Functional brain segmentation using inter-subject correlation in fMRI — Supplementary material

1. Supplementary tables

Table 1: Region names of the Harvard-Oxford cortical brain atlas together with their abbreviations. In the text and figures we additionally use the shorthand notations "L" and "R" to denote the left and right sides of the brain.

Abbr	Brain region
ACC	Cingulate Gyrus, anterior division
AngG	Angular Gyrus
CntrOper	Central Opercular Cortex
Cuneus	Cuneal Cortex
FMC	Frontal Medial Cortex
FOper	Frontal Operculum Cortex
FrontPole	Frontal Pole
FrOrbC	Frontal Orbital Cortex
HeschlG	Heschl's Gyrus (includes H1 and H2)
IFGop	Inferior Frontal Gyrus, pars opercularis
IFGtr	Inferior Frontal Gyrus, pars triangularis
Ins	Insular Cortex
intCal	Intracalcarine Cortex
ITGant	Inferior Temporal Gyrus, anterior division
ITGpos	Inferior Temporal Gyrus, posterior division
ITGto	Inferior Temporal Gyrus, temporooccipital part
LingualG	Lingual Gyrus
LOCinf	Lateral Occipital Cortex, inferior division
LOCsup	Lateral Occipital Cortex, superior division
MFG	Middle Frontal Gyrus
MTGant	Middle Temporal Gyrus, anterior division
MTGpos	Middle Temporal Gyrus, posterior division
MTGto	Middle Temporal Gyrus, temporooccipital part
OccipPole	Occipital Pole
OFuG	Occipital Fusiform Gyrus
ParaCingC	Paracingulate Gyrus
PCgG	Cingulate Gyrus, posterior division
PHGant	Parahippocampal Gyrus, anterior division
PHGpos	Parahippocampal Gyrus, posterior division
PlanTemp	Planum Temporale
PoG	Postcentral Gyrus
POper	Parietal Operculum Cortex
PPolare	Planum Polare
PreCentG	Precentral Gyrus
PreCun	Precuneous Cortex
SCA	Subcallosal Cortex
SFG	Superior Frontal Gyrus
SMA	Juxtapositional Lobule Cortex (formerly Supplementary Motor Cortex)
SMGant	Supramarginal Gyrus, anterior division
SPL	Superior Parietal Lobule
STGant	Superior Temporal Gyrus, anterior division
STGpos	Superior Temporal Gyrus, posterior division
SupraCalc	Supracalcarine Cortex
SupramGpos	Supramarginal Gyrus, posterior division
TFuant	Temporal Fusiform Cortex, anterior division
TFupos	Temporal Fusiform Cortex, posterior division
tmp	Temporal Pole
TOFuG	Temporal Occipital Fusiform Cortex

Table 2: Locations and sizes of the clusters found for the StudyForrest data. If cluster comprises of spatially disjoint subclusters, information is listed separately for the largest subclusters in the decreasing order of their size. The following information is listed for each cluster/subcluster: 1) the brain region name where the center of mass (COM) of the cluster is located (if available), 2) the coordinate of COM (in MNI space), 3) the most representative cortical brain region covered by the cluster (in terms of the number of voxels), and 4) cluster size (in terms of the total number of voxels).

Cluster index	**COM area**	**COM coordinate**	**Cortical area**	**Size**
1 (1)		58.8 −29.0 5.0	PlanTemp R	4567
(2)	PlanTemp L	−53.8 −28.2 6.5	PlanTemp L	4257
2 (1)		55.2 −33.5 8.0	AnG R	7045
(2)	PlanTemp L	−50.1 −30.3 9.4	POper L	5455
(3)	IFGop R	51.9 12.4 14.9	IFGop R	436
3 (1)	PreCun R	10.9 −61.6 25.8	PreCun R	6130
4 (1)	IFGtr L	−40.5 32.8 16.8	MFG L	2706
(2)	HeschlG L	−50.3 −23.4 10.4	CntrOper L	2313
(3)	PlanTemp R	50.3 −28.8 11.8	SupramGpos R	2000
(4)	IFGtr R	44.2 31.6 15.1	FrontPole R	1919
(5)	PCgG R	5.7 −43.4 27.4	PCgG R	790
(6)	LOCinf R	53.4 −62.5 −2.7	LOCinf R	717
(7)	PreCun L	−7.1 −70.7 31.7	PreCun L	460
5 (1)	AnG L	−48.5 −54.9 16.5	AnG L	2514
(2)	LOCsup R	52.1 −63.5 22.3	LOCsup R	2246
(3)	IFGop R	50.6 18.9 10.4	IFGop R	894
(4)	MTGpos R	61.4 −32.9 −5.7	MTGpos R	548
(5)	STGant L	−52.8 −8.6 −11.3	MTGpos L	526
(6)	ParaCingC L	−7.8 41.1 −9.6	ParaCingC L	448
(7)	LOCsup L	−37.2 −79.3 27.7	LOCsup L	409
6 (1)		21.1 36.0 17.3	FrontPole R	5288
(2)		−28.5 39.7 15.1	FrontPole L	4655
(3)	FOper L	−38.8 8.5 8.6	Ins L	1561
(4)	Ins R	33.9 9.2 2.5	Ins R	1340
7 (1)	SupramGpos L	−52.2 −47.3 11.8	AnG L	3402
(2)	IFGop R	49.4 13.3 12.1	IFGop R	2869
(3)	MTGto R	54.2 −38.5 −1.3	MTGpos R	1590
(4)	IFGop L	−52.3 14.4 15.0	IFGop L	1475
(5)	ParaCingC R	2.8 41.6 22.3	ParaCingC R	1086
(6)	AnG R	56.6 −51.6 31.3	AnG R	546
8 (1)	LOCsup L	−38.7 −66.8 27.0	LOCsup L	2490
(2)	PreCun R	19.2 −58.8 35.4	PreCun R	1442
(3)		−12.6 36.5 −8.0	ParaCingC L	988
(4)	LOCsup R	44.9 −69.2 25.3	LOCsup R	576
(5)	MTGpos L	−53.9 −13.3 −13.7	MTGpos L	407
(6)	TOFuG R	27.7 −45.0 −9.5	LingualG R	287
(7)	PPolare R	44.1 −17.1 −4.1	PPolare R	269
9 (1)	CntrOper L	−46.9 −15.0 18.2	CntrOper L	3978
(2)	CntrOper R	57.1 −7.0 13.4	CntrOper R	988
(3)	OccipPole L	−18.6 −93.1 −3.0	OccipPole L	932
(4)	Ins R	36.4 −0.8 7.2	Ins R	545
(5)	ParaCingC R	6.4 51.4 12.1	ParaCingC R	412
(6)	PreCun R	3.5 −52.2 11.4	PreCun R	352
(7)	MTGto L	−53.1 −61.6 −1.2	LOCinf L	333
10(1)	OccipPole R	5.6 −91.3 2.7	OccipPole R	9508
11(1)		16.4 33.4 13.3	FrontPole R	9000
(2)		−43.7 9.8 12.6	IFGop L	2549
(3)	MTGto L	−50.8 −58.9 7.0	MTGto L	589
(4)		48.4 −0.8 26.7	PreCentG R	405
(5)	MTGto R	52.1 −58.5 8.1	MTGto R	386
(6)	LOCsup L	−42.1 −75.3 17.8	LOCsup L	324
(7)	LingualG R	25.3 −46.2 −10.9	LingualG R	296
12(1)		23.8 30.3 −5.8	FrOrbC R	1724
(2)	MTGpos L	−58.3 −40.2 −3.9	MTGpos L	1013
(3)	ParaCingC R	6.3 53.8 14.7	FrontPole R	896
(4)	ITGto R	50.9 −52.6 −8.8	ITGto R	847
(5)	LOCinf L	−40.9 −72.3 9.3	LOCinf L	838
(6)	IFGop L	−48.6 19.5 2.3	IFGop L	675
(7)	LOCsup R	45.0 −70.5 21.2	LOCsup R	668
13(1)	ParaCingC L	0.0 48.6 2.5	FrontPole L	6671

