Ensembles of models in fMRI : stable learning in large-scale settings

Andrés Hoyos-Idrobo

▶ **To cite this version:**

Andrés Hoyos-Idrobo. Ensembles of models in fMRI : stable learning in large-scale settings. Medical Imaging. Université Paris Saclay (COmUE), 2017. English. NNT : 2017SACLS029 . tel-01526693

HAL Id: tel-01526693

https://theses.hal.science/tel-01526693v1

Submitted on 23 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

NNT: 2017SACLS029

Thèse de Doctorat
de
l'Université Paris-Saclay
préparée à
l'Université Paris-Sud

École Doctorale n°580
Sciences et Technologies de l'Information et de la communication

Spécialité de doctorat : Informatique

Par

M. Andrés Hoyos-Idrobo

Ensembles des modeles en fMRI: l'apprentissage stable à grande échelle

Thèse présentée et soutenue à Palaiseau, le 20 Janvier 2017:

Composition du Jury :

M. Martin Lindquist	Professeur, Johns Hopkins University	Rapporteur
M. Florent Krzakala	Professeur, University Pierre et Marie Curie and École Normale Supérieure	Rapporteur
M. Erwan le Pennec	Professeur, École polytechnique	Examinateur, Président du jury
M. Christophe Phillips	Chargé de recherche, University of Liège	Examinateur
M. Bertrand Thirion	Directeur de recherche, Inria	Directeur de thèse
M. Gaël Varoquaux	Chargé de recherche, Inria	Co-encadrant de thèse

andrés hoyos idrobo

NNT: 2017SACLS029

PhD Thesis
of
University of Paris-Saclay
prepared at
University of Paris-Sud

Doctoral School n°580
Information and Communication Sciences and Technologies

PhD Specialty : Computer Science

By

Mr. Andrés Hoyos-Idrobo

Ensembles of models in fMRI: stable learning in large-scale settings

Thesis presented and defended in Palaiseau, the 20th of January 2017:

Composition of the Jury:

Mr. Martin Lindquist	Professor, Johns Hopkins University	Reviewer
Mr. Florent Krzakala	Professor, University Pierre et Marie Curie and École Normale Supérieure	Reviewer
Mr. Erwan le Pennec	Professor, École polytechnique	Examiner, Jury president
Mr. Christophe Phillips	Research Associate, University of Liège	Examiner
Mr. Bertrand Thirion	Research Director, Inria	Doctoral advisor
Mr. Gaël Varoquaux	Research Associate, Inria	Co-supervisor

Version as of May 1, 2017

andrés hoyos idrobo

Acknowledgements

First, I would like to express my deep gratitude to my advisors Betrand Thirion and Gaël Varoquaux that guided me through these 3 years of discovery and improvement, showing me the fascinating world of machine learning in neuroimaging. Bertrand always incited me to be rigorous, and Gaël remarked me the importance of communication. In addition to Bertrand and Gaël, I am also grateful to have the opportunity to collaborate with Jonas Kahn. I would also like to thank the other members of my thesis committee: Martin Lindquist, Christophe Phillips, Erwan le Pennec, and Florent Krzakala.

I am fortunate to have been a member of the Parietal team, whose spirit is just awesome. Namely, Aina Frau Pascual –Lets keep in touch!–, Michael Eickenberg –do you know e***t?–, Ana Luisa Pinho –Oh, my goodness that concert was epic!!–, Mehdi Rahim –you know deportivo Cali!–, Alexandre Abraham –do you have Columbian "sugar"?–, Jerôme Dockès –ça va chef ?–, Loïc Steve –hardcore developer–, Elvis Dohmatob –Est-ce que on fait un match ?–, Darya Chyzhyk –No hugs for you–, Kamalaker Reddy –Kamalakarrrrrrr–, Danilo Bzdok –Oscar Wilde, Chuck Norris, . . .–, Solveig Badillo –thanks for inviting me to your parties–, Yannick Schwartz –hola guapa!, cuanto?–, Salma Bougacha –telenovelas–, Daria La Rocca –Pero que estás diciendo!–, Fabian Pedregosa –thank you for all the tortillas españolas–, and also Arthur Mensch, Olivier Grisel, Carole Lazarus, Philippe Ciuciu, Virgile Fritsch, Viviana Siless, Giorgio Patrini, Nicolas Chauffert, Léonard Blier, Joao Loula, Patricio Cerda, Hubert Pellé, Gaspar Pizarro, Ronald Phlypo, Moritz Boos, Kyle Kastner, Torsten Ruest, Konstantin Shmelkov, Guillaume Lemaitre, Andre Manoel, Loubna El Gueddari, and charlotte van schie.

Big thanks also to other people from Neurospin. In particular, Valentina borghesani –best thesis sprints ever!–, Martin Perez-Guevara –panecito de amor–, Elodie Doger De Spéville – –, Darinka Trübutshek –you are gonna be fine . . .–, Pedro Pinheiro Chagas –so nice–, Parvaneh Adibpour –you are so nice!–, Mainak Jas –Lets play table tennis–, Laetitia Grabot –good movie recommendations–, Tadeusz Kononowicz –Dude, nice talks–, Baptiste Gauthier –the sarcasm level is unmeasurable–, Mathieu Dubois –full of craziness–, and also Fosca Al Roumi, Clement Moutard, Benoît Martin, Bianca Trovò.

I would also like to thank the administrative staff of Inria and Microsoft-Research-Inria for making this thesis possible. Special thanks goes to Régine Bricquet, Tiffany Caristan, Marie Domingues, and Hélène Bessin-Rousseau. I extend this gratitude to the director of the Microsoft Research-Inria Joint Centre, Laurent Massoulie.

Last but not least, I would like to thank my family in particular the three women that I love the most: my wife Andrea Ceballos-Herrera, my mother Sofía Idrobo-Peña, and my grand mother René Peña, for her patience, support and love all along the way.

In any case, thank you all for sharing a piece of your life with me and giving me the strength to overcome this endeavor. I will always remember you, unless Alzheimer strikes :)

See you around, because of reasons.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Ensembles de modèles pour l'IRMf: l'apprentissage stable à grande échelle

Mots-clé: IRMf, clustering, décodage, réduction de dimension.

Résumé: En imagerie médicale, des collaborations internationales ont lançé l'acquisition de centaines de Terabytes de données - et en particulier de données d'Imagerie par Résonance Magnétique fonctionelle (IRMf) - pour les mettre à disposition de la communauté scientifique. Extraire de l'information utile de ces données nécessite d'importants prétraitements et des étapes de réduction de bruit. La complexité de ces analyses rend les résultats très sensibles aux paramètres choisis. Le temps de calcul requis augmente plus vite que linéairement: les jeux de données sont si importants qu'il ne tiennent plus dans le cache, et les architectures de calcul classiques deviennent inefficaces.

Pour réduire les temps de calcul, nous avons étudié le feature-grouping comme technique de réduction de dimension. Pour ce faire, nous utilisons des méthodes de clustering. Nous proposons un algorithme de clustering agglomératif en temps linéaire: Recursive Nearest Agglomeration (ReNA). ReNA prévient la création de clusters énormes, qui constitue un défaut des méthodes agglomératives rapides existantes. Nous démontrons empiriquement que cet algorithme de clustering engendre des modèles très précis et rapides, et permet d'analyser de grands jeux de données avec des ressources limitées. En neuroimagerie, l'apprentissage statistique peut servir à étudier l'organisation cognitive du cerveau. Des modèles prédictifs permettent d'identifier les régions du cerveau impliquées dans le traitement cognitif d'un stimulus externe. L'entraînement de ces modèles est un problème de très grande dimension, et il est nécéssaire d'introduire un a priori pour obtenir un modèle satisfaisant.

Afin de pouvoir traiter de grands jeux de données et d'améliorer la stabilité des résultats, nous proposons de combiner le clustering et l'utilisation d'ensembles de modèles. Nous évaluons la performance empirique de ce procédé à travers de nombreux jeux de données de neuroimagerie. Cette méthode est hautement parallélisable et moins coûteuse que l'état de l'art en temps de calcul. Elle permet, avec moins de données d'entraînement, d'obtenir de meilleures prédictions. Enfin, nous montrons que l'utilisation d'ensembles de modèles améliore la stabilité des cartes de poids résultantes et réduit la variance du score de prédiction.

Title: Ensembles of models in fMRI: stable learning in large-scale settings

Keywords: fMRI, clustering, decoding, dimensionality reduction.

Abstract: In medical imaging, collaborative worldwide initiatives have begun the acquisition of hundreds of Terabytes of data that are made available to the scientific community. In particular, functional Magnetic Resonance Imaging –fMRI– data. However, this signal requires extensive fitting and noise reduction steps to extract useful information. The complexity of these analysis pipelines yields results that are highly dependent on the chosen parameters. The computation cost of this data deluge is worse than linear: as datasets no longer fit in cache, standard computational architectures cannot be efficiently used.

To speed-up the computation time, we considered dimensionality reduction by feature grouping. We use clustering methods to perform this task. We introduce a linear-time agglomerative clustering scheme, Recursive Nearest Agglomeration (ReNA). Unlike existing fast agglomerative schemes, it avoids the creation of giant clusters. We then show empirically how this clustering algorithm yields very fast and accurate models, enabling to process large datasets on budget. In neuroimaging, machine learning can be used to understand the cognitive organization of the brain. The idea is to build predictive models that are used to identify the brain regions involved in the cognitive processing of an external stimulus. However, training such estimators is a high-dimensional problem, and one needs to impose some prior to find a suitable model.

To handle large datasets and increase stability of results, we propose to use ensembles of models in combination with clustering. We study the empirical performance of this pipeline on a large number of brain imaging datasets. This method is highly parallelizable, it has lower computation time than the state-of-the-art methods and we show that, it requires less data samples to achieve better prediction accuracy. Finally, we show that ensembles of models improve the stability of the weight maps and reduce the variance of prediction accuracy.

andrés hoyos idrobo

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

École Doctorale n°580
Sciences et Technologies de l'Information et de la Communication

Résumé – Ensembles de modèles pour l'IRMf: l'apprentissage stable à grande échelle

En imagerie médicale, des collaborations internationales ont lançé l'acquisition de centaines de Terabytes de données et en particulier de données d'Imagerie par Résonance Magnétique fonctionelle (IRMf) - pour les mettre à disposition de la communauté scientifique. Extraire de l'information utile de ces données nécessite d'importants prétraitements et des étapes de réduction de bruit. La complexité de ces analyses rend les résultats très sensibles aux paramètres choisis. Le temps de calcul requis augmente plus vite que linéairement: les jeux de données sont si importants qu'il ne tiennent plus dans le cache, et les architectures de calcul classiques deviennent inefficaces.

A la croisée des mathématiques, de l'informatique et des neurosciences, j'ai contribué par mon travail de thèse à des innovations sur 3 aspects différents: *i)* au niveau algorithmique, j'ai proposé une nouvelle méthode de clustering agglimeratif en temps linéaire, pour extraire des régions cérébrales à partir de données d'imagerie fonctionnelle prenant en compte la variabilité inter-individuelle, ainsi que des autres signaux structurés; *ii)* dans le domaine applicatif, j'ai utilisé ces régions cérébrales avec des modèles de machine learning pour améliorer sensiblement l'état de l'art sur la stabilité des algorithmes pour le décodage des images du cerveaux. J'ai réalisé une analyse statistique complète de ces résultats afin de les vérifier, et *iii)* dans le domaine informatique, j'ai fait parti du développement de un package Python open-source permettant une collaboration plus aisée entre neuro-scientifiques et informaticiens proposant des implémentations performantes d'algorithmes propres au domaine et capable de traiter de grands jeux de données.

Réduction de la dimension par regroupement des voxels

Les performances des méthodes de machine learning dépendent grandement de la dimension des données d'apprentissage. Les données IRMf étant composées d'une succession de 150 à 1000 images 3D comptant 100 000 voxels cérébraux, il est impossible de les analyser directement en raison du fléau de la dimension. Pour s'affranchir de cet effet, et pour réduire les temps de calcul, nous avons étudié reagroupage des voxels –le feature-grouping– comme technique de réduction de dimension. Avec cet technique nous segmentantons le cerveau en sous-unité fonctionnelles, les régions cérébrales. Dans tous les cas, ces images sont donc utilisées en entrée d'autres algorithmes, comme une analyse en composantes indépendantes ou un algorithme de classification. Ces algorithmes souffrent d'une grande lenteur et exigence mémoire pour des entrées d'une dimension pareille. Il est donc utile de réduire cette dimension. Pour être clair, nous parlons de réduire à 2000 dimensions, gardant ainsi l'essentiel du signal, pas de résumer en trois paramètres. Mais les algorithmes de réduction de dimension doivent eux-mêmes être rapide et suffisamment fidèles. Pour ce faire, nous utilisons des méthodes de clustering agglomeratif. Nous proposons un nouveau algorithme de clustering agglomératif en temps linéaire: Recursive Nearest Agglomeration (ReNA). ReNA prévient la création de clusters énormes, qui constitue un défaut des méthodes agglomératives rapides existantes.

Contributions

Dans la continuité d'ussage des méthodes de clustering au sein de mon laboratoire, j'ai développé une méthode de clustering agglomeratif en temps linéaire: ReNA. Elle promeut des parcellations compactes **[4]**. Cette méthode peut être appliqué pour ajouter une contrainte spatiale au niveau du groupe (multi-sujet), en réduiçant le temps de calcule des

méthodes comme l'inférence fondée sur une parcelle aléatoire, analyse des composants principaux, au les estimateurs linéaires. Enfin, j'ai proposé une méthode d'extraction de clusters à partir ces cartes cérébrales, pouvant autant fonctionner á partir de coefficients des estimateurs lineares ainsi que d'ICA, afin de reduire le temps de calcul tant qu'il ameliore le conditionement du problème d'optimization.

Résultats

L'absence de vérité de terrain est un frein à l'évaluation de la pertinence de nos modèles. Néanmoins, nous pouvons utiliser le score de prédiction comme un proxi du performance. Premièrement, nous comparons le performance des estimateurs avec différent méthodes de réduction de la dimensionalité **[1]**. Nous démontrons empiriquement que cet algorithme de clustering engendre des modèles très précis et rapides, et permet d'analyser de grands jeux de données avec des ressources limitées. Cette methode n'est pas limité aux lattices 3D, car il peut être appliqué a n'importe quelle topologie qui peuve être répresenté comme un graph. Pour cela, il nous faut un calcule de distances ou de similarités entre les features.

Apprentissage statistique à grande échelle

En neuroimagerie, l'apprentissage statistique peut servir à étudier l'organisation cognitive du cerveau. Des modèles prédictifs permettent d'identifier les régions du cerveau impliquées dans le traitement cognitif d'un stimulus externe. L'entraînement de ces modèles est un problème de très grande dimension, et il est nécéssaire d'introduire un a priori pour obtenir un modèle satisfaisant. Néanmoins, les a prioris qui marchent le mieux souffrent d'une grande lenteur. Une possible solution est l'utilisation des ensembles de modèles, mais l'implementation naïe de ce stratégie est lent. Le but est de constuir un algorithme qui peut turner sur un ordinateur standard (8 coeurs, 32Gb RAM). Cependant, on propose une pipeline composé par une étape de clustering et la combination de différent modeles pour reduire le temps de calcul et améliorer le performance de prédiction. J'ai testé le score de prédiction, la stabilité des poids et le temps de calcule dans 8 différents études de IRMf et avec plusieurs éstimateurs. J'ai ensuite réalisé une étude post-hoc des résultats qui ont mené à plusieurs conclusions importantes

Contribution

Afin de pouvoir traiter de grands jeux de données et d'améliorer la stabilité des résultats, nous proposons de combiner le clustering et l'utilisation d'ensembles de modèles **[2]**. Cette méthode utilise le boucle de validation croisée, en choisissant les modès avec le meilleur score de prédiction, les coefficients des ces modèles sont donc moyennés. Cet stratégie est moins dépendant des hypothèses du bruit, et sa parallelization est plus simple. j'ai comparé la méthode proposé avec plusieurs des méthodes couramment utilisées.

Résultats

Nous évaluons la performance empirique de ce procédé à travers de nombreux jeux de données de neuroimagerie. Cette méthode est hautement parallélisable et moins coûteuse que l'état de l'art en temps de calcul. Elle permet, avec moins de données d'entraînement, d'obtenir de meilleures prédictions. Enfin, nous montrons que l'utilisation d'ensembles de modèles améliore la stabilité des cartes de poids résultantes et réduit la variance du score de prédiction **[3]**.

Développement logiciel

En parallèle de mon travail de thèse, j'ai été contributeur de nilearn, un package Python destiné à faciliter l'application de méthode de machine-learning sur des données de neuroimagerie et ainsi permettre une collaboration plus efficace entre experts en machine learning et neuroscientifiques. En se basant sur le célèbre package scikit-learn, nilearn rend possible l'intégration directe des algorithmes conçus pas des experts techniques pour des applications neuroscientifiques. De plus, nilearn fournit des implémentations de plusieurs algorithmes de référence du domaines (ICA, clustering, etc.) rapides et optimisées pour un grand volume de données. Nilearn participe aussi à rendre la science plus reproductible puisque qu'une analyse entière peut-être retranscrite dans un unique script et mise à la disposition de la communauté.

[1] Fast brain decoding with random sampling and random projections, Andrés Hoyos-Idrobo, Gaël Varoquaux and Bertrand Thrion. *PRNI - IEEE International Workshop on Pattern Recognition in NeuroImaging*. 2016, Trento, Italy.
[2] Improving sparse recovery on structured images with bagged clustering. Andrés Hoyos-Idrobo, Gaël Varoquaux and Bertrand Thirion. *PRNI - IEEE International Workshop on Pattern Recognition in Neuroimaging*. 2015, Stanford University, Palo Alto, USA.
[3] Stable brain decoding with ensembles of estimators. Andrés Hoyos-Idrobo, Gaël Varoquaux, Yannick Schwartz and Bertrand Thrion. *Neuroimage.*
[4] Recursive nearest agglomeration (ReNA): fast clustering for approximation of structured signals. Andrés Hoyos-Idrobo, Gaël Varoquaux, Jonas Kahn and Bertrand Thrion. *IEEE TPAMI - Transactions on Pattern Analysis and Machine Intelligence.*

Contents

Notations

Notation	Description	Definition
$\mathbf{x}$	Column vectors are written using bold lower-case	
$\mathbf{x}_i$		the i-th component of $\mathbf{x}$
$\mathbf{X}$	Matrices are written using bold capital letters	
$\mathbf{X}_{*,j}$	The jth column vector of $\mathbf{X}$	
$\mathbf{X}_{i,*}$	the ith row vector of $\mathbf{X}$	
$\mathcal{P}$	Letters in calligraphic denote sets or graphs, and it will be clarified by the context	
$\{\mathcal{C}_i\}_{i=1}^k$	be short for the set	$\{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$
$\lvert\mathcal{P}\rvert$	the cardinality of a set $\mathcal{P}$	
$\lVert\mathbf{x}\rVert_p$	ℓ_p norm of a vector $\mathbf{x}$	$\left(\sum_{i=1}^k \lvert\mathbf{x}_i\rvert^p\right)^{\frac{1}{p}}$, for $p = [1, \infty)$.
$[n]$		[n] = $\{1, \ldots, n\}$

andrés hoyos idrobo

Introduction

This thesis was prepared with the Parietal team at Inria, Saclay, France. It was partly funded by the Microsoft Research-Inria Joint Centre.

This thesis work falls within the join Microsoft Research – Inria Medilearn project. The Medilearn project gathers researchers from the Machine Learning and Perception group (MSR Cambridge), and the Inria Asclepios and Parietal teams. Our work aims at developing approaches to learn from large-scale datasets of medical images, and the automatic analysis and indexation of medical images [99][92][83]. In particular, we design machine algorithms to adequately handle large-scale neuroimaging datasets.

Context – "big data" in medical Imaging

In various fields of science, experts have tried to predict a phenomenon based on passed observations or measurements. Generally, scientists have tried to address this by deriving theoretical frameworks based on first principles or accumulated knowledge to analyze, model and understand the phenomenon under study. Nowadays, we are also using data to build data-aware models that are capable of making predictions or decisions. This is the aim of statistical machine learning.

Personalized medicine: In the near future, machine learning could lead to personalized medicine. The key idea behind personalized medicine is to use the predicted response or risk of disease of the individual an patient to tailor medical decisions, practices, interventions and/or products. One example is the promise of a computer algorithm able to predict whether or not a patient has a mental disorder e.g. Alzheimer's disease [116][48][135]. However, medical imaging is not yet powerful enough to achieve this. Nevertheless, we have learned an important lesson: in medical imaging, we need a lot of data to train reliable estimators. In this field, we have to deal with high-dimensional data, where the number of features is greater than the number of samples. Thus we frequently have models that are too complicated for the amount of data available[1]. A consequence is that hyperparameter selection is very difficult due to sampling noise [162]. In practice, this is often done by

[1] In a classical bias-variance trade-off, this corresponds to a high variance setting.

cross-validation like techniques [9].

Large-scale datasets in medical imaging: In medical imaging, building large datasets is not an easy task, as the cost of acquisitions remains high. However, in recent years, more scientists have become aware of the benefits of data sharing, as they see it as an effort to increase reproducibility and reliability of scientific results. Thus, collaborative worldwide initiatives have begun the acquisition of *hundreds of Terabytes* of data that are made available to the scientific community [2,3,4] [46][59][133][98]. These huge datasets open the possibility to ask new and interesting questions. But they also pose new challenges, as processing and analyzing them becomes prohibitive in standard computer architectures.

[2] https://www.braininitiative.nih.gov/

[3] `http://www.ukbiobank.ac.uk/`

[4] `https://ninesights.ninesigma.com/web/hea`

Machine learning and brain imaging studies

In brain-imaging studies, in particular functional Magnetic Resonance Imaging (fMRI), neuroscientists are currently acquiring many datasets that display, in spite of their high noise, brain activation patterns giving access to meaningful representations of brain organization [50]. This ongoing accumulation is intensified via new large-scale international initiatives such as the Human Connectome Project (HCP) [46], with more than 1600 subjects, but also open repositories of functional neuroimaging datasets such as the OpenfMRI [130] with 52 datasets and 1930 subjects across datasets; the UK biobank[5] contains recordings from 5 000 individuals and is growing up to 100 000. However, in task-based fMRI, we often have hundreds of thousands of voxels, whereas the number of brain images remains in the order of thousands. Thus, the sample size continues to be small, and the training of a machine learning algorithm is a high-dimensional problem[6].

[5] `http://www.ukbiobank.ac.uk/`

[6] In neuroscience, each individual experiment typically contains hundreds of observations, and it is often considered as large enough [36]. This is markedly bellow common sample sizes in machine-learning problems [30][140]. Indeed, data analysis in brain imaging has historically been driven by very simple models i.e. linear models, whereas other problems considered in the machine-learning community allow to use richer models.

Pipelines to analyze fMRI data: In fMRI, extensive fitting and "denoising" steps are required to extract useful information from brain images [132]. The two most prominent of these preprocessing steps are head-motion correction and inter-subject registration i.e. spatial normalization. Then, to delineate the brain regions associated with a set of tasks, we model the signal of interest and then do statistical inference or prediction. The main caveat here is that these pipelines are complex, and the results that we obtain are highly dependent on the chosen parameters.

Decoding brain activation images: To understand the cognitive organization of the brain, cognitive neuroscientists usually design

experiments to probe neural activity during various tasks in order to segment the brain into several brain functional units and characterize them. Since 2001[67], a popular approach consists in using machine-learning algorithms to build data-aware models to predict these experimental tasks on unseen data points. In the neuroimaging community this is called brain imaging *decoding* or Multi-Voxel Pattern Analysis (MVPA). The main advantage of these methods is that they allow one to aggregate heterogeneous datasets, each one under a different experimental setting[7]. Therefore, *decoding* can be used on these datasets to capture a large variety of brain processes[8], giving the possibility to understand the large-scale functional organization of the brain.

[7] The analysis of several experimental designs can be done with forward inference. However, they are not explored in this thesis.

[8] The neuroscience question here is that of functional specificity [155].

Decoding with linear models: We use linear models for decoding, as these models can be relatively well-behaved in very high-dimensional settings. In addition, these models yield weight maps that delineate predictive brain regions. These weight maps can easily be interpreted as loadings on brain regions tied to cognitive tasks.
From the neuroscience point-of-view they can lead to probing well-posed questions [111], such as the neurologic signature of physical pain [165].

We need stable algorithms in neuroimaging

Generally speaking, any neuroimaging study based on group level results (univariate or multivariate) assumes overall consistency of the functional specialization of cortical areas across subjects. This universality assumption is shared with cognitive neuropsychology [26], and constitutes the basis of the most widely adopted inferences schemes in cognitive neuroscience [70] [127]. This assumption implies an underlying spatial structure. Then, the acquired brain activation images can be cast as the result of a generative process acting on the activity of groups of neurons[9]. In other words, the physiological activity can be parametrized by a spatially-structured neighborhood i.e. topology. It is important to take this information in account when designing decoding algorithms, in order to model local correlations.

[9] The acquired signals display local correlations. In fMRI, neighboring voxels respond similarly.

Stable weight maps: A necessary condition to achieve reproducibility and reliability results in neuroimaging is stability [175]. In particular, we need brain decoding algorithms with stable loadings on voxels, that are typically expected to delineate brain regions. This stability is measured with respect to data perturbations[10] –e.g. boostrap, cross-validation, jackknife. To assess stability, we have to train the decoding algorithm several times, enough to build an empirical distribution of the predictive performance. This can be prohibitive as, generally, work-stations used by neuroscientists are

[10] These perturbations are defined as sampling from an underlying distribution or replicating the experiment for a new set of data.

not sophisticated[11]. Hence, the computation resources required to assess stability are still limited.

[11] For instance, Graph-net can take weeks in a standard workstation, as it is reported in [106]

Summary

In neuroimaging, we are accumulating large volumes of data, yet the computing resources available to train machine-learning algorithms are not strong enough in view of this accumulation. Additionally, standard pipelines to analyze fMRI data are complex, making the setting of parameters and hyperparameters difficult. Thus, we need to use the available computation resources adequately.

Decoding opens the possibility of mapping a large variety of brain processes [131], giving a framework to understand the functional specificity of the brain. To do this, decoding algorithms have to be:

1. *Spatially stable:* The weight maps of the decoder have to be stable to data samplings/resamplings. In addition, the brain regions delineated by the decoder have to be consistent across subjects.

2. *Fast to compute:* To learn from large collaborative neuroimaging datasets, one needs to train several times the decoder to set the hyperparameters and assess its performance. This repetitive task entails computation costs that are intractable in common workstations.

Layout of the manuscript

Part I: Modeling and analyzing fMRI data

The first part introduces functional-imaging techniques and the standard tools that can be used to localize brain-activation patterns. We start by describing the assumptions to model the fMRI BOLD signal and show how we can obtain statistical maps. We also introduce statistical machine learning methods used in neuroimaging. In particular, we present linear models and how to deal with high-dimensional datasets, as is the case of neuroimaging. We describe the use of data sampling/resampling schemes to perform hyperparameter selection.

Part II: Contribution – fast decoding

In this part, we benchmark standard fast dimension reduction techniques –i.e. random sampling and random projections– in the context of brain decoding. In particular, we show that random sampling yields interpretable brain weight maps. Then, we introduce and analyze feature grouping as a dimension reduction technique that implicitly takes into account the spatial structure of

brain images. In addition, we propose a fast clustering algorithm, that finds clusters of roughly the same size in images. Finally, we validate the proposed algorithm on several datasets and statistical learning tasks.

Part III: Contribution – improving the stability of brain decoders In this part, we propose a simple scheme to train ensembles of models in a high-dimensional setting. We show empirical evidence on several neuroimaging datasets that ensembles of models yield more stable weight maps, as well as prediction accuracy. In terms of computation time, they are faster than state-of-the-art decoding methods –i.e. total-variation (TV) and Graph-net. In addition, they are highly parallelizable.

Part I

Modeling and analyzing fMRI data

1 Background – studying the brain

In this chapter, we describe how cognitive neuroscientists use the functional Magnetic Resonance Imaging (fMRI) signal to analyze brain function. We start with a brief description of some functional neuroimaging modalities. Then, we introduce the standard tools used to model and analyze fMRI data. In particular, we introduce the basic intuitions about the signal of interest, the BOLD signal, which can be seen as a delayed and blurred version of the neural activity evoked by a particular task. Finally, we present the brain imaging datasets used throughout this manuscript.

1.1 Functional neuroimaging modalities

Today, neuroscientists have at their disposal several functional brain imaging methods [72]. Each of these modalities makes it possible to study brain structures, as well as their function. Nevertheless, they represent different aspect of neuronal activity, and actually have different time and spatial scales. Hence, the selection of a functional brain imaging modality depends on the experimental hypothesis that one wants to test.

A brief summary of some functional neuroimaging modalities is presented in Fig. 1.1.

Electroencephalography (EEG): It measures the electrical activity on the scalp. EEG represents direct measures of neural activity. It has a good temporal resolution, typically of the order of milliseconds. In particular, scalp-recorded EEG has been used extensively in research, clinical neurophysiology, and practice, but its spatial resolution remains poor. This lack of spatial resolution is mainly due to the attenuation and distortion of the electrical signal by intervening tissues such as cerebrospinal fluid, skull and scalp.

Invasive versions of EEG improve spatial resolution by placing subdural and/or depth electrodes for a more direct recording of spontaneous or evoked neural activity. In humans the following two are typically used: *i)* Electrocorticography (ECoG), where the electrodes are placed on the surface of the brain; and *ii)* Stereotactic EEG (sEEG) uses depth electrodes localized

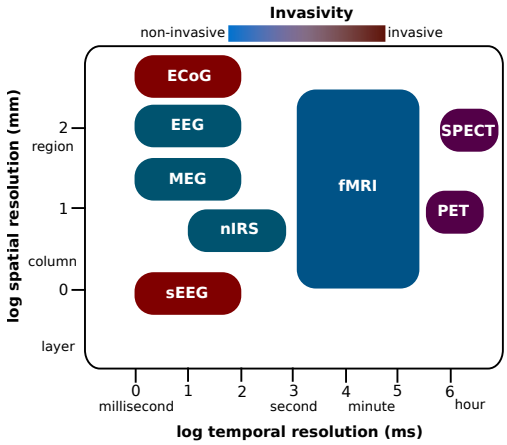


Figure 1.1: **Commonly used functional neuroimaging methods:** Spatial and temporal resolutions of different modalities commonly used for functional imaging. fMRI is a non-invasive technique, it gives a spatial resolution of 1 to 3 milliliters with a temporal resolution of 1 to 3 seconds.

with a stereotactic technique.

Magnetoencephalography (MEG): It measures the magnetic field induced by neural electrical activity. Although EEG and MEG are produced by the same physiological process, MEG displays an improved spatial resolution as the sources of activity are less distorted by intervening tissues. MEG is mainly used in neuroscience research.

Positron emission tomography (PET): Is an imaging modality used to track glucose consumption. It is based on the detection of radioactive tracers induced in a subject. It can generate precise radioactivity distribution maps of the target, and consequently, it can provide detailed information of biochemical or physiological processes with a precise anatomical context. It is widely used in neuroscience research, clinical application, and neuropsychopharmacological drug development.

Functional magnetic resonance imaging (fMRI): It uses a strong magnetic field to indirectly measure neuronal activity. fMRI is a widely used modality that makes it possible to safely and noninvasively detect physiological changes that indicate brain activity with a good spatial resolution (1 - 3mm^3), and a temporal resolution on the order of 1 - 3 seconds. It is important to note that fMRI has full-brain coverage, and it is not limited to the cerebral cortex. fMRI is thus a suitable modality for the localization of brain functions.

1.2 fMRI – blood flow and neuronal activity

Neuroscientific studies increasingly rely on non-invasive imaging of the hemodynamic response to neural activation using functional magnetic resonance imaging [132]. In particular, they seek to establish whether cognitive processes elicit activity in the brain regions. This is tested with a

set of predefined experimental conditions, designed to isolate some cognitive process of interest. The fMRI signal is well-suited to observe these spatial differences.

Blood Oxygen Level Dependent signal (BOLD)

The most common fMRI modality[1] takes advantage of the fact that neurons in the brain increase their consumption of energy when they are active, and this metabolic activity is implicitly related to the oxygen consumption. Thus, we can indirectly measure neural activity by acquiring a signal depending on local metabolic demands –oxygenation consumption– of active neurons. This is the core concept behind the Blood Oxygen Level Dependent signal, or BOLD [117][118].

[1] There are other modalities, like arterial spin labeling (ASL) [171] method, where the contrast comes form blood flow and perfusion, independent of blood oxygenation.

The BOLD signal arises from the interplay between blood flow, blood volume, and blood oxygenation in response to changes in neuronal activity. This signal captures the difference between oxygenated and deoxygenated hemoglobin. This is possible because hemoglobin exists in two different states, each of which has different magnetic properties and produces different local changes of magnetic susceptibility: oxyhemoglobin is diamagnetic, and deoxyhemoglobine is paramagnetic.

During brain activity, the energy consumption is increased and so is oxygen consumption. Then, there is a local increase of blood flow and volume, which increases the homogeneity of magnetic susceptibility, hence the BOLD signal. However, the delivered oxygen exceeds metabolic need. The magnetic susceptibility of blood is altered compared to the surrounding tissue. This creates local magnetic inhomogeneities that decrease the BOLD signal. Then an inflow of oxygenated blood, much higher than the oxygen consumed, changes this ratio again and BOLD signal increases [24]. All these steps make the BOLD signal a blurred and delayed representation of the original neural signal [91].

Major components of fMRI analysis

The Fig. 1.2 shows a standard preprocessing pipeline in fMRI.

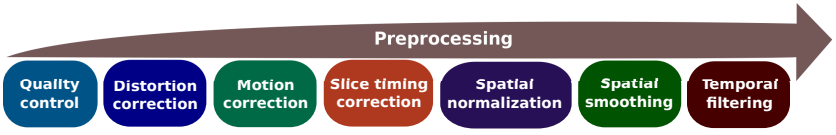


Figure 1.2: **Preprocessing of fMRI data:** Illustration of the complex pipeline necessary to extract useful information from the fMRI BOLD signal.

Here, we present a summary of major component of fMRI analysis:

Preprocessing:

1. *Quality control:* Checking whether the data are not corrupted by artifacts such as spikes or incomplete brain coverage, ensuring no missing regions.
2. *Distortion correction:* This reduces the spatial distortions that often occur in fMRI, coming from the interaction of the magnetic field with different tissues and materials in echo-planar images.
3. *Motion correction:* Head motion has typically a magnitude of a few millimeters. One has to reduce this displacement by realigning the scans across time using some form of image matching.
4. *Slice timing correction:* Brain slices are not acquired at the same time, hence one has to correct differences in timing across different slices in the image to fit a uniform model on the whole brain.
5. *Spatial normalization:* The brain of each individual subject is different. To perform the group analysis, one has to align each brain into a common space[2].
6. *Spatial smoothing:* One can in general assume a local structure of brain signal; if there is activation in one voxel; it is highly probable that its neighbors will present activation too. Under this assumption, the use of spatial smoothing increases the signal-to-noise-ratio (SNR), by reducing undesired noise.
7. *Temporal filtering:* Filtering of the data in time reduces low-frequency noise.

[2] A reference anatomical template used for fMRI registration is the Montreal Neurological Institute (MNI).

Analysis: This is covered by the next section.

9. *Statistical modeling:* Fitting of a statistical model to the data in order to estimate the response to a task.
10. *Statistical inference:* Estimation of the statistical significance of the results, always correcting for the large number of statistical tests performed.
11. *Visualization:* Visualization of the results and estimation of effect sizes.

