1}B^1 + A_{\ 2}B^2 + A_{\ 3}B^3 = \sum_{i=1}^{3} A_{\ i}B^i \equiv A_{\ i}B^i \quad ; \quad a, b, c, \ldots = 1, 2, 3 \qquad .
\end{aligned}
$$

Yleensä suhteellisuusteorian laskuissa käytetään myös yleensä konventiota, jossa redusoitu Planckin vakio ja valonnopeus on asetettu arvoon yksi, $\hbar = c = 1$. Einsteinin summaussäännön lisäksi on hyvä esitellä suhteellisuusteoriassa esiintyvien geometristen objektien, kuten vektoreiden, notaatioita. Euklidisessa avaruudessa vektorit määritellään yksinkertaisimmillaan pisteenä tai ”nuolena” karteesisessa koordinaatistossa. Tämä määritelmä ei yksin riitä yleisen suhteellisuusteorian pohjalle, joten määritellään vektorit, dualivektorit sekä tensorit tarkemmalla geometrisella määritelmällä Minkowskin (merkitään M) avaruudessa, josta on myöhemmin helppo siirtyä muihin yleisempiin aika-avaruuksiin, toisin sanoen monistoihin. Monisto on aika-avaruus, joka voidaan jakaa pieniin palasiin ja kartoittaa R^n - avaruuksiin sekä yhdistää jatkuvalla differentioituvalla tavalla. [3]

Vektorit on tärkeää määritellä koordinaateista riippumattomalla tavalla. Määritellään vektorit siis Minkowskin avaruudessa jonkin mielivaltaisen parametrisoidun jatkuvan käyrän suuntaisderivaattana. Olkoon käyrä

$$c : \mathbb{R} \to M, \qquad c(\lambda) \in M, \tag{1}$$

jossa c on määritelty käyrä ja λ käyräparametri. Käyrän c avulla voidaan nyt määritellä vektorioperaattori

$$v = \frac{d}{d\lambda} = \frac{dx^{\mu}(c(\lambda))}{d\lambda}\frac{\partial}{\partial x^{\mu}} \equiv v^{\mu}\frac{\partial}{\partial x^{\mu}} = v^{\mu}\widehat{e}_{(\mu)}, \tag{2}$$

jossa v^{μ} ovat vektorin komponentit ja $\widehat{e}_{(\mu)}$ ovat kantavektorit. Vektori v on siis operaattori, joka operoi funktioihin $v(f) : M \to \mathbb{R}$ ja antaa tällöin sille annetun funktion f derivaatan käyrää $c(\lambda)$ pitkin. Vektori on nyt itsessään invariantti Lorentz-muunnoksissa, mutta sen kantavektorit eivät ole. Kuitenkin kantavektorit voidaan muuntaa normaalisti Lorentz-muunnoksen avulla riippumatta siitä, mitkä vektorin numeeriset komponentit ovat. Vektorit siis muuntuvat Lorentz-muunnoksessa seuraavanlaisesti:

$$x^{\mu'} = \Lambda^{\nu}{}_{\mu'}x^{\nu} = x^{\nu}, \qquad \widehat{e}_{(\nu')} = \Lambda^{\mu}{}_{\nu'}\widehat{e}_{(\mu)} \tag{3}$$

Määritellään vektoreiden lisäksi duaalivektorit.

$$\omega = \omega_{\mu}dx^{\mu} = \omega_{\mu}\widehat{\theta}^{(\mu)}, \tag{4}$$

jossa $dx^{\mu} = \widehat{\theta}^{(\mu)}$ on duaalikantavektorit ja ω_{μ} on duaalivektorin komponentit. Duaalivektori muuntuu Lorentz-muunnoksissa seuraavanlaisesti

$$\omega_{\mu'} = \Lambda^{\nu}{}_{\mu'}\omega_{\nu}, \qquad \widehat{\theta}^{(\rho')} = \Lambda^{\rho}{}_{\sigma'}\widehat{\theta}^{(\sigma)}. \tag{5}$$

Tensorit ovat matemaattisia olioita, jotka kuvaavat esimerkiksi voimia, geometrioita tai vastaavia suureita. Olennaisesti tensorit suhteellisuusteoriassa ovat multilineaarisia kuvauksia kollektiivista k duaalivektoreista ja l vektoreista R:nään. Tensorin (k, l) komponentit koordinaattikannassa saadaan operoimalla tensoria kantavektoreilla ja -

duaaleilla [3]

$$T^{\mu_1 \cdots \mu_k}{}_{\nu_1 \cdots \nu_l} = T(dx^{\mu_1}, \cdots, dx^{\mu_k}, \partial_{\nu_1}, \cdots, \partial_{\nu_l}). \tag{6}$$

Intuitiivisesti vektoreita voidaan ajatella normaaleina sarakevektoreina ja duaalivektoreita rivivektoreina. Tensorit ovat puolestaan intuitiivisesti moniulotteisia taulukoita. Vektoreista päästään duaaleihin ja takaisin operoimalla niitä metriikalla:

$$g^{\mu\nu} A_\nu = A^\mu, \qquad g_{\mu\nu} B^\nu = B_\mu, \tag{7}$$

jossa $g_{\mu\nu}$, $g^{\mu\nu}$ on metriikka ja käänteismetriikka sekä A on mielivaltainen duaali ja B vektori. [3]

2.2 Avaruuden kaarevuus ja formalismi

Newtonilaisessa fysiikassa tiedetään, että jonkin kappaleen gravitaationaali- ja inertiaalimassat ovat samat, $m_i = m_g$. Tämä periaate tunnetaan Heikkona Yhtäsuuruus-Periaatteena, tai toisin sanoen WEP:inä (eng. Weak Equivalence Principle, WEP) [3]. Newtonin toinen laki sanoo, että johonkin kappaleeseen kohdistettu voima on suoraan verrannollinen kappaleen inertiaalimassaan ja kiihtyvyyteen

$$F = m_i a. \tag{8}$$

Inertiaalimassa on olennaisesti vastus, joka koetaan kun kappaleeseen kohdistetaan voima. Nyt Newtonin gravitaatiolaki voidaan kirjoittaa gravitaatiopotentiaalikentän, Φ, gradientin avulla

$$F_g = -m_g \nabla \Phi, \tag{9}$$

josta voimme nähdä suoraan WEP:in nojalla, että vapaasti putoavalle mielivaltaiselle kappaleelle sen kiihtyvyys on gravitaatiopotentiaalikentän gradientti

$$a = -\nabla \Phi. \tag{10}$$

Tästä voidaan päätellä aika-avaruudessa olevan käyriä, joita pitkin mielivaltaiset kiihdyttämättömät kappaleet kulkevat. Näitä käyriä kutsutaan inertiaaleiksi (tai "vapaasti putoaviksi") liikeradoiksi. Myöhemmin näitä liikeratoja kutsutaan myös geodeeseiksi, joka on olennaisesti määritelmä lyhyimmälle matkalle kahden pisteen välille aika-avaruudessa. Tästä päästään suoraan päätelmään, että massa kaareuttaa aika-avaruutta ja tämä kaarevuus tunnetaan gravitaationa. [3]

Kaareutunutta aika-avaruutta kuvataan monistoilla. Differentioituvat monistot ovat yksi fundamentaaleista matemaattisista ja fysikaalisista konsepteista. Yleensä on totuttu analysoimaan n- dimensioista Euklidista avaruutta, $\mathbb{R}^n$ ja sen ominaisuuksia, kuten laakeaa ja positiivisesti määriteltyä n-dimensioista metriikkaa komponenteilla δ_{ij}. Euklidisten avaruuksien lisäksi on kuitenkin olemassa muitakin avaruuksia, kuten pallo tai torus, jotka käsitetään "kaareutuneiksi" tai muuten topologisesti monimutkaisiksi avaruuksiksi joihin halutaan käyttää tunnettuja työkaluja. [3] [5]

Monisto määritellään siten että monisto vastaa avaruutta, joka voi olla kaareutunut tai topologisesti monimutkainen, mutta lokaaleissa alueissa se näyttää $\mathbb{R}^n$:ltä sekä on jatkuvasti differentioituva. Monistot siis konstruoidaan sitomalla yhteen näitä avaruuden lokaaleja osia. Ehtona tälle on se, että lokaalien osien dimensiot vastaavat toisiaan, jolloin monisto on n-dimensioinen. Esimerkkejä monistoista on muun muassa: $\mathbb{R}^n$, n-pallo (S^n) sekä n-Torus (T^n). [3] [5]

Matemaattisesti monisto ei kaipaa metriikkaa, mutta metriikka on kuitenkin määritelty moniston ylimääräiseksi rakenteeksi kuvaamaan sen geometriaa. Yleisessä suhteellisuusteoriassa käsitellään monistoja, jotka on varustettu metriikalla, joka kuvaa gravitaatiota. Metriikka määritellään symmetrisenä (0,2) tensorina

$$g = g_{\mu\nu} dx^{\mu} \otimes dx^{\nu} = \begin{pmatrix} g_{00} & g_{01} & g_{02} & g_{03} \\ g_{10} & g_{11} & g_{12} & g_{13} \\ g_{20} & g_{21} & g_{22} & g_{23} \\ g_{30} & g_{31} & g_{32} & g_{33} \end{pmatrix}, \tag{11}$$

jolle pätee

$$\det(g_{\mu\nu}) \neq 0, \tag{12}$$

ja

$$g^{\mu\nu} g_{\mu\nu} = g_{\lambda\sigma} g^{\lambda\sigma} = \delta^{\mu}_{\sigma}, \tag{13}$$

jossa $g^{\mu\nu}$ on käänteismetriikka ja δ^μ_σ on Kroneckerin delta, jolle pätee $\delta^\mu_\sigma = 1$, kun $\sigma = \mu$, muulloin se on nolla. Metriikalla on yleisessä suhteellisuusteoriassa tärkeä rooli. Metriikalla on monia sovelluksia, joita esimerkiksi lueteltuna: Metriikka antaa menneisyyden ja nykyisyyden käsitteen; Metriikan avulla voidaan laskea matkan pituus ja ominaisaika; Metriikka määrittelee lyhyimmän matkan kahden pisteen välillä, geodeesit, ja näin ollen testihiukkasen liikkeen; Metriikkka korvaa newtonisen gravitaatiokentän Φ; Metriikka antaa lokaalin inertiaalikehyksen käsitteen ja sitä myöten pyörimättömyyden tunteen; Metriikka määrittelee kausaliteetin määrittämällä valonnopeuden suurimmaksi nopeudeksi, millä mikään signaali voi kulkea; Metriikka korvaa Newtonin mekaniikan Euklidisen kolmiulotteisen pistetulon. [3] Metriikka antaa pituuden määritelmän halutussa aika-avaruudessa. Usein tämä esitetään viivaelementin avulla

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu, \tag{14}$$

joka voidaan esittää esimerkiksi ”normaalissa” laakeassa aika-avaruudessa Minkowskin metriikkana:

$$ds^2 = dt^2 + dx^2 + dy^2 + dz^2 \equiv \mathrm{diag}(-1, 1, 1, 1) = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \eta_{\mu\nu}. \tag{15}$$

Teknisesti metriikka ja viivaelementti eivät ole tasan sama asia, mutta se on kuitenkin yleisesti hyväksyttyä. Viivaelementti voi näyttää hyvin erilaiselta Minkowskin metriikkaan verrattuna riippuen siitä millaisessa avaruudessa liikutaan sekä siitä, että millaisia approksimaatioita tehdään laskujen ratkaisemiseksi. Metriikan ohella yleisen avaruuden kaarevuutta kuvaa koordinaateista riippumattomat Riemannin kaarevuustensorit. [3] Aika-avaruuden kaarevuuden todennuksen jälkeen sekä intuitiivisen ajatuksen luomisen jälkeen kaarevuudesta on mahdollista formalisoida edellä esitetyt konseptit kuvaamaan aika-avaruutta. Olennaisimmat termit aika-avaruuden kuvaamiseen ovat Christoffelin konnektio, kovariantti derivaatta sekä Riemannin tensori. Näiden lisäksi olennainen osa formalismia ovat geodeeesit, joita ei kylläkään tässä tutielmassa käsitellä. [3] Kaarevan avaruuden kaikki ilmenemismuodot liittyvät vahvasti termiin nimeltä konnektio. Konnektio yleisesti antaa käsityksen siitä, kuinka viereisten tangenttiavaruuksien vektoreita relatoidaan keskenään. Christoffelin symboli (käytetään myös termiä Christoffelin konnektio) saadaan metriikasta laskemalla

$$\Gamma^{\lambda}_{\mu\nu} = \frac{1}{2} g^{\lambda\sigma}\Big(\partial_{\mu} g_{\nu\sigma} + \partial_{\nu} g_{\sigma\mu} - \partial_{\sigma} g_{\mu\nu}\Big). \tag{16}$$

Vaikka Christoffelin symboli näyttää tensorilta, sitä se ei ole. Tämän vuoksi sitä sanotaan ”objektiksi” tai ”symboliksi”. Fundamentaalisti konnektioita käytetään kovariantin derivaatan (∇_{μ}) määrittelemiseen ja laskemiseen. Kovariantti derivaatta on olennaisesti yleistys osittaisderivaatasta. Vektorikentän kovariantti derivaatta saadaan laskemalla

$$\nabla_{\mu} V^{\nu} = \partial_{\mu} V^{\nu} + \Gamma^{\nu}_{\mu\sigma} V^{\sigma}. \tag{17}$$