Table 3: Locations and sizes of the clusters found for the ICBM data. If cluster comprises of spatially disjoint subclusters, information is listed separately for the largest subclusters in the decreasing order of their size. The following information is listed for each cluster/subcluster: 1) the brain region name where the center of mass (COM) of the cluster is located (if available), 2) the coordinate of COM (in MNI space), 3) the most representative cortical brain region covered by the cluster (in terms of the number of voxels), and 4) cluster size (in terms of the total number of voxels).

Cluster index	**COM area**	**COM coordinate**	**Cortical area**	**Size**
1 (1)		2 −90 4	OccipPole L	2031
2 (1)	OccipPole R	30 −92 0	OccipPole R	950
(2)	OccipPole L	−24 −94 −4	OccipPole L	896
3 (1)	IFGop L	−48 14 20	PreCentG L	3437
(2)	SMA L	−2 6 58	SMA L	619
(3)	MTGto L	−54 −58 −4	MTGto L	328
(4)	PreCun R	6 −60 36	PreCun R	250
4 (1)	SupraCalc R	2 −84 6	OccipPole R	3386
5 (1)	LingualG R	2 −70 0	LOCsup L	5991
6 (1)	LOCinf L	−42 −68 −6	LOCinf L	1800
(2)		42 −62 −8	LOCinf R	1692
(3)	SPL L	−44 −42 52	SPL L	521
7 (1)	MFG R	44 20 28	MFG R	1225
(2)	ACC L	0 44 0	ParaCingC R	1034
(3)	MFG L	−40 30 20	FrontPole L	746
(4)	LOCsup R	42 −58 46	LOCsup R	742
(5)	ParaCingC L	0 18 42	ParaCingC R	531
(6)		−36 −58 42	LOCsup L	516
(7)	PCgG L	−4 −54 30	PreCun L	445
8 (1)		50 −8 18	FrontPole R	4728
(2)		−34 −6 22	FrontPole L	3900
(3)	PreCun R	4 −52 36	PreCun R	2893
(4)		26 −74 −28	ITGto R	348
(5)		28 −72 −48	ITGto R	285
9 (1)	PreCun R	10 −66 38	LOCsup R	5663
(2)	PreCentG R	44 2 38	PreCentG R	1180
(3)		−14 −2 56	PreCentG L	884
10(1)	Cuneus L	−2 −80 22	LOCsup L	3253
11(1)	PlanTemp L	−54 −20 4	STGpos L	4946
(2)	STGpos R	56 −22 2	STGpos R	2858
(3)	SFG L	−6 16 62	SFG L	281
(4)	MFG L	−40 8 50	MFG L	270
12(1)		0 18 6	ParaCingC L	5174
(2)		38 28 22	FrontPole R	4153
(3)		0 −70 −26	OccipPole L	2081
(4)	LOCsup L	−42 −62 36	LOCsup L	1170
(5)	PCgG L	−2 −38 38	PCgG L	1132
(6)		−34 38 18	FrontPole L	715
(7)		38 −58 36	LOCsup R	620
13(1)	LingualG L	−2 −62 6	LingualG L	5222
14(1)	PreCentG L	−4 −30 54	PoG L	9443
(2)	PreCentG R	54 2 16	PreCentG R	437
(3)		12 −56 −16	LingualG R	403
(4)		22 −52 −48	LingualG R	384
(5)	LOCinf R	48 −62 8	LOCinf R	332

2. Performance evaluation

We used the adjusted rand index (ARI) (Hubert and Arabie, 1985) to assess the stability and performance of clustering. ARI compares two partitions/segmentations U and V based on information of the contingency table (see Table 4). In the contingency table, the partition U consists of R clusters $u_1, u_2, \ldots, u_R$ and the partition V consists of C clusters $v_1, v_2, \ldots, v_C$. Elements n_{ij} are the numbers of voxels belonging to clusters u_i and v_j, $n_{i.}$ are the row sums of the table, $n_{.j}$ are the column sums of the table, and n is the total number of data points (voxels) in the data set. Using these notations, ARI compares two partitions as follows:

$$ARI = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}. \quad (1)$$

Table 4: Illustration of the contingency table for comparing two partitions

Cluster	v_1	v_2	...	v_C	Sums
u_1	n_{11}	n_{12}	...	n_{1C}	$n_{1.}$
u_2	n_{21}	n_{22}	...	n_{2C}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
u_R	n_{R1}	n_{R2}	...	n_{RC}	$n_{R.}$
Sums	$n_{.1}$	$n_{.2}$	...	$n_{.C}$	n

The expected value of ARI is 0, indicating that the agreement between two partitions is the same as between two random labelings of the data. The maximum value of ARI is 1, indicating identical partitions.

