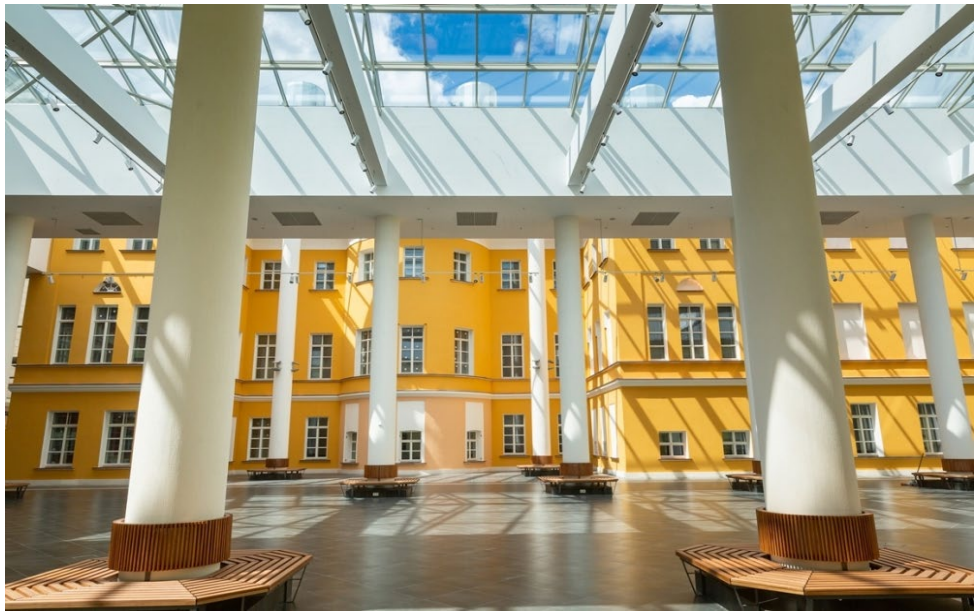
NATIONAL RESEARCH
UNIVERSITY HIGHER
SCHOOL OF ECONOMICS



MOSCOW CAMPUS

CENTER FOR BIOELECTRIC
INTERFACES

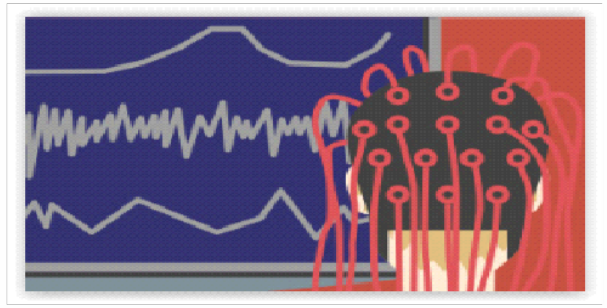
Продвинутые методы анализа сигналов в ИМК.

Alexei Ossadtchi

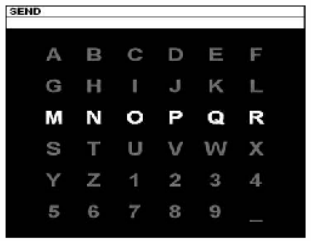
Сегодня

- Compact CNN for interpreatble BCI
- Weights interpretation
- Using covariance matrices as features
- Riemannian manifold
- Parallel transport for domain adaptation
- Практическая часть.

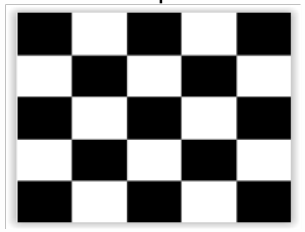
Введение: три типа нейроинтерфейсов



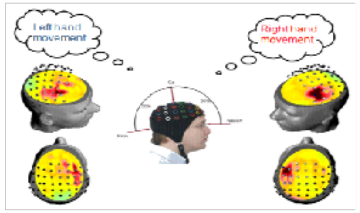
позиционные



на зрительных потенциалах



на воображаемых движениях



Декодирование реакции на внешние воздействия

Декодирование внутреннего состояния

Phenomenological model

Observed data

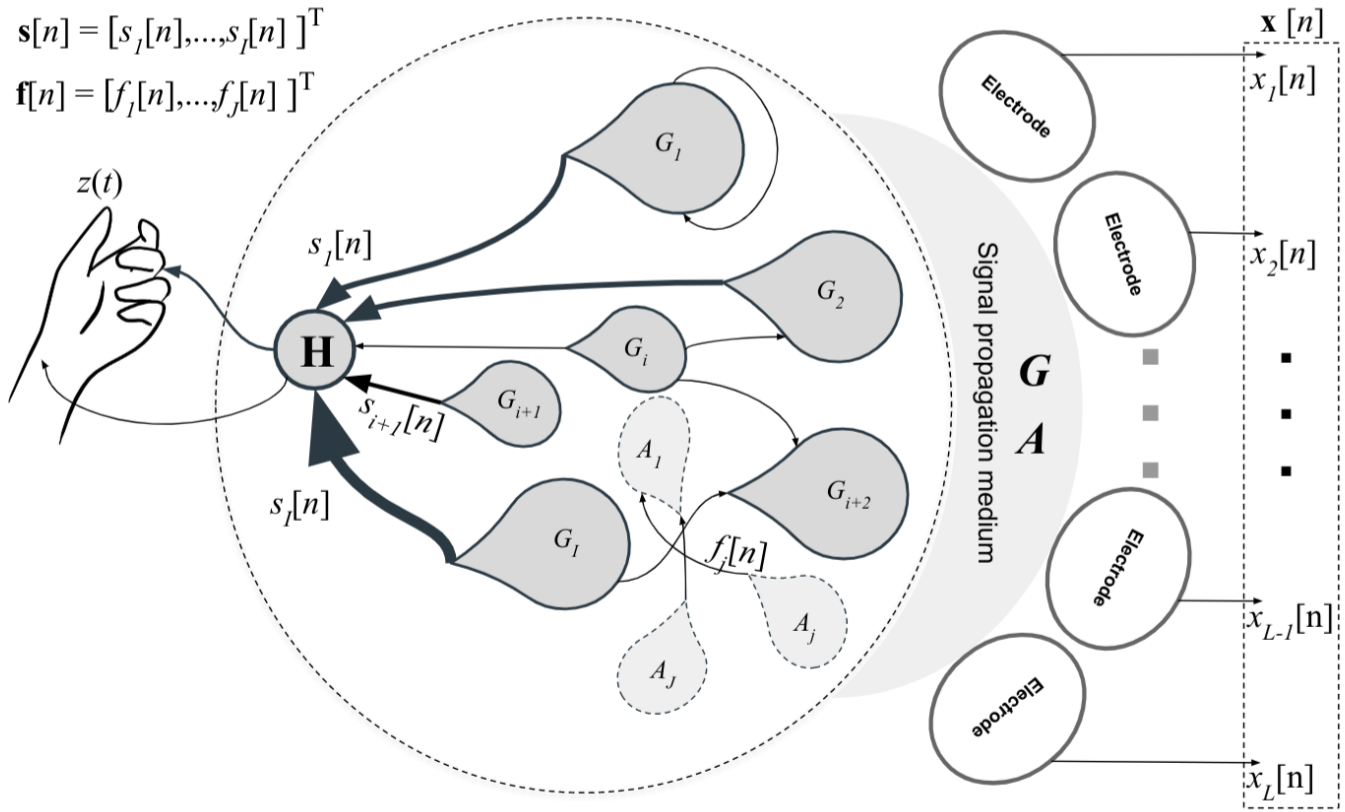
Target LFPs as measured by the sensors

Task unrelated LFP as measured by the sensors

$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{A}\mathbf{f}(t) = \sum_{i=1}^{I} \mathbf{g}_i s_i(t) + \sum_{j=1}^{J} \mathbf{a}_j f_j(t)$$

$$z(t) = \overset{?}{\mathcal{F}}(\mathbf{x}(t))$$

$$z(t) = \mathbf{H}(\mathbf{env}(t)) = \mathbf{H}(\mathbf{V}(\mathbf{s}(t)))$$

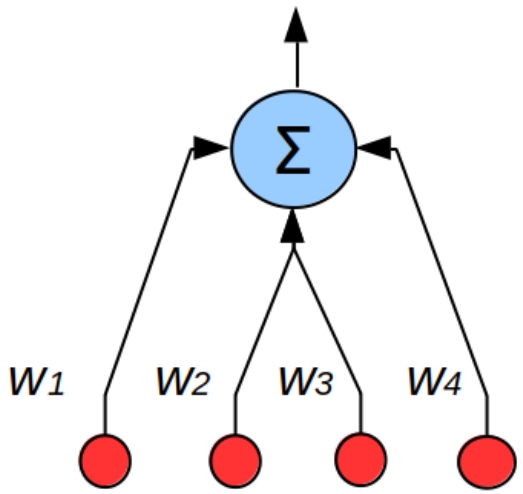


Classically...

