Continuation of Nesterov's Smoothing for Regression with Structured Sparsity in High-Dimensional Neuroimaging

Fouad Hadj-Selem, Tommy Lofstedt, Elvis Dohmatob, Vincent Frouin, Mathieu Dubois, Vincent Guillemot, Edouard Duchesnay

▶ **To cite this version:**

Fouad Hadj-Selem, Tommy Lofstedt, Elvis Dohmatob, Vincent Frouin, Mathieu Dubois, et al.. Continuation of Nesterov's Smoothing for Regression with Structured Sparsity in High-Dimensional Neuroimaging. IEEE Transactions on Medical Imaging, In press, 2018, 10.1109/TMI.2018.2829802 . cea-01883286

HAL Id: cea-01883286
https://cea.hal.science/cea-01883286v1

Submitted on 28 Sep 2018

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.

Continuation of Nesterov's Smoothing for Regression with Structured Sparsity in High-Dimensional Neuroimaging

Fouad Hadj-Selem*, Tommy Löfstedt*, Elvis Dohmatob, Vincent Frouin, Mathieu Dubois, Vincent Guillemot, and Edouard Duchesnay*

***Abstract*—Predictive models can be used on high-dimensional brain images to decode cognitive states or diagnosis/prognosis of a clinical condition/evolution. Spatial regularization through structured sparsity offers new perspectives in this context and reduces the risk of overfitting the model while providing interpretable neuroimaging signatures by forcing the solution to adhere to domain-specific constraints. Total Variation (TV) is a promising candidate for structured penalization: it enforces spatial smoothness of the solution while segmenting predictive regions from the background. We consider the problem of minimizing the sum of a smooth convex loss, a non-smooth convex penalty (whose proximal operator is known) and a wide range of possible complex, non-smooth convex structured penalties such as TV or overlapping group Lasso. Existing solvers are either limited in the functions they can minimize or in their practical capacity to scale to high-dimensional imaging data. Nesterov's smoothing technique can be used to minimize a large number of non-smooth convex structured penalties. However, reasonable precision requires a small smoothing parameter, which slows down the convergence speed to unacceptable levels. To benefit from the versatility of Nesterov's smoothing technique, we propose a first order continuation algorithm, CONESTA, which automatically generates a sequence of decreasing smoothing parameters. The generated sequence maintains the optimal convergence speed towards any globally desired precision. Our main contributions are: To propose an expression of the duality gap to probe the current distance to the global optimum in order to adapt the smoothing parameter and the convergence speed. This expression is applicable to many penalties and can be used with other solvers than CONESTA. We also propose an expression for the particular smoothing parameter that minimizes the number of iterations required to reach a given precision. Further, we provide a convergence proof and its rate, which is an improvement over classical proximal gradient smoothing methods. We demonstrate on both simulated and high-dimensional structural neuroimaging data that CONESTA significantly outperforms many state-of-the-art solvers in regard to convergence speed and precision.**

***Keywords*—Neuroimaging, Prediction, Signature, Machine Learning, Structured Sparsity, Convex Optimization.**

E. Duchesnay, V. Frouin, V. Guillemot and M. Dubois are with NeuroSpin, CEA, Université Paris-Saclay - France.

F. Hadj-Selem is with the Energy Transition Institute: VeDeCoM - France.

T. Löfstedt is with the Department of Radiation Sciences, Umeå University, Umeå - Sweden.

E. Dohmatob is with PARIETAL Team, INRIA / CEA, Université Paris-Saclay - France.

*Contributed equally.

I. Introduction

Multivariate machine learning (ML) applied in neuroimaging offers new perspectives in early diagnosis and prognosis of brain diseases [1] using different MRI modalities, as well as brain decoding in cognitive neurosciences [19] using functional MRI. Indeed, ML algorithms can evaluate a vast number of brain-related features and capture complex relationships in the data with the purpose of making inferences at a single-subject level. The learning and application of ML algorithms should address the risk of overfitting when applied to high-dimensional training data; while simultaneously achieving their goal of providing predictive and interpretable brain patterns. Additionally, these predictive brain patterns should be stable with regard to some variability of the training subjects with similar clinical and demographical characteristics.

Issues regarding increased interpretability and high dimensionality can be addressed simultaneously using *e.g.* a limited ($\leq$ 100) number of atlas-based regions of interest (ROI) where the signal within each ROI is averaged. In a comparison study [10], voxel-based approaches outperformed region-based approaches, suggesting that the benefits obtained by dimensionality reduction do not compensate for the loss of informative signal caused by averaging within pre-defined regions. Indeed, regional approaches rely on the assumption that the spatial extent of the unknown predictive brain region matches with one of the atlas-based ROIs. Hence, this implies that ML algorithms should perform a data-driven selection of features that are organized in a few regions while still working at the voxel level.

This objective can be addressed with spatial regularization that simultaneously reduces the risk of overfitting while enforcing the solution to be organized in interpretable regions of connected voxels. Such regularizations extend classical predictors by introducing penalties that enforce local smoothness on the model parameters. Graph-Net [18], [12] forces adjacent voxels to have similar weights using a squared ℓ_2 penalty on the spatial gradients of the weight map. This penalty is differentiable, which considerably simplifies the optimization problem. However, Graph-Net results in a smooth map without clearly identified regions of piece-wise constant clusters of related voxels. The Total Variation (TV) penalty forces sparsity on the spatial derivatives of the weight map (using an ℓ_{12}-norm), segmenting the weight map into spatially-contiguous parcels with almost constant values [23]. TV can be combined

with sparsity-inducing penalties (such as ℓ_1 (Lasso) [27]) to obtain segmenting properties that extracts predictive regions from a noisy background with zeros [17], [14], [13]. TV, together with Lasso, produces the desired foreground-background segmentation by imposing constant-valued parcels.

We propose a solver that addresses a very general class of optimization problems including many group-wise penalties (allowing overlapping groups) such as Group Lasso and TV. The function that we wish to minimize has the form

$$f(\boldsymbol{\beta}) = \overbrace{g(\boldsymbol{\beta})}^{\text{smooth}} + \overbrace{\kappa h(\boldsymbol{\beta}) + \gamma s(\boldsymbol{\beta})}^{\text{non-smooth}}, \tag{1}$$

where $\boldsymbol{\beta}$ is the vector of parameters to be estimated. g is the sum of a differentiable loss $\mathcal{L}$ with any differentiable penalty, *e.g.*, the least-squares loss with the ℓ_2 (ridge) penalty: $g(\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$. The h is a penalty whose proximal operator is known, *e.g.*, the ℓ_1 penalty: $h(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1$. This regularization will induce sparsity on the parameter map. The s is a complex penalty on the structure of the input variables, for which the proximal operator is either not known, or for which the proximal operator is too expensive to compute. The only assumption is that s is an ℓ_{12} group norm of the following form:

$$s(\boldsymbol{\beta}) = \sum_{\phi \in \Phi} \|\boldsymbol{A}_\phi \boldsymbol{\beta}_\phi\|_2, \tag{2}$$

where $\Phi = \{\phi_1, \cdots, \phi_{|\Phi|}\}$ denotes the set of, possibly overlapping groups of features and $\boldsymbol{A}_\phi$ any linear operator on each group. Overlapping group Lasso and TV belong to this class of functions. The contribution of the penalties is controlled by the scalar weights: κ, λ and γ.

Although many solvers have already been proposed that can minimize this function, their practical use in the context of high-dimensional neuroimaging data ($\geq 10^5$ features) remains an open issue (see Supplementary (Sec. SM 2.2) and [28], [13]). This paper compares the convergence speed of the proposed solver to that of four state-of-the-art solvers: (i) An *Inexact* proximal gradient algorithm (Inexact FISTA) [26], where the proximal operator of s is approximated numerically. While the main algorithm (FISTA) enjoys a convergence rate of $\mathcal{O}\big(1/k^2\big)$ (with k the number of iterations), the precision of the approximation of the proximal operator is required to decrease as $\mathcal{O}\big(1/k^{4+\delta}\big)$ for any $\delta > 0$ [26, Proposition 2]. This results in prohibitive computations in order to reach a reasonable precision, especially with high-dimensional $\boldsymbol{\beta}$ vectors as found with brain images that generally involve $\geq 10^4$ (functional MRI) to $\geq 10^5$ (structural MRI) features. (ii) The Excessive Gap Method (EGM) [24], which does not allow true sparsity nor general loss functions. (iii) The alternating direction method of multipliers (ADMM) [7], which is computationally expensive and/or difficult to compute for general structured penalties and which suffers from a difficulty of setting the regularization parameter, as mentioned in [13], [26]. (iv) Nesterov's smoothing technique [24], which provides an appealing and generic framework in which a large range of non-smooth convex structured penalties can be minimized without computing their proximal operators. However, reasonable precision ($\approx 10^{-3}$ or higher) requires a very small smoothing parameter, which brings down the convergence rate to unacceptable levels.

We propose a continuation solver, called *CONESTA* (short for **CO**ntinuation with **NE**sterov smoothing in a **S**hrinkage-**T**hresholding **A**lgorithm), based on Nesterov's smoothing technique, that automatically generates a decreasing sequence of smoothing parameters in order to maintain the optimal convergence speed towards any globally desired precision. Nesterov's smoothing technique makes the solver highly versatile: it can address a large class of complex penalties (the function s in Eq. 1) where the proximal operators are either not known or expensive to compute. The problem can be minimized using an accelerated proximal gradient method, possibly also utilizing (non-smoothed, *e.g.* ℓ_1) sparsity-inducing penalties. CONESTA can be understood as a smooth touchdown procedure that uses the duality gap to probe the distance to the ground (global optimum) and dynamically adapts its speed (the smoothing parameter) according to this distance.