As our second performance measure, we used the Dice index (Dice, 1945) to define common spatial occurrence of two clusters. We used the Dice index as a similarity measure when we matched individual clusters (u_i and v_j) between two segmentations using the Munkres assignment algorithm (Munkres, 1957). For the algorithm, the Dice index between every cluster pair (u_i,v_j) was formed in the following way. First, we reshaped segmentation maps U and V into n-dimensional vectors, where n corresponds to the total number of voxels in the fMRI data. Then, for each cluster i (or j), we constructed binary vectors B_{u_i} (or B_{v_j}), where the element of the vector was 1 if the corresponding voxel belonged to cluster u_i (or v_j), and was 0 otherwise. Then we computed the Dice index as follows:

$$I_{Dice}(u_i, v_j) = \frac{2\sum_{l=1}^{n}\left(B_{u_i}[l] \cdot B_{v_j}[l]\right)}{\sum_{l=1}^{n} B_{u_i}[l] + \sum_{l=1}^{n} B_{v_j}[l]}, \tag{2}$$

where $B_{u_i}[l]$ (or $B_{v_j}[l]$) denotes lth voxel in the binarized vector. The resulting Dice index values vary between 0–1, where 1 denotes the exact similarity and 0 denotes no overlap between the clusters.

3. Validation of proposed cluster detection algorithm

Here, we validate our cluster detection algorithm presented in Appendix of the main article and compare its performance with existing algorithms using synthetic data sets. For this purpose, We generated data sets comprising of 40 spherical clusters drawn from 2- and 10-dimensional Gaussian distributions. The mean vectors of the distributions were drawn randomly from a uniform distribution (the range for 2-dimensional data was between –5 and 5 and the range for 10-dimensional data was between –1 and 1). The standard deviation of each cluster was 0.2 and the number of data points was 50. We also corrupted some data sets with spurious data (outliers). Spurious data points were drawn randomly from a uniform distribution over the same range as the mean vectors. For better assessment of the clustering quality, we controlled the separation of the clusters by ensuring that there were no data points within 2σ -tolerance regions of the clusters.

We compared the performance of our method (called SNN) against the following clustering algorithms: K-means (MacQueen, 1967), K-means++ (Arthur and Vassilvitskii, 2007), Farthest first traversal algorithm (Hochbaum and Shmoys, 1985, Gonzalez, 1985), Ward's minimum variance method (Ward, 1963), and affinity propagation (AP) (Frey and Dueck, 2007). For all the tested methods, we used the Euclidean distance as a dissimilarity measure. We used the minimum SSE criterion with all the tested methods except with farthest first traversal, which minimizes the maximum distance between points and cluster centers. Because the solutions of K-means and K-means++ algorithms are dependent on the initial selection of cluster centers, we run these algorithms 100 times using different random initializations and selected the solution with the minimum SSE. For Farthest first traversal, we tested each possible traversal, meaning that the total number of initializations corresponded to the total number of observations in the data. The Ward's minimum variance method and the AP algorithm require that the full (dis)similarity matrix is available, restricting their use only for relatively small data sets. For large data sets, the sparse (leveraged) version of the AP algorithm has been proposed, which samples from the full set of potential similarities and performs several rounds of the algorithm for resulting sparse graphs[1]. In practice, sparse version of AP is necessary for whole-brain fMRI data analysis and therefore we investigated its performance.

While comparing different algorithms, we assumed that the total number of clusters is known to make the comparison between methods more straightforward. Thus, for the K-means, K-means++, Farthest first traversal, and Ward's minimum variance method, we fixed the total number of clusters for K=40. Unlike these methods, AP and our SNN method estimate the total number of clusters indirectly from intrinsic properties of the data sets based on the preference parameter p and neighborhood size k, respectively. For these two methods, we run them using several parameters and selected the result having

[1] See: http://www.psi.toronto.edu/affinitypropagation/faq.html

the closest match with the actual number of clusters.

We wrote a customized code for our method using Matlab and C programming languages. For the K-means and Ward's minimum variance methods, we used the implementations of the Statistics Toolbox of the Matlab. For the K-means++, we used the efficient Matlab implementation by Laurent Sorber[2]. For the AP, we used the efficient C-language implementation provided by Frey and Dueck (2007).

Figure 1(A) shows the results for the 2-dimensional data sets when there were no outliers present. The results of the methods are organized in the decreasing order of the clustering quality. Clearly, SNN, AP, Sparse AP and Ward's method provided nearly perfect partition of the data sets. The K-means and K-means++ algorithms provided somewhat lower results although these methods are theoretically optimal for detecting well-separated spherical Gaussian clusters. This indicates that in practice these methods are highly sensitive for initial placement of the centroids even when the total number random initializations is high. The worst result was obtained with the Farthest first traversal – however, this is not surprising as the method was not designed for minimizing the SSE.

Figure 1(B) shows the corresponding results as a function of outliers in the data. Clearly, the Ward's method and the Farthest first were most sensitive for the effect of outliers. Interestingly, SNN, AP and Sparse AP provided very high-quality results even in the presence of high number of outliers.

