FastSRM: A fast, memory efficient and identifiable implementation of the shared response model

November 4, 2021

Abstract

The shared response model (SRM) provides a simple but effective framework to analyse fMRI data of subjects exposed to naturalistic stimuli. However when the number of subjects or runs is large, fitting the model requires a large amount of memory and computational power, which limits its use in practice. Furthermore, SRM is not identifiable which makes the shared response difficult to interpret.

In this work, we implement an identifiable version of SRM and show on real data that it improves the stability of the recovered shared response. We then introduce the FastSRM that relies on a dimension reduction step that we prove to be optimal in the sense that working with reduced data does not induce any change in the algorithm trajectory. We show experimentally using synthetic and real fMRI data that FastSRM is considerably faster and more memory efficient than competitive implementations.

The experiments performed in the paper are *fully* reproducible: our code available at `https://github.com/hugorichard/FastSRM` downloads the data, runs the experiments and plot the figures.

1 Introduction

What makes the analysis of functional magnetic resonance imaging (fMRI) data particularly difficult is that the number of features (voxels) is usually much larger than the number of available samples (timeframes). In such cases, dimension reduction methods are precious tools that allow to reduce the computational burden of subsequent data analysis. The shared response model [8] provides a principled way to perform dimension reduction of brain imaging data from subjects performing the same task and is therefore a natural basis for common source identification [20] [21], classification [23] [7][26] or transfer learning [27].

However SRM has initially been designed to work within regions of interest using only few subjects. When using full brain data, computational costs become high. In addition, memory requirements are difficult to meet since the full dataset needs to hold in memory. Another issue of SRM is its non-identifiability (see the proof in [21] Appendix D) which makes it challenging to interpret the extracted shared components as they are defined up to a rotation.

In this work, we introduce FastSRM, an identifiable model that uses an optimal atlas based decomposition to speed up the computations with provably no loss of performance. FastSRM gains identifiability by imposing that the covariance of the shared components is diagonal (see [21] Appendix D for a proof). The gains in speed and memory usage are based on the intuition that a compressed representation of the input should be good enough to find a suitable estimate of the shared components as they live in reduced space. It turns out that there exists an optimal atlas that reduces the data in a such a way that the obtained shared components (and other parameters) are the same as if they were computed on full data. We show that FastSRM brings several practical benefits: on real data, it produces more stable estimates of the shared components than its non-identifiable counter-part and on synthetic as well as on real data, it is much faster and more memory efficient than other implementations that do not make use of the optimal atlas.

Our python implementation is available at `https://github.com/hugorichard/FastSRM`. All our experiments are fully reproducible. Scripts to download the data, run the experiments and plot the figures are included. A continuous integration pipeline runs the tests automatically when any change to the core algorithm is made.

2 Background: the shared response model (SRM)

The shared response model [8] is a multi-view latent factor model. The data $\mathbf{x}_1 \dots \mathbf{x}_m$ are modeled as random vectors following the model:

$$\mathbf{x}_i = A_i \mathbf{s} + \mathbf{n}_i \tag{1}$$

$$A_i^\top A_i = I_p \tag{2}$$

where $\mathbf{x}_i \in \mathbb{R}^v$ is the data of view i, $A_i \in \mathbb{R}^{p \times v}$ is the mixing matrix of view i, $\mathbf{n}_i$ is the noise of view i and $\mathbf{s} \in \mathbb{R}^p$ are the shared components referred to as the *shared response* in fMRI applications.

The mixing matrices A_i are assumed to be orthogonal so that $A_i^\top A_i = I_p$. However, in general the matrix $A_i A_i^\top$ is different from identity. The noise $\mathbf{n}_i$ is assumed to be Gaussian with covariance Σ_i and independent across views. We assume the number of features v to be much larger than the number of components p: $v \gg p$.

The conceptual figure 1 illustrates an application of the shared response model to fMRI data. The mixing matrices are spatial topographies specific to each subjects while the shared components give the common timecourses. In [8, 2], two versions of the shared response model are introduced which we now present.

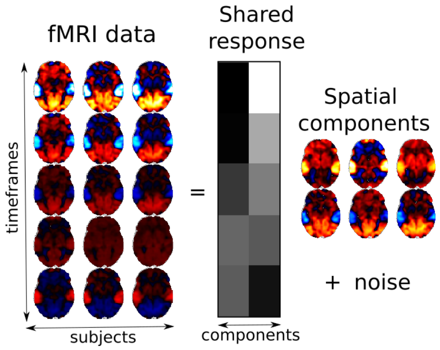

Figure 1: **Shared response model**: The raw fMRI data are modeled as a weighted combination of subject-specific spatial components with additive noise. The weights are shared between subjects and constitute the shared response to the stimuli.

2.1 Deterministic shared response model

Let us consider n observations of $\mathbf{x}_i$ and $\mathbf{s}$ that we stack into matrices $X_i \in \mathbb{R}^{v,n}$ and $S \in \mathbb{R}^{p,n}$.

The deterministic shared response model sees both the mixing matrices A_i and the n observations of the shared response S as parameters to be estimated. The noise variance is fixed to a multiple of identity: $\forall i, \Sigma_i = \sigma^2 I_v$ where σ is an hyper-parameter to choose.

The model is optimized by maximizing the log-likelihood.

The likelihood is given by: $p(\mathbf{x}) = \prod_i \mathcal{N}(\mathbf{x}_i; A_i \mathbf{s}, \sigma^2 I)$ and therefore the empirical expected negative log-likelihood is given up to a constant independent of A_i and S by:

$$\mathcal{L} = \frac{1}{n} \sum_i \|A_i S - X_i\|^2 = \frac{1}{n} \left(\|S\|^2 - 2\langle A_i S, X_i \rangle + \|X_i\|^2 \right) \tag{3}$$

$\mathcal{L}$ is optimized by performing alternate minimization on $(A_1 \dots A_m)$ and S. Note that the hyper-parameter σ does not have an influence on the loss and can therefore be safely ignored.