As a first contribution, we propose an expression of the duality gap to control the convergence of the global non-smooth problem. This expression is applicable to a large range of structured penalties, which makes it useful beyond the scope of the present paper. The duality gap provides an upper bound on the current error, which can be used to derive the next target precision to be reached by the smoothed problem.

As a second contribution, we propose an expression that provides the particular smoothing parameter that minimizes the number of iterations required to reach the above-mentioned prescribed precision.

As a third contribution, we provide a convergence proof and rate, which is an improvement over classical (without continuation) proximal gradient smoothing.

The fourth contribution is the experimental demonstration, on both simulated and high-dimensional structural neuroimaging data, that CONESTA significantly outperforms the state-of-the-art solvers, *i.e.*, EGM, ADMM, classical FISTA with fixed smoothing and Inexact FISTA, in regards to convergence speed and/or precision of the solution. We further demonstrate the relevance of TV for extracting stable predictive signatures from structural MRI data.

A. Reformulating TV as a linear operator

Before discussing the minimization with respect to $\boldsymbol{\beta}$, we provide details on the encoding of the spatial structure within the penalty s. It is essential to note that the algorithm is independent of the spatial structure of the data. All the structural information is encoded in the linear operator, $\boldsymbol{A}$, and it is computed outside of the algorithm. Thus, the algorithm has the ability to address various structured data (arrays, meshes) and, most importantly, also other penalties aside from the TV penalty. Using our generic CONESTA solver for any penalty, which can be written in the form of Eq. 1, requires only building the linear operator $\boldsymbol{A}$; and a projection function, detailed in Eq. 9. This section presents the formulation and the design of $\boldsymbol{A}$ in the specific case of the TV penalty applied to the parameter vector $\boldsymbol{\beta}$ measured on a 3-dimensional (3D) image.

A brain mask is used to establish a mapping, $\phi(i,j,k)$, from integer coordinates (i,j,k) in the 3D grid of the brain image, and an index $\phi \in \{1,\dots,P\}$ in the collapsed (vectorized) image. We extract the spatial (forward) neighborhood at (i,j,k), of size ≤ 4, corresponding to a voxel and its three neighboring voxels, within the mask, in the positive i, j and k directions. The TV penalty is defined as

$$\mathrm{TV}(\boldsymbol{\beta}) \equiv \sum_{i,j,k} \left\| \nabla\big(\boldsymbol{\beta}_{\phi(i,j,k)}\big) \right\|_2, \tag{3}$$

where $\nabla(\boldsymbol{\beta}_{\phi(i,j,k)})$ denotes the spatial gradient of the parameter map, $\boldsymbol{\beta} \in \mathbb{R}^P$, at the 3D position (i,j,k) mapped to element $\phi(i,j,k)$ in $\boldsymbol{\beta}$. A first order approximation of the spatial gradient, $\nabla(\boldsymbol{\beta}_{\phi(i,j,k)})$, can be computed by applying the linear operator $\boldsymbol{A}'_\phi \in \mathbb{R}^{3\times 4}$ to the parameter vector $\boldsymbol{\beta}'_{\phi(i,j,k)} \in \mathbb{R}^4$ as

$$\nabla\left(\boldsymbol{\beta}_{\phi(i,j,k)}\right) \equiv \underbrace{\begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{A}'_\phi} \underbrace{\begin{bmatrix} \boldsymbol{\beta}_{\phi(i,j,k)} \\ \boldsymbol{\beta}_{\phi(i+1,j,k)} \\ \boldsymbol{\beta}_{\phi(i,j+1,k)} \\ \boldsymbol{\beta}_{\phi(i,j,k+1)} \end{bmatrix}}_{\boldsymbol{\beta}'_{\phi(i,j,k)}}, \tag{4}$$

where thus $\boldsymbol{\beta}'_{\phi(i,j,k)}$ contains the elements at linear indices in the collapsed parameter map, $\boldsymbol{\beta}$, corresponding to the spatial neighborhood in the 3D image at $\boldsymbol{\beta}_{\phi(i,j,k)}$. Then, $\boldsymbol{A}'_\phi$ is extended, by zeros, to a large but very sparse matrix $\boldsymbol{A}_{\phi(i,j,k)} \in \mathbb{R}^{3\times P}$ such that $\boldsymbol{A}'_\phi \boldsymbol{\beta}'_{\phi(i,j,k)} = \boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}$. If some neighbors lie outside of the mask, the corresponding rows in $\boldsymbol{A}_{\phi(i,j,k)}$ are removed (or set to zero). Approximating TV by

$$\mathrm{TV}(\boldsymbol{\beta}) = \sum_{i,j,k} \|\boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}\|_2 \tag{5}$$

allows us to use the TV, as the structured penalty s, in Eq. 1. Finally, with a vertical concatenation of all the $\boldsymbol{A}_{\phi(i,j,k)}$ matrices, we obtain the full linear operator $\boldsymbol{A} \in \mathbb{R}^{3P\times P}$ that will be used in the following sections.

II. Nesterov's smoothing of the structured penalty

We consider the convex non-smooth minimization of Eq. 1 with respect to $\boldsymbol{\beta}$. This problem includes a general structured penalty, s, that (for the purpose of this paper) covers the specific case of TV. The accelerated proximal gradient algorithm (FISTA) [4] can be used to solve the problem when applying only *e.g.* the ℓ_1 penalty. However, since the proximal operator of TV, together with the ℓ_1 penalty, has no closed-form expression, standard implementations of those algorithms are not suitable. In order to overcome this barrier we used Nesterov's smoothing technique [25], which consists of approximating the non-smooth penalty for which the proximal operator is unknown (*e.g.*, TV) with a smooth function (for which the gradient is known). Non-smooth penalties with known proximal operators (*e.g.*, ℓ_1) are not affected by this smoothing. Hence, as described in [9], this allowed us to use an exact accelerated proximal gradient algorithm.

Using the dual norm of the ℓ_2-norm (*i.e.* the ℓ_2-norm), Eq. 5 can be reformulated as

$$\begin{aligned}\mathrm{TV}(\boldsymbol{\beta}) &= \sum_{i,j,k} \|\boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}\|_2 \\ &= \sum_{i,j,k} \max_{\|\boldsymbol{\alpha}_{\phi(i,j,k)}\|_2 \leq 1} \boldsymbol{\alpha}_{\phi(i,j,k)}^\top \boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}, \end{aligned} \tag{6}$$

where $\boldsymbol{\alpha}_{\phi(i,j,k)} \in \mathcal{K}_{\phi(i,j,k)} = \{\boldsymbol{\alpha}_{\phi(i,j,k)} \in \mathbb{R}^3 : \|\boldsymbol{\alpha}_{\phi(i,j,k)}\|_2 \leq 1\}$ is a vector of auxiliary variables in the ℓ_2 unit ball, associated with $\boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}$. As with $\boldsymbol{A} \in \mathbb{R}^{3P\times P}$, which is the vertical concatenation of all the $\boldsymbol{A}_{\phi(i,j,k)}$, we concatenate all the $\boldsymbol{\alpha}_{\phi(i,j,k)}$ to form $\boldsymbol{\alpha} \in \mathcal{K} = \{[\boldsymbol{\alpha}_1^\top,\dots,\boldsymbol{\alpha}_P^\top]^\top : \boldsymbol{\alpha}_l \in \mathcal{K}_l, \forall\, l = \phi(i,j,k) \in \{1,\dots,P\}\} \in \mathbb{R}^{3P}$. The set $\mathcal{K}$ is the Cartesian product of closed 3D unit balls in Euclidean space and, therefore, a compact convex set. Eq. 6 can now further be written as

$$\mathrm{TV}(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}\in\mathcal{K}} \boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{\beta} = s(\boldsymbol{\beta}), \tag{7}$$

and with this formulation of s, we can apply Nesterov's smoothing technique. For a given smoothing parameter, $\mu > 0$, the function s is approximated by the smooth function

$$s_\mu(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}\in\mathcal{K}} \left\{ \boldsymbol{\alpha}^\top \boldsymbol{A}\boldsymbol{\beta} - \frac{\mu}{2}\|\boldsymbol{\alpha}\|_2^2 \right\}, \tag{8}$$

for which $\lim_{\mu\to 0} s_\mu(\boldsymbol{\beta}) = s(\boldsymbol{\beta})$. Nesterov [25] demonstrates this convergence using the inequality in Eq. 12. The value of $\boldsymbol{\alpha}^*_\mu(\boldsymbol{\beta}) = [\boldsymbol{\alpha}^{*\top}_{\mu,1},\dots,\boldsymbol{\alpha}^{*\top}_{\mu,\phi(i,j,k)},\dots,\boldsymbol{\alpha}^{*\top}_{\mu,P}]^\top$ that maximizes Eq. 8 is the concatenation of projections of the vectors $\boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta} \in \mathbb{R}^3$ onto the ℓ_2 ball $\mathcal{K}_{\phi(i,j,k)}$, *i.e.* $\boldsymbol{\alpha}^*_{\mu,\phi(i,j,k)}(\boldsymbol{\beta}) = \mathrm{proj}_{\mathcal{K}_{\phi(i,j,k)}}\left(\frac{\boldsymbol{A}_{\phi(i,j,k)}\boldsymbol{\beta}}{\mu}\right)$, where

$$\mathrm{proj}_{\mathcal{K}_{\phi(i,j,k)}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x} & \text{if } \|\boldsymbol{x}\|_2 \leq 1 \\ \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2} & \text{otherwise.} \end{cases} \tag{9}$$

