**Université de Rennes I**
**IMT Atlantique**

# EEG inverse problem

**TP2 - Tikhonov & TV-L1 regularizations**

29 february 2024

**Students:** Franco Martin DI MARIA

franco.di-maria@imt-atlantique.net

# 1 Introduction

A popular approach to obtaining a solution to the inverse problem is to regularize the problem by solving the following optimization problem:

$$\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2^2 + \lambda f(\mathbf{s}) \tag{1}$$

The first term ensures that the solution reconstructs the data well, and the second term, called the regularization term, incorporates assumptions about the sources. The regularization parameter $\lambda$ allows managing the balance between the two terms. Proposing different regularization terms has led to different algorithms. In this project, we focus on regularization terms of the form $\mathrm{L}^2$ ($f(\mathbf{s}) = \|\mathbf{s}\|_2^2$, Tikhonov regularization), exploited in the MNE algorithm (minimum norm estimate), and of the TV-L1 norm ($f(\mathbf{s}) = \|\mathbf{T}\mathbf{s}\|_1 + \alpha\|\mathbf{s}\|_1$), used by the SISSY algorithm (source imaging based on structured sparsity). Here, $T$ is a linear operator implementing the gradient on the surface mesh modeling the cortex. The elements $T_{e,d}$ of $\mathbf{T}, e = 1, \ldots, E, d = 1, \ldots, D$, where $E$ is the number of edges in the mesh, are defined as follows:

$$T_{e,d} = \begin{cases} 1 & \text{if } d = d_{e,1} \\ -1 & \text{if } d = d_{e,2} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $d_{e,1}$ and $d_{e,2}$ are the indices of the dipoles sharing the $e$-th edge.

# 2 Preparation

The objective of (distributed) source localization methods in EEG is to solve the optimization problem 1. For most regularization terms, such as the TV-L1 standard, this requires the use of an iterative algorithm. On the other hand, for the L2 norm, an analytical solution can be deduced.

# 3 MNE Algorithm

This algorithms exploits the regularization terms of the form $\mathrm{L}^2$: $f(\mathbf{s}) = \|\mathbf{s}\|_2^2$, also known as Tikhonov regularization, as follows:

$$\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2^2 + \lambda\|\mathbf{s}\|_2^2 \tag{3}$$

From the minimum norm estimate, the following formula can be derived:

$$\mathbf{s} = \mathbf{A}^{\mathrm{T}} \left(\mathbf{A}\mathbf{A}^{\mathrm{T}} + \lambda\mathbf{I}\right)^{-1} \mathbf{x} \tag{4}$$

## 3.1 Proof

$$\begin{aligned}
\min_s \quad L(s) &= \min_s \quad ||x - As||_2^2 + \lambda||s||_2^2 \\
\frac{d}{ds}L(s) &= 2(-A)^t(x - As) + 2\lambda s \\
0 &= -2A^t(x - As) + 2\lambda s \\
0 &= -A^t x + A^t As + \lambda s \\
A^t x &= (A^t A + \lambda I)s \\
A^t x &= (A^t A + \lambda I)s \\
(A^t A + \lambda I)^{-1} A^t x &= s
\end{aligned}$$

$$s = (A^t A + \lambda I)^{-1} A^t x \tag{5}$$

Equation 5 is useful when $A \in R^{N \times D}$ with $N >> D$, since the resulting matrices are much smaller. Therefore, the cost of computing their inverse is lower.

However, if $N << D$, we can utilize the following inversion lemma to obtain an equivalent solution with smaller matrices:

$$\left(\mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{B}+\mathbf{P}^{-1}\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{R}^{-1}=\mathbf{P}\mathbf{B}^{\mathrm{T}}\left(\mathbf{B}\mathbf{P}\mathbf{B}^{\mathrm{T}}+\mathbf{R}\right)^{-1}$$

with:

$$\begin{aligned} B &= A \\ R &= I \\ P &= \lambda^{-1} I \end{aligned}$$

So that:

$$\begin{aligned} s &= (A^t A + \lambda I)^{-1} A^t x \\ &= \lambda^{-1} I A^t (A \lambda^{-1} I A^t + I)^{-1} x \\ &= \lambda^{-1} A^t (\lambda^{-1} A A^t + \lambda^{-1} \lambda I)^{-1} x \\ &= \lambda^{-1} A^t (\lambda^{-1})^{-1} (A A^t + \lambda I)^{-1} x \\ &= \lambda^{-1} \lambda A^t (A A^t + \lambda I)^{-1} x \\ &= A^t (A A^t + \lambda I)^{-1} x \end{aligned}$$

And so, we found the original equation of MNE algorithm:

$$s = A^t (A A^t + \lambda I)^{-1} x \tag{6}$$

# 4 Implementation MNE

```
[caption={MNE.m},label={lst:MNE}]
function [SOut]=MNE(x,A,lambda)

[N,~]=size(A);

SOut = A' * inv(A * A' + lambda * eye(N)) * x;
```

# 5 Study of MNE algorithm

This section will delve into an exploration of the MNE algorithm across various scenarios.

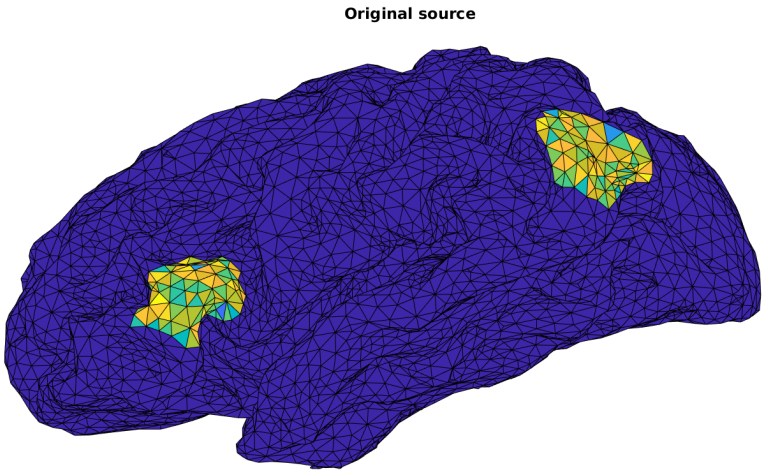



Figure 1: Original source (ground truth)