1.3 Encoding – modeling fMRI data

The main interest of cognitive neuroscientists is to understand how mental function are implemented. Yet, their organization in the brain is a step in this direction. In order to assess this information, they build experiments composed of several mental tasks/conditions designed to manipulate specific mental processes. Thereafter, they test whether a particular brain region correlates with the stimuli or not. A model is needed to perform these

tests. Parametric tests are often preferred due to their simplicity, but non-parametric tests make it possible to relax the underlying assumptions.

General Linear Model (GLM)

Let us recall that the BOLD signal is a delayed and blurred version of the original neural signal [91]. Therefore, we can express voxel activations due to experimental conditions as a noisy linear forward model[52]. For simplicity, we can use a Linear Translation Invariant (LTI) filter to model this blurriness.

In brief, the LTI assumption[28][19], implies

Scaling (homogeneity): The amplitude of the measured signal corresponds to the amplitude of neuronal activity. Relative difference in amplitude makes sense, hence the amplitude difference in the signal between two conditions can be used to determine if the neuronal activity is different.

Additivity/superposition: It allows us to differentiate between the response in any brain region to multiple closely spaced stimuli.

Time invariant: If the stimulus is shifted by t seconds, the BOLD signal will be shifted by the same amount.

Hemodynamic Response Function (HRF)

The HRF is composed of several stages (see Fig. 1.3): *Initial dip* – Generally ignored in most models of fMRI data [132], it reflects early oxygen consumption before blood flow and volume occur [23]. *Peak height* – This is the most common feature of interest due its relation with the activity in the tissue [91]. *Time to peak* – This increase in BOLD signal peaks about 4 to 6 seconds following activation. *Poststimulus undershoot* – The HRF generally shows a late undershoot, that is relatively small compared with the positive response. It is due to a combination of reduced blood flow and increased blood volume, and it can persist up to 20 seconds after the stimulus.

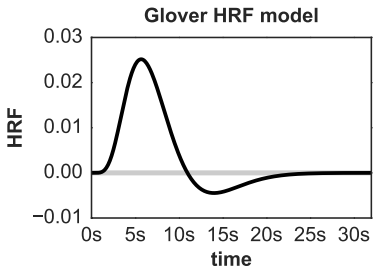


Figure 1.3: **Canonical Hemodynamic Response Function (HRF)[58]:** The HRF first follows an increase of the signal (from 1 to 5.2s), then decreases (from 5.2 to 12.2s), and finally returns to baseline with undershoot (from 12.2 to 20s).

Design Matrix

The design matrix defines the temporal effects that can be observed in the fMRI data. The columns in the design matrix consist of the convolution of the HRF by the occurrence of stimuli presentation in the experimental design, as well as confounding variables and filters (i.e. motion, respiratory cycle, cardiac rhythm fluctuations, and low-frequency signals).

The Fig. 1.4 shows an example of a simple design matrix (only 4 conditions are modeled), each task/condition/event is convolved by the HRF, leading to an approximation of the BOLD signal for each task. These activations are linearly combined to model a clean BOLD signal.

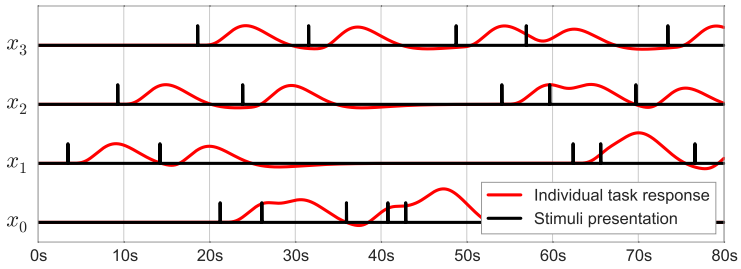


Figure 1.4: **Illustration of the construction of the design matrix:** Each of the four fMRI experimental conditions is convolved with the HRF, yielding an atom to approximate the acquired BOLD signal.

Putting all the pieces together, the GLM is formulated as follows:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{1.1}$$

where $\mathbf{Y} \in \mathbb{R}^{n \times p}$ represents the acquired fMRI data, n denotes the number of scans, and p the number of voxels. $\mathbf{X} \in \mathbb{R}^{n \times k}$ denotes the design matrix, where k is the number of regressors. Each condition regressor is an indicator of occurrence of stimuli in the experimental design convoluted with the HRF (see Fig. 1.5). The design matrix also includes some nuisance confounds such as subject motion, additional confounds such as session or study-dependent effects, as well as low-frequency signals that model drifts in the data. In addition, the time derivative of the experimental conditions is also often included, as it takes into account signal delays in the hemodynamic response.

$\boldsymbol{\beta} \in \mathbb{R}^{k \times p}$ denotes the effects (the weight of the regressors), and $\boldsymbol{\epsilon} \in \mathbb{R}^{n \times p}$ denotes a noise component. For each voxel j, this noise can be modeled as a Gaussian white noise with zero mean and variance σ_j^2. It is often modeled as an $AR(1)$ process, as the BOLD signal is positively correlated in the time domain. Assuming Gaussian noise with variance σ_j^2 and full column rank of $\mathbf{X}$, the best unbiased estimator of $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} = \mathbf{X}^{\dagger}\mathbf{Y} = \boldsymbol{\beta} + \mathbf{X}^{\dagger}\boldsymbol{\epsilon}, \tag{1.2}$$

where $\mathbf{X}^{\dagger}$ denotes the moore-Penrose pseudo inverse of $\mathbf{X}$. Note that $\mathbf{X}^{\dagger}\boldsymbol{\epsilon}$ is also Gaussian with zero mean.

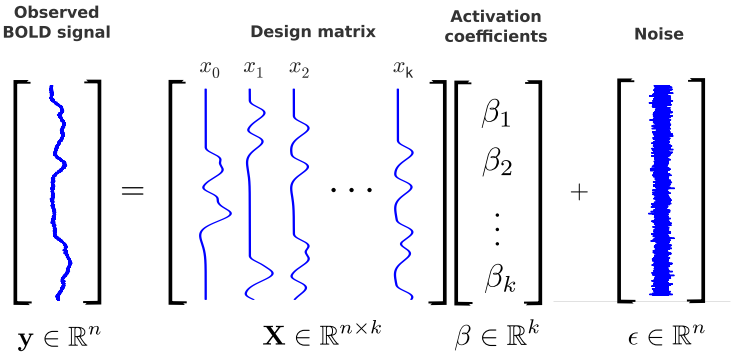


Figure 1.5: **The GLM:** The signal in each voxel is modeled as a linear combination of the smoothed stimuli presentation, plus noise. By stacking regressors, this model can include several sources of noise. Based on [122].

1.4 Statistical methods for task-based fMRI analysis

In cognitive neuroscience, researchers design experiments to test whether a brain region shows more activity for a particular condition than for other (e.g. it responds to a condition A and not B). We are particularly interested in establishing a relative difference between two or more contrasts, in form of a statistical test [90]. We encode these comparisons in a vector $\mathbf{c} \in \mathbb{R}^k$ –e.g. with a contrast vector $\mathbf{c} = [+1, 0, -1, 0]$ we test if the first versus the third condition/event are significantly different. In this setting, for each voxel $j \in [n]$, the *null hypothesis* $H_0 : \mathbf{c}^\top \boldsymbol{\beta}_j = 0$, the difference is due to chance; the *alternative hypothesis* corresponds to $H_1 \quad \mathbf{c}^\top \boldsymbol{\beta}_j \neq 0$. After fitting the GLM, we have

$$\mathbf{c}^\top \boldsymbol{\beta}_j = \mathbf{c}^\top \hat{\boldsymbol{\beta}}_j - \mathbf{c}^\top \mathbf{X}^\dagger \boldsymbol{\epsilon}_j, \tag{1.3}$$

which is a Gaussian variable $\mathbf{c}^\top \hat{\boldsymbol{\beta}}_j \sim \mathcal{N}(\mathbf{c}^\top \boldsymbol{\beta}_j, \sigma_j^2 \mathbf{c}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{c})$. Then, we can use several statistical test, we often consider:

z-statistic:

$$z_j = \frac{\mathbf{c}^\top \boldsymbol{\beta}_j}{\sigma_j \sqrt{\mathbf{c}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{c}}}. \tag{1.4}$$

Usually we don't have access to σ_j, then we have to estimate it[3] as $\hat{\sigma}_j^2 = \|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_j\|^2 / (n-k)$. Then we can build a t-statistic as follows

t-statistic:

$$t_j = \frac{\mathbf{c}^\top \hat{\boldsymbol{\beta}}_j}{\hat{\sigma}_j \sqrt{\mathbf{c}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{c}}} \sim t_{n-k}. \tag{1.5}$$

We obtain a map representing the brain activity with one test per voxel –i.e. t-map, F-map, χ^2-map. These kind of maps are more generally called statistical parametric maps (SPMs) [51]. The Fig. 1.6 shows an example of a brain activation map of a contrast between two conditions: face and scrambled face on the Henson dataset –see datasets description in section 1.5.

[3] The unbiased estimator is estimated from:

$$\begin{aligned}
\mathbb{E}\left[\|\mathbf{r}_j\|^2\right] &= \mathbb{E}\left[\|\mathbf{Y}_j - \mathbf{X}\hat{\boldsymbol{\beta}}_j\|^2\right] \\
&= \mathbb{E}\left[\|(\mathbf{I} - \mathbf{X}\mathbf{X}^\dagger)\boldsymbol{\epsilon}_j\|^2\right] \\
&= \mathbb{E}\left[\boldsymbol{\epsilon}_j{}^\top (\mathbf{I} - \mathbf{X}\mathbf{X}^\dagger)\boldsymbol{\epsilon}_j\right] \\
&= \mathbb{E}\left[\mathrm{Tr}((\mathbf{I} - \mathbf{X}\mathbf{X}^\dagger)\boldsymbol{\epsilon}_j\boldsymbol{\epsilon}_j{}^\top)\right] \\
&= \hat{\sigma}_j^2 \,\mathrm{Tr}(\mathbf{I} - \mathbf{X}\mathbf{X}^\dagger) \\
&= \hat{\sigma}_j^2 (n-k)
\end{aligned}$$

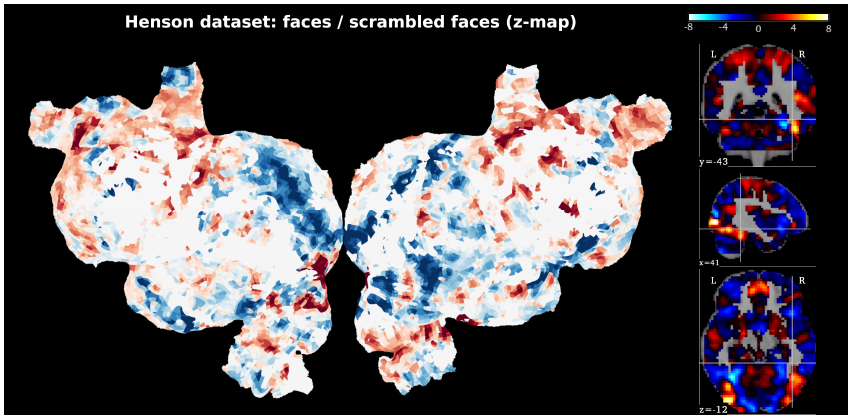


Figure 1.6: **Example of brain activation maps:** Beta maps for a visual stimulus. The thresholding of the left hand side image is arbitrary and for visualization purpose only.

The p-value resulting from a hypothesis testing corresponds to the probability of an observation considering that H_0 is true. After, we can set an arbitrary significance level –e.g. 5% or 1%–, which controls that the number of false positives –Type I error– does not exceed the chosen threshold. But, we are controlling the *type I* error per voxel and we are testing around 50 000 voxels individually. This yields a large number of false positives and a further correction has to be applied. This problem is referred to as the *multiple testing problem* and it is a critical issue in fMRI data analysis.

As mentioned, at voxel level, classical methods provide simple means to control the level of false positive risk through appropriate selection of the significance level. Then, to account for the multiplicity, we have to measure the false positive risk over the whole image. Basically, there are two measures of false positive risk: the family-wise error rate and the false discovery rate.

The family-wise error rate (FWER): Is the probability of any *Type I* error among all the tests when performing multiple hypothesis tests [113]. We can apply different corrections of the significance level:

Bonferroni correction assumes that all voxels are independent, and corrects the significance level by dividing it by the number of tests performed. This method becomes conservative when there is a strong correlation between tests. Due to the smoothness of fMRI data, *Bonferroni* is conservative and has little power to detect true effects.

Random Field Theory assumes that the data are smooth and have a lower probability of exceeding a certain threshold by chance. This threshold is chosen to control the probability of maximum above a given threshold [172].

Resampling methods use the data to obtain empirical null distributions of interest [44]. The most common resampling techniques are permutation tests and bootstrap. They are accurate, but computationally demanding.

The false discovery rate (FDR): It is the expected proportion of rejected hypotheses that are false positive. The Benjamini-Hocheberg procedure [17] consists of an adaptive Bonferroni correction that applies a different correction depending on the significance of the voxel [54].

1.5 Getting the data

Nowadays, there is a growing awareness of the importance of data sharing in the neuroimaging community [129]. It is indeed important to increase reproducibility and the validation of scientific claims by sharing the data leading to those results. Some of the pioneering projects are OpenfMRI [130] and the Human Connectome Project (HCP) [46].

These initiatives come with the opportunity to ask new questions about the functional structure of the brain; they allow researchers to use several datasets built with different experimental conditions, and they open the door to meta-analysis on full brain images.

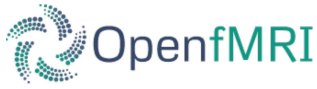
Datasets

In this section, we introduce the MRI data used throughout this document. The detailed information on tasks and sample size is summarized in Table 6.1.

Resource name	Data type	URL
OpenfMRI	raw & maps	`https://openfmri.org`
OASIS	raw	`http://www.oasis-brains.org`
HCP	raw	`https://db.humanconnectome.org`
BioMag2010	raw	`ftp://ftp.mrc-cbu.cam.ac.uk/personal/rik.henson/wakemandg_hensonrn/`

Table 1.1: Table listing some resources offering raw MRI data.

OpenfMRI: The OpenfMRI [130] database is a repository of human brain imaging data collected using MRI and EEG techniques. OpenfMRI is a game changing initiative, that allow researchers to make their MRI data openly available to the research community, by making the data accessible by direct download under a permissive license. The main goal of this initiative is to increase the validity of scientific claims due to ease of analysis replication of the results. As of today, it features as many as 52 task-fMRI datasets and 1930 subjects across all datasets.



(a) OpenfMRI

(b) OASIS

(c) HCP

Figure 1.7: **Some raw MRI data sharing initiatives:** (a) OpenfMRI contains 52 datasets, summing up to 1564 subjects; (b) The OASIS dataset contains Voxel-Based-Morphometry of 403 subjects; (c) The HCP shares data from over 500 subjects (currently 900) in 7 different fMRI tasks

Haxby dataset: This dataset is already contained in OpenfMRI, but its importance to the community of statistical learning in neuroimaging leads us to describe it separately.

The experimental condition of this dataset [67] is a visual object recognition task obtained from 5 subjects –plus one removed for quality reasons– which aims at studying the face and object representation in human ventral temporal cortex. The data consist of 12 sessions, each containing 9 volumes per object category (i.e. face, house, chair, cat, bottle, scissors, shoes, and scrambled picture). The data were resampled at $3mm$, yielding about $p = 30\,000$ voxels per volume.

The Open Access Series of Imaging Studies (OASIS) The Open Access Series of Imaging Studies (OASIS)[4] [98] is a project aimed at making MRI data sets of the brain freely available to the scientific community. OASIS is made available by the Washington University Alzheimer's Disease Research Center, Dr. Randy Buckner at the Howard Hughes Medical Institute (HHMI) at Harvard University, the Neuroinformatics Research Group (NRG) at

[4] OASIS was supported by grants P50 AG05681, P01 AG03991, R01 AG021910, P50 MH071616, U24 RR021382, R01 MH56584.

Washington University School of Medicine, and the Biomedical Informatics Research Network (BIRN).

This dataset consists of 403 anatomical brain images (Voxel Based Morphometry) of subjects aged between 60 and 96 years. These images were preprocessed with the SPM8 software [11] to obtain modulated grey matter density maps [100] sampled in the MNI space at 2mm resolution. These images were masked to an average mask of the grey matter, which yields about $p = 140\,398$ voxels.

Human Connectome Project (HCP): This dataset[5] [46] contains 500 participants (among 13 were removed for quality reasons), acquired at rest –typically analyzed via ICA– and during cognitive tasks [15] –typically analyzed with linear models. None the subjects has any psychiatric or neurological history. Informed consent was obtained from all participants by the Washington University in St. Louis institutional review board. The primary goal of this dataset is network discovery, which is facilitated by probing experimental task paradigms that are known to tap on well characterized neural networks [15].

Over two image acquisition sessions, paradigms were administered on *1)* working memory/cognitive control processing, *2)* incentive processing, *3)* visual and somatosensory-motor processing, *4)* language processing (semantic and phonological processing), *5)* social cognition, *6)* relational processing, and *7)* emotional processing. All data were acquired on the same Siemens Skyra 3T scanner at Washington University. We profited from the HCP "minimally preprocessed" pipeline [56]. A general linear model (GLM) was implemented by FILM from the FSL suite with model regressors obtained by the convolution with a *canonical* hemodynamic response function with its temporal derivatives. The resulting GLM-derived activation maps (18 per subject) are sampled at 2mm resolution in MNI space, which yields about 220 000 voxels.

[5] Data were provided in part by the Human Connectome Project, WU-Minn Consortium (Principal Investigators: David Van Essen and Kamil Ugurbil; 1U54MH091657) funded by the 16 NIH Institutes and Centers that support the NIH Blueprint for Neuroscience Research; and by the McDonnell Center for Systems Neuroscience at Washington University.

1.6 Summary

fMRI is one of the working horses of cognitive neuroscientists, as its spatial resolution makes it suitable to study the functional structure of the brain. To gain this information, researchers design experiments to test whether or not there is a difference in a hypothesized brain region. Usually, this consists of fitting a model followed by statistical inference. Regarding the modeling step, a General Linear Model (GLM) is often used to model each voxel individually –i.e. mass univariate modeling. This model relates the observable data –BOLD signal– to unobservable parameters –coefficients– as that we can then use inference to quantify the uncertainty in the estimated parameter values [114]. However, performing massively univariate tests is prone to inflating the number of false positives and we thus have to apply multiple comparisons

correction to control them.

It has been shown that linearity assumption may be insufficient, as the shape of the HRF varies across brain regions, across trials, across subjects [62][13]. Yet, taking this variability into account increases even more the degrees of freedom in our pipeline.

Dataset	Description	Samples	# blocks (sess. /subj.)	Task
Haxby (ds105) [67]	fMRI 5 different subjects, leading to 5 experiments per task	209	12 sess.	bottle / scramble cat / bottle cat / chair cat / face cat / house cat / scramble chair / scramble chair / shoe face / house face / scissors scissors / scramble scissors / shoe shoe / bottle shoe / cat shoe / scramble
Duncan (ds107) [41]	fMRI across subjects	196	49 subj.	consonant / scramble consonant / word objects / consonant objects / scramble objects / words words / scramble
Wager (ds108) [163]	fMRI across subjects	390	34 subj.	negative cue / neutral cue negative rating / neutral rating negative stim / neutral stim
Cohen (ds009)	fMRI across subjects	80	24 subj.	successful / unsuccessful stop
Moran (ds109) [107]	fMRI across subjects	138	36 subj.	false picture / false belief
Henson [71]	fMRI across subjects	286	16 subj.	famous / scramble famous / unfamiliar scramble / unfamiliar
Knops [80]	fMRI, across subjects	114	19 subj.	right field / left field
HCP [46]	fMRI across subjects	8500	500 subj.	17 cognitive tasks face / shape match / rel punish / reward story / math
Oasis [98]	VBM across subjects	403	403 subj.	Gender discrimination

Table 1.2: **The different neuroimaging datasets and their experimental tasks** used throughout this manuscript.

2 Decoding – machine learning meets fMRI

In this chapter, we introduce some of the concepts required to gain an overall picture of how machine learning can help us to understand the functional structure of the brain. Brain activation *decoding* is a supervised-learning technique that tries to find a link between brain signal activity and experimental conditions/stimuli. It is a data-driven method designed to find suitable discriminative maps, allowing one to make predictions on unseen data. In addition, classification accuracy serves as whole-brain test, also known as omnibus test. Since decoding predicts quantities that are directly observable, it can serve as a complementary test to validate models that model brain activity as the linear combination of different experimental conditions (encoding models). It also allows us to test several experimental conditions across experiments, providing a way of capturing a large variety of brain processes. However, given the low sample size of fMRI datasets, finding a model that leads to reproducible results is not an easy task and additional constraints must be imposed, yielding models that are slow to train. This additional computation cost hinders subsequent analysis, which often uses data perturbation –e.g. sub-sampling, resampling, permutation tests–, where we need to fit several models to select hyperparameters or to assess the stability of the brain maps and/or the prediction power.

In fMRI data analysis, encoding models, also called forward models, model brain responses following a stimulus. On the other hand, decoding algorithms, or inverse models, perform inference in the opposite direction; predicting the stimuli form brain images. This practice is referred as *decoding* or *Multivoxel Pattern Analysis* –MVPA– in the neuroimaging community [31][63][141], and has become a central tool in neuroimage data processing [68][128][115]. A simple scheme of encoding and decoding is presented in Fig. 2.1.

An advantage of this kind of approach is that it performs a unique test for the whole brain, i.e. an omnibus test, avoiding multiple-testing problems. Its multivariate nature enables to capture distributed patterns that are predictive

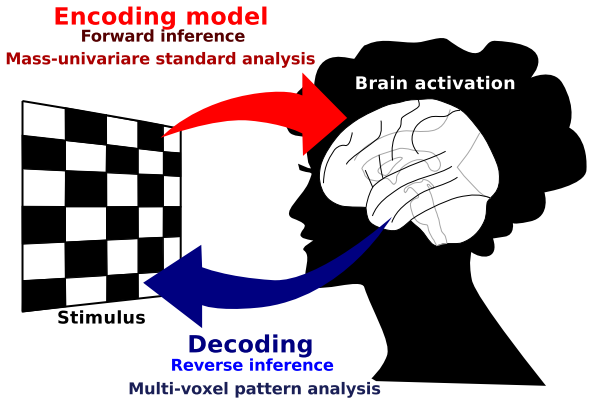


Figure 2.1: **Illustration of encoding and decoding:** Schematic of the difference between encoding and decoding models in brain imaging.

of a given target. In addition, decoding models can serve as a useful tool to validate an encoding model, as they often predict quantities that are directly observable [160]. It is often considered that decoding provides an increase in sensitivity compared to standard mass-univariate analysis [115], and it can provide a good framework to interpret overlapping activations [124].

Studying the functional specialization of the brain, requires to testing more functions than the number of conditions that can be addressed in one experimental paradigm. For this reason, investigators rely on sharing and publishing their data and results [129][59][46]. Gathering data coming from several experiments with different experimental protocols, can leverage the relevant information about cognitive concepts of interest [164], avoiding dataset-specific biases. Nevertheless, to cover more concepts, one has to deal with more data. This poses new challenges, in particular, how to handle these data and the computational costs and memory footprint of subsequent analysis.

2.1 Supervised statistical learning

Let us start with the general description of the learning problems that we tackle throughout this document. We assume two spaces $\mathcal{X}$ and $\mathcal{Y}$. We refer to $\mathcal{X}$ as the input/sample space, and $\mathcal{Y}$ as the output/target space. We assume that the input-output pair $(\mathbf{x}, \mathbf{y})$ is a random variable distributed according to an unknown probability distribution P. Given a sequence of n independent and identically distributed –i.i.d– paired training data $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ sampled from P, one would like to build the following predictive model:

$$\mathbf{y} = h(\mathbf{x}) + \epsilon, \tag{2.1}$$

where h is some fixed but unknown function of $\mathbf{x}$, and ϵ is a random error term, which is independent from $\mathbf{x}$ and has mean zero. Then, learning boils down to selecting a prediction function by solving an optimization problem.

Formally, our goal is to determine a prediction function $h : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to an output space $\mathcal{Y}$ such that, given $\mathbf{x} \in \mathcal{X}$, $h(\mathbf{x})$ offers a good prediction of the output $\mathbf{y} \in \mathcal{Y}$. This function must generalize the concepts that can be learned from a given set of examples and avoid memorization, also known as overfitting. To choose a prediction function h, we minimize a risk measure over an adequately selected family of prediction functions [159] $\mathcal{H}$ –hypothesis class.

As the function h is fixed, it can be parametrized by a real vector $\omega \in \mathbb{R}^p$ –weights/coefficients map–, $h(\cdot;\omega)$. To define the optimization problem, we assume a given loss function, $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, that measures how well the estimator performs. In other words, $l(h(\mathbf{x};\omega),\mathbf{y})$ simply indicates whether $h(\mathbf{x};\omega)$ correctly predicts $\mathbf{y}$ or not.

We proceed by minimizing the expected risk,

$$R(\omega) = \int l(h(\mathbf{x};\omega),\mathbf{y})dP. \tag{2.2}$$

Nevertheless, in practice, we do not have access to the probability distribution P representing the true relationship between inputs and outputs. For this reason, we use an estimation of the expected risk: the empirical risk. We use our training set of $n \in \mathbb{N}$ independently drawn input-output samples $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^n$ to define the empirical risk, as follows:

$$R_n(\omega) = \frac{1}{n}\sum_{i=1}^{n} l(h(\mathbf{x}_i;\omega),\mathbf{y}_i). \tag{2.3}$$

This approximation converges asymptotically ($n \to \infty$) to the expected risk, and minimizing it is taken as our problem of interest. However, for small sample sizes this approximation is coarse and large deviations are possible. As a consequence, our prediction function h *overfits* the sample data; it learns only sample specific details, rather than global properties of P.

Our interest throughout this document is brain decoding, where each $\mathbf{x}_i \in \mathbb{R}^p$ is a brain image –i.e. predictor variable– composed of p voxels. In this setting, $\mathbf{X} \in \mathbb{R}^{n\times p}$ is just the vertical stacking of all the training brain images, and $\mathbf{y}$ is the behavioral/categorical variable to be fit, i.e. the target. In the case of binary classification, $\mathbf{y}_i \in \{-1,+1\}$; in the case of regression, $\mathbf{y}_i \in \mathbb{R}$.

In brain decoding, the relationship between stimuli and each predictor can be adequately summarized using a linear model, as non-linear estimators usually yield the same performance as linear ones [104]. In addition, using a linear model can be beneficial for a better understanding of how any individual predictor is associated with the response [65]. This is a favorable property and can be helpful to advance the understanding of brain function.

Losses

In classification of fMRI data, the most common losses are the hinge loss and the logistic loss[1] [85] [126]. These losses correspond to support vector machine (SVM) and logistic regression, respectively [162]. For the SVM, the loss is flat and exactly zero for well-classified training examples with a misclassification cost that grows linearly with the distance to the decision boundary. The logistic regression uses a soft version of the hinge loss, which decreases exponentially as the training examples are well classified –see Fig. 2.2.

[1] Both hinge and logistic losses are upper-bounds of the 0-1-loss –i.e. average number of misclassified samples. They are surrogate losses of the 0-1-loss.

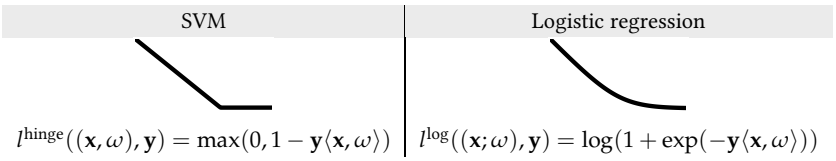


Figure 2.2: **Illustration of common losses in fMRI:** The classifiers used most often in fMRI are the support vector machine –SVM– and the logistic regression. SVMs use a hinge loss; this loss is continuous and non-smooth, assigning exactly a zero value for all well-classified points. The logistic loss uses a logistic function, which is continuous and smooth and it does not assign zero penalty to any point.

In regression, the most common loss is the mean squared error loss,

$$l^{\text{mse}}((\mathbf{x};\omega),\mathbf{y}) = \frac{1}{2}(\mathbf{y} - \langle \mathbf{x},\omega\rangle)^2. \tag{2.4}$$

Up to now, we defined our hypothesis class to be $\mathcal{H} : \{h(\cdot;\omega) : \omega \in \mathbb{R}^p\}$, but this class tends to select the most complex h, leading to overfitting. One possible solution to this problem is to include a measure of the complexity of h. This approach is closely related to the structural risk minimization [158].

Regularization

In neuroimaging, the number n of samples is typically only a few hundreds while p represents thousands of voxels: $p \gg n$. Even if the high dimensionality of the data opens the way to richer models, the low-sample regime prevents one from finding a unique solution. Then, training an estimator h is an ill-posed inverse problem. To overcome this issue and to reduce overfitting, we must add some additional constraints to the empirical risk, Eq. 2.3 [145]. This approach is known as regularized loss minimization, and is a learning rule in which we jointly minimize the empirical risk and a regularization function, as follows

$$\hat{\omega}(\lambda) \in \underset{\omega\in\mathbb{R}^p}{\operatorname{argmin}} \{R_n(\omega) + \lambda\Omega(\omega)\}, \quad \lambda > 0, \tag{2.5}$$

where Ω is a regularization function –i.e. prior information– and λ is a regularization parameter to control the penalization of the solution. This additional parameter λ, often called hyperparameter, needs to be estimated as well. To be able to use efficient optimization algorithms, we often choose R_n and Ω to be convex but not necessarily smooth.

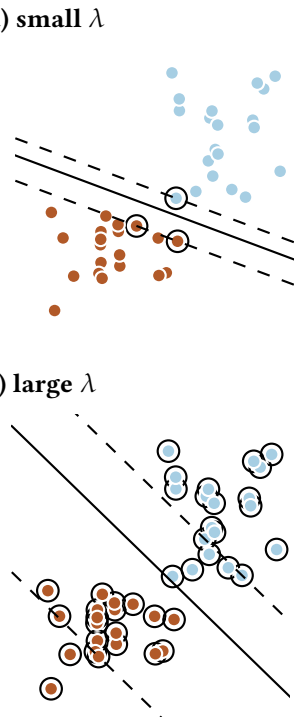


Figure 2.3: **Regularization with SVM-ℓ_2:** blue and brown points are training samples of each class. The SVM learns a separating line between the two classes. In a

In brief, a regularization function measures the complexity of the hypothesis, and allows one to control the trade-off between low empirical risk and simpler hypothesis.

Regularizers: The convex function Ω imposes structure on the solution h – the inverse problem. In machine learning and signal processing literature, the two most used penalties are the ℓ_2 and ℓ_1 penalties. The squared ℓ_2-penalty, $\|\omega\|_2^2 = \sum_{i=1}^{p} \omega_i^2$, it penalizes large values on the solution [154], whereas the ℓ_1-penalty, $\|\omega\|_1 = \sum_{i=1}^{p} |\omega_i|$, forces a large fraction of uninformative coefficients to zero: it induces sparsity [153].

When the variables are correlated as in fMRI data, the ℓ_1-penalized methods can select overly sparse solutions, leading to a high rate of false negatives in the support estimation [161], even selecting entirely different subsets of coefficients when the data are resampled. Hence, the weight maps obtained through the ℓ_1-penalty are not stable. On the other hand, the squared ℓ_2-penalty returns full brain maps, making the identification of brain regions problematic.

To mitigate this issue, we can take advantage of two properties of brain images to define the Ω regularizer: spatial structure and sparsity.

1. *Spatial structure:* In medical imaging, the acquired signals are the resulting process acting in a neighborhood structure –i.e. a topology. This yields an underlying spatial structure that can be taken into account by the regularizer.

2. *Sparsity:* There is evidence that the brain is functionally segregated into local areas that differ in their anatomy and physiology [155]. This is a strong prior, as under this assumption only a fraction of the brain is useful to predict a specific task.

   The TV-ℓ_1[103] and Graph-net [61] penalties were successfully introduced

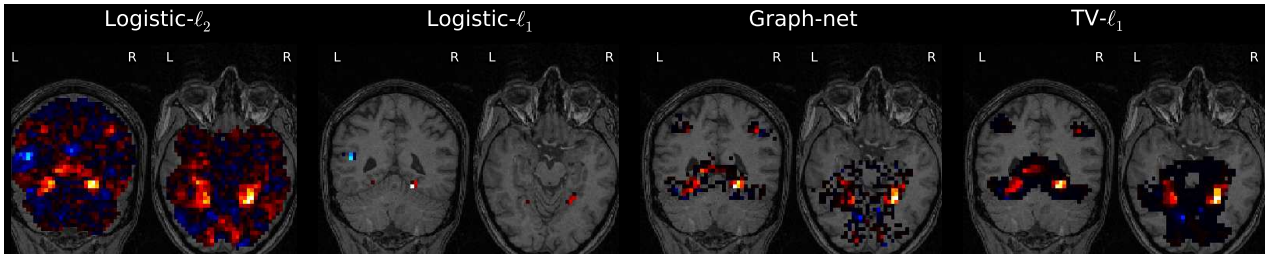


Figure 2.4: **Illustration of discriminative weights obtained through various regularizations:** This figure shows the unthresholded coefficients of a logistic regression discriminating between faces and houses. From left to right, the logistic regression penalized with: ℓ_2, ℓ_1, Graph-net, and TV-ℓ_1. The ℓ_1 penalty yields very sparse maps, avoiding locally connected regions. On the other hand, the ℓ_2 penalty imposes smoothness and assigns activity to the whole brain. Finally, structured penalizations –Graph-net and TV-ℓ_1 yield maps that respect local activation pattern, while keeping many zero coefficients.

to the neuroimaging community as a proposal to include structure and sparsity assumptions. The Graph-net penalty consists of the sum of an ℓ_1-penalty on all the coordinates and a squared ℓ_2-penalty of the spatial gradient, it promotes sparse and smooth groups. The TV-ℓ_1-penalty is defined as the sum of an ℓ_1-penalty and an $\ell_{2,1}$-group[2] penalty of the spatial gradient. Both penalties promote "blobby" coefficient maps.

Fig. 2.1 shows the coefficient maps obtained with a logistic regression and various penalties on the Haxby dataset, performing discrimination between faces and places.

[2] $\ell_{2,1}$-penalty, written $\|\mathbf{v}\|_{2,1} = \sum_{g\in\mathcal{G}} \|\mathbf{v}_g\|_2$

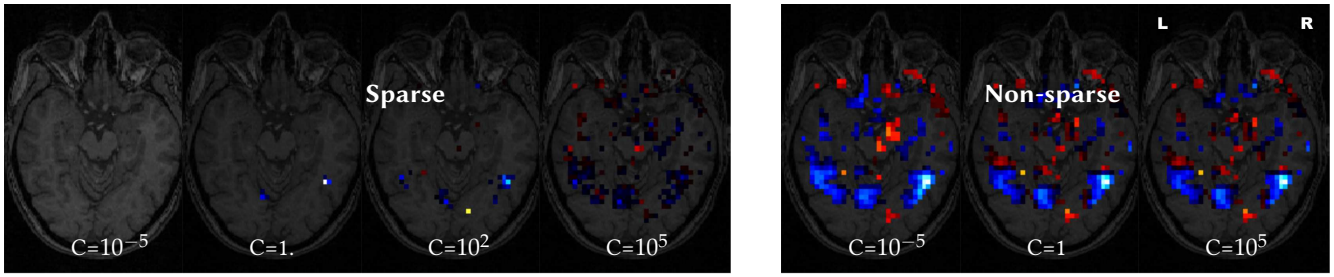


Figure 2.5: **Varying amount of regularization** on the face vs house discrimination in the Haxby 2001 data [67]. *Left)* with an ℓ_1 logistic regression, more regularization induces sparsity. *Right)* with a SVM-ℓ_2, more regularization means that weight maps are a combination of a larger number of images of the training set, although this has only a small visual impact on the corresponding brain maps. Source [162].