The function s_μ, *i.e.* Nesterov's smooth transform of s, is convex and differentiable. Its gradient is given by Nesterov [25] as

$$\nabla s_\mu(\boldsymbol{\beta}) = \boldsymbol{A}^\top \boldsymbol{\alpha}^*_\mu(\boldsymbol{\beta}). \tag{10}$$

The gradient is Lipschitz-continuous, with constant

$$L\big(\nabla(s_\mu)\big) = \frac{\|\boldsymbol{A}\|_2^2}{\mu}, \tag{11}$$

in which $\|\boldsymbol{A}\|_2$ is the matrix spectral norm of $\boldsymbol{A}$. Moreover, Nesterov [25] provides the following inequality, relating s_μ and s

$$s_\mu(\boldsymbol{\beta}) \leq s(\boldsymbol{\beta}) \leq s_\mu(\boldsymbol{\beta}) + \mu M, \quad \forall \boldsymbol{\beta} \in \mathbb{R}^P, \tag{12}$$

where $M = \max_{\boldsymbol{\alpha}\in\mathcal{K}} \frac{1}{2}\|\boldsymbol{\alpha}\|_2^2 = \frac{P}{2}$.

Thus, a new (smoothed) function, closely related to Eq. 1, arises as

$$f_\mu(\boldsymbol{\beta}) = \overbrace{\underbrace{\mathcal{L}(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_2^2}_{g(\boldsymbol{\beta})} + \gamma \underbrace{\left\{ \boldsymbol{\alpha}^*_\mu(\boldsymbol{\beta})^\top \boldsymbol{A}\boldsymbol{\beta} - \frac{\mu}{2}\|\boldsymbol{\alpha}^*\|_2^2 \right\}}_{s_\mu(\boldsymbol{\beta})}}^{\text{smooth}} + \kappa \overbrace{\underbrace{\|\boldsymbol{\beta}\|_1}_{h(\boldsymbol{\beta})}}^{\text{non-smooth}}. \tag{13}$$

Hence, we can explicitly compute the gradient of the smooth part, $\nabla(g+\gamma s_\mu)$, Eq. 10, its Lipschitz constant, Eq. 11, and also the proximal operator of the non-smooth part. We thus have all the necessary ingredients to minimize the function using *e.g.* an accelerated proximal gradient method [4]. Given a starting point, $\boldsymbol{\beta}^0$, and a smoothing parameter, μ, FISTA (Algorithm 1) minimizes the smoothed function and reaches a given precision, ε_μ.

Algorithm 1 FISTA$\big(\boldsymbol{\beta}^0,\, \varepsilon_\mu,\, \mu,\, \boldsymbol{A},\, g,\, s_\mu,\, h,\, \gamma,\, \kappa\big)$

1: $\boldsymbol{\beta}^1 = \boldsymbol{\beta}^0$; $k = 2$
2: Step size $t_\mu = \left(L(\nabla(g)) + \gamma \frac{\|\boldsymbol{A}\|_2^2}{\mu}\right)^{-1}$
3: **repeat**
4: $\quad \mathbf{z} = \boldsymbol{\beta}^{k-1} + \frac{k-2}{k+1}\left(\boldsymbol{\beta}^{k-1} - \boldsymbol{\beta}^{k-2}\right)$
5: $\quad \boldsymbol{\beta}^k = \mathrm{prox}_{\kappa h}\big(\mathbf{z} - t_\mu \nabla(g+\gamma s_\mu)(\mathbf{z})\big)$
6: **until** $\textsc{Gap}_\mu(\boldsymbol{\beta}^k) \leq \varepsilon_\mu$ (see Sec. III-A)
7: **return** $\boldsymbol{\beta}^k$

III. The CONESTA algorithm

The step size, t_μ, computed in Line 2 of Algorithm 1, must be smaller than or equal to the reciprocal of the Lipschitz constant of the gradient of the smooth part, *i.e.* of $g+\gamma s_\mu$ [4]. This relationship between t_μ and μ implies a trade-off between speed and precision: Indeed, the FISTA convergence rate, given in the Supplementary (Eq. SM 2.3), shows that a high precision (small μ and t_μ) will lead to a slow convergence. Conversely, poor precision (large μ and t_μ) will lead to rapid convergence.

To optimize this trade-off, we propose a continuation approach (Algorithm 2) that decreases the smoothing parameter with respect to the distance to the minimum. On the one hand, when we are far from $\boldsymbol{\beta}^*$ (the minimum of Eq. 1), we can use a large μ to rapidly decrease the objective function. On the other hand, when we are close to $\boldsymbol{\beta}^*$, we need a small μ in order to obtain an accurate approximation of the original objective function.

A. Duality gap

The distance to the unknown $f(\boldsymbol{\beta}^*)$ is estimated using a duality gap. Duality formulations are often used to control the achieved precision level when minimizing convex functions. The duality gap provides an upper bound of the error, $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*)$, for any $\boldsymbol{\beta}^k$, when the minimum is unknown. Moreover, it vanishes at the minimum:

$$\begin{aligned} \textsc{Gap}(\boldsymbol{\beta}^k) \geq f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*) &\geq 0, \\ \textsc{Gap}(\boldsymbol{\beta}^*) &= 0. \end{aligned} \tag{14}$$

The duality gap is the cornerstone of the CONESTA algorithm. Indeed, it is used three times:

(i) As the stopping criterion in the inner FISTA loop (Line 6 in Algorithm 1). FISTA will stop as soon as the current precision is achieved using the current smoothing parameter, μ. This prevents unnecessary iterations toward the approximated (smoothed) objective function.

(ii) In the ith CONESTA iteration, as a way to estimate the current error $f(\boldsymbol{\beta}^i) - f(\boldsymbol{\beta}^*)$ (Line 7 in Algorithm 2). The next desired precision, ε^{i+1}, and the smoothing parameter, μ^{i+1} (using Theorem 2), are derived from this value.

(iii) Finally, as the global stopping criterion in CONESTA (Line 10 in Algorithm 2). This guarantees that the obtained approximation of the minimum, $\boldsymbol{\beta}^i$, at convergence, satisfies $f(\boldsymbol{\beta}^i) - f(\boldsymbol{\beta}^*) < \varepsilon$.

Eq. 13 decomposes the smoothed objective function as a sum of a strongly convex loss, $\mathcal{L}$, and the penalties. Moreover, the least-squares loss, $\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2$, can be re-written as a function of $\boldsymbol{X}\boldsymbol{\beta}$ by $l(\boldsymbol{z}) \equiv \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{y}\|^2$, where $\boldsymbol{z} = \boldsymbol{X}\boldsymbol{\beta}$. Therefore, we can equivalently express the smoothed objective function as

$$\begin{aligned} f_\mu(\boldsymbol{\beta}) &= \mathcal{L}(\boldsymbol{\beta}) + \Omega_\mu(\boldsymbol{\beta}) \\ &= l(\boldsymbol{X}\boldsymbol{\beta}) + \Omega_\mu(\boldsymbol{\beta}), \end{aligned}$$

where Ω_μ represents all penalty terms of Eq. 13. Our aim is to compute the duality gap to obtain an upper bound estimation of the distance to the optimum. At any step k of the algorithm, given the current primal $\boldsymbol{\beta}^k$ and the dual $\boldsymbol{\sigma}(\boldsymbol{\beta}^k) \equiv \nabla\mathcal{L}(\boldsymbol{X}\boldsymbol{\beta}^k)$ variables [6], we can compute the duality gap using the Fenchel duality rules [22]. This requires computing the Fenchel conjugates, l^* and Ω_μ^*, of l and Ω_μ, respectively. While the expression of l^* is straightforward, to the best of our knowledge, there is no explicit expression for Ω_μ^* when using a complex penalty such as TV or group Lasso. Therefore, as an important theoretical contribution of this paper, we provide the expression for Ω_μ^* in order to compute an approximation of the duality gap that maintains its properties (Eq. 14).

Theorem 1 (Duality gap for the smooth problem)**.** *The following estimation of the duality gap satisfies Eq. 14 , for any iterate $\boldsymbol{\beta}^k$:*

$$\textsc{Gap}_\mu(\boldsymbol{\beta}^k) \equiv f_\mu(\boldsymbol{\beta}^k) + l^*(\boldsymbol{\sigma}(\boldsymbol{\beta}^k)) + \Omega_{\mu,k}^*(-\boldsymbol{X}^\top\boldsymbol{\sigma}(\boldsymbol{\beta}^k)), \tag{15}$$

with dual variable

$$\boldsymbol{\sigma}(\boldsymbol{\beta}^k) \equiv \nabla l(\boldsymbol{X}\boldsymbol{\beta}^k) = \boldsymbol{X}\boldsymbol{\beta}^k - \boldsymbol{y}, \tag{16}$$

and the Fenchel conjugates

$$\begin{aligned} l^*(\boldsymbol{z}) =& \frac{1}{2}\|\boldsymbol{z}\|_2^2 + \langle \boldsymbol{z}, \boldsymbol{y}\rangle \\ \Omega_{\mu,k}^*(\boldsymbol{z}) \equiv& \frac{1}{2\lambda}\sum_{j=1}^{P}\left(\left[\left|z_j - \gamma\big(\boldsymbol{A}^\top\boldsymbol{\alpha}_\mu^*(\boldsymbol{\beta}^k)\big)_j\right| - \kappa\right]_+^2\right) \\ &+ \frac{\gamma\mu}{2}\big\|\boldsymbol{\alpha}_\mu^*(\boldsymbol{\beta}^k)\big\|_2^2, \end{aligned} \tag{17}$$

where $[\,\cdot\,]_+ = \max(0, \cdot)$.