Muunlaisten tensoreiden kovariantit derivaatat saadaan samantyylisillä ilmaisuilla. Konnektio ilmaantuu myös geodeesiyhtälössä, joka on yleisilmaus lyhyimmälle matkalle kahden pisteen välillä. Lopulta viimeinen tarvittava tekninen esitys kaarevuudelle sisältyy Riemannin tensoriin, (1, 3) tensori, joka saadaan konnektioiden avulla laskemalla

$$R^{\rho}_{\sigma\mu\nu} = \partial_{\mu} \Gamma^{\rho}_{\nu\sigma} - \partial_{\nu} \Gamma^{\rho}_{\mu\sigma} + \Gamma^{\rho}_{\mu\lambda} \Gamma^{\lambda}_{\nu\sigma} - \Gamma^{\rho}_{\nu\lambda} \Gamma^{\lambda}_{\mu\sigma}. \tag{18}$$

Kaikki tarvittava tieto avaruuden (tai moniston) kaarevuuteen liittyen saadaan Riemannin tensorista. Tensori häviää vain silloin, kun metriikka on täydellisen laakea. Suhteellisuusteorian Einsteinin yhtälöt puolestaan yhdistävät Riemannin tensorin tietyt komponentit energia-liikemäärä tensoriin. [3]

2.3 Einsteinin yhtälöt

Einsteinin yhtälöt ovat yksiä tärkeimpiä yhtälöitä suhteellisuusteoriassa, sillä Einsteinin yhtälöt kuvaavat sitä, miten metriikka kytkeytyy energiaan ja liikemäärään. Näillä Einsteinin yhtälöillä voidaan ratkaista esimerkiksi massa-energia-jakauma tietynlaisessa avaruudessa tai päinvastoin, millainen avaruus seuraa tietystä massajakaumasta. Einsteinin yhtälöt voidaan johtaa monella tavalla. [3, 7] Einstein itse johti yhtälöt etsimällä korvaajan Newtonin potentiaalin Poissonin yhtälölle

$$\nabla^2\Phi = 4\pi G\rho. \tag{19}$$

Käyttämällä tietoa, että uudessa yhtälössä täytyy olla jokin tensori, joka koostuu metriikan toisista derivaatoista ja on suoraan verrannollinen energia-liikemäärä tensoriin

$$[\nabla^2 g]_{\mu\nu} \propto T_{\mu\nu}, \tag{20}$$

voidaan valita metriikan kenttäyhtälöksi arvaus

$$G_{\mu\nu} = \kappa T_{\mu\nu}. \tag{21}$$

Tästä ratkaisemalla vakion κ arvon sekä tarkastamalla antaako arvaus Newtonisen gravitaation yhtälön Newtonilaisella rajalla voidaan johtaa Einsteinin yhtälöt

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2}Rg_{\mu\nu} = 8\pi G T_{\mu\nu}, \tag{22}$$

jossa G on Newtonin gravitaatiovakio, $R_{\mu\nu}$ on Riccin tensori, R on Riccin skalaari ja $T_{\mu\nu}$ on energia-liikemäärä tensori. Riccin tensori saadaan kontraktoimalla Riemannin tensorin, yhtälö (18), ylin indeksi keskimmäisen alaindeksin kanssa, jolloin saadaan Riccin tensorille yhtälö

$$R_{\mu\nu} = R^{\lambda}_{\mu\lambda\nu} = \partial_\lambda \Gamma^{\lambda}_{\nu\mu} - \partial_\nu \Gamma^{\lambda}_{\lambda\mu} + \Gamma^{\lambda}_{\lambda\sigma}\Gamma^{\sigma}_{\nu\mu} - \Gamma^{\lambda}_{\nu\sigma}\Gamma^{\sigma}_{\lambda\mu}. \tag{23}$$

Riccin tensori on siis $4x4$ - matriisi, joten sillä on yhteensä 16 komponenttia. Riemannin tensorille pätee symmetria ensimmäisen ja kolmannen alaindeksin vaihdon suhteen

$$R^{\lambda}_{\mu\lambda\nu} = R^{\lambda}_{\nu\lambda\mu} \Rightarrow R_{01} = R_{10}, \tag{24}$$

joten lopulta Riccin tensorin laskemiseen riittää kymmenen komponentin laskeminen. Yksi Riccin tensorin komponentti lasketaan Einsteinin summaussäännön (osio 2.1) mukaan

$$R^{\lambda}_{\mu\lambda\nu} = R^{0}_{\mu0\nu} + R^{1}_{\mu1\nu} + R^{2}_{\mu2\nu} + R^{3}_{\mu3\nu}. \tag{25}$$

Riccin skalaari saadaan ottamnalla jälki Riccin tensorista metriikan suhteen

$$R = g^{\mu\nu} R_{\mu\nu}. \tag{26}$$

Nämä yhtälöt kertovat, miten aika-avaruus kytkeytyy energia-liikemäärään. Toinen, modernimpi tapa johtaa Einsteinin yhtälöt on aktion ja yleisten liikeyhtälöiden avulla. Energia-liikemäärä -tensori on tensorillinen määre fysiikassa, joka kuvaa energian ja liikemäärän vuota aika-avaruudessa. Lisää tietoa aktiosta ja Riccin tensorista esimerkiksi lähteissä [3, 8].

3 Hydrostaattiset tasapainoyhtälöt

Tavoitteena tässä osiossa on ratkaista Einsteinin yhtälöt pallosymmetrisessä aikainvariantissa metriikassa. Tuloksena ratkaisusta saadaan relativistinen hydrostaattinen tasapainoyhtälö tähdille, toisin sanoen Tolman-Oppenheimer-Volkoff -yhtälö (TOV-yhtälö). Osioon on otettu mallia lähteestä [3].

Ratkaisua varten lasketaan Riccin tensori ja Riccin skalaari, joiden avulla määritetään Einsteinin tensori. Tämän jälkeen aineelle oletetaan ideaalifluidin energia-liikemäärä tensori sekä ratkaistaan Eisnteinin yhtälöistä (22) saatava neljän yhtälön yhtälöryhmä.

Osiossa pyrin selittämään TOV:n johdon mahdollisimman ymmärrettävästi. Tämän nojalla osioon sisällytän jonkin verran esimerkkilaskuja, joista osa on laskettu käsin. Suurimman osan välituloksista kuitenkin laskin myös symbolisesti laskentaohjelmalla. Välituloksia vertasin myös kirjallisuudesta [3] löytyviin tuloksiin ja yhtälöihin.

Lopuksi johdan vielä epärelativistisen Newtonilaisen hydrostaattisen tasapainoyhtälön TOV:n avulla. Molemmat hydrostaattiset tasapainoyhtälöt on johdettu myös esimerkiksi lähteissä [9, 10].

3.1 Pallosymmetrinen ja aikainvariantti aika-avaruus

Yhtälön (14) mukaan yleisen, aikainvariantin ja pallosymmetrisen metriikan viivaelementti on muotoa [3]

$$ds^2 = -e^{2\alpha(r)}dt^2 + e^{2\beta(r)}dr^2 + r^2 d\Omega^2, \tag{27}$$

jossa $d\Omega^2$ tarkoittaa kulmaosaa

$$d\Omega^2 = d\theta^2 + \sin^2(\theta) d\phi^2, \tag{28}$$

jossa $\alpha(r)$ ja $\beta(r)$ mielivaltaisia säteestä riippuvia funktioita, käytetään myöhemmin myös pelkistettyä α ja β -merkintää. Tästä saadaan siis metriikan komponenteiksi

$$\begin{aligned}
g_{00} &= -e^{2\alpha},\\
g_{11} &= e^{2\beta},\\
g_{22} &= r^2,\\
g_{33} &= r^2 \cdot \sin^2(\theta).
\end{aligned}$$

Tarkennuksena sanottakoon, että indeksien tapauksissa pätee merkinnät

$$\begin{aligned}
0 &= t & &\Rightarrow & \partial_0 &= \partial_t,\\
1 &= r & &\Rightarrow & \partial_1 &= \partial_r,\\
2 &= \theta & &\Rightarrow & \partial_2 &= \partial_\theta,\\
3 &= \phi & &\Rightarrow & \partial_3 &= \partial_\phi.
\end{aligned}$$

Kun etsitään metriikalle ratkaisuja, niin otetaan osiossa 2.3 esitetyt Einsteinin yhtälöt (22) ja ratkaistaan tarvittavat suureet. Lasketaan Riccin tensorin ensimmäinen komponentti, R_{00}, mahdollisimman auki esimerkein. Näin ollen yhtälöistä (23) ja (25) saadaan

$$\begin{aligned}
R_{00} = R^{\lambda}_{0\lambda 0} &= \partial_\lambda \Gamma^{\lambda}_{00} - \partial_0 \Gamma^{\lambda}_{\lambda 0} + \Gamma^{\lambda}_{\lambda\sigma}\Gamma^{\sigma}_{00} - \Gamma^{\lambda}_{0\sigma}\Gamma^{\sigma}_{\lambda 0}\\
&= \left[\partial_0 \Gamma^{0}_{00} - \partial_0 \Gamma^{0}_{00} + \Gamma^{0}_{0\sigma}\Gamma^{\sigma}_{00} - \Gamma^{0}_{0\sigma}\Gamma^{\sigma}_{00}\right] + \left[\partial_1 \Gamma^{1}_{00} - \partial_0 \Gamma^{1}_{10} + \Gamma^{1}_{1\sigma}\Gamma^{\sigma}_{00} - \Gamma^{1}_{0\sigma}\Gamma^{\sigma}_{10}\right]\\
&+ \left[\partial_2 \Gamma^{2}_{00} - \partial_0 \Gamma^{2}_{20} + \Gamma^{2}_{2\sigma}\Gamma^{\sigma}_{00} - \Gamma^{2}_{0\sigma}\Gamma^{\sigma}_{20}\right] + \left[\partial_3 \Gamma^{3}_{00} - \partial_0 \Gamma^{3}_{30} + \Gamma^{3}_{3\sigma}\Gamma^{\sigma}_{00} - \Gamma^{3}_{0\sigma}\Gamma^{\sigma}_{30}\right],
\end{aligned}$$

josta saadaan σ:n yli summaamalla lopulta

$$
\begin{aligned}
R_{00} =& \Big[\partial_0 \Gamma^0_{00} - \partial_0 \Gamma^0_{00} \\
&+ \Big(\Gamma^0_{00}\Gamma^0_{00} + \Gamma^0_{01}\Gamma^1_{00} + \Gamma^0_{02}\Gamma^2_{00} + \Gamma^0_{03}\Gamma^3_{00}\Big) - \Big(\Gamma^0_{00}\Gamma^0_{00} + \Gamma^0_{01}\Gamma^1_{00} + \Gamma^0_{02}\Gamma^2_{00} + \Gamma^0_{03}\Gamma^3_{00}\Big)\Big] \\
&+ \Big[\partial_1 \Gamma^1_{00} - \partial_0 \Gamma^1_{10} \\
&+ \Big(\Gamma^1_{10}\Gamma^0_{00} + \Gamma^1_{11}\Gamma^1_{00} + \Gamma^1_{12}\Gamma^2_{00} + \Gamma^1_{13}\Gamma^3_{00}\Big) - \Big(\Gamma^1_{00}\Gamma^0_{10} + \Gamma^1_{01}\Gamma^1_{10} + \Gamma^1_{02}\Gamma^2_{10} + \Gamma^1_{03}\Gamma^3_{10}\Big)\Big] \\
&+ \Big[\partial_1 \Gamma^1_{00} - \partial_0 \Gamma^1_{10} \\
&+ \Big(\Gamma^1_{10}\Gamma^0_{00} + \Gamma^1_{11}\Gamma^1_{00} + \Gamma^1_{12}\Gamma^2_{00} + \Gamma^1_{13}\Gamma^3_{00}\Big) - \Big(\Gamma^1_{00}\Gamma^0_{10} + \Gamma^1_{01}\Gamma^1_{10} + \Gamma^1_{02}\Gamma^2_{10} + \Gamma^1_{03}\Gamma^3_{10}\Big)\Big] \\
&+ \Big[\partial_1 \Gamma^1_{00} - \partial_0 \Gamma^1_{10} \\
&+ \Big(\Gamma^1_{10}\Gamma^0_{00} + \Gamma^1_{11}\Gamma^1_{00} + \Gamma^1_{12}\Gamma^2_{00} + \Gamma^1_{13}\Gamma^3_{00}\Big) - \Big(\Gamma^1_{00}\Gamma^0_{10} + \Gamma^1_{01}\Gamma^1_{10} + \Gamma^1_{02}\Gamma^2_{10} + \Gamma^1_{03}\Gamma^3_{10}\Big)\Big].
\end{aligned}
$$

Tästä nähdään suoraan, että Christoffelin symboleita on 34, jotka täytyisi laskea. Kuitenkin huomaamalla symmetrian

$$\Gamma^\sigma_{\mu\nu} = \Gamma^\sigma_{\nu\mu} \tag{29}$$

avulla symboleiden määrä vähenee 25:een kappaleeseen. Yhtälöstä (16) saadaan laskettua tarvittavat Christoffelin symbolit. Laskujen helpottamiseksi on hyvä huomata, että metriikan ollessa diagonaalinen

$$g_{\mu\nu} = \mathrm{diag}\Big(- e^{2\alpha}, e^{2\beta}, r^2, r^2 \sin^2(\theta)\Big), \tag{30}$$

sen käänteismetriikka muodostuu diagonaalikomponenttien käänteisluvuista

$$g^{\mu\nu} = \mathrm{diag}\Big(\frac{1}{-e^{2\alpha}}, \frac{1}{e^{2\beta}}, \frac{1}{r^2}, \frac{1}{r^2 \sin^2(\theta)}\Big). \tag{31}$$