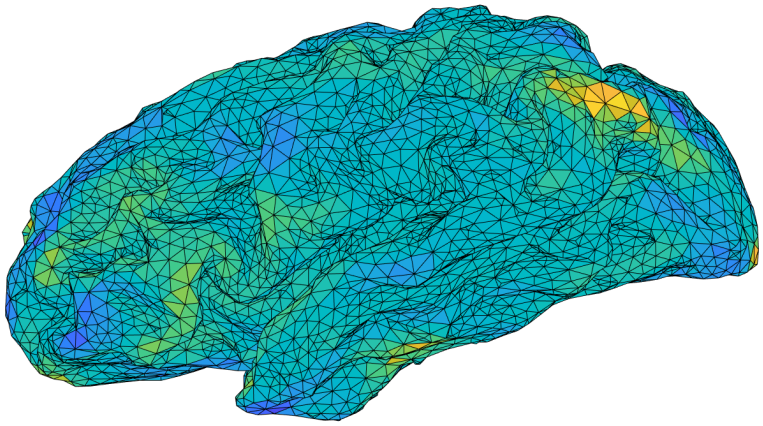
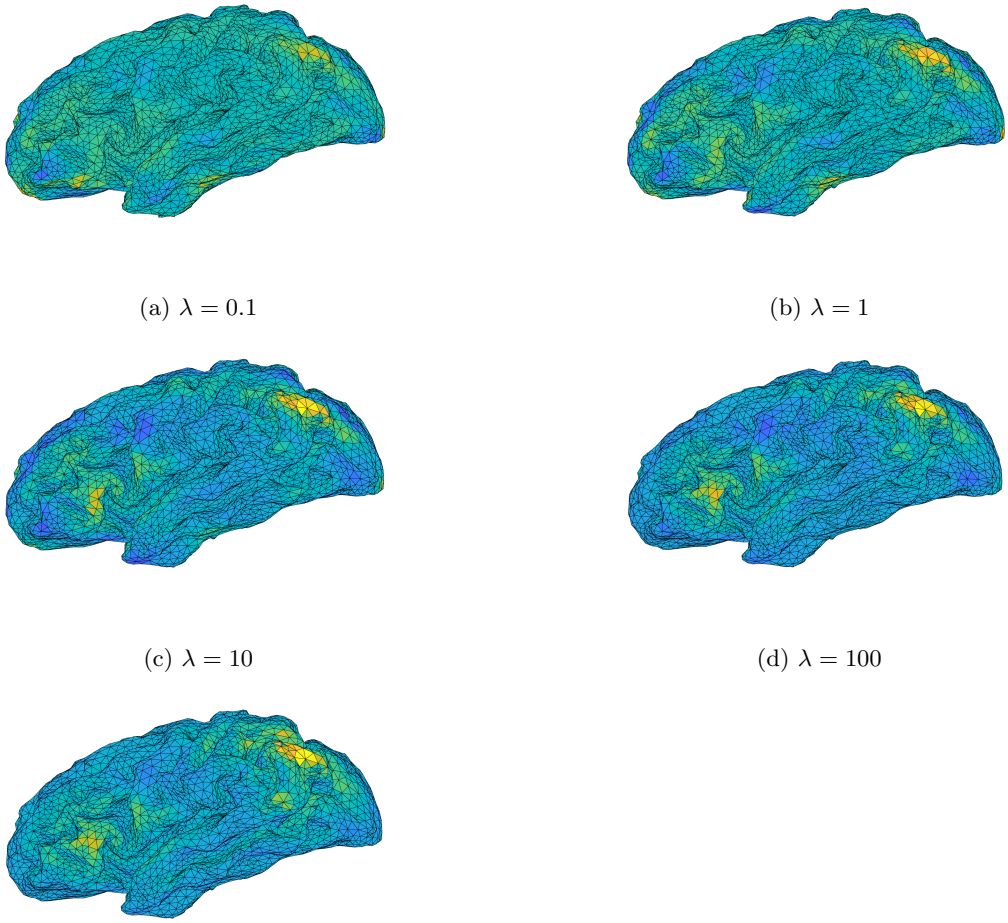


Figure 2: MNE: Source reconstruction with $SNR = 1$ and $\lambda = 1$

## 5.1 Influence of $\lambda$

In this section, we will examine the impact of the regularization parameter $\lambda$ on the outcome of MNE source reconstruction, maintaining a constant signal-to-noise ratio of 1.



(a) $\lambda = 0.1$

(b) $\lambda = 1$

(c) $\lambda = 10$

(d) $\lambda = 100$

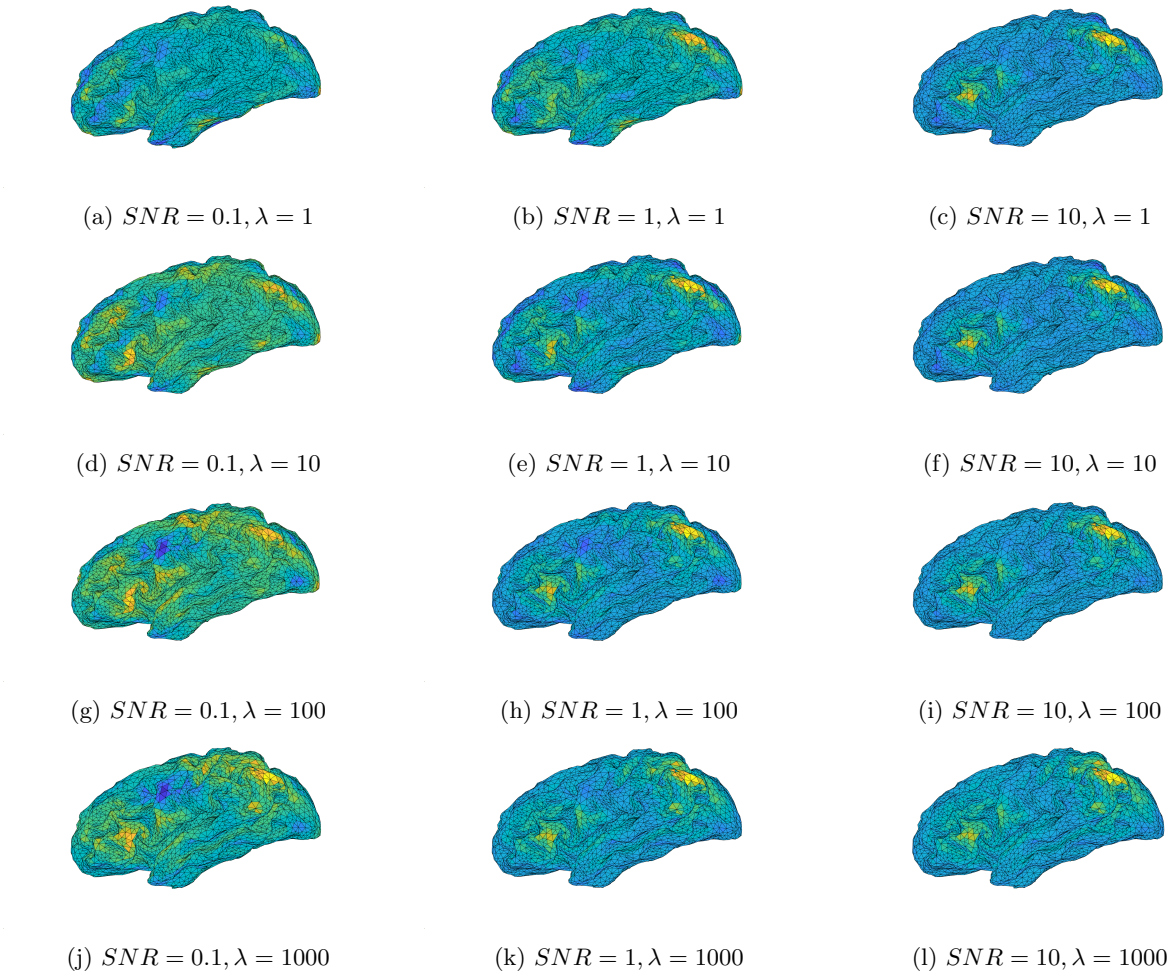
(e) $\lambda = 1000$

Figure 3: MNE: Source reconstructions with different values of $\lambda$ (SNR = 1)

At first glance, Figure 3 suggests that increasing the regularization parameter $\lambda$ enhances the accuracy of source reconstruction. However, upon closer examination, it becomes clear that the source localization in SubFigure 3d is more precise compared to that in SubFigure 3e. This indicates that while a larger value of $\lambda$ is beneficial, it should not exceed a certain threshold.