The proof of this theorem can be found in Supplementary (Sec. SM 3.1.3). Note that there we also provide the expression and proof of the Fenchel conjugate for the non-smoothed problem, *i.e.*, using Ω instead of Ω_μ.

The expression in Eq. 15 of the duality gap of the smooth problem combined with the inequality in Eq. 12 provides an

estimation of the distance to the minimum of the original non-smoothed problem. The sought distance is decreased geometrically by a factor $\tau \in (0, 1)$ at the end of each continuation, and the decreased value defines the precision that should be reached by the next iteration (Line 8 of Algorithm 2). Thus, the algorithm dynamically generates a sequence of decreasing precisions, ε^i. Such a scheme ensures the convergence (see Sec. III-C) towards a globally desired final precision, ε, which is the only parameter that the user needs to provide.

B. Determining the optimal smoothing parameter

Given the current precision, ε^i, we need to compute a smoothing parameter $\mu_{opt}(\varepsilon^i)$ (Line 9 in Algorithm 2) that minimizes the number of FISTA iterations required to achieve such a precision when minimizing Eq. 1 via Eq. 13 (*i.e.*, such that $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*) < \varepsilon^i$). We have the following theorem giving the expression of the optimal smoothing parameter, for which a proof is provided in the Supplementary (Sec. SM 3.2).

Theorem 2 (Optimal smoothing parameter, μ)**.** *For any given $\varepsilon > 0$, selecting the smoothing parameter as*

$$\mu_{opt}(\varepsilon) = \frac{-\gamma M \|\boldsymbol{A}\|_2^2 + \sqrt{(\gamma M \|\boldsymbol{A}\|_2^2)^2 + ML(\nabla(g))\|\boldsymbol{A}\|_2^2 \varepsilon}}{ML(\nabla(g))}, \tag{18}$$

minimizes the worst case bound on the number of iterations required to achieve the precision $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*) < \varepsilon$.

Note that $M = P/2$ (Eq. 12) and the Lipschitz constant of the gradient of g as defined in Eq. 13 is $L(\nabla(g)) = \lambda_{\max}(\boldsymbol{X}^\top \boldsymbol{X}) + \lambda$, where $\lambda_{\max}(\boldsymbol{X}^\top \boldsymbol{X})$ is the largest eigenvalue of $\boldsymbol{X}^\top \boldsymbol{X}$.

C. Convergence analysis of CONESTA

The user only has to provide the globally prescribed precision ε, which will be guaranteed by the duality gap. Other parameters are related to the problem to be minimized (*i.e.* g, γ, s, κ, h) and the encoding of the data structure $\boldsymbol{A}$. Finally, the value of τ was set to 0.5. Indeed, experiments shown in Supplementary (Sec. SM 4.2) have demonstrated that values of 0.5 or 0.2 led to similar and increased speeds compared to larger values, such as 0.8.

CONESTA can be understood as a smooth touchdown procedure that uses the duality gap to probe the distance to the ground (global optimum) in order to dynamically adapt its speed (the smoothing). Indeed, each continuation step of CONESTA (Algorithm 2) probes (Line 7) an upper bound ε^i of the current distance to the optimum ($f(\boldsymbol{\beta}^i) - f(\boldsymbol{\beta}^*)$) using the duality gap. Then, Line 8 computes the next precision to be reached, ε^{i+1}, decreasing ε^i by a factor $\tau \in (0, 1)$. Line 9 derives the optimal smoothing parameter, μ^{i+1}, required to reach this precision as fast as possible. Finally, Line 5 transforms back the precision with respect to the original problem into a precision for the smoothed problem, ε_μ^i, using the inequality in Eq. 12. Therefore, at the next iteration, FISTA (Line 6) will decrease f_μ^i until the error reaches ε_μ^i. Thanks to Line 5, this implies that the true error (toward the non-smoothed problem) will be smaller than ε^i. The resulting weight vector, $\boldsymbol{\beta}^{i+1}$, will be the initial value for the next continuation step using updated parameters. Note that we use the duality gap for the smoothed problem, $\text{GAP}_{\mu=\mu^i}$ (and ε_μ^i), and transform it back and forth using Eq. 12 to obtain the duality gap for the non-smooth problem, GAP (and ε^i). We do this because the gap on Line 7 has already been computed at the last iteration of the FISTA loop (Line 6), since it was used in the stopping criterion. Moreover, GAP_μ converges to zero for any fixed μ unlike GAP.

Algorithm 2 CONESTA$\big(\varepsilon,\ \boldsymbol{A},\ g,\ s,\ h,\ \gamma,\ \kappa,\ \tau = 0.5\big)$

1: Initialize $\boldsymbol{\beta}^0 \in \mathbb{R}^P$
2: $\varepsilon^0 = \tau \cdot \text{GAP}_{\mu=10^{-8}}(\boldsymbol{\beta}^0)$
3: $\mu^0 = \mu_{opt}\left(\varepsilon^0\right)$
4: **repeat**
5: $\quad \varepsilon_\mu^i = \varepsilon^i - \mu^i \gamma M$
6: $\quad \boldsymbol{\beta}^{i+1} = \text{FISTA}(\boldsymbol{\beta}^i,\ \varepsilon_\mu^i,\ \mu^i,\ \boldsymbol{A},\ g,\ s_{\mu^i},\ h,\ \gamma,\ \kappa)$
7: $\quad \varepsilon^i = \text{GAP}_{\mu=\mu^i}(\boldsymbol{\beta}^{i+1}) + \mu^i \gamma M$
8: $\quad \varepsilon^{i+1} = \tau \cdot \varepsilon^i$
9: $\quad \mu^{i+1} = \mu_{opt}\big(\varepsilon^{i+1}\big)$
10: **until** $\varepsilon^i \leq \varepsilon$
11: **return** $\boldsymbol{\beta}^{i+1}$

The initialization (Line 2) is a particular case where we use GAP_μ with a negligible smoothing value of *e.g.* $\mu = 10^{-8}$. We then derive the initial smoothing parameter on Line 3. Therefore, if we start close to the solution the algorithm will automatically pick a small smoothing parameter, which makes CONESTA an excellent candidate for warm-restart.

The following theorem ensures the convergence and convergence speed of CONESTA.

Theorem 3 (Convergence of CONESTA)**.** *Let $\left(\mu^i\right)_{i=0}^\infty$ and $\left(\varepsilon^i\right)_{i=0}^\infty$ be defined recursively by CONESTA (Algorithm 2). Then, we have that*

(i) $\lim_{i\to\infty} \varepsilon^i = 0$, *and*

(ii) $f(\boldsymbol{\beta}^i) \xrightarrow{i\to\infty} f(\boldsymbol{\beta}^*)$.

(iii) Convergence rate of CONESTA with fixed smoothing (without continuation): For any given desired precision $\varepsilon > 0$, using a fixed smoothing (line 6 of Algorithm 2) with an optimal value of μ, equal to $\mu_{opt}(\varepsilon)$, if the number of iterations k is larger than

$$\frac{\sqrt{8\|\boldsymbol{A}\|_2^2 M \gamma^2 \|\boldsymbol{\beta}^0 - \boldsymbol{\beta}^*\|_2^2}}{\varepsilon} + \frac{\sqrt{2L(\nabla(g))\|\boldsymbol{\beta}^0 - \boldsymbol{\beta}^*\|_2^2}}{\sqrt{\varepsilon}}.$$

then the obtained $\boldsymbol{\beta}^k$ satisfies $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*) < \varepsilon$.

(iv) Convergence rate of CONESTA (with continuation), assuming uniqueness of the minimum (β^*): For any given desired precision $\varepsilon > 0$, if the total sum of all the inner FISTA iterations is larger than

$$C/\varepsilon,$$

where $C > 0$ is a constant, then the obtained solution (obtained from (iii)), i.e. $\boldsymbol{\beta}^i$, satisfies $f(\boldsymbol{\beta}^i) - f(\boldsymbol{\beta}^*) < \varepsilon$.

The proof is provided Sec. SM 3.3 in the supplementary. Claim (i): Sec. SM 3.3.1 demonstrates that the sequence of decreased precisions ε^i converges toward any prescribed precision. Claim (ii): Sec. SM 3.3.2 demonstrates, at each step of the sequence, the solutions of the smoothed problem converge toward the solution of the non-smoothed problem. Claim (iii): Sec. SM 3.3.3 demonstrates the number of iterations required to converge toward $\boldsymbol{\beta}^*$ using the auxiliary smoothed problem (without continuation) with a fixed and optimal smoothing value. Finally, claim (iv), Sec. SM 3.3.4 provides the convergence rate with respect to the total number of iterations.