2.2 Validation and model selection

In neuroimaging, the role of a decoder is to predict a stimulus/condition or health status given a brain image. Hence, we use the accuracy or predictive power to assess its performance on this task. Accuracy is defined as the expected error on the prediction, formally:

$$\text{accuracy} = \mathbb{E}[f(\mathbf{y}_{\text{predicted}}, \mathbf{y}_{\text{true}})] \tag{2.6}$$

where f is function to measure the error. For classification, typical examples are the 0-1 loss, and functions that consider both precision[3] and recall[4], e.g. F_β-score, or receiver operating characteristic (ROC). For multi-class problems, where there is more than 2 categories in $\mathbf{y}$, or for unbalanced classes, a more elaborate choice is advisable, to distinguish misses and false detections for each class [162].

Model selection

The role of *model selection* is to choose a statistical model from a class of models which captures the global properties of the distribution P. To make the choice, we measure how the model generalizes to unseen data. Hence, it gives an idea of how well it performs in real scenarios.

Many model selection schemes have been proposed, each one designed for a particular application. One can perform model selection with a sequence of hypothesis tests [35]. Another approach uses information theoretic criteria

[3] Precision is the number of correct positive results divided by the number of all positive results

[4] Recall is the number of correct positive results divided by the number of positive results that should have been returned.

[7][143], that measure the information loss between candidate models and an approximation of the true model. Most schemes perform the selection in terms of predictive power of the model on unseen data, e.g. by cross-validation.

Data perturbation – cross-validation and related methods: In fMRI decoding studies, we often use data-driven approaches to select a suitable model; we subsample or resample our dataset to generate pseudo-datasets. We fit a model on each of these pseudo-datasets. Then, we evaluate the trained model on unseen data and select the model with the best average predictive ability.

For cross-validation, we iteratively divide the data into a training and testing set, where we learn the prediction function on the training set and evaluate it on the unseen test set. There are several cross-validation schemes. Here, we describe some of the most popular classes [9]. This choice is not representative of all the literature:

- *Evaluate exhaustively data splitting:* Every possible subset of p data points is let out for assessing the predictive ability, e.g. leave-one-out, leave-p-out.
- *Partial data splitting:* Considering the combination of all possible subsets is computationally intractable. Then, as an alternative we can select disjoint sets at random, avoiding testing all possible sets, e.g. K-fold, monte-carlo CV.
- *Other cross-validation-like risk estimators:* Additionally, we have several methods that were designed to overcome possible drawbacks of cross validation. For instance, in case of discontinuous error functions, *Bootstrap* [42] can smooth over possible discontinuities. Bootstrap is a data resampling method for approximating the sampling distribution of a statistics and its characteristics. For instance, it can be used to construct a boostraped estimate of the Kullback-Leibler divergence [146].

Hyperparameter selection

We try to assess the task-related neural information at every location in the brain via decoding models. Unfortunately, we are in high-dimensional settings as the number of parameters of these models is much larger than the sample size. For this reason, we need to impose some prior knowledge, applied through a regularization function. This penalization allows one to control the trade-off between *underfitting* and *overfitting*.

In words, *underfitting* happens when the resulting model is too constrained by the prior, so that it does not exploit the richness of the data. On the

other hand, with *overfitting*, the model does not generalize as it learns even the sampling noise. Both cases, *underfitting* and *overfitting*, are detrimental for the predictive power of the model, thus the regularization parameter has to be chosen carefully. Roughly speaking, the selection of the regularization parameter is a problem of bias-variance trade-off, and the choice is determined by the predictive power of the model [162]. In general, the best trade-off is a data-specific choice.

In practice, we often use nested cross-validation to set the regularization parameters[5]. In this scheme, the data are repeatedly split into a decoding set and a validation set. The decoding set is split in multiple train and test sets, forming an inner "nested" cross-validation loop in which we select the hyperparameter, whereas the external loop varying the validation is used to measure prediction performance –see Fig. 2.6.

[5] Let us note that the amount of penalization will change the appearance of the weight/coefficient map and the predictive power as well.

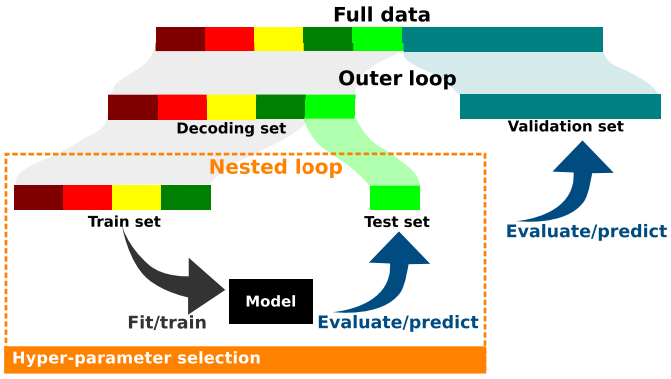


Figure 2.6: **Nested cross-validation:** Two cross-validation loops are run one inside the other. The inner/nested loop is used to select the hyperparameter of the estimator –i.e. the amount of penalization–, whereas the outer loop is used to assess the predictive power of the decoder.

Fig. 2.7 is a didactic view on the parameter-selection problem, and shows the tuning curves of several decoders for the discrimination between scissors and scramble on the Haxby dataset. The discrepancy between the tuning curve, computed with nested cross-validation and the validation curve, is an indication of the uncertainty on the cross-validated estimate of prediction power.

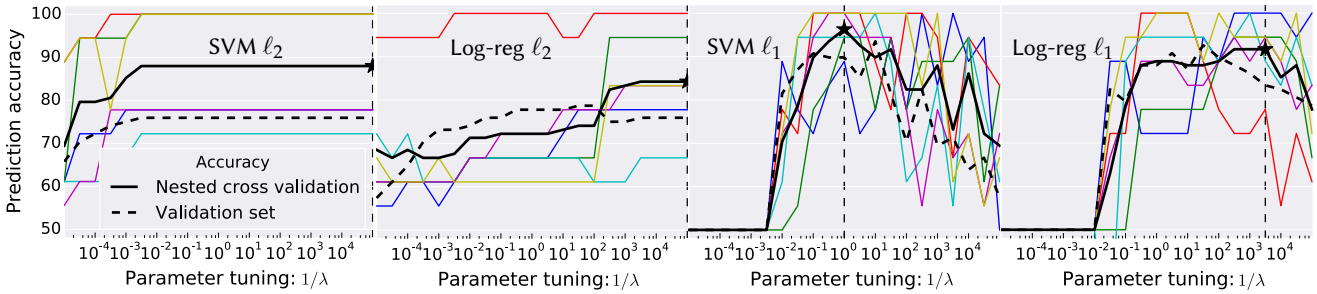


Figure 2.7: **Tuning curves** for SVM-ℓ_2, logistic regression-ℓ_2, SVM-ℓ_1, and logistic regression-ℓ_1, on the scissors / scramble discrimination for the Haxby dataset [67]. The thin colored lines are test scores for each of the internal cross-validation folds, the thick black line is the average

2.3 Stability – the need for reproducibility

Reproducibility is imperative for any scientific discovery. In neuroimaging, more often scientific findings are based on statistical analysis of high-dimensional data. At minimum, a necessary condition for reproducibility is the stability of statistical results relative to samplings/resamplings of the data. Hence, we need stability for interpretable and reliable encoding/decoding models.

Stable coefficient/weight maps: As in many fields of applied mathematics as numerical analysis or partial differential equations, we need stability to deal with real world problems. In machine learning, the concept of stability is often associated with good generalization performance [18]. In a broader sense, we consider that statistical stability holds if the predictive power of the model is robust to appropriate perturbation of the data, i.e. bootstrap, cross-validation, etc.. In brief, statistical stability differs from robust statistics as the former deals with data perturbation, while robust statistics methods deal with perturbations of model assumptions.

In neuroimaging studies, decoders solve a high-dimensional multivariate statistical estimation problem that is very ill-posed given the sample size. This leads to degenerate solutions: many brain maps yield the same predictive performance. We use regularization to overcome this issue, but the increase in the computation cost is directly proportional to the complexity of the penalty.

Brain signals are spatially correlated, hence ℓ_1-penalized estimators are naturally unstable. However, there are several methods to improve the stability of the solution given by this kind of estimators. One approach is called stability selection. This method consists of building an empirical distribution of the support (non-zero voxels) using data randomization, then based on an arbitrary threshold, a linear model is fit on the selected voxels [12][101]. A second approach is based on a selection of a regularization parameter, using as criterion the variance of the predictive power of the model through several data perturbations [175][88].

Another approach to mitigate the instability of sparse estimators is to group similar features together, reducing multicollinearity effects. One way to do this is to apply clustering as a preprocessing step, using clusters centroids to fit the sparse model [22]; we can also use grouping penalties –e.g. Graph-net or TV-ℓ_1, or we can estimate the stability of the support of groups of features, combining clustering and stability selection ideas [161].

Statistical significance maps: We typically use permutation tests[6] to assess which features drive the decision boundary of the estimator [108][167].

[6] Permutation tests consist of randomly exchanging labels on data points when performing significance tests.

This method is used to establish a null distribution on the weight vector components at each voxel. But these permutation tests are extremely expensive computationally, and hence prohibited in practice. In response to this problem, an analytical approximation of the distribution of the weights of an SVM with ℓ_2 penalty is proposed in a large-sample size regime [53]. Therefore, the problem remains open for low sample sizes, which is actually the case we are interested in.

2.4 Discussion – about brain decoding

We have shown that methods to decode brain images are susceptible to overfitting. Hence, we have to use out-of-sample data to evaluate their predictive power.

Regarding interpretation, decoding allows potential insights in to brain function or structure that drives prediction, but we have to recall that it tests the overall significance of the model. Hence it does not test which variables have a significant contribution to the model [66]. Nevertheless, decoding is a complementary analysis tool to encoding models [169]. In addition, decoding also offers a possibility to ask new questions about the functional structure of the brain, making it possible to analyze jointly several datasets with different experimental conditions[7]. This can be leveraged with the accumulation of task-based fMRI data.

[7] This can also be done with univariate methods, but it is not explored in this thesis.

We have shown how state-of-the-art sparse decoding methods, namely Graph-net and TV-ℓ_1, take the underlying structure of fMRI signals into account, yielding more stable results than the vanilla ℓ_1-penalty. Hence, it deals with the multicollinearity problem.

We have presented the use of data perturbation schemes –e.g. bootstrap, cross validation, etc– through all the steps of the analysis pipeline, as it is used to assess the predictive power of the model, to set its hyperparameters, and even to evaluate the stability of the resulting weights/coefficients. However, we need to train the estimator a large number of times[8], and performing this becomes prohibitive if fitting the estimator takes time, as it is the case for spatially-structured estimators[9]. While there have been some attempts to speed up such estimators [38][37], the computational cost hinders the use of permutation tests.

[8] The number of times depends on the sample size and an arbitrary desired precision

[9] Graph-net can take weeks in a standard workstation, as it is reported in [106]

In the next chapters, we will present a way to improve the stability of the prediction power and the coefficients, while reducing the computational cost with a fast and highly distributed approach.

Part II

Contribution – fast decoding

3 Faster brain decoding with classic data approximations schemes

In this chapter we present a first attempt to speed up brain activation *decoding*. These multivariate techniques are becoming standard brain mapping tools, but they entail much larger computational costs. With datasets growing, acquiring larger cohorts and higher-resolution imaging, this cost is increasingly a burden. Here we consider the use of random sampling and projections as fast data approximation techniques for brain images. We evaluate their prediction accuracy and computation time on various datasets and discrimination tasks. We show that the weight maps obtained after random sampling are highly consistent with those obtained with the whole feature space, while having a fair prediction performance. Altogether, we present the practical advantage of random sampling methods in neuroimaging, showing a simple way to embed back the reduced coefficients, with only a small loss of information.

The work presented in this chapter has been published in:

Fast brain decoding with random sampling and random projections, Andrés Hoyos-Idrobo, Gaël Varoquaux and Bertrand Thrion. *PRNI - IEEE International Workshop on Pattern Recognition in NeuroImaging*. 2016, Trento, Italy.

3.1 Introduction – brain decoding and datasets' growing sizes

As we introduced in previous chapters, *decoding* uses predictive models to link brain regions with an experimental condition or a behavior. It has become a central tool in neuroimaging [119]. In particular, linear estimators can highlight the brain maps that lead to the identification of cognitive labels [109][111]. Yet, to date, decoding is still orders of magnitude slower than

standard analysis. This discourages the use of non-parametric hypothesis testing –*e.g.* permutation testing. Additionally, large cohorts are needed to fully tap the potential of decoding, increasing both power and reliability in group studies. A striking example is the Human Connectome Project [46] –30 Terabytes of data and growing.

Increasing data sizes pose tractability challenges for all processing steps, but this is especially true for multivariate statistics: multivariate estimators entail high computation costs. The literature of machine learning on massive datasets often relies on dimension reductions to mitigate the impact of data size on computational cost. Data reduction is very beneficial, as first, it reduces memory requirements, and, as a consequence, it decreases computation time.

Various approaches are used as standard tools in the literature to compress datasets. Typically, a data matrix is represented by a *sketch matrix* –random linear image of the matrix–[87] that is significantly smaller than the original, but approximates it well. Two sketching strategies are common employed: *i)* approximating the matrix by a small subset of its rows (or columns) (e.g. Nyström [55] and CUR [97]); *ii)* randomly combining matrix rows, relying on subspace embedding and strong concentration phenomena, e.g. random projections [77]. Random projections are appealing as they come with theoretical guaranties quantifying the expected distortion.

These sketching techniques can be used to render tractable in the large-data limit a model like PCA, which has a computational cost that grows super-linearly.

Distance-based estimators

Given the paired data samples $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, where each $\mathbf{x}_i \in \mathbb{R}^p$ is a brain image –*e.g.* predictor variable– and each $\mathbf{y}_i \in \mathbb{R}$ is the behavioral/categorical variable to be fit –*e.g.* *the target*. The goal is to estimate a function that can be used to predict future responses based on observing only brain images. Henceforth, the data are represented as a matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, n observed brain images composed of p voxels.

Kernel-based methods: Here, we give a brief reminder about kernels. A kernel is a function that quantifies the similarity of two observations (e.g. pairwise distance between brain images). Kernel-based methods use a feature mapping $\mathbf{\Phi}$ to reveal the discriminant informations in a high-dimensional space $\mathcal{F}$. In brief, a kernel-method pipeline is: *i)* embedding the data $\mathbf{X}$ into $\mathcal{F}$ using the feature mapping $\mathbf{\Phi}$, and *ii)* performing the estimation (e.g. classification). In neuroimaging, the feature mapping $\mathbf{\Phi} : \mathbb{R}^p \to \mathcal{F}$ is often chosen as linear for interpretability of the weights [109].

Kernel-based methods rely on the idea that inner products in high-dimensional feature spaces can be computed in implicit form via kernel

function $\mathbf{K}_{i,j} = \langle \mathbf{\Phi}(\mathbf{x}_i), \mathbf{\Phi}(\mathbf{x}_j) \rangle$ resulting in the Gram matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$. This is important, because the decision function of many classification algorithms (e.g. SVM and logistic regression) can be carried out just on the basis of the values of the kernel function over pairs of domain points. The kernel function in a linear setting leads to a symmetric positive semidefinite matrix $\mathbf{K}_{i,j} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ [1].

[1] From the viewpoint of geometry, in a Euclidean space, the distance describes the dissimilarity of a pair of points while the inner product describes the similarity. But, they are both simply related to each other if the data are centered.

3.2 Background – matrix sketching as fast dimension reduction technique

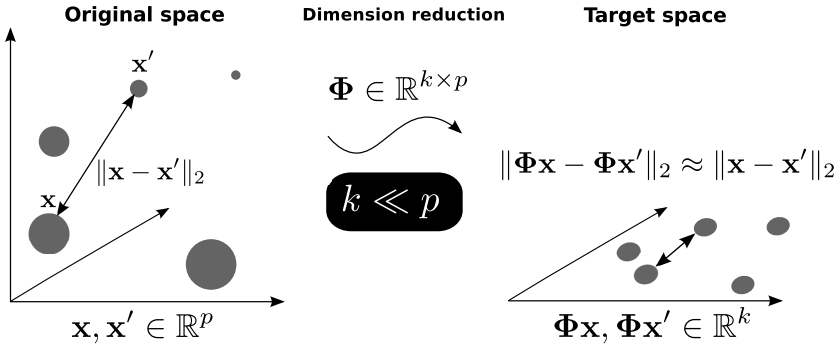


Figure 3.1: **Lipschitz embedding via linear mapping:** Perform a linear dimension reduction with a mapping that preserves the Euclidean distance between the points

Signal approximation with random projections or random sampling techniques is now central to many data analysis, machine learning, or signal processing algorithms.

Johnson-Lindenstrauss-Embeddings

Let $\mathbf{X} \in \mathbb{R}^{p \times n}$ be a data matrix composed of n samples and p features (*i.e.* pixels/voxels). We are interested in an operator $\mathbf{\Phi} \in \mathbb{R}^{k \times p}$ that reduces the dimension of the data in the feature direction, acting as a preprocessing step to make further analysis more tractable. This operator should maintain approximately the pairwise distance between pairs of images $(\mathbf{X}_{*,i}, \mathbf{X}_{*,j}) \in \mathbf{X}^2$ for $(i,j) \in [n]^2$,

$$\|\mathbf{\Phi}\, \mathbf{X}_{*,i} - \mathbf{\Phi}\, \mathbf{X}_{*,j}\|_2^2 \approx \|\mathbf{X}_{*,i} - \mathbf{X}_{*,j}\|_2^2, \quad \forall (i,j) \in [n]^2. \tag{3.1}$$

Note that this approximation needs to hold only on the data submanifold, and not the entire $\mathbb{R}^p$.

Random projections: A standard choice, is to build $\mathbf{\Phi}$ with random projections, $\mathbf{\Phi}_{\mathrm{RP}}$ [69].

It is particularly attractive due to its algorithmic simplicity and theoretical guaranties that make it ϵ-isometric (see Eq. 3.2).

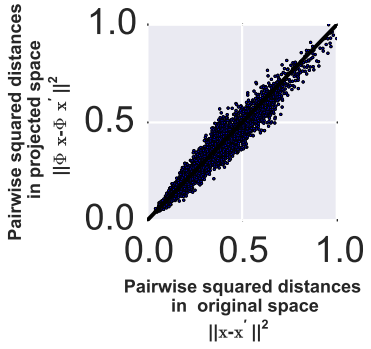


Figure 3.2: **Random projections is an ϵ-embedding:** Random projections preserve the pairwise squared distance in the projected space with a distortion up to ϵ, where the number of projections required is $O(\log(n)/\epsilon^2)$.

Lemma 3.2.1. *[77] By the Johnson-Lindenstrauss lemma, the pairwise distances among a collection $\mathcal{X}$ of n-points in $\mathbb{R}^p$ are approximately maintained when the points are mapped randomly to an Euclidean space of dimension $k = O(\epsilon^{-1} \log n)$ up to a distortion at most ϵ. More precisely, given $\epsilon, \delta \in (0,1)$ and $k \leq p$, there exists a random linear projection $\mathbf{\Phi}_{\mathsf{RP}} : \mathbb{R}^p \to \mathbb{R}^k$ such that for every $\mathbf{x}, \mathbf{x}'$ in $\mathcal{X}$, the following relations hold:*

$$(1-\epsilon)\|\mathbf{x}-\mathbf{x}'\|_2^2 \leq \|\mathbf{\Phi}_{\mathsf{RP}}\,\mathbf{x} - \mathbf{\Phi}_{\mathsf{RP}}\,\mathbf{x}'\|_2^2 \leq (1+\epsilon)\|\mathbf{x}-\mathbf{x}'\|_2^2, \qquad (3.2)$$

with probability at least $1-\delta$.

These Johnson-Lindenstrauss embeddings have been widely used in the last years. By providing a low-dimensional representation of the data, they can speed up algorithms dramatically, in particular when their run time depends super-linearly on the dimension of the data. In addition, as this representation of the data is accurate in the sense of the ℓ_2 norm, it can be used to approximate shift-invariant kernels [94][136].

The $\mathbf{\Phi}_{\mathsf{RP}}$ matrix can be generated by sampling from a Gaussian distribution with rescaling. In practice, a simple and efficient generation scheme can yield a very sparse random matrix with good properties [4]. The computational performance of random projections can be further improved *i)* with fast orthogonal decompositions [6] [156], *ii)* by reducing the number of necessary projections when the data lie on a submanifold of $\mathbb{R}^p$ [14].

This approach suffers from two important limitations: *i)* inverting the random mapping from $\mathbb{R}^p$ to $\mathbb{R}^k$ is difficult, requiring more constraints on the data (e.g. sparsity), which entails another estimation problem. As a result, it yields less meaningful or easily interpretable results, as the ensuing inference steps cannot be made explicit in the original space. *ii)* This approach is suboptimal for structured datasets, since it ignores the properties of the data, such as a possible spatial continuity.

Random sampling – interpolative decomposition

A related technique is random sampling, and in particular the Nyström approximation method. This method is mainly used to build a low-rank approximation of a matrix. It is particularly useful with kernel-based methods when the number n of samples is large, given that the complexity of building a kernel matrix is at least quadratic in n [170]. It has become a standard tool when dealing with large-scale datasets [55].

The key idea is to preserve the spectral structure of a kernel matrix $\mathbf{K}$ using a subset of columns of this matrix, yielding a low-rank approximation. This can be cast as building a data-driven feature mapping $\mathbf{\Phi}_{\mathsf{Nys}} \in \mathbb{R}^{k \times p}$. In a linear setting, the kernel matrix is defined as $\mathbf{K} = \mathbf{X}^\top \mathbf{X}$, which leads to the following approximation:

$$\mathbf{K}_{i,j} = \left\langle \mathbf{X}_{*,i}, \mathbf{X}_{*,j} \right\rangle \approx \left\langle \mathbf{\Phi}_{\mathsf{Nys}}\,\mathbf{X}_{*,i}, \mathbf{\Phi}_{\mathsf{Nys}}\,\mathbf{X}_{*,j} \right\rangle. \qquad (3.3)$$

Here, building a basis $\mathbf{\Phi}_{\mathsf{Nys}}$ is achieved by randomly sampling $k \ll n$ points from $\mathbf{X}$, and then normalizing them –*i.e.* whitening the subsampled data– see algorithm 1. The cost of the SVD dominates the complexity of this method $O(pk \min\{p,k\})$. This method is well suited for signals with a common structure, for instance images that share a common spatial organization captured by $\mathbf{\Phi}_{\mathsf{Nys}}$. As the Nyström approximation captures the structure of the data, it can also act as a regularization [138].

Nyström feature mapping: Here, we present the standard implementation of the Nyström approximation for linear kernels. Algorithm 1 allows to build a data-driven feature mapping that is used to reduce the dimensionality of the data matrix. The algorithm is summarized as follows: first, we select images uniformly at random[2], then we calculate the kernel of these samples and use it to normalize the selected images.

[2] We can use other sampling probabilities (e.g. the leverage score [96]), but by sampling uniformly we are assuming a regular structure

Algorithm 1 Nyström: Learning the feature mapping

Require: The training data matrix $\mathbf{X} \in \mathbb{R}^{p \times n}$, number k of components, where $k < n$.

Ensure: The feature mapping $\mathbf{\Phi}_{\mathsf{Nys}} \in \mathbb{R}^{k \times p}$

1: $\mathbf{r} \leftarrow$ Generate uniform sampling of k components
2: $\mathbf{X}_{*,\mathbf{r}} \in \mathbb{R}^{p \times k}$ {Subsample of k columns}
3: $\tilde{\mathbf{K}} = \mathbf{X}^{\mathsf{T}}{}_{*,\mathbf{r}}\, \mathbf{X}_{*,\mathbf{r}}$ {Kernel matrix of the subsampled data}
4: $\mathbf{\Phi}_{\mathsf{Nys}} = \tilde{\mathbf{K}}^{-1/2} \mathbf{X}_{*,\mathbf{r}}$ {Normalization: via SVD}
5: **return** $\mathbf{\Phi}_{\mathsf{Nys}}$

3.3 Experiments – empirical verification

In this section, we investigate brain decoding after random sampling and random projections. To achieve reliable empirical conclusions, we evaluate the performance across several neuroimaging studies, using both anatomical and functional images. We compare prediction accuracy obtained without compression to that using random projections and Nyström approximation under linear settings. We also quantify and compare the execution time. In all the experiments, $n > 180$ and we split the data into train and test set, changing the proportion of these sets according to the dataset. All dimensionality reduction procedures are calibrated on the train set and used to reduced the test set.

Datasets

Detailed descriptions of these datasets are given in section 1.5.

Haxby [67]: We perform a binary discrimination between pairs of visual stimuli. The prediction is performed on two left-out sessions.

The Open Access Series of Imaging Studies (OASIS)[98]: We perform across-subject gender discrimination, leaving half of the subjects out to measure the accuracy.

Human Connectome Project (HCP)[46]: Form this dataset, we took images related to 5 tasks: *1)* working memory/cognitive control processing, *2)* incentive processing, *3)* visual and somatosensory-motor processing, *4)* language processing (semantic and phonological processing), *5)* social (theory of mind). We perform across-subject discrimination of 17 experimental conditions selected from the aforementioned task-related datasets.

Benchmarking of linear classifiers

We explore standard classifiers of the neuroimaging literature: SVM-ℓ_2 and logistic regression-ℓ_2. Firstly, we analyze the performance of various standard solvers in the primal and dual space[3]: *i)* for the SVM we use Liblinear and LibSVM, setting the regularization parameter by inner cross validation. *ii)* For the logistic regression, we use Liblinear with inner cross validation to set the regularization parameter; and L-BGFGS with warm start setting, the parameter via regularization path. To build the confidence interval, we perform a 10-fold cross validation maintaining the proportion between the labels at each iteration. We measure the accuracy[4] obtained on test data and the computation time to train the classifier. We compare the performance of all classifiers with an SVM (LibSVM), which is often used by default.

[3] Note that not all the algorithms are designed to work in both spaces (primal and dual), this is the case of LibSVM which only works on the dual space

[4] Accuracy is defined by: $\frac{\text{number of correctly predicted data}}{\text{total number of samples}} \times 100\%$

Fig. 3.3 displays the results of the discrimination of visual objects on the Haxby dataset and the gender prediction on the OASIS dataset. These tasks cover a range from easy to difficult discrimination problems. The prediction accuracy of the logistic regression (LibLinear) is statistically different to the SVM (LibSVM) $-p < 0.001$, paired Wilcoxon rank test. The estimators using a primal solver yield the same distribution.

Regarding computation time, logistic regression-ℓ_2 using L-BFGS with warm start has the best performance: it displays a good trade-off between prediction accuracy and computation time. Henceforth, we refer to this choice simply as logistic regression-ℓ_2, making the solver implicit.

Bringing the reduction to the brain space

In this experiment, we show the capability of the Nyström method to approximate the coefficients of a linear estimator in the brain space. We use a logistic regression-ℓ_2 as a classifier, using as a solver an L-BFGS with warm start, to discriminate 7 paired visual objects on the Haxby dataset, and gender on the OASIS dataset. We compare the prediction accuracy and the correlation, between the weight maps obtained using all the voxels with the approximation via Nyström method for $k = 100$.

Fig. 3.4 shows the weight map (hyperplane of discrimination), to

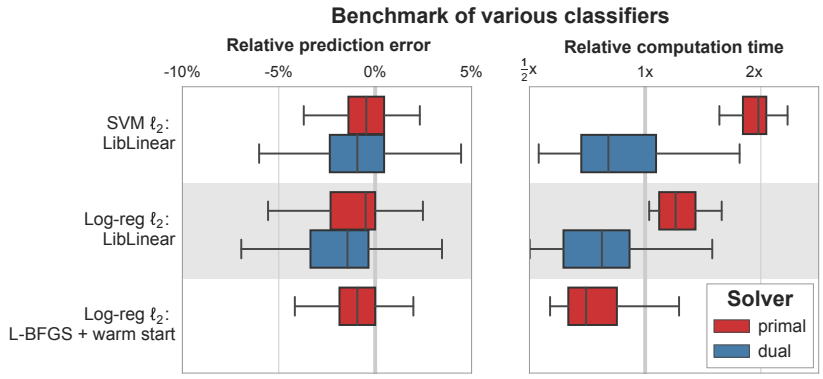


Figure 3.3: **Comparison of the performance of various solvers relative to SVM (LibSVM):** Comparison of the performance of the SVM and logistic regression with ℓ_2 penalty on the discrimination of 7 paired visual object recognition tasks for the Haxby dataset, and the discrimination of gender of the OASIS dataset. *(left)* The prediction accuracy of the logistic regression using LibLinear is statistically different to the obtained using the SVM (LibSVM); *(right)* Regarding the computation time, the logistic regression using L-BFGS with warm start, displays a good trade-off between computation time and accuracy.

discriminate between face and house. Contours show the well-known Fusiform Face Area (FFA) and Parahippocampal Plane Area (PPA) regions, respectively involved in the face and house recognition tasks. They are highlighted by the coefficients of the classifier. This is considered as an easy classification task and finding the structure by Nyström approximation only requires a small number k of components ($k > 30$).

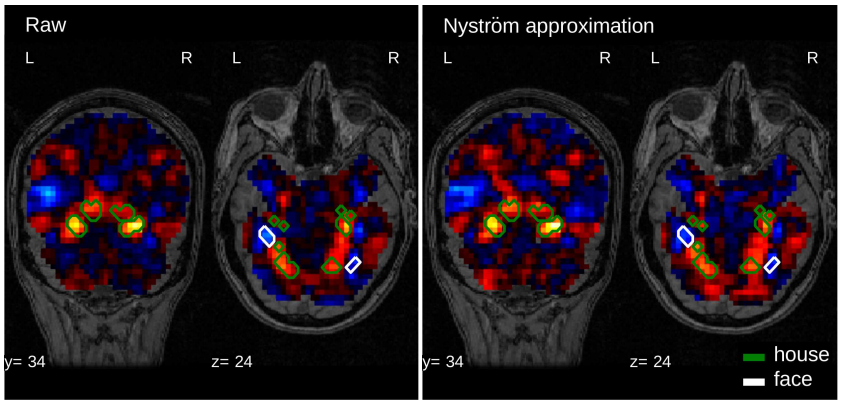


Figure 3.4: **Approximation in the brain space:** Weight maps (unthresholded) of a ℓ_2-logistic regression, obtained for the discrimination of face and house on the Haxby dataset. The contours show the FFA (green) and PPA (white) regions, respectively involved in the face and house recognition tasks. These regions are highlighted by the coefficients obtained using two methods: *(left)* the whole feature space, and *(right)* the Nyström method with $k = 100$.

Fig. 3.5 shows the consistency between the discriminative weights found after applying the Nyström method and raw data. Nyström displays a high consistency with the raw data, having a correlation score > 0.7 for all the conditions. Regarding the prediction accuracy, we can see that the Nyström method exhibits only slightly worst performance than raw.

Random sampling, random projections and decoding

Now, we compare the effect of random sampling and projections across different discrimination tasks and datasets. To this end, we use 7 visual object discrimination of the Haxby dataset, gender discrimination of the OASIS dataset, and 17 cognitive tasks of the HCP dataset. We train a binary logistic regression on the first two datasets and multinomial logistic regression on

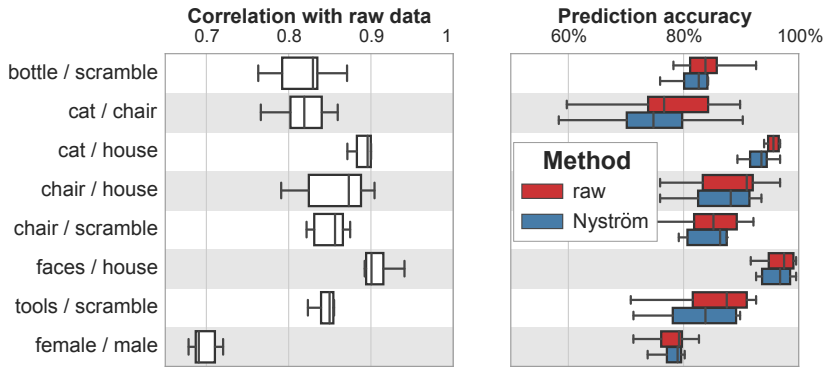


Figure 3.5: **Consistency of the discriminative weights after dimensionality reduction:** Discrimination of various conditions using a logistic regression with ℓ_2 penalty. The dimension is reduced to $k = 100$. *(left)* Correlation between coefficients obtained using the raw data and the Nyström method. The weight maps found after approximation are generally consistent with the maps found without reduction; *(right)* Regarding prediction accuracy, the Nyström method is comparable with the performance obtained with the raw data.

the last one. We reduce the dimensionality of the feature space form p to k, where k is set to $k = 100$ in this experiment. We use random projections and Nyström method to carry out this task.

The results of the use of random sampling and projections with respect to raw data are summarized in Fig. 3.6. We can see that the accuracy of the classifier after dimensionality reduction by Nyström method is close to the one obtained using the whole feature space. This indicates that there is a reliable linear structure underlying the brain images, which is captured by Nyström approximation with only a small number k of components. In contrast, the estimator after random projections shows lower performance. This is because random projections act in the feature direction, needing a larger number k of components to approximate the pairwise distances.

Regarding computational time, both methods have an equal performance in average. Note that using the Nyström method yields impressive time savings on the HCP dataset.

Implementation aspects: We rely on scikit-learn[123] for machine learning tasks (logistic regression and SVM) and on Nilearn[2] to interact with neuroimaging data.

3.4 Discussion – brain decoding with random sampling

Our validation over 27 decoding tasks on 3 different datasets, varying from moderate to large size datasets, shows that random sampling overperforms random projections for decoding brain images. The dimensionality reduction by random projections does not take the structure of the signal into account, making it difficult to find an appropriate pseudo-inverse to bring the weight maps back into the brain space. On the other hand, the Nyström method tries to approximate the spectral properties of the data matrix $\mathbf{X}$, relying on a data-driven approach. In this sense, it takes into account the intrinsic structure –e.g. smoothness, latent network structure– being consistent with the feature

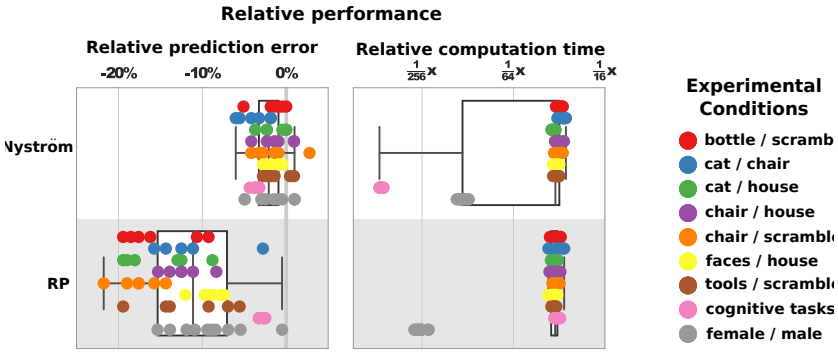


Figure 3.6: **Impact of the dimensionality reduction on the prediction performance:** Discrimination using logistic regression of 25 conditions on 3 datasets, after dimensionality is reduction to $k = 100$. *left)* Each point represents the impact of the corresponding dimensionality reduction scheme on the prediction accuracy, relatively to the prediction obtained with raw data. The Nyström approximation method has a better performance than random projections. *right)* In most of the conditions, the time performance of both methods is almost the same, yielding impressive time saving. On the HCP dataset, Nyström is considerably faster while keeping the same prediction performance as random projections.

space and controlling the spatial maps. This leads to an easy scheme to embed the coefficients back in brain space, making it possible to perform further analysis on brain maps.