## 5.2 Influence of SNR and $\lambda$

This section aims to investigate the impact of both the signal-to-noise ratio (SNR) and the regularization parameter $\lambda$ on the outcome of MNE source reconstruction.



(a) $SNR = 0.1, \lambda = 1$ (b) $SNR = 1, \lambda = 1$ (c) $SNR = 10, \lambda = 1$

(d) $SNR = 0.1, \lambda = 10$ (e) $SNR = 1, \lambda = 10$ (f) $SNR = 10, \lambda = 10$

(g) $SNR = 0.1, \lambda = 100$ (h) $SNR = 1, \lambda = 100$ (i) $SNR = 10, \lambda = 100$

(j) $SNR = 0.1, \lambda = 1000$ (k) $SNR = 1, \lambda = 1000$ (l) $SNR = 10, \lambda = 1000$

Figure 4: MNE: Source reconstructions with different values of SNR and $\lambda$

Figure 4 reinforces our earlier hypothesis that having a large value of $\lambda$ is beneficial if it remains below a certain threshold. Additionally, the signal-to-noise ratio (SNR) appears to be directly proportional to the accuracy of the reconstructions: higher SNR values lead to better reconstructions, while lower SNR values result in poorer reconstructions.

## 5.3 Heuristics for finding the optimal $\lambda$

The purpose of this section is to evaluate and compare three distinct heuristics: the L-curve criterion, the Discrepancy principle, and Generalized cross-validation. The objective is to determine the optimal value of $\lambda$ for our specific problem.

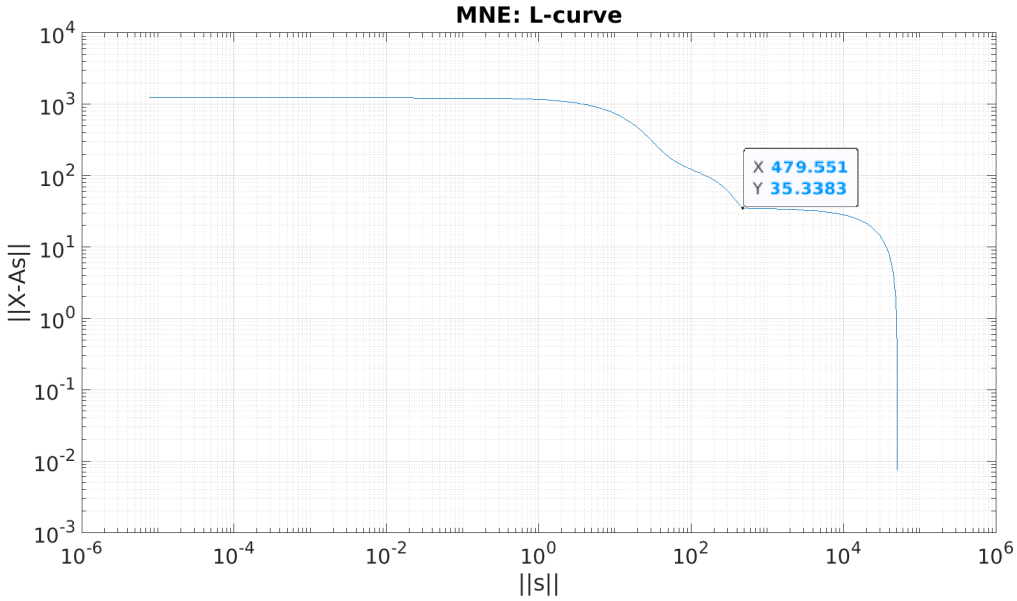



Figure 5: MNE: L-Curve Criterion

By referencing the coordinates of the fold presented in Figure 5, we can derive the corresponding value of $\lambda$ utilized in the computation of $||X - As||_2$ and $||s||_2$ at that particular point, yielding $\lambda = 0.0027$.

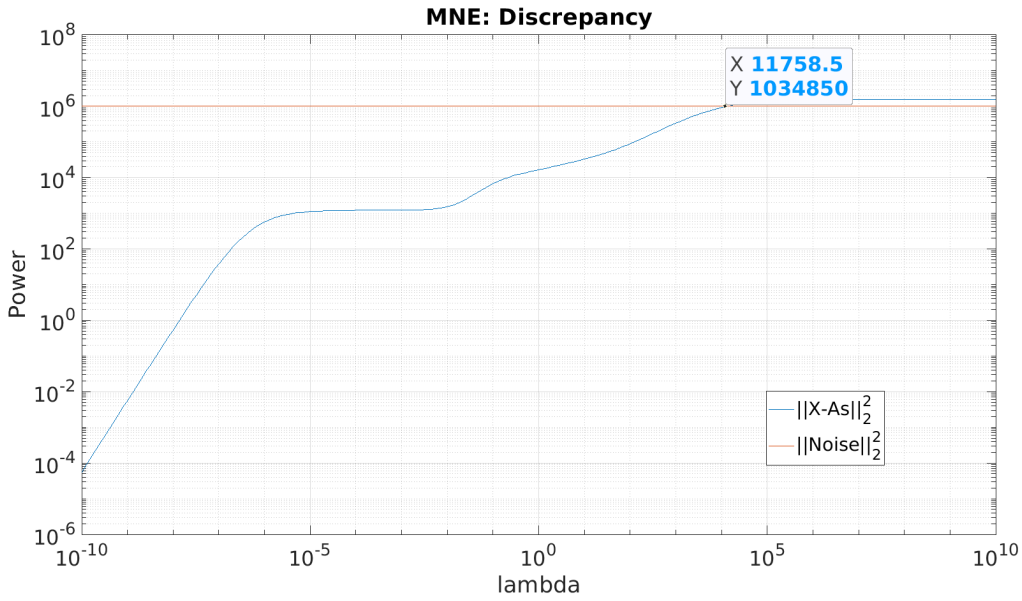



Figure 6: MNE: Discrepancy principle

As depicted in Figure 6, the estimated value of $\lambda$ is 11758.5 .