- Spatial filtering (ICA, CSP, SSD)
- Temporal filtering
- Power estimation
- Classification (LDA, SVM, shNN)

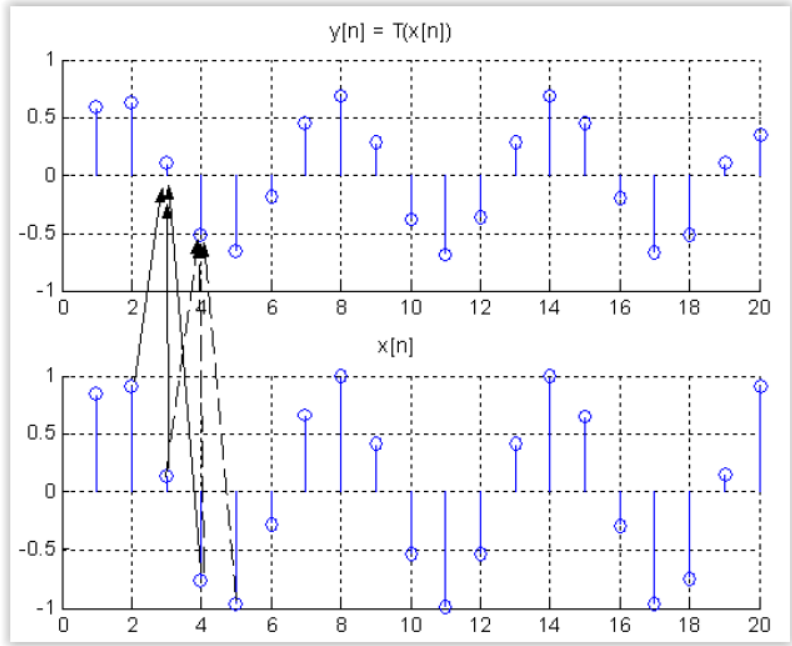
Spatial filtering

$$s(t) = \sum w_i x_i(t)$$



Space

Temporal filtering



Time

Interpretable building block

- We believe that the instantaneous band power (envelope) carries information about activity of specific neuronal populations
- Neuronal popultaiton activity is related to limb movement
- Let us make such a power detector to be a building block for more complex architectures

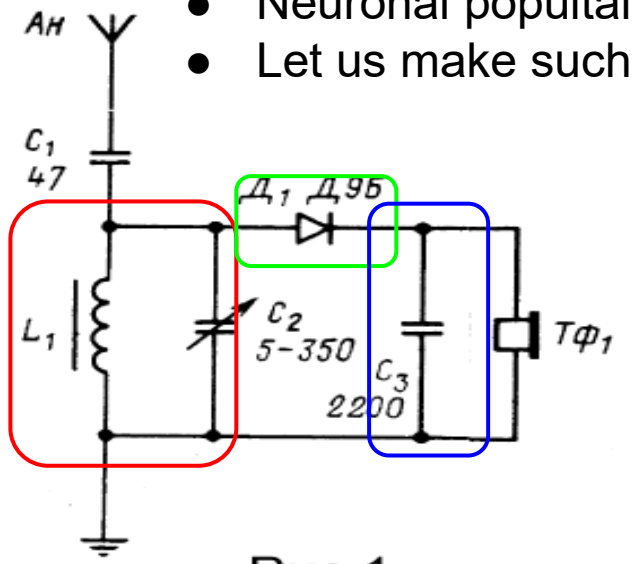
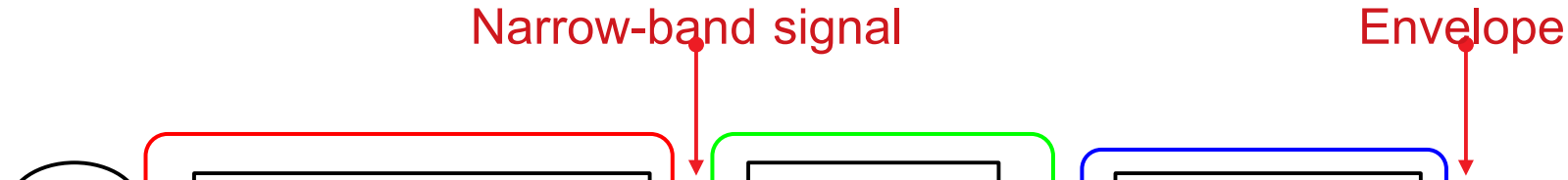
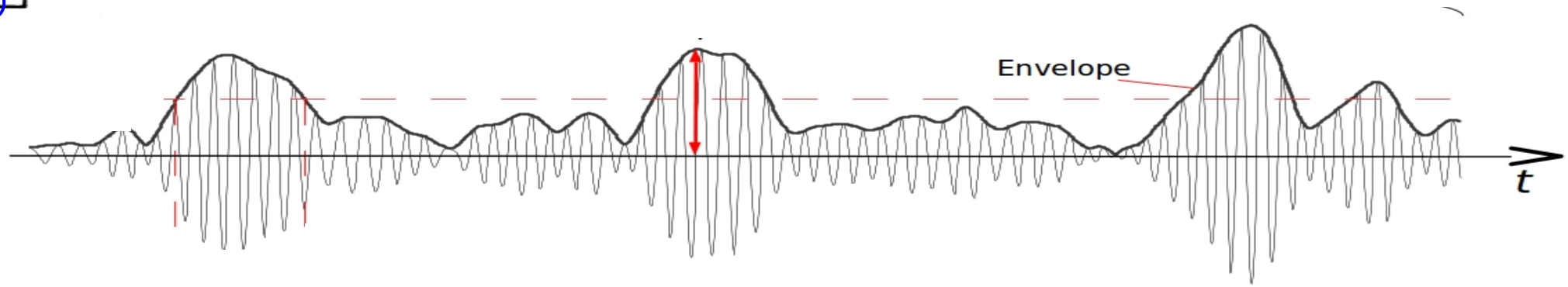