The gradient with respect to S is given by $\sum_i A_i^\top (A_i S - X_i) = \sum_i (S - A_i^\top X_i)$ yielding the closed form updates:

$$S \leftarrow \frac{1}{m} \sum_i (A_i^\top X_i) \tag{4}$$

From (3), minimizing $\mathcal{L}$ with respect to A_i is equivalent to maximizing $\langle A_i, X_i S^\top \rangle$ and therefore we have:

$$A_i \leftarrow \mathcal{P}(\frac{1}{n} X_i S^\top) \tag{5}$$

where $\mathcal{P}$ is the projection on the Stiefel manifold: $\mathcal{P}(M) = M(M^\top M)^{-\frac{1}{2}}$.

The complexity of Deterministic SRM is in $\tilde{O}(\mathrm{n_{iter}} mpvn)$ where n is the number of samples and $\mathrm{n_{iter}}$ the number of iterations. We monitor the convergence by looking at the ℓ_∞ norm of the gradient. Note that we can monitor the gradient without any increase in complexity. Indeed, after the updates with respect to each mixing matrix have been carried out, only the gradient with respect to S remains: $\sum_i (S - A_i^\top \mathbf{x}_i)$. The algorithm is stopped when the gradient falls below a chosen tolerance.

2.2 Probabilistic SRM

In Probabilistic SRM , $\Sigma_i = \sigma_i^2 I_v$ and the shared components are assumed to be Gaussian $\mathbf{s} \sim \mathcal{N}(0, \Sigma_s)$.

In [8] and [2], Σ_s is only assumed to be definite positive. As already highlighted in introduction, this causes the model to be unidentifiable (see [21] Appendix D for a proof). Enforcing a diagonal Σ_s ensures identifiability (provided the diagonal values are different). So we assume here that Σ_s is diagonal (and refer the interested reader to [8] and [2] for the original formulation of Probabilistic SRM without the diagonal constraint)

The model is optimized via the expectation maximization algorithm. Denoting $\mathbb{V}[\mathbf{s}|\mathbf{x}] = (\sum_i \frac{1}{\sigma_i^2} I + \Sigma_s^{-1})^{-1}$ and $\mathbb{E}[\mathbf{s}|\mathbf{x}] = \mathbb{V}[\mathbf{s}|\mathbf{x}] \sum_i \frac{1}{\sigma_i^2} A_i^\top \mathbf{x}_i$, we have

$$p(\mathbf{x}, \mathbf{s}) = \prod_i \frac{\exp(-\frac{\|\mathbf{x}_i - A_i \mathbf{s}\|^2}{2\sigma_i^2})}{(2\pi\sigma_i^{2v})^{\frac{1}{2}}} \frac{\exp(-\frac{1}{2}\langle \mathbf{s}, \Sigma_s^{-1}\mathbf{s}\rangle)}{(2\pi|\Sigma_s|)^{\frac{1}{2}}} \tag{6}$$

$$= c_1 \exp(-\frac{1}{2}\left(\sum_i \frac{1}{\sigma_i^2}\|\mathbf{x}_i\|^2 - 2\langle \sum_i \frac{1}{\sigma_i^2} A_i^\top \mathbf{x}_i, \mathbf{s}\rangle \right. \tag{7}$$

$$\left. + \sum_i \frac{1}{\sigma_i^2}\|\mathbf{s}\|^2 + \langle \mathbf{s}, \Sigma_s^{-1}\mathbf{s}\rangle \right)) \tag{8}$$

$$= c_2(\mathbf{x}) \exp(-\frac{1}{2}\left(\langle \mathbf{s} - \mathbb{E}[\mathbf{s}, \mathbf{x}], \mathbb{V}[\mathbf{s}|\mathbf{x}]^{-1}(\mathbf{s} - \mathbb{E}[\mathbf{s}|\mathbf{x}])\rangle\right)) \tag{9}$$

where

$$c_1 = \frac{1}{(2\pi\sigma_i^{2v})^{\frac{1}{2}}} \frac{1}{(2\pi|\Sigma_s|)^{\frac{1}{2}}} \tag{10}$$

and

$$c_2(\mathbf{x}) = c_1 \exp(-\frac{1}{2}(\sum_i \frac{1}{\sigma_i^2}\|\mathbf{x}_i\|^2 - \langle \mathbb{E}[\mathbf{s}, \mathbf{x}], \mathbb{V}[\mathbf{s}|\mathbf{x}]^{-1}\mathbb{E}[\mathbf{s}|\mathbf{x}]\rangle)) \tag{11}$$

are independent of $\mathbf{s}$. Therefore,

$$\mathbf{s}|\mathbf{x} \sim \mathcal{N}(\mathbb{E}[\mathbf{s}|\mathbf{x}], \mathbb{V}[\mathbf{s}, \mathbf{x}]) \tag{12}$$

The negative expected completed log-likelihood is given by

$$\mathcal{L} = \sum_i \frac{1}{2} v \log(\sigma_i^2) + \frac{1}{2\sigma_i^2} \mathbb{E}[\|\mathbf{x}_i - A_i \mathbf{s}\|^2] \tag{13}$$

updates are therefore given by:

$$\sigma_i^2 \leftarrow \frac{1}{v}(\mathbb{E}[\|\mathbf{x}_i - A_i \mathbb{E}[\mathbf{s}|\mathbf{x}]\|^2] + \|\mathrm{diag}(\mathbb{V}[\mathbf{s}|\mathbf{x}])\|^2) \tag{14}$$

$$A_i \leftarrow \mathcal{P}(\mathbb{E}[\mathbf{x}_i \mathbb{E}[\mathbf{s}|\mathbf{x}]^\top]) \tag{15}$$

$$\Sigma_s \leftarrow \mathbb{V}[\mathbf{s}|\mathbf{x}] + \mathbb{E}[\mathbb{E}[\mathbf{s}|\mathbf{x}]\mathbb{E}[\mathbf{s}|\mathbf{x}]^\top] \tag{16}$$