Figure 1(C) shows the results for 10-dimensional data as a function of the number of data points. Again, SNN and AP provided excellent results. The result of the Sparse AP was very high for small data sets, but started to degrade once the size of the data set was increased. This is natural because in Sparse AP only a small subset of data points is used to run AP algorithm which of course cannot explain details in large data sets with sufficient accuracy.

Figure 1(D) shows the computation times of the methods for the 10-dimensional data as function of the number of data points. The fastest methods were SNN and Ward's method. The K-means and K-means++ were somewhat slower which can be explained by the high number of initializations (100) used. AP, Sparse AP and Farthest first were very slow when compared with the SNN and Ward's method. Slow computation times of the Farthest first can be explained by the high number of initializations used.

Overall, our method and AP outperformed the other methods in terms of clustering quality. In fact, both methods provided highly consistent and similar results in our tests. Figure 2 shows some examples of the estimated cluster centroids from synthetic 2-dimensional data sets. However, the sparse version of the AP, which would be required to analyze large fMRI data sets, turned out to be less accurate than our method and original AP algorithm. Moreover, our method was superior against both the original and sparse AP in terms of computation time. Thus, we integrated our method with the FuSeISC.

[2]See `http://www.mathworks.com/matlabcentral/fileexchange/28804-k-means++/content/kmeans.m`)

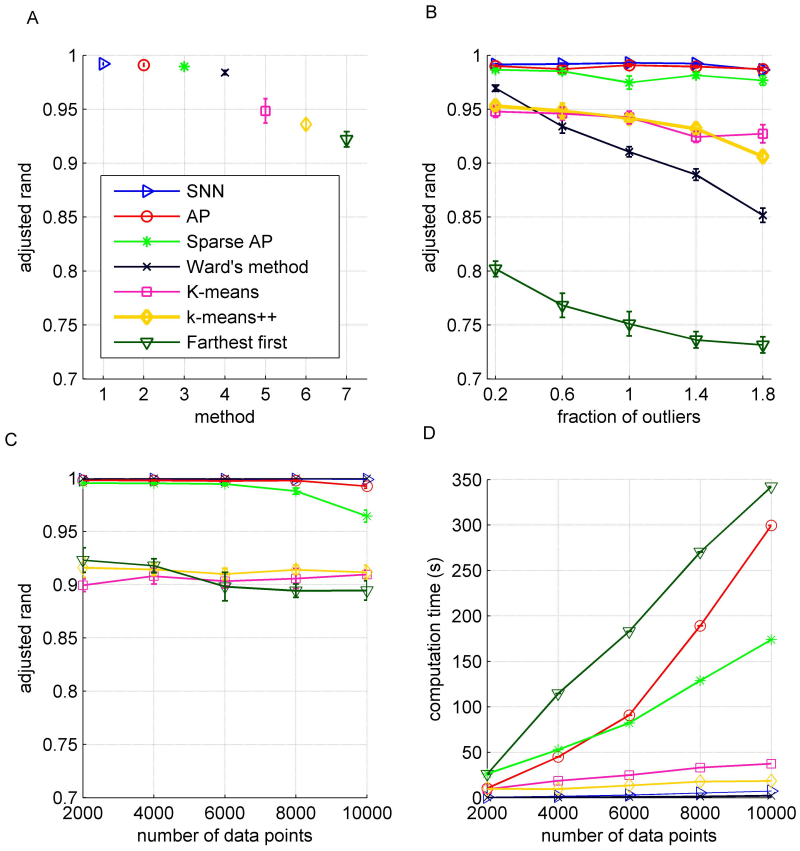

Figure 1: Performance comparison of the seven clustering algoritms: our SNN based method (denoted by "SNN"), Affinity propagation (denoted by "AP"), Sparse affinity propagation (denoted by "Sparse AP"), Ward's minimum variance method (denoted by "Ward's method"), K-means, K-means++, and farthest first traversal (denoted by "Farthest first"). For each method, the average results across ten realizations are shown together with the standard error bars. (A) clustering quality for 2-dimensional data containing 40 spherical clusters, (B) the corresponding results after corrupting the data with outliers, (C) clustering quality for 10-dimensional data involving 40 clusters as a function of the sample size, and (D) the corresponding computation times.

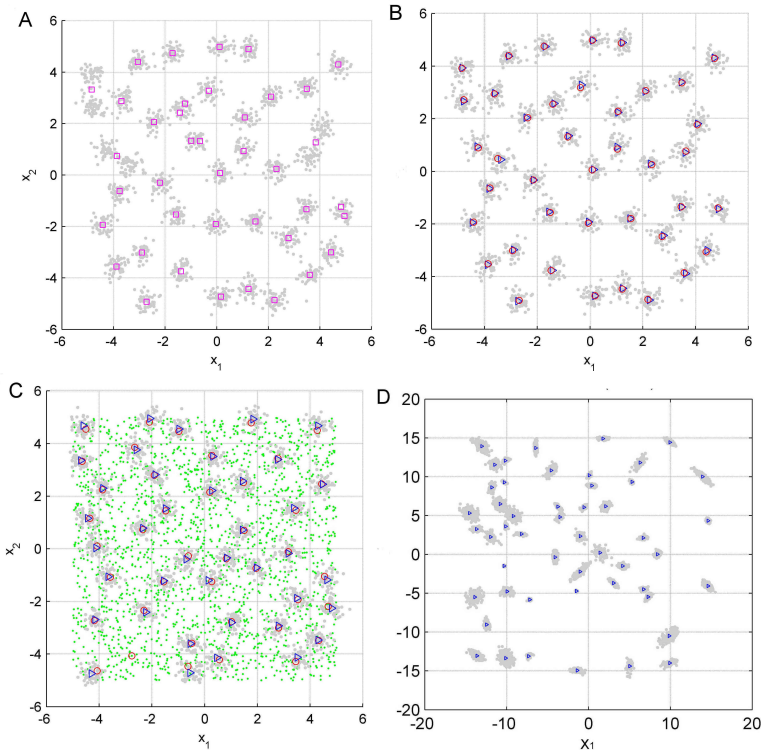

Figure 2: Estimated cluster centroids for 2-dimensional synthetic data sets containing 40 Gaussian clusters: (A) The centroids of the best K-means result among 100 random initializations. Despite of high number of initializations, K-means failed to detect all clusters correctly. (B) The centroids of our method (blue triangle) and AP (red circle) for the same data set. Both methods detected all the clusters correctly. (C) The corresponding results when spurious data points (shown in green color) were present. Even in this case, SNN and AP were able to find the clusters. (D) The centroids of our method for the data set when clusters had arbitrary non-spherical shapes, and therefore SSE criterion was replaced with BIC. The estimation of centroids was highly successful also in this case (the mean ARI across 10 realizations was 0.99).