The continuation technique improves the convergence rate compared to the simple smoothing using a single value of μ. Indeed, it has been demonstrated in [5] (see also [9]) that the convergence rate obtained with a single value of μ, even optimized, is $\mathcal{O}(1/\varepsilon) + \mathcal{O}(1/\sqrt{\varepsilon})$. However, the CONESTA algorithm achieves $\mathcal{O}(1/\varepsilon)$ for simply (non-strongly) convex functions.

In Theorem 3 we need the uniqueness hypothesis in (iv) to obtain the boundedness of the FISTA iterations. The authors in [8] have proved this boundedness property for a slightly modified version of the FISTA algorithm and general convex function. Nevertheless, we could not use these results because it should be adapted to the smoothed version, which has not been done yet to the best of our knowledge. This would require establishment of the boundedness of the FISTA iterated uniformly with respect to μ as it converges to zero.

Moreover, we avoid the strong convexity hypothesis because if the objective function is strongly convex with strong convexity parameter of the g function, then FISTA enjoys global linear convergence rate and especially ensures the convergence of $\boldsymbol{\beta}^k$ to the minimum, $\boldsymbol{\beta}^*$. In fact, we have that $\|\boldsymbol{\beta}^k - \boldsymbol{\beta}^*\|_2 < \sqrt{2\sigma/\varepsilon}$ as far as $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*) < \varepsilon$ (see Remark 3 in [9]). In this case, not only is the hypothesis in (iv) naturally satisfied, but it is also a favorable case to apply a continuation technique since the estimations on the successive intermediate $\boldsymbol{\beta}^k$ will ensure a faster convergence contrary to the simple convergence case in which we don't make use of this rate. Nevertheless, we did not include this case in this paper since it will engender the modification of the optimal smoothing value μ_{opt} and consequently all the other estimations due to the different convergence rate of FISTA. These questions will instead be considered in future works.

IV. Experiments

In this section we compare CONESTA to the state-of-the-art algorithms mentioned above (see Supplementary (Sec. SM 6) for details), *i.e.*, ADMM, EGM, Inexact FISTA and FISTA with fixed μ. We will use these algorithms to solve the problem on both simulated and high-dimensional structural neuroimaging data.

We used FISTA with fixed μ using two values of μ, chosen as follows: (i) Chen's μ where $\mu = \varepsilon/(2\gamma M)$ as was used in [9] and (ii) Large $\mu = (\text{Chen's } \mu)^{1/2}$. The first proposal for μ ensures that we reach the desired precision, although convergence may be slow. The second proposal has a value of μ that may not guarantee we reach the desired precision before convergence.

A. Simulated 1D data set where the minimum of f is known

We generated simulated data where we control the true minimizer, $\boldsymbol{\beta}^*$, and the associated regularization parameters, κ, λ and γ [21]. The experimental setup for the simulated 1D data set was inspired by that of [3] and is shown in Tab. I. This setup is a designed experiment with multiple small- and medium-scale data sets having combinations of low, medium and high degrees of correlation between variables, low, medium and high levels of sparsity, and low, medium and high signal-to-noise ratios.

TABLE I: The experimental setup for the simulation study. The parameters varied were: the size of the data set, (n, p), the correlation between variables, the sparsity of the data, and the signal-to-noise ratio. Five runs were performed for each combination of settings in order to assess the variability of the different algorithms. The medium level for sparsity (marked with an asterisk, '*') was only performed for the low and medium sizes.

Level	Size ($n \times p$)	Correlation	Sparsity	Signal-to-noise ratio
Low	200×200	Low	50 %	0.5
Medium	632×1514	Medium	72.5 %*	1.0
High	2000×10000	High	95 %	5.0

A candidate version of the predictors, denoted $\boldsymbol{X}_0$, was drawn from a multivariate Gaussian distribution $\mathcal{N}(\mathbf{1}, \Sigma)$, where Σ was generated according to the constant correlation model described in [20], where $\text{diag}(\Sigma) = \mathbf{1}$ and off-diagonal elements are all equal to ρ which is drawn from $\mathcal{N}(0, d_c/\sqrt{n})$. Here, d_c controls the dispersion of the correlation and takes values of $d_c = 1, 4.5$ and 8 for low, middle and high correlation cases. A sparsity parameter (Tab. I) fixes the percent of null weights within weight vector $\boldsymbol{\beta}^*$. The remaining weights that contribute to the prediction were drawn from $\mathcal{U}(0, 1)$ and sorted in ascending order. The residuals, $\mathbf{e}$, were drawn from a Gaussian distribution, $\mathcal{N}(1, 1)$, and then scaled to unit ℓ_2-norm. Given a candidate data matrix, $\boldsymbol{X}_0$, the true minimizer $\boldsymbol{\beta}^*$, the true residual vector $\boldsymbol{e}$, and regularization constants, κ, λ and γ, that were arbitrarily set to $\kappa = 0.618$, $\lambda = 1 - \kappa$ and $\gamma = 1.618$, we generated the simulated data set $(\boldsymbol{X}, \boldsymbol{y})$ such that $\boldsymbol{\beta}^*$ is the solution that minimizes Eq. 1, with $s(\boldsymbol{\beta}^*) = \text{TV}(\boldsymbol{\beta}^*)$. The final data matrix, $\boldsymbol{X}$, is obtained by applying a scaling factor to each column of the candidate data matrix, $\boldsymbol{X}_0$. All details of the scaling procedure to produce a data set with a known exact solution are described in the ancillary paper [21]. Once $\boldsymbol{X}$ is found, the $\boldsymbol{y}$ vector is computed as $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^* - \mathbf{e}$. For each combination of settings, we generated data knowing the exact solution $\boldsymbol{\beta}^*$ of the given minimization problem. Thus, we could compute the error $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*)$ for any current solution $\boldsymbol{\beta}^k$. Five runs

were performed for each combination of the settings, resulting in a total of 405 simulation runs.

1) Result of the comparison: We applied the minimization algorithms to simulated data that were generated using the settings shown in Tab. I. For each of the settings of the designed experiment, we measured the number of iterations and the time (in seconds) required to reach a certain precision level, ε, for each of the tested minimization algorithms. For each precision level, ranging from 1 to 10^{-6}, for each algorithm and for each data set, we *ranked* each algorithm according to the time they required to reach a given precision. Then, we averaged the ranks across data sets (see Tab. II) and we tested (Friedman test [15]) the significance of the difference in ranks between algorithms.

TABLE II: Average rank of the convergence speed of the algorithms to reach precisions ($f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*)$) ranging from 1 to 10^{-6}. We have here reported whether the average rank of a given algorithm was significantly larger $>$ (slower) or significantly smaller $<$ (faster) than CONESTA (a missing '$>$' or '$<$' means non-significant). P-values were calculated with a post hoc analysis of the Friedman test corrected for multiple comparisons. Note that all reported significant differences had a corrected p-value of 10^{-3} or smaller. For a given data set, all algorithms were evaluated with a limited upper execution time. Thus, some high precisions (*e.g.*, $10^{-5}, 10^{-6}$) were not always reached within the limited time. For FISTA with a large μ, the high precisions may in fact not be reachable at all. In those situations, the execution time was set to $+\infty$.

	Average rank of the time to reach a given precision						
Algorithm	1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
CONESTA	3.3 –	2.7 –	2.2 –	**1.6** –	**1.3** –	**1.0** –	**1.0** –
ADMM	2.9	**2.1**	**1.9**	1.8	1.7 $>$	1.8 $>$	1.5 $>$
EGM	**1.8**$<$	2.2	2.0	2.3 $>$	2.3 $>$	2.2 $>$	1.7 $>$
FISTA large μ	4.7$>$	6.0 $>$	4.6 $>$	3.4 $>$	2.7 $>$	2.2 $>$	1.7 $>$
FISTA Chen's μ	6.2$>$	5.0 $>$	4.4 $>$	3.4 $>$	2.7 $>$	2.2 $>$	1.7 $>$
Inexact FISTA	6.2$>$	5.4 $>$	4.3 $>$	3.3 $>$	2.7 $>$	2.2 $>$	1.7 $>$

Tab. II indicates that EGM is the fastest algorithm for the lowest (*i.e.*, largest) precision, $\varepsilon = 1$, and it significantly outperformed CONESTA. ADMM was the second fastest algorithm, but it was not significantly faster than CONESTA. CONESTA was the third fastest and significantly outperformed the Inexact FISTA and FISTA with either μ.

The average convergence rank of CONESTA improved for the higher precision levels. However, it remained the third fastest algorithm for levels $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-2}$, after ADMM and the EGM, but was now no longer significantly slower than either algorithm. However, superiority over all FISTA with Nesterov's smoothing and Inexact FISTA remained and became significant. For higher (smaller than 10^{-3}) precisions, CONESTA outperformed all other solvers. Its superiority was significant for all precisions, except for the comparison with ADMM at $\varepsilon = 10^{-3}$.