Regarding computation time, both methods yield impressive speed gains: more than 16 times faster. However, the prediction accuracy is not better than that obtained with raw. Indeed, these methods do not separate the signal from the noise. In our setting, brain decoding, brain images present a continuous spatial structure, whereas noise is assumed to be unstructured. Hence, both signals can be separated using local spatial information; it would be interesting to investigate adding the spatial structure as a prior to the dimension reduction scheme. We think that this approach is better suited for medical images, it could reduce noise, and improve subsequent analysis.

4 Dimension reduction of structured signals by feature grouping

In the previous chapter we have introduced *matrix sketching* as a technique to reduce the dimensionality of the signal in brain decoding settings. We have presented some MRI experiments showing a better performance of data-driven dimension-reduction methods outperform fully random ones.

In this chapter, we assume that the signal of interest has a spatial structure. Thus, it can be modeled as a generative process acting on a neighborhood, e.g. in brain activation images, if a voxel displays activity is highly probable that its neighbors are active too. Under this assumption, the features are locally connected and the grouping of features –*feature grouping*– can serve as a dimension reduction method, summarizing the data to decrease computational costs and memory footprint of subsequent analysis. Such scheme adapts to common statistics across the data, unlike random projections which only preserves the Euclidean distance between data points.

Throughout this chapter we consider *feature grouping* as a dimension reduction scheme of structured signals. In particular, we analyze the approximation of the spectral norm of the signal, giving a worst-case bound that is useful to understand the properties that ensure a good approximation. We also show the spatial condition required to reduce unstructured noise –i.e. denoising–, increased statistical power.

The contributions developed in this chapter, along with the next two, have been submitted to:

Recursive nearest agglomeration (ReNA): fast clustering for approximation of structured signals. Andrés Hoyos-Idrobo, Gaël Varoquaux, Jonas Kahn and Bertrand Thrion. *IEEE TPAMI - Transactions on*

Pattern Analysis and Machine Intelligence.

4.1 Introduction – high-dimensional signals and large-scale datasets are now everywhere

Cheap and ubiquitous sensors lead to a rapid increase of data sizes, in the sample direction –the number of measurements –but also the feature direction– the richness of each measurement. These "big data" put a lot of strain on data management and analysis. Indeed, they entail large memory and storage footprint, and the algorithmic cost of querying or processing them is often super-linear in the data size. Yet, such data often display a structure, for instance originating from the physical process probed by the sensor. This structure implies that the data can be well approximated by a lower-dimensionality representation, dropping drastically the cost of subsequent data management or analysis.

The deluge of huge sensor-based data is ubiquitous: in imaging sciences – *e.g.* biological [8] or medical [46]– in genomics [120][29], with time series, as in seismology [5]. Researchers have introduced a variety of strategies to mitigate the computational costs created by the rapid increase in signal resolution. Many approaches integrate reduced signal representations in statistical analysis.

Super-voxels, leveraging the image structure, accelerate optical flow algorithms for huge biological microscopy images [8]. Fast large-scale image search can use a reduced representation of the images combining mid-level features with the image topology [76]. For information retrieval, state-of-the-art indexing of times series can be achieved with a symbolic representation [89] that finds a regular piecewise constant approximation of the signal, and then associates a symbolic code with this quantization. In genomics, clustering can extract a small number of *candidate* SNPs (single nucleotype polymorphisms) from a large set of SNPs, speeding up subsequent analysis [137][33]. Using the topology of the DNA strand to restrict this clustering makes it faster and more relevant for genomic studies [33]. Similarly, as medical images exhibit strong spatial structure, clustering [152][73] or super-voxels [95][166] are used to speed up statistical analysis. Finally, in the analysis of large seismic time-series [5], building indexes of the database is crucial for fast retrieval of similar waveforms. Dimension reduction is a good strategy for such fast indexing [79][5], even more so if it captures the structure of the signals [27].

Here, we are interested in representing *structured* signals, and using this structure to improve the data approximation and speed up its computation. Signal processing often models such signals as generated from a random process acting on a neighborhood structure. Individual features of the data then form vertices of a graph [147]. This structure graph is given by the

specificity of the acquisition process, such as the physics of the sensors. Note that such a description is not limited to regular grids, such as time-series or images, and encompasses for instance data on a folded surface [148].

Additionally, we want this scheme to account for the graph structure of the data. For this, we use *feature grouping*, approximating a signal with constant values over a partition of features. We use a clustering algorithm to adapt the partition to the data statistics.

4.2 The feature-grouping to approximate structured signals

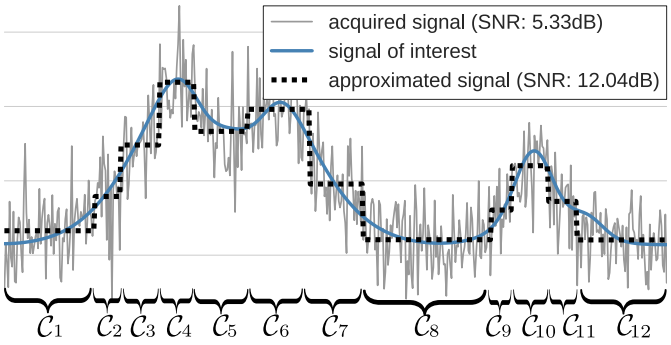


Figure 4.1: **Illustration of the approximation of a signal:** Piece-wise constant approximation of a 1D signal contaminated with additive Gaussian noise $\mathbf{x} \in \mathbb{R}^{512}$. The approximated signal is represented as ${\mathbf{\Phi}_{\mathsf{FG}}}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}\ \mathbf{x}$, where the matrix $\mathbf{\Phi}_{\mathsf{FG}} \in \mathbb{R}^{12\times 512}$ is built using the clustering of the intensities with a spatial constraint (i.e. spatially-constrained Ward clustering). Only 12 clusters can preserve the structure of the signal and decrease the noise, as seen from the signal-to-noise-ratio (dB).

Feature grouping defines a matrix $\mathbf{\Phi}$ that extracts piece-wise constant approximations of the data [22]. Let $\mathbf{\Phi}_{\mathsf{FG}}$ be a matrix composed with constant amplitude groups (clusters). Formally, the set of k clusters is given by $\mathcal{P} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$, where each cluster $\mathcal{C}_q \subset [p]$ contains a set of indexes that does not overlap other clusters, $\mathcal{C}_q \cap \mathcal{C}_l = \emptyset$, for all $q \neq l$. Thus, $(\mathbf{\Phi}_{\mathsf{FG}}\ \mathbf{x})_q = \alpha_q \sum_{j\in\mathcal{C}_q} \mathbf{x}_j$ yields a reduction of a data sample $\mathbf{x}$ on the q-th cluster, where α_q is a constant for each cluster. With an appropriate permutation of the indexes of the data $\mathbf{x}$, the matrix $\mathbf{\Phi}_{\mathsf{FG}}$ can be written as

$$\mathbf{\Phi}_{\mathsf{FG}} = \begin{bmatrix} \alpha_1\text{—}\alpha_1 & 0\text{—}0 & \ldots & 0\text{—}0 \\ 0\text{—}0 & \alpha_2\text{—}\alpha_2 & \ldots & 0\text{—}0 \\ \vdots & \vdots & \ddots & \vdots \\ 0\text{—}0 & 0\text{—}0 & \ldots & \alpha_k\text{—}\alpha_k \end{bmatrix} \in \mathbb{R}^{k\times p}.$$

We choose $\alpha_q = 1/\sqrt{|\mathcal{C}_q|}$ to set the non-zero singular values of $\mathbf{\Phi}_{\mathsf{FG}}$ to 1, making it an orthogonal projection.

We call $\mathbf{\Phi}_{\mathsf{FG}}\ \mathbf{x} \in \mathbb{R}^k$ the *reduced* version of $\mathbf{x}$ and ${\mathbf{\Phi}_{\mathsf{FG}}}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}\ \mathbf{x} \in \mathbb{R}^p$ the *approximation* of $\mathbf{x}$. Note that having an approximation of the data means that the ensuing inference steps can be made explicit in the original space. As the matrix $\mathbf{\Phi}_{\mathsf{FG}}$ is sparse, this approximation follows the same principle as [84], speeding up computational time and reducing memory storage.

Let $M(\mathbf{x})$ be the approximation error for a data $\mathbf{x}$ given a feature grouping matrix $\mathbf{\Phi}_{\mathsf{FG}}$,

$$M(\mathbf{x}) = \left\| \mathbf{x} - \mathbf{\Phi}_{\mathsf{FG}}{}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x} \right\|_2^2, \tag{4.1}$$

this is often called inertia in the clustering literature. This corresponds to the sum of all the local errors (the approximation error for each cluster), $M(\mathbf{x}) = \sum_{q=1}^{k} m_q$, where $m_q(\mathbf{x})$ is the sum of squared differences between the values in the q-th cluster and its representative center, as follows

$$m_q(\mathbf{x}) = \left\| \mathbf{x}_{\mathcal{C}_q} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q}{\sqrt{|\mathcal{C}_q|}} \right\|_2^2, \tag{4.2}$$

where $\mathbf{x}_{\mathcal{C}_q}$ are the values $\mathbf{x}_i$ such that $i \in \mathcal{C}_q$. The squared norm of the data $\mathbf{x}$ is then decomposed in two terms: fidelity and inertia, taking the form (see section 2 in supplementary materials):

$$\|\mathbf{x}\|_2^2 = \underbrace{\|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|_2^2}_{\text{Reduced norm}} + \underbrace{\sum_{q=1}^{k} \left\| \mathbf{x}_{\mathcal{C}_q} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q}{\sqrt{|\mathcal{C}_q|}} \right\|_2^2}_{M(\mathbf{x})\text{: Inertia}}. \tag{4.3}$$

Eq. 4.3 is key to understanding the desired properties of a matrix $\mathbf{\Phi}_{\mathsf{FG}}$. In particular, it shows that it is beneficial to work in a large k regime to reduce the inertia.

Corollary 4.2.1. *Let $\mathbf{x} \in \mathbb{R}^p$ be a signal, and $\mathbf{\Phi}_{\mathsf{FG}}$ be a feature-grouping matrix, the following holds*

$$\|\mathbf{x}\|_2^2 - \sum_{q=1}^{k} \left\| \mathbf{x}_{\mathcal{C}_q} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q}{\sqrt{|\mathcal{C}_q|}} \right\|_2^2 = \|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|_2^2 \leq \|\mathbf{x}\|_2^2. \tag{4.4}$$

Proof. As $\mathbf{\Phi}_{\mathsf{FG}}{}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}$ is an orthogonal operator, Eq. 4.3 corresponds to an orthogonal decomposition.

We start by writing down the ℓ_2 norm of the data vector $\mathbf{x}$ for every point inside all the clusters $\{\mathcal{C}_q\}_{q=1}^{k}$. Then, we perform simple manipulations, as follows

$$\begin{aligned}
\|\mathbf{x}\|_2^2 &= \sum_{q=1}^{k} \sum_{i \in \mathcal{C}_q} \mathbf{x}_i^2 \\
&= \sum_{q=1}^{k} \sum_{i \in \mathcal{C}_q} \left(\mathbf{x}_i^2 + \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q^2}{|\mathcal{C}_q|} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q^2}{|\mathcal{C}_q|} \right) \\
&= \sum_{q=1}^{k} (\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q^2 + \sum_{q=1}^{k} \sum_{i \in \mathcal{C}_q} \left(\mathbf{x}_i^2 - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q^2}{|\mathcal{C}_q|} \right) \\
&= \|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|_2^2 + \sum_{q=1}^{k} \left\| \mathbf{x}_{\mathcal{C}_q} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q}{\sqrt{|\mathcal{C}_q|}} \right\|_2^2.
\end{aligned} \tag{4.5}$$

Finally, the right hand inequality of Eq. 4.4 comes naturally after scaling each cluster (the non-zero singular values are set to 1). □

Capturing signal structure

We consider data with a specific structure, e.g. spatial data. Well-suited dimensionality reduction can leverage this structure to bound the approximation error. We assume that the data $\mathbf{x} \in \mathbb{R}^p$ are generated from a process acting on a space with a neighborhood structure (topology). To encode this structure, the data matrix $\mathbf{X}$ is associated with an undirected graph $\mathcal{G}$ with p vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_p\}$. Each vertex of the graph corresponds to an index in the data matrix $\mathbf{X}$ and the presence of an edge means that these features are connected. For instance, for 2D or 3D image data, the graph is a 2D or 3D lattice connecting neighboring pixels. The graph defines a graph distance between features $\text{dist}_{\mathcal{G}}$. In practice, we perform the calculations with the adjacency matrix $\mathbf{G}$ of the graph $\mathcal{G}$.

Definition 4.2.1. *L-Smoothness of the signal: A signal $\mathbf{x} \in \mathbb{R}^p$ structured by a graph $\mathcal{G}$, is pairwise Lipschitz smooth with parameter L when it satisfies*

$$|\mathbf{x}_i - \mathbf{x}_j| \leq L\, dist_{\mathcal{G}}(v_i, v_j), \quad \forall (i,j) \in [p]^2. \tag{4.6}$$

This definition means that the signal is smooth with respect to the graph that encodes the underlying structure. Note that $dist_{\mathcal{G}}$ has no unit since the scale is fixed by having each edge have length 1.

Lemma 4.2.1. *Let $\mathbf{x} \in \mathbb{R}^p$ be a pairwise L-Lipschitz signal, and $\mathbf{\Phi}_{\mathsf{FG}} \in \mathbb{R}^{k \times p}$ be a fixed feature grouping matrix, formed by $\{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$ clusters. Then the following holds:*

$$\|\mathbf{x}\|^2 - L^2 \sum_{q=1}^{k} |\mathcal{C}_q|\, diam_{\mathcal{G}}(\mathcal{C}_q)^2 \leq \|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|^2 \leq \|\mathbf{x}\|^2, \tag{4.7}$$

where $diam_{\mathcal{G}}(\mathcal{C}_q) = \sup_{v_i, v_j \in \mathcal{C}_q} dist_{\mathcal{G}}(v_i, v_j)$.

We see that the approximation is better if: *i)* the cluster sizes are about the same, and *ii)* the clusters have a small diameter. These arguments are based only on the assumption of smoothness of the signal. Refining Eq. 4.7 gives an intuition on how a partition could be adapted to the data: We can see that the approximation is better if: *i)* the signal in a cluster is homogeneous; *ii)* clusters in irregular areas are smaller.

Proof. The corollary 4.2.1 shows that the lower bound of the representation is only affected by the inertia (see left hand side of Eq. 4.4). So, the worst case corresponds to the upper bound of the inertia $M(\mathbf{x}) = \sum_{q=1}^{k} m_q(\mathbf{x})$. Then,

the inertia of each cluster can be bounded as follows

$$\begin{aligned} m_q(\mathbf{x}) &= \left\| \mathbf{x}_{\mathcal{C}_q} - \frac{(\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x})_q}{\sqrt{|\mathcal{C}_q|}} \right\|_2^2 \\ &= \sum_{j \in \mathcal{C}_q} \left(\mathbf{x}_j - \frac{1}{|\mathcal{C}_q|} \sum_{i \in \mathcal{C}_q} \mathbf{x}_i \right)^2 \\ &\leq |\mathcal{C}_q| \max_{j \in \mathcal{C}_q} \left| \mathbf{x}_j - \frac{1}{|\mathcal{C}_q|} \sum_{i \in \mathcal{C}_q} \mathbf{x}_i \right|^2 \end{aligned} \tag{4.8}$$

The last inequality corresponds to the worst case for the sum inside the cluster.

Let us constrain our analysis to L-smooth signals $\mathbf{x} \in \mathbb{R}^p$ structured by graph $\mathcal{G}$ (see Definition 2.1). Under this assumption, we can bound the inertia of each cluster as follows:

$$\begin{aligned} m_q(\mathbf{x}) &\leq |\mathcal{C}_q| \max_{j \in \mathcal{C}_q} \left| \mathbf{x}_j - \frac{1}{|\mathcal{C}_q|} \sum_{i \in \mathcal{C}_q} \mathbf{x}_i \right|^2 \\ &\leq |\mathcal{C}_q|\, L^2 \sup_{i,j \in \mathcal{C}_q} \mathrm{dist}_{\mathcal{G}}(v_i, v_j)^2 \\ &= L^2\, |\mathcal{C}_q|\, \mathrm{diam}_{\mathcal{G}}(\mathcal{C}_q)^2, \end{aligned} \tag{4.9}$$

where the second inequality follows from the pairwise L-Lipschitz condition. Finally, plugging Eq. 4.9 into Eq. 4.5 we have:

$$\|\mathbf{x}\|^2 - L^2 \sum_{q=1}^{k} |\mathcal{C}_q|\, \mathrm{diam}_{\mathcal{G}}(\mathcal{C}_q)^2 \leq \|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|^2 \leq \|\mathbf{x}\|_2^2. \tag{4.10}$$

□

Corollary 4.2.2. *Let L_q be the smoothness index inside cluster C_q, for all $q \in [k]$. This is the minimum L_q such that:*

$$|\mathbf{x}_i - \mathbf{x}_j| \leq L_q\, dist_{\mathcal{G}}(v_i, v_j), \quad \forall (i,j) \in C_q^2.$$

Then the following two inequalities hold:

$$\begin{aligned} \|\mathbf{x}\|_2^2 - \sum_{q=1}^{k} |\mathcal{C}_q| \sup_{\mathbf{x}_i, \mathbf{x}_j \in \mathbf{x}_{\mathcal{C}_q}} |\mathbf{x}_i - \mathbf{x}_j|_2^2 \leq \\ \|\mathbf{x}\|_2^2 - \sum_{q=1}^{k} L_q^2\, |\mathcal{C}_q|\, diam_{\mathcal{G}}(\mathcal{C}_q)^2 \leq \|\mathbf{\Phi}_{\mathsf{FG}}\, \mathbf{x}\|_2^2. \end{aligned} \tag{4.11}$$

Proof. As in Eq. 4.9, the second inequality is consequence of adding the local L-smoothness condition. □

Approximating signals with unstructured noise

In this section, we analyze the regularity condition of the signal of interest $\mathbf{s}$, its relation with the noise and maximum cluster size. Let $\mathbf{x}$ be the acquired signal, which is a fixed signal of interest $\mathbf{s}$ contaminated with an i.i.d. zero-mean Gaussian noise $\mathbf{n}$ with variance σ^2, $\mathbf{x} = \mathbf{s} + \mathbf{n}$.

Proposition 4.2.1. *Let $\mathbf{x} = \mathbf{s} + \mathbf{n} \in \mathbb{R}^p$ be an acquired signal, where $\mathbf{s}$ is a fixed smooth L-Lipschitz signal and $\mathbf{n}$ an i.i.d. zero-mean Gaussian noise with variance σ^2. Then, for a given grouping matrix $\mathbf{\Phi}_{\mathsf{FG}} \in \mathbb{R}^{k \times p}$ the mean squared error of the approximation (MSE_{approx}) is upper-bounded by*

$$MSE_{approx} \leq L^2 \sum_{q=1}^{k} |\mathcal{C}_q| \; diam_{\mathcal{G}}(\mathcal{C}_q)^2 + \frac{k}{p} MSE_{orig}. \tag{4.12}$$

Proof. With the purpose of ensuring clarity, we define $\mathbf{A} = \mathbf{\Phi}_{\mathsf{FG}}{}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}$. Let $\mathrm{MSE}_{\mathrm{appox}} = \mathbb{E}_{\mathbf{n}} \left[\|\mathbf{s} - \mathbf{A}\,\mathbf{x}\|_2^2 \right]$ and $\mathrm{MSE}_{\mathrm{orig}} = \mathbb{E}_{\mathbf{n}} \left[\|\mathbf{n}\|_2^2 \right]$ be the mean squared error with and without approximation, receptively. As we are dealing with Gaussian noise, the risk of the raw data is $\mathrm{MSE}_{\mathrm{orig}} = p\,\sigma^2$. Given that $\|\mathbf{s}\|_2^2$ is fixed, it is enough to show $\mathrm{MSE}_{\mathrm{approx}} \leq \mathrm{MSE}_{\mathrm{orig}}$ to ensure an increase in the SNR.

We start by writing down the $\mathrm{MSE}_{\mathrm{approx}}$, then we separate the components thanks to the i.i.d assumption and plug the upper-bound of the inertia, as follows

$$\begin{aligned}
\mathrm{MSE}_{\mathrm{approx}} &= \mathbb{E}_{\mathbf{n}} \left[\|\mathbf{s} - \mathbf{A}\,\mathbf{x}\|_2^2 \right] \\
&= \|(\mathbf{I} - \mathbf{A})\,\mathbf{s}\|_2^2 + \mathbb{E}_{\mathbf{n}} \left[\|\mathbf{A}\,\mathbf{n}\|_2^2 \right] \\
&= \|(\mathbf{I} - \mathbf{A})\,\mathbf{s}\|_2^2 + k\,\sigma^2 \\
&\leq L^2 \sum_{q=1}^{k} |\mathcal{C}_q| \,\mathrm{diam}_{\mathcal{G}}(\mathcal{C}_q)^2 + k\,\sigma^2.
\end{aligned}$$

□

Corollary 4.2.3. *Let $\mathbf{x} = \mathbf{s} + \mathbf{n} \in \mathbb{R}^p$ be an acquired signal, where $\mathbf{s}$ is a fixed a pairwise smooth L-Lipschitz signal and $\mathbf{n}$ is an i.i.d. zero-mean Gaussian noise with variance σ^2. For a given grouping matrix $\mathbf{\Phi}_{\mathsf{FG}} \in \mathbb{R}^{k \times p}$, the noise after approximation is reduced, $MSE_{approx} \leq MSE_{orig}$, only if the L^2 smoothness parameter satisfy*

$$L^2 \leq \frac{(p-k)}{\sum_{q=1}^{k} |\mathcal{C}_q| \; diam_{\mathcal{G}}(\mathcal{C}_q)^2} \sigma^2. \tag{4.13}$$

Proof. This is a direct result of the proposition 4.2.1 after some arithmetic

manipulations,

$$\begin{aligned}\mathrm{MSE}_{\mathrm{approx}} &\leq L^2 \sum_{q=1}^{k} |\mathcal{C}_q| \operatorname{diam}_{\mathcal{G}}(\mathcal{C}_q)^2 + p\,\sigma^2 - (p-k)\,\sigma^2 \\ &\leq L^2 \sum_{q=1}^{k} |\mathcal{C}_q| \operatorname{diam}_{\mathcal{G}}(\mathcal{C}_q)^2 + \mathrm{MSE}_{\mathrm{orig}} - (p-k)\,\sigma^2.\end{aligned}$$

Then, to satisfy $\mathrm{MSE}_{\mathrm{approx}} \leq \mathrm{MSE}_{\mathrm{orig}}$, we must have:

$$L^2 \sum_{q=1}^{k} |\mathcal{C}_q| \operatorname{diam}_{\mathcal{G}}(\mathcal{C}_q)^2 \leq (p-k)\,\sigma^2,$$

which lead us to the upper-bound of the Lipschitz constant. □

Cluster of the same size: This is a particular case, where we assume that the clusters $\mathcal{P} = \{\mathcal{C}_q\}_{q=1}^{k}$ have the same size, $\frac{p}{k}$. Under this assumption, the following holds:

$$\mathrm{MSE}_{\mathrm{approx}} \leq p \left(\frac{L}{k}\right)^2 + \frac{k}{p}\,\mathrm{MSE}_{\mathrm{orig}} = O\left(\max\left\{\frac{p}{k^2}, \frac{k}{p}\right\}\right). \tag{4.14}$$

We need to balance both therms in the right-hand side of 4.14 in order to maximize the rate of decay. This implies that $\frac{p}{k^2} = \frac{k}{p}$ therefore $k = p^{2/3}$ and $\mathrm{MSE}_{\mathrm{approx}} = O(k^{-1/2})$.

4.3 Conclusion – signal approximation by feature grouping

We presented a theoretical analysis of the approximation of the signal using feature grouping. We showed that feature grouping can preserve well the pairwise Euclidean distances between structured signals. This property makes it well suited for ℓ_2-based algorithms –see chapter 3–, like shift-invariant kernel-based methods, or to approximate queries in information-retrieval settings.

In addition, we introduced a sufficient regularity condition to show when this scheme leads to a reduction of the unstructured noise. This noise attenuation can yield a beneficial improvement on the performance of subsequent analysis.

An important aspect of feature grouping compared to other fast dimension reductions, such as random projections, is that the features of the reduced representation it creates of the data make sense for the application. Consequently, the dimension reduction step can be inverted, and any statistical analysis performed after reduction can be reported with regard to the original signal.

5 Recursive nearest agglomeration - ReNA: A fast structured clustering algorithm

In the previous chapter, we introduced and analyzed *feature grouping* as a dimension reduction technique to approximate structured signals –e.g. images.

Here, we propose a data-driven approach to build the feature agglomeration matrix. In particular, via clustering of features. Nevertheless, there are some algorithmic challenges, as an impediment to fast dimension reduction is that good clustering comes with large algorithmic costs; the main issue with fast algorithms: they create huge clusters, this impair the approximation of the signal.

In this chapter, we address these drawbacks by contributing a linear-time agglomerative clustering scheme, Recursive Nearest Agglomeration (ReNA). Unlike existing fast agglomerative schemes, it avoids the creation of giant clusters. This algorithm relies on a recursive extraction of the connected components of a 1-Nearest-Neighbor graph, reducing the graph at each iteration until the desired number of clusters is reached.

5.1 Introduction – data-aware feature grouping

Spatially and information-aware compression schemes are probably better suited for structured signals –e.g. images– [3]. We propose here using clustering procedures to find a suitable feature partition to group features, respecting the outline of signals' structures. But, as we discussed in chapter 3.4, the approximation of the signal improves quadratically on the number of clusters. Therefore, a good approximation still requires many clusters. In this setting, a standard clustering method as K-means yields computationally expensive estimation procedures. On the contrary, agglomerative clustering algorithms are amongst the fastest approaches to extract many clusters with graph-connectivity constraints. However, they fail to create clusters of evenly-distributed size, favoring a few huge clusters[1].

[1] This phenomenon is known as percolation in random graphs [150].

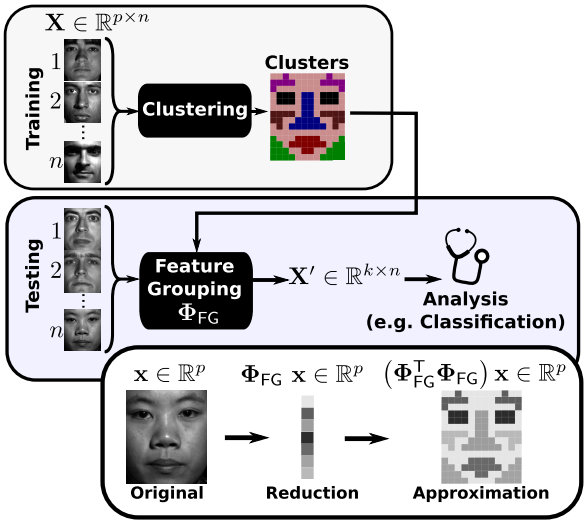


Figure 5.1: **Feature grouping for statistical analysis of structured images:** Illustration of the different steps of feature grouping-based data approximation . The approach consists in finding a data-driven spatial reduction $\mathbf{\Phi}_{\mathsf{FG}}$ using clustering. Then, the data $\mathbf{x}$ are reduced to $\mathbf{\Phi}_{\mathsf{FG}}\,\mathbf{x}$ and then used for further statistical analysis (e.g. classification).

Finding a feature grouping matrix: We now consider a data-driven approach to build the matrix $\mathbf{\Phi}_{\mathsf{FG}}$. We rely on feature clustering: a clustering algorithm is used to define the groups of features from the data. $\mathbf{X} \in \mathbb{R}^{p \times n}$ is represented by a reduced version $\mathbf{\Phi}_{\mathsf{FG}}\,\mathbf{X}$, where p is potentially very large (greater than 100 000), whereas k is smaller but close enough to p (e.g. $k = \lfloor p/20 \rfloor$). As illustrated in Fig. 5.1, once the reduction operator $\mathbf{\Phi}_{\mathsf{FG}}$ has been learned, it can be applied to new data.

5.2 Existing fast clustering algorithms

K-means clustering is a natural choice as it minimizes the total inertia in Eq. 4.3. But it tends to be expensive in our setting: The conventional k-means algorithm has a complexity of $O(n\,k)$ per iteration [3]. However, the larger the number of clusters, the more iterations are needed to converge, and the worst case complexity is given[2] by $O(p^{k+2/n})$ [10]. This complexity becomes prohibitive with many clusters.

[2] Note that here n and p are swapped compared to common clustering literature, as we are doing *feature* clustering.

Super-pixel approaches: In computer vision, feature clustering can be related to the notion of *super-pixels* (super-voxels for 3D images). The most common fast algorithm for super-pixels is SLIC [3], which has a low computational cost and produces super-pixels/super-voxels of roughly even sizes. SLIC performs a local clustering of the image values with a spatial constrain, using as a distance measure the combination of two Euclidean distances: image values and spatial positions. The SLIC algorithm is related to K-means, but it performs a fixed small number of iterations, resulting in a complexity of $O(np)$. Its main drawback is that, in the large-k regime, it can be difficult to control precisely the number of clusters, as some clusters often end up empty in the final assignment.

Agglomerative clustering: Agglomerative clustering algorithms are fast in the setting of a large number k of clusters. Unlike most clustering algorithms, adding a graph structure constraint makes them even faster, as they can then discard association between non-connected nodes.

Agglomerative clustering schemes start off by placing every data element in its own cluster, then they proceed by merging repeatedly the closest pair of clusters until one obtain the desired number of clusters [65]. Various methods share the same approach, differing only in the linkage criterion used to identify the clusters to be merged. The most common linkages are single, average, complete [65] and Ward [168]. Average-linkage, complete-linkage, and Ward are generally preferable over single-linkage, as they tend to yield more balanced clusters. Yet single-linkage clustering is often used as it is markedly faster; it can be obtained via a Minimum Spanning Tree and has a complexity of $O(np + p \log p)$ [110]. Average-linkage, complete-linkage and Ward have a worse case complexity of $O(np^2)$ [110].

The approximation properties of feature grouping are given by the distribution of cluster sizes (Eq. 4.3). Balanced clusters are preferable for low errors. Nevertheless, agglomerative clustering on noisy data can often lead to a "preferential attachment" behavior, where large clusters grow faster than smaller ones. In this case, the largest cluster dominates the distortion, as in Lemma. 4.2.1. By considering that the clusters are connected components on a similarity graph, this behavior can be linked to percolation theory [150], that characterizes the appearance of a giant connected component (i.e. a huge cluster). In this case, the clustering algorithm is said to *percolate*, and thus cannot yield balanced cluster sizes.

In brief, single-linkage clustering is fast but suffers form percolation issues [125] and Ward's algorithm performs often well in terms of goodness of fit for large k [152].

More sophisticated agglomerative strategies have been proposed in the framework of computer vision (e.g. [49]), but they have not been designed to avoid percolation and do not make it possible to control the number k of clusters.

5.3 Contributed clustering algorithm – ReNA

Preliminaries: neighbors graphs

For feature clustering on structured signals, an algorithm should take advantage of the generative nature of the data, e.g. for images, work with local image statistics. Hence we rely on neighborhood graphs [45].

Neighborhood graphs form an important class of geometric graphs with many applications in signal processing, pattern recognition, or data clustering. They are used to model local relationships between data points, with ϵ-neighborhood graphs[3] or k-nearest neighbor graphs.

[3] ϵ-neighborhood graph: v_i and v_j are connected if $\|v_i - v_j\| \leq \epsilon$.

The ϵ-nearest neighbor graph is the core of one of the most popular scalable clustering algorithms, DBSCAN [47]. It is a density-based clustering method for which the number k of clusters is implicitly set by the ϵ neighborhood's radius. Its main drawback is its high sensitivity to the choice of this very parameter. K-nearest neighbor graphs, on the other hand, are not well suited for clustering as they tend to percolate for k greater than or equal to 2. In contrast, the 1-nearest neighbor graph (1-NN) is not likely to percolate[151]. For this reason, we use the 1-NN graph.

ReNA: algorithm outline

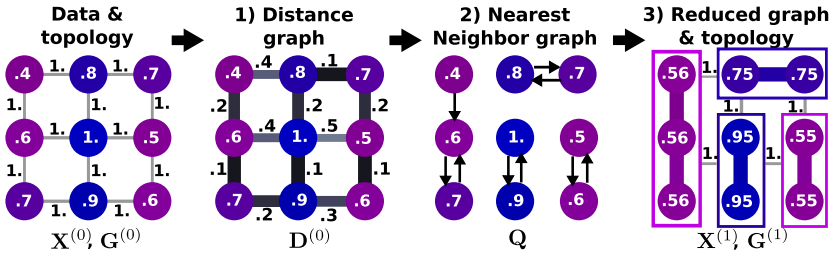


Figure 5.2: **The nearest neighbor grouping:** The algorithm receives a data matrix $\mathbf{X}$ represented on a regular square lattice $\mathbf{G}$. *left)* The nodes correspond to the feature values and the edges are the encoded topological structure. *1) Graph representation:* We calculate the similarity matrix $\mathbf{D}$. *2) Finding 1-NN:* We proceed by finding the 1-nearest neighbors subgraph $\mathbf{Q}$ according to the similarity measure. *3) Getting the clusters and reduction step:* We extract the connected components of $\mathbf{Q}$ and merge the connected nodes.

In a nutshell, our algorithm relies on extracting the connect components of a 1-NN graph. To reach the desired number k of clusters, we apply it recursively. The algorithm outline is as follows:

Initialization: We start by placing each of the p features of the data $\mathbf{X}$ in its own cluster $\mathcal{P} = \{\mathcal{C}_1, \ldots, \mathcal{C}_p\}$. We use the binary adjacency matrix $\mathbf{G} \in \mathbb{R}^{p \times p}$ of the graph $\mathcal{G}$, that encodes the topological structure of the features.

Nearest neighbor grouping: We use the nearest neighbor of a similarity graph as linkage criterion. We then extract the connected components of this subgraph to reduce the data matrix $\mathbf{X}$ and the topological structure $\mathbf{G}$. These operations are summarized in the next steps:

1. **Graph representation:** We build the similarity graph $\mathcal{D}$ of the data $\mathbf{X}$, represented by the adjacency matrix $\mathbf{D} \in \mathbb{R}^{p \times p}$. The weights are constrained by $\mathbf{G}$[4].
2. **Finding 1-NN:** Creating a 1-nearest neighbor graph $\mathcal{Q}$, represented by the adjacency matrix $\mathbf{Q} \in \mathbb{R}^{p \times p}$, where each vertex of $\mathcal{D}$ is associated with its nearest neighbor in the sense of the dissimilarity measure.
3. **Getting the clusters:** We use [121] to extract the set of connected components of $\mathbf{Q}$ and assign them to the new set of clusters $\mathcal{P}$.
4. **Reduction step:** The clusters are used to reduce the graph $\mathbf{G}$ and the data $\mathbf{X}$.

[4] This corresponds to an element-wise condition, where a similarity weight is assigned only if the edges are connected according to $\mathbf{G}$.

Stopping condition: Nearest neighbor grouping can be performed repeatedly on the reduced versions of the graph $\mathbf{G}$ and the data $\mathbf{X}$ until the desired number k of clusters is reached[5].

[5] In practice the size of the matrices $\mathbf{X}^{(t)}, \mathbf{D}^{(t)}, \mathbf{G}^{(t)}$ decreases during the iterations. Therefore it is necessary to express the partition $\mathcal{P}$ on a reduced index set. To simplify notations, we have not detailed this operation in the algorithm.