4. Effect of neighborhood size k

In practice, our initialization procedure requires that the neighborhood size k is selected *a priori*. Here we present simulation results for different values of k and discuss the choice of k. Figure 3(A) shows the results of the SNN for the 2-dimensional data as a function of the neighborhood size k. As can be seen, the clustering quality is nearly perfect when k is chosen between 20 and 45. This is natural because the local neighborhood is somewhat smaller than the number of data points in the clusters and the SNN graph can thus capture very well the variations within each cluster. This result indicates that to detect all the clusters in data, k should be chosen slightly lower than the smallest cluster size of interest in the data set. If the minimum cluster size is not known, we can plot the total number of clusters found as a function of k (see Fig. 3(B)). Clearly, there is a stable region in the number of clusters: by choosing any of the solutions within this region we can recover a correct clustering result. We use this heuristic with real data sets to find a good choice for k.

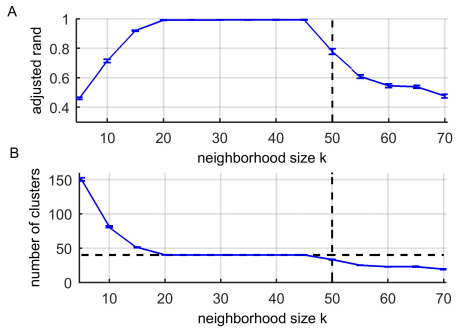

Figure 3: Clustering results and the estimated number of clusters for our SNN based method as a function of neighborhood size parameter k: (A) Clustering quality, (B) the total number of estimated clusters. By comparing the two curves, it can be seen that the solution is near perfect when k is chosen from the stable region (the range between k=20 and k=45) in the curve shown in (B). The vertical line denotes the cluster sizes and the horizontal line corresponds to the total number of clusters.

5. Effect of model selection criterion

Here we compare results obtained with two model selection criteria, SSE and BIC, with simulated fMRI data. Figure 4(A) presents the performance of the functional segmentation for the simulated ICBM data as a function of the neighborhood parameter k. Two curves are shown, one for the SSE and one for the BIC criterion used in the selection of the candidate graph. For a wide range of parameters, ARI values resulted in "moderate agreement" (ARI between 0.4–0.6) between the ground truth and the estimated cluster labeling computed across the 72,577 voxels. The difference in ARI values between the two criteria is relatively minor for most solutions. The exceptions are the largest values of k, which show higher performance for the SSE criterion. Figure 4(B) shows the total number of clusters found by the two criteria. As we expected, the number of clusters reduces as a function of chosen resolution (k). Especially the curve of the SSE shows stabilization in the number of clusters as a function of k (for $k \geq 175$). Based on these results, we concluded that the SSE criterion to select the best candidate SNN graph is suitable for the analysis of fMRI data.

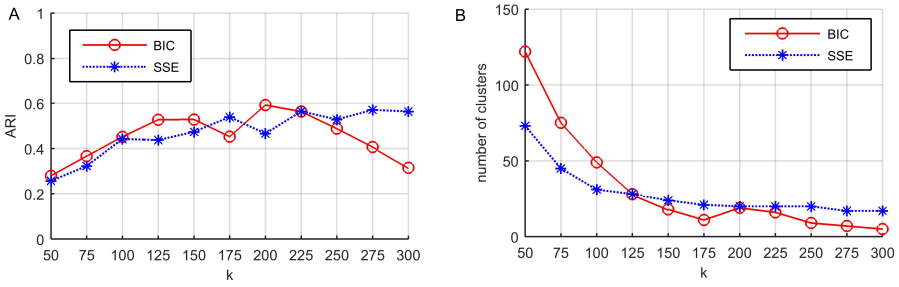

Figure 4: The results of FuSeISC for the simulated ICBM data: (A) segmentation quality, and (B) the total number of clusters. The results are plotted as a function of k for the two criteria, SSE and BIC, used in the selection of the candidate SNN graph.

6. Clusters detected as noise

Fig. 5 shows the number of voxels within the noise mask for each cluster. The noise mask consisted of cerebral white-matter, brainstem, and ventricles. Figs. 6 and 7 show the discarded clusters over an anatomical image as well as their ISC features, together with the retained clusters. The discarded clusters of the StudyForrest data were spatially fragmented around the white-matter area. In contrast, the white-matter area in the ICBM data was segmented into a single huge cluster. Clearly, the discarded clusters of the ICBM data could have also been detected by their very low ISC mean values. For the StudyForrest data, some—but not all—noise clusters could have been detected just by their ISC features (low ISC mean or high ISC variability).