It is useful to access the log-likelihood to check the implementation of the algorithm and monitor the convergence. From equation (9), the likelihood is given by:

$$p(\mathbf{x}) = c_2(\mathbf{x}) \int_{\mathbf{s}} \exp(-\frac{1}{2} \left(\langle \mathbf{s} - \mathbb{E}[\mathbf{s}, \mathbf{x}], \mathbb{V}[\mathbf{s}|\mathbf{x}]^{-1}(\mathbf{s} - \mathbb{E}[\mathbf{s}|\mathbf{x}]) \rangle \right)) d\mathbf{s} \tag{17}$$

$$= c_2(\mathbf{x})(2\pi|\mathbb{V}[\mathbf{s}|\mathbf{x}]|)^{\frac{1}{2}} \tag{18}$$

replacing $c_2(\mathbf{x})$ by its expression and taking the log, the expected negative log-likelihood is (up to constants) given by:

$$\begin{aligned}\mathbb{E}[-\log(p(\mathbf{x}))] &= \sum_i \frac{v}{2} \log(\sigma_i^2) + \frac{1}{2}\log(|\Sigma_s|) - \frac{1}{2}\log(|\mathbb{V}[\mathbf{s}|\mathbf{x}]|) \\ &+ \sum_i \frac{1}{2}\frac{1}{\sigma_i^2} \mathbb{E}[\|\mathbf{x}_i\|^2] - \frac{1}{2}\mathbb{E}[\langle \mathbb{E}[\mathbf{s}, \mathbf{x}], \mathbb{V}[\mathbf{s}|\mathbf{x}]^{-1} \mathbb{E}[\mathbf{s}|\mathbf{x}] \rangle] \end{aligned} \tag{19}$$

The complexity of Probabilistic SRM is $\tilde{O}(\mathrm{n}_{\mathrm{iter}} mpvn)$, the same as in Deterministic SRM. We can monitor the convergence by looking at the log-likelihood decrease at each iteration and stop the algorithm when the magnitude of the decrease is below some tolerance. The storage requirements of Deterministic or Probabilistic SRM are in $\tilde{O}(mvn)$ which simply means that the dataset needs to hold in memory.

3 The FastSRM algorithm

3.1 Reducing the computational burden by use of atlases

SRM algorithms use different set of parameters θ to represent the data. In deterministic SRM $\theta = (A_i)_{i=1}^m, S$ where $(A_i)_{i=1}^m$ are the mixing matrices and S are the n observations of the shared response S while in probabilistic SRM $\theta = (A_i)_{i=1}^m, \Sigma_s, (\sigma_i)_{i=1}^m$ where $(A_i)_{i=1}^m$ are the mixing matrices, $(\sigma_i)_{i=1}^m$ the noise levels and Σ_s the components variance.

In fMRI, the classical approach used to reduce the data is to apply an atlas. A deterministic atlas such as [5] is a parcellation of the brain into r regions.

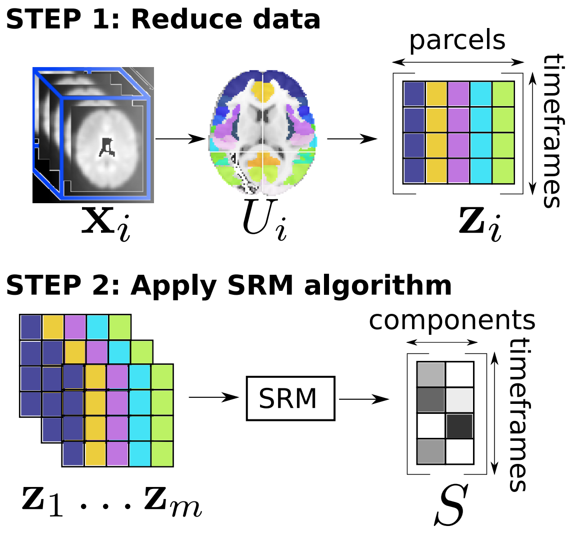

Figure 2: **FastSRM algorithm** In step 1, data $\mathbf{x}_i$ are projected onto an atlas U_i that may depend on the subject i (top). In step 2 a SRM algorithm is applied on reduced data to compute the shared response.

Reducing an image using a deterministic atlas corresponds to averaging the signal within each region of the atlas. A probabilistic atlases such as [9] describes each region as a set of weights across the full brain. Therefore, the image reduction can be done with a matrix product.

In FastSRM we consider a set of view specific atlases $U_i \in \mathbb{R}^{v \times r}$ such that $U_i^\top U_i = I_r$ where r is the number of regions in the atlas. Data are reduced using $\mathbf{z}_i = U_i^\top \mathbf{x}_i$ and an SRM algorithm is applied on data $\mathbf{z}_i$ yielding parameters θ'. The figure 2 illustrates this process.

Note that the parameters obtained with FastSRM θ' are different from the parameters obtained with the corresponding SRM algorithm θ (the unmixing matrices in θ' do not even have the same shape as the unmixing matrices in θ). However, as we will see in the next section, there exists atlases such as the correspondence between θ and θ' can be made explicit.

From a computational stand point, the dimension reduction provides a large reduction in memory usage. Indeed as the original data are seen only once, it is no longer necessary to keep the full dataset in memory (we can load data X_i one after the other and similarly for the atlases U_i). Therefore the memory consumption is only in $\tilde{O}(vn)$ (where v is the number of voxels and n is the number of samples) which is lower than SRM by a factor of m, the number of subjects. The number of subjects is typically between 10 and 1000. This yields a practical benefit: on fMRI datasets with many subjects, one no longer needs a large cluster to run the shared response model but only a modern laptop. Additionally, low memory consumption reduces the risk of thrashing [10], a phenomenon that causes large increase in computation time when the memory used is close to the total available memory in the hardware.