Fig. 5.2 presents one iteration of the nearest neighbor grouping on a regular square lattice. The pseudo-code of ReNA is given in algorithm 2 and an illustration on a 2D brain image in Fig. 5.3.

The algorithm is iterated until the desired number of clusters k is reached[6]. As the number of vertices is divided by 2 at each step, the number of iterations is at most $O\{\log(p/k)\}$; in practice, we never have to go beyond 5 iterations. The cost of computing similarities is linear in n and, as all the operations involved are also linear in the number of vertices p, the total procedure is $O(np)$.

Algorithm 2 Recursive nearest neighbor (ReNA) clustering

Require: Data $\mathbf{X} \in \mathbb{R}^{p \times n}$, sparse matrix $\mathbf{G} \in \mathbb{R}^{p \times p}$ representing the associated connectivity graph structure, nearest-neighbor subgraph extraction function NN, connected components extraction function ConnectComp [121], desired number k of clusters.

Ensure: Clustering of the features $\mathcal{P} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$

$q = p$ {Initializing the number of clusters to p}
$t = 0$
$\mathbf{X}^{(t)} = \mathbf{X}$
$\mathbf{G}^{(t)} = \mathbf{G}$
while $q > k$ **do**
  $\mathbf{D}^{(t)}_{i,j} \leftarrow \mathbf{G}^{(t)}_{i,j} \|\mathbf{X}^{(t)}_{i,*} - \mathbf{X}^{(t)}_{j,*}\|_2^2,\ (i,j) \in [q]^2$
  {Create a similarity weighted graph.}
  $\mathbf{Q} \leftarrow \mathrm{NN}(\mathbf{D}^{(t)})$ {1-nearest neighbor graph.}
  $\mathcal{P} \leftarrow \mathrm{ConnectComp}(\mathbf{Q})$, [6]
  {Sets of connected components of 1-nearest neighbor graph.}
  $\mathbf{U}_{i,j} \leftarrow \begin{cases} 1 & \text{if } i \in \mathcal{C}_j \\ 0 & \text{otherwise} \end{cases}, (i,j) \in [q] \times [|\mathcal{P}|]$
  {Assignment matrix}
  $\mathbf{X}^{(t+1)} \leftarrow (\mathbf{U}^\top \mathbf{U})^{-1} \mathbf{U}^\top \mathbf{X}^{(t)},\ \mathbf{X}^{(t+1)} \in \mathbb{R}^{|\mathcal{P}| \times n}$
  {Reduced data matrix. Note that the computation boils down to sample averages}
  $\mathbf{G}^{(t+1)} \leftarrow \mathrm{support}(\mathbf{U}^\top \mathbf{G}^{(t)} \mathbf{U}),\ \mathbf{G}^{(t+1)} \in \mathbb{R}^{|\mathcal{P}| \times |\mathcal{P}|}$
  {Reduced between-cluster topological model; the non-zero values are then replaced by ones.}
  $q = |\mathcal{P}|$ {Update the number of clusters}
  $t = t + 1$
end while
return $\mathcal{P}$

[6] At each iteration, a connected components routine extracts them from $\mathbf{Q}$ and returns them as a set of clusters $\mathcal{P}$. In the last iteration of the algorithm, if there are less than k connected components, $\mathbf{Q}$ is pruned of its edges with smallest edge values to keep only the $q - k$ shortest edges, so that no less than k components are formed.

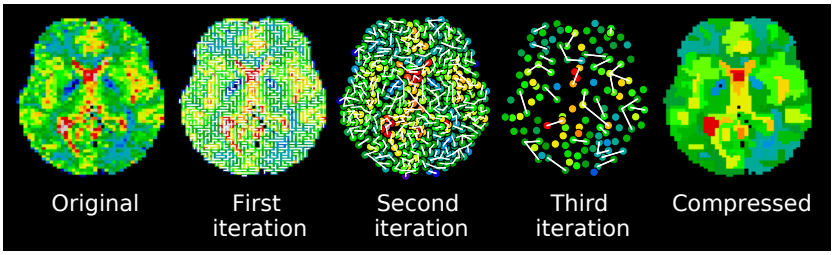


Figure 5.3: **Illustration of the working principle of the Recursive Nearest Neighbor, ReNA:** The white lines represent the edges of the graph. The algorithm receives the original image, considering each feature (i.e. pixel or voxel in the image) as a cluster. From now on, for each iteration, the nearest clusters are merged, yielding a reduced graph, until the desired number of clusters is found.

5.4 Conclusion – ReNA, a non-percolating fast clustering method

We proposed a linear-time graph-structured clustering algorithm, ReNA, that is efficient with many clusters. This algorithm iteratively performs 1-nearest neighbor grouping, reduces the graph at each iteration, then averages the input features and repeats the process until it reaches the desired number of clusters. Hence, this clustering strategy takes the underlying regularity in the observed signal into account; and it will likely find balanced cluster sizes as the 1-nearest neighbor graph does not percolate in general.

6 Validating the proposed clustering algorithm – ReNA

In the previous chapters, we introduced and analyzed *feature grouping* as a dimension reduction technique for structured signals – e.g. brain images. We also proposed a new clustering algorithm –see chapter 5–, *ReNA*, that finds balanced cluster sizes in linear-time.

In this chapter, we empirically validate that *ReNA* approximates the data as well as traditional variance-minimizing clustering schemes that have a quadratic complexity. As a consequence, data reduction by this algorithm is very beneficial for analysis of large-scale structured datasets, as the dimension reduction is very fast, and it reduces the computational cost of various estimators without losing accuracy.

In addition, our theoretical analysis –see chapter 3.4– is backed by extensive experiments on publicly-available data that illustrate the computation efficiency and the denoising properties of the resulting dimension reduction schemes.

6.1 Experimental Study

In this chapter, we conduct a series of experiments to assess the quality of the dimensionality reduction scheme and its viability as a preprocessing step for several statistical analyses. Table. 6.1 gives a summary of the datasets used.

We investigate the performance of feature grouping with a variety of clustering algorithms: single-linkage, average-linkage, complete-linkage, Ward, SLIC, and ReNA. We compare them with other fast dimensionality reductions: random projections, random sampling, as well as image downsampling. We measure their ability to represent the data and characterize their percolation behavior when it is relevant. To evaluate their denoising properties, we use them in prediction tasks. We study not only ℓ_2 methods, but also methods relying on higher moments of the data distribution: ℓ_1 penalization and independent component analysis (ICA).

To present the results, we use the *fraction of the signal*, which is defined as the ratio between the number k of components and the greatest possible value of k. Here, we have two cases: *i)* for random projections and feature grouping methods, the maximum value of k corresponds to the number p of

features, $k/p \times 100\%$; *ii)* for random sampling, the maximum value of k is the number of samples, $k/n \times 100\%$. Downsampling the images with linear interpolation can be seen as using data-independent clusters, all of the same size.

datasets

Dataset	Description	n	p	Task
Synthetic	Cube	10	$\{8, 16, 64, 128\}^3$	Time complexity
		1 000	240 000	Distortion
Faces [173]	Grayscale face images	2 414	32 256	Recognition of 38 subjects
OASIS [98]	Anatomical brain images	403	140 398	Gender discrimination
				Age prediction
HCP [46]	Functional brain images	8 294	254 000	Predicting 17 cognitive tasks
				Spatial ICA

Table 6.1: Summary of the datasets and the tasks performed with them.

Synthetic data: We generate a synthetic data set composed of 1 000 3D images with and without noise. Each one is a cube of $p = 50^3$ voxels containing a spatially smooth random signal (FWHM=8 voxels), which is our signal of interest $\mathbf{X}$. The acquired signal $\mathbf{S}$ is our signal of interest contaminated by zero-mean additive white Gaussian noise, with a Signal-to-Noise Ratio (SNR) of $2.06dB$.

The extended Yale B face recognition dataset[173]: This dataset was designed to study illumination effects on face recognition[173] and consists of $n = 2\,414$ images of 38 identified individuals under 64 lighting conditions. Each image was converted to grayscale, cropped, and normalized to 192×168, leaving $p = 32\,256$ features. For the face recognition task, there are 38 classes, one per subject.

The Open Access Series of Imaging Studies (OASIS)[98]: This dataset is described in section 1.5. We perform two prediction tasks with this dataset: i) Gender classification and ii) age regression.

Human Connectome Project (HCP)[46]: We consider a functional Magnetic Resonance Imaging acquired in the HCP –see the description in section 1.5.

Task-related data: The tasks relate to different cognitive labels on working memory/cognitive control processing. With these data, we perform across-subject discrimination of 17 experimental conditions selected from the aforementioned task-related datasets.

Resting-state data: We use the two resting-state sessions from 93 subjects. Each session represents about 1GB of data, with $p \approx 220\,000$ and $n = 1200$, totaling 200 GB of dense data for all subjects and sessions. With this data, we perform a spatial ICA.

Implementation aspects: We use scikit-learn for machine-learning tasks (logistic and Ridge regression), fast-ICA, clustering and sparse random projections [123]. We rely on scikit-image for SLIC[157], and on Nilearn[2] to handle neuroimaging data. Code for ReNA and experiments is available online[1].

[1] https://github.com/ahoyosid/ReNA

6.2 Quality assessment experiments

We perform experiments to measure distortion properties and computation times of the dimensionality reduction methods.

Empirical computational complexity

We empirically assess the scalability of the algorithms as function of the input size. The test is carried out for a synthetic dataset composed of 10 cubes. We varied their dimension $p \in \{8, 16, 64, 128\}^3$ and fix the number of clusters to $k = \lfloor \frac{p}{20} \rfloor$. We repeat 10 times, and report the average computation time.

Fig. 6.1 reports the computation time to estimate $\mathbf{\Phi}$ for the different approximation schemes. It shows that Nyström, single-linkage, complete-linkage, SLIC and ReNA have a behavior linear in the number p of features. Random projections, average-linkage and Ward display a sub-quadratic time complexity, whereas downsampling has a sub-linear behavior. Single-linkage and ReNA outperform other agglomerative methods, reducing the computation time by a factor of 10. Profiling random projections reveals that its run time is dominated by the random number generation. The scikit-learn implementation uses a Mersenne Twister algorithm, with good entropic properties to the cost of increased computations [78].

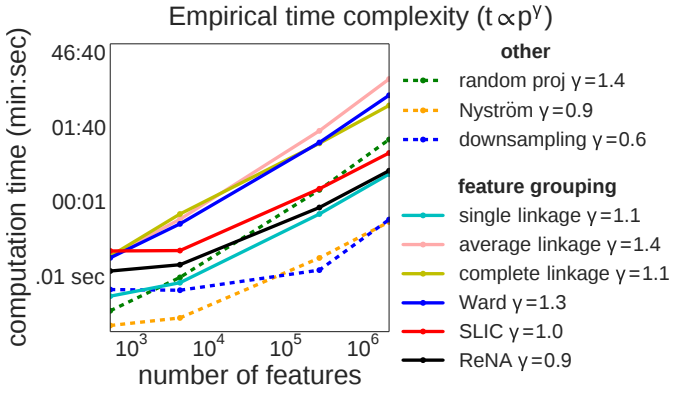
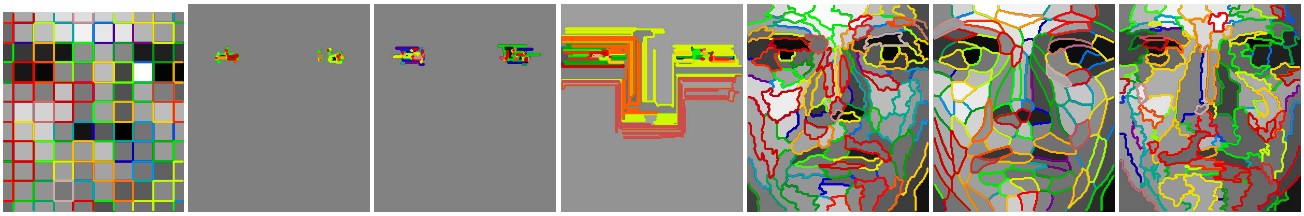


Figure 6.1: **Complexity of the computation time on a synthetic dataset:** Evaluation of the time complexity to find $\mathbf{\Phi}$ per algorithm on a synthetic data cube, for various feature space dimensions $p \in \{8, 16, 64, 128\}^3$ and fixing the number of clusters to $k = \lfloor \frac{p}{20} \rfloor$. Downsampling displays a sub-linear time complexity, whereas Nyström, single-linkage, complete-linkage, SLIC and ReNA present a linear behavior. Complete-linkage, Ward and random projections have a sub-quadratic time behavior.

Evaluating feature grouping properties

We evaluate the performance of $\mathbf{\Phi}$ with three measures: *i)* The computation time when varying the number k of clusters for a fixed dimension p, *ii)* the signal distortion for various number of clusters; *iii)* the size of the largest cluster. Tuning dimensionality reduction to the data at hand may capture noise in addition to signal. Hence we apply the learned $\mathbf{\Phi}$ on left-out data, in a cross-validation scheme splitting the data randomly 50 times. Each time, we extract the clustering on half of the noisy data and apply it to the other half to measure distortion with regards to the non-noisy signal. We vary the number of clusters $k \in [0.01p, p]$, because all the tested algorithms worked properly in this range. In particular, the implementation used for SLIC [157] posed problems to control the number of clusters in a larger regime. For the Nyström approximation, we vary the dimensionality $k \in [0.01n, n]$.



(a) downsampling (b) single-linkage (c) average-linkage (d) complete-linkage (e) Ward (f) SLIC (g) ReNA

Figure 6.2: **Example of clusters obtained for the extended Yale B face dataset using various feature grouping schemes:** Clusters found on the faces images ($k = 120$). Single, average and complete linkage clustering fail to represent the spatial structure of the data, finding a huge cluster leaving only small islands apart. Downsampling fails to capture the global appearance. In contrast, methods yielding balanced clusters maintain this structure. Colors are random.

The clusters found by the clustering algorithms on the faces dataset are shown on Fig. 6.2. We can see that single, average, and complete linkage have percolated, failing to retain the spatial structure of the faces. Downsampling also fails to capture this structure, while Ward, SLIC and ReNA perform well in this task. Additionally, Fig. 6.2 shows the approximations of brain images using clustering methods. As previously, percolating algorithms fail to represent spatial features of the data.

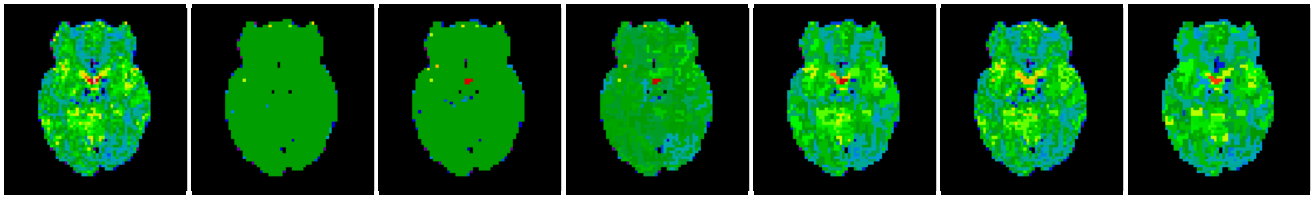
Distortion: We want to test whether the reduction $\mathbf{\Phi}\,\mathbf{X}$ of the noisy data is true to the uncorrupted signal $\mathbf{S}$,

$$\|\mathbf{\Phi}\,\mathbf{X}_{*,i} - \mathbf{\Phi}\,\mathbf{X}_{*,j}\|_2 \approx \|\mathbf{S}_{*,i} - \mathbf{S}_{*,j}\|_2,\ \forall (i,j) \in [n]^2. \tag{6.1}$$

Then, to assess the quality of this approximation (see Eq. 3.2 and Eq. 4.7), we randomly split half of the data to form a train and test clean signals $(\mathbf{S}^{\text{train}}, \mathbf{S}^{\text{test}})$ and a train corrupted data matrix $\mathbf{X}^{\text{train}}$. We learn $\mathbf{\Phi}$ on the train corrupted data $\mathbf{X}^{\text{train}}$. On the test data, we fit a proportionality constant η that relates the distances in reductions of the corrupted data with the corresponding distances in the clean signals[2].

We denote $\delta^{\text{orig}}_{(i,j)}$ the norm of the difference of the i and j uncorrupted signals, $\|\mathbf{S}^{\text{test}}_{*,i} - \mathbf{S}^{\text{test}}_{*,j}\|_2$, and $\delta^{\text{noisy}}_{(i,j)}$ the norm of the difference of the i and j scaled noisy signals, $\eta\|\mathbf{\Phi}\,\mathbf{X}^{\text{test}}_{*,i} - \mathbf{\Phi}\,\mathbf{X}^{\text{test}}_{*,j}\|_2$, for all $(i,j) \in \left[\left\lfloor \frac{n}{2} \right\rfloor\right]^2$ (note that

[2] We assume that the approximation in Eq. 6.1 can be summarized by a proportionality



(a) Original (b) single-linkage (c) average-linkage (d) complete-linkage (e) Ward (f) SLIC (g) ReNA

Figure 6.3: **Approximation of an MRI image obtained with various feature grouping algorithms:** A compressed representation of an MRI image (slice) using various clustering methods for a number of clusters $k = 1000$. Traditional agglomerative clustering methods exhibit giant clusters, loosing meaningful information. In contrast, Ward, SLIC and ReNA algorithms present a better performance, finding balanced clusters.

$\delta \in \mathbb{R}^{n(n-1)/8}$). We then use the relative distortion (RD) between δ^{orig} and δ^{noisy} to quantity the denoising effect of each method:

$$\text{RD}(\delta^{\text{orig}}, \delta^{\text{noisy}})(\text{dB}) = -\mathbf{10}\,\mathbf{log_{10}}\,\frac{\|\delta^{\text{noisy}} - \delta^{\text{orig}}\|_2^2}{\|\delta^{\text{orig}}\|_2^2}. \tag{6.2}$$

This measure gives us an insight on the distortion and possibly denoising effect. In particular, it shows us for which fraction of the signal the condition of Eq. 4.6 is satisfied. This experiment is carried out on two datasets: *i)* Synthetic data, and *ii)* brain activation images (motor tasks) from the HCP dataset.

The results on the distortion behavior are presented in Fig. 6.4 (*top*). Note that SLIC displays an early stopping due the its inability to control the number of clusters. In synthetic data, the clustering methods based on first order linkage criteria (single, average, complete linkage) fail to represent the data accurately. By contrast, SLIC, Ward and ReNA achieve the best representation performance. These methods also show an expected denoising effect inside the useful fraction of the signal value ($k = \{\lfloor p/20 \rfloor, \lfloor p/10 \rfloor\}$), whereby the learned approximation matches approximately the smoothing kernel that characterizes the input signal. Downsampling also exhibits a denoising effect, needing more components than the non-percolating methods. For the HCP dataset, the denoising effect is subtle, given that we do not have access to noiseless signals. Downsampling and Random projections find a plateau in the relative distortion (RD) curve, meaning that the signal has a low entropy that is captured with only few components. In both datasets, dimensionality reduction by random projections and random sampling fail to diminish the noise component, this is due to their property to maintain approximate distance, representing also the noise.

Percolation behavior: Given that percolation is characterized by the occurrence of one huge cluster for mild or large k values, we monitored the size of the largest cluster when varying the number k of clusters.

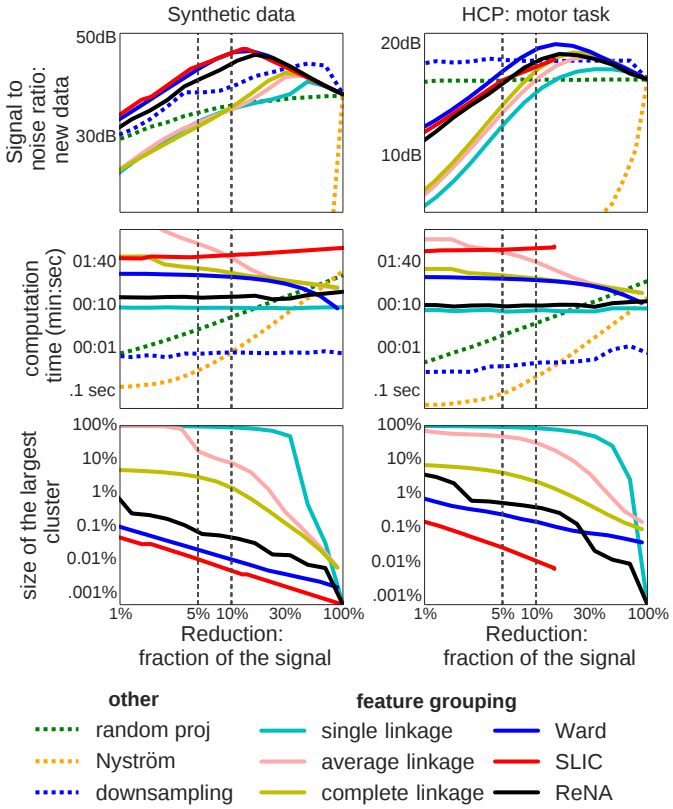


Figure 6.4: **Quality assessment of various approximation techniques on synthetic and brain imaging data:** Evaluation of the performance for several number k of clusters. *(top)* Empirical measure of the distortion, RD of the approximated distance. For a fraction of the signal between 5% and 30% Ward, SLIC and ReNA present a denoising effect, improving the approximation of the distances. In contrast, traditional agglomerative clustering fails to preserve the distance in the reduced space. Downsampling displays an intermediate performance. *(center)* Regarding computation time, downsampling and random sampling outperform all the alternatives, followed by random projections and single-linkage. The proposed method is almost as fast as single-linkage. *(bottom)* Percolation behavior, obtained through the size of the largest cluster. Ward, SLIC and ReNA are the best avoiding huge clusters. The vertical dashed lines indicate the useful value range for practical applications ($k \in [\lfloor p/20 \rfloor, \lfloor p/10 \rfloor]$).

The results of tracking the size of the largest cluster are presented in Fig. 6.4 (*bottom*). This shows that among the traditional agglomerative methods, single, average and complete linkage display the worst behavior with a persistent percolation. Complete-linkage exhibits a more complex behavior, with the occurrence of relatively large clusters in the large k regime, that grows slowly in the small k regime. On the other hand, Ward and SLIC are the most resilient methods to percolation, both known for their tendency to create equally large clusters. Finally, ReNA achieves a slightly worst performance, but mostly avoids huge clusters. The results are again across both datasets.

Computation time: Computation-time of the different methods are displayed in Fig. 6.4 (*center*). Dimension reduction by downsampling is overall fastest, as it does not require any training and the computational time lies in the linear interpolation. It is followed by the Nyström approximation, with faster computation for a small number k of components. While the computation time of Nyström approximation and random projections increases with the reduction fraction of signal, clustering algorithms are faster, as they require less merges. Random projections are faster than clustering approaches to reduce signals to a size smaller than 30% of their original size. In the clustering approaches, single-linkage and ReNA are

the fastest, as expected. Note that the cost of the clustering methods scales linearly with the number of samples, hence can be reduced by subsampling: using less data to build the feature grouping.

6.3 Use in prediction tasks

To evaluate the denoising properties of dimension reduction, we now consider their use in prediction tasks. We use linear estimators, as they are standard in high dimensional large-scale problems. In particular, we focus on linear estimators with ℓ_2 or ℓ_1 penalties. For the ℓ_2 case, the estimation problem ${\mathbf{\Phi}_{\mathsf{FG}}}^{\mathsf{T}} \mathbf{\Phi}_{\mathsf{FG}}$ acts like a kernel of the quadratic form. For such estimators, dimension reductions that preserve pairwise distance are well motivated theoretically [136]

For each problem, we use the relevant metric (i.e. explained variance[3] for regression and accuracy[4] for classification) to assess the performance with different dimension reduction methods, as each one yields a different estimator. These results are then compared with those obtained based on raw data.

[3] The explained variance is defined as $R^2 = 1 - \frac{\mathrm{Var(model - signal)}}{\mathrm{Var(signal)}}$

[4] Accuracy is defined by: $1 - \frac{\text{number of miss-classifications}}{\text{total number of samples}}$

Spatial approximation on a faces recognition task

Face recognition is a classic computer-vision task. A standard pipeline to tackle this problem consists of dimensionality reduction of the data and then training a classifier to recognize an unseen face image. Some of the pipelines include random projections, PCA, downsampling [173] or dictionary learning [174]. It has been shown that images of a subject with a fixed pose and varying illumination lie close to a low-dimensional subspace [16]. This justifies the use of data approximation, in particular with linear projections.

To perform the classification task, we use an ℓ_2 or ℓ_1 logistic regression with a multi class *one-vs-rest* strategy and set the regularization parameter λ by 10-fold nested cross-validation. We mimic a study on reduced face representations [173], computing prediction accuracy for various feature-space dimensions $k \in \{30, 56, 120, 504\}$, corresponding to downsampling ratios of $\{1/32, 1/24, 1/16, 1/8\}$ respectively. Prediction error is measured in 50 iterations of a cross-validation loop randomly selecting half of images in each subject for the training and the other half for testing.

Fig. 6.5 reports the prediction accuracies. For high reduction factors ($k = 30$), Ward, Nyström, and ReNA perform up to 10% better than random projections or downsampling: representations adjusted on the data outperform data-independent reduction operators. For raw data, without reduction, prediction accuracy is around 95.3% and 94.1% for the ℓ_1 and ℓ_2 penalization respectively. Similar performance is obtained after reducing the signal by a factor of 64 with random projections, Nyström, downsampling, Ward, SLIC or ReNA. In contrast, single, average and complete linkage

clustering fail to achieve the same performance. This shows the importance of finding balanced clusters.

Regarding computation time, data reduction speeds up the convergence of the logistic regression. Nyström and downsampling are the fastest methods. They are followed by random projections, single-linkage and ReNA, all of them having similar performances. Average, complete linkage, Ward and SLIC are slightly slower on this dataset.

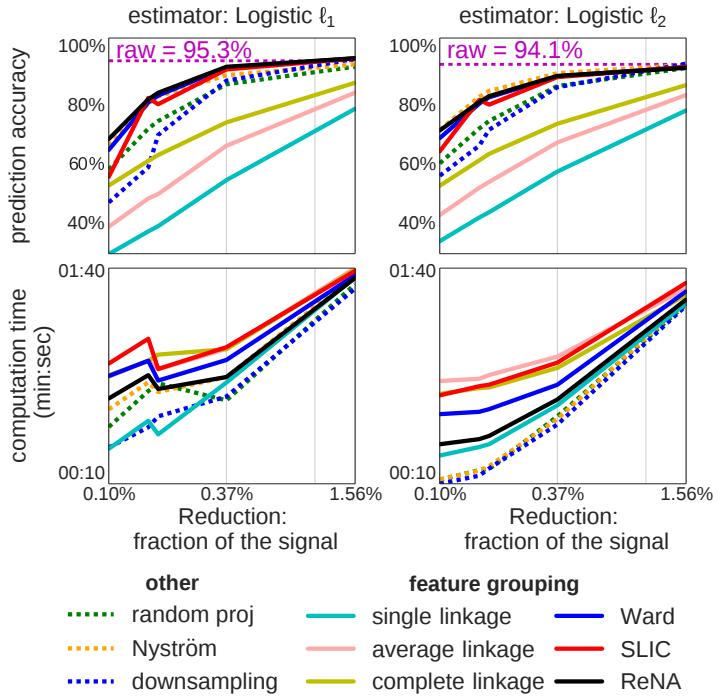


Figure 6.5: **Impact on face prediction accuracy for various approximation schemes:** Prediction accuracy as function of the feature space dimension obtained for various approximation schemes and classifiers for the recognition of human identities on the extended Yale B dataset. The clustering methods finding balanced clusters need less features to have a fair performance, and they also obtain significantly higher scores than the percolating methods.

Convergence time – a good solution on a time budget

We now turn to evaluating dimension reduction on brain imaging data. Machine learning techniques is often used on brain images to link brain regions with external variables (e.g. experiment conditions or cognitive tasks) [105][142]. Linear models are generally used as their coefficients form brain maps that can be interpreted [111]. With progress in MRI, brain images are becoming bigger, leading to computational bottlenecks. The Human Connectome Project (HCP) is prototypical of these challenges, scanning 1 200 subjects with high-resolution protocols. We consider two brain-imaging prediction problems: *i)* gender classification using anatomical brain images of the 400 subjects from OASIS dataset and *ii)* the prediction of 17 cognitive tasks using functional brain images from 483 subjects of the HCP dataset. Here we study how dimension reduction can speed up convergence of classifiers. To do so, we measure the computation time needed to reach a stable solution for an ℓ_2 logistic regression, including the cost of computing the compressed

representation and of training the classifier. We use a multinomial logistic regression with an ℓ_2 penalty and an L-BFGS-based solver.

In Fig. 6.6, we report prediction results as a function of computing time on the OASIS and HCP dataset. In both datasets, feature clustering with single, average, and complete linkage lead to poor prediction. This was expected due their tendency to find unbalanced clusters (percolation). On the other hand, Ward, SLIC, and ReNA obtain better prediction accuracy than raw data. All three approaches converge to similar accuracies, though with different convergence time. On the HCP dataset, SLIC takes more time to find the clusters, but it requires only a few iterations to converge, likely because it finds good quality clusters. Downsampling displays uneven performance, on the HCP dataset it performs slightly better than raw, while performing marginally worse on the OASIS dataset. In both datasets, the Nyström approximation and random projection achieve a lower prediction level than raw, this is because these methods capture both data and noise.

Improved performance of feature clustering compared to raw data highlights the importance of its signal denoising effect. ReNA strikes an excellent balance, as it reaches accuracies above that of raw data fastest.

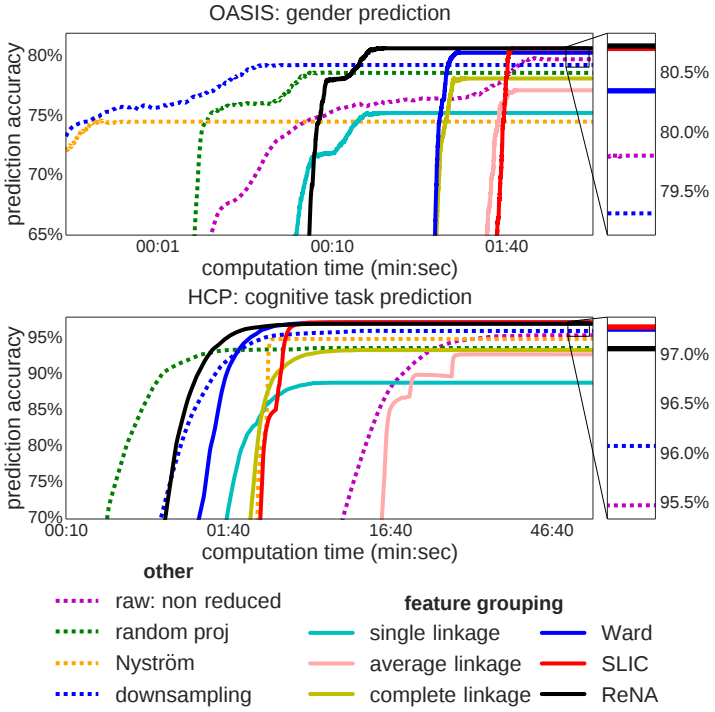


Figure 6.6: **Computation time taken to reach a solution:** Quality of the fit of a ℓ_2 penalized logistic regression as function of computation time for a fixed number of clusters. In both datasets, Ward, SLIC and ReNA obtain significantly higher scores than estimation on non-reduced data with less computation time to reach a stable solution. Note that the time displayed does include cluster computation.

Impact of the approximation on prediction accuracy

Here, we examine the impact of the signal approximation on prediction accuracy. We use various datasets (i.e. faces, anatomical and functional brain images), setting $k = \lfloor p/20 \rfloor$ for random projections, downsampling

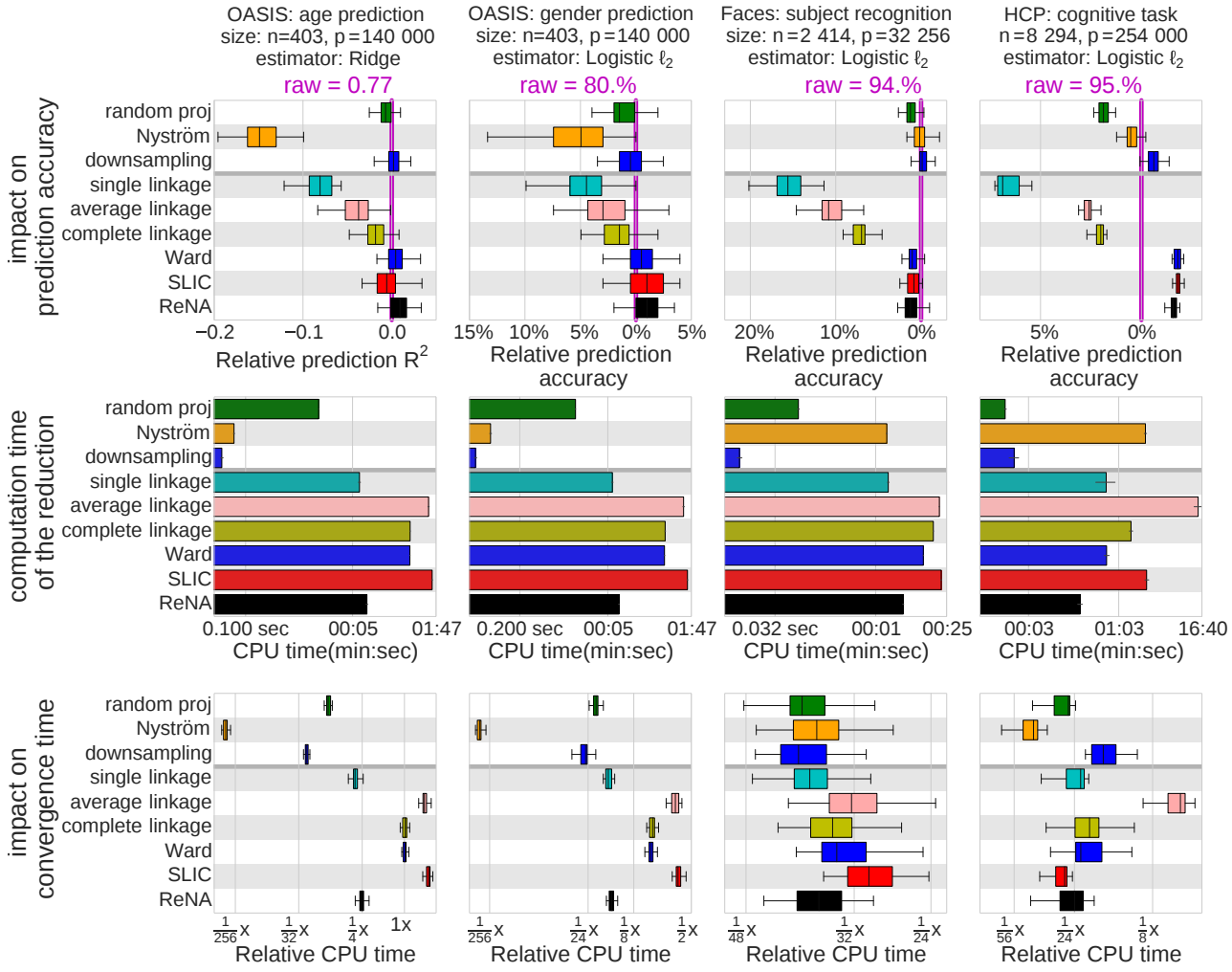


Figure 6.7: **Impact of reduction methods on prediction for various datasets:** *(Top)* Each bar represents the impact of the corresponding option on the prediction accuracy, relatively to the mean prediction with non-reduced data. Downsampling has the same performance as raw data. On the other hand, random projections, Nyström, single, average and complete linkage algorithms are consistently the worst ones across datasets. Ward, SLIC and ReNA perform at least as good as non-reduced data. *(middle)* Regarding the computation time to find a reduction, single-linkage and ReNA are consistently the best among the clustering algorithms. Random projections perform better than Nyström when the number of samples is large. Downsampling is the fastest across datasets. *(Bottom)* The time to converge for single-linkage and ReNA is almost the same. Average, complete-linkage and Ward are consistently the slowest. SLIC performs well on large datasets. Nyström and random projections are the fastest across datasets. Single-linkage and ReNA are the fastest clustering methods. ReNA strikes a good trade-off between time and prediction accuracy.

and clustering methods, and $k = \lfloor n/10 \rfloor$ for Nyström. For the faces dataset, we use the $k = 504$ for all the methods; this value corresponds to a downsampling ratio of $1/8$. In addition, we predict age on the OASIS dataset, using a Ridge regression and setting its regularization parameter via cross validation. For each prediction task, we measure for all dimension reduction schemes the prediction accuracy, relatively to the mean prediction with raw data.