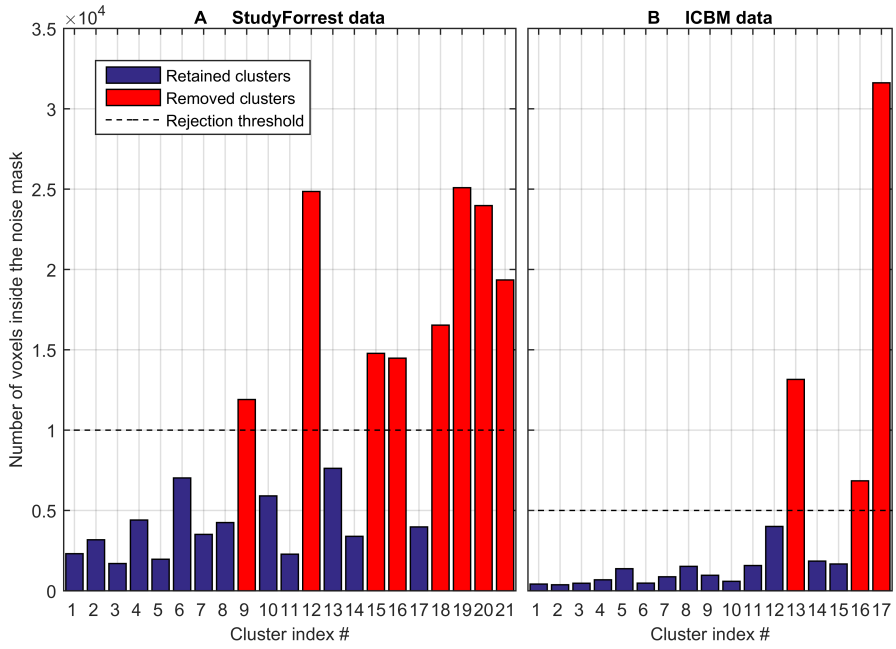

Figure 5: Total number of voxels within the noise mask for each cluster: (A) StudyForrest, and (B) ICBM data. The bar graph shows the total number of voxels for each cluster within the noise mask consisting of ventricles, white-matter area and brainstem. Clusters having the highest numbers of voxels within this mask were discarded as noise. The exact number of discarded clusters was determined based on visual inspection of the spatial distributions of the clusters so that the clusters mainly distributed close or inside the noise mask were discarded.

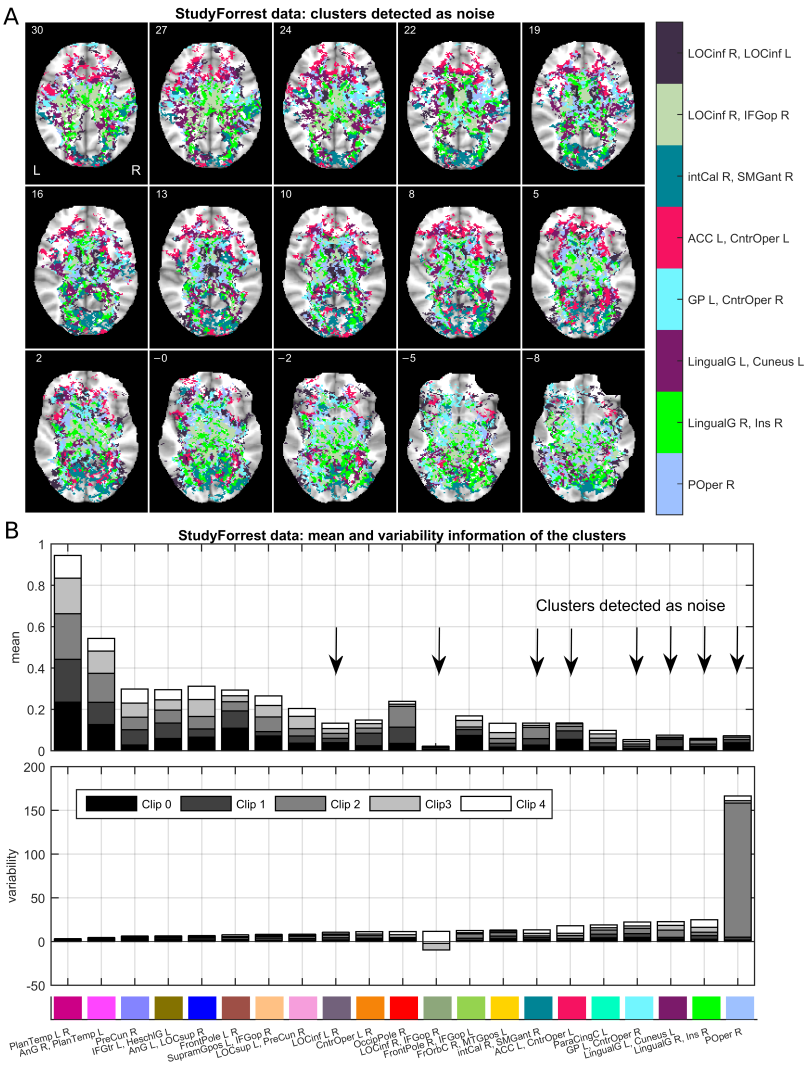

Figure 6: Investigation of the discarded clusters from the StudyForrest data: (A) spatial maps of the discarded clusters, and (B) ISC mean and variability information of all clusters in the increasing order of relative ISC variability. The discarded clusters are denoted by vertical arrows.