Tällöin Christoffelin symbolin yhtälössä (16) esiintyvä σ summautuu $\{0, 1, 2, 3\}$, mutta nyt σ saa saman arvon kuin λ, koska kaikki muut käänteismetriikan komponentit ovat nollia, $g^{\mu\nu} = 0$, kun $\mu \neq \nu$. Yhtälöstä (16) saadaan siis laskettua konnektiot seuraavasti

$$\begin{aligned}
\Gamma^0_{10} &= \frac{1}{2} g^{00}\Big(\partial_1 g_{00} + \partial_0 g_{01} - \partial_0 g_{10}\Big) \\
&= \frac{1}{2} g^{00} \partial_1 g_{00} \\
&= \frac{1}{2} \frac{1}{-e^{2\alpha}} - 2e^{2\alpha} \partial_1 \alpha \\
&= \partial_1 \alpha(r).
\end{aligned}$$

Vastaavasti laskemalla huomataan, että suurin osa konnektioista on triviaaleja ja lopulta ainoat vaadittavat ei-triviaalit konnektiot Riccin tensorin laskuun ovat

$$\begin{aligned}
\Gamma^0_{10} &= \partial_1 \alpha, & \Gamma^1_{00} &= e^{2\alpha - 2\beta} \partial_1 \alpha, \\
\Gamma^1_{11} &= \partial_1 \beta, & \Gamma^1_{22} &= -\frac{r}{e^{2\beta}}, \\
\Gamma^1_{33} &= -\frac{r \sin^2(\theta)}{e^{2\beta}}, & \Gamma^2_{21} &= \Gamma^3_{31} = \frac{1}{r}, \\
\Gamma^2_{33} &= -\cos(\theta) \sin(\theta), & \Gamma^3_{32} &= \cot(\theta).
\end{aligned} \tag{32}$$

Nyt Riccin tensorin ensimmäisen komponentin lasku sievenee huomattavasti triviaalien konnektioiden supistuessa pois muotoon

$$\begin{aligned}
R_{00} &= \partial_1 \Gamma^1_{00} + \Gamma^1_{11} \Gamma^1_{00} - \Gamma^1_{00} \Gamma^0_{10} + \Gamma^2_{21} \Gamma^1_{00} + \Gamma^3_{31} \Gamma^1_{00} \\
&= e^{2\alpha - 2\beta} \Big[(\partial_1 \alpha)^2 + 2r \partial_1 \alpha - \partial_1 \alpha \partial_1 \beta + \partial_1^2 \alpha \Big]
\end{aligned}$$

Loput Riccin tensorin ei-triviaalit komponentit saadaan vastaavasti:

$$\begin{aligned}
R_{11} &= -(\partial_1 \alpha)^2 + \frac{2\partial_1 \beta}{r} + \partial_1 \alpha \partial_1 \beta - \partial_1^2 \alpha, \\
R_{22} &= \frac{1}{e^{2\beta}} \Big[e^{2\beta} - r\partial_1 \alpha + r \partial_1 \beta - 1 \Big], \\
R_{33} &= \frac{\sin^2(\theta)}{e^{\beta}} \Big[e^{2\beta} - r\partial_1 \alpha + r \partial_1 \beta - 1 \Big].
\end{aligned}$$

Tästä on helppo huomata, että Riccin tensori on diagonaalinen ja tällöin Riccin skalaari saadaan yhtälön (26) avulla muotoon

$$R = \frac{2}{r^2 e^{\beta}} \Big[e^{2\beta} - r^2 (\partial_1 \alpha)^2 + 2r \partial_1 \beta + r \partial_1 \alpha \Big(r \partial_1 \beta - 2 \Big) - r^2 \partial_1^2 \alpha - 1 \Big].$$

Tarvittavien tulosten avulla voidaan siirtyä ratkaisemaan Einsteinin yhtälöitä (22).

3.2 Tolman-Oppenheimer-Volkoff -yhtälö

Nyt Tolman-Oppenheimer-Volkoff -yhtälön johtoa varten ratkaistu Riccin tensori ja Riccin skalaari voidaan sijoittaa metriikan kanssa Einsteinin yhtälöihin (22), josta saadaan Einsteinin tensori. Einsteinin yhtälöiden (22) oikealle puolelle sijoitetaan ideaalifluidin määräämä energia-liikemäärä tensori, jolloin saadaan neljän yhtälön yhtälöryhmä, joka lopulta voidaan ratkaista.

Mallinnettaessa tähteä, sen voidaan olettaa koostuvan ideaalifluidista tietyssä elämänkaaren vaiheessa, jolloin Einsteinin yhtälöissä (22) esiintyvä energia-liikemäärä -tensori määräytyy yhtälön

$$T_{\mu\nu} = (\rho + p)U_\mu U_\nu + p g_{\mu\nu} \tag{33}$$

mukaan [3]. Yhtälössä $\rho(r) = \rho$ on tähden energiatiheys säteen funktiona, $p(r) = p$ on tähden sisäinen paine säteen funktiona ja U_μ on nelinopeus valittuna ajanlaatuiseen suuntaan. Nelinopeuden määritelmistä lepokoordinaatistossa

$$g_{\mu\nu} U^\mu U^\nu = -1, \tag{34}$$

jossa $U^\mu = (1, 0, 0, 0)$, seuraa, että nelinopeus saa muodon

$$\Rightarrow U_\mu = (e^\alpha, 0, 0, 0). \tag{35}$$

Nyt energia-liikemäärä -tensorin komponentit saadaan laskettua suoraan yhtälöstä (33):

$$\begin{aligned} T_{00} &= (\rho + p)U_0 U_0 + p g_{00} \\ &= (\rho + p)e^{2\alpha} + p(-e^{2\alpha}) \\ &= \rho e^2 \alpha. \end{aligned}$$

Vastaavasti muut komponentit laskemalla saadaan energia-liikemäärä -tensoriksi

$$T_{\mu\nu} = \mathrm{diag}(\rho e^{2\alpha}, p e^{2\beta}, p r^2, p r^2 \sin^2(\theta)). \tag{36}$$

Tämän jälkeen sijoitetaan Riccin tensori, Riccin skalaari, metriikka ja energia-liikemäärä tensori Einsteinin yhtälöihin (22)

$$R_{\mu\nu} - \frac{1}{2}Rg_{\mu\nu} = 8\pi G T_{\mu\nu}, \tag{37}$$

josta saadaan kolme toisistaan riippumatonta yhtälöä komponenteille $tt, rr, \theta\theta$ sekä $\theta\theta$-yhtälöstä riippuva $\phi\phi$. Einsteinin yhtälöistä (22) saadaan siis yhtälöryhmä

$$\begin{cases} \text{tt}: & \frac{1}{r^2 e^{2\beta}}\Big[e^{2\beta} + 2r\partial_1\beta - 1\Big] = 8\pi G\rho, & (38) \\ \text{rr}: & \frac{1}{r^2 e^{2\beta}}\Big[-e^{2\beta} + 2r\partial_1\alpha + 1\Big] = 8\pi G p, & (39) \\ \theta\theta: & \frac{1}{e^{2\beta}}\Big[(\partial_1\alpha)^2 + \frac{\partial_1\alpha - \partial_1\beta}{r} - \partial_1\alpha\partial_1\beta + \partial_1^2\alpha\Big] = 8\pi G p, & (40) \end{cases}$$

jossa neljäs $\phi\phi$-yhtälö voidaan jättää huomioitta Einsteinin tensorin komponentin $G_{\phi\phi} = \sin^2(\theta)G_{\theta\theta}$ sekä energia-liikemärä -tensorin $T_{\phi\phi} = \sin^2(\theta)T_{\theta\theta}$ riippuvuuksien vuoksi. Näissä kolmessa yhtälössä on neljä vapausastetta, $\alpha(r)$, $\beta(r)$, $\rho(r)$ ja $p(r)$. Tästä voidaan huomata, että tt-komponentin yhtälö riippuu vain $\beta(r)$ ja $\rho(r)$ funktioista.

Lähdetään eliminoimaan vapausasteita määrittelemällä funktio $m(r)$, jolla korvataan $\beta(r)$ siten, että

$$m(r) \equiv \frac{r - re^{-2\beta})}{2G} \Rightarrow e^{2\beta} = \left[1 - \frac{2Gm(r)}{r}\right]^{-1}, \tag{41}$$

jolloin metriikka (27) saadaan muotoon

$$ds^2 = -e^{2\alpha(r)}dt^2 + \left[1 - \frac{2Gm(r)}{r}\right]^{-1}dr^2 + r^2 d\Omega^2. \tag{42}$$

Tästä metriikan muodosta huomataan, että sen rr-komponentti g_{rr} on selvä yleistys Schwarzschildin metriikasta, joka kuvaa aika-avaruutta pallosymmetrisen ulkopuolella ja jolle $m(r) = M =$ vakio. Nyt sijoittamalla yhtälö (41) tt-yhtälöön (38), saadaan se sievennettyä muotoon

$$\frac{dm}{dr} = 4\pi\rho r^2, \tag{43}$$

jota integroimalla energiatiheyden yli säteen suhteen voidaan identifioida $m(r)$-

funktio massaksi

$$m(r) = 4\pi \int_0^r \rho(r')r'^2 dr'. \tag{44}$$

Nyt voidaan kuvitella tähti, jonka säde on R ja jonka jälkeen ollaan Schwarzschildin metriikan kuvaamassa tyhjiössä. Jotta Einsteinin yhtälöiden ratkaisu on fysikaalisesti järkevä, täytyy metriikan tähden sisäpuolella ja ulkopuolella olla yhtenevät, joten tähden rajapinnalla ei saa olla epäjatkuvuuksia. Tästä seuraa, että tähden massa (Schwarzschildin massa) täytyy saada integraalista

$$M = m(R) = 4\pi \int_0^R \rho(r)r^2 dr. \tag{45}$$

Tämä voidaan siis fysikaalisesti tulkita massana jonkin määrätyn (tai mitatun) säteen sisällä.

Tähden massan identifioinnin jälkeen rr-komponentti (39) voidaan kirjoittaa $m(r)$:n (41) avulla muotoon

$$\frac{d\alpha}{dr} = \frac{Gm(r) + 4\pi G r^3 p}{r\Big[r - 2Gm(r)\Big]}. \tag{46}$$

$\theta\theta$-komponentin yhtälöä (40) on epäkäytännöllistä soveltaa suoraan, vaan sovelletaan jatkuvuusyhtälöä viimeisen vapausasteen eliminoimiseksi, jolloin saadaan relaatio α:n ja $p(r)$:n välille. Jatkuvuusyhtälö kertoo, että energia-liikemäärä tensorin kovariantti derivaatta on nolla [3]

$$\nabla_\mu T^{\mu\nu} = 0. \tag{47}$$

Kovariantti derivaatta saadaan johdettua tensorille yhtälön (17) mukaan (johtoa ei käydä tässä tarkemmin läpi), jolloin (0, 2) tensorille kovariantti derivaatta saa lopulta muodon

$$\nabla_\mu T^{\mu\nu} = \partial_\mu T^{\mu\nu} + \Gamma^\mu_{\mu\lambda} T^{\lambda\nu} + \Gamma^\nu_{\mu\lambda} T^{\mu\lambda} = 0. \tag{48}$$

Tällä yhtälöllä on neljä komponenttia, yksi jokaiselle ν:lle. Laskua varten tarvitaan siis energia-liikemäärä -tensorin käänteistensori sekä aiemmin lasketut Christoffelin symbolit (32). Energia-liikemäärä -tensorin käänteistensori saadaan laskettua operoimalla sitä käänteismetriikalla yhtälön (7) mukaan

$$T^{\mu\nu} = g^{\mu\lambda} g^{\nu\sigma} T_{\lambda\sigma}. \tag{49}$$

Esimerkkinä ensimmäinen T^{00} komponentti

$$T^{00} = g^{0\lambda} g^{0\sigma} T_{\lambda\sigma} = g^{00} g^{00} T_{00} = \left[\frac{1}{-e^{2\alpha}}\right]^2 \rho e^{2\alpha} = \frac{\rho}{e^{2\alpha}}.$$

Vastaavasti yhtälöllä (49) laskemalla saadaan käänteinen energia-liikemäärä tensori muotoon

$$T^{\mu\nu} = \mathrm{diag}\bigg(\frac{\rho}{e^{2\alpha}}, \frac{p}{e^{2\beta}}, \frac{p}{r^2}, \frac{p}{r^2 \sin^2(\theta)}\bigg). \tag{50}$$

Nyt voidaan laskea jatkuvuusyhtälön (48) komponentit. Esimerkkilasku lasketaan tt-komponentille termi kerrallaan. Valitaan siis $\nu = t = 0$. Tällöin yhtälön (48) ensimmäinen termi on

$$\partial_\mu T^{\mu 0} = \partial_0 T^{00} + \partial_1 T^{10} + \partial_2 T^{20} + \partial_3 T^{30} = \partial_0 T^{00} = 0,$$

jossa ei-diagonaalitermit, T^{i0}, on automaattisesti nollia, sekä T^{00} komponentin aikaderivaatta on nolla. Toinen termi

$$\Gamma^{\mu}_{\mu\lambda} T^{\lambda 0} = \Gamma^{\mu}_{\mu 0} T^{00} = \Big[\Gamma^{0}_{00} + \Gamma^{1}_{10} + \Gamma^{2}_{20} + \Gamma^{3}_{30}\Big] T^{00} = 0,$$

jossa konnektiot ovat triviaaleja jo laskettujen konnektioiden (32) perusteella. Kolmannesta termistä saadaan

$$\Gamma^{0}_{\mu\lambda} T^{\mu\lambda} = \Gamma^{0}_{00} T^{00} + \Gamma^{0}_{11} T^{11} + \Gamma^{0}_{22} T^{22} + \Gamma^{0}_{33} T^{33} = 0,$$

jossa on hyödynnetty aikaisemmin laskettuja konnektioita (32). Nähdään siis, että $\nu = t = 0$, eli yhtälö (48) antaa nollatuloksen. Nollatulos tarkoittaa sitä, että se toteutuu automaattisesti annettujen funktioiden, $\alpha(r)$, $\beta(r)$, $\rho(r)$ ja $p(r)$ sekä niiden r-riippuvuuden puitteissa. Vastaavasti laskemalla termeittäin muut jatkuvuusyhtälön (48) komponentit, $\nu = 1, 2, 3$, saadaan tulokseksi, että ainoastaan $\nu = r = 1$ antaa uuden rajoituksen jatkuvuusyhtälön toteutumiselle. Tästä rajoituksesta saadaan relaatio $\alpha(r)$ ja $p(r)$ välille:

$$\frac{d\alpha}{dr}(\rho + p) = -\frac{dp}{dr}. \tag{51}$$

Yhdistämällä tämä relaatio yhtälöön (46), saadaan Tolman-Oppenheimer-Volkoff-differentiaaliyhtälö

$$\frac{dp}{dr} = -\frac{\Big(\rho + p\Big)\Big[Gm(r) + 4\pi G r^3 p\Big]}{r\Big[r - 2Gm(r)\Big]}. \tag{52}$$

Johdettu Tolman-Oppenheimer-Volkoff -yhtälö (52) on siis relativistinen hydrostaattinen tasapainoyhtälö, joka kuvaa sitä, miten tähti pysyy tasapainossa oman gravitaationsa ja sisäisen paineen kanssa. Se kytkee massan, $m(r)$, ja energiatiheyden, $\rho(r)$, keskenään yhtälön (43) nojalla. TOV:n ratkaisemiseksi tarvitaan vielä tähden tilanyhtälö, joka usein määritellään paineena energiantiheyden funktiona. Tällöin tilanyhtälö voidaan ilmaista muodossa

$$p = p(\rho). \tag{53}$$

Yhdessä nämä yhtälöt, (43), (52) ja (53), siis olennaisesti kuvaavat tähden rakenteen. Saadut yhtälöt ovat linjassa tunnetun tiedon ja esimerkiksi lähteiden [3, 10, 11] kanssa.