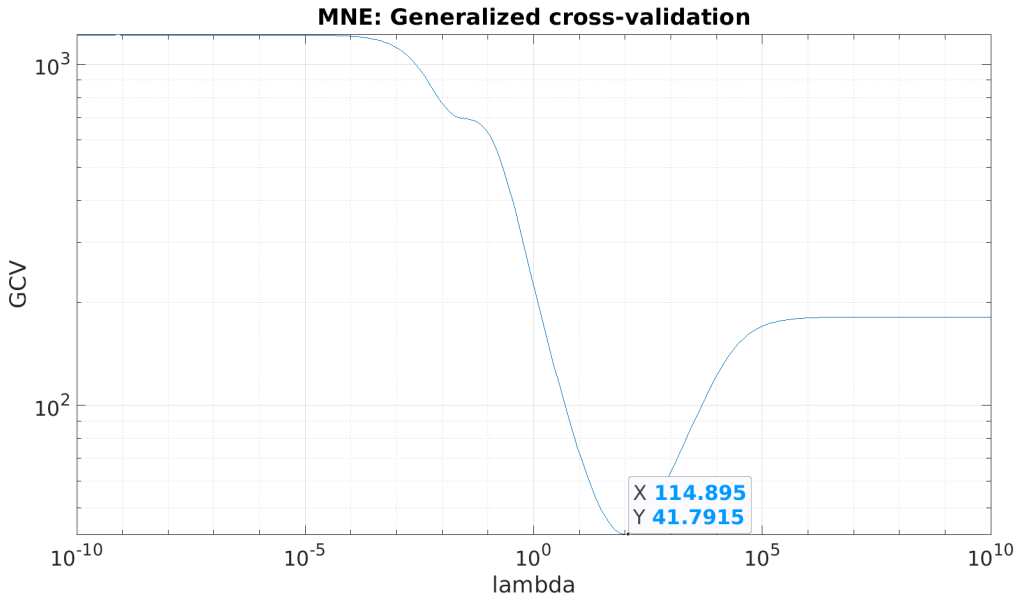



Figure 7: MNE: Generalized cross-validation

As depicted in Figure 6, the estimated value of $\lambda$ is 114.895 .

After assessing the three heuristics under a signal-to-noise ratio of 1, the following observations emerge. Firstly, the L-curve criterion appears inadequate for selecting the optimal regularization parameter $\lambda$. The resulting low value, as depicted in 3a, leads to suboptimal source reconstruction quality.

Conversely, while the discrepancy principle yields a technically correct value, it tends to be excessively large, as evidenced by the results in 3e, which may adversely affect the source reconstruction quality.

In contrast, the global cross-validation method emerges as the most favorable heuristic. It produces a $\lambda$ value that significantly improves source reconstruction without being overly inflated, as demonstrated in SubFigure 3d.

### 5.4 Final choice

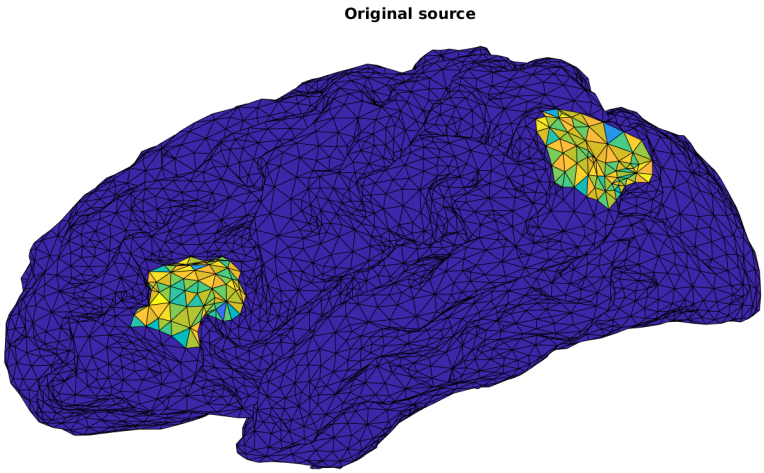



Figure 8: Original source (ground truth)

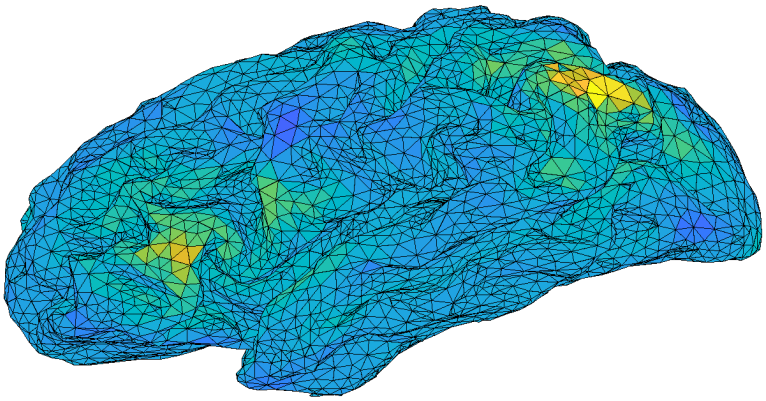


Figure 9: MNE: $\lambda = 114.895$, $SNR = 1$

## 6 SISSY Algorithm

This algorithm use the TV-L1 norm ($f(\mathbf{s}) = \|\mathbf{Ts}\|_1 + \alpha\|\mathbf{s}\|_1$), as follows:

$$\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{As}\|_2^2 + \lambda\left(\|\mathbf{z}\|_1 + \alpha\|\mathbf{y}\|_1\right) \quad \text{s.t.} \quad \mathbf{z} = \mathbf{Ts}, \mathbf{y} = \mathbf{s} \tag{7}$$

Using Lagrange multipliers $\mathbf{u}$ and $\mathbf{v}$, we can arrive to the following formulation of the algorithm:

$$\mathbf{s}^{(k+1)} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A} + \rho\left(\mathbf{T}^{\mathrm{T}}\mathbf{T} + \mathbf{I}\right)\right)^{-1}\left(\mathbf{A}^{\mathrm{T}}\mathbf{x} + \rho\mathbf{T}^{\mathrm{T}}\mathbf{z}^{(k)} + \mathbf{T}^{\mathrm{T}}\mathbf{u}^{(k)} + \rho\mathbf{y}^{(k)} + \mathbf{v}^{(k)}\right) \tag{8}$$

$$\mathbf{z}^{(k+1)} = \mathrm{prox}_{\lambda/\rho}\left(\mathbf{T}\mathbf{s}^{(k+1)} - \frac{1}{\rho}\mathbf{u}^{(k)}\right) \tag{9}$$

$$\mathbf{y}^{(k+1)} = \mathrm{prox}_{\lambda\alpha/\rho}\left(\mathbf{s}^{(k+1)} - \frac{1}{\rho}\mathbf{v}^{(k)}\right) \tag{10}$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \rho\left(\mathbf{z}^{(k+1)} - \mathbf{T}\mathbf{s}^{(k+1)}\right) \tag{11}$$

$$\mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + \rho\left(\mathbf{y}^{(k+1)} - \mathbf{s}^{(k+1)}\right) \tag{12}$$

## 6.1 Proof

To derive the above expression, we will employ the ADMM optimization algorithm, which provides the following augmented Lagrangian to resolve:

$$L(s,z,y,u,v) = \frac{1}{2}||x-As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z-Ts)^t u + (y-s)^t v + \frac{\rho}{2}||Z-Ts||_2^2 + \frac{\rho}{2}||y-s||_2^2 \tag{13}$$

### 6.1.1 $s^{(k+1)}$

$$\begin{aligned}
\min_s \quad L(s) = \min_s \quad & \frac{1}{2}||x-As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 \\
& + (z-Ts)^t u + (y-s)^t v + \frac{\rho}{2}||z-Ts||_2^2 + \frac{\rho}{2}||y-s||_2^2 \\
\frac{d}{ds}L(s) &= -A^t(x-As) - T^t u - v - \rho T^t(z-Ts) - \rho(y-s) \\
0 &= -A^t x + A^t As - T^t u - v - \rho T^t z + \rho T^t Ts - \rho y + \rho s \\
A^t x + T^t u + v + \rho T^t z + \rho y &= A^t As + \rho T^t Ts + \rho s \\
A^t x + \rho T^t z + T^t u + \rho y + v &= (A^t A + \rho T^t T + \rho I)s \\
A^t x + \rho T^t z + T^t u + \rho y + v &= (A^t A + \rho(T^t T + I))s \\
(A^t A + \rho(T^t T + I))^{-1}(A^t x + \rho T^t z + T^t u + \rho y + v) &= s
\end{aligned}$$