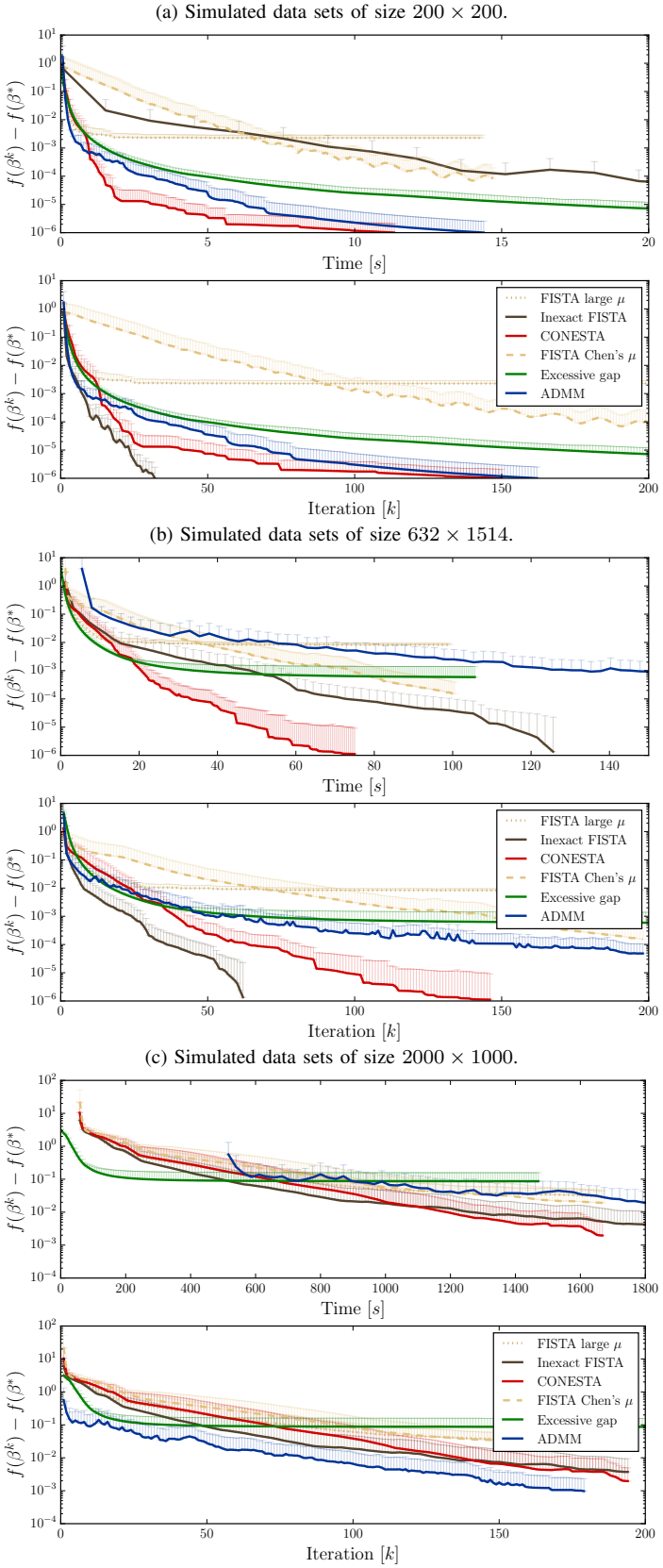

Fig. 1: For datasets of different sizes: (a) 200 × 200, (b) 632 × 1514, (c) 2000 × 1000, the y-axis shows the median error (+MAD), over all experimental settings, as a function of the computational time (top plot) and the number of iterations (bottom plot).

Fig. 1 presents the median error (*i.e.*, the median over $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*)$, and the median absolute deviation, MAD), over all experimental settings (Tab. I) of correlation, sparsity and signal-to-noise ratio, as a function of the time (top plot) and the number of iterations (bottom plot). Results are shown separately for each different sizes of data set: (a) 200×200, (b) 632×1514 and (c) 2000×10000. Note that we did not perform any particular code optimization in order to fairly measure the time difference between the algorithms. Moreover, many solvers share large portions of code, especially FISTA, Inexact FISTA and CONESTA. All solvers are provided by the ParsimonY Python library (see Appendix Sec. A). The experiments were conducted on a Linux-based operating system (Ubuntu 16.04 LTS) using a single core of an Intel i7-5600U@2.6 GHz processor with 16 GB of RAM.

The figure confirms the efficacy of EGM for the lowest precision, at least for the small and medium size data sets. Then, on all data sets, EGM appears to be slow to reach higher precisions. ADMM appears to be particularly fast in the beginning where it converges quickly to the middle precision levels, after which it begins to level out and be outperformed by CONESTA. This ADMM behavior is well-known and is discussed in *e.g.* [7]. The Inexact FISTA performs impressively when only looking at the number of iterations. However, if we consider the time that takes in account the cost of the inner loop for Inexact FISTA, it is almost always outperformed by CONESTA at the higher precisions ($\varepsilon < 10^{-1}$), which correspond to tens of outer FISTA iterations and hundreds of approximation iterations.

CONESTA demonstrates a stable behavior: whatever the setting of the data set, it is always among the best solvers for low precisions and it outperforms others solvers for higher precisions ($\varepsilon \leq 10^{-3}$).

B. Experiments on a structural MRI data set

We applied the solvers on a structural MRI data set of 199 subjects from the Alzheimer's Disease Neuroimaging Initiative (ADNI, http://adni.loni.usc.edu/) cohort. We included 119 controls and 80 patients with mild cognitive impairment (MCI) that converted to AD within 800 days, see Supplementary (Sec. SM 4) for details. Our goal was to predict the subjects' ADAS (AD Assessment Scale-Cognitive Subscale) score ($\boldsymbol{y}$), measured 800 days after brain image acquisition. Images were segmented (SPM8) and spatially normalized into a template (voxels size: 1.5 mm isotropic) using DARTEL [2]. Gray matters (GM) probability maps were modulated with the Jacobian determinants of the nonlinear deformation field. 286 214 voxels of GM were retained and concatenated to form the input data $\boldsymbol{X}$.

1) Relevance of the TV penalty in the context of neuroimaging: Before comparing convergence speed of the solvers, this section demonstrates the improvements obtained with structured sparsity (ℓ_1 and TV) in the context of neuroimaging. We compared Lasso (ℓ_1), Ridge regression (ℓ_2), elastic net ($\ell_1 + \ell_2$), with and without the TV penalty, using 5-fold cross-validation (5CV), in terms of (i) prediction performance (coefficient of determination, R-squared, R^2); and (ii) more importantly, the stability of the $\boldsymbol{\beta}$ maps against variations of the learning samples. Indeed, clinicians expect that the identified neuroimaging predictive signature, *i.e.* the non-zero weights of the $\boldsymbol{\beta}$ map, to be similar if other patients with similar clinical conditions would have been used.

We used two similarity measures to assess the stability of those $\boldsymbol{\beta}$ maps across the re-sampling. First, we computed the mean correlation between pairs of $\boldsymbol{\beta}$ maps obtained across the 5CV: pairwise correlations were transformed into z-scores using Fisher's z-transformation, averaged and finally transformed back into an average correlation, denoted $\overline{r}_{\boldsymbol{\beta}}$. Second, we computed an inter-rater agreement measure: the Fleiss' κ statistics, to assess the agreement between supports (non-zero weights of the $\boldsymbol{\beta}$ maps) recovered by the different penalties across the 5CV. The weights of the five $\boldsymbol{\beta}$ maps were partitioned into three categories according to their signs. This led to negative, positive and out-of-support voxels. The Fleiss' kappa statistic, $\kappa_{\boldsymbol{\beta}}$, was then computed for the five raters. As the 5CV folds share 60 % of their training samples, no unbiased significance measure can be directly obtained from the $\overline{r}_{\boldsymbol{\beta}}$ and $\kappa_{\boldsymbol{\beta}}$ statistics. Thus, we used permutation testing to assess empirical p-values.

TABLE III: Predictive performances: R-squared (R^2) and stability of $\boldsymbol{\beta}$ maps measured as the average correlation between $\boldsymbol{\beta}$ map pairs ($\overline{r}_{\boldsymbol{\beta}}$, using Fisher's z-transformation) and the inter-rater measure ($\kappa_{\boldsymbol{\beta}}$, Fleiss' κ statistic) of agreement between the supports recovered across the 5CV. Significance notations: ***: $p \leq 10^{-3}$, **: $p \leq 10^{-2}$, *: $p \leq 0.05$.

	Prediction		Stability of $\boldsymbol{\beta}$ maps			
Method	R^2	Δ	$\overline{r_{\boldsymbol{\beta}}}$	Δ	$\kappa_{\boldsymbol{\beta}}$	Δ
ℓ_1	0.50*		0.58***		0.29***	
$\ell_1 + TV$	0.52**	$+0.02^{**}$	0.67***	$+0.09^{***}$	0.36***	$+0.07^{***}$
$\ell_1 + \ell_2$	0.53*		0.58***		0.33***	
$\ell_1 + \ell_2 + TV$	0.51**	-0.02	0.65***	$+0.07^{***}$	0.35***	$+0.02^{***}$

The predictive performances, displayed in Tab. III, were not markedly improved with the TV penalty. However, using TV in conjunction with the ℓ_1 penalty was found to significantly improve predictive performance as compared with the usage of ℓ_1 alone.