3.3 Newtonilainen hydrostaattinen tasapainoyhtälö

Hydrostaattinen tasapainoyhtälö voidaan johtaa pienten nopeuksien alueella, toisin sanoen epärelativistisella alueella, siirtymällä SI-yksiköihin sekä asettamalla valonnopeus äärettömään. TOV-yhtälö (52) saadaan SI-yksiköihin kertomalla sitä sopivasti valonnopeudella c ja ottamalla yhteisen tekijän termeistä. Tällöin TOV (52) saadaan muotoon

$$\frac{dp}{dr} = -\frac{Gm\rho}{r^2}\left(1 + \frac{p}{\rho c^2}\right)\left(1 + \frac{4\pi r^3 p}{mc^2}\right)\left(1 - \frac{2Gm}{rc^2}\right)^{-1}, \tag{54}$$

jossa c on valonnopeus. Nyt asettamalla valonnopeuden lähestymään ääretöntä, $c \to \infty$, sulkutermit yhtälössä (54) saavat arvon yksi:

$$\left(1 + \frac{p}{\rho c^2}\right)\left(1 + \frac{4\pi r^3 p}{mc^2}\right)\left(1 - \frac{2Gm}{rc^2}\right)^{-1} \to 1, \tag{55}$$

ja TOV-yhtälö saa epärelativistisen muodon

$$\frac{dp}{dr} = -\frac{Gm\rho}{r^2}. \tag{56}$$

Tätä Newtonilaista tasapainoyhtälöä on käsitelty laajemmin esimerkiksi lähteessä [9].

4 Valkoinen kääpiö

Tässä osiossa käyn läpi valkoisten kääpiöiden taustaa ja syntyprosessia sekä ratkaisen aiemmin johdetut hydrostaattiset tasapainoyhtälöt numeerisesti. Ratkaisuna numeriikasta saadaan valkoisen kääpiön massa, paine sekä energiatiheys säteen funktiona. Useaa tähteä mallintamalla saadaan valkoisen kääpiön massa-säde relaatio. Lopuksi tutkin myös relativistisen TOV:n antamaa energiatiheyden ratkaisun suhdetta Newtonilaiseen teoriaan.

4.1 Synty

Valkoiset kääpiöt ovat eräitä tiheimpiä olemassa olevia kappaleita havaittavassa maailmankaikkeudessa. Ne ovat himmeitä ja tiheitä tähtien jäännöksiä sekä viimeisiä matalien ja keskimassaisten tähtien havaittavia elämänkaaren vaiheita. Valkoiset kääpiöt muodostuvat tähdistä, joiden massa on välillä $0{,}5M_\odot \ldots 10M_\odot$ ($M_\odot$:lla merkitään Auringon massaa). [12]

Valkoisia kääpiöitä syntyy pääasiassa kolmella eri tavalla. Päävaiheen tähden massan ollessa alle puoli Auringon massaa, $< 0{,}5M_\odot$, tähti ei ole tarpeeksi kuuma heliumfuusioon, vaan tähti polttaa kaiken vedyn muodostuessaan samalla siniseksi kääpiöksi. Vedyn loputtua tähdestä fuusioreaktio sammuu, ja tähti on muuttunut valkoiseksi kääpiöksi [13]. Päävaiheen tähden massan ollessa välillä $0{,}5M_\odot \ldots 8M_\odot$ tähti on riittävän kuuma ylläpitääkseen heliumfuusiota samalla synnyttäen hiiltä ja happea kolmois-alpha-prosessissa [14]. Heliumin palaessa loppuun fuusioreaktiossa tähden ydin muuttuu koostumukseltaan hiileksi ja hapeksi, jolloin tähti menettää uloimmat kerroksensa avaruuteen. Näiden menetettyjen kerrosten materiaaleista muodostuu planetaarinen tähtisumu sekä tähden ytimestä jää hiilestä ja hapesta koostuva valkoinen kääpiö [13]. Päävaiheen tähden massan ollessa välillä $8M_\odot \ldots 10M_\odot$ tähti on tarpeksi kuuma ylläpitämään myös hiili- ja neonfuusiota. Tässä vaiheessa valkoista kääpiötä kasassa pitävä elektroni-degeneraatiopaine ei riitä pitämään tähden ydintä kasassa vaan se romahtaa ja tähti räjähtää ytimen romahduksen yhteydessä supernovana. Supernovasta on mahdollista jäädä jäljelle olosuhteiden puitteissa hiilestä, neonista ja magnesiumista koostuva valkoinen kääpiö [15].

Valkoisissa kääpiöissä ei siis tapahdu fuusiota, vaan niiden luminositeetti ja lämpötila johtuu päävaiheen tähden termisen energian jäänteistä [16]. Valkoiset kääpiöt koostuvat pääosin kylmästä fermikaasusta, ja tämän fermikaasun massasta suurin osa on atomien ytimissä. Valkoisen kääpiön koossa pitävä paine vuorostaan syntyy degeneroituneista elektroneista, jolloin ne ovat Paulin kieltosäännön mukaan alimmilla mahdollisilla energiatiloilla hylkien toisiaan ja estäen niitä romahtamasta atomien ytimiin yhteen singulaariin pisteeseen. Valkoisen kääpiön oman fuusioreaktion puutteen vuoksi sitä pitää kasassa tämä fermikaasun degeneraatiosta johtuva paine. Tästä paineesta ja degeneraatiosta johtuen ideaalista fermikaasusta muodostuva materia on hyvin tiheää. Tälle fermikaasulle voidaan johtaa tilanyhtälö tietyin approksimaatioin. [4, 17]

4.2 Tilanyhtälö ja rakenne

Degeneroituneen elektronikaasun tilanyhtälön johtoa ei tässä työssä käydä erityisen yksityiskohtaisesti läpi, sillä siitä on jo olemassa kattavia lähteitä. Aiheesta löytyy mm. suomenkielinen opinnäytetyö [18]. Tämä osio perustuu pääosin lähteisiin [4, 17]. Laskujen yksinkertaistamista varten valkoisten kääpiöiden materiaa voidaan approksimoida kylmänä ja ideaalina fermikaasuna. Tämä tarkoittaa käytännössä degeneroitunutta elektronikaasua, joka on lähellä absoluuttista nollapistettä. Näiden oletusten nojalla degeneroituneelle elektronikaasun tilanyhtälölle voidaan johtaa statistisen fysiikan avulla kohtalaisen tarkka approksimaatio luonnollisissa yksiköissä ($c = \hbar = \epsilon_0 = k_B = 1$) muotoon [17]

$$P_{deg} = \frac{m_e^4}{24\pi^2}\left[x\sqrt{1+x^2}(2x^3-3) + 3\log[x+\sqrt{1+x^2}]\right], \tag{57}$$

jossa x on degeneroituneen elektronikaasun suurienergisten elektronien relativistisuutta kuvaava kerroin

$$x = \sqrt[3]{\frac{3\pi^2\rho}{\mu_e m_e^3 m_p}}. \tag{58}$$

Näissä yhtälöissä esiintyvät muuttujat ovat μ_e on kaasun muodostavien atomeiden massaluvun ja järjestysluvun suhde, m_e on elektronin massa, m_p on protonin massa ja ρ energiatiheys. Kaasun koostumusta kuvaava vakio voidaan päätellä valkoisen kääpiön syntyprosessista riippuen. Karkeana arviona valkoisen kääpiön voidaan approximoida koostuvan heliumista, hiilestä tai hapesta, jolloin vakio saa arvon $\mu_e = 2$. [17]

Valkoisen kääpiön rakenteen selvittämiseksi ja elektronikaasun paineen hankalan muodon vuoksi on hyödyllistä ilmoittaa hydrostaattiset tasapainoyhtälöt, (52) ja (56), energiatiheyden derivaattana ketjusäännön ($\frac{dP}{dr} = \frac{dP}{d\rho}\frac{d\rho}{dr}$) avulla muodossa (luonnollisissa yksiköissä $c = \hbar = \epsilon_0 = k_B = 1$)

$$\frac{d\rho}{dr} = -\frac{(\rho + p)[Gm + 4\pi G r^3 P_{deg}]}{r[r - 2Gm]} \left(\frac{dP_{deg}}{d\rho} \right)^{-1}, \tag{59}$$

ja

$$\frac{d\rho}{dr} = -\frac{Gm\rho}{r^2} \left(\frac{dP_{deg}}{d\rho} \right)^{-1}, \tag{60}$$

joissa

$$\frac{dP_{deg}}{d\rho} = \frac{dP_{deg}}{df}\frac{df}{dx}\frac{dx}{d\rho} \tag{61}$$

$$= ab(3\rho^{\frac{2}{3}})^{-1}\left[\frac{8x^4}{\sqrt{1+x^2}}\right]. \tag{62}$$

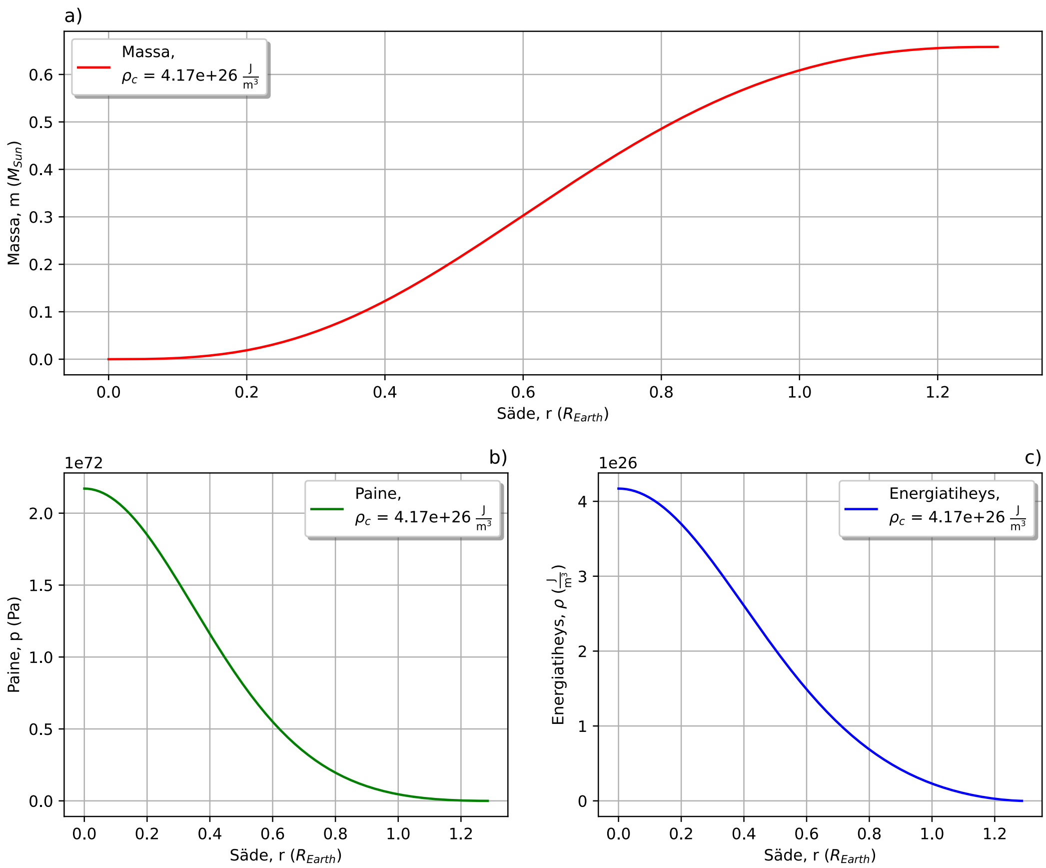
TOV-yhtälössä gravitaatiovakio luonnollisissa yksiköissä on $G = 6{,}707113 \cdot 10^{-39}\mathrm{GeV}^{-2}$.