Therefore,

$$s^{(k+1)} = (A^t A + \rho(T^t T + I))^{-1}(A^t x + \rho T^t z^{(k)} + T^t u^{(k)} + \rho y^{(k)} + v^{(k)}) \tag{14}$$

## 6.2 $z^{(k+1)}$

$$\min_s \quad L(z) = \min_s \quad \frac{1}{2}||x-As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z-Ts)^t u + (y-s)^t v + \frac{\rho}{2}||z-Ts||_2^2 + \frac{\rho}{2}||y-s||_2^2$$

$L(z)$ is non-differentiable due to the L1 norm in the term $||z||_1$. To find $z^{(k+1)}$, we will complete squares to arrive at an equation for the proximity operator, as follows:

$$
\begin{aligned}
\min_z \quad L(z) &= \min_z \quad \frac{1}{2}||x - As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z - Ts)^t u + (y - s)^t v + \frac{\rho}{2}||z - Ts||_2^2 + \frac{\rho}{2}||y - s||_2^2 \\
&= \min_z \quad \lambda||z||_1 + (z - Ts)^t u + \frac{\rho}{2}||z - Ts||_2^2 \\
&= \min_z \quad \lambda||z||_1 + (z - Ts)^t u + \frac{\rho}{2}(z - Ts)^t(z - Ts) \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}(z - Ts)^t(z - Ts) + (z - Ts)^t u \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}((z - Ts)^t(z - Ts) + \frac{2}{\rho}(z - Ts)^t u) \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}((z - Ts)^t(z - Ts) + 2(z - Ts)^t\frac{1}{\rho}u + \frac{1}{\rho^2}u^t u - \frac{1}{\rho^2}u^t u) \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}((z - Ts)^t(z - Ts) + 2(z - Ts)^t\frac{1}{\rho}u + \frac{1}{\rho^2}u^t u) - \frac{1}{2\rho}u^t u \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}||z - Ts + \frac{1}{\rho}u||_2^2 - \frac{1}{2\rho}u^t u \\
&= \min_z \quad \lambda||z||_1 + \frac{\rho}{2}||z - Ts + \frac{1}{\rho}u||_2^2 \\
&= \min_z \quad \frac{\lambda}{\rho}||z||_1 + \frac{1}{2}||z - Ts + \frac{1}{\rho}u||_2^2 \\
&= \min_z \quad \frac{1}{2}||z - Ts + \frac{1}{\rho}u||_2^2 + \frac{\lambda}{\rho}||z||_1 \\
&= prox_{\lambda/\rho}(Ts - \frac{1}{\rho}u)
\end{aligned}
$$

Therefore,

$$
z^{(k+1)} = prox_{\lambda/\rho}(Ts^{(k+1)} - \frac{1}{\rho}u^{(k)}) \tag{15}
$$

#### 6.2.1 $y^{(k+1)}$

The computation of $y^{(k+1)}$ is similar to the one of $z^{(k+1)}$:

$$
\begin{aligned}
\min_y \quad L(y) &= \min_y \quad \frac{1}{2}||x - As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z - Ts)^t u + (y - s)^t v + \frac{\rho}{2}||z - Ts||_2^2 + \frac{\rho}{2}||y - s||_2^2 \\
&= \min_y \quad \lambda\alpha||y||_1 + (y - s)^t v + \frac{\rho}{2}||y - s||_2^2 \\
&= \textit{... (complete squares) ...} \\
&= \min_y \quad \lambda\alpha||y||_1 + \frac{\rho}{2}||y - s + \frac{1}{\rho}v||_2^2 \\
&= \min_y \quad \frac{\lambda\alpha}{\rho}||y||_1 + \frac{1}{2}||y - s + \frac{1}{\rho}v||_2^2 \\
&= \min_y \quad \frac{1}{2}||y - s + \frac{1}{\rho}v||_2^2 + \frac{\lambda\alpha}{\rho}||y||_1 \\
&= prox_{\lambda\alpha/\rho}(s - \frac{1}{\rho}v)
\end{aligned}
$$

Therefore,

$$
y^{(k+1)} = prox_{\lambda\alpha/\rho}(s^{(k+1)} - \frac{1}{\rho}v^{(k)}) \tag{16}
$$

### 6.2.2 $u^{(k+1)}$

For the computation of $u^{(k+1)}$, the ADMM algorithm specifies that the maximum argument should be taken as follows:

$$\begin{aligned}\max_u \quad L(u) &= \max_u \quad \frac{1}{2}||x - As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z - Ts)^t u + (y - s)^t v + \frac{\rho}{2}||z - Ts||_2^2 + \frac{\rho}{2}||y - s||_2^2 \\ &= \max_u \quad (z - Ts)^t u\end{aligned}$$

In order to maximize the previous expression, we will use the gradient descend method as follows:

$$u^{(k+1)} = u^k + \rho(z^{k+1} - Ts^{k+1}) \tag{17}$$

### 6.2.3 $v^{(k+1)}$

The computation of $u^{(k+1)}$ is similarly to the on for $u^{(k+1)}$:

$$\begin{aligned}\max_v \quad L(v) &= \max_v \quad \frac{1}{2}||x - As||_2^2 + \lambda||z||_1 + \lambda\alpha||y||_1 + (z - Ts)^t u + (y - s)^t v + \frac{\rho}{2}||z - Ts||_2^2 + \frac{\rho}{2}||y - s||_2^2 \\ &= \max_v \quad (y - s)^t v\end{aligned}$$

$$v^{(k+1)} = v^k + \rho(y^{k+1} - s^{k+1}) \tag{18}$$

# 7 Implementation SISSY

```
[caption={SISSY.m},label={lst:SISSY}]
function [s]=SISSY(x, A, T, lambda, alpha, Niter)
rho = 1;
[~, D] = size(A);
y = zeros(D, 1);
v = zeros(D, 1);

[E, ~] = size(T);
z = zeros(E, 1);
u = zeros(E, 1);

P   = sparse(rho*(T.'*T+speye(size(T,2))));
APi = A/P;
L   = chol(eye(size(A,1))+APi*A.','lower');
s1  = A.'*x;

for k=1:Niter
    b = s1+rho*(T.'*(z+u/rho)+y+v/rho);
    s = P\b-APi'*(L'\(L\(APi*b)));
    z = prox_op(T*s - (1/rho)*u, 'L1', lambda/rho);
    y = prox_op(s - (1/rho)*v, 'L1', (lambda*alpha)/rho);
    u = u + rho*(z - T*s);
    v = v + rho*(y - s);
end
```