Fig. 6.3 summarizes the impact of dimension reduction on prediction accuracy and computation time. Dimension reduction with clustering algorithms that yield balanced clusters (Ward, SLIC, and ReNA) achieve similar or better accuracy as with raw data while bringing drastic time savings. Random projections and the percolating methods give consistently worse prediction accuracy than raw data. On the OASIS dataset, downsampling, SLIC, and Ward achieve the same prediction accuracy as raw, and perform better than raw on other datasets. Nyström only performs as good as raw data on the faces dataset with an ℓ_2 penalized logistic regression. ReNA has a slightly worst performance than raw only in this dataset, and displays a better performance than raw on the remaining datasets (p-value $< 10^{-4}$). This illustrates the reduction of the spatial noise exhibited by non-percolating clustering methods.

6.4 Use in a spatial ICA task

Aside from the ℓ_1-penalized estimator, the data processing steps studied above depend only on the pairwise distances between samples. We now we investigate dimension reduction before an Independent Component Analysis (ICA), which probes higher moments of the data distribution. We use ICA on resting-state fMRI from the HCP dataset. ICA is used routinely on rest fMRI to separate signal from noise and obtain a spatial model of the functional connectome [149]. We use 93 subjects, with two resting-state fMRI sessions, each containing 1200 brain images.

We compare ICA on the raw data and after dimension reduction to five percent of the number of voxels ($k = \lfloor \frac{p}{20} \rfloor$). For the Nyström method the dimension is set to ten percent of the number n of samples ($k = \lfloor \frac{n}{10} \rfloor$). In each subject, we extract 40 independent components as it is a standard number in the literature. We investigate *i)* how similar the components obtained are before and after reduction; *ii)* how similar the components of session 1 and session 2 are with different reduction approaches. This second experiment gives a measure of the variability due to noise. In both cases, we measure similarity between components with the absolute value of their correlation, and match them across sessions with the Hungarian algorithm.

Fig. 6.4 summarizes the use of dimension reduction in ICA of rest fMRI. We find that the 40 components are highly similar before and after data reduction with downsampling and Ward: the average absolute correlation greater than

0.8. SLIC and ReNA have a slightly worse performance, with an average correlation greater than 0.74. On the other hand, single-linkage, average-linkage, complete-linkage, Nystöm, and random projections do not recover the components (average correlation < 0.4). As expected, the components between sessions obtained by non-percolating clustering (Ward, SLIC, and ReNA) are similar to the original ones. Donwsampling improves the similarity with respect to raw: the estimation problem is simpler and less noisy. On the opposite, single, average, and complete linkage yield a degradation of the similarity. This is caused by their tendency to percolate, hence dismiss information. Random projections and Nyström perform poorly. Indeed, they average data across the images, destroying the high-order moments of the data by creating signals more Gaussian than the originals. As a consequence, ICA cannot recover the sources derived from the original data. By contrast, the non-percolating clustering algorithms extract local averages of the data, that preserve its non-Gaussianity, as it has a spatial structure. Hence the spatial ICA is successful even though it has access to less samples. Finally, dimensionality reduction using ReNA speeds up the total analysis by a factor of 15.

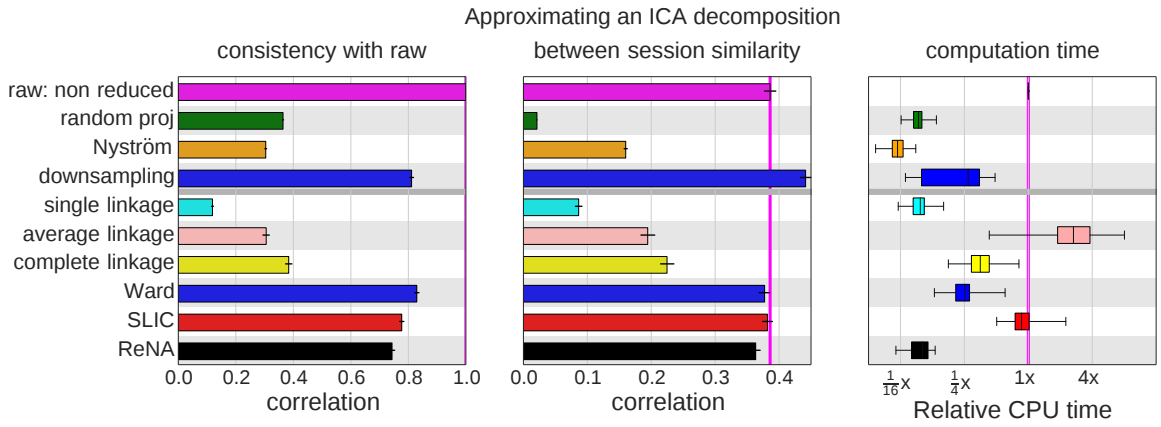


Figure 6.8: **Reproducibility of a spatial ICA after dimension reduction:** Reproducibility of 40 spatial independent components on 93 individual functional brain images dataset, with a fixed reduced dimension. *(Left)* the reproducibility of downsampling, Ward, SLIC and ReNA with respect the non-compressed components is high. *(Middle)* across two sessions, donwsampling yields components more consistent than raw data. Ward, SLIC and ReNA perform as good as raw data, while other approaches fail to do so. *(Right)* regarding computational time, ReNA outperforms downsampling, Ward and SLIC, with a performance similar to single-linkage and random projections. It yields a gain factor of 16 with respect to raw data.

6.5 Summary and Discussion

Fast dimension reduction is a crucial tool to tackle the rapid growth in datasets size, sample-wise and feature-wise. In particular, grouping features

is natural when there is an underlying regularity in the observed signal, such as spatial structure in images or a more general neighborhood structure connecting features. We studied here a data-driven approach to feature grouping, where groups are first learned from a fraction of the data using a clustering algorithm, then used to build a compressed representation for further analysis.

Our experiments have shown that on moderate-to-large datasets, non-percolating feature-grouping schemes –i.e. Ward, SLIC, and ReNA– most often outperform state-of-the-art fast data-approximation approaches for machine learning, namely random projection and random sampling. Using these methods in a predictive pipeline increases the quality of statistical estimations: they yield more accurate predictions than using all features. This indicates that feature grouping leads to a good approximation of the data, capturing the structure and reducing the noise. This denoising is due to the smoothness of the signal of interest: unlike the noise, the signal displays structure captured by feature grouping.

A key benefit of the ReNA clustering algorithm is that it is very fast while avoiding percolation. As a result, it gives impressive speed-ups for real-world multivariate statistical problems: often more than one order of magnitude. Note that the computational cost of ReNA is linear in the number of samples, hence additional computation gains can be obtained by sub-sampling its training data, as in Nyström approaches. In this work, we did not investigate the optimal choice of the number k of clusters, because we do not view compressed representations as a meaningful model per se, but as an approximation to reduce data dimension without discarding too much information. The range $k \in \left[\left\lfloor \frac{p}{20} \right\rfloor, \left\lfloor \frac{p}{10} \right\rfloor\right]$ is a useful regime it gives a good trade-off between computational efficiency and data fidelity. In our experiments, $k = \left\lfloor \frac{p}{20} \right\rfloor$ gave enough data fidelity for statistical analysis to perform at least as good as raw data. In this regime, Ward clustering gives slightly better approximations of the original data, however it is slower, often by several factors, hence is not a practical solution.

We have shown that feature grouping is useful beyond ℓ_2-distance-based methods: it also gives good performance on estimators relying on higher order moments (e.g. ICA) or sparsity (ℓ_1-based regression or classification)[5]. As future work, it would be interesting to investigate the use of ReNA-based feature grouping in expensive sparse algorithms, for instance with sparse dictionary learning, where feature sub-sampling can give large speed ups [102]. Similarly, the combination of clustering, randomization, and sparsity has also been shown to be an effective regularization for some ill-posed inverse problems [161][22]. We conjecture that ReNA clustering is particularly well-suited for these problems. This is all the more important that computation cost is a major roadblock to the adoption of such estimators.

Given that ReNA clustering is extremely fast, the proposed featuring-grouping is a promising avenue to speed up any statistical analysis of large

[5] On the faces datasets for the ℓ_1 estimator, ReNA performs as well as raw data.

datasets where the information is in the large-scale structure of the signal. Such an approach is crucial for domains where the resolution of the sensors is rapidly increasing, in medical or biological imaging, genomics or geospatial data.

Part III

Contribution – improving the stability of brain decoders

7 Decoding with ensembles of models

In the previous chapters, we have considered *feature grouping* as a dimension reduction scheme of structured signals. We have proposed a fast agglomerative clustering algorithm that finds balanced clusters, *ReNA*. In addition, we validated it by extensive experiments, showing its computational efficiency and denoising behavior.

In this chapter, we are interested in brain activation decoding. In particular, we consider the *stability* of the decoder as a first step towards *reproducibility*. Here, the stability is measured relatively to data perturbations. These perturbations are obtained by sampling from an underlying distribution or replicating the experiment with a new set of data.

We propose to use ensembles of models –e.g. model aggregation– to improve the stability of decoders weight maps, as well as their prediction accuracy. But given the high-dimensional setting of neuroimaging data, choosing the hyperparameters of the aggregated estimator is computationally expensive. To tackle this, we make use of the decoder set in the nested cross-validation loop to find a "good" estimator per iteration, then we build the decoder by aggregating these estimators. In addition, we include an implicit spatial constraint by using *feature grouping*.

The contributions developed in this chapter, along with the next two, have been published in:

Improving sparse recovery on structured images with bagged clustering. Andrés Hoyos-Idrobo, Gaël Varoquaux and Bertrand Thirion. *PRNI - IEEE International Workshop on Pattern Recognition in Neuroimaging*. 2015, Stanford University, Palo Alto, USA.

7.1 Introduction: decoding needs stability

Decoding models reconstruct stimuli or behavior from brain images. These models have become a standard tool in neuroimaging data processing [68][160][115]. In clinical applications, these models open the possibility of performing diagnosis or prognosis [48][34]. They are also used as evidence of the link between brain regions [67][108] and an observed behavior. Decoding can map a large variety of cognitive processes [127].

In brain decoding, the main goal is to retrieve and understand the patterns of brain images that drive a good prediction, that is: identifying the brain regions involved in the cognitive processing of an external stimulus. Yet, training a reliable decoder is challenging due to the dimensionality of the problem: the number of samples is small –hundreds or less–, whereas the number of features is typically the number of voxels in the brain –up to hundreds of thousands. Linear models, e.g. linear support vector machines (SVM), are often used [126], as they have shown a good performance in a small-sample regime. In addition, their classification/regression weights form brain maps used for interpretation of the discriminative pattern [108].

However, the high dimensionality of the problem leads to multiple weight maps yielding the same predictive power, and some form regularization has to be applied [64]. In across-subject settings, spatial and sparse penalties such as total-variation (TV) [103] and Graph-net [61] help the decoder to capture the important brain regions shared across subjects. TV and its variants are considered as the state-of-the-art regularizers for brain images, as they handle local correlations present in the data. The main drawback of spatial sparsity as in TV and related penalties is their computational cost.

Assessing the predictive power of the decoder is important, as it provides a figure of merit of the model. The setting of hyperparameters is also essential, and typically requires some measure of accuracy. In practice, we use cross-validation to perform these tasks [162]. This method requires training the decoder several times to build an empirical distribution of the predictive performance. These repeated calculations entail computational costs that are intractable in standard workstations [106]. The computation resources are an important limitation to account for.

Another desired characteristic of decoding algorithms is the stability to data perturbations. These perturbations are defined as sampling from an underlying distribution or replicating the experiment for a new set of data. In practice, we do not have access to this distribution. Instead we have to rely on data sampling/resampling methods to create an empirical distribution –e.g bootstrap, cross-validation. Stability assumes that small variations in the data yield a commensurate variation in the weight map.

One approach to reduce the variability of the decoder is to use ensembles of models [176]. A simple version of this idea is to build a decoder by averaging several "good" models. One way is to use bootstrap resampling to generate several training sets and corresponding models, and then aggregate them: *Bagging*[1] [20]. This approach is easily parallelizable, as each model is trained independently. However, the application of this idea does not translate straightforwardly to a high-dimension setting, as choosing the hyperparameter of the aggregated model is computationally expensive.

[1] Bagging is a sobriquet for Bootstrap aggregating

In this chapter, we propose a simple scheme to reduce the variance of the weight maps of the decoder. This method consists in averaging the estimator with the best predictive power per loop inside the nested cross-validation. In addition, we include a spatial denoising step using *feature grouping*. We assume that by averaging several models the resulting decoder is more robust to violations of modeling assumptions.

Brain decoding

In neuroimaging, a decoder is a predictive model that, given n brain images, infers an external variable $\mathbf{y}$. In practice, we arrange n observed brain images composed of p voxels in a matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$. $\mathbf{y}$ denotes a target variable giving the experimental condition or health status of subjects. In linear-regression setting, $\mathbf{y} \in \mathbb{R}^n$, and in the case of classification $\mathbf{y} \in \{-1, 1\}^n$. Typically, we use the following linear predictive model [65]:

$$\mathbf{y} = f(\mathbf{X}\boldsymbol{\omega} + \boldsymbol{\epsilon}), \tag{7.1}$$

where f represents the decision function in the classification or the identity in the case of regression; $\boldsymbol{\omega} \in \mathbb{R}^p$ denotes a fixed but unknown weight vector/map, and $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is a random error term, which is independent of $\mathbf{X}$ and has mean zero. Formally, we have a sequence of n independent and identically distributed (i.i.d.) input-output pairs $\{(\mathbf{X}_i, \mathbf{y}_i)\}_{i=1}^n$ distributed according to an unknown distribution P. Then, our aim is to find a suitable estimator, parametrized by the unknown $\boldsymbol{\omega}$ weight maps.

In spite of a recently growing effort on the accumulation of neuroimaging data [130], the number n of samples per-class remains in the order of a few hundreds, whereas p can be hundreds of thousands of voxels ($p \gg n$). In this high-dimensional setting there are many equivalent solutions and some form of regularization or prior is necessary to restrict model complexity. A standard approach relies on solving the following optimization problem:

$$\hat{\boldsymbol{\omega}}(\lambda) = \underset{\boldsymbol{\omega} \in \mathbb{R}^p}{\operatorname{argmin}} \left\{ \mathcal{L}(\mathbf{y}, \mathbf{X}; \boldsymbol{\omega}) + \lambda \Omega(\boldsymbol{\omega}) \right\}, \quad \lambda > 0, \tag{7.2}$$

where $\mathcal{L}$ is a data-fidelity term, a loss function that measures the quality of the estimator (e.g. logistic loss, hinge loss); Ω denotes the penalty/regularization term, and λ is the parameter that controls the amount of regularization. In

practice, we chose Ω to be a convex but not necessarily smooth. Two of the most often used penalties are: *1)* ℓ_2-penalty, $\|\boldsymbol{\omega}\|_2^2 = \sum_{i=1}^{p} \boldsymbol{\omega}_i^2$, that penalizes large $\boldsymbol{\omega}$-coefficients, it is non-sparse; *2)* ℓ_1-penalty, $\|\boldsymbol{\omega}\|_1 = \sum_{i=1}^{p} |\boldsymbol{\omega}_i|$, that promotes a small number of non-zero $\boldsymbol{\omega}$-coefficients: it yields sparse solutions [153].

Model selection

In high-dimensional settings, the number of candidate models is much larger than the sample size. Therefore, we use regularization to constrain the complexity of the solution, and this penalization is controlled by the λ regularization parameter. Then, our aim is to find a model that exploits the richness of the data, finding the best bias-variance trade-off. We use the predictive power of the decoder to chose the right amount of regularization.

Estimating the predictive power: The accuracy, or the predictive power of a decoder is defined as the expected error on the prediction, formally:

$$\text{accuracy} = \mathbb{E}\left[\eta(\mathbf{y}_{\text{pred}}, \mathbf{y}_{\text{ground truth}})\right], \tag{7.3}$$

where η is measure of the error, most often[2] the fraction of instances for which $\mathbf{y}^{\text{pred}} \neq \mathbf{y}^{\text{ground truth}}$. We need the underlying distribution of the data to compute the Eq. 7.3, but in practice it is unknown. Instead, we use data perturbation schemes to assess the predictive power of each model [9]. We create b pseudo datasets, $\left\{(\mathbf{X}^{*(j)}, \mathbf{y}^{*(j)})\right\}_{j=1}^{b}$, where each one draws m samples with or without replacement from $(\mathbf{X}, \mathbf{y})$. Each pseudo dataset is split into a train set and a test set: $(\mathbf{X}_{\text{train}}^{*(j)}, \mathbf{y}_{\text{train}}^{*(j)})$ and $(\mathbf{X}_{\text{test}}^{*(j)}, \mathbf{y}_{\text{test}}^{*(j)})$ respectively. We use the train set to fit the decoder, and use the test set to measure its ability to generalize to new data .

[2] In multiclass problems or for unbalanced classes, the measure of the error have to be more elaborate to distinguish misses and false detections for each class.

Hyperparameters selection: In general, the setting of the hyperparameter is a data-specific choice, as it is governed by the amount of data and the signal-to-noise-ratio (SNR). The most common approach to set it is to use cross-validation to measure the predictive power of various amounts of regularization and retain the value that maximizes the predictive power across several data perturbations [162]. To assess predictive power in addition, the standard scheme is nested cross-validation, which consists of two cross-validation loops run one inside the other: an outer loop is used to assess the predictive power of the decoder, and an inner/nested loop is used to set the hyperparameter (see Fig. 7.1).

In most of the non-parametric approaches to select a regularization parameter, we first define a suitable and finite set of l hyperparameters, $\lambda \in [\lambda_1, \ldots, \lambda_l]$, $\lambda_i > 0$ for $i \in [1, \ldots, l]$. Hence, we fit l models on these b datasets, as follows:

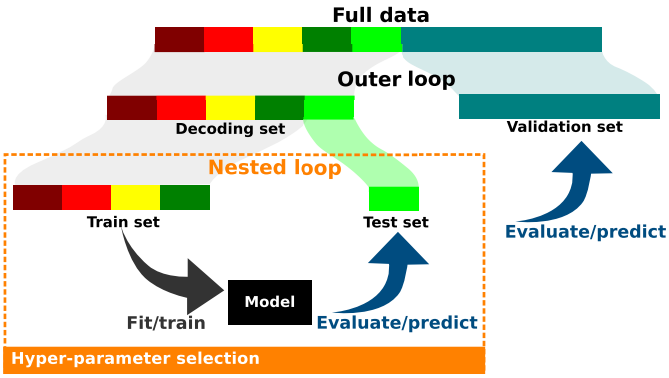


Figure 7.1: **Illustration of nested cross-validation:** Two cross-validation loops are run one inside the other. The inner loop is used to set the hyperparameters, whereas the outer loop is used to assess the predictive power of the decoder.

$$\hat{\omega}^{(j)}(\lambda_i) = \underset{\omega \in \mathbb{R}^p}{\operatorname{argmin}} \left\{ \mathcal{L}(\mathbf{y}^{*(j)}_{\text{train}}, \mathbf{X}^{*(j)}_{\text{train}}; \omega) + \lambda_i \Omega(\omega) \right\}, \tag{7.4}$$

where $j \in [1, \ldots, b]$ and $i \in [1, \ldots, l]$. One measures the prediction error of each i-model on $\mathbf{X}^{(j)}_{\text{test}}$. Then, one choses the λ_i value that maximizes the predictive power across b datasets. Nevertheless, we must note that in the case of multicollinearity, a strong correlations of the columns of $\mathbf{X}$, the ℓ_1-penalty is not stable as it tends to arbitrarily select one among the correlated variables and not the others [175][161]. One way to tackle this is the use of spatially informed penalties as Graph-net [61] or TV related [103][43]. However, these complex penalties come with a higher computational cost [106], which hinders the statistical validation of the predictive power as we have to perform several iterations of the outer cross-validation loop.

Ensembling methods for better stability

Model averaging: Model selection can run into some issues due to instability in the choice of the model, as any perturbation of the original data entails the selection of a completely different hyperparameter [9]. Model averaging mitigates this problem by aggregating the output of several suitable models [112]. A simple version of this idea is bootstrap aggregation (bagging) [20]. This method improves the predictive power of the base estimator, reducing the variance if the errors each of model are sufficiently uncorrelated. The bagged/aggregated estimator is built by averaging predictors.

Bagging in regression [20]: We present Breiman's proof of bagging regressors to demonstrate its benefits. Let D be a training set which contains a sample of independent (x, y) drawn from the distribution P. Define $h(x, D)$ to be prediction function based on the sample D. Let $h_{\text{aggr}}(x) = \mathbb{E}_D[h(x, D)]$ be an aggregate of prediction functions.

Take x to be a fixed predictor and y an output value. Then

$$\begin{aligned}\mathbb{E}_D[(y - h(x,D))^2] &\geq \mathbb{E}_D[y - h(x,D)]^2 \quad \text{(by Jensen's inequality)} \\ &= y^2 - 2y\mathbb{E}_D[h(x,D)] + \mathbb{E}_D[h(x,D)]^2 \\ &= (y - h_{\text{aggr}}(x))^2.\end{aligned}$$

Hence, the averaged predictor has lower mean-squared error than the base predictor. This improvement depends on how unequal $\mathbb{E}_D[h(x,D)^2] \geq \mathbb{E}_D[h(x,D)]^2$ are. The more the $h(x,D)$ vary with respect each other, the more improvement the aggregation may produce.

To understand the effect of averaging, let us assume the extreme case where the models created by sampling are i.i.d. Let the aggregation be the mean of the predicted values $h_{\text{aggr}}(x) = \frac{1}{b}\sum_{i=1}^{b} h_i(x,D)$, where $h_i(x,D)$ denotes the prediction function based on the sample D. The predictions are i.i.d., and the variance of each of them is defined by $\mathbb{E}_D[(y - h_i(x,D))^2] = \sigma^2$. Hence, the variance of the aggregated estimator is:

$$\begin{aligned}\mathbb{E}_D\left[(y - h_{\text{aggr}}(x))^2\right] &= \frac{1}{b^2}\mathbb{E}_D\left[(\sum_{i=1}^{b} y - h_i(x,D))^2\right] \\ &= \frac{1}{b^2}\sum_{i=1}^{b}\mathbb{E}_D\left[(y - h_i(x,D))^2\right] \quad \text{(by independence)} \\ &= \frac{1}{b}\sigma^2.\end{aligned}$$

We can see that averaging decreases the error as $\sqrt{b}$. Note that the i.i.d. case studied here is the most favorable case.

In particular, averaging linear models[3] boils down to:

$$\hat{\boldsymbol{\omega}}_{\text{bagg}}(\lambda) = \frac{1}{b}\sum_{j=1}^{b}\hat{\boldsymbol{\omega}}^{(j)}(\lambda), \tag{7.5}$$

where b is often chosen as 50 or 100, depending on the sample size and on the computation cost to evaluate the estimator $\hat{\boldsymbol{\omega}}$ [21]. Note that λ has to be set, requiring another nested loop of cross-validation.

[3] The bagged estimator is a Monte-Carlo approximation of $\mathbb{E}[\boldsymbol{\omega}]$.

Stability-based ensembles: In neuroimaging, another approach to improve the stability of the estimators is to train the base estimator on several random partitions of the feature space, then select representative features according to a consensus. These partitions can be defined using various criteria, for instance: *i)* random voxels selection[82][81]; *ii)* using clustering in decoding [161] and encoding [32] settings. Yet, these decoders need to fit more models, to accumulate selection statistics, and hence entail computational costs that are intractable given the number of models to fit.

7.2 Improving ensembling in high-dimensional settings

Dimension reduction – feature agglomeration: In neuroimaging, dimension reduction is routinely used to alleviate problems due to high-dimensionality, and it can be performed within the cross-validation loop to avoid overfitting. A common way to select features is univariate feature screening, which uses a score (e.g. statistical test, correlation) to remove non-predictive variables [37]. However, this method does not take into account the spatial structure of brain images. Hence, we can reduce the dimension of the data by grouping similar neighboring voxels, moving form the voxel-space to a parcel-space. To do this, we can use anatomical/functional atlases or data-driven approaches.

Here we rely on the ideas developed in the previous part –see chapter 3.4–, where we use a fraction of the training data to train a clustering algorithm, finding suitable groups of features or parcellations. Then, we use these parcels on the remaining data to work at a parcel level. Formally, we define a feature-grouping matrix $\mathbf{\Phi} \in \mathbb{R}^{p \times k}$, where $k \ll p$, and each column has a constant value with support at each parcel. We normalize each column to have unit ℓ_2-norm [74]. To reduce the dimension, we multiply the data by the feature-grouping matrix, $\mathbf{X}_{\text{reduced}} = \mathbf{X}\mathbf{\Phi}$. We can also build an approximation[4], $\mathbf{X}_{\text{approx}} = \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^{\dagger}$. When the feature-grouping matrix has independent columns, this approximation boils down to $\mathbf{X}_{\text{approx}} = \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^{\top}$.

[4] This approximation can be seen as the application of an anisotropic smoothing

This approach increases the SNR at the expense of spatial resolution without excluding potentially informative variables. It can be used in combination with sparse methods to alleviate their unstability when dealing with correlated variables [22][161].

Faster regularized ensembles of linear models: Setting the hyperparameters of ensembles of models can be computationally expensive, as a single aggregated estimator requires fitting b base estimators (see Eq. 7.5). To tackle this computational bottleneck, we can average weight vectors of nested cross-validation folds at best performing hyperparameter values (in the sense of predictive power). In addition, we can also add an implicit spatial constraint using clustering of features, applying it at each fold to increase the entropy of the clusters shapes. For completeness we detail the proposed pipeline in Algorithm 3 and Fig. 7.2.

7.3 Summary – training ensembles of models in neuroimaging

We proposed a scheme to train ensembles of linear models in high-dimensional settings. This scheme aggregates the best estimators of each nested cross-validation loop. Additionally, we include an implicit spatial constraint by using feature grouping at each iteration. Hence, this scheme

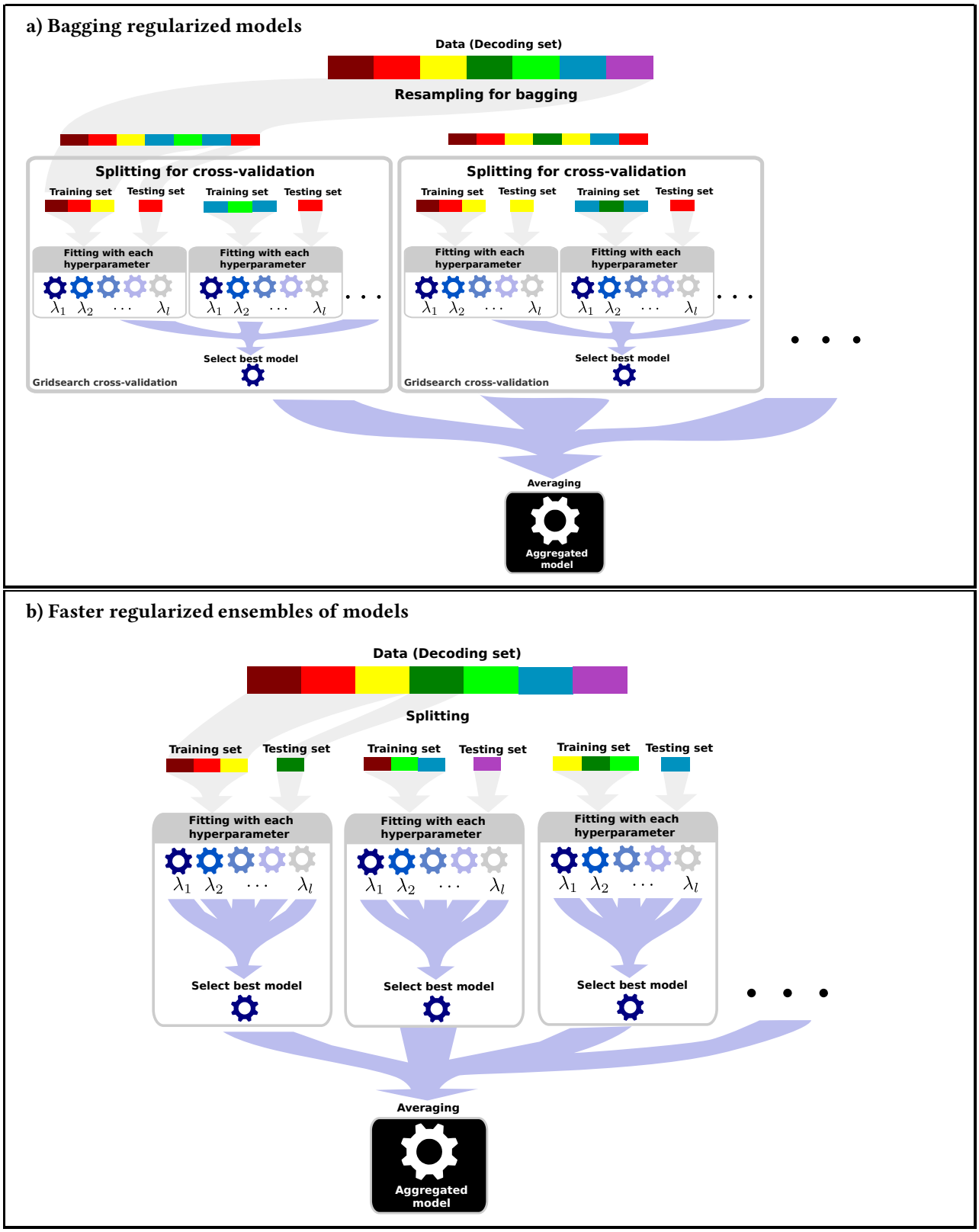


Figure 7.2: **Regularized ensembling of models:** Fast regularized ensembles of models uses one loop less than bagging of regularized models.

Algorithm 3 Fast regularized ensembling of models

Require: Training data $\mathbf{X} \in \mathbb{R}^{n \times p}$, desired number k of clusters, sampling parameter, number b of estimators to aggregate, set of hyperparameters $[\lambda_1, \ldots, \lambda_l]$, penalty Ω and loss $\mathcal{L}$.

Ensure: ω_{bagg}

1: **for** $j = 1$ **to** b **do**

2: *Build pseudo-dataset:* $\left\{(\mathbf{X}^{*(j)}, \mathbf{y}^{*(j)})\right\} \leftarrow \{(\mathbf{X}_i, \mathbf{y}_i)\}_{i=1}^{m}$, where $\mathbf{X}^{*(j)} \in \mathbb{R}^{m \times p}$, and $\mathbf{y}^{*(j)} \in \mathbb{R}^m$ {draw m samples from $(\mathbf{X}, \mathbf{y})$ at random}

3: *Split into a training set and a testing set:* $\left(\mathbf{X}^{*(j)}_{\text{train}}, \mathbf{y}^{*(j)}_{\text{train}}\right), \left(\mathbf{X}^{*(j)}_{\text{test}}, \mathbf{y}^{*(j)}_{\text{test}}\right)$ {Select $\lfloor \frac{m}{2} \rfloor$ samples at random (without replacement)}

4: *Build feature-grouping matrix:* $\mathbf{\Phi}^{(j)} \in \mathbb{R}^{p \times k}$ {Clustering of features using Eg. 2}

5: *Dimension reduction:* $\tilde{\mathbf{X}}_{\text{red}} \leftarrow \mathbf{X}^{*(j)}_{\text{train}} \mathbf{\Phi}^{(j)}$, where $\tilde{\mathbf{X}}_{\text{red}} \in \mathbb{R}^{\lfloor \frac{m}{2} \rfloor \times k}$

6: *Univariate feature selection:* use [37]

7: **for** $i = 1$ **to** l **do**

8: *Estimate weight map:* $\hat{\omega}^{(i)}_{\text{red}} = \underset{\omega \in \mathbb{R}^p}{\operatorname{argmin}} \left\{ \mathcal{L}(\mathbf{y}^{*(j)}, \tilde{\mathbf{X}}_{\text{red}}; \omega) + \lambda_i \Omega(\omega) \right\}$, $\hat{\omega}^{(i)}_{\text{red}} \in \mathbb{R}^k$

9: **end for**

10: *Assing to* $\omega^{(j)}_{best}$ *the* $\hat{\omega}^{(i)}_{red}$, $i \in [l]$ *with the best performance on the test set*

11: *Return to voxel-space:* $\omega^{(j)}_{\text{approx}} = \hat{\omega}^{(j)}_{\text{best}} \mathbf{\Phi}^{\mathsf{T}(j)}$, where $\omega_{\text{approx}} \in \mathbb{R}^p$

12: **end for**

13: **return** $\hat{\omega}_{\text{bagg}} \leftarrow \frac{1}{b} \sum_{j=1}^{b} \omega^{(j)}_{\text{approx}}$

can improve the stability of the weight maps.

8 Validating ensembles of models: brain decoding

In the previous chapter, we have considered *ensembles of models* to improve the performance of decoders. In this chapter, we empirically validate this approach on several binary discrimination tasks. Additionally, we do an intensive benchmark of many decoders to analyze their stability.

We show that ensembles of models improve the stability of weight maps, while reducing the variability of the prediction accuracy. It also improves the small-sample recovery behavior, needing less data samples to find weight maps that are similar to the ones obtained using the whole dataset. When this scheme is combined with clustering, there is an additional gain in weight map stability. In terms of computation time, it is generally faster than state-of-the-art structured decoders. This scheme is easily parallelizable across the nested cross-validation loops, thus displaying additional speed ups. Finally, *ReNA* has shown to be well suited to be used with ensembles of models, as it is very beneficial for weight-map denoising while incurring small additional computation time.

The contributions developed in this chapter have been submitted to:

Stable brain decoding with ensembles of estimators. Andrés Hoyos-Idrobo, Gaël Varoquaux, Yannick Schwartz and Bertrand Thrion. *Neuroimage.*

8.1 Empirical studies: stable brain decoding

In this section, we conduct a series of experiments to highlight the practical aspects of model averaging in brain decoding. We use several MRI datasets to investigate their prediction performance, weight-map stability, and computation time.

Experiments on real neuroimaging data

To achieve reliable empirical conclusions, we consider a large number of different neuroimaging studies. We investigate model ensembles in several binary classification problems based on 8 fMRI datasets. We perform within-subject discrimination across sessions between various types of visual stimuli on the Haxby dataset [67]. In addition, we discriminate in an across subjects setting: *i)* different categories of visual stimuli from [40]; *ii)* conditions different levels of affective content with data from [163]; *iii)* mentalization with data from[107]; *iv)* famous, familiar, and scrambled faces from a visual-presentations dataset [71]; *v)* left and right saccades in data from [80]; *vi)* relational and emotion processing, language, and gambling protocols from the human connectome project (HCP)[46]; *vii)* response inhibition on openfMRI ds009 [133]. We use the trial-by-trial (Z-score) maps computed in a first-level GLM to perform all across-subject predictions. Additionally, we predict the gender from VBM maps using the OASIS dataset [98]. See section 1.5 for a more detailed description of these datasets.