After preprocessing, the reduced representation $\mathbf{z}_i$ is used instead of the original data $\mathbf{x}_i$ yielding a time complexity of $\tilde{O}(\mathrm{T}_{\mathrm{preprocessing}} + \mathrm{n}_{\mathrm{iter}} mpnr)$. Let us highlight that an experiment is often run multiple times such as when cross validated results are needed. In these cases, the pre-processing is performed only once and the apparent complexity becomes $\tilde{O}(\mathrm{n}_{\mathrm{iter}} mpnr)$ which is faster than SRM by a factor of $\frac{v}{r}$. The number of regions in large atlases is about $r = 1000$ and in full brain data, the number of voxels is about 300 000 so that $\frac{v}{r}$ is typically about 1000. It remains to show how to draw a correspondence between FastSRM and SRM which is addressed in the following section.

3.2 An optimal atlas

In principle, FastSRM can be used with any atlas. However, in general, working with reduced data induces a loss of information that can be minimized if the atlas is carefully chosen. In any case, there is little hope to recover the parameters that would have been obtained from SRM from the parameters of FastSRM. Yet, we show that there exists an optimal atlas in the sense that SRM and FastSRM yield the same results.

Let us consider $\mathbf{x}_i = U_{\mathbf{x}_i} \mathbf{z}_i$ a PCA of $\mathbf{x}_i$ with the maximum number of components. As the number of samples n is lower than the number of features, $U_{\mathbf{x}_i} \in \mathbb{R}^{v \times n}$ and $\mathbf{z}_i \in \mathbb{R}^n$. We also have $U_{\mathbf{x}_i}^\top U_{\mathbf{x}_i} = I$. Therefore $U_{\mathbf{x}_i}$ constitutes a possible choice of subject specific atlas. As the next property shows, $U_{\mathbf{x}_i}$ is an optimal atlas for deterministic FastSRM.

Proposition 1 (Optimal atlas for deterministic FastSRM)**.** *Let $(A_i)_i, S$ be the solution obtained by deterministic SRM on data $(X_i)_i$ and $(A'_i)_i, S'$ the solution obtained by deterministic FastSRM on data $(X_i)_i$ using atlases $(U_{X_i})_i$ where $X_i = U_{\mathbf{x}_i} Z_i$ is a PCA of X_i. Then $A_i = U_{X_i} A'_i$ and $S = S'$.*

Proof. Updates of the mixing matrices A_i in deterministic SRM equation (5) can be written:

$$A_i \leftarrow \mathcal{P}(\frac{1}{n} X_i S^\top) = U_{X_i} \mathcal{P}(\frac{1}{n} Z_i S^\top) \tag{20}$$

where $\mathcal{P}$ is the projection on the Stiefel manifold: $\mathcal{P}(M) = M(M^\top M)^{-\frac{1}{2}}$.

Therefore we can look for A_i as $A_i = U_{X_i} \tilde{A}_i$. We can show that $\tilde{A}_i$ is orthogonal. Indeed,

$$A_i^\top A_i = I_p \tag{21}$$

$$\implies \tilde{A}_i^\top U_{X_i}^\top U_{X_i} \tilde{A}_i = I_p \tag{22}$$

$$\implies \tilde{A}_i^\top \tilde{A}_i = I_p \tag{23}$$

Then, we use the fact that

$$\|X_i - A_i S\|^2 = \|U_{X_i} Z_i - U_{X_i} \tilde{A}_i S\|^2 = \|Z_i - \tilde{A}_i S\|^2 \tag{24}$$

so that $A'_i = \tilde{A}_i$.

Therefore, the solution of deterministic SRM on data $(\mathbf{z}_i)_{i=1}^m$ and $(\mathbf{x}_i)_{i=1}^m$ are linked by the change of parameters $A_i = U_{\mathbf{x}_i} A'_i$ and $S = S'$. This concludes the proof.

□

In the case of probabilistic SRM we can obtain very similar results. However the algorithm applied on reduced data need to be slightly modified.

We call probSRM(ψ) the probabilistic SRM algorithm modified such that updates

$$\sigma_i^2 \leftarrow \frac{1}{v}(\mathbb{E}[\|\mathbf{x}_i - A_i \mathbb{E}[\mathbf{s}|\mathbf{x}]\|^2] + \|\mathrm{diag}(\mathbb{V}[\mathbf{s}|\mathbf{x}])\|^2) \tag{25}$$

are replaced by updates

$$\sigma_i^2 \leftarrow \frac{1}{\psi}(\mathbb{E}[\|\mathbf{x}_i - A_i \mathbb{E}[\mathbf{s}|\mathbf{x}]\|^2] + \|\mathrm{diag}(\mathbb{V}[\mathbf{s}|\mathbf{x}])\|^2) \tag{26}$$

We have the following result:

Proposition 2 (Optimal atlas for probabilistic FastSRM)**.** *Let $(A_i)_i, \sigma_i, \Sigma_s$ be the solution obtained by probabilistic SRM on data $\mathbf{x}_i$ and $(A'_i)_i, \sigma'_i, \Sigma'_s$ the solution obtained by ProbSRM(v) on data $\mathbf{z}_i = U_{\mathbf{x}_i}^\top \mathbf{x}_i$. Then $A_i = U_{\mathbf{x}_i} A'_i$, $\sigma_i = \sigma'_i$ and $\Sigma_s = \Sigma'_s$.*

Proof. Updates of the mixing matrices A_i in probabilistic SRM equation (15) can be written:

$$A_i \leftarrow U_{\mathbf{x}_i} \mathcal{P}(\mathbb{E}[\mathbf{z}_i \mathbb{E}[\mathbf{s}|\mathbf{x}_i]^T]) \tag{27}$$

so we can look for A_i as $A_i = U_{\mathbf{x}_i} \tilde{A}_i$ and, as in the deterministic case, $\tilde{A}_i$ is orthogonal. Therefore equality (24) holds.