# 8 Study of SISSY algorithm

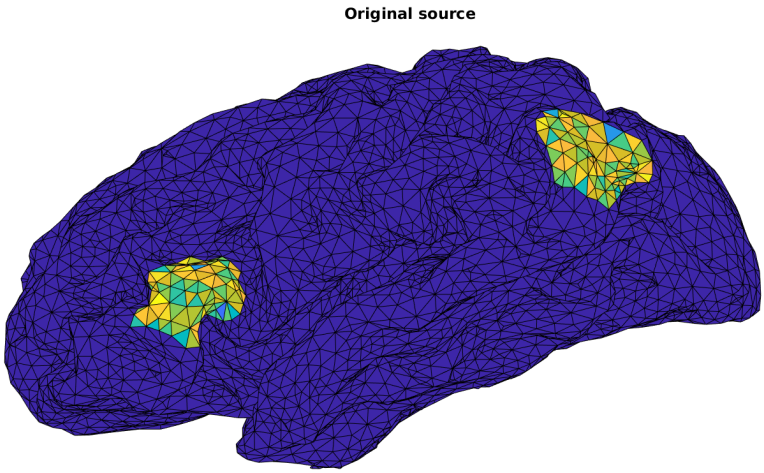



Figure 10: Original source (ground truth)

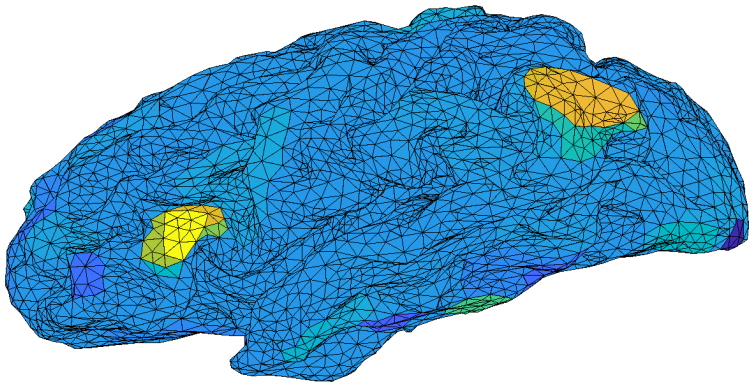
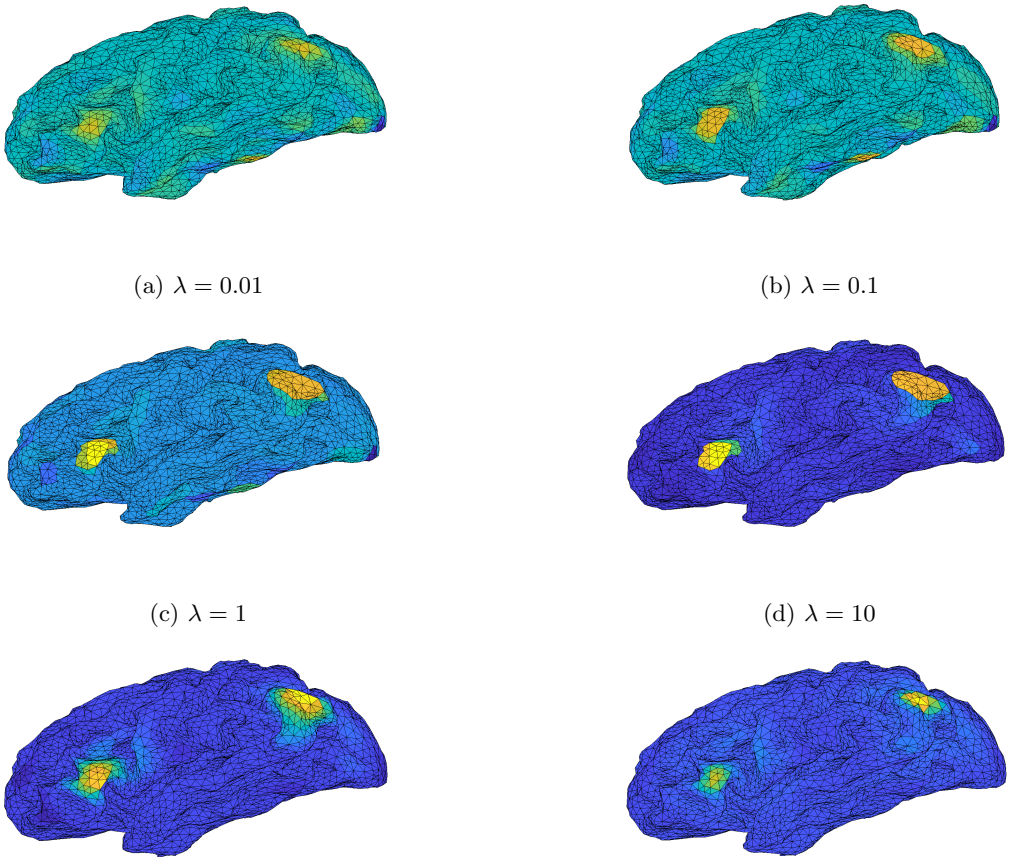


Figure 11: SISSY: Source reconstruction with $SNR = 10$, $\lambda = 1$ and $\alpha = 0.1$

## 8.1 Influence of $\lambda$

In this section, we will examine the impact of the regularization parameter $\lambda$ on the outcome of SISSY source reconstruction, maintaining a constant signal-to-noise ratio of 1.



(a) $\lambda = 0.01$ (b) $\lambda = 0.1$

(c) $\lambda = 1$ (d) $\lambda = 10$

(e) $\lambda = 100$ (f) $\lambda = 1000$

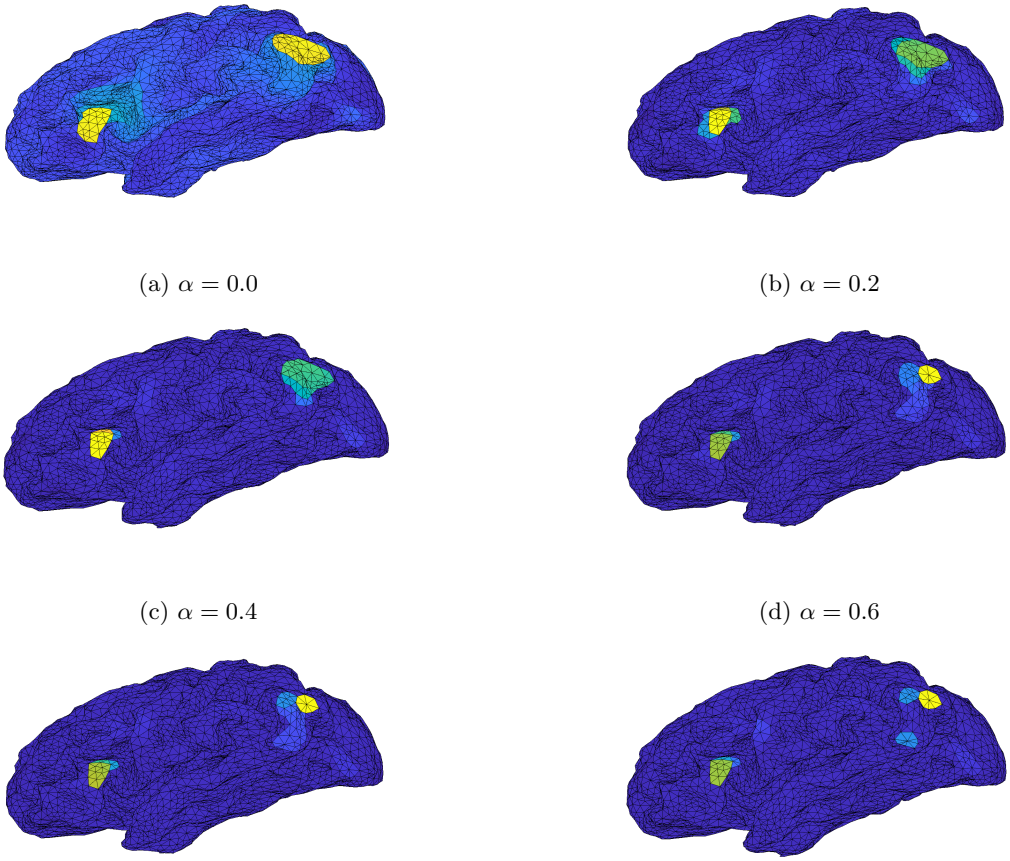
Figure 12: SISSY: Source reconstructions with different values of $\lambda$ ($SNR = 10$, $\alpha = 0.1$)