Nyt valkoisen kääpiön rakenne voidaa selvittää ratkaisemalla yhtälöt (59) ja (43) elektronikaasun tilanyhtälön (57) avulla numeerisesti. Nämä yhtälöt numeerista ratkaisua varten on kirjoitettu liitteessä D ja yhtälöryhmän ratkaisija on liitteessä A. Ohjelmat ratkaisevat yhtälöt luonnollisissa yksiköissä, mutta ratkaisut esitetään SI-yksiköissä. Yksiköiden muunnoksissa käytettiin apuna lähdettä [19]. Ratkaisua varten yhtälöryhmälle täytyy antaa alkuarvaukset massalle sekä energiatiheydelle integrointialueen alkupisteessä. Singulariteettien ja numeeristen ongelmien välttämiseksi valitaan integrointirajan alkupisteeksi $r_{min} = 0{,}001\ \mathrm{m} = 5{,}067 \cdot 10^{12}\ \mathrm{GeV}^{-1}$ ja määritetään massan alkuehto

$$m(r_{min}) = \frac{4}{3}\pi\rho(r_{min})r_{min}^3, \tag{63}$$

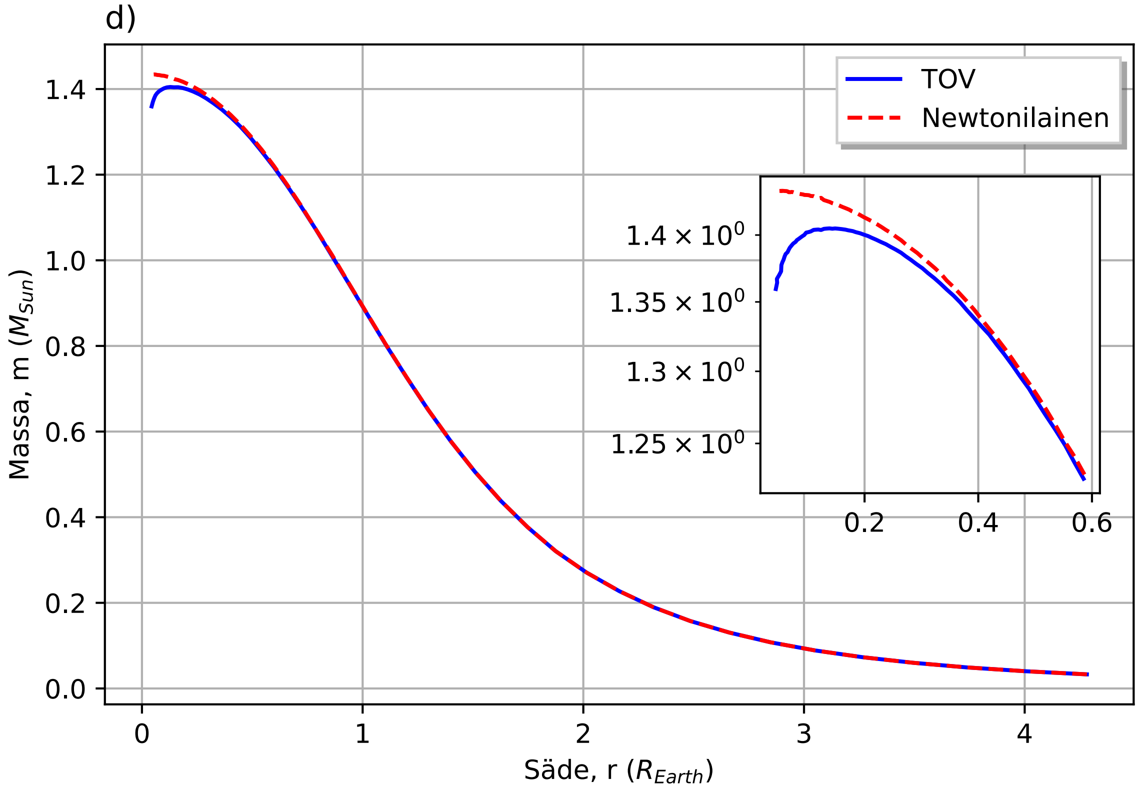
jossa alkuehto $\rho(r_{min})$ vastaa energiatiheyttä valkoisen kääpiön keskipisteessä [9].

Valitsemalla keskipisteen energiatiheydeksi $\rho_c = \rho(r_{min}) = 4{,}17 \cdot 10^{26}\,\frac{\mathrm{J}}{\mathrm{m}^3}(= 2 \cdot 10^{-11}\ \mathrm{GeV}^4)$ saadaan massan alkuehdoksi yhtälöllä (63) $m(r_{min}) = 19{,}43\ \mathrm{kg}(= 1{,}09 \cdot 10^{28}\mathrm{GeV})$. Määritellään myös valkoisen kääpiön rajapinta sille säteelle R_{WD}, jossa elektronikaasun paine saavuttaa arvon nolla $P(R_{WD}) = 0$ [9]. Näillä parametreilla ratkaisuksi saadaan valkoisen kääpiön massa, paine ja energiatiheys, jotka on esitetty säteen funktiona kuvion 1 kuvaajissa a, b ja c.



Kuvio 1. Valkoisen kääpiön rakenne alkuehdolle $\rho_c = 4{,}17 \cdot 10^{26}\,\frac{\mathrm{J}}{\mathrm{m}^3}(= 2 \cdot 10^{-11}\ \mathrm{GeV}^4)$ ja vastaavalle massalle. Suureet on ilmaistu SI-yksiköissä ja kaikkien kuvaajien x-akselin säde on skaalattu maan säteellä (6371 km). Kuvaajassa a) on massa säteen funktiona, jossa massa on skaalattu Auringon massalla (Auringon massa $= M_{Sun} = 1{,}9891 \cdot 10^{30}$ kg). Kuvaajassa b) on degeneroituneen elektronikaasun paine säteen funktiona sekä kuvaajassa c) energiatiheys valkoisen kääpiön sisällä säteen funktiona.

4.3 Relativistisen ja epärelativistisen ratkaisun vertailu



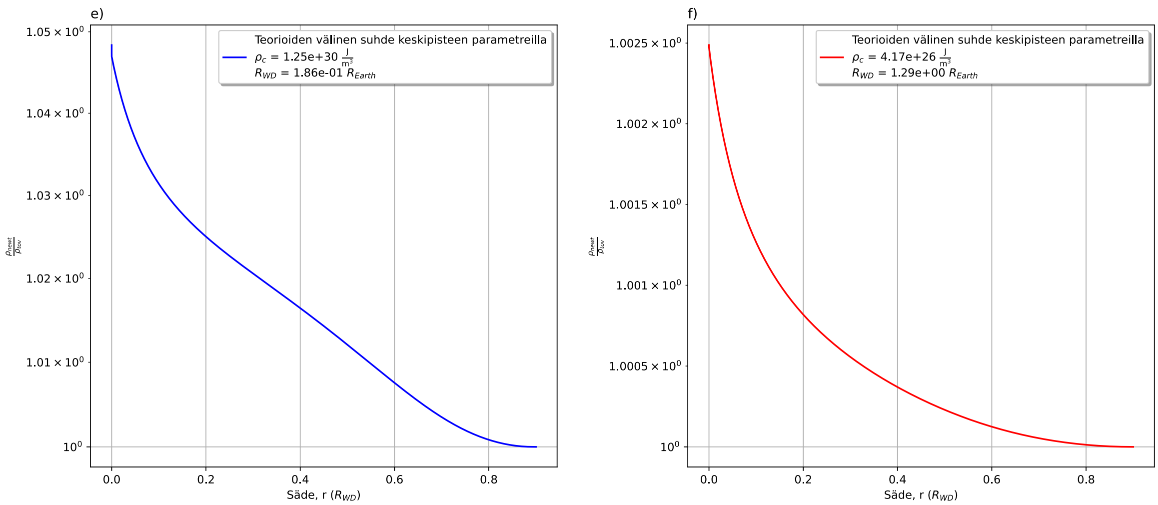
Kuvio 2. Valkoisen kääpiön massa-säde relaatio relativistisella (sininen käyrä) ja epärelativistisella (punainen käyrä) teorialla esitettynä. Pikkukuvassa samat käyrät zoomattuna (0 - 0,6)R_{Earth} alueelle. Käyrät ratkaistiin keskipisteen energiatiheyksien alueella $4{,}17 \cdot 10^{26}\ \frac{\mathrm{J}}{\mathrm{m}^3} \leq \rho_c \leq 1{,}67 \cdot 10^{32}\ \frac{\mathrm{J}}{\mathrm{m}^3}$. x-akseli on skaalattu Maan säteellä ja y-akseli Auringon massalla.

Valkoisia kääpiöitä voidaan mallintaa useammalla eri keskipisteen energiatiheydellä ja tutkia näiden massa-säde-relaatiota. Massa-säde-relaatio kertoo valkoisen kääpiön kokonaismassan ja säteen kytkeytymisen toisiinsa. Seuraavaksi ratkaisen massa-säde relaation sekä relativistella yhtälöllä (59), että Newtonilaisella yhtälöllä (60). Vaikka valkoisen kääpiön lämpötila vaihtelee sen elinaikana suurestikin lämpötilan laskiessa ($T_{WD} \approx 10^7\mathrm{K} \ldots 10^3\mathrm{K}$, jossa T_{WD} on valkoisen kääpiön lämpötila [20, 21]) ja havaittujen valkoisten kääpiöiden lämpötilojen ollessa kertaluokkaa $T_{WD} \approx 10^3\mathrm{K} \ldots 10^4\mathrm{K}$ [22], niiden voidaan katsoa olevan suurimman osan elinajastaan matalan lämpötilan alueella, $T_{WD} \lesssim 10^4$K, joten on perusteltua olettaa niiden olevan epärelativistisia systeemejä, mistä johtuen massa-säde-käyrien eri teorioille pitäisi olla hyvin samanlaatuiset.

Kuvasta 2 huomataan degeneroituneen elektronikaasun hieman epätyypillinen käytös verrattuna esimerkiksi vetykaasusta koostuvaan niin sanottuun elävään tähteen, jossa tähden energiatiheyttä kasvattamalla tähden kokonaismassa kasvaa säteen kasvaessa. Degeneroituneelle elektronikaasulle energiatiheyttä kasvattaessa sen kokonaismassa kasvaa, mutta säde pienenee Chandrasekharin rajaan ($\approx 1{,}4 M_{Sun}$ [17]) asti. Tämän lisäksi on havaittavissa selvästi relativistisen ja epärelativistisen teorian välinen ero massa-säde relaatioon suurilla keskipisteen tiheyksillä. Matalilla keskipisteen tiheyksillä teoriat antavat lähelle saman tuloksen, mutta suuremmilla tiheyksillä noin alle $1R_{Earth}$ säteisien valkoisten kääpiöiden kohdalla teoriat alkavat erottua jo toisistaan epärelativistisen teorian antaessa systemaattisesti suurempia arvoja kuin relativistinen teoria. Kuvasta on myös havaittavissa, että relativistinen teoria rajoittaa valkoisen kääpiön kokonaismassan noin 1,4 M_{Sun}:aan. Tämä raja tunnetaan myös Chandrasekharin rajana, joka on maksimaalinen mahdollinen massa valkoisille kääpiöille. Rajan tarkka arvo riippuu valkoisen kääpiön koostumuksesta. Massa-säde relaation ratkaisija on liitteessä B.

Massa-säde relaatioiden lisäksi relativistisen ja epärelativistisen teorioiden eroa voidaan vertailla ottamalla molempien teorioiden antamien ratkaisuiden energiatiheyksien suhde erilaisilla keskipisteiden energiatiheyksillä. Kuvion 2 massa-säde relaatiosta voidaan jo suoraan tehdä johtopäätös, että relativistiset korjaukset teoriaan pienillä tiheyksillä ovat hyvin pieniä ja energiatiheyksien suhde on lähellä yhtä, kun taas suurilla tiheyksillä käyrät erottuvat toisistaan selkeästi.

Kuvion 3 kuvaajissa e) ja f) on näiden kahden eri teorian antamien energiatiheyksien välinen suhde. Matalan tiheyden kuvaajasta f) huomataan, että pienillä tiheyksillä suhteellisuusteorian antamat korjaukset ovat verrattain pieniä ja ne lähestyvät huomattavan nopeasti nollaa. Suurilla tiheyksillä taas kuvaajassa e) huomataan, että suhteellisuusteorian antamat korjaukset ovat kertaluokkaa suurempia ja energiatiheyksien suhteeseen muodostuu kumpu, jonka vuoksi teorioiden energiatiheyksien suhde lähestyy hitaammin ykköstä kuin matalilla tiheyksillä. Energiatiheyksien suhteiden numeerinen lasku on liitteessä C.



Kuvio 3. Relativistisen teorian ja epärelativistisen teorian antamien energitiheyksien välinen suhde. X-akseli on skaalattu kyseisen valkoisen kääpiön säteeseen, joka on ilmaistu kuvaajan laatikossa.Y-akselilla on energiatiheyksien suhde logaritmisessa skaalassa. Vasen kuvaaja e) on tiheälle tähdelle ja oikea kuvaaja f) harvemmalle tähdelle. Keskipisteiden energiatiheydet merkattu laatikkoon.