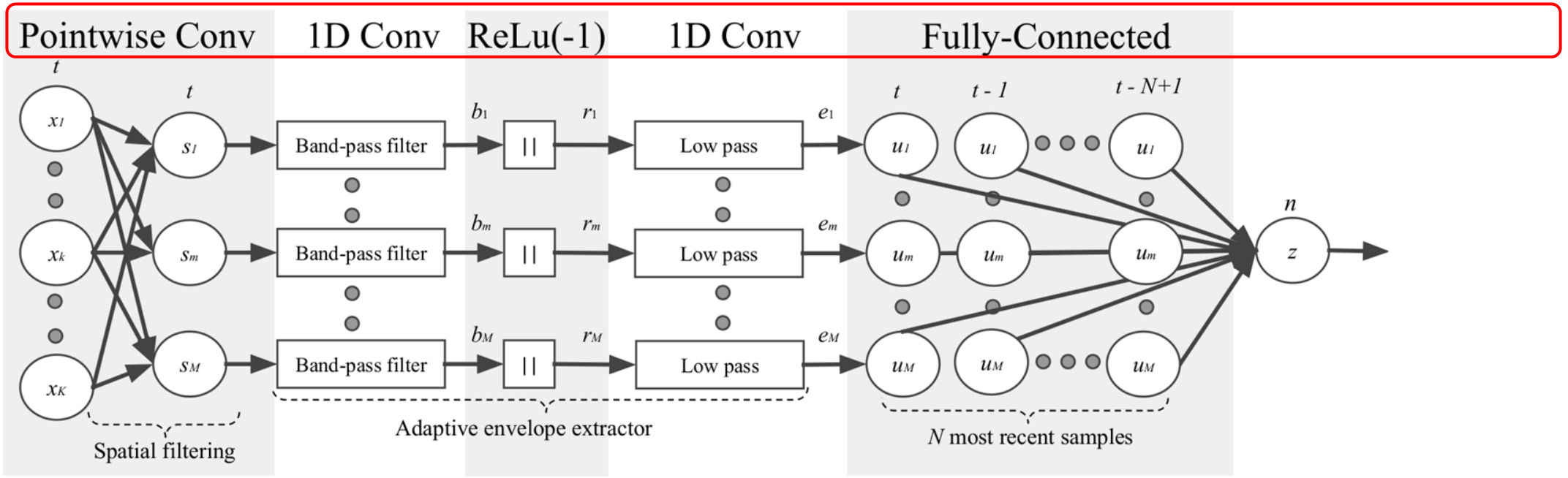
Рис.1

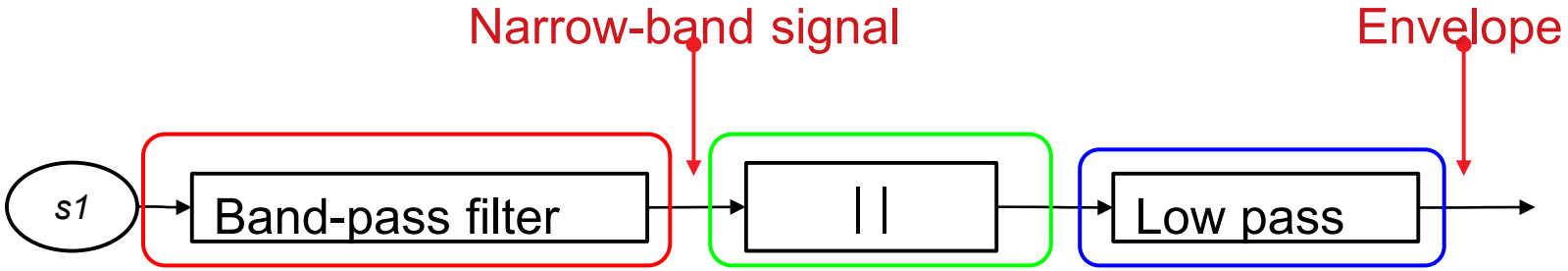




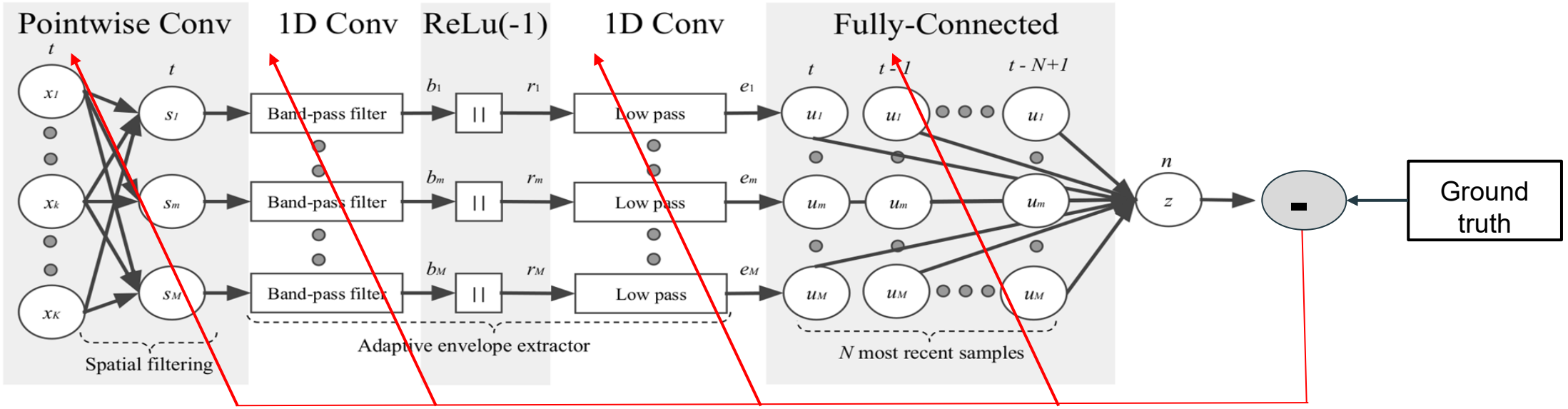
- Once we find a solution by training an architecure made of such blocks we will be able to intepret the weights in the context of the generative model linking band power (related to activity of neuronal populations) to limb kinematics

DL primitives are made to do just that!





And can be trained to adjust all the filter weights



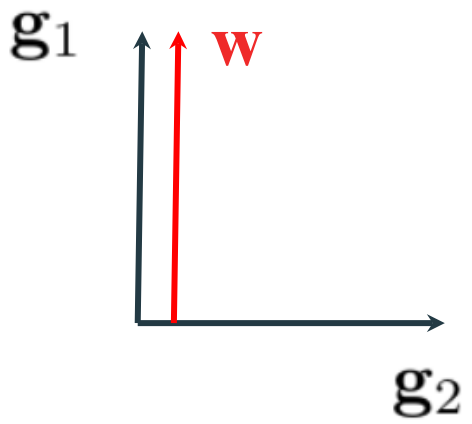
$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{e}(t) = \sum_{i=1}^{N} \mathbf{g}_i j_i(t) + \mathbf{e}(t)$$

$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) = \mathbf{g}_1 j_1(t) + \mathbf{g}_2 j_2(t)$$

$$\mathbf{g}_1 \perp \mathbf{g}_2, ||\mathbf{g}_1|| = ||\mathbf{g}_2|| = 1$$

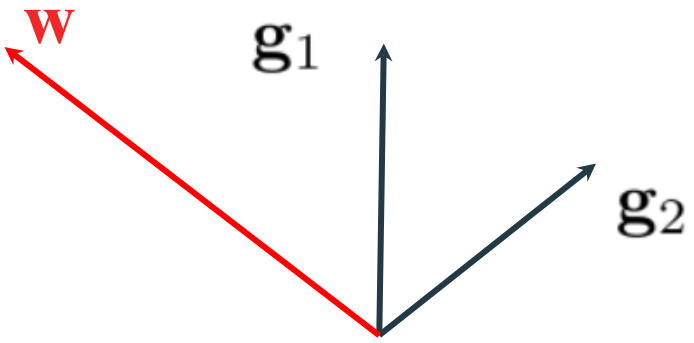
$$\mathbf{w}||\mathbf{g}_1, \mathbf{w}^T \mathbf{g}_1 = 1, \mathbf{w} \perp \mathbf{g}_2$$

$$y(t) = \mathbf{w}^T \mathbf{x}(t) = 1 j_1(t) + 0 j_2(t) = j_1(t)$$



$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{e}(t) = \sum_{i=1}^{N} \mathbf{g}_i j_i(t) + \mathbf{e}(t)$$

$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) = \mathbf{g}_1 j_1(t) + \mathbf{g}_2 j_2(t)$$



$$\mathbf{g}_1 \perp \mathbf{g}_2, ||\mathbf{g}_1|| = ||\mathbf{g}_2|| = 1$$

$$\mathbf{w}||\mathbf{g}_1, \mathbf{w}^T\mathbf{g}_1 = 1, \mathbf{w} \perp \mathbf{g}_2$$

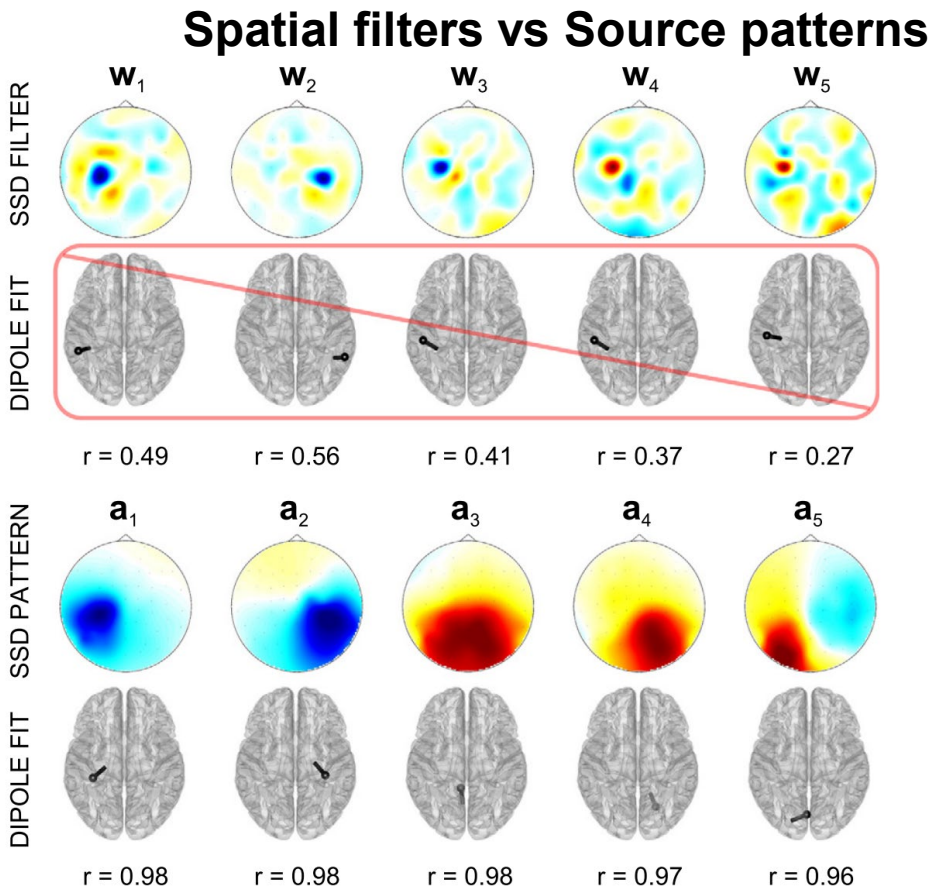
$$y(t) = \mathbf{w}^T\mathbf{x}(t) = 1 j_1(t) + 0 j_2(t) = j_1(t)$$