The plot depicted in Figure 12 illustrates a notable trend: when $\lambda$ is low, the SYSSy source reconstruction exhibits significant noise. Conversely, for higher values of $\lambda$, the dipoles tend to become less sparse and exhibit increased homogeneity. The optimal balance between sparsity and homogeneity appears to lie somewhere in between these extremes.

For the subsequent analysis, we opt to select $\lambda = 10$ as the visually determined optimal value.

## 8.2 Influence of $\alpha$

Having set $\lambda = 10$, in this section, we will focus on the impact of the parameter $\alpha$ on the outcome of SISSY source reconstruction, maintaining a constant signal-to-noise ratio of 1.



(a) $\alpha = 0.0$ (b) $\alpha = 0.2$

(c) $\alpha = 0.4$ (d) $\alpha = 0.6$

(e) $\alpha = 0.8$ (f) $\alpha = 1.0$

Figure 13: SISSY: Source reconstructions with different values of $\alpha$ ($SNR = 10$, $\lambda = 10$)

The plot depicted in Figure 13 illustrates a clear trend: for lower values of $\alpha$, the SISSY source reconstructions tend to exhibit more noise. Conversely, as $\alpha$ increases, the noise decreases, but the active sources also become smaller. It appears that a value around $\alpha = 0.2$ is suitable for this case, and thus, this is the value we will choose.

In summary, based on our simulation and visual analysis of the results, we determine that the optimal values for $\lambda$ and $\alpha$ are approximately 10 and 0.2, respectively.

## 8.3 L0 restriction criterion

In this section we will use L0 restriction criterion to search for the best $\lambda$, given $\alpha = 0.2$ and $SNR = 10$.

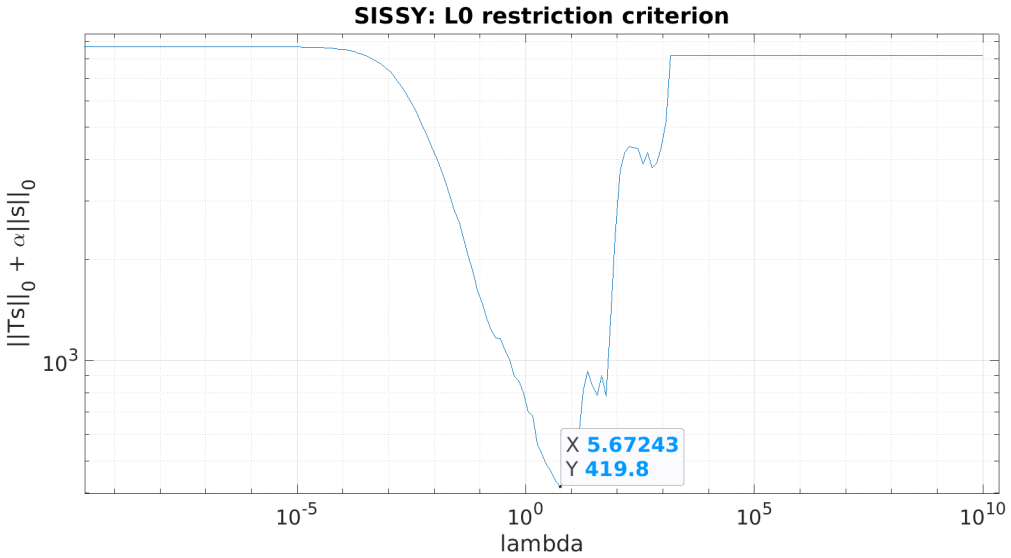



Figure 14: SISSY: L0 restriction criterion

As shown in Figure 14, the optimal value of $\lambda$ given $\alpha = 0.2$ is approximately 5.67243, determined by the provided heuristic. Remarkably, this result is coherent and of the same magnitude as the experimentally chosen value of $\lambda = 10$.

## 8.4 Final choice

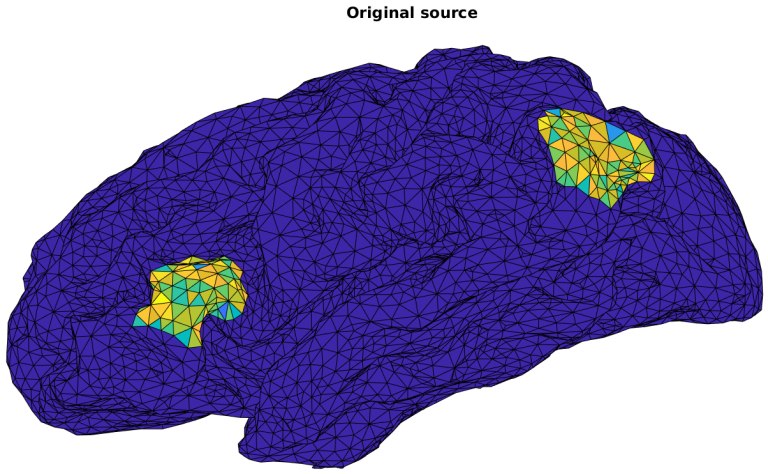



Figure 15: Original source (ground truth)

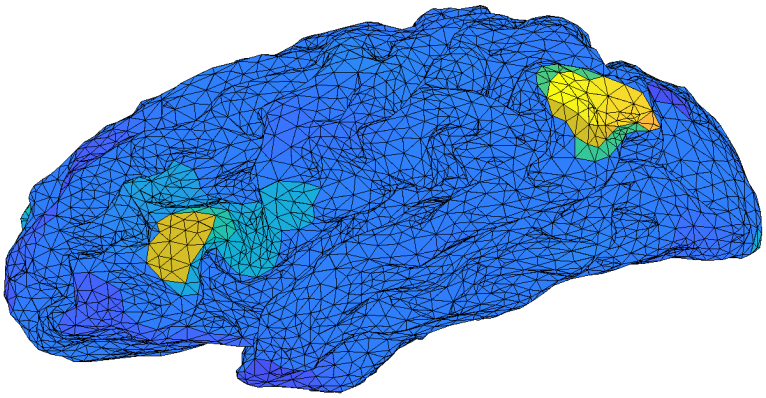


Figure 16: SISSY: $\lambda = 5.67243$, $\alpha = 0.2$, $SNR = 10$

# 9 Conclusion

In conclusion, based on the findings illustrated in Figures 9 and 16, the SISSY algorithm has demonstrated its superiority in achieving superior source reconstruction. The reconstructed sources are both well-localized and sparse, highlighting the effectiveness of the SISSY approach. Nevertheless, MNE still remains as a good approximations whose execution time is better that other algorithms like Gibbs sampling seen in TP1.