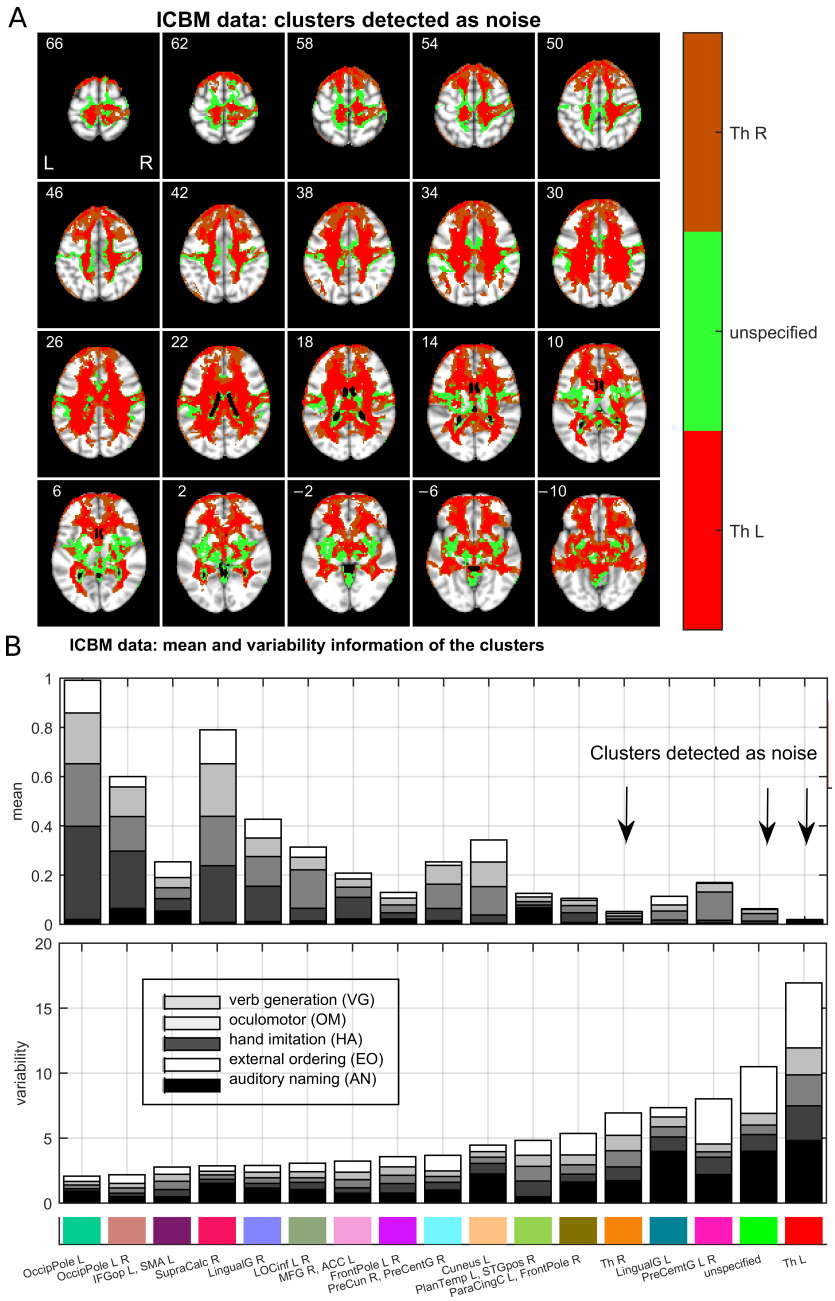

Figure 7: Investigation of the discarded clusters from the ICBM data: (A) spatial maps of the discarded clusters, and (B) ISC mean and variability information of all clusters in the increasing order of relative ISC variability. The discarded clusters are denoted by vertical arrows.

7. Mean and variability features of the StudyForrest data

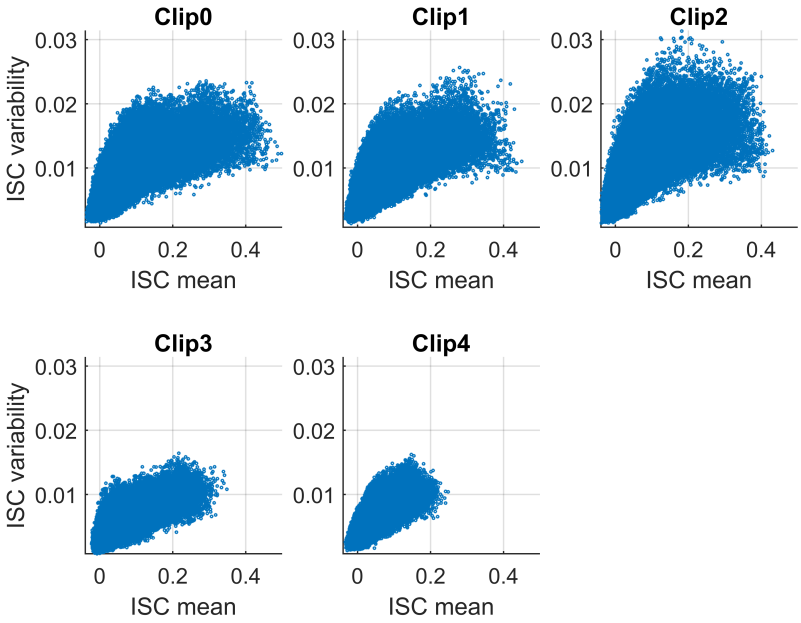

Figure 8: Scatter plots of the ISC mean and variability features of the StudyForrest data for each time series (Clips0–CLips4). The number of data points in each plot corresponds to the number of voxels within the brain (449,612). The ISC variability tends to increase together with the ISC mean.

8. Spatial maps of different ICBM data sets

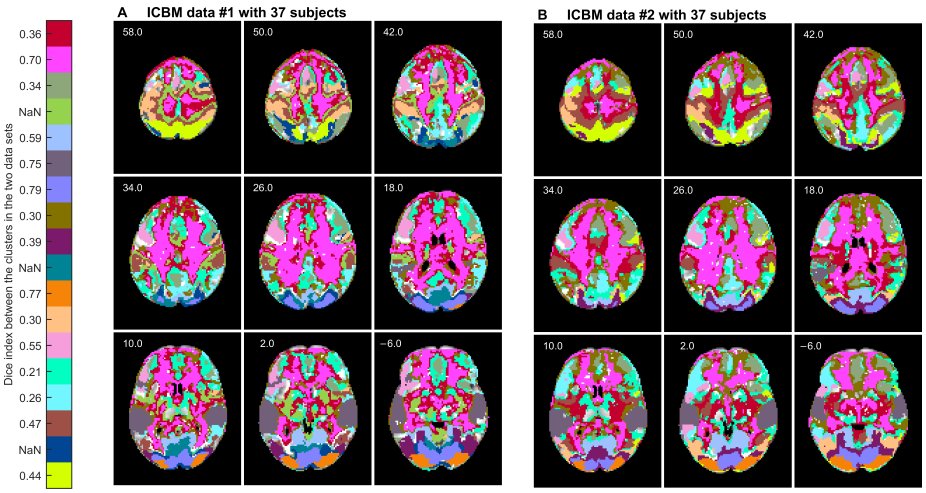

Figure 9: Raw functional segmentation results ($k = 250$) of two real ICBM data set with 37 different subjects. Clusters in the two data sets are matched using the Munkres assignment algorithm. Similarity between the clusters according to the Dice index is shown next to the color bar, "NaN" meaning that the corresponding cluster is present only in the leftmost data set.