Fig. 2 demonstrates that the TV penalty provides a major breakthrough regarding support recovery of the predictive brain regions. Contrary to the elastic net penalty that highlights scattered and seemingly meaningless patterns, when using the elastic net penalties with TV (*i.e.*, $\ell_1 + \ell_2 + TV$) the parameter map is smooth and corresponds to brain regions known to be involved in AD [16]. The most important cluster of non-zero weights, found in the left hippocampus and the ventricular enlargement, was identified. The right panel of Fig. 2 confirms that TV produces more stable supports than when using only elastic net. Without TV, some (few) voxels within the hippocampus were at most selected three times during the 5CV rounds. These results demonstrate the ineffectiveness of elastic net in the context of biomarker identification. Conversely, with the TV penalty, voxels within the hippocampus were always selected. The measures of the stability of $\boldsymbol{\beta}$ maps presented

in Tab. III demonstrated that TV significantly improves the reproducibility of the parameter maps, leading to meaningful predictive signatures.

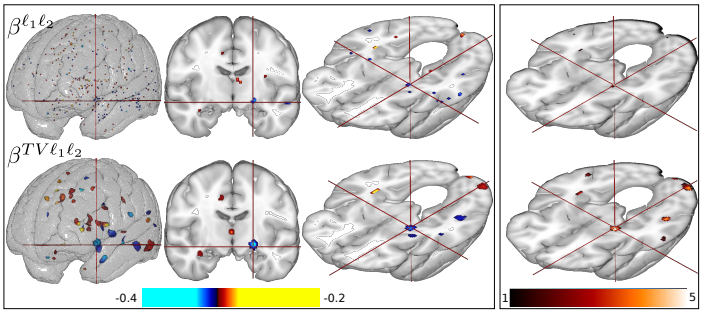

Fig. 2: Left panel: weight map found with elastic net ($\boldsymbol{\beta}^{\ell_1\ell_2}$) and with the elastic net and TV penalties ($\boldsymbol{\beta}^{\ell_1\ell_2 TV}$) found using CONESTA. Right panel: counts of the number of times each voxels were part of the support (non-zero weights) across the 5CV.

2) Comparisons of the convergence speed of the minimization algorithms: After having demonstrated the relevance of TV for extracting stable signatures, we revisit the comparison of the convergence speed of CONESTA to that of the state-of-the-art solvers. In this case: EGM, FISTA with two different fixed μ and Inexact FISTA. Note that ADMM was excluded in this example. The version in [29] is made for 1D underlying data, and the authors did not adapt the method to the case of 2D or 3D underlying data. Note also that without the particular form of the $\boldsymbol{A}^\top\boldsymbol{A}$ matrix (it is tridiagonal in the $\ell_1 + \mathrm{TV}_{1D}$ case), ADMM would perform much slower, because it requires us to solve a linear system in each iteration. This is thus a major drawback of ADMM, that it cannot easily be adapted to arbitrary complex penalties.

First, we fixed a desired precision, $\epsilon = 10^{-6}$, that was used as the stopping criterion for all algorithms. It was also used to derive the smoothing parameter for FISTA with fixed μ. We ran CONESTA and Inexact FISTA until they reached a precision of 10^{-7}, evaluated with the duality gap. The smallest value of $f(\boldsymbol{\beta}^k)$ was considered as the global minimum $f(\boldsymbol{\beta}^*)$ used to compute the errors $f(\boldsymbol{\beta}^k) - f(\boldsymbol{\beta}^*)$ in Tab. 3 and Tab. IV.

TABLE IV: Execution time ratios of each state-of-the-art algorithm over the time required by CONESTA to reach the same precision.

	Time ratio over CONESTA to reach a given precision						
Algorithm	10^1	10^0	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
EGM	0.97	0.39	0.24	1.05	1.08	1.74	3.64
Inexact FISTA	1.13	1.92	2.64	3.97	4.38	5.45	6.07
FISTA Chen's μ	2.7	10.65	17.5	29.95	16.72	13.47	10.45
FISTA large μ	1.04	0.97	1.12	1.14	—	—	—

Fig. 3 and Tab. IV shows that FISTA with fixed μ is either too slow (Chen's μ) or, as expected, does not reach the desired precision (as with the large μ). However, CONESTA compete with FISTA with large μ and EGM during the first iterations. This demonstrates two points: CONESTA dynamically picked an efficient (large enough) smoothing parameter and that the gap stopping criterion, used in the nested FISTA loop, allows us to stop before reaching a plateau, and thus to quickly change to a smaller μ (illustrated with dots in Fig. 3 where the continuation steps occurred).

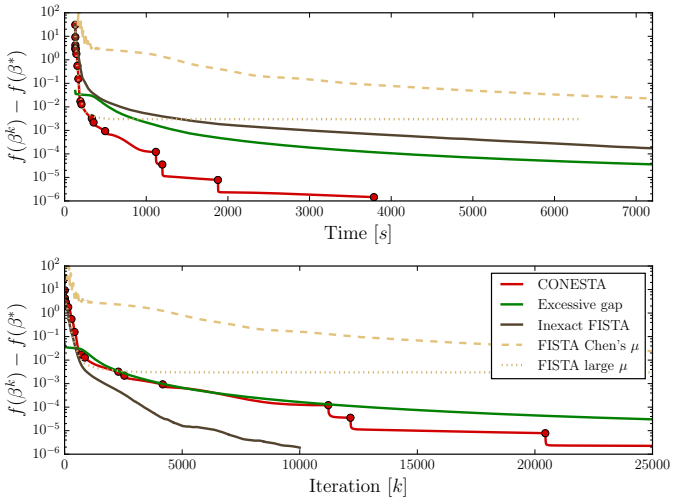

Fig. 3: The error as a function of the computational time (top plot) and the number of iterations (bottom plot). The vertical axis is plotted on a logarithmic scale. Dots on the CONESTA curve indicate where the continuation steps take place, *i.e.* where the dynamic selection of a new smoothing parameter happened.

Fig. 3, bottom panel, shows the fast convergence of Inexact FISTA as a function of the number of iterations. However, the top panel of Fig. 3 shows that it is always considerably slower, in terms of the execution time, compared to the EGM or CONESTA. Tab. IV reveals that Inexact FISTA is 4.38 times slower than CONESTA to reach an error of 10^{-3} and this difference in speed increases with higher precisions. This demonstrates the hypothesis we stated in the introduction, that Inexact FISTA becomes slower after many iterations due to the necessity to decrease the precision faster than $1/k^4$ (k being the number of FISTA iterations), in the approximation.

As a conclusion, Fig. 3 and Tab. IV illustrate that on high-dimensional MRI data sets, CONESTA outperformed all other algorithms for precisions higher than $\varepsilon \leq 10^{-2}$.

3) Required precision and its gap estimate: The figure Fig. 4, top panel, indicates that for precisions higher than 10^{-3} (using both the duality gap estimation of the precision and the true precision), the similarity of coefficient maps to the true solution is reaching a plateau. An early stopping at $\varepsilon = 10^{-2}$ would provide a different solution than the expected one using either measures of precision: $\mathrm{corr}(\boldsymbol{\beta}^k, \boldsymbol{\beta}^*) = 0.92$ with the gap estimate and $\mathrm{corr}(\boldsymbol{\beta}^k, \boldsymbol{\beta}^*) = 0.45$ with the true precision. Moreover the figure demonstrates the relevance of the duality

gap as a stopping criterion: stopping the convergence at 10^{-3}, using the duality gap, provides a map with a 0.97 correlation with the true solution. The bottom panel shows that less than 10^4 iterations are sufficient to reach the target precision of 10^{-3}, which is less than 30 minutes of computation. It also shows that, for useful precisions ($\varepsilon \leq 10^{-1}$), the duality gap is an accurate upper-bound of the true error.

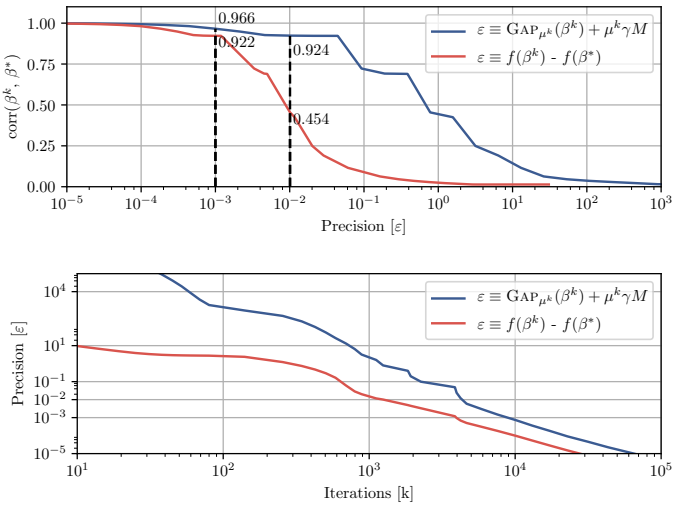

Fig. 4: Top panel: Correlation between the coefficient maps $\boldsymbol{\beta}^k$ and the true solution $\boldsymbol{\beta}^*$ as a function of the true precision (red line) and precision estimated with the duality gap. The true solution has been estimated by running 10^6 iterations of CONESTA and Inexact FISTA. Bottom panel: True precision (red line) and precision estimated with the duality gap (blue line) as a function of the number of iteration. 10^4 iterations are sufficient to reach and outperform the required precision of 10^{-3}.

V. Conclusion

We investigated in this paper an optimization problem where a linear regression model was combined with several convex penalties, including a structured penalty under very general assumptions on the structure and on the smooth loss function.