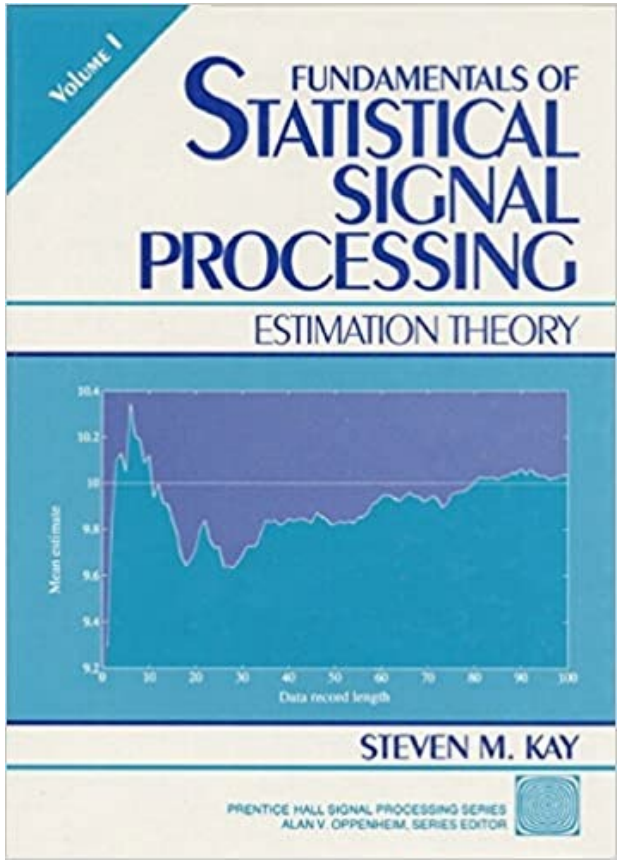
$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{g}_1 = \mathbf{R}^{-1/2}\mathbf{R}^{-1/2}\mathbf{g}_1$$

$$\mathbf{w}^T\mathbf{x}(t) = = \mathbf{g}_1^T \mathbf{R}^{-1/2} \mathbf{R}^{-1/2}\mathbf{x}(t)$$

S. Haufe, Neuroimage, 2014/ Best paper award!

Interpreting spatial weigts of a simple regression



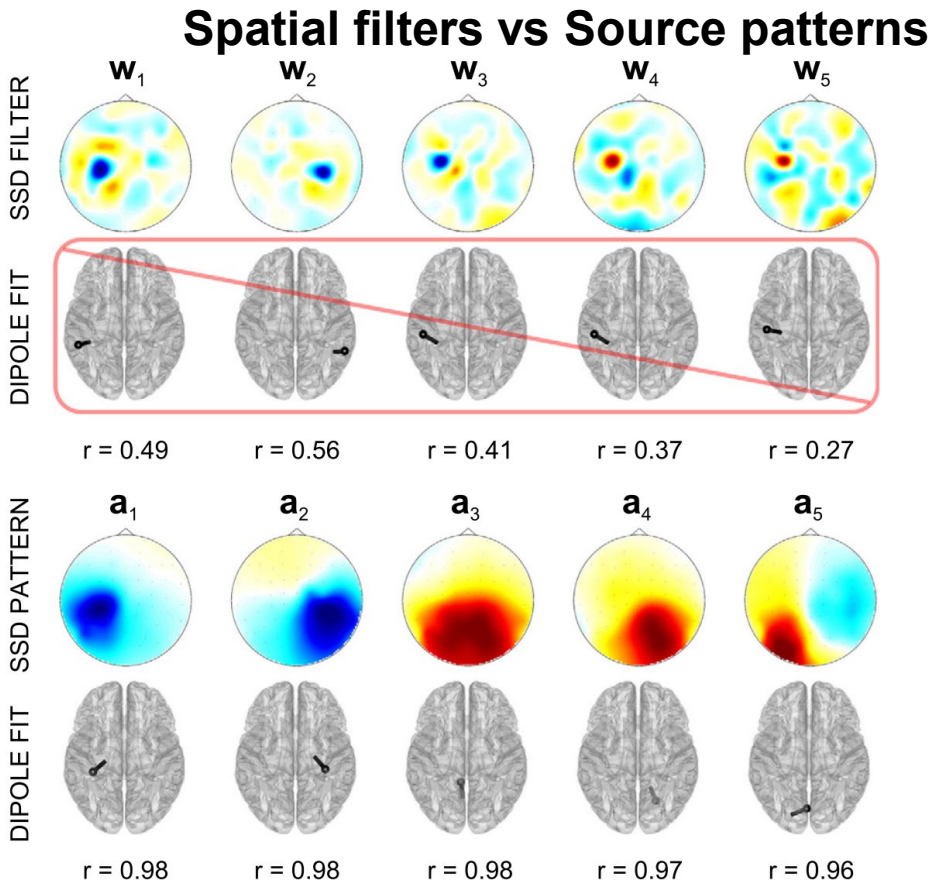


$$\boldsymbol{x}(t)=\boldsymbol{g}s(t)+\boldsymbol{n}(t) \qquad \hat{s}(t)=\boldsymbol{w}\,\boldsymbol{x}(t) \qquad \boldsymbol{w}\neq\boldsymbol{g}$$

Wiener optimal, MMSE -> min

S. Haufe, Neuroimage, 2014/ Best paper award!

Filter vs. Pattern



$$x_1(t) = s(t) + n(t)$$

$$x_2(t) = -n(t)$$

$$\boldsymbol{x}(t) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} s(t) + \begin{bmatrix} 1 \\ -1 \end{bmatrix} n(t)$$

$$\hat{s}(t) = x_1(t) + x_2(t) = \begin{bmatrix} 1 & , & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$\boldsymbol{x}(t) = \boldsymbol{g}s(t) + \boldsymbol{n}(t) \qquad \hat{s}(t) = \boldsymbol{w}\,x(t)$$

$$\boldsymbol{w} \neq \boldsymbol{g} \qquad \hat{\boldsymbol{g}} = \boldsymbol{C}_x \boldsymbol{w} \boldsymbol{C}_s^{-1} \approx \boldsymbol{C}_x \boldsymbol{w}$$

S. Haufe, Neuroimage, 2014/ Best paper award!

Filter vs. Pattern

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t)$$

$$\hat{\mathbf{s}}(t) = \mathbf{W}^T\mathbf{x}(\mathbf{t}) = \mathbf{W}^T(\mathbf{A}\mathbf{s}(t) + \mathbf{n}(t));$$

$$estimation\ error = \hat{\mathbf{s}}(t) - \mathbf{s}(t) = \mathbf{W}^T\mathbf{x}(t) - \mathbf{s}(t)$$

$$orthogonality\ principle: \quad E\{\left(\mathbf{W}^T\mathbf{x} - \mathbf{s}\right)\mathbf{x}^T\} = \mathbf{0}$$

$$\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\} = E\{\mathbf{s}\mathbf{x}^T\}$$

$$\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\} = E\{\mathbf{s}\left(\mathbf{s}^T\mathbf{A}^T + \mathbf{n}^T\right)\}$$

$$\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\} = E\{\mathbf{s}\mathbf{s}^T\}\mathbf{A}^T + E\{\mathbf{s}\mathbf{n}^T\}$$

$$\mathbf{W}^T E\{\mathbf{x}\mathbf{x}^T\} = E\{\mathbf{s}\mathbf{s}^T\}\mathbf{A}^T$$

$$\mathbf{W}^T\mathbf{\Sigma}_\mathbf{x} = \mathbf{\Sigma}_\mathbf{s}\mathbf{A}^T$$

$$\mathbf{\Sigma}_\mathbf{x}\mathbf{W} = \mathbf{A}\mathbf{\Sigma}_\mathbf{s}$$

$$\mathbf{\Sigma}_\mathbf{x}\mathbf{W}\mathbf{\Sigma}_\mathbf{s}^{-1} = \mathbf{A}$$