Standard preprocessing and first-level analysis were applied using SPM. All MR data were variance-normalized and spatially smoothed at 6 mm FWHM for fMRI data and 2 mm FWHM for VBM data.

Experimental setup In all classification tasks, we use nested cross-validation for an accurate measure of the predictive power. We repeatedly split the data into a validation set and a decoding set. We choose validation sets of 20% the data, respecting the sample dependence structure (leaving out subjects or sessions). We set 10 folds for the outer cross validation loop.

As is standard practice in fMRI decoding [126], we use univariate feature selection on the training set to select 20% of voxels and train the decoder on the selected features. We compare several decoders, split into two groups:

- *Non-ensembles:* Graph-net [61], TV-ℓ_1 [103], SVM-ℓ_1, and SVM-ℓ_2.
- *Ensembles:* Ensembles of SVM-ℓ_1, SVM-ℓ_2, both, with and without clustering. These estimators are fitted using the proposed pipeline –see Algorithm 3.

We use scikit-learn [123] for the SVM with ℓ_1 and ℓ_2 penalties. We use nilearn [2] for Graph-net and TV-ℓ_1. When clustering is applied, we set the number k of clusters to 10% of the number of p voxels[1]. We rely on the fast agglomerative clustering presented in [74] –see chapter 5. Regarding the ensembles of models, we use 50% of decoding sets to train the decoder[2].

In the first experiment, we empirically validate the performance of various decoders on different discrimination tasks. In a second experiment, we explore the training speedup of decoders in a multi-core setting. Then we

[1] We consider a useful dimension reduction range, $k \in \left[\frac{p}{20}, \frac{p}{10}\right]$. This regime gives a good trade-off between computational efficiency and data fidelity [74].

[2] In the standard bootstrap the whole dataset is resampled. It can however be approximated with a subsampling of 50% of the data [144][134][39]

evaluate the similarity between weight maps obtained using all the data and the ones obtained for different sample sizes (small-sample recovery). Finally, we measure the stability of ensembles of models.

8.2 Results: evaluating the performance of decoders

Benchmarking decoders

For all discriminative conditions, we measure the prediction error on several left-out validation sets to assess the predictive power of the decoders. Additionally, we measure the correlation between the weight maps obtained in each cross-validation fold, and the computation time required to train the decoder. To perform this analysis, we split the datasets into two types: within-subject and across-subject. Throughout this experiment, we set the number b of estimators used in the ensembles of models to 50. This choice is discussed in Fig. 8.2.

Fig. 8.2 summarizes the relative performance with respect to the mean across decoders per discriminative task. In within-subject settings, all sparse methods have good prediction performance. Decoding using the standard SVM with both ℓ_1 and ℓ_2 penalty is fast, but the weight maps are less stable than the ones found by sparse structured methods –i.e. Graph-net and TV-ℓ_1. However, these complex penalties come with higher computation costs. As expected, using ensembles of models reduces the variance of the prediction, while increasing the stability of the weight maps. This effect is enhanced when including a clustering step. The computation time of ensembles of models with or without clustering is less than that of structured sparse classifiers.

For the discriminative task across subjects, ensembles of models consistently improve prediction accuracy as well as the stability of the weight maps, while keeping a computation cost less than structured sparse classifiers. In addition, the use of spatial clustering has a beneficial impact on the spatial stability. In all the presented cases, ensembles of models improve stability of the weight maps of the base estimator, while preserving the prediction accuracy. Note however that, for the combination of the SVM-ℓ_2 and clustering, it does not display any additional benefit.

Table 8.1 shows the comparison between each decoder and the decoder displaying the best prediction accuracy, namely ensembles of SVM-ℓ_1 for within-subject problems, and ensembles of SVM-ℓ_2 with clustering for across-subjects problems. These results confirm the above observations.

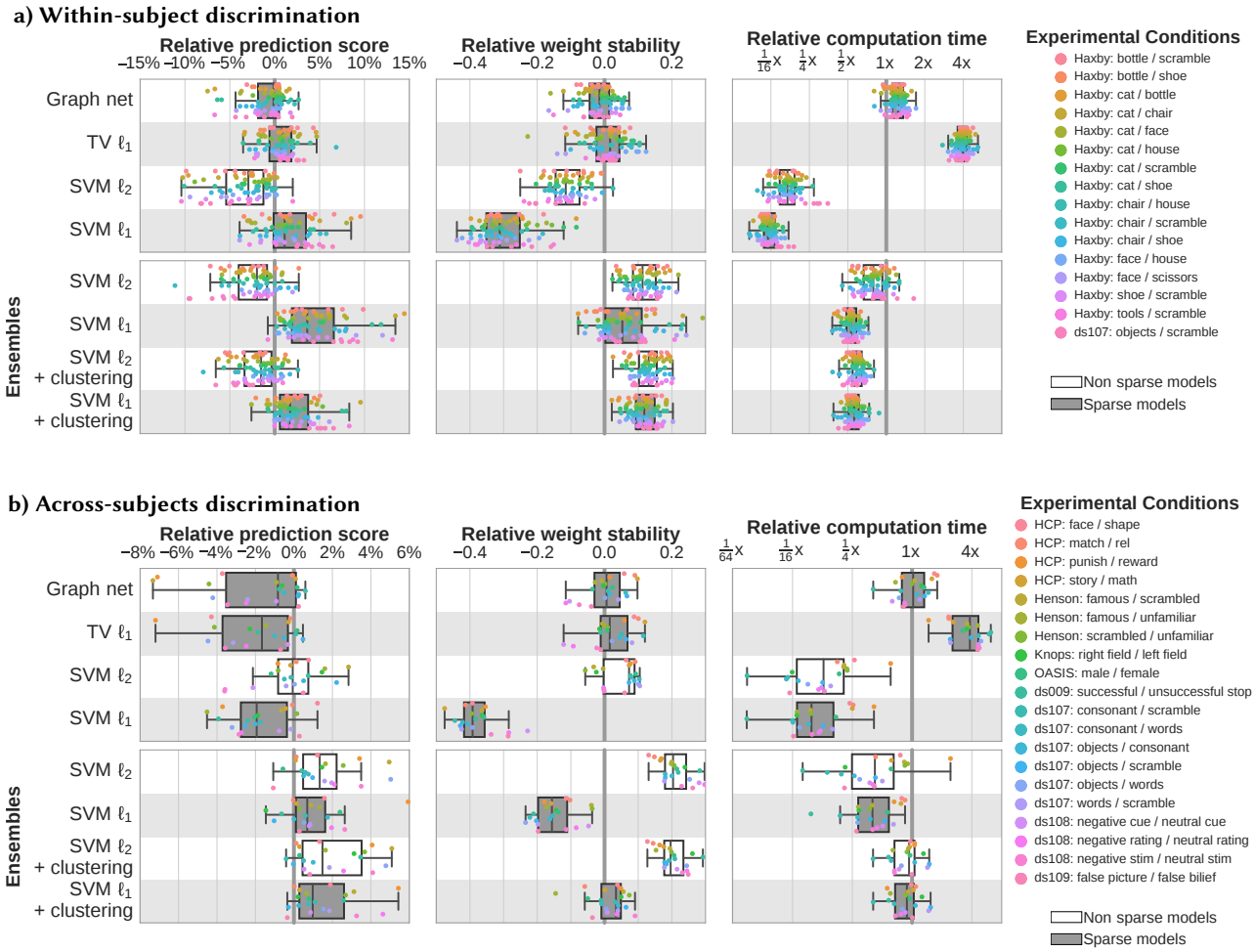


Figure 8.1: **Relative performance:** Relative prediction accuracy, weight stability and computation time for different classification tasks. Values are displayed relative to the mean over all the classifiers. *a)* ensembling of models improves prediction accuracy, and when applied with clustering it also reduces the variability. The ensembles of models consistently improves the weights stability, with a computation time smaller than TV-ℓ_1 and Graph-net. *b)* The ensembles of models with and without clustering slightly improve the prediction accuracy, while significantly improving the stability. Note that the computation time is obtained using a single CPU.

Delineating brain regions

An important question regarding brain decoders is whether they segment well the brain regions that support the decoding. The validation of this question is hard, yet there is evidence that relying on ensembles of models is a good approach [176][86]. Fig. 8.2 displays the decoder maps for the face-recognition tasks. For these tasks, we expect prediction to be driven by the functional areas of the visual cortex [60]. Indeed, the maps outline regions in known visual areas –e.g. the fusiform face area (FFA).

In both within-subject and across-subject datasets, the SVM-ℓ_1 maps are

a) Within-subject discrimination

	Classifier	prediction score	weight stability	computation time
	Graph-net	<**1e-10** (>)	**6.8e-9** (>)	<**1e-10** (<)
	TV-ℓ_1	<**1e-10** (>)	4.5e-5 (>)	<**1e-10** (<)
	SVM-ℓ_2	<**1e-10** (>)	<**1e-10** (>)	<**1e-10** (>)
	SVM-ℓ_1	**4.5e-8** (>)	<**1e-10** (>)	<**1e-10** (>)
Ensembles	SVM-ℓ_2	<**1e-10** (>)	**2.1e-6** (<)	(<)**1e-10** (<)
Ensembles	SVM-ℓ_1	**Reference**		
Ensembles	SVM-ℓ_2 + clustering	<**1e-10** (>)	**6.1e-9** (<)	4.7e-6 (<)
Ensembles	SVM-ℓ_1 + clustering	**5.8e-7** (>)	**5.3e-7** (<)	4.6e-3 (<)

b) Across-subjects discrimination

	Classifier	prediction score	weight stability	computation time
	Graph-net	5.2e-4 (>)	8.9e-5 (>)	0.12 (<)
	TV-ℓ_1	2.5e-4 (>)	8.9e-5 (>)	8.9e-5 (<)
	SVM-ℓ_2	7.3e-4 (>)	8.9e-5 (>)	8.9e-5 (>)
	SVM-ℓ_1	8.9e-5 (>)	8.9e-5 (>)	8.9e-5 (>)
Ensembles	SVM-ℓ_2	0.097 (>)	0.093 (<)	6.8e-4 (>)
Ensembles	SVM-ℓ_1	0.086 (>)	8.9e-5 (>)	8.9e-5 (>)
Ensembles	SVM-ℓ_2 + clustering	**Reference**		
Ensembles	SVM-ℓ_1 + clustering	0.28 (>)	8.9e-5 (>)	0.17 (>)

Table 8.1: **Comparison of performance:** Each decoder is compared with a reference. The values correspond to p-values obtained by paired Wilcoxon rank test. The direction in the parenthesis denotes the sign of the mean difference, and bold text denotes a significant results ($p < 10^{-6}$).

unstructured, and even if using ensembles of this model improves the stability of the weight maps, these maps remain scattered with a large number of small clusters. However, the use of clustering yields less and larger clusters, with maps that are qualitatively similar to TV-ℓ_1 maps. Graph-net and SVM-ℓ_2 display similar behavior, yielding various small clusters around large clusters of activation. In the case of SVM, the use of ensembles can reduce the number of small clusters. The combination with clustering enhances this effect. Note that setting the threshold to visualize regions is difficult task as the noise level is unknown.

Setting the number of estimators to ensemble

The choice of the number b of estimators to combine also affects the stability of the weight maps and the running time. Setting b corresponds to choosing the amount of spatial smoothing, and the computation time that one is willing to pay to train a decoder. We measure the performance of ensembles of classifiers on three different datasets and across 10 folds of cross-validation. In practice, we often use a number of estimators between 50 and 100, but to verify if the model converges, we consider here a range from 10 to 640 estimators.

Fig. 8.2 shows that for ensembles of classifiers, prediction accuracy does not depend on the number of estimators, whereas the computation time is almost linear ($t \propto b^{\gamma}$, where $\gamma \approx 1$). We use the running time as a constraint to finally set the number of estimators to 50, as the weight stability of non-sparse classifiers are at least 95% of the asymptotic optimum. In addition, this is a good compromise between stability and computation cost. Henceforth, this number of estimators is used throughout all experiments.

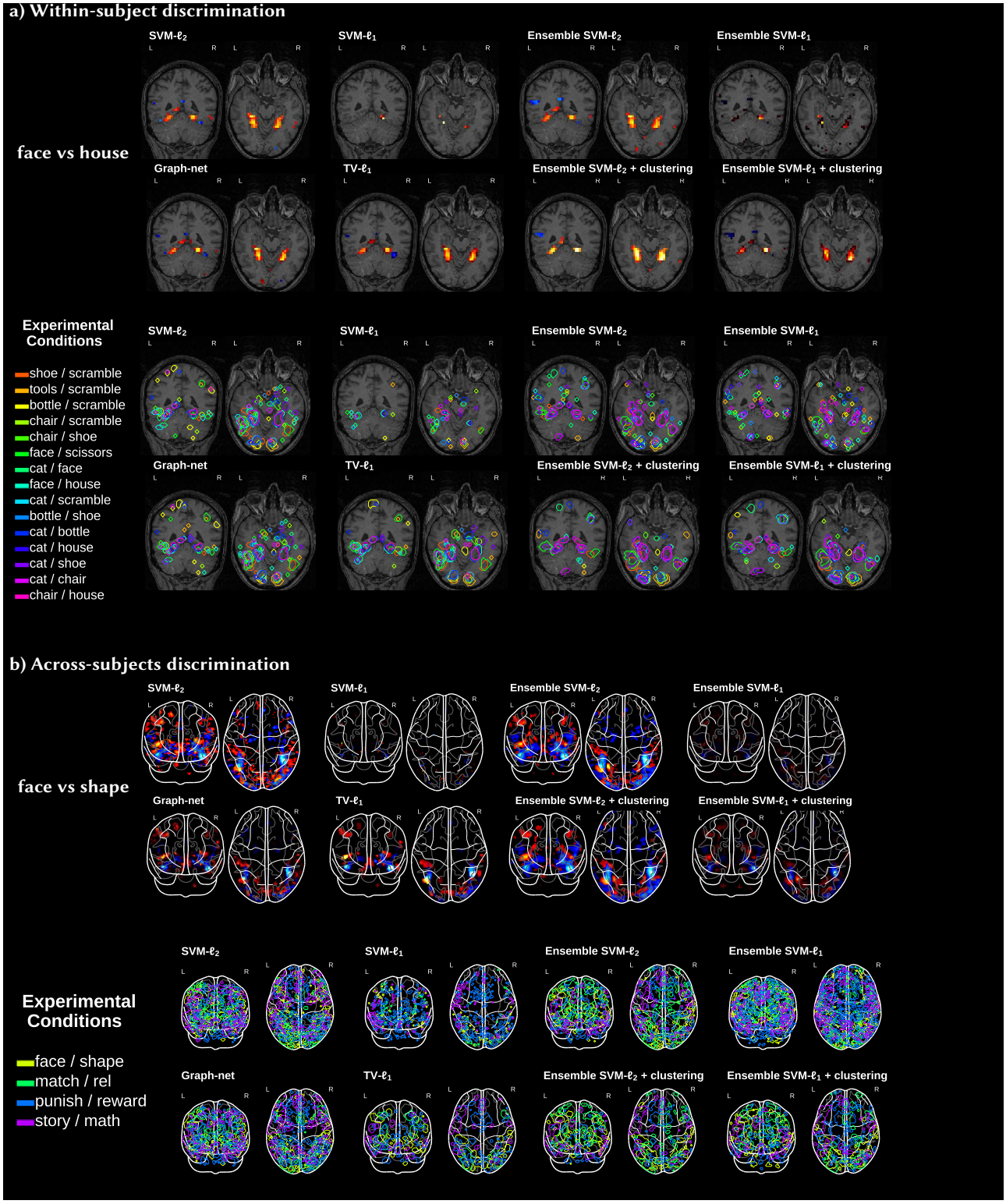


Figure 8.2: **Qualitative comparison of decoder weight maps:** Weight maps for different discriminative tasks on the Haxby and HCP datasets. The maps are thresholded at the 99 percentile for visualization purposes. In both dataset, *(top)* illustration of weight maps for the face-recognition task; *(bottom)* outlines of the other tasks.

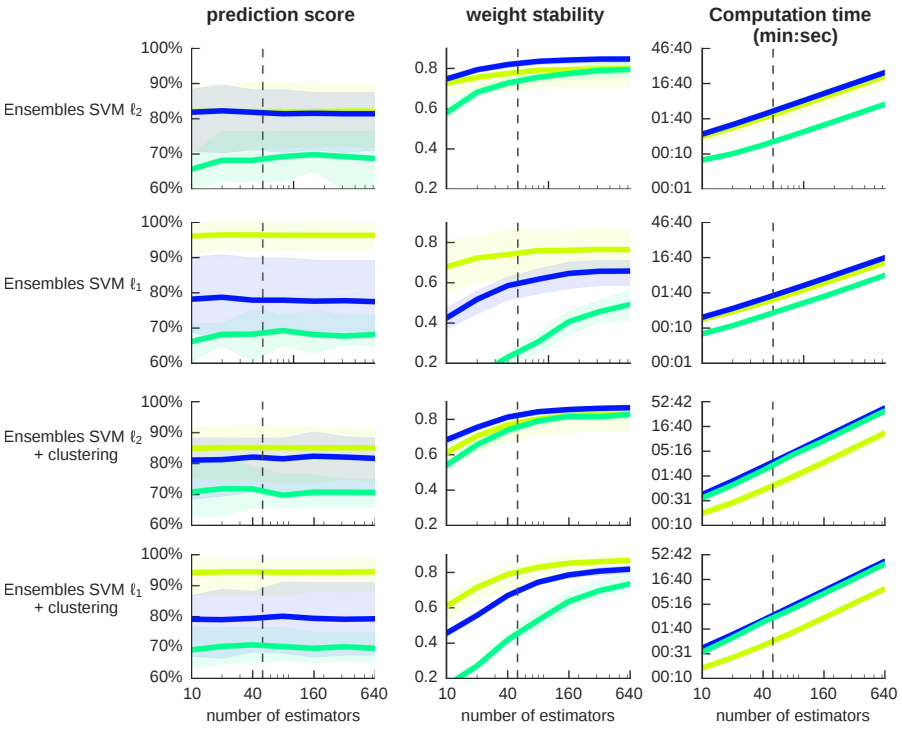


Figure 8.3: **Tuning curve of ensembles of models:** Quality of ensembles of classifiers as a function of the number of estimators. For each decoder, prediction accuracy is almost constant, hence it does not depend on the number of estimators. The use of clustering slightly improve the weight stability of the SVM-ℓ_2. In contrast, SVM-ℓ_1 consistently obtains higher stability when it is combined with clustering. Regarding computation time, the ensembles of models are almost linear in the number of estimators ($t \propto b^{\gamma}$, where $\gamma \approx 1$). Therefore, setting the number b of estimators depends on the computational resources available. The vertical dashed line denotes 50 estimators, which gives a good trade-off across performance metrics and datasets.

8.3 Results: parallel computing of brain decoders

One important feature of ensembling models is scalability, as these methods can be trained in parallel in a multi-core, shared-memory environment. This corresponds to current standard workstations, which frequently have a large number of CPUs. Here, we measure the training time of various decoders across 5 folds of cross-validation. We perform face-discrimination tasks on two datasets with different sizes.

Fig. 8.3 shows that, in general, there is not an ideal decrease in the computation time as more CPUs are added. The SVM with ℓ_1 and ℓ_2 penalty are the fastest. In contrast, TV-ℓ_1 is the slowest, followed by Graph-net. In both datasets, ensembles of models display most of the speed up at 10 CPUs, and reach a minimum at 20. These methods are much faster that Graph-net. The combination of ensembles of models with clustering does increase computation cost as we use a fast clustering algorithm.

8.4 Results: small-sample recovery behavior of decoders

In fMRI, despite growing efforts in data accumulation [133][46], the sample size remains small in comparison with the number of voxels. Therefore, an important aspect of the brain decoders is their sample complexity –i.e. the number of samples required to bound the estimation error. Yet, assessing the recovery of weight maps is difficult, as we do not have access to the asymptotic result. To bypass this problem, we measure the similarity between

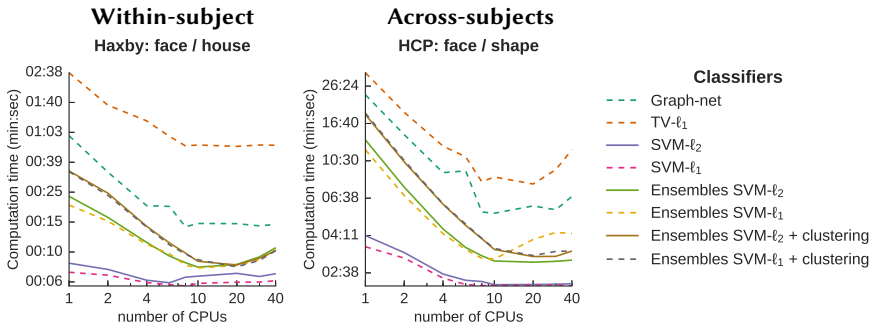


Figure 8.4: **Computation time of decoders:** Total wall clock averaged across 5-fold CV. In general, the speed-up in computation time is not ideal: it has a plateau. The fastest methods are the SVM with ℓ_1 and ℓ_2 penalty, followed by the ensembles of models, that display most of the speed-up at 10 CPUs; past this value, the computation time slowly reduces until finally reaching a minimum at 20 CPUs. In contrast, TV-ℓ_1 and Graph-net are consistently the slowest methods.

the weight maps obtained with different sample sizes and the ones obtained using the whole dataset. This can give us an intuition of the small-sample recovery behavior.

Fig. 8.4 shows that across datasets, the ensembles of ℓ_2 with and without clustering, and ensembles of SVM-ℓ_1 with clustering are consistently the best. In contrast, the SVM-ℓ_1 fails to recover the final weight maps. TV-ℓ_1, Graph-net, and SVM-ℓ_2 have a good performance on both datasets. Ensembles of SVM-ℓ_1 outperform these methods on the within-subject discrimination, as the weight similarity rapidly increases. On across-subject datasets, the ensembles of SVM-ℓ_1 with clustering has almost the same performance as TV-ℓ_1, Graph-net, and SVM-ℓ_2, differing only after using 80% of the data for training.

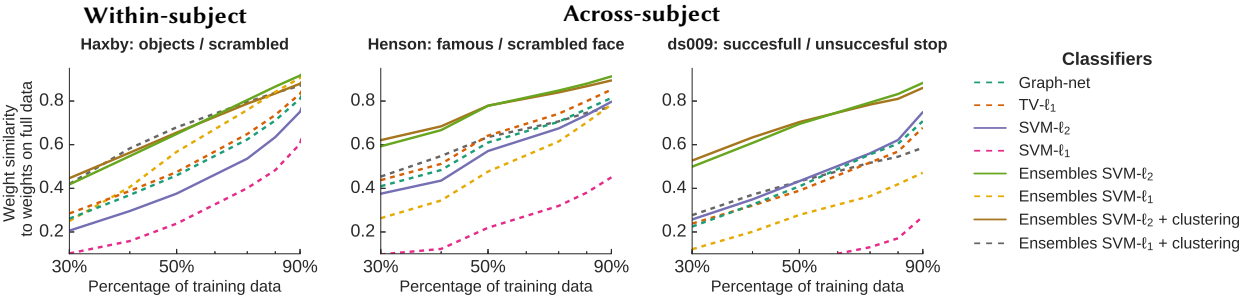


Figure 8.5: **Small-sample recovery behavior of decoders:** Evaluation of the correlation between decoder weight maps for each sample size and the ones obtained using the full dataset. Ensembles of SVM-ℓ_2 with and without clustering have consistently the best small-sample recovery performance, followed by TV-ℓ_1, Graph-net, and SVM-ℓ_2. *Within-subject)* ensembles of SVM-ℓ_1 rapidly increase their weights similarity, outperforming TV-ℓ_1 and Graph-net. *Across-subjects)* the combination of clustering and ensembles of SVM-ℓ_1 has a performance as good as TV-ℓ_1. In both discrimination tasks, SVM-ℓ_1 fails to recover the final decoder weight maps.

8.5 Results: assessing weight stability with ensembles of models

Reproducibility is a main concern in neuroimaging. At minimum, a necessary condition is the stability of the weight maps of the decoder. To assess this information, we use the ratio mean to standard deviation[3] of weight maps obtained during the nested cross-validation loop.

[3] This corresponds to a Z value if the data is centered. It also corresponds to an alternative definition of the Signal-to-Noise-Ratio (SNR), mainly used in image processing [139].

Fig. 8.5 shows maps of F values, the maps obtained with standard stability selection, and the maps obtained using our method. On both datasets, we can observe that F-tests highlight well-localized regions of the brain. Very similar regions are clearly outlined by the combination of clustering and ensembles of models, whereas the ensembles of SVM-ℓ_1 modules yield more spatially scattered points. The use of clustering reduces the noise, resulting in less variance.

8.6 Discussion: using ensembles of models

Decoding has become a central tool in neuroimage data processing. The high-dimensional nature of these data hinders reproducible results. However, at minimum, reproducibility manifests itself in stability of weight maps relative to data perturbations.

In this chapter, we have devised a strategy to train ensembles of models, which improves the stability of brain decoders. This scheme is summarized as follows: *i)* For each fold of the nested cross-validation loop, we select the estimator with the best predictive power; *ii)* we build an estimator by storing the models for all folds of cross-validation and averaging them. This approach differs from the stability selection methods [101][161], as here we use the amplitude of predictive weight maps for the model aggregation, and not only their support.

Using ensembles of models The predictive power of ensembles of modes is barely affected by the number of estimators used during the aggregation step. On the other hand, the stability of the resulting decoder improves as more estimators are used. On across-subjects datasets, the use of clustering improves the stability of sparse methods. In terms of computation time, this scheme displays an almost linear complexity in the number of estimators. Thus, setting the number of estimators is an arbitrary choice, it depends only on the computation resources available.

Comparing decoders In both within and across subjects datasets, ensembles of models have shown an improvement of the performance of the base estimator. This strategy reduces the variance of predictive power and increases the stability of weight maps of the base estimator. It also improves the small-sample behavior of the base estimators, boosting the consistency

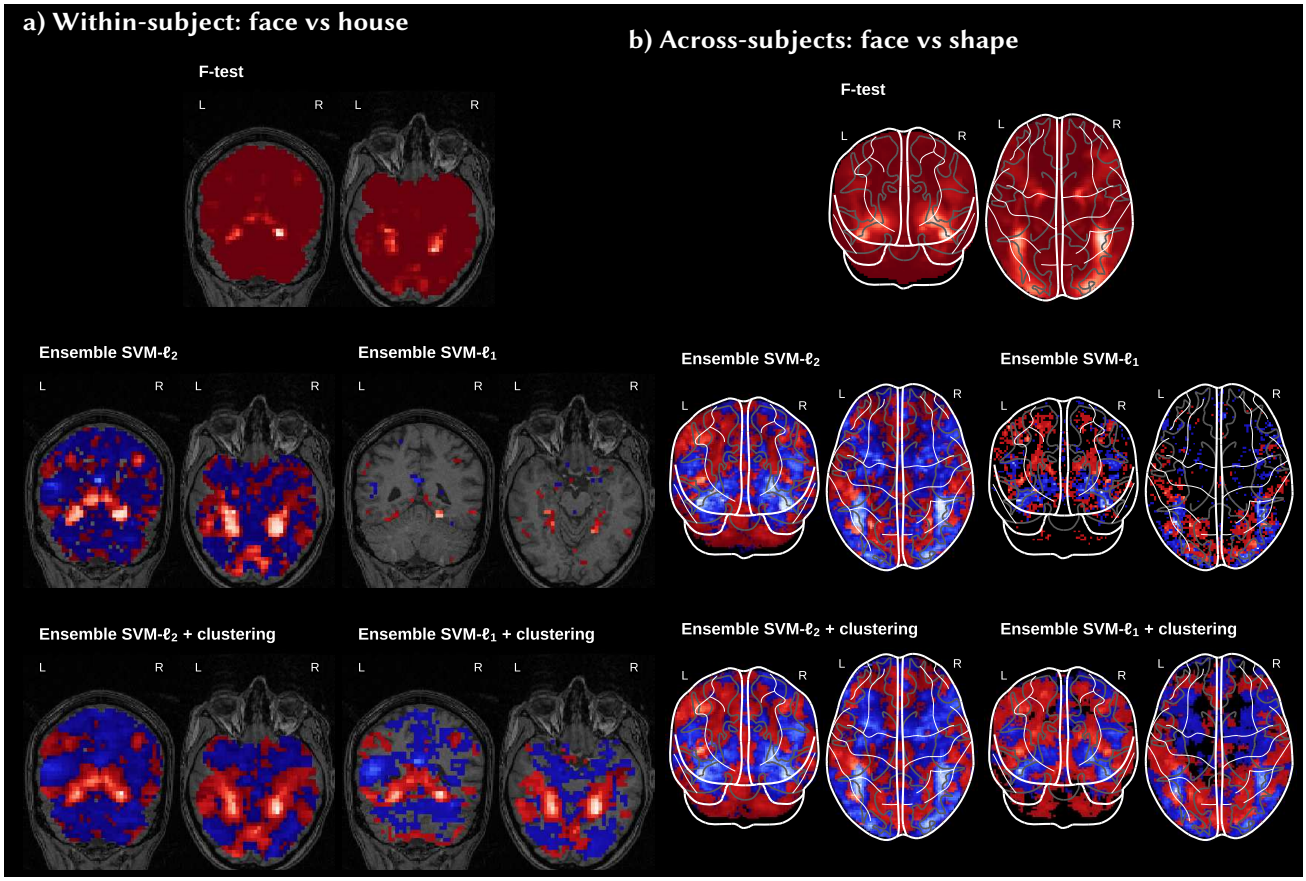


Figure 8.6: **Stability of weight maps of ensembles of models:** Evaluation of the squared ratio of the mean weight map and the standard deviation of weights obtained on the nested cross-validation. Higher values represent lower variability. On top, F-tests highlight well localized regions of the brain. In both datasets, ensembles of SVM-ℓ_2 with and without clustering, and ensembles of SVM-ℓ_1 with clustering localize similar regions to the F-tests. In contrast, ensembles of SVM-ℓ_1 modules fail to delineate brain regions. The combination of clustering and ensembles of methods decreases the variability and yields larger standardized effects. Note that these maps are unthresholded.

of weight maps. In addition, this scheme leads to qualitatively good brain regions delineation.

In terms of computation time, the use of ensembles of methods yields decoders that are slower than the base estimators. But they are faster than state-of-the-art decoders, namely TV-ℓ_1 and Graph-net. Nevertheless, the speed up of ensembles of methods can be enhanced by parallelizing the training of each estimators to aggregate. Thus, the training time is dominated by the fitting of each decoder. However, this gain is not ideal, and there is a plateau in the speed-up when the number of CPUs increases.

Regarding the combination of ensembles of methods and clustering, it has a spatial denosing effect on the resulting weight maps. This is reflected in the reduction of the variability and an increase in the prediction power. But,

when the base estimator is a sparse method, the averaging step reduces the sparsity, and yields weight maps with many small values instead.

Concluding remarks Our extensive empirical validation (36 decoding tasks, taken from 9 datasets) shows that the ensembles of methods, in particular SVM-ℓ_2 with clustering, give the best stability-prediction trade-off, with a good qualitative delineation of brain regions. The use of this scheme is a recommended practice in neuroimaging. Averaging several "good" estimators yields a model that can adapt to the properties of the noise present in the data. Hence, it is more robust to violations of modeling assumptions. The application of this scheme with clustering benefits to the spatial stability of weight maps, a key requirement of any cross-population study of functional imaging signals. In addition, the aggregation gives an estimate of the variance that can be used for post-hoc analysis of the weight maps.

9 Conclusions and perspectives

9.1 Contributions

Throughout this thesis, we have worked on scaling up brain decoders to handle larger datasets, while also improving their stability. We proposed a fast agglomerative clustering algorithm, and used it to perform dimension reduction by feature grouping. We relied on ensembles of linear models combined with clustering to yield a stable and easily parallelizable decoder. We validated this approach by extensive experiments on a large number of openly available neuroimaging datasets.

Feature grouping: We considered feature grouping as an alternative dimension-reduction scheme to matrix sketching, as it is well suited for signals with an underlying structure. This structure can be represented with a graph, e.g. the 3-dimensional grid underlying medical images. We have analyzed some properties of feature grouping as: *i)* how strongly it underestimates the energy of the signal *ii)* its ability to reduce independent noise. In both cases, we considered this behavior as a function of the regularity inside a cluster, and the cluster size.

Fast agglomerative clustering: We proposed a clustering algorithm that finds balanced clusters in linear time. This algorithm is convenient for feature grouping, as it scales linearly in the number of features. We investigated its performance in different machine-learning scenarios, such as: matrix decomposition, classification, regression, and compression. This algorithm has consistently shown a denoising behavior, performing as well as state-of-the art clustering methods –i.e. SLIC and Ward. In real datasets, we assessed denoising using the predictive power of an estimator, given that we do not have access to the ground truth. In addition, it can be used to train a decoder on a budget, needing less computation time to converge.

Brain decoding with ensembles of models: We propose the use of ensembles of models to find a more stable decoder. To train this estimator in a high-dimensional setting, we aggregate "good" models found during the nested cross-validation step. We conducted a series of experiments, where we

showed an improvement in the predictive power and weight-maps stability. This scheme also leads to better small-sample recovery of discriminative weight maps. Its computation is easily parallelizable, nevertheless its one-core implementation is faster than state-of-the-art decoders. Additionally, We showed that adding implicit spatial constraints yields an improvement in stability with respect state-of-the-art decoders, with a good visual qualitative quality of the weight maps of the decoder.

Experiment reproducibility: We believe that reproducibility of experiments is important. As such, all the developped code is based on and will be released[1] in the Nilearn [2] Python library. Nilearn provides an API to easily use machine learning algorithms with neuroimaging data.

[1] For the moment, the code can be found in:
`https://github.com/ahoyosid/ReNA`
`https://github.com/nilearn/nilearn/pull/`

9.2 Perspectives

Two-sample tests for high-dimensional structured signals: In high-dimensional settings, testing whether or not two samples of data come from the same distribution can be extremely challenging, as classical hypothesis testing methods can be ineffective, or not applicable at all. In [93] the authors used random projections to approximate the covariance matrix, performing a Hotelling test statistic with a reduced signal space. This approach is extended in [75], where the location shift between the two populations is expected to be related to a known graph structure. However, the authors used a permutation procedure to estimate the distribution of the number of false positive subgraphs, and this can be computationally expensive. This can be alleviated with the proposed fast dimension-reduction scheme for graph-structured signals: ReNA.

Building stable functional brain atlases: To understand the structure of the human cerebral cortex, an important step is to build a map that displays its major subdivisions, i.e. cortical areas. In [57] the authors used multiple functional neuroimaging modalities to build this map, but we consider that more data have to be taken into account, leveraging the information of other collaborative data sharing initiatives. However, when relying on data-driven approaches to build this map, one has to deal with important issues, such as: setting the optimal number of brain areas, labeling problems, unbalanced classes, and small-sample sizes –per class. The use of ontologies and structured cross-validation were proposed to overcome labeling issues [142]. Thus, ensembles of models combined with clustering can be used to: *i)* to make tractable training on large datasets; *ii)* improve small-sample prediction and stability. The resulting maps can be used to understand the relationship between brain organization and individual differences in behavior, as well as for clinical applications.