5 Päätäntö

Tutkielmassani kävin läpi tarvittavia suhteellisuusteorian pohjatietoja relativistisen hydrostaattisen tasapainoyhtälön, Tolman-Oppenheimer-Volkoff -yhtälön, johtamiseksi sekä tarkastelin yhtälöä epärelativistisella rajalla johtaen newtonilaisen hydrostaattisen tasapainoyhtälön. Avaruuden kaarevuuden, metriikan ja vektoreiden konseptuaalisen käsityksen luomisen jälkeen siirryin esittelemään matemaattista formalisimia, johon kuuluu muun muassa kovariantti derivaatta, Riemannin tensori, Christoffelin konnektio sekä Einsteinin yhtälöt. Teoreettisen formalismin jälkeen johdin TOV-yhtälöt sekä ratkaisin ne numeerisesti Python-ohjelmointikielellä luonnollisissa yksiköissä ja tulokset esittelin SI-yksiköissä. Käytetyt ohjelmat on esitelty liitteissä A, B, C ja D.

Työssä laskin tarvittavat Christoffelin konnektiot Riccin tensorin ratkaisua varten esimerkkilaskuja esitellen. Riccin tensorista päästiin Einsteinin yhtälöihin, joista saatiin ideaalifluidin energia-liikemäärä tensorin avulla ratkaisuksi Tolman-Oppenheimer-Volkoff yhtälön. Yhtälöt olettavat staattisen ja pallosymmetrisen aika-avaruuden. Sovellettaessa tasapainoyhtälöitä esimerkiksi valkoisen kääpiön tapauksessa oletin valkoisen kääpiön pyörimismäärän ja pimeän aineen vaikutuksen olevan nolla, näiden olettamuksen lisäksi oletin valkoisen kääpiön koostuvan heliumista, hiilestä tai hapesta.

Numeriikasta saadut tulokset olivat pääosin odotettuja. Valkoisen kääpiön energiatiheys ja paine laskivat oletetunlaisesti massan kasvaessa hitaasti maksimiarvoonsa. Massa-säde relaatiosta pystyttiin ratkaisemaan Chandrasekharin raja sekä vertailemaan relativistisen ja epärelativistisen teorian eroa tiheiden tähtien tapauksessa. Tulosten luotettavuudesta ja Python-ohjelmien toimivuudesta saadaan hyvä käsitys vertailemalla tuloksia lähteisiin [11, 23, 24], joissa on laskettu valkoisten kääpiöiden massa-säde relaatiota lämpötilan nollapisteessä sekä tehty yleisiä suhteellisuusteoreettisia laskuja valkoisille kääpiöille. Luotettavuuden lisäksi lisää tietoa käytetyistä metodeista sekä Python-kielestä löytyy lähteestä [25].

Huomiona sanottakoon, että ohjelmassa käytettyjen muuttujien vuoksi ajoaika kasvaa huomattavasti integrointiaskeleen pienentyessä paremman resoluution saavuttamiseksi. Ohjelmaa voisi parantaa muun muassa esittämällä ratkaistavat muuttujat eri tavalla skaalaamalla ne esimerkiksi keskipisteen energiatiheyden yksiköihin. Koodi on saatavilla myös julkisena githubista lähteestä [26]. Tuloksissa en myöskään käsitellyt numeerista virhettä ollenkaan, joten TOV-käyrän epätasaisuus kuvassa 2 $R_{WD} \leq 0.2R_{Earth}$ alueella johtuu todennäköisesti tästä numeerisesta virheestä. Yksittäisten tähtien kohdalla eri teorioiden antamat energiatiheyden suhteet olivat odotetut. Tiheille valkoisille kääpiöille energiatiheyksien suhde oli kertaluokkaa suurempi kuin harvoille valkoisille kääpiöille. Energiatiheyksien suhteeseen muodostuneesta kummusta huolimatta suhteellisuusteoreettiset korjaustermit hävisivät nopeasti tähden rajapintaa lähestyttäessä. Harvemmille valkoisille kääpiöille energiatiheyksien suhde näyttää lähes samanlaiselta ja suhteellisuusteoreettiset korjaustermit ovat hyvin pieniä ja häviävät nopeasti.

Sopivia jatkotutkimuksen aiheita olisi esimerkiksi kuvion 3 kuvaajassa e) esiintyvän kummun lähtöperän selvittäminen tai muiden muuttujien, kuten pyörimismäärän ja pimeän aineen vaikutuksen, mallintaminen äärimmäisen tiheissä kappaleissa.

Lähteet

[1] J. C. Mbagwu, Z. L. Abubakar ja J. O. Ozuomba. ”A Review Article on Einstein Special Theory of Relativity”. *International Journal of Theoretical and Mathematical Physics* 0 (2020), s. 65–71. URL: http://article.sapub.org/10.5923.j.ijtmp.20201003.03.html.

[2] A. Dey. *General Relativity: A Simple Discussion [Review]*. The Journal of Young Physicists, tammikuu 2021. URL: https://www.journalofyoungphysicists.org/post/general-relativity-a-simple-discussion-review (viitattu 27. 01. 2023).

[3] S. M. Carroll. *Spacetime and geometry : an introduction to general relativity*. Cambridge Cambridge University Press, 2019.

[4] S. Chandrasekhar. *An introduction to the study of stellar structure*. New York] Dover Publications, 1967.

[5] R. M. Wald. *General relativity*. Univ. Of Chicago Press, 2009.

[6] D. McMahon ja P. M. Alsing. *Relativity Demystified*. McGraw Hill Professional, joulukuu 2005.

[7] A. Einstein. ”Die Grundlage der allgemeinen Relativitätstheorie”. *Annalen der Physik* 354 (1916), s. 769–822. DOI: 10.1002/andp.19163540702.

[8] R. P. Feynman, F. B. Morinigo ja W. G. Wagner. ”Feynman Lectures on Gravitation”. *European Journal of Physics* 24 (toukokuu 2003). DOI: 10.1088/0143-0807/24/3/702. (Viitattu 27. 01. 2023).

[9] S. D. Miller. ”The Fundamental Equilibrium Equation For Gaseous Stars And The Tolman-Oppenheimer-Volkoff Equation – Derivations And Applications With Emphasis On Optimisational-Variational Methods”. *arXiv:2109.02017 [gr-qc, physics:math-ph]* (syyskuu 2021). DOI: 10.48550/ARXIV.2109.02017. URL: https://arxiv.org/abs/2109.02017 (viitattu 27. 01. 2023).

[10] E. Chávez Nambo ja O. Sarbach. ”Static spherical perfect fluid stars with finite radius in general relativity: a review”. *Revista Mexicana de Física E* 18 (heinäkuu 2021). DOI: 10.31349/revmexfise.18.020208. (Viitattu 27. 01. 2023).

[11] A. Mathew ja M. K. Nandy. ”General relativistic calculations for white dwarfs”. *Research in Astronomy and Astrophysics* 17 (toukokuu 2017), s. 061. DOI: 10.1088/1674-4527/17/6/61. (Viitattu 27. 01. 2023).

[12] D. Saumon, S. Blouin ja P.-E. Tremblay. ”Current Challenges in the Physics of White Dwarf Stars”. *Physics Reports* 988 (marraskuu 2022), s. 1–63. DOI: 10.1016/j.physrep.2022.09.001. URL: https://www.sciencedirect.com/science/article/pii/S0370157322003180?via$%5C%$3Dihub (viitattu 27. 01. 2023).

[13] S. Jeffery. ”Stellar Evolution beyond the Main Sequence”. *Astronomy Now Magazine* (kesäkuu 1998). URL: https://web.archive.org/web/20150404004046/http://star.arm.ac.uk/~csj/pus/astnow/astnow.html (viitattu 27. 01. 2023).

[14] K. Nomoto, F. Thielemann ja S. Miyaji. ”The triple alpha reaction at low temperatures in accreting white dwarfs and neutron stars”. *Astronomy and Astrophysics* 149 (elokuu 1985), s. 239–245. URL: https://ui.adsabs.harvard.edu/abs/1985A$%5C%$26A...149..239N/abstract (viitattu 27. 01. 2023).

[15] K. Werner ym. ”On Possible Oxygen/Neon White Dwarfs: H1504+65 and the White Dwarf Donors in Ultracompact X-ray Binaries”. *arXiv:astro-ph/0410690* 1 (lokakuu 2004). DOI: 10.48550/ARXIV.ASTRO-PH/0410690. URL: https://arxiv.org/abs/astro-ph/0410690 (viitattu 27. 01. 2023).

[16] N. T. Tillman ja D. Dobrijevic. *White dwarfs: Facts about the Dense Stellar Remnants*. Space.com, maaliskuu 2022. URL: https://www.space.com/23756-white-dwarf-stars.html (viitattu 27. 01. 2023).

[17] R. Kippenhahn, A. Weigert ja A. Weiss. *Stellar Structure and Evolution*. 2. painos. Springer Berlin, 2012. URL: https://link.springer.com/book/10.1007/978-3-642-30304-3 (viitattu 27. 01. 2023).

[18] O. Väisänen. *Neutronitähden tilanyhtälön määrittäminen massa-säde -mittausten avulla*. LuK-tutkielma. Jyväskylän yliopisto, Fysiikan laitos, 2018.

[19] A. Myers. *Natural System of Units in General Relativity*. 2016. URL: https://www.seas.upenn.edu/~amyers/NaturalUnits.pdf (viitattu 27. 01. 2023).

[20] N. C. Hambly, S. J. Smartt ja S. T. Hodgkin. ”WD 0346+246: a Very Low Luminosity, Cool Degenerate in Taurus”. *The Astrophysical Journal* 489 (1997), s. L157–L160. DOI: 10.1086/316797. (Viitattu 27. 01. 2023).

[21] G. P. McCook ja E. M. Sion. ”A Catalog of Spectroscopically Identified White Dwarfs”. *The Astrophysical Journal Supplement Series* 121 (maaliskuu 1999), s. 1–130. DOI: 10.1086/313186. (Viitattu 27. 01. 2023).

[22] D. J. Eisenstein ym. ”A Catalog of Spectroscopically Confirmed White Dwarfs from the Sloan Digital Sky Survey Data Release 4”. *The Astrophysical Journal Supplement Series* 167 (marraskuu 2006), s. 40–58. DOI: 10.1086/507110. URL: https://arxiv.org/abs/astro-ph/0606700 (viitattu 07. 04. 2021).

[23] A. Carvalho, R. M. MarinhoJr ja M. Malheiro. ”Mass-Radius Relation for White Dwarfs Models at Zero Temperature”. *Journal of Physics: Conference Series* 706 (huhtikuu 2016). DOI: 10.1088/1742-6596/706/5/052016. (Viitattu 27. 05. 2021).

[24] M. Shuntov. ”High-Energy Astrophysics: White Dwarfs and Neutron Stars -M1/S2 Project Report” (huhtikuu 2018). DOI: 10.13140/RG.2.2.13518.77121. URL: https://www.researchgate.net/publication/324845118_High-Energy_Astrophysics_White_Dwarfs_and_Neutron_Stars_-_M1S2_Project_Report?channel=doi&linkId=5ae787e3a6fdcc03cd8db2bc&showFulltext=true (viitattu 27. 01. 2023).

[25] M. E. J. Newman. *Computational physics*. Createspace, 2013.

[26] A. Voutilainen. *TOV_solver*. GitHub, elokuu 2022. URL: https://github.com/Awisuals/TOV_solver (viitattu 27. 01. 2023).

A Python, TOV-yhtälön solveri

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Oct  5  2022

@author: Antero
"""
from scipy.integrate import solve_ivp
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec

from functions import *
from structure_equations import *
"""
Write DE-group as:
    EoS -> pressure
    TOV -> Energy density
And solve it.
"""

def set_initial_conditions(rmin, G, K, rho0=0., p0=0., a=0):
    """
    Utility routine to set initial data. Can be given either
    pressure or energy density at core. Value a tells
    which is first.

    Parameters
    ----------
    rmin : Float
        Lower limit for integration.
    G : Float
        Polytrope constant power.
    K : Float
        Polytrope constant of proportionality.
    rho0 : Float, optional
        Given energy density at core. The default is 0.
    p0 : Float, optional
        Given pressure at core. The default is 0.
    a : Int, optional
        Choice for given initial value. Another is then 0 and
        calculated from EoS. Can take both also.

        Choice:
            a = 0 is for given rho0.
            a = 1 is for given p0.
            a = 2 is for given rho0 and p0.
            a = 3 is for given rho0 and computes p0.

        The default is 0.

    Returns
    -------
    m : Float
        Mass inside radius rmin.
    p : Float
        pressure at r ~ 0.
    rho : Float
        Energy density at r ~ 0.

    """
    rho_values0 = [rho0, EoS_choiser(0, p=p0, Gamma=G, Kappa=K), rho0, rho0]
    p_values0 = [EoS_choiser(1, rho=rho0, Gamma=G, Kappa=K), p0, p0, EoS_choiser(2,
    rho=rho0)]
    if a == 0:
        rho = rho_values0[a]
        p = p_values0[a]
    if a == 1:
        p = p_values0[a]
        rho = rho_values0[a]
    if a == 2:
        rho = rho0
        p = p0
```

```python
    if a == 3:
        rho = rho_values0[a]
        p = p_values0[a]
    m = 4./3.*np.pi*rho0*rmin**3
    return m, p, rho


def TOV_rho(r, y, K, G, interpolation, eos_choise, tov_choise, rho_center):

    # HUOM! Tänne alkuarvaukset luonnollisissa yksiköissä GeV^x!
    # NOTE! Here are the initial guesses in natural units GeV^x!

    # Asetetaan muuttujat taulukkoon
    # Paine valitaan valitsin-funktiossa.
    # //
    # Let's set the variables in the table.
    # The energy density is selected in the selector function.
    m = y[0].real + 0j
    rho = y[1].real + 0j
    p = EoS_choiser(eos_choise, interpolation, G, K, 0, rho).real + 0j

    # Ratkaistavat yhtälöt // Equations to be solved
    dy = np.empty_like(y)
    # Massa ja Energiatiheys DY // Mass and energy density DE
    dy[0] = Mass_in_radius(rho, r)                    # dmdr
    dy[1] = TOV_choiser(tov_choise, m ,p, rho, r)    # drhodr

    return dy


def found_radius(t, y, d1, d2, d3, d4, d5, d6):
    """
    Event function: Defined boundary of pressure
    ODE integration stops when this function returns True

    Parameters
    ----------
    t : TYPE
        DESCRIPTION.
    y : TYPE
        DESCRIPTION.
    d1, d2, d3, d4, d5, d6 : args
        Dummy params.