We proposed an algorithm, CONESTA, that aims to overcome a set of obstacles from current state-of-the-art algorithms, notably non-separability and non-smoothness of penalties. Our results show that the proposed algorithm possesses both desirable theoretical convergence guarantees and practical scalability properties under various settings involving complex structured constraints and high dimensionality. The robustness and scalability of CONESTA is a key feature to processing high-dimensional whole-brain neuroimaging data. Specifically, the main benefits of CONESTA compared to state-of-the-art methods can be summarized in the following points:

1) CONESTA has a convergence rate of $\mathcal{O}(1/\varepsilon)$, an improvement over FISTA with fixed smoothing whose rate is $\mathcal{O}(1/\varepsilon) + \mathcal{O}(1/\sqrt{\varepsilon})$ [5], [9].
2) CONESTA outperformed (in terms of execution time and precision of the solution) several state-of-the-art optimization algorithms on both simulated and neuroimaging data.
3) CONESTA is a robust solver that resolves practical problems affecting other solvers in very high dimensions encountered in neuroimaging. For instance: (i) The EGM does not allow true sparsity, nor complex loss functions. (ii) The Inexact FISTA converges faster (in terms of the number of iterations) compared to CONESTA. However, as observed on the high-dimensional MRI dataset, after hundreds of iterations, solving the subproblem using an inner FISTA loop makes it much slower (*e.g.*, it took 4 times longer to reach $\varepsilon < 10^{-3}$) compared to CONESTA. (iii) ADMM is often not robust to the exact choice of the parameters, and does not scale well to arbitrary complex penalties or *e.g.* 3D images. Conversely, CONESTA requires only a global precision and, thanks to the duality gap, it will dynamically adapt the decreasing smoothing sequence.
4) The algorithm that we propose is able to include any combination of terms from a variety of smooth and non-smooth penalties with smooth losses. For example, in [11] we exploited the versatility of CONESTA as a building block to solve a Principal Components Analysis problem with both TV and elastic-net penalties. CONESTA does not require the proximal operators or any other auxiliary minimization problem, contrary to many other similar state-of-the-art algorithms. For instance, any convex penalty in the form of a sum of ℓ_p-norms of linear operators applied on all or on a subset of all variables could be used with this algorithm together with any smooth loss function for which we know the gradient.
5) As a result of the proposed duality gap, CONESTA has a rigorous stopping criterion that allows a desired precision to be reached with theoretical certainty.

Finally, CONESTA has been designed to work well with warm restarts. Indeed, the whole smoothing sequence is dynamically adapted to the distance to the global minimum. This is a crucial feature for a solver that handles high-dimensional neuroimaging problems, where the user will explore a grid of possible penalty parameters. Since a first solution obtained with a set of parameter is similar to that sought with a slight modification of a parameter, the first solution will be an efficient starting point to quickly find the solution to the second problem.

Appendix
ParsimonY: structured and sparse machine learning in Python

The algorithms investigated here are provided in the ParsimonY Python library found at https://github.com/neurospin/pylearn-parsimony. ParsimonY is a Python library that intends to be compliant with the scikit-learn machine learning library, and provides efficient solvers for a very general class of optimization problems including many group-wise penalties (allowing overlapping groups) such as Group Lasso and TV. Listing 1 shows an implementation of the solution to a problem

with elastic net and TV (default solver is CONESTA) assuming a data set X of size $199 \times 286\,217$ where a row contains the age, gender, education and the $286\,214$ GM voxels of each of the 199 patients. Here we left the first three demographic columns un-penalized (`penalty_start=3`), (defaults is 0, all columns are penalized). The neuroimaging dataset can be found at ftp://ftp.cea.fr/pub/unati/brainomics/papers/ols_nestv.

Listing 1: Elastic net and TV with the ParsimonY library

```
import parsimony.functions.nesterov.tv as tv
from parsimony.estimators import LinearRegressionL1L2TV

mask_ima = nibabel.load("mask.nii")
Atv = tv.linear_operator_from_mask(mask_ima.get_data())
mod = LinearRegressionL1L2TV(l1=0.33, l2=0.33, ltv=0.33,
    A=Atv, penalty_start=3)
mod.fit(X, y)  # Fit the model
mod.predict(X)  # Predict scores
```

REFERENCES

[1] Mohammad R. Arbabshirani, Sergey Plis, Jing Sui, and Vince D. Calhoun. Single subject prediction of brain disorders in neuroimaging: Promises and pitfalls. *NeuroImage*, March 2016.

[2] John Ashburner and Karl J. Friston. Unified segmentation. *NeuroImage*, 26(3):839–851, July 2005.

[3] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Convex optimization with sparsity-inducing norms. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[4] Amir Beck and Marc Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009.

[5] Amir Beck and Marc Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012.

[6] J.M. Borwein and A.S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, 2006.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, January 2011.

[8] A. Chambolle and Ch. Dossal. On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm". *Journal of Optimization Theory and Applications*, 166(3):968–982, 2015.

[9] Xi Chen, Qihang Lin, Seyoung Kim, Jaime G. Carbonell, and Eric P. Xing. Smoothing proximal gradient method for general structured sparse regression. *The Annals of Applied Statistics*, 6(2):719–752, Jun 2012.

[10] Remi Cuingnet, Emilie Gerardin, Jerme Tessieras, Guillaume Auzias, Stephane Lehericy, Marie-Odile Habert, Marie Chupin, Habib Benali, Olivier Colliot, and Alzheimer's Disease Neuroimaging Initiative. Automatic classification of patients with alzheimer's disease from structural MRI: a comparison of ten methods using the ADNI database. *NeuroImage*, 56(2):766–781, May 2011.

[11] A. de Pierrefeu, T. Lfstedt, F. Hadj-Selem, M. Dubois, R. Jardri, T. Fovet, P. Ciuciu, V. Frouin, and E. Duchesnay. Structured Sparse Principal Components Analysis with the TV-Elastic Net Penalty. *IEEE Transactions on Medical Imaging*, PP(99):1–1, 2017.

[12] Elvis Dohmatob, Michael Eickenberg, Bertrand Thirion, and Gal Varoquaux. Speeding-up model-selection in GraphNet via early-stopping and univariate feature-screening. June 2015.

[13] Elvis Dohmatob, Alexandre Gramfort, Bertrand Thirion, and Gal Varoquaux. Benchmarking solvers for TV-l1 least-squares and logistic regression in brain imaging. *Pattern Recognition in Neuroimaging (PRNI)*, June 2014.

[14] M. Dubois, F. Hadj-Selem, T. Lfstedt, M. Perrot, C. Fischer, V. Frouin, and E. Duchesnay. Predictive support recovery with TV-Elastic Net penalty and logistic regression: An application to structural MRI. In *2014 International Workshop on Pattern Recognition in Neuroimaging*, pages 1–4, June 2014.

[15] Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, March 1940.

[16] Giovanni B. Frisoni, Nick C. Fox, Clifford R. Jack, Philip Scheltens, and Paul M. Thompson. The clinical use of structural MRI in alzheimer disease. *Nature reviews. Neurology*, 6(2):67–77, February 2010.

[17] A. Gramfort, B. Thirion, and G. Varoquaux. Identifying Predictive Regions from fMRI with TV-L1 Prior. In *2013 International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, pages 17–20, June 2013.

[18] Logan Grosenick, Brad Klingenberg, Kiefer Katovich, Brian Knutson, and Jonathan E. Taylor. Interpretable whole-brain prediction analysis with GraphNet. *NeuroImage*, 72:304–321, May 2013.

[19] John-Dylan Hanes (Ed.). Multivariate Decoding and Brain Reading [Special issue]. *NeuroImage*, 56(2):385–850, May 2011.

[20] Johanna Hardin, Stephan Ramon Garcia, and David Golan. A method for generating realistic correlation matrices. *Annals of Applied Statistics*, 7(3):1733–1762, 2013.

[21] Tommy Löfstedt, Vincent Guillemot, Vincent Frouin, Edouard Duchesnay, and Fouad Hadj-Selem. Simulated data for linear regression with structured and sparse penalties: Introducing pylearn-simulate. Accepted in *Journal of Statistical Software*. Preprint: http://hal-cea.archives-ouvertes.fr/cea-00914960.

[22] Julien Mairal. *Sparse coding for machine learning, image processing and computer vision*. PhD thesis, École normale supérieure de Cachan, Nov 2010.

[23] Vincent Michel, Alexandre Gramfort, Gaël Varoquaux, Evelyn Eger, and Bertrand Thirion. Total Variation Regularization for fMRI-Based Prediction of Behavior. *IEEE Transactions on Medical Imaging*, 30(7):1328–1340, 2011.

[24] Yurii Nesterov. Excessive Gap Technique in Nonsmooth Convex Minimization. *SIAM Journal on Optimization*, 16(1):235–249, January 2005.

[25] Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, December 2005.

[26] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Convergence Rates of Inexact Proximal-Gradient Methods for Convex Optimization. In *NIPS'11 - 25 th Annual Conference on Neural Information Processing Systems*, December 2011.

[27] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[28] Gaël Varoquaux, Michael Eickenberg, Elvis Dohmatob, and Bertand Thirion. FAASTA: A fast solver for total-variation regularization of ill-conditioned problems with application to brain imaging. In *Colloque GRETSI*, Lyon, France, September 2015. P. Gonçalves, P. Abry.

[29] Bo Wahlberg, Stephen Boyd, Mariette Annergren, and Yang Wang. An ADMM Algorithm for a Class of Total Variation Regularized Estimation Problems. *ArXiv e-prints*, March 2012.