SpaceNet: Multivariate brain decoding and segmentation

Elvis DOHMATOB
(Joint work with: M. EICKENBERG, B. THIRION, & G. VAROQUAUX)

Inria
PARIETAL
Introducing the model
Brain decoding

- **We are given:**
  - \( n = \# \) scans; \( p = \) number of voxels in mask
  - design matrix: \( X \in \mathbb{R}^{n \times p} \) (brain images)
  - response vector: \( y \in \mathbb{R}^{n} \) (external covariates)

- Need to predict \( y \) on new data.

- Linear model assumption: \( y \approx Xw \)

- We seek to **estimate the weights map, \( w \) that ensures best prediction / classification scores**
The need for regularization

- **ill-posed problem**: high-dimensional ($n \ll p$)
- Typically $n \sim 10 - 10^3$ and $p \sim 10^4 - 10^6$
- We need **regularization** to reduce dimensions and encode practitioner’s priors on the weights $w$
Why spatial priors?

3D spatial gradient (a linear operator)
\[ \nabla : w \in \mathbb{R}^p \longrightarrow (\nabla_x w, \nabla_y w, \nabla_z w) \in \mathbb{R}^{p \times 3} \]

- penalize image grad \( \nabla w \) \( \Rightarrow \) regions

- Such priors are reasonable since brain activity is spatially correlated

- more stable maps and more predictive than unstructured priors (e.g. SVM)

[Hebiri 2011, Michel 2011, Baldassare 2012, Grosenick 2013, Gramfort 2013]
SpaceNet is a family of “structure + sparsity” priors for regularizing the models for brain decoding.

SpaceNet generalizes

- TV [Michel 2001],
- Smooth-Lasso / GraphNet [Hebiri 2011, Grosenick 2013], and
- TV-L1 [Baldassare 2012, Gramfort 2013].
2 Methods
The SpaceNet regularized model

\[ y = Xw + \text{"error"} \]

- Optimization problem (regularized model):

\[
\text{minimize } \frac{1}{2} \| y - Xw \|_2^2 + \text{penalty}(w)
\]

- \( \frac{1}{2} \| y - Xw \|_2^2 \) is the loss term, and will be different for squared-loss, logistic loss, ...
The SpaceNet regularized model

\[ \text{penalty}(\mathbf{w}) = \alpha \Omega_\rho(\mathbf{w}), \text{ where} \]

\[ \Omega_\rho(\mathbf{w}) := \rho \| \mathbf{w} \|_1 + (1 - \rho) \begin{cases} \frac{1}{2} \| \nabla \mathbf{w} \|_2^2, & \text{for GraphNet} \\ \| \mathbf{w} \|_{TV}, & \text{for TV-L1} \end{cases} \]

\[ \alpha \ (0 < \alpha < +\infty) \text{ is total amount regularization} \]

\[ \rho \ (0 < \rho \leq 1) \text{ is a mixing constant called the } \ell_1\text{-ratio} \]

\[ \rho = 1 \text{ for Lasso} \]
The SpaceNet regularized model

\[ \text{penalty}(w) = \alpha \Omega_\rho(w), \text{ where} \]

\[ \Omega_\rho(w) := \rho \|w\|_1 + (1 - \rho) \begin{cases} \frac{1}{2} \|\nabla w\|^2, & \text{for GraphNet} \\ \|w\|_{TV}, & \text{for TV-L1} \end{cases} \]

- \( \alpha (0 < \alpha < +\infty) \) is total amount regularization
- \( \rho (0 < \rho \leq 1) \) is a mixing constant called the \( \ell_1 \)-ratio
  - \( \rho = 1 \) for Lasso
- Problem is \textbf{convex, non-smooth}, and heavily-ill-conditioned.
Interlude: zoom on ISTA-based algorithms

- **Settings**: \( \min f + g; \) \( f \) smooth, \( g \) non-smooth
  \( f \) and \( g \) convex, \( \nabla f \) L-Lipschitz; both \( f \) and \( g \) convex

**ISTA**: \( \mathcal{O}(L_{\nabla f}/\epsilon) \)  
- **Step 1**: Gradient descent on \( f \)
- **Step 2**: Proximal operator of \( g \)

**FISTA**: \( \mathcal{O}(L_{\nabla f}/\sqrt{\epsilon}) \)  
- = ISTA with a “**Nesterov acceleration**” trick!

[Beck Teboulle 2009]
2 FISTA: Implementation for TV-L1

Convergence time in seconds

$\alpha$-ratio

$\ell_1$ ratio

1e-3 1e-3 1e-3 1e-3 1e-3 1e-2 1e-2 1e-2 1e-2 1e-2 1e-1 1e-1 1e-1 1e-1
0.10 0.25 0.50 0.75 0.90 0.10 0.25 0.50 0.75 0.90 0.10 0.25 0.50 0.75

[DOKOMATOBO 2014 (PRNI)]
Augment $X$: $\tilde{X} := [X \ c_{\alpha, \rho} \nabla]^T \in \mathbb{R}^{(n+3p) \times p}$

$\Rightarrow \tilde{X}z^{(t)} = Xz^{(t)} + c_{\alpha, \rho} \nabla(z^{(t)})$

1. **Gradient descent step** (datafit term):
   
   $w^{(t+1)} \leftarrow z^{(t)} - \gamma \tilde{X}^T (\tilde{X}z^{(t)} - y)$

2. **Prox step** (penalty term):
   