    Returns
    -------
    None
        Checks when energy density reaches zero.

    """
    d1, d2, d3, d4, d5, d6 = d1, d2, d3, d4, d5, d6
    pressure = EoS_degelgas(y[1].real)
    return pressure >= 0*d6# 4.3e21 # 1e16


# Määritellään funktio TOV-yhtälöiden ratkaisemiseksi ja koodin ajon
# helpottamiseksi. Funktiolle annetaan kasa parametreja ja se ratkaisee
# aijemmin määritellyt yhtälöt.
# //
# Let's define a function to solve the TOV equations and to help run the code
# easier. The function is given a bunch of parameters and it solves
# previously defined equations.
def TOV_solver(ir=[], n=0, R_body=0, kappa_choise=0, rho_K=0, p_K=0, rho_c=0,
               p_c=0, a=0, eos_choise=0, tov_choise=0, interpolation=0, body=""):
    """
    Appropriate initial values and equations can be chosen. Solves TOV equations
    in this case for the corresponding astrophysical body. As a solution
    the mass, pressure, energy density and radius of an astrophysical body
    are obtained.

    Solver works in natural units (but can be manually changed) so be sure
    to input values in correct units! Function returns solutions in natural
```

```
    units also.

    Parameters
    ----------
    ir : Array
        Integration range, pre defined to [5.067e12, np.inf].
    n : Float
        Polytrope index.
    R_body : Float, optional
        Approximate the radius of the astrophysical body
        to be modeled. The default is 0..
    kappa_choise : Int, optional
        Choise for which way Kappa is computed:
            0=kappa_from_p0rho0 (needs corresponding p and rho)
            1=kappa_from_r0rho0n (needs approximate radius and CENTRAL rho)
    rho_K : Float, optional
        Energy density for which we want calculate
        the corresponding constant of proportionality. The default is 0..
    p_K : Float, optional
        Pressure for which we want calculate
        the corresponding constant of proportionality.
        ONLY NEEDED WHEN kappa_choise=0. Has to be corresponding
        to rho_K. The default is 0..
    rho_c : Float, optional
        Central energy density. Used to compute initial values.
        The default is 0..
    p_c : Float, optional
        Central pressure. Used to compute initial values. The default is 0..
    a : Int, optional
        Can be:
            a=0, 1, 2, 3
        Choice for given initial value. Please refer to
        set_initial_conditions-function documentation for
        additional information. The default is 0..
    EoS_choise : Int, optional
        Choise for what EoS is used to compute initial values.
        Choise:
            0=Polytrope EoS.
            1=Interpolated EoS from data.
            2=Degenerate electron gas
        Please refer to EoS_choiser-function for additional information.
        The default is 0..
    TOV_choise : int, optional
        Choise for tov-equation:
            0=TOV, integrate p
            1=NEWT, integrate p
            2=TOV, integrate rho
            3=NEWT, integrate rho
    interpolation : interpolate, optional
        Has to be given if choise EoS_choise=1. Otherwise can be ignored.
        The default is 0..
    body : String
        Changes title for graphs. Input depending what is modeled.
        ATM isn't necessary.

    Returns
    -------
    r : Array
        Radius solution for modeled body.
    m : Array
        Mass solution for modeled body.
    p : Array
        Pressure solution for modeled body.
    rho : Array
        Energy density solution for modeled body.

    """
     # Määrätään vakioita.
    # //
    # Determine constants.
    M_sun = 1.9891e30               # kg
    R_earth = 6371                  # km
```

```python
        # Asetetaan integrointiparametrit.
        # Integraattori adaptiivinen, lopettaa integroinnin tähden rajalla.
        # //
        # Let's set the integration parameters.
        # Integrator adaptive, stops the integration at the star boundary.
        if len(ir)!=0: rmin, rmax = ir[0], ir[1]
        else: rmin, rmax = 5.067e12, np.inf

        # Ode-ratkaisijan lopettaminen ehdon täyttyessä.
        # //
        # Termination of ode solver when met with condition.
        found_radius.terminal = True # Should be true when works
        found_radius.direction = -1

        # Asetetaan alkuarvot // Set initial values
        if n != 0: Gamma = gamma_from_n(n); Kappa = kappa_choiser(kappa_choise, p_K, rho_K, Gamma, R_body, n)
        else: Gamma, Kappa = 0.1, 0.1

        m0, p0, rho0 = set_initial_conditions(rmin, Gamma, Kappa, rho_c, p_c, a)
        y0 = m0, p0, rho0

        # Tulostetaan annetut parametrit // Print given params
        print("\n Model of your choise and semi-realistic params for it:" +
        "\n Integration range = " + str(ir) +
        "\n Model = "          + body +
        "\n n = "              + str(n) +
        "\n R_body = "         + str(R_body) +
        "\n kappa_choise = " + str(kappa_choise) +
        "\n rho_K = "          + str(rho_K) +
        "\n p_K = "            + str(p_K) +
        "\n rho_c = "          + str(rho_c) +
        "\n p_c = "            + str(p_c) +
        "\n a = "              + str(a) +
        "\n eos_choise = "     + str(eos_choise) +
        "\n tov_choise = "     + str(tov_choise) +
        "\n interpolate = "    + str(interpolation) + "\n \n")

        print("Tulostetaan polytrooppivakiot:"
              + "\n Kappa: " + str(Kappa)
              + "\n Gamma: " + str(Gamma) + "\n \n")

        print("Asetetut alkuarvot (m, p ja rho):"
              + "\n m: " + str(y0[0])
              + "\n p: " + str(y0[1])
              + "\n rho: " + str(y0[2]) + "\n \n")

        # Ratkaistaan TOV annetuilla parametreilla
        # //
        # Let's solve the TOV with the given parameters
        # NOTE Best performance/resolution: One solution max_step=1e20
        #                                   Closer to core smaller 1e18
        #                                   MR-relation zoom 5e19.
        soln = solve_ivp(TOV_rho, (rmin, rmax), (m0.real, rho0.real), method='BDF',
        dense_output=False, events=found_radius, max_step = 1e18,
        args=(Kappa, Gamma, interpolation, eos_choise, tov_choise, rho0.real))

        print("\n Solverin parametreja:")
        print(soln.nfev, 'evaluations required')
        print(soln.t_events)
        print(soln.y_events)
        print("\n")

        # Energiatiheyden alkuarvaus // Energy denisty initial guess
        # SI-units
        rho0_si = '{:0.2e}'.format(rho0.real * 2.0852e37)
        # TOV ratkaisut // TOV solutions
        # Ratkaisut yksiköissä // Solutions in units:
        # [m] = GeV, [p] = GeV^4 ja [rho] = GeV^4
        r = soln.t * 1.9733e-16 * 1e-3 / R_earth # 2.6544006e-25*1e9
        m = soln.y[0].real * 1.7827e-27 / M_sun # / 1.9891e30 # 1.7827e-36
```

```
    rho = soln.y[1].real * 2.0852e37 # 3.16435553043e40
    p = EoS_degelgas(rho) * 2.0852e37 # 3.16435553043e40

    print("Saadut TOV ratkaisut ([m] = GeV, [p] = GeV^4 ja [rho] = GeV^4): \n")
    print("Säde: \n \n" + str(r.real) +
    "\n \n Massa: \n \n" + str(m.real) +
    "\n \n Energiatiheys: \n \n" + str(rho.real) +
    "\n \n Paine : \n \n" + str(p.real) + "\n \n")

    # # # Piirretään ratkaisun malli kuvaajiin yksiköissä:
    # # # //
    # # # Let's plot the model of the solution on graphs in units:
    # # # [m] = kg, [p] = Pascal ja [rho] = J/m^3
    gs = gridspec.GridSpec(2, 2)
    plt.figure()

    ax1 = plt.subplot(gs[0, :])
    ax2 = plt.subplot(gs[1, 0])
    ax3 = plt.subplot(gs[1, 1])
    ax1.plot(r, m, color='r', label=fr'Massa, ' '\n' fr'$\rho_{"c"}$ = {rho0_si}' r' $\frac{\mathrm{J}}{\mathrm{m}^{3}}$')
    ax2.plot(r, p, color='g', label=fr'Paine, ''\n' fr'$\rho_{"c"}$ = {rho0_si}' r' $\frac{\mathrm{J}}{\mathrm{m}^{3}}$')
    ax3.plot(r, rho, color='b', label=fr'Energiatiheys,' '\n' fr'$\rho_{"c"}$ = {rho0_si}' r' $\frac{\mathrm{J}}{\mathrm{m}^{3}}$')

    ax1.set(xlabel=r'Säde, r ($R_{Earth}$)',
            ylabel= r'Massa, m ($M_{Sun}$)',
            xscale="linear", yscale="linear")
    ax1.set_title('a)', loc="left")
    ax1.legend(shadow=True, fancybox=True)
    ax1.grid()
    ax2.set(xlabel=r'Säde, r ($R_{Earth}$)',
            ylabel=r'Paine, p (Pa)',
            xscale="linear", yscale="linear")
    ax2.set_title('b)', loc="right")
    ax2.legend(shadow=True, fancybox=True)
    ax2.grid()
    ax3.set(xlabel=r'Säde, r ($R_{Earth}$)',
            ylabel=r'Energiatiheys, $\rho$ $(\frac{\mathrm{J}}{\mathrm{m}^{3}})$',
            xscale="linear", yscale="linear")
    ax3.set_title('c)', loc="right")
    ax3.legend(shadow=True, fancybox=True)
    ax3.grid()

    plt.show()

    print("Tähden säde: \n" + str(r[-1]) +
          "\n Tähden massa: \n" + str(m[-1]) +
          "\n \n")

    return r.real, m.real, p.real, rho.real


def Models(model, args=[]):
    """
    Main function to drive TOV_solver. couple of example
    parameters given and then passes them to solver.

    Parameters
    ----------
    model : string
        Choose between given models in model_choise array.
    args : list, optional
        Insert custom params, by default []
    """
    model_choise = ["WD_NREL", "WD_REL"]
    model_params = [[0, 0, 0, 0, 0, 2e-14+0j, 0, 3, 2, 3, 0,
                     "Non-relativistic White Dwarf"],
                    [0, 0, 0, 0, 0, 2e-11+0j, 0, 3, 2, 2, 0,
                     "Relativistic White Dwarf"]]
```

```
    if model == "CUSTOM":
        n               =args[0]
        R_body          =args[1]
        kappa_choise    =args[2]
        rho_K           =args[3]
        p_K             =args[4]
        rho_c           =args[5]
        p_c             =args[6]
        a               =args[7]
        rho_func        =args[8]
        p_func          =args[9]
        interpolation   =args[10]
        body            =args[11]
    else:
        for i, m in enumerate(model_choise):
            if m == model:
                print(m)
                print(i)
                n               =model_params[i][0]
                R_body          =model_params[i][1]
                kappa_choise    =model_params[i][2]
                rho_K           =model_params[i][3]
                p_K             =model_params[i][4]
                rho_c           =model_params[i][5]
                p_c             =model_params[i][6]
                a               =model_params[i][7]
                rho_func        =model_params[i][8]
                p_func          =model_params[i][9]
                interpolation   =model_params[i][10]
                body            =model_params[i][11]

    TOV_solver(ir=[],
               n=n,
               R_body=R_body,
               kappa_choise=kappa_choise,
               rho_K=rho_K,
               p_K=p_K,
               rho_c=rho_c,
               p_c=p_c,
               a=a,
               eos_choise=rho_func,
               tov_choise=p_func,
               interpolation=interpolation,
               body=body)