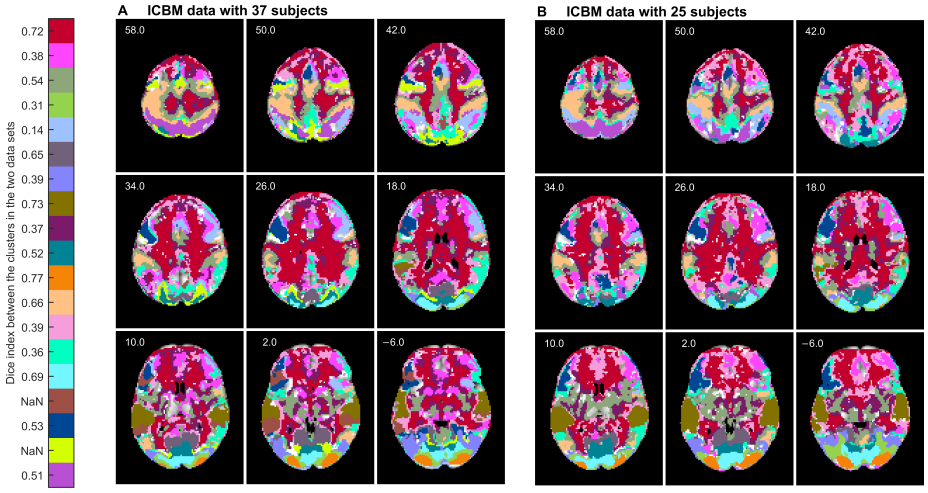

Figure 10: Raw functional segmentation result ($k = 225$) of (A) real ICBM data set with 37 subjects, and (B) real ICBM data set with 25 subjects. Clusters in the two data sets are matched using the Munkres assignment algorithm. Similarity between the clusters according to the Dice index is shown next to the color bar, "NaN" meaning that the corresponding cluster is present only in the leftmost data set.

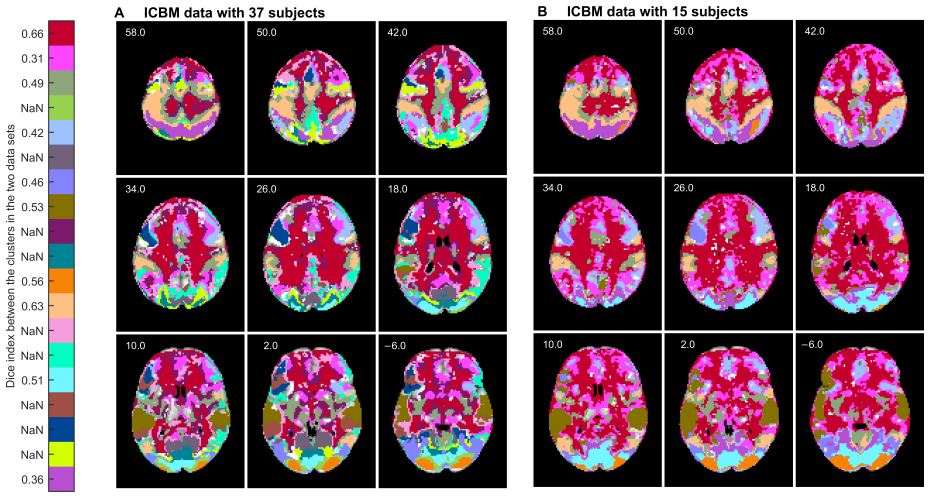

Figure 11: Raw functional segmentation result ($k = 225$) of (A) real ICBM data set with 37 subjects, and (B) real ICBM data set with 15 subjects. Clusters in the two data sets are matched using the Munkres assignment algorithm. Similarity between the clusters according to the Dice index is shown next to the color bar, "NaN" meaning that the corresponding cluster is present only in the leftmost data set.

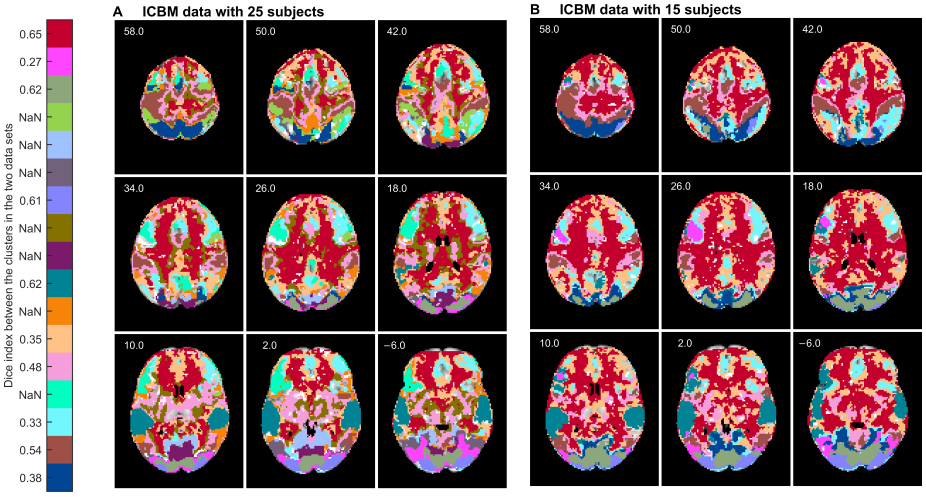

Figure 12: Raw functional segmentation result ($k = 225$) of (A) real ICBM data with 25 subjects, and (B) real ICBM data set with 15 subjects. Clusters in the two data sets are matched using the Munkres assignment algorithm. Similarity between the clusters according to the Dice index is shown next to the color bar, "NaN" meaning that the corresponding cluster is present only in the leftmost data set.