Filter vs. Pattern

$$x_1(t) = s(t) + n(t)$$

$$x_2(t) = -n(t)$$

$$\boldsymbol{x}(t) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} s(t) + \begin{bmatrix} 1 \\ -1 \end{bmatrix} n(t)$$

$$\hat{s}(t) = x_1(t) + x_2(t) = \begin{bmatrix} 1 & , & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

$$\boldsymbol{w} \neq \boldsymbol{g} \qquad \hat{\boldsymbol{g}} = \boldsymbol{C_x w C_s^{-1}} \approx \boldsymbol{C_x w}$$

```
T = 1000;
g = [1 0]';
a = [1 -1]';
s = randn(1,T); % var = 1
n = randn(1,T); % var = 1
X = g*s +a*n;
Cx = g*g' + a*a';
%Cx = g*g'*var(s) + a*a'*var(n);
w = [1 1]';
g_est = Cx*w
Cx_est = X*X'/T;
s_est = w'*X;
g_est_emp = Cx_est*w*(var(s_est))^(-1)
```

g_est =

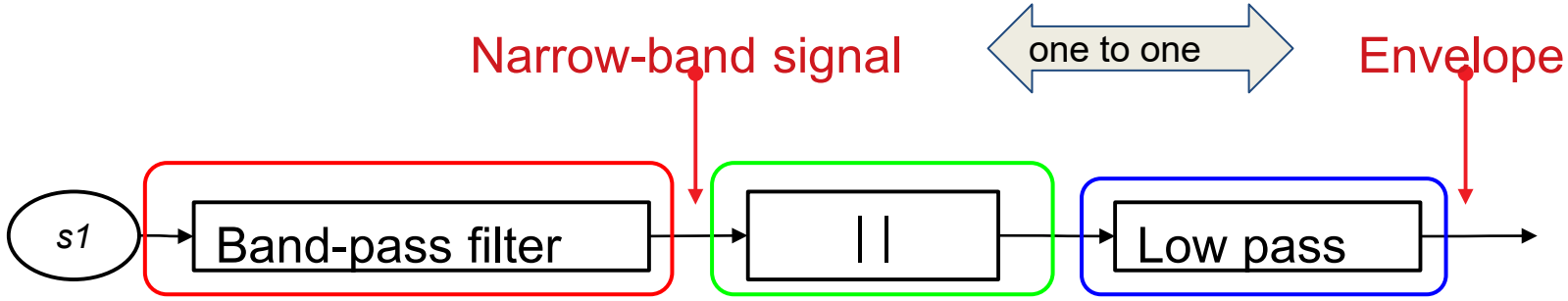
1
0

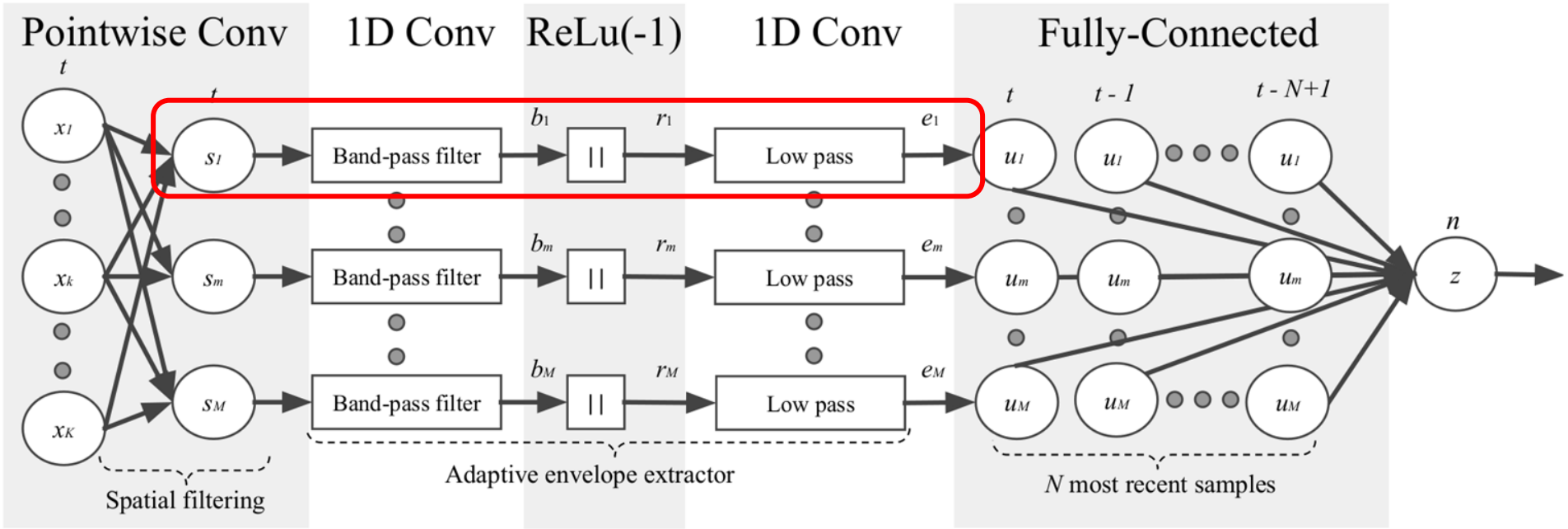
g_est_emp =

1.0012
-0.0003

S. Haufe, Neuroimage, 2014/ Best paper award!

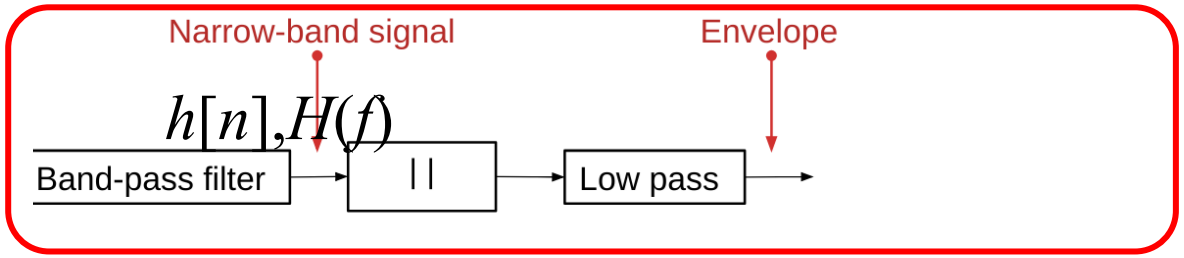
Interpreting temporal weights





Interpreting temporal weights of 1 channel

$x[k] = s[k] + n[k]$



Wiener filter

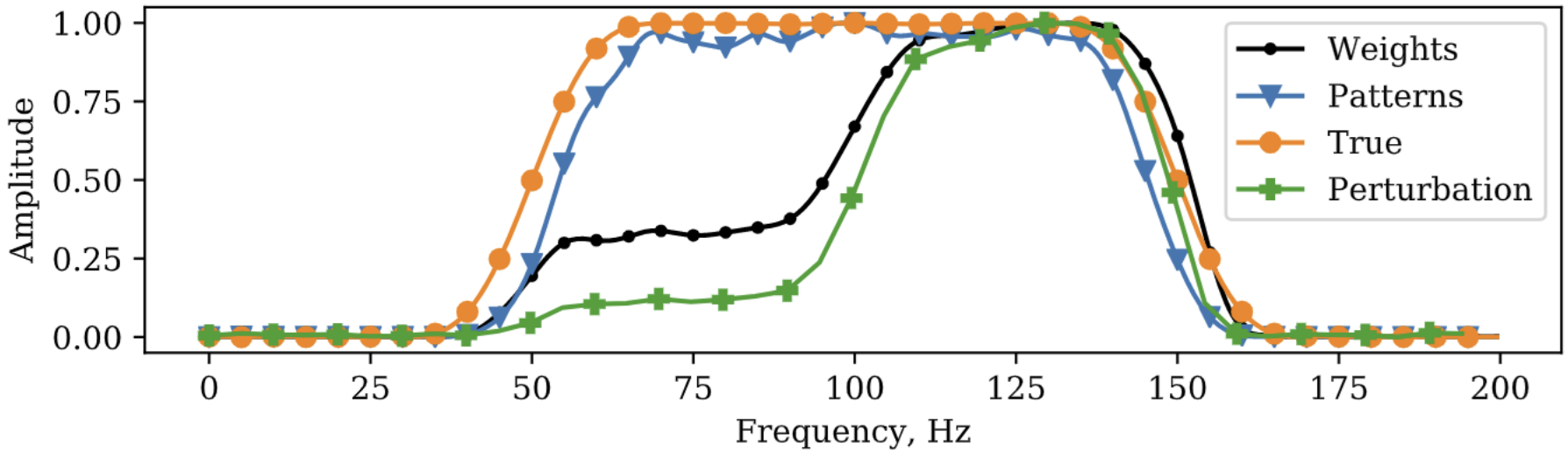
$$H(f) = \frac{P_{sx}(f)}{P_{xx}(f)}$$

$$s_i(t) \perp s_j(t) \;\wedge\; s(t) \perp n(t)$$

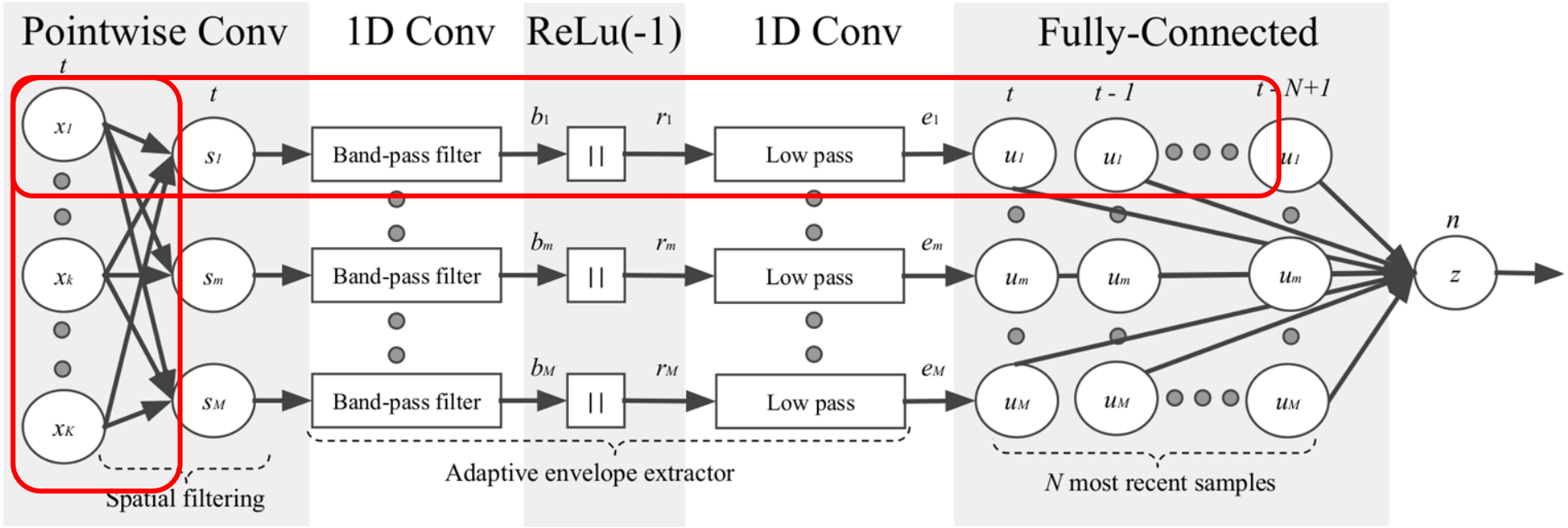
$h[k]$ - optimal weights