Going beyond task-based fMRI: The methods proposed in this thesis can be applied to resting-state fMRI data, where the correlation between brain regions is the main feature, i.e. functional connectivity: *i)* one can find brain regions in linear time: the proposed clustering algorithm (ReNA) can be used to build a data-driven atlas [1]. Using this implicit spatial constraint helps to deal with the mismatch between subjects; *ii)* one can use ensembles of models to improve the predictive power of a base estimator. In addition, this has a smoothing effect that produces continuous maps.

Faster training and becoming ambitious with scalability: It has been proven that the combination of clustering and randomization is an extremely effective tool in high-dimensional estimation problems, in particular when there is multicollinearity [32][161]. However, good spatial clustering algorithms are super-linear in computation time. Then these schemes are intractable in practice, given the number of resampling iterations needed to achieve stable results. In these settings, ReNA renders these methods feasible to practical implementations.

Feature grouping with ReNA can be easily plugged into semi-supervised methods [25], and online dictionary learning [102]. It can be used to improve the symbolic representation of time series [89], and when it is combined with ensembles of models it makes it possible to tackle scalability issues. This allows one to continue the idea presented in [142], and use even more datasets.

Open problem – Statistical analysis of weight maps: The weight maps of the decoder carry information about the probability of predictive brain regions conditioned to the target. However, the prediction of the target depends on all non-zero weights, and to visualize important regions one has to set an arbitrary threshold. Yet, the noise level of the weight maps remains unknown.

9.3 Concluding remarks

Decoding is a way to find predictive brain regions for different experimental conditions and datasets, but these datasets are now becoming huge and training in standard workstations is intractable. This thesis opens the door to finding more stable decoding results, while making it possible to use standard computer architectures. We have shown the denoising properties of model aggregation and feature grouping in neuroimaging settings. As this scheme yields more stable results, this approach can be useful for cognitive neuroscientists to find more insights about the functional specialization of the brain.

Bibliography

[1] A. Abraham, E. Dohmatob, B. Thirion, D. Samaras, and G. Varoquaux. Extracting brain regions from rest fMRI with total-variation constrained dictionary learning. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 607–615. Springer Berlin Heidelberg, 2013.

[2] A. Abraham, F. Pedregosa, M. Eickenberg, P. Gervais, A. Muller, J. Kossaifi, A. Gramfort, B. Thirion, and G. Varoquaux. Machine learning for neuroimaging with scikit-learn. *arXiv preprint arXiv:1412.3919*, 2014.

[3] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transaction on Pattern Analysis and Maching Intelligence*, 34:2274–2282, 2012.

[4] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66:671–687, 2003.

[5] T. Addair, D. Dodge, W. Walter, and S. Ruppert. Large-scale seismic signal analysis with hadoop. *Computers & Geosciences*, 66:145, 2014.

[6] N. Ailon and B. Chazelle. The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. *SIAM J. Comput.*, 39:302–322, 2009.

[7] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[8] F. Amat, E. W. Myers, and P. J. Keller. Fast and robust optical flow for time-lapse microscopy using super-voxels. *Bioinformatics*, 29:373–80, 2013.

[9] S. Arlot and A. Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010.

[10] D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *Annual Symposium on Computational Geometry*, page 144, 2006.

[11] J. Ashburner, G. Barnes, C. Chen, J. Daunizeau, G. Flandin, K. Friston, D. Gitelman, S. Kiebel, J. Kilner, V. Litvak, R. Moran, W. Penny, K. Stephan, D. Gitelman, R. Henson, C. Hutton, V. Glauche, J. Mattoutee, and C. Phillips. Spm8 manual. *Functional Imaging Laboratory, Institute of Neurology*, page 41, 2008.

[12] F. Bach. Bolasso: Model consistent Lasso estimation through the bootstrap. *International Conference on Machine Learning*, 2008.

[13] S. Badillo, T. Vincent, and P. Ciuciu. Group-level impacts of within- and between-subject hemodynamic variability in fMRI. *NeuroImage*, 82:433–448, 2013.

[14] R. G. Baraniuk and M. B. Wakin. Random projections of smooth manifolds. *Found. Comput. Math.*, 9:51–77, 2009.

[15] D. M. Barch, G. C. Burgess, M. P. Harms, S. E. Petersen, B. L. Schlaggar, M. Corbetta, M. F. Glasser, S. Curtiss, S. Dixit, C. Feldt, D. Nolan, E. Bryant, T. Hartley, O. Footer, J. M. Bjork, R. Poldrack, S. Smith, H. Johansen-Berg, A. Z. Snyder, D. C. V. Essen, and W. U.-M. H. Consortium. Function in the human connectome: task-fMRI and individual differences in behavior. *Neuroimage*, 80:169–189, 2013.

[16] R. Basri and D. Jacobs. Lambertian reflection and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:218–233, 2003.

[17] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57:289–300, 1995.

[18] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[19] G. M. Boynton, S. A. Engel, G. H. Glover, and D. J. Heeger. Linear systems analysis of functional magnetic resonance imaging in human V1. *The Journal of Neuroscience*, 16:4207–422, 1996.

[20] L. Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

[21] P. Bühlmann and B. Yu. Analyzing bagging. *Annals of Statistics*, 30: 927–961, 2002.

[22] P. Bühlmann, P. Rütimann, S. van de Geer, and C.-H. Zhang. Correlated variables in regression: clustering and sparse estimation. *Journal of Statistical Planning and Inference*, 143:1835–1871, 2013.

[23] R. B. Buxton, E. C. Wong, and L. R. Frank. Dynamics of blood flow and oxygenation changes during brain activation: The balloon model. *Magnetic resonance in medicine*, 39:855–864, 1998.

[24] R. B. Buxton, K. Uludağ, D. J. Dubowitz, and T. T. Liu. Modeling the hemodynamic response to brain activation. *Neuroimage*, 23:S220–S233, 2004.

[25] D. Bzdok, M. Eickenberg, O. Grisel, B. Thirion, and G. Varoquaux. Semi-supervised factored logistic regression for high-dimensional neuroimaging data. In *Advances in neural information processing systems*, pages 3348–3356, 2015.

[26] A. Caramazza and M. Coltheart. Cognitive neuropsychology twenty years on. *Cognitive Neuropsychology*, 23:3–12, 2006.

[27] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Trans. Database Syst.*, 27:188–228, 2002.

[28] M. S. Cohen. Parametric analysis of fMRI data using linear systems methods. *NeuroImage*, 6:93–103, 1997.

[29] J. R. Cole, Q. Wang, J. A. Fish, B. Chai, D. M. McGarrell, Y. Sun, C. T. Brown, A. Porras-Alfaro, C. R. Kuske, and J. M. Tiedje. Ribosomal database project: data and tools for high throughput rRNA analysis. *Nucleic Acids Research*, 2013.

[30] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[31] D. D. Cox and R. L. Savoy. Functional magnetic resonance imaging (fMRI) "brain reading": detecting and classifying distributed patterns of fMRI activity in human visual cortex. *NeuroImage*, 19:261–270, 2003.

[32] B. Da Mota, V. Fritsch, G. Varoquaux, T. Banaschewski, G. J. Barker, A. L. Bokde, U. Bromberg, P. Conrod, J. Gallinat, H. Garavan, et al. Randomized parcellation based inference. *NeuroImage*, 89:203–215, 2014.

[33] A. Dehman, C. Ambroise, and P. Neuvial. Performance of a blockwise approach in variable selection using linkage disequilibrium information. *BMC bioinformatics*, 16:1, 2015.

[34] O. Demirci, V. P. Clark, V. A. Magnotta, N. C. Andreasen, J. Lauriello, K. A. Kiehl, G. D. Pearlson, and V. D. Calhoun. A review of challenges in the use of fMRI for disease classification/characterization and a projection pursuit application from a multi-site fMRI schizophrenia study. *Brain imaging and behavior*, 2:207–226, 2008.

[35] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7:1–30, 2006.

[36] J. E. Desmond and G. H. Glover. Estimating sample size in functional MRI (fMRI) neuroimaging studies: Statistical power analyses. *Journal of Neuroscience Methods*, 118:115–128, 2002.

[37] E. Dohmatob, M. Eickenberg, B. Thirion, and G. Varoquaux. Speeding-up model-selection in GraphNet via early-stopping and univariate feature-screening. pages 17–20, 2015.

[38] E. D. Dohmatob, A. Gramfort, B. Thirion, and G. Varoquaux. Benchmarking solvers for TV-ℓ_1 least-squares and logistic regression in brain imaging. *IEEE PRNI*, 2014.

[39] L. Dümbgen, R. J. Samworth, and D. Schuhmacher. Stochastic search for semiparametric linear regression models. *From Probability to Statistics and Back: High-Dimensional Models and Processes – A Festschrift in Honor of Jon A. Wellner*, 9:78–90, 2013.

[40] K. Duncan, C. Pattamadilok, I. Knierim, and J. Devlin. Consistency and variability in functional localisers. *Neuroimage*, 46:1018–1026, 2009.

[41] K. J. Duncan, C. Pattamadilok, I. Knierim, and J. T. Devlin. Consistency and variability in functional localisers. *Neuroimage*, 46:1018, 2009.

[42] B. Efron and R. Tibshirani. Improvements on cross-validation: the .632+ bootstrap method. *J. Amer. Statist. Assoc*, 92:548–560, 1997.

[43] M. Eickenberg, E. Dohmatob, B. Thirion, and G. Varoquaux. Total variation meets sparsity: statistical learning with segmenting penalties. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer Berlin Heidelberg, 2015.

[44] T. E.Nichols and A. P.Holmes. Nonparametric permutation tests for functional neuroimaging: A primer with examples. *Human brain mapping*, 15:1–25, 2001.

[45] D. Eppstein, M. S. Paterson, and F. Yao. On nearest-neighbor graphs. *Discrete and Computational Geometry*, 17:263–282, 1997.

[46] D. V. Essen, K. Ugurbil, E. Auerbach, D. Barch, T. Behrens, R. Bucholz, A. Chang, L. Chen, M. Corbetta, S. Curtiss, S. D. Penna, D. Feinberg, M. Glasser, N. Harel, A. Heath, L. Larson-Prior, D. Marcus, G. Michalareas, S. Moeller, R. Oostenveld, S. Petersen, F. Prior, B. Schlaggar, S. Smith, A. Snyder, J. Xu, and E. Yacoub. The human connectome project: A data acquisition perspective. *NeuroImage*, 62: 2222–2231, 2012.

[47] M. Ester, H. peter Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[48] Y. Fan, N. Batmanghelich, C. M. Clark, and C. Davatzikos. Spatial patterns of brain atrophy in MCI patients, identified via high-dimensional pattern classification, predict subsequent cognitive decline. *NeuroImage*, 39:1731–1743, 2008.

[49] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59:167–181, 2004.

[50] M. D. Fox, A. Z. Snyder, J. L. Vincent, M. Corbetta, D. C. V. Essen, and M. E. Raichle. The human brain is intrinsically organized into dynamic, anticorrelated functional networks. *Proceedings of the National Academy of Sciences*, pages 9673–9678, 2005.

[51] K. J. Friston, A. P. Holmes, K. J. Worsley, J.-P. Poline, C. D. Frith, and R. S. Frackowiak. Statistical parametric maps in functional imaging: a general linear approach. *Human brain mapping*, 2:189–210, 1993.

[52] K. J. Friston, A. P. Holmes, K. J. Worsley, J.-B. Poline, C. D. Frith, and R. S. Frackowiak. Statistical parametric maps in functional imaging: a general linear approach. *Human Brain Mapping*, 2:189–210, 1994.

[53] B. Gaonkar and C. Davatzikos. Analytic estimation of statistical significance maps for support vector machine based multivariate image analysis and classification. *Neuroimage*, 78:270–283, 2013.

[54] C. R. Genovese, N. A. Lazar, and T. Nichols. Thresholding of statistical maps in functional neuroimaging using the false discovery rate. *Neuroimage*, 4:870–878, 2002.

[55] A. Gittens and M. W. Mahoney. Revisiting the nystrom method for improved large-scale machine learning. In *International Conference on Machine Learning*, pages 567–575, 2013.

[56] M. F. Glasser, S. N. Sotiropoulos, J. A. Wilson, T. S. Coalson, B. Fischl, J. L. Andersson, J. Xu, S. Jbabdi, M. Webster, and J. R. Polimeni. The minimal preprocessing pipelines for the human connectome project. *Neuroimage*, 80:105–124, 2013.

[57] M. F. Glasser, T. S. Coalson, E. C. Robinson, C. D. Hacker, J. Harwell, E. Yacoub, K. Ugurbil, J. Andersson, C. F. Beckmann, M. Jenkinson, S. M. Smith, and D. C. V. Essen. A multi-modal parcellation of human cerebral cortex. *Nature*, 536:171–178, 2016.

[58] G. H. Glover. Deconvolution of impulse response in event-related BOLD fMRI. *NeuroImage*, 9:416–429, 1999.

[59] K. J. Gorgolewski, G. Varoquaux, G. Rivera, Y. Schwartz, S. S. Ghosh, C. Maumet, T. E. Nichols, R. A. Poldrack, J.-B. Poline, T. Yarkoni, and D. S. Margulies. Neurovault.org: A web-based repository for collecting

and sharing unthresholded statistical maps of the human brain. *Front. Neuroinform.*, 2015.

[60] K. Grill-Spector and R. Malach. The human visual cortex. *Annu. Rev. Neurosci.*, 27:649–677, 2004.

[61] L. Grosenick, B. Klingenberg, K. Katovich, B. Knutson, and J. E. Taylor. Interpretable whole-brain prediction analysis with GraphNet. *Neuroimage*, 2013.

[62] D. A. Handwerker, J. M. Ollinger, and M. D'Esposito. Variation of BOLD hemodynamic responses across subjects and brain regions and their effects on statistical analyses. *NeuroImage*, 21:1639–1651, 2004.

[63] M. Hanke, Y. O. Halchenko, P. B. Sederberg, S. J. Hanson, J. V. Haxby, and S. Pollmann. PyMVPA: a python toolbox for multivariate pattern analysis of fMRI data. *Neuroinformatics*, 7:37–53, 2009.

[64] T. Hastie, R. Tibshirani, M. B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W. C. Chan, D. Botstein, and P. Brown. 'gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, 1, 2000.

[65] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2 edition, 2009.

[66] S. Haufe, F. Meinecke, K. Görgen, S. Dähne, J.-D. Haynes, B. Blankertz, and F. Bießmann. On the interpretation of weight vectors of linear models in multivariate neuroimaging. *NeuroImage*, 87:96–110, 2014.

[67] J. V. Haxby, M. I. Gobbini, M. L. Furey, A. Ishai, J. L. Schouten, and P. Pietrini. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293:2425–2430, 2001.

[68] J.-D. Haynes and G. Rees. Decoding mental states from brain activity in humans. *Nat Rev Neurosci*, 7:523–534, 2006.

[69] C. Hedge, A. C. Sankaranarayanan, W. Yin, and R. G. Baraniuk. NuMax: a convex approach for learning near-isometric linear embeddings. *IEEE Transactions on Signal Processing*, 63:6109–6121, 2015.

[70] R. Henson. Forward inference using functional neuroimaging: Dissociations versus associations. *Trends in cognitive sciences*, 10:64–69, 2006.

[71] R. Henson, T. Shallice, M. Gorno-Tempini, and R. Dolan. Face repetition effects in implicit and explicit memory tests as measured by fMRI. *Cerebral Cortex*, 12:178, 2002.

[72] F. G. Hillary and J. DeLuca. *functional neuroimaging in clinical populations*. Guilford Press, 2007.

[73] A. Hoyos-Idrobo, Y. Schwartz, G. Varoquaux, and B. Thirion. Improving sparse recovery on structured images with bagged clustering. *IEEE PRNI*, 2015.

[74] A. Hoyos-Idrobo, G. Varoquaux, J. Kahn, and B. Thirion. Recursive nearest agglomeration (ReNA): fast clustering for approximation of structured signals. *arXiv:1609.04608*, 2016.

[75] L. Jacob, P. Neuvial, and S. Dudoit. More power via graph-structured tests for differential expression of gene networks. *The Annals of Applied Statistics*, 6:561–600, 2012.

[76] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87:316–336, 2010.

[77] W. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability*, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society, 1984.

[78] H. G. Katzgraber. Random numbers in scientific computing: An introduction. *CoRR*, abs/1005.4117, 2010.

[79] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3:263, 2001.

[80] A. Knops, B. Thirion, E. M. Hubbard, V. Michel, and S. Dehaene. Recruitment of an area involved in eye movements during mental arithmetic. *Science*, 324:1583, 2009.

[81] L. I. Kuncheva and J. J. Rodríguez. Classifier ensembles for fMRI data analysis: an experiment. *Magnetic Resonance Imaging*, 28:583–593, 2010.

[82] L. I. Kuncheva, J. J. Rodriguez, C. O. Plumpton, D. E. J. Linden, and S. J. Johnston. Random subspace ensembles for fMRI classification. *IEEE Transactions on Medical Imaging*, 29:531–542, 2010.

[83] L. Le Folgoc. *Apprentissage statistique pour la personnalisation de modèles cardiaques à partir de données d'imagerie*. PhD thesis, Nice, 2015.

[84] L. Le Magoarou and R. Gribonval. Flexible multi-layer sparse approximations of matrices and applications. *arXiv preprint arXiv:1506.07300*, 2015.

[85] S. Lemm, B. Blankertz, T. Dickhaus, and K.-R. Müller. Introduction to machine learning for brain imaging. *NeuroImage*, 56:387–399, 2011.

[86] G. Leung and A. R. Barron. Information theory and mixing least-squares regressions. *IEEE Transactions on information theory*, 52, 2006.

[87] E. Liberty. Simple and deterministic matrix sketching. In *SIGKDD*, pages 581–588. ACM, 2013.

[88] C. Lim and B. Yu. Estimation stability with cross-validation (ESCV). *Journal of Computational and Graphical Statistics*, 25:464–492, 2016.

[89] J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, pages 107–144, 2007.

[90] M. A. Lindquist. The statistical analysis of fMRI data. *Statistical Science*, 23:439–464, 2008.

[91] N. K. Logothetis, J. Pauls, M. Augath, T. Trinath, and A. Oeltermann. Neurophysiological investigation of the basis of the fMRI signal. *Nature*, 412:150–157, 2001.

[92] H. Lombaert, A. Criminisi, and N. Ayache. Spectral forests: Learning of surface data, application to cortical parcellation. In *MICCAI*, pages 547–555, 2015.

[93] M. E. Lopes, L. J. Jacob, and M. J. Wainwright. A more powerful two-sample test in high dimensions using random projection. In *Advances in Neural Information Processing Systems*, 2011.

[94] Y. Lu, P. S. Dhillon, D. P. Foster, and L. H. Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in neural information processing systems*, pages 369–377, 2013.

[95] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua. Supervoxel-based segmentation of mitochondria in em image stacks with learned shape features. *IEEE transactions on medical imaging*, 31:474–486, 2012.

[96] M. W. Mahoney. Randomized algorithms for matrices and data. *Found. Trends Mach. Learn.*, 3:123–224, 2011.

[97] M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106:697–702, 2009.

[98] D. S. Marcus, T. H. Wang, J. Parker, et al. Open access series of imaging studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults. *J Cogn Neurosci*, 19: 1498, 2007.

[99] J. Margeta, A. Criminisi, R. C. Lozoya, D. Lee, and N. Ayache. Fine-tuned convolutional neural nets for cardiac MRI acquisition plane recognition. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, pages 1–11, 2015.

[100] A. Mechelli, C. J. Price, K. J. Friston, and J. Ashburner. Voxel-based morphometry of the human brain: Methods and applications. *Current Medical Imaging Reviews*, 1, 2005.

[101] N. Meinshausen and P. Bühlmann. Stability selection. *J Roy Stat Soc B*, 72:417–473, 2010.

[102] A. Mensch, J. Marial, B. Thirion, and G. Varoquaux. Dictionary learning for massive matrix factorization. In *International conference on machine learning*, 2016.

[103] V. Michel, A. Gramfort, G. Varoquaux, E. Eger, and B. Thirion. Total variation regularization for fMRI-based prediction of behavior. *IEEE Transactions on Medical Imaging*, 30:1328–1340, 2011.

[104] M. Misaki, Y. Kim, P. A. Bandettini, and N. Kriegeskorte. Comparison of multivariate classifiers and response normalizations for pattern-information fMRI. *NeuroImage*, 53:103–118, 2010.

[105] T. M. Mitchell, R. Hutchinson, R. S. Niculescu, F. Pereira, X. Wang, M. Just, and S. Newman. Learning to decode cognitive states from brain images. *Machine Learning*, 57:145, 2004.

[106] H. Mohr, U. Wolfensteller, S. Frimmel, and H. Ruge. Sparse regularization techniques provide novel insights into outcome integration processes. *Neuroimage*, 104, 2015.

[107] J. M. Moran, E. Jolly, and J. P. Mitchell. Social-cognitive deficits in normal aging. *J. Neurosci*, 32:5553, 2012.

[108] J. Mourão-Miranda, A. L. Bokde, C. Born, H. Hampel, and M. Stetter. Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional MRI data. *NeuroImage*, 28:980–995, 2005.

[109] J. Mourão-Miranda, A. L. Bokde, C. Born, H. Hampel, and M. Stetter. Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional MRI data. *NeuroImage*, 28, 2005.

[110] D. Müllner. Modern hierarchical, agglomerative clustering algorithms. *ArXiv e-prints*, 2011.

[111] T. Naselaris, K. N. Kay, S. Nishimoto, and J. L. Gallant. Encoding and decoding in fMRI. *Neuroimage*, 56:400–410, 2011.

[112] A. Nemirovski. Topics in non-parametric statistics. *Lectures on Probability Theory and Statistics: Ecole d'Ete de Probabilites de Saint-Flour XXVIII-1998*, 28:85, 2000.

[113] T. Nichols and S. Hayasa. Controlling the familywise error rate in functional neuroimaging: a comparative review. *Statistical Methods in Medical Research*, 12:419–446, 2003.

[114] T. E. Nichols, S. Das, S. B. Eickhoff, A. C. Evans, T. Glatard, M. Hanke, N. Kriegeskorte, M. P. Milham, R. A. Poldrack, J.-B. Poline, E. Proal, B. Thirion, D. C. V. Essen, T. White, and B. T. T. Yeo. Best practices in data analysis and sharing in neuroimaging using MRI. *BioRxiv: 054262*, 2016.

[115] K. A. Norman, S. M. Polyn, G. J. Detre, and J. V. Haxby. Beyond mind-reading: multi-voxel pattern analysis of fMRI data. *Trends in Cognitive Sciences*, 10, 2006.

[116] S. E. O'Bryant, G. Xiao, R. Barber, J. Reisch, R. Doody, T. Fairchild, P. Adams, S. Waring, and R. Diaz-Arrastia. A serum protein-based algorithm for the detection of Alzheimer's disease. *Archives of neurology*, 67, 2010.

[117] S. Ogawa, T. M. Lee, A. R. Kay, and D. W. Tank. Brain magnetic resonance imaging with contrast dependent on blood oxygenation. *Proceedings of the National Academy of Sciences*, 87:9868–9872, 1990.

[118] S. Ogawa, D. Tank, R. Menon, J. Ellermann, S. Kim, H. Merkle, and K. Ugurbil. Intrinsic signal changes accompanying sensory stimulation: functional brain mapping with magnetic resonance imaging. *Proceedings of the National Academy of Sciences*, 89:5951–5955, 1992.

[119] A. J. O'Toole, F. Jiang, H. Abdi, N. Pénard, J. P. Dunlop, and M. A. Parent. Theoretical, statistical, and practical perspectives on pattern-based classification approaches to the analysis of functional neuroimaging data. *J Cogn Neurosci*, 19, 2007.

[120] R. Overbeek, N. Larsen, G. D. Pusch, M. D'Souza, E. S. Jr, N. Kyrpides, M. Fonstein, N. Maltsev, and E. Selkov. WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Nucleic Acids Research*, 28:123–125, 2000.

[121] D. J. Pearce. An improved algorithm for finding the strongly connected components of a directed graph. Technical report, 2005.

[122] F. Pedregosa. *Feature extraction and supervised learning on fMRI : from practice to theory*. PhD thesis, 2015.

[123] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825, 2011.

[124] M. V. Peelen and P. E. Downing. Using multi-voxel pattern analysis of fMRI data to interpret overlapping functional activations. *Trends in cognitive sciences*, 11, 2007.

[125] M. Penrose. Single linkage clustering and continuum percolation. *Journal of Multivariate Analysis*, 53:94–109, 1995.

[126] F. Pereira, T. Mitchell, and M. Botvinick. Machine learning classifiers and fMRI: A tutorial overview. *NeuroImage*, 45:S199–S209, 2009.

[127] R. Poldrack. Can cognitive processes be inferred from neuroimaging data? *Trends in cognitive sciences*, 10:59–63, 2006.

[128] R. A. Poldrack. Inferring mental states from neuroimaging data: from reverse inference to large-scale decoding. *Neuron*, 72:692–697, 2011.

[129] R. A. Poldrack and K. J. Gorgolewski. Making big data open: data sharing in neuroimaging. *Nature Neuroscience*, 17:1510–1517, 2014.

[130] R. A. Poldrack and K. J. Gorgolewski. OpenfMRI: Open sharing of task fMRI data. *NeuroImage*, 2015.

[131] R. A. Poldrack and T. Yarkoni. From brain maps to cognitive ontologies: Informatics and the search for mental structure.

[132] R. A. Poldrack, J. A. Mumford, and T. E. Nichols. *Handbook of functional MRI data analysis*. Cambridge University Press, 2011.

[133] R. A. Poldrack, D. M. Barch, J. P. Mitchell, T. D. Wager, A. D. Wagner, J. T. Devlin, C. Cumba, O. Koyejo, and M. P. Milham. Toward open sharing of task-based fMRI data: the OpenfMRI project. *Frontiers in Neuroinformatics*, 7, 2013.

[134] J. Praestgaard and J. A. Wellner. Exchangeably weighted bootstraps of the general empirical process. *The Annals of Probability*, 21:2053–2086, 1993.

[135] M. Rahim, B. Thirion, A. Abraham, M. Eickenberg, E. Dohmatob, C. Comtat, and G. Varoquaux. Integrating multimodal priors in predictive models for the functional characterization of Alzheimer's disease. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 207–214. Springer International Publishing, 2015.

[136] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, 2007.

[137] A. Rinaldo, S.-A. Bacanu, B. Devlin, V. Sonpar, L. Wasserman, and K. Roeder. Characterization of multilocus linkage disequilibrium. *Genetic epidemiology*, 28:193, 2005.

[138] A. Rudi, R. Camoriano, and L. Rosasco. Less is more: Nyström computational regularization. In *Advances in neural information processing systems*, pages 1648–1656, 2015.

[139] J. C. Russ. *The image processing handbook*. CRC Press, 2007.

[140] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115:211–252, 2015.

[141] J. Schrouff, C. Kussé, L. Wehenkel, P. Maquet, and C. Phillips. Decoding semi-constrained brain activity from fmri using support vector machines and gaussian processes. *PLoS one*, 7(4):e35860, 2012.

[142] Y. Schwartz, B. Thirion, and G. Varoquaux. Mapping cognitive ontologies to and from the brain. In *Advances in neural information processing systems*, 2013.

[143] G. E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[144] R. D. Shah and R. J. Samworth. Variable selection with error control: another look at stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75:55–80, 2013.

[145] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University press, 2014.

[146] R. Shibata. Bootstrap estimate of kullback-leibler information for model selection. *Statistica Sinica*, 7:375–394, 1997.

[147] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs. *IEEE Signal processing magazine*, 83, 2013.

[148] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *Signal Processing Magazine, IEEE*, 30:83–98, 2013.

[149] S. M. Smith, C. F. Beckmann, J. Andersson, E. J. Auerbach, J. Bijsterbosch, G. Douaud, E. Duff, D. A. Feinberg, L. Griffanti, M. P. Harms, M. Kelly, T. Laumann, K. L. Miller, S. Moeller, S. Petersen, J. Power, G. Salimi-Khorshidi, A. Z. Snyder, A. T. Vu, M. W. Woolrich, J. Xu, E. Yacoub, K. Uğurbil, D. C. V. Essen, and M. F. Glasser. Resting-state fMRI in the human connectome project. *NeuroImage*, 80:144–168, 2013.

[150] D. Stauffer and A. Aharony. *Introduction to Percolation Theory*. Oxford University Press, New York, 1971.

[151] S.-H. Teng and F. F. Yao. k-nearest-neighbor clustering and percolation theory. *Algorithmica*, 49:192–211, 2007.

[152] B. Thirion, G. Varoquaux, E. Dohmatob, and J.-B. Poline. Which fMRI clustering gives good brain parcellations? *Frontiers in Neuroscience*, 8, 2014.

[153] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58, 1994.

[154] A. Tikhonov. On the stability of inverse problems. 1943.

[155] G. Tononi, O. Sporns, and G. M. Edelma. A measure for brain complexity: Relating functional segregation and integration in the nervous system. *Proceedings of the National Academy of Sciences*, 91: 5033–5037, 1994.

[156] J. A. Tropp. Improved analysis of the subsampled randomized hadamard transform. 2011.

[157] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 2014.

[158] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Information Science and Statistics. Springer-Verlag, 2000.

[159] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of probability and its applications*, 16:264–280, 1971.

[160] G. Varoquaux and B. Thirion. How machine learning is shaping cognitive neuroimaging. *GigaScience*, 3, 2014.

[161] G. Varoquaux, A. Gramfort, and B. Thirion. Small-sample brain mapping: sparse recovery on spatially correlated designs with randomization and clustering. In *International conference on machine learning*, page 1375, 2012.

[162] G. Varoquaux, P. R. Raamana, D. A. Engemann, A. Hoyos-Idrobo, Y. Schwartz, and B. Thirion. Assessing and tuning brain decoders: cross-validation, caveats, and guidelines. *arxiv preprint arxiv:1606.05201*, 2016.

[163] T. Wager, M. Davidson, B. Hughes, M. Lindquist, and K. Ochsner. Neural mechanisms of emotion regulation: evidence for two independent prefrontal-subcortical pathways. *Neuron*, 59:1037–1050, 2008.

[164] T. D. Wager, M. Lindquist, and L. Kaplan. Meta-analysis of functional neuroimaging data: current and future directions. *Social cognitive and affective neuroscience*, 2:150–158, 2007.

[165] T. D. Wager, L. Y. Atlas, M. A. Lindquist, M. Roy, C.-W. Woo, and E. Kross. An fmri-based neurologic signature of physical pain. *New England Journal of Medicine*, 368:1388–1397, 2013.

[166] H. Wang and P. A. Yushkevich. Multi-atlas segmentation without registration: a supervoxel-based approach. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 535–542. Springer, 2013.

[167] Z. Wang, A. Childress, J. Wang, and J. Detre. Support vector machine learning-based fMRI data group analysis. *Neuroimage*, 36:1139–1151, 2007.

[168] J. H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236, 1963.

[169] S. Weichwald, T. Meyer, O. Özdenizci, B. Schölkopf, T. Ball, and M. Grosse-Wentrup. Causal interpretation rules for encoding and decoding models in neuroimaging. *NeuroImage*, 110:48–59, 2015.

[170] C. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

[171] D. Williams, J. Detre, J. Leigh, and A. Koretsky. Magnetic resonance imaging of perfusion using spin inversion of arterial water. *Proceedings of the National Academy of Sciences*, 89:212–216, 1992.

[172] K. Worsley, A. Evans, S. Marrett, and P. Neelin. A three-dimensional statistical analysis for cbf activation studies in human brain. *Journal of Cerebral Blood Flow Metabolism*, 12:900–918, 1992.

[173] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transaction on Pattern Analysis and Maching Intelligence*, 31:210–227, 2009.

[174] Z. J. Xiang, H. Xu, and P. J. Ramadge. Learning sparse representations of high dimensional data on large scale dictionaries. In *Advances in neural information processing systems*, pages 900–908. 2011.

[175] B. Yu. Stability. *Bernoulli*, 19:1484–1500, 2013.

[176] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Ensembles de modèles pour l'IRMf: l'apprentissage stable à grande échelle
Mots-clé: IRMf, clustering, décodage, réduction de dimension.

Résumé: En imagerie médicale, des collaborations internationales ont lançé l'acquisition de centaines de Terabytes de données - et en particulier de données d'Imagerie par Résonance Magnétique fonctionelle (IRMf) - pour les mettre à disposition de la communauté scientifique. Extraire de l'information utile de ces données nécessite d'importants prétraitements et des étapes de réduction de bruit. La complexité de ces analyses rend les résultats très sensibles aux paramètres choisis. Le temps de calcul requis augmente plus vite que linéairement: les jeux de données sont si importants qu'il ne tiennent plus dans le cache, et les architectures de calcul classiques deviennent inefficaces.
Pour réduire les temps de calcul, nous avons étudié le feature-grouping comme technique de réduction de dimension. Pour ce faire, nous utilisons des méthodes de clustering. Nous proposons un algorithme de clustering agglomératif en temps linéaire: Recursive Nearest Agglomeration (ReNA). ReNA prévient la création de clusters énormes, qui constitue un défaut des méthodes agglomératives rapides existantes. Nous démontrons empiriquement que cet algorithme de clustering engendre des modèles très précis et rapides, et permet d'analyser de grands jeux de données avec des ressources limitées. En neuroimagerie, l'apprentissage statistique peut servir à étudier l'organisation cognitive du cerveau. Des modèles prédictifs permettent d'identifier les régions du cerveau impliquées dans le traitement cognitif d'un stimulus externe. L'entraînement de ces modèles est un problème de très grande dimension, et il est nécéssaire d'introduire un a priori pour obtenir un modèle satisfaisant.
Afin de pouvoir traiter de grands jeux de données et d'améliorer la stabilité des résultats, nous proposons de combiner le clustering et l'utilisation d'ensembles de modèles. Nous évaluons la performance empirique de ce procédé à travers de nombreux jeux de données de neuroimagerie. Cette méthode est hautement parallélisable et moins coûteuse que l'état de l'art en temps de calcul. Elle permet, avec moins de données d'entraînement, d'obtenir de meilleures prédictions. Enfin, nous montrons que l'utilisation d'ensembles de modèles améliore la stabilité des cartes de poids résultantes et réduit la variance du score de prédiction.

Title: Ensembles of models in fMRI: stable learning in large-scale settings
Keywords: fMRI, clustering, decoding, dimensionality reduction.

Abstract: In medical imaging, collaborative worldwide initiatives have begun the acquisition of hundreds of Terabytes of data that are made available to the scientific community. In particular, functional Magnetic Resonance Imaging –fMRI– data. However, this signal requires extensive fitting and noise reduction steps to extract useful information. The complexity of these analysis pipelines yields results that are highly dependent on the chosen parameters. The computation cost of this data deluge is worse than linear: as datasets no longer fit in cache, standard computational architectures cannot be efficiently used.
To speed-up the computation time, we considered dimensionality reduction by feature grouping. We use clustering methods to perform this task. We introduce a linear-time agglomerative clustering scheme, Recursive Nearest Agglomeration (ReNA). Unlike existing fast agglomerative schemes, it avoids the creation of giant clusters. We then show empirically how this clustering algorithm yields very fast and accurate models, enabling to process large datasets on budget. In neuroimaging, machine learning can be used to understand the cognitive organization of the brain. The idea is to build predictive models that are used to identify the brain regions involved in the cognitive processing of an external stimulus. However, training such estimators is a high-dimensional problem, and one needs to impose some prior to find a suitable model.
To handle large datasets and increase stability of results, we propose to use ensembles of models in combination with clustering. We study the empirical performance of this pipeline on a large number of brain imaging datasets. This method is highly parallelizable, it has lower computation time than the state-of-the-art methods and we show that, it requires less data samples to achieve better prediction accuracy. Finally, we show that ensembles of models improve the stability of the weight maps and reduce the variance of prediction accuracy.