Models("WD_REL")
```

B Python, Massa-Säde relaatio

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Oct  5  2022

@author: Antero
"""
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1.inset_locator import inset_axes

from functions import *
from structure_equations import *
from TOV_solver_rho import *
"""
Ratkaistaan massa-säde relaatio. Etsitään TOV-yhtälöiden ratkaisuja
jollakin rhospan-alueella. Ratkaistaan yhtälöitä siis tähden keskipisteen eri
energiatiheyksien arvoilla. Etsii ratkaisuja rho-muotoisesta tov-yhtälöstä.

Etsitään tähden raja (find_radius) paineen ratkaisusta ja sitä vastaava
massa massan kuvaajasta. Tallennetaan nämä arvot taulukkoon ja piirretään
kuvaaja.

Mallinnetaan nyt useaa tähteä ja piirretään
Massa-Säde - relaatio.
//
Let's solve the mass-radius relation. We are looking for solutions to the
TOV equations in some rhospan area. So let's solve the equations
from the center of the star with varying values of energy densities. Finds
solutions to a tov equation in rho form.


Let's find the limit of the star (find_radius) from the pressure solution
and its equivalent mass from the mass solution. Let's save these values in
an array and plot them.

Now let's model several stars and plot them
Mass-Radius - relation.
"""

def MR_relaatio(rho_min1, rho_min2, rho_max, N_MR1, N_MR2):
    """Solves mass-radius - relation.

    Parameters
    ----------
    rho_min1 : float
        Lower limit of central energy densities for bigger graph.
    rho_min2 : float
        Lower limit of central energy densities for inset graph.
    rho_max : float
        Higher limit of central energy densities.
    N_MR1 : int
        Bigger graph points.
    N_MR2 : int
        Inset graph points.
    """

    # Build N_MR1 or N_MR2 amount of star models
    rhospan1 = np.logspace(np.log10(rho_min1), np.log10(rho_max), N_MR1)
    rhospan2 = np.logspace(np.log10(rho_min2), np.log10(rho_max), N_MR2)
    print("\n \n rhospan: " + str(rhospan1))

    R_tov = []
    M_tov = []

    R_newt = []
    M_newt = []

    R_tov_zoom = []
    M_tov_zoom = []

    R_newt_zoom = []
    M_newt_zoom = []
```

```python

    # Ratkaise TOV jokaiselle rho0:lle rhospan alueessa.
    # //
    # Solve the TOV for each rho0 in the range of rhospan.
    for rho0 in rhospan1:
        r_tov, m_tov, p_tov, rho_tov = TOV_solver(ir=[],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
                p_c=0,
                a=3,
                eos_choise=2,
                tov_choise=2,
                interpolation=0,
                body="TOV White dwarf")
        r_boundary = r_tov[-1]
        m_boundary = m_tov[-1]
        R_tov.append(r_boundary)
        M_tov.append(m_boundary)

        r_newt, m_newt, p_newt, rho_newt = TOV_solver(ir=[],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
                p_c=0,
                a=3,
                eos_choise=2,
                tov_choise=3,
                interpolation=0,
                body="NEWT White dwarf")
        r_boundary = r_newt[-1]
        m_boundary = m_newt[-1]
        R_newt.append(r_boundary)
        M_newt.append(m_boundary)

    print("\n \n rhospan2: " + str(rhospan2))

    for rho0 in rhospan2:
        r_tov_zoom, m_tov_zoom, p_tov_zoom, rho_tov_zoom = TOV_solver(ir=[],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
                p_c=0,
                a=3,
                eos_choise=2,
                tov_choise=2,
                interpolation=0,
                body="TOV White dwarf")
        r_boundary = r_tov_zoom[-1]
        m_boundary = m_tov_zoom[-1]
        R_tov_zoom.append(r_boundary)
        M_tov_zoom.append(m_boundary)

        r_newt_zoom, m_newt_zoom, p_newt_zoom, rho_newt_zoom = TOV_solver(ir=[],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
                p_c=0,
                a=3,
                eos_choise=2,
```

```
                  tov_choise=3,
                  interpolation=0,
                  body="NEWT White dwarf")
           r_boundary = r_newt_zoom[-1]
           m_boundary = m_newt_zoom[-1]
           R_newt_zoom.append(r_boundary)
           M_newt_zoom.append(m_boundary)

       # Plottaa massa-säde - relaation.
       # //
       # Pplot the mass-radius relation.
       fig, ax1 = plt.subplots()
       axins1 = inset_axes(ax1, width='30%', height='40%', loc='lower right',
                           bbox_to_anchor=(-0.12, 0.3, 1.1, 1.2),
                           bbox_transform=ax1.transAxes)

       ax1.plot(R_tov, M_tov, color='b', label=fr'TOV')
       ax1.plot(R_newt, M_newt, color='red', label='Newtonilainen' , linestyle='--')
       ax1.set(# title="a)", title_position='left',
               xlabel=r'Säde, r ($R_{Earth}$)',
               ylabel= r'Massa, m ($M_{Sun}$)',
               xscale="linear", yscale="linear")
       ax1.set_title('d)', loc="left")
       ax1.legend(shadow=True, fancybox=False)
       ax1.grid()

       axins1.plot(R_tov_zoom, M_tov_zoom, color='b')
       axins1.plot(R_newt_zoom, M_newt_zoom, color='red', linestyle='--')
       axins1.set(# title="a)", title_position='left',
               xscale="linear", yscale="log")
       axins1.grid()

       plt.show()
       return

   MR_relaatio(2e-14+0j, 8e-10+0j, 8e-6+0j, 50, 75)
```

C Python, Teorioiden suhde

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Nov  18  2022

@author: Antero
"""
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec

from structure_equations import *
from TOV_solver_rho import *

def Relativistic_Terms():
        """
        Solves TOV-equations and newt-pressure twice for two white dwarves
        with different central energy density and plots the differences between
        these two theories.

        Returns
        -------
        None

        """

        # Määritellään ratkaistavien WD:en parametreja.
        # //
        # The parameters of the WDs to be solved are defined.

        # Pressure at the center.
        rho_center_thin =  2e-11+0j
        rho_center_dense = 2e-7+0j
        # Radius of WD with previous energy densities.
        skaala_thin = 1.286770048879275
        skaala_dense = 0.131694215476 74232
        # How close to the core we want to solve the equations again
        # to get good resolution for the differenve between theories.
        kerroin_thin = 0.9
        kerroin_dense = 0.9
        # Some formatting for graphs.
        rho_center_d_si = '{:0.2e}'.format(rho_center_dense.real * 2.0852e37)
        rho_thin_d_si = '{:0.2e}'.format(rho_center_thin.real * 2.0852e37)
        R_koko_thin = '{:0.2e}'.format(skaala_thin)
        R_koko_dense = '{:0.2e}'.format(skaala_dense)

        def Tov_Newt_suhde(k, s, rho0):
                """
                Solves the structure of WD with given core density.

                Parameters
                ----------
                k : float
                    How close to the core we want to be.
                s : float
                    Radius of wanted star for integration range.
                rho0 : float
                    Core density.

                Returns
                -------
                array
                    Returns the radius of stars and the ratio
                    between solutions of these theories.
                """
                r_tov, m_tov, p_tov, rho_tov = TOV_solver(ir=[5.067e12, k*(s/(
                1.9733e-16 * 1e-3 / 6371))],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
```

```
                p_c=0,
                a=3,
                eos_choise=2,
                tov_choise=2,
                interpolation=0,
                body="TOV White dwarf")

        r_newt, m_newt, p_newt, rho_newt = TOV_solver(ir=[5.067e12, k*(s/(1.9733e-16 * 1e-3 / 6371))],
                n=0,
                R_body=0,
                kappa_choise=0,
                rho_K=0,
                p_K=0,
                rho_c=rho0,
                p_c=0,
                a=3,
                eos_choise=2,
                tov_choise=3,
                interpolation=0,
                body="NEWT White dwarf")

        Delta_Rho = rho_newt[:-1] / rho_tov[:-1]
        return r_newt, Delta_Rho

    r_thin, rho_thin = Tov_Newt_suhde(kerroin_thin, skaala_thin, rho_center_thin)
    r_dense, rho_dense = Tov_Newt_suhde(kerroin_dense, skaala_dense, rho_center_dense)

    # Plotataan kuvat. // Plot all graphs.

    gs = gridspec.GridSpec(1, 2)
    plt.figure()

    ax1 = plt.subplot(gs[0, 0])
    ax2 = plt.subplot(gs[0, 1])

    ax1.plot(np.flip(r_dense[:-1])/skaala_dense, rho_dense, color='b',
        label=fr'Teorioiden välinen suhde keskipisteen parametreilla' '\n' fr'$\rho_{"c"}$ = {rho_center_d_si}' r' $\frac{\mathrm{J}}{\mathrm{m}^{3}}$' '\n' r'$R_{WD}$' fr' = {R_koko_dense}' r' $R_{Earth}$')
    ax2.plot(np.flip(r_thin[:-1])/skaala_thin, rho_thin, color='r',
        label=fr'Teorioiden välinen suhde keskipisteen parametreilla' '\n' fr'$\rho_{"c"}$ = {rho_thin_d_si}' r' $\frac{\mathrm{J}}{\mathrm{m}^{3}}$' '\n' r'$R_{WD}$' fr' = {R_koko_thin}' r' $R_{Earth}$')

    ax1.set(xlabel=r'Säde, r ($R_{WD}$)',
        ylabel= r'$\frac{\rho_{newt}}{\rho_{tov}}$',
        xscale="linear", yscale="log")
    ax1.set_title('e)', loc="left")
    ax1.legend(shadow=True, fancybox=True)
    ax1.grid()

    ax2.set(xlabel=r'Säde, r ($R_{WD}$)',
        ylabel=r'$\frac{\rho_{newt}}{\rho_{tov}}$',
        xscale="linear", yscale="log")
    ax2.set_title('f)', loc="left")
    ax2.legend(shadow=True, fancybox=True)
    ax2.grid()

    plt.show()
    return

Relativistic_Terms()
```

D Python, Määriteltyjä tilanyhtälöitä

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Oct  5  2022

@author: Antero
"""
import numpy as np
import scipy.constants as sc
import natpy as nat
"""
Määritetään Tolman-Oppenheimer-Volkoff - yhtälöt (m, p, EoS),
jotka ratkaisemalla saadaan kuvattua tähden rakenne.
//
Determine the Tolman-Oppenheimer-Volkoff equations (m, p, EoS),
by solving which the structure of the star can be described.
"""

def Mass_in_radius(rho, r):
    dmdr = 4*np.pi*rho*r**2
    return dmdr


def EoS_cd_p2rho(interpolation, p_point):
    """
    Equation of state from custom interpolated (P, RHO)-data.

    Parameters
    ----------
    interpolation : Interpolate function.
        Returns rho when given p.
    p_point : Float, Array
        Pressure of a astrophysical body.

    Returns
    -------
    Float, Array
        Returns RHO as a function of P.

    """
    return interpolation(p_point)


def EoS_r2p(rho, Gamma, Kappa):
    """
    Equation of state, EoS.

    Given the energy density and constants returns the pressure.

    Parameters
    ----------
    rho : Float
        Energy density.
    Gamma : Float
        Polytrope constant.
    Kappa : Float
        Constant of proportionality.

    Returns
    -------
    p : Float
        Pressure.

    """
    p = Kappa*rho**Gamma
    return p


def EoS_p2r(p, Gamma, Kappa):
    """
    Equation of state, EoS.

    Given pressure and constants returns the energy density.
```

```python

    Parameters
    ----------
    p : Float
        Pressure.
    Gamma : Float
        Polytrope constant.
    Kappa : Float
        Constant of proportionality.

    Returns
    -------
    rho : Float
        Energy density.

    """
    rho = (p/Kappa)**(1/Gamma)
    return rho


def EoS_degelgas(rho):
    """
    Degenerate electron gas equation of state.

    Parameters
    ----------
    rho : float
        Pressure.

    Returns
    -------
    float
        Pressure from given energy density.
    """
    # Tilanyhtälön vakkioita // Equation of state constants
    m_e = nat.convert(sc.electron_mass * nat.kg, nat.GeV).value
    m_p = nat.convert(sc.proton_mass * nat.kg, nat.GeV).value
    a = (m_e**4)/(3*np.pi**2)
    b = ((3*np.pi**2)/(2*m_p*m_e**3))**(1/3)
    # Tilanyhtälö // Equation of state
    x = lambda rho : b*(rho)**(1/3)
    f = lambda x : (1/8)*(x*(1+x**2)**(1/2)*(2*x**2-3)+3*np.log(x+(1+x**2)**(1/2)))
    return a*f(x(rho))


def Eos_degelgas_deriv(rho):
    """
    Degenerate electron gas equation of state derivate.
    dp/drho.

    Parameters
    ----------
    rho : float
        Energy density.

    Returns
    -------
    float
        Derivate value.
    """
    # Vakioita // Constants
    m_e = nat.convert(sc.electron_mass * nat.kg, nat.GeV).value
    m_p = nat.convert(sc.proton_mass * nat.kg, nat.GeV).value
    a = (m_e**4)/(3*np.pi**2)
    b = ((3*np.pi**2)/(2*m_p*m_e**3))**(1/3)
    # Derivaatta // derivate
    x = lambda y: b*y**(1/3)
    dpdrho = (a*b)/(3*rho**(2/3))*((x(rho)**4)/((1+x(rho)**2)**(1/2)))
    return dpdrho


def EoS_choiser(choise, interpolation=0., Gamma=0., Kappa=0., p=0., rho=0.):
```

```python
    """
    Chooses wanted EoS for DE-group as rho and returns it
    with given parameters.

    Parameters
    ----------
    choise : Int
        Chooses between defined EoS to return.
        Can be 0, 1 or 2.
    interpolation : args
    p : args
    Gamma : argsz
    Kappa : args

    Returns
    -------
    returnable : Float, Array
        Energy density with calculated with wanted EoS and given params.

    """
    if choise == 0:
        returnable = EoS_p2r(p, Gamma, Kappa) # Energy density
    if choise == 1:
        # TODO add interpolate functions and params
        returnable = interpolation # EoS_cd_p2rho(interpolation, p) # Energy density
    if choise == 2:
        returnable = EoS_degelgas(rho) # Pressure
    return returnable


def TOV_choiser(choise, m=0., p=0., rho=0., r=0.):
    """
    Chooses between different tov functions.

    Parameters
    ----------
    choise : int
        Chooses function:
            choise=0 dpdr tov
            choise=1 dpdr newt
            choise=2 drhodr tov
            choise=3 drhodr newt
    m : float
        Mass.
    p : float
        Pressure.
    rho : float
        Energy density.
    r : float
        Radius.

    Returns
    -------
    float
        Value for chosen hydrostatic equilibrium equation deriv.
    """
    # Gravitational constant in GeV^-2.
    G = 6.707113e-39
    if choise == 0:
        # dpdr tov
        tov = -(rho+p)*(m + 4*np.pi*r**3*p)/(r*(r-2*m))
    if choise == 1:
        # dpdr newt.
        tov = -(m*rho)/(r**2)
    if choise == 2:
        # drhodr tov
        tov = -(((rho+p)*(G*m + 4*np.pi*G*r**3*p))/(r*(r-2*G*m)))*(Eos_degelgas_deriv(rho))**(-1)
    if choise == 3:
        # drhodr newt.
        tov = (-(G*m*rho)/(r**2))*(Eos_degelgas_deriv(rho))**(-1)
    return tov
```