Then we consider the expected negative log-likelihood of probabilistic srm:

$$\begin{aligned}
\mathcal{L} &= \sum_i \frac{1}{2} v \log(\sigma_i^2) + \frac{1}{2}\log(|\Sigma_s|) + \mathbb{E}[\int_{\mathbf{s}} \sum_i \frac{1}{2\sigma_i^2}\|\mathbf{x}_i - A_i \mathbf{s}\|^2 \\
&\quad + \frac{1}{2}\langle \mathbf{s}, \Sigma_s^{-1}\mathbf{s}\rangle d\mathbf{s}] && (28) \\
&= \sum_i \frac{1}{2} v \log(\sigma_i^2) + \frac{1}{2}\log(|\Sigma_s|) + \mathbb{E}[\int_{\mathbf{s}} \sum_i \frac{1}{2\sigma_i^2}\|\mathbf{z}_i - \tilde{A}_i \mathbf{s}\|^2 \\
&\quad + \frac{1}{2}\langle \mathbf{s}, \Sigma_s^{-1}\mathbf{s}\rangle d\mathbf{s}] && (29)
\end{aligned}$$

where we use equality (24).

Optimizing the log-likelihood via expectation maximization yields the exact same updates as probabilistic SRM on data $\mathbf{z}_i$ except that updates

$$\sigma_i^2 \leftarrow \frac{1}{t}(\mathbb{E}[\|\mathbf{z}_i - \tilde{A}_i \mathbb{E}[\mathbf{s}|\mathbf{z}]\|^2] + \|\mathrm{diag}(\mathbb{V}[\mathbf{s}|\mathbf{z}])\|^2) \tag{30}$$

are replaced by updates

$$\sigma_i^2 \leftarrow \frac{1}{v}(\mathbb{E}[\|\mathbf{z}_i - \tilde{A}_i \mathbb{E}[\mathbf{s}|\mathbf{z}]\|^2] + \|\mathrm{diag}(\mathbb{V}[\mathbf{s}|\mathbf{z}])\|^2) \tag{31}$$

so that $\tilde{A}_i = A_i'$.

Therefore, the updates in both algorithms are linked by $A_i = U_{\mathbf{x}_i} A_i'$, $\sigma_i' = \sigma_i$ and $\Sigma_s' = \Sigma_s$.

This concludes the proof.

□

Proposition 1 and Proposition 2 show that no information is lost by replacing $\mathbf{x}_i \in \mathbb{R}^v$ by its reduced representation $\mathbf{z}_i \in \mathbb{R}^n$. A key observation is that Proposition 1 and Proposition 2 hold whether or not the model for deterministic (respectively probabilistic) SRM is indeed the generative model of the data.

A complexity analysis shows that finding the optimal atlas becomes the limiting step of the pipeline. Even with fast implementations, the subject specific PCA is costly. However FastSRM only works on $\mathbf{z}_i$ so we do not need to know the value of $U_{\mathbf{x}_i}$. In practice, we observe data $X_i \in \mathbb{R}^{v \times n}$ and we want to get $Z_i \in \mathbb{R}^{n \times n}$ such that $X_i = U_{\mathbf{x}_i} Z_i$. This can be done by performing an SVD of $X_i^\top X_i$ yielding $X_i^\top X_i = V_i D_i V_i^\top$ and setting $Z_i = D_i^{\frac{1}{2}} V_i^\top$. Although computing the product $X_i^\top X_i$ has time complexity $\tilde{O}(vt^2)$ (the same as a PCA) the constant in the $\tilde{O}$ is exactly one so this operation costs a lot less than the PCA on full data. When estimates of the mixing matrices are needed, they can be obtained by applying equation (20) in the deterministic SRM case and equation (27) in the probabilistic SRM case which only costs $\tilde{O}(mvp^2)$. In practice the cost of the matrix products $X_i^\top X_i$ is often still the limiting step of the pipeline (this depends on the number of iterations) but as we show in the next chapter, it is much more efficient than performing SRM on the full data. Note than if memory allows it, these matrix products can be computed in parallel.

4 Related work

The implementation of SRM proposed in [8] is quadratic in the number of features which prevent its application to full brain data. In [2], the authors address this issue by using the inversion lemma to remove the quadratic dependence. Their implementation is now the most widely used one. In our work, we propose to reduce further the computation time and memory usage compared to the implementation in [2] by the use of optimal atlases.

Other popular methods used to speed up SRM are searchlight [26] or piecewise [3] approaches. While these methods are efficient, they do not optimize the same objective as the original SRM algorithm and are arguably less principled since the searchlights or pieces are chosen a priori.

We see SRM as a well principled formulation of the hyperalignment method [13]. Many methods exist to perform hyperalignment: deep hyperalignment [25] , robust SRM [23] , multi-view canonical correlation analysis [16], ShICA [20], MultiViewICA [21] optimal transport [4] and many more. In this work, we do not claim that FastSRM is the best method. We only see that SRM is widely used and aim to provide a faster, more memory efficient and identifiable implementation. In our view FastSRM should be seen as a plugin replacement for SRM.

5 Experiments

We make several experiments on both synthetic and real data. We used nilearn [1] for fMRI data preprocessing, Brainiak [15] for the non-identifiable version of SRM that implements the work of [8] and [2], Numpy [12] for array processing, Scipy [24] its implementation of the Hungarian algorithm, Matplotlib [14] for plotting and Sklearn [19] for machine learning pipelines.

5.1 Comparing Fitting time and performance of FastSRM and SRM on synthetic data

We generate synthetic data $\mathbf{x}_i$ according to the model of probabilistic SRM. The parameters σ_i, A_i and Σ_s are generated randomly. We sample the value of the subject specific noise level from a normal distribution: $\sigma_i \sim \mathcal{N}(0, 0.1)$. The mixing matrices A_i are obtained by sampling their coefficient from a standardized normal distribution. Lastly, the covariance of the shared response Σ_s is diagonal and the diagonal values are obtained by sampling from a Dirichlet distribution with parameter $(1 \dots 1)$. We set the number of voxels to $v = 125\ 000$, the number of subjects to $m = 10$ and the number of components to $p = 50$. We generate $n = 1000$ samples.