   $w^{(t+1)} \leftarrow \text{soft}_{\alpha \rho \gamma}(w^{(t+1)})$

3. **Nesterov acceleration**:
   
   $z^{(t+1)} \leftarrow (1 + \theta^{(t)})w^{(t+1)} - \theta^{(t)}w^{(t)}$

**Bottleneck**: $\sim 80\%$ of runtime spent doing $Xz^{(t)}$!

We badly need speedup!
Whereby we **detect and remove irrelevant voxels** before optimization problem is even entered!
$X^T y$ maps: relevant voxels stick-out

100% brain vol
The $2X^Ty$ maps: relevant voxels stick-out

100% brain vol

50% brain vol
$X^T y$ maps: relevant voxels stick-out

100% brain vol

50% brain vol

20% brain vol
The 20% mask has the 3 bright blobs we would expect to get

... but contains much less voxels ⇒ less run-time
Our screening heuristic

- \( t_p := p \)th percentile of the vector \(|X^T y|\).
- Discard \( j \)th voxel if \(|X_j^T y| < t_p\)

\( k = 100\% \) voxels \hspace{1cm} \( k = 50\% \) voxels \hspace{1cm} \( k = 20\% \) voxels

Marginal screening [Lee 2014], but without the (invertibility) restriction \( k \leq \min(n, p) \).

The regularization will do the rest...
See [DOHMATOB 2015 (PRNI)] for a more detailed exposition of speedup heuristics developed.
Automatic model selection via cross-validation

**regularization parameters:**

\[0 < \alpha_L < \ldots < \alpha_3 < \alpha_2 < \alpha_1 = \alpha_{max}\]

**mixing constants:**

\[0 < \rho_M < \ldots < \rho_3 < \rho_2 < \rho_1 \leq 1\]

Thus \( L \times M \) grid to search over for best parameters

\[
\begin{array}{cccc}
(\alpha_1, \rho_1) & (\alpha_1, \rho_2) & (\alpha_1, \rho_3) & \ldots & (\alpha_1, \rho_M) \\
(\alpha_2, \rho_1) & (\alpha_2, \rho_2) & (\alpha_2, \rho_3) & \ldots & (\alpha_2, \rho_M) \\
(\alpha_3, \rho_1) & (\alpha_3, \rho_2) & (\alpha_3, \rho_3) & \ldots & (\alpha_3, \rho_M) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
(\alpha_L, \rho_1) & (\alpha_L, \rho_2) & (\alpha_L, \rho_L) & \ldots & (\alpha_L, \rho_M)
\end{array}
\]
The final model uses average of the per-fold best weights maps (bagging)

This bagging strategy ensures more stable and robust weights maps
3 Some experimental results
3 Weights: SpaceNet versus SVM

Faces vs objects classification on [Haxby 2001]

Smooth-Lasso weights

TV-L1 weights

SVM weights
Classification scores: SpaceNet versus SVM

![Classification accuracy chart](chart.png)
3 Concluding remarks

- SpaceNet enforces both sparsity and structure, leading to better prediction / classification scores and more interpretable brain maps.

- The code runs in $\sim 15$ minutes for “simple” datasets, and $\sim 30$ minutes for very difficult datasets.
SpaceNet enforces both sparsity and structure, leading to better prediction / classification scores and more interpretable brain maps.

The code runs in \(\sim 15 \text{ minutes}\) for “simple” datasets, and \(\sim 30 \text{ minutes}\) for very difficult datasets.

In the next release, SpaceNet will feature as part of Nilearn [Abraham et al. 2014]
http://nilearn.github.io
Why $X^T y$ maps give a good relevance measure?

- In an orthogonal design, least-squares solution is

$$\hat{w}_{LS} = (X^T X)^{-1} X^T y = (I)^{-1} X^T y = X^T y$$

$\Rightarrow$ (intuition) $X^T y$ bears some info on optimal solution even for general $X$
Why $X^T y$ maps give a good relevance measure?

- In an orthogonal design, least-squares solution is
  $$\hat{w}_{LS} = (X^T X)^{-1} X^T y = (I)^{-1} X^T y = X^T y$$
  $\Rightarrow$ (intuition) $X^T y$ bears some info on optimal solution even for general $X$

- Marginal screening: Set $S = \text{indices of top } k$ voxels $j$ in terms of $|X_j^T y|$ values
  - In [Lee 2014], $k \leq \min(n, p)$, so that
    $$\hat{w}_{LS} \sim (X_S^T X_S)^{-1} X_S^T y$$
  - We don’t require invertibility condition
    $k \leq \min(n, p)$. Our spatial regularization will do the rest!
Why $X^T y$ maps give a good relevance measure?

- In an orthogonal design, least-squares solution is
  \[
  \hat{w}_{LS} = (X^T X)^{-1} X^T y = (I)^{-1} X^T y = X^T y
  \]
  \[\Rightarrow \text{(intuition)} \quad X^T y \text{ bears some info on optimal solution even for general } X\]

- Marginal screening: Set $S = \text{indices of top } k \text{ voxels } j \text{ in terms of } |X_j^T y|$ values

  - In [Lee 2014], $k \leq \min(n, p)$, so that
    \[
    \hat{w}_{LS} \sim (X_S^T X_S)^{-1} X_S^T y
    \]

  - We don’t require invertibility condition $k \leq \min(n, p)$. Our spatial regularization will do the rest!