$H(f) =$ DTFT($h[k]$)

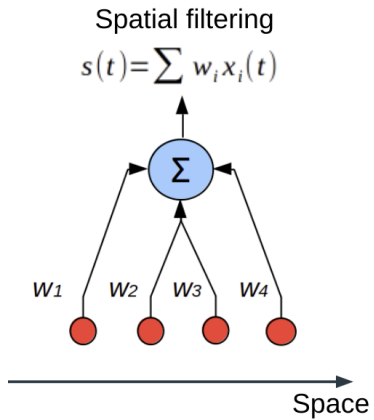
$$P_{ss}(f) = H(f) P_{xx}(f) \quad \Leftarrow \quad H(f) = \frac{P_{ss}(f)}{P_{ss}(f) + P_{nn}(f)} = \frac{P_{ss}(f)}{P_{xx}(f)}$$

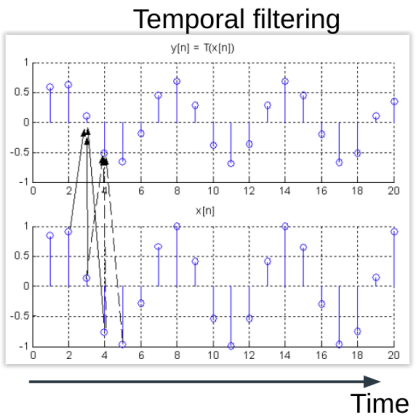


Consider a spatial-temporal chunk of data

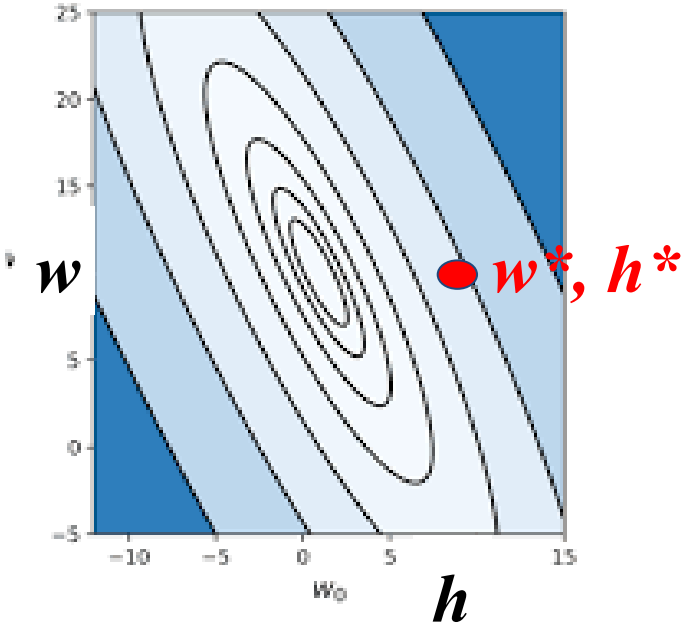


$$b_m(t) = \mathbf{w}_m^T \mathbf{X}(t) \mathbf{h}_m$$

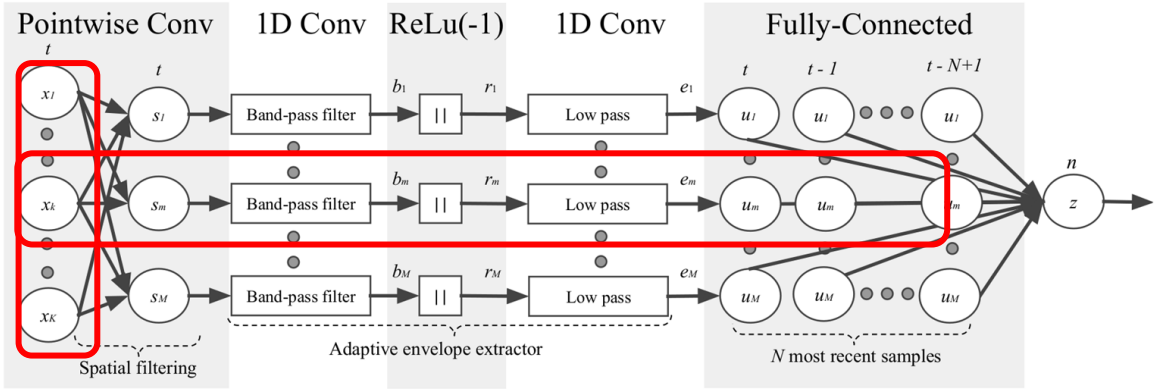




Weights interpretation in a multi-branch architecture



Take the *m*-th branch



$$\mathbf{w}_m^* = argmin_{\mathbf{w}_m}\{\| b_m(t) - \mathbf{w}_m^T \mathbf{X}(t)\mathbf{h}_m^* \|_2^2\} = argmin_{\mathbf{w}_m}\{\| b_m(n) - \mathbf{w}_m^T \mathbf{y}(t) \|_2^2\}$$

Spatial pattern of the informative source:

$$\mathbf{g}_m = E\{\mathbf{y}(t)\mathbf{y}^T(t)\}\mathbf{w}_m^* = \mathbf{R}_m^y \mathbf{w}_m^*$$

$$\mathbf{h}_m^* = argmin_{\mathbf{h}_m}\{\| b_m(t) - \mathbf{w}_m^{*T} \mathbf{X}(t)\mathbf{h}_m \|_2^2\} = argmin_{\mathbf{h}_m}\{\| b_m(t) - \mathbf{v}^T(t)\mathbf{h}_m \|_2^2\}$$

Temporal pattern of the informative source:

$$\mathbf{q}_m = E\{\mathbf{v}(t)\mathbf{v}^T(t)\}\mathbf{h}_m^* = \mathbf{R}_m^v \mathbf{h}_m^*$$

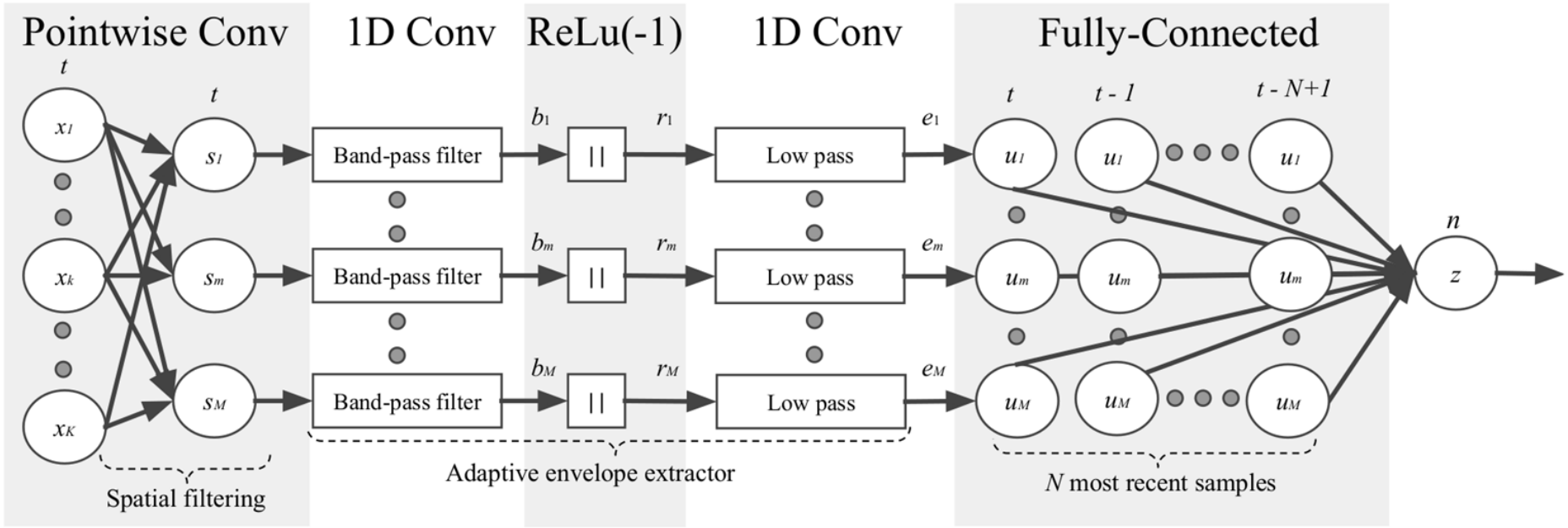
Frequency domain portrait:

$$Q_m^*(f) = P_m^{ss}(f) = P_m^{vv}(f)H_m^*(f)$$

Verbalized & colored

- When interpreting spatial weights of the m-th branch one has to take into account the spatial structure of the temporally filtered data with the m-th branch temporal filter

- When intepreting temporal weights one has to take into account the temporal (frequency domain) structure of the spatially filtered data with the m-th branch spatial filter



Common mistakes

- Several papers describing compact architectures similar to ours have been published
- Spatial weights are interpreted without taking into account branch-specific temporal filter. Instead, non filtered input data are used to compute data covariance to be applied to transform the weights into *naive spatial patterns*
- Temporal filter weights are interpreted as is, in the frequency domain using a simple FFT of weights and without considering the frequency domain properties of the input data spatially filtered with branch-specific spatial filter
- Erroneously interpreted spatial weights lead to mislocalization of pivotal neuronal sources obtained as parameters of the fitted electromagnetic
- Erroneously interpreted preclude us from doing a proper interpetation of dynamical properties of the pivotal neuronal populations

Phenomenological model

Observed data

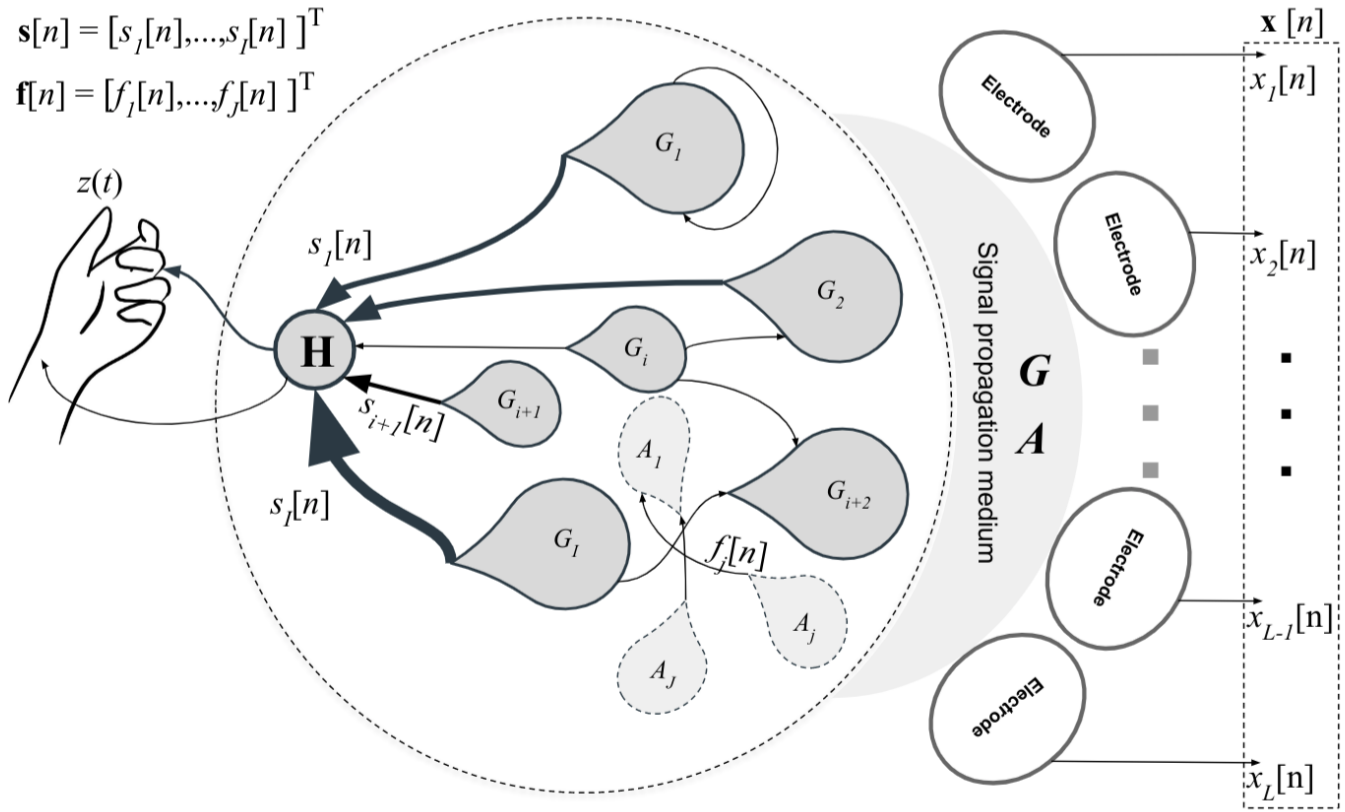
Target LFPs as measured by the sensors

Task unrelated LFP as measured by the sensors

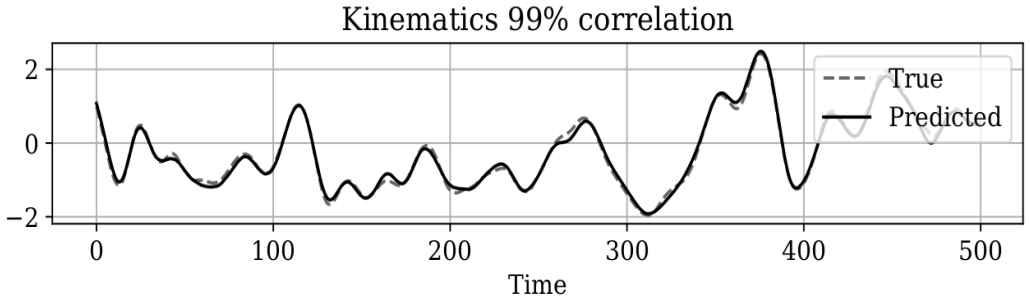
$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{A}\mathbf{f}(t) = \sum_{i=1}^{I} \mathbf{g}_i s_i(t) + \sum_{j=1}^{J} \mathbf{a}_j f_j(t)$$

$$z(t) = \overset{?}{\mathcal{F}}(\mathbf{x}(t))$$

$$z(t) = \mathbf{H}(\mathbf{env}(t)) = \mathbf{H}(\mathbf{V}(\mathbf{s}(t)))$$



Simulations, no noise



$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{A}\mathbf{f}(t) = \sum_{i=1}^{I} \mathbf{g}_i s_i(t) + \sum_{j=1}^{J} \mathbf{a}_j f_j(t)$$