We benchmark deterministic SRM, probabilistic SRM and their FastSRM counterparts in terms of fitting time and performance. Note that in this section, the identifiable implementation of deterministic SRM and probabilistic SRM described in section 2.2 and 2.1 is used. Algorithms are designated by the atlas they use and therefore SRM algorithms are refered to as *None* because no atlas is used and FastSRM algorithms will have the label *Optimal*. Note that it would be possible to use FastSRM with sub-optimal atlases (there exists a wide variety of atlases available [22, 5, 17]) but without any guarantees that the performance are the same as SRM.

We use a number of iterations between 1 and 100 and report the performance, fitting time and a measure of convergence. In FastSRM, we do not compute the unmixing matrices but only the shared response. We measure the performance of an algorithm by computing the error between the true component $S \in \mathbb{R}^{p \times n}$ and the predicted component $\hat{S} \in \mathbb{R}^{p \times n}$ using the quantity:

$$\mathrm{mse}(\hat{S}, S) = min_{A \in \mathbb{R}^{p \times p}} \|A\hat{S} - S\|_F^2 = \|S\hat{S}^{\dagger}\hat{S} - S\|_F^2 \tag{32}$$

This way of measuring errors is insensitive to the indeterminacies in DetSRM. We measure the fitting time in seconds. Lastly, we measure convergence by computing the gradient ℓ_∞ norm in case of DetSRM given by $\max(\mathrm{abs}(G))$

where G is the gradient and use the distance between consecutive values of the loss for ProbSRM. Results are plotted in Figure 3.

We empirically see that the optimal approach is equivalent to using no atlas in terms of performance. This is predicted by the theory in section 3.2 where we demonstrate that these two algorithms yield exactly the same output from the same input. In general probabilistic methods give much better results than their deterministic counterpart. This shows the superiority of likelihood based methods. In terms of fitting time, FastSRM is about a thousand time faster than SRM after 100 iterations. When no atlas is used, the number of iterations has a very strong impact on performance while it has a small impact when the optimal atlas is used. Lastly, looking at the convergence curves, we see that even after 100 iterations, algorithms did not fully converge. This means that in practice a much larger number of iterations is needed which would yield an even higher difference in fitting time between methods using no atlas and methods using the optimal atlas.

5.2 Experiment on fMRI data: identifiability increases stability

We evaluate the performance of the different SRM implementations on the *Sherlock* datasets where fMRI of 17 participants watching "Sherlock" BBC TV show (episode 1) is performed. These data are downloaded from `http://arks.princeton.edu/ark:/88435/dsp01nz8062179`. Data were acquired using a 3T scanner with an isotropic spatial resolution of 3 mm. More information including the preprocessing pipeline is available in [6]. Subject 5 is removed because of missing data, leaving us with 16 participants. Although SHERLOCK data contains originally only 1 run, we split it into 4 blocks of 395 timeframes and one block of 396 timeframes for the needs of our experiments.

We first show that identifiability is a desirable property as it increases stability. Then we show that our FastSRM algorithm (that works with optimal atlases) matches the performance of SRM (that works on full data) but use much less computational resources.

5.3 Identifiability increases stability

Assuming that the data follow the model, identifiability ensures that the correct parameters can be recovered without ambiguity. In real fMRI data, the model cannot be expected to hold perfectly, but we can hope for greater stability in the parameters recovered than if an unidentifiable model is used.

To measure the stability of the common components obtained from the Sherlock dataset, we divide the subjects into two roughly equal groups and extract the common components in each group. The components are then matched using the Hungarian algorithm and the stability measure is obtained from the average correlation of the matched components. The procedure is repeated 9 times using the Brainiak implementation (that is not identifiable) and our FastSRM implementation. We plot the results as a scatter plot in Figure 4,

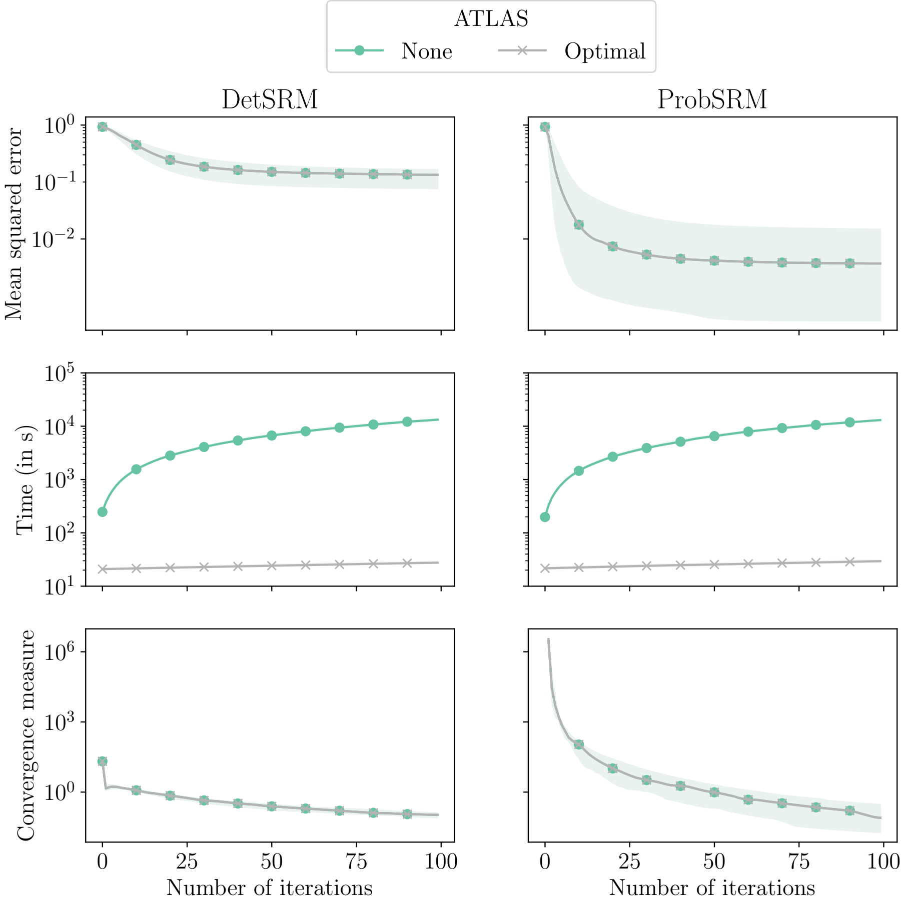

Figure 3: **Benchmark of SRM algorithms on synthetic data:** Performance, fitting time and convergence of SRM algorithms in the deterministic (left) or probabilistic (right) case. As expected, when optimal atlases are used, the performance is the same as if no atlas is used but the fitting time is much lower. This is even more pronounced when the number of iterations is high (and looking at convergence curves, we see that more iterations could be performed to be closer to a stationary point).