# 10 Anne: Simulations implementation

```
[caption={TP2_inverse_problems.m},label={lst:TP2_inverse_problems}]
clear;
close all;
clc;

load TP_data;

%generate linear mixture of source signals
Xs=G*S;

%determine maximum of the signal of interest (here an epileptic spike) to
%apply source loclization algorithms to this time point in the following
[~,id]=max(mean(S,1));

%generate Gaussian random noise
Noise=randn(size(Xs));

%normalize noise
Noise=Noise/norm(Noise,'fro')*norm(Xs,'fro');

%signal to noise ratio
SNR=1;

%generate noisy data according to given SNR
X=Xs+1/sqrt(SNR)*Noise;

%% MNE: Lambda variation
lambda    = logspace(-1,3,5);
fig_count = 1;
for k=1:length(lambda)
    fprintf("Figure=%d, Lambda=%d\n", fig_count, lambda(k))
    Shat=MNE(X(:,id),G,lambda(k));
    figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;
    fig_count = fig_count + 1;
end

%% MNE:Lambda and SNR variation
lambda    = logspace(0,3,4);
SNR       = logspace(-1,1,3);
fig_count = 1;
for j=1:length(SNR)
    X=Xs+1/sqrt(SNR(j))*Noise;
    for i=1:length(lambda)
        fprintf("Figure=%d, SNR=%d, Lambda=%d\n", fig_count, SNR(j), lambda(i))
        Shat=MNE(X(:,id),G,lambda(i));
        figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;
        fig_count = fig_count + 1;
    end
end
%% MNE: L-curve criterion
SNR    = 1;
X      = Xs+1/sqrt(SNR)*Noise;
lambda = logspace(-10,10,200);
for i=1:length(lambda)
    Shat        = MNE(X(:,id),G,lambda(i));
    err_reco(i) = norm(X(:,id) - G*Shat, 2);
    norm_s(i)   = norm(Shat, 2);
end

```

```
figure; loglog(norm_s, err_reco);
title("MNE: L-curve");
ylabel("||X-As||_2");
xlabel("||s||_2");
grid();
set(gca,'fontsize', 24);

[~, idx_err_reco] = min(abs(err_reco - 35.3));
[~, idx_norm_s]   = min(abs(norm_s - 479.6));
fprintf("lambda(||X-As||_2) = %d\n", lambda(idx_err_reco));
fprintf("lambda(||s||_2) = %d\n", lambda(idx_norm_s));
%% MNE: Discrepancy principle
SNR    = 1;
X      = Xs+1/sqrt(SNR)*Noise;
lambda = logspace(-10,10,200);
for i=1:length(lambda)
    Shat         = MNE(X(:,id),G,lambda(i));
    err_reco_2(i) = norm(X(:,id) - G*Shat, 2).^2;
end
norm_n_2 = ones(size(err_reco_2))*norm(1/sqrt(SNR)*Noise, 2).^2;

figure;
loglog(lambda, err_reco_2, 'DisplayName','||X-As||_2^2'); hold on;
loglog(lambda, norm_n_2, 'DisplayName','||Noise||_2^2');
title("MNE: Discrepancy");
ylabel("Power");
xlabel("lambda");
grid();
legend;
set(gca,'fontsize', 24);

fprintf("lambda= %d\n", 1168.5);
%% MNE: Generalized Cross-Validation
%  code to be added in the L-curve iteration
N      = size(G, 1);
SNR    = 1;
X      = Xs+1/sqrt(SNR)*Noise;
lambda = logspace(-10,10,200);
for i=1:length(lambda)
    Shat       = MNE(X(:,id),G,lambda(i));
    err_reco_2 = norm(X(:,id) - G*Shat, 2).^2;
    trace_2    = trace(eye(N) - G*G'*inv(G*G' + lambda(i)*eye(N))).^2;
    GCV(i)     = err_reco_2 / trace_2;
end

figure; loglog(lambda, GCV);
title("MNE: Generalized cross-validation");
ylabel("GCV");
xlabel("lambda");
grid();
set(gca,'fontsize', 24);

[~, idx_GCV] = min(GCV);
fprintf("lambda(GCV) = %d\n", lambda(idx_GCV));

%% MNE: Final choice
SNR    = 1;
X      = Xs+1/sqrt(SNR)*Noise;
lambda = 114.895;
Shat   = MNE(X(:,id),G,lambda);
figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;

```

```
%% SISSY: Lambda variation
SNR     = 10;
X       = Xs+1/sqrt(SNR)*Noise;
T       = variation_operator(mesh,'face');
alpha = 0.1;
Niter = 60;
lambda = logspace(-2,3,6);
fig_count = 1;
for k=1:length(lambda)
    fprintf("Figure=%d, Lambda=%d\n", fig_count, lambda(k))
    Shat=SISSY(X(:,id),G,T,lambda(k), alpha, Niter);  %  function to be implemented
    figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;
    fig_count = fig_count + 1;
end

%% SISSY: Alpha variation
SNR     = 10;
X       = Xs+1/sqrt(SNR)*Noise;
T       = variation_operator(mesh,'face');
lambda = 10;
Niter = 60;
alpha = linspace(0, 1, 6);
fig_count = 1;
for k=1:length(alpha)
    fprintf("Figure=%d, Alpha=%d\n", fig_count, alpha(k))
    Shat=SISSY(X(:,id),G,T,lambda, alpha(k), Niter);
    figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;
    fig_count = fig_count + 1;
end

%% SISSY: L0 restriction criterion
SNR     = 10;
X       = Xs+1/sqrt(SNR)*Noise;
T       = variation_operator(mesh,'face');
alpha = 0.2;
Niter = 60;
lambda = logspace(-5,5,100);
for k=1:length(lambda)
    Shat=SISSY(X(:,id),G,T,lambda(k), alpha, Niter);
    norm_L0_Ts   = sum(T*Shat > 0.01*max(T*Shat));
    norm_L0_s    = sum(Shat > 0.01*max(Shat));
    L0_restriction(k) = norm_L0_Ts + alpha*norm_L0_s;
end

figure; loglog(lambda, L0_restriction);
title("SISSY: L0 restriction criterion");
ylabel("||Ts||_0 + \alpha||s||_0");
xlabel("lambda");
grid();
set(gca,'fontsize', 24);

%% SISSY: Final choice
SNR     = 10;
X       = Xs+1/sqrt(SNR)*Noise;
T       = variation_operator(mesh,'face');
lambda = 5.67243;
alpha  = 0.2;
Niter  = 60;
Shat=SISSY(X(:,id),G,T,lambda, alpha, Niter);
figure; trisurf(mesh.f,mesh.v(:,1),mesh.v(:,2),mesh.v(:,3),Shat); axis off;
```