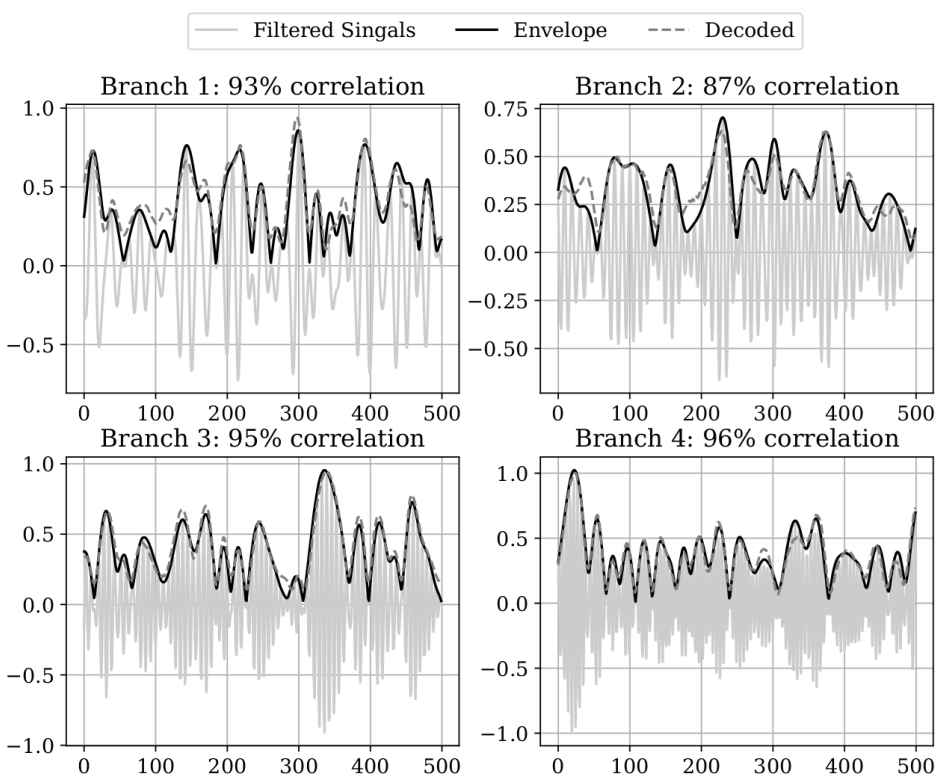
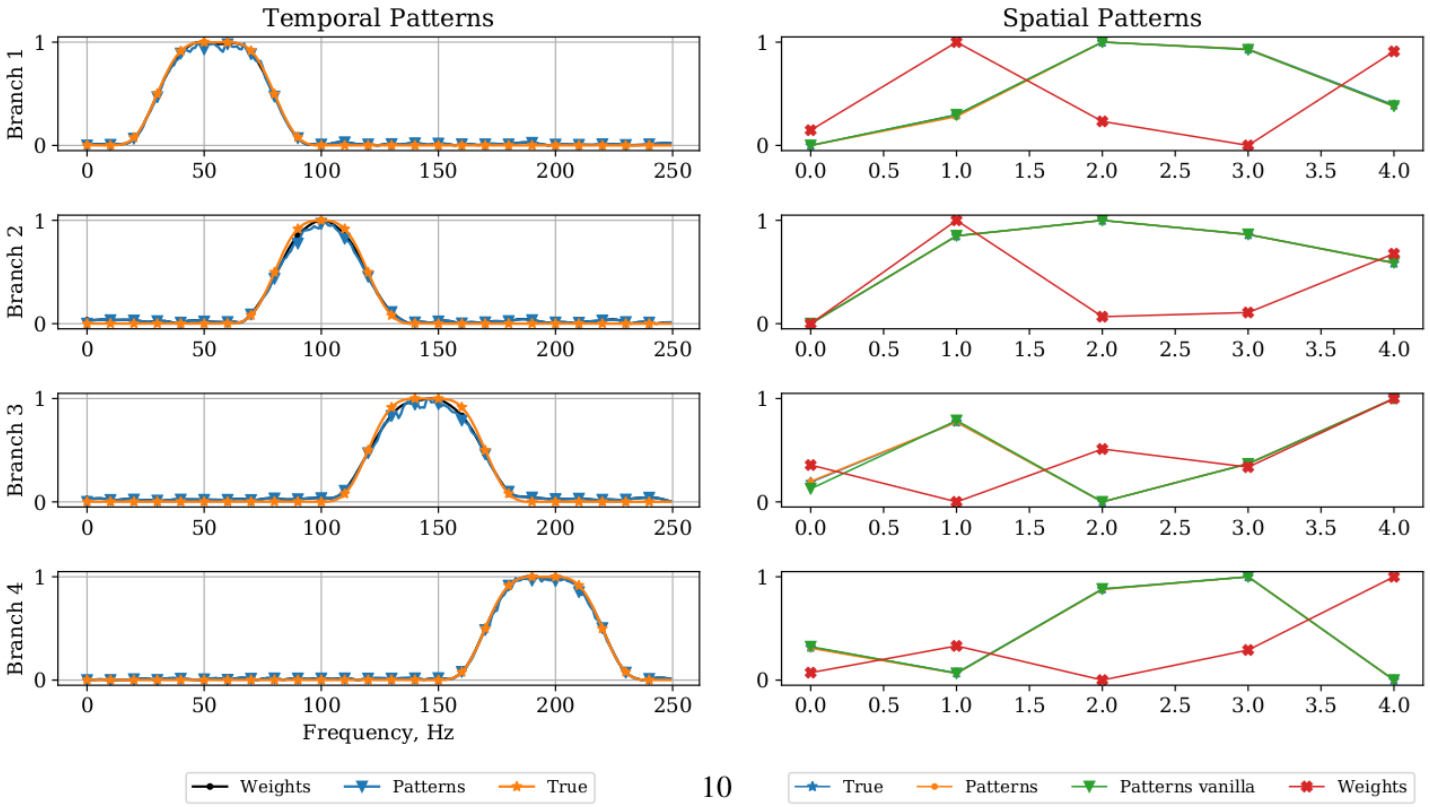
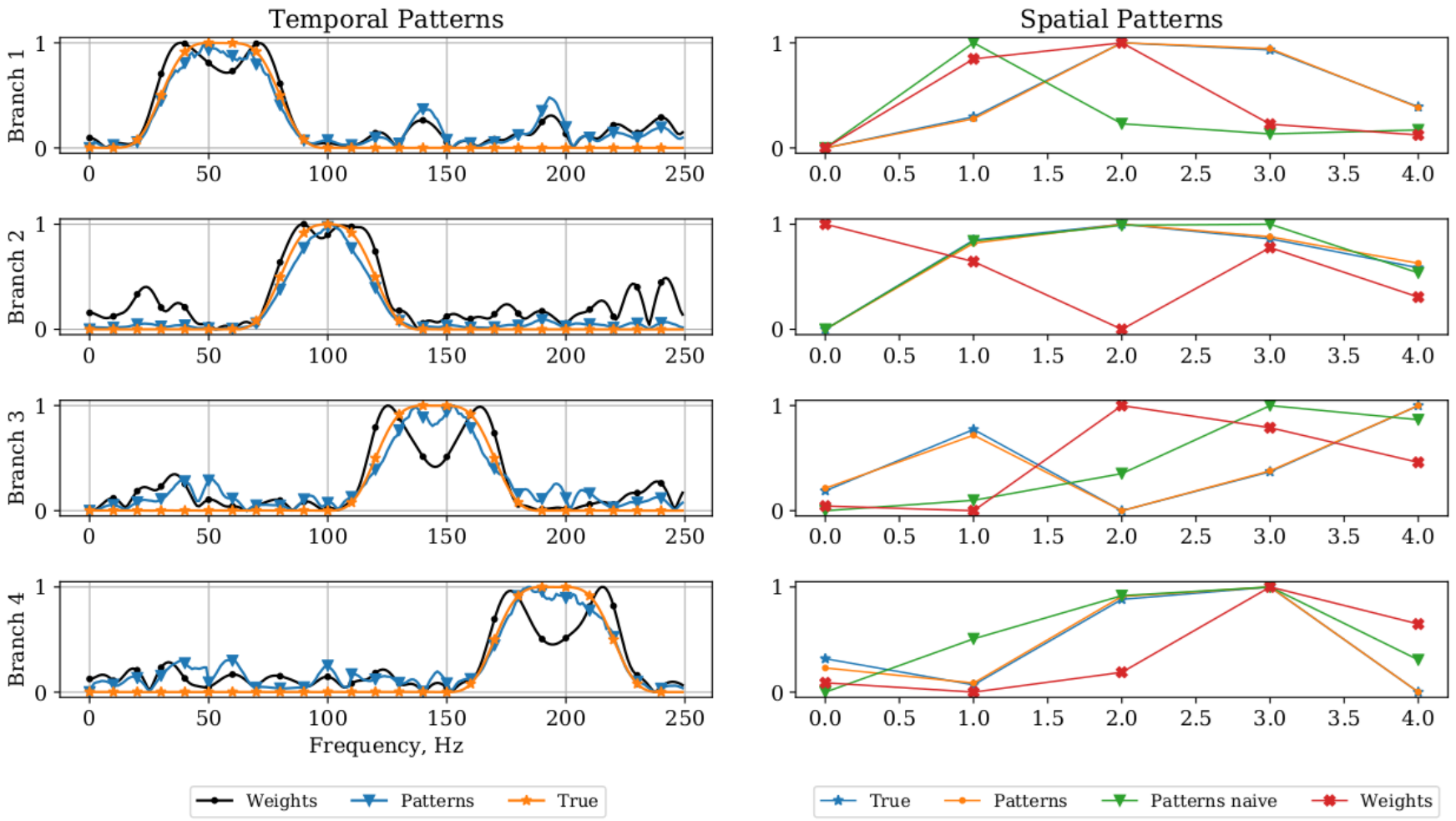


Figure 5: Branch envelopes decoding. Comparison between the true and the decoded envelope.



Simulations, colored noise

$$\mathbf{x}(t) = \mathbf{G}\mathbf{s}(t) + \mathbf{A}\mathbf{f}(t) = \sum_{i=1}^{I} \mathbf{g}_i s_i(t) + \sum_{j=1}^{J} \mathbf{a}_j f_j(t)$$



Monte-Carlo simulations