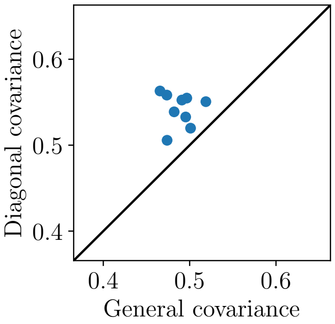

Figure 4: **Identifiability increases stability:** We first divide the subjects into two groups and extract the common components in each group. The components of the two groups are then matched using the Hungarian algorithm and the stability index is determined by the average correlation of the matched components. The procedure is repeated 9 times with the Brainiak implementation (not identifiable since the shared components covariance is unconstrained) and our FastSRM implementation (identifiable since the shared components covariance is diagonal).

where, for each repetition , the x-axis indicates the stability measure obtained with Brainiak's implementation (not identifiable since the shared components covariance is unconstrained) and the y-axis the stability measure obtained with FastSRM (identifiable since the shared components covariance is diagonal). We see that introducing a diagonal source covariance improves stability.

5.4 Comparing fitting time, memory usage and performance on a timesegment matching experiment

The timesegment matching experiment is first introduced in [8]. In a nutshell, the time-segment matching accuracy measures the similarity between two multivariate time-series by trying to localize a time-segment in one time-series by correlation with the other. In the context of movie watching, this measure has a lot of sense: if we split the movies in scenes and compute a representation per scene and per subject, it makes sense to assume that different subjects watching the movie would still have closer representation of the same scenes than of different scenes. This explains why timesegment matching is a standard evaluation of SRM-like methods also used in [11], [18] or [26].

We now describe more precisely the experimental design. We split the runs into a train and test set. After fitting the model on the training set, we apply the unmixing matrices $W_i = A_i^{-1}$ of each subject on the test set yielding individual components matrices. We estimate the shared responses by averaging the individual components of each subjects but one. We select a target time-segment (9 consecutive timeframes) in the shared responses and try to localize the corresponding time segment in the components of the left-out subject using a maximum-correlation classifier.

The time-segment is said to be correctly classified if the correlation between the sample and target time-segment is higher than with any other time-segment (partially overlapping time windows are excluded).

We use 5-Fold cross-validation across runs: the training set contains 80% of the runs and the test set 20%, and repeat the experiment using all possible

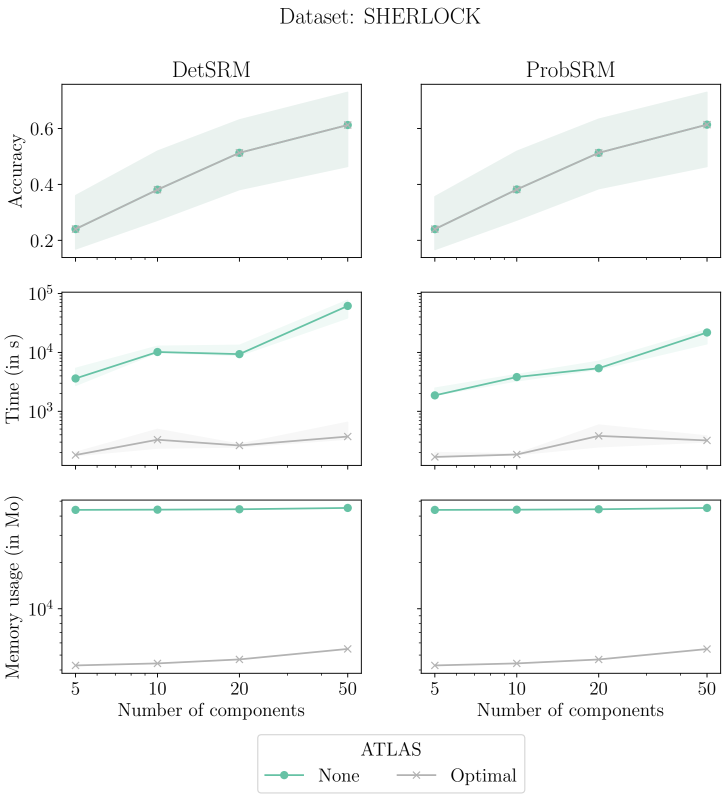

Figure 5: **Benchmark of SRM algorithms on fMRI data** (top) Timesegment matching accuracy (middle) Fitting time (bottom) Memory usage. When the optimal atlas is used, the accuracy is the same as when no atlas is used but the fitting time is reduced by a factor 10 to 100 and so is the memory usage.

choices for left-out subjects. The mean accuracy is reported in Figure 5 (bottom). When the optimal atlas is used, the accuracy is the same as when no atlas is used but the fitting time is reduced by a factor 10 to 100 and so is the memory usage.

6 Conclusion

As studies using naturalistic stimuli will tend to become more common and large within and across subjects, we need scalable models in terms of computation time and memory usage. This is what FastSRM provides.

FastSRM is an efficient implementation of SRM that uses optimal atlases to speed up computations and reduce memory requirements with provably no loss of performance.

We show on synthetic and real data that FastSRM is much faster and more memory efficient that implementations not using the optimal atlas. Furthermore, FastSRM is identifiable. On real data, we show that compared to Brainiak's implementation (which is not identifiable), FastSRM provides more stable estimates of the shared components.

FastSRM inherits from all the weaknesses of SRM including the fact that mixing matrices are assumed to be orthogonal which is rather unrealistic. It remains to be seen whether dimension reduction and learning of the shared components could be done jointly and efficiently without assuming orthogonal mixing matrices. Our optimal atlas provide the intuition that SRM-like methods do not need the full data to provide accurate estimates of shared components. We believe such insights may guide the design of future methods.

References

[1] Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Mueller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gaël Varoquaux. Machine learning for neuroimaging with scikit-learn. *Frontiers in neuroinformatics*, 8:14, 2014.

[2] Michael J Anderson, Mihai Capota, Javier S Turek, Xia Zhu, Theodore L Willke, Yida Wang, Po-Hsuan Chen, Jeremy R Manning, Peter J Ramadge, and Kenneth A Norman. Enabling factor analysis on thousand-subject neuroimaging datasets. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1151–1160. IEEE, 2016.

[3] Thomas Bazeille, Elizabeth Dupre, Hugo Richard, Jean-Baptiste Poline, and Bertrand Thirion. An empirical evaluation of functional alignment using inter-subject decoding. *NeuroImage*, page 118683, 2021.

[4] Thomas Bazeille, Hugo Richard, Hicham Janati, and Bertrand Thirion. Local optimal transport for functional brain template estimation. In *International Conference on Information Processing in Medical Imaging*, pages 237–248. Springer, 2019.

[5] Pierre Bellec, Pedro Rosa-Neto, Oliver C Lyttelton, Habib Benali, and Alan C Evans. Multi-level bootstrap analysis of stable clusters in resting-state fmri. *Neuroimage*, 51(3):1126–1139, 2010.

[6] J. Chen, Y.C. Leong, K.A. Norman, and U. Hasson. Shared experience, shared memory: a common structure for brain activity during naturalistic recall. *bioRxiv*, 2016.

[7] Janice Chen, Yuan C Leong, Christopher J Honey, Chung H Yong, Kenneth A Norman, and Uri Hasson. Shared memories reveal shared structure in neural activity across individuals. *Nature neuroscience*, 20(1):115–125, 2017.

[8] Po-Hsuan Chen, Janice Chen, Yaara Yeshurun, Uri Hasson, James Haxby, and Peter J Ramadge. A reduced-dimension fMRI shared response model. In *Advances in Neural Information Processing Systems*, pages 460–468, 2015.

[9] Kamalaker Dadi, Gaël Varoquaux, Antonia Machlouzarides-Shalit, Krzysztof J Gorgolewski, Demian Wassermann, Bertrand Thirion, and Arthur Mensch. Fine-grain atlases of functional modes for fmri analysis. *NeuroImage*, 221:117126, 2020.

[10] Peter J Denning. Thrashing: Its causes and prevention. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, pages 915–922, 1968.

[11] J Swaroop Guntupalli, Ma Feilong, and James V Haxby. A computational model of shared fine-scale structure in the human connectome. *PLoS computational biology*, 14(4):e1006120, 2018.

[12] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[13] James V Haxby, J Swaroop Guntupalli, Andrew C Connolly, Yaroslav O Halchenko, Bryan R Conroy, M Ida Gobbini, Michael Hanke, and Peter J Ramadge. A common, high-dimensional model of the representational space in human ventral temporal cortex. *Neuron*, 72(2):404–416, 2011.

[14] John D Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95, 2007.

[15] Manoj Kumar, Michael J Anderson, James W Antony, Christopher Baldassano, Paula P Brooks, Ming Bo Cai, Po-Hsuan Cameron Chen, Cameron T Ellis, Gregory Henselman-Petrusek, David Huberdeau, et al. Brainiak: The brain imaging analysis kit. *Aperture*, 2020.

[16] Yi-Ou Li, Tülay Adali, Wei Wang, and Vince D Calhoun. Joint blind source separation by multiset canonical correlation analysis. *IEEE Transactions on Signal Processing*, 57(10):3918–3929, 2009.

[17] Arthur Mensch, Julien Mairal, Bertrand Thirion, and Gaël Varoquaux. Extracting universal representations of cognition across brain-imaging studies. *arXiv preprint arXiv:1809.06035*, 2018.

[18] Samuel A Nastase, Yun-Fei Liu, Hanna Hillman, Kenneth A. Norman, and Uri Hasson. Leveraging shared connectivity to aggregate heterogeneous datasets into a common response space. *bioRxiv*, 2019.

[19] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[20] H. Richard, P. Ablin, B. Thirion, A. Gramfort, and A. Hyvarinen. Shared independent component analysis for multi-subject neuroimaging. In *Advances in Neural Information Processing Systems 33*, December 2021.

[21] H. Richard, L. Gresele, A. Hyvarinen, B. Thirion, A. Gramfort, and P. Ablin. Modeling shared responses in neuroimaging studies through multiview ICA. In *Advances in Neural Information Processing Systems 33*, December 2020.

[22] Alexander Schaefer, Ru Kong, Evan M Gordon, Timothy O Laumann, Xi-Nian Zuo, Avram J Holmes, Simon B Eickhoff, and BT Thomas Yeo. Local-global parcellation of the human cerebral cortex from intrinsic functional connectivity mri. *Cerebral Cortex*, 28(9):3095–3114, 2017.

[23] Javier S Turek, Cameron T Ellis, Lena J Skalaban, Nicholas B Turk-Browne, and Theodore L Willke. Capturing shared and individual information in fmri data. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 826–830. IEEE, 2018.

[24] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[25] Muhammad Yousefnezhad and Daoqiang Zhang. Deep hyperalignment. *arXiv preprint arXiv:1710.03923*, 2017.

[26] Hejia Zhang, Po-Hsuan Chen, Janice Chen, Xia Zhu, Javier S Turek, Theodore L Willke, Uri Hasson, and Peter J Ramadge. A searchlight factor model approach for locating shared information in multi-subject fMRI analysis. *arXiv preprint arXiv:1609.09432*, 2016.

[27] Hejia Zhang, Po-Hsuan Chen, and Peter Ramadge. Transfer learning on fMRI datasets. In *International Conference on Artificial Intelligence and Statistics*, pages 595–603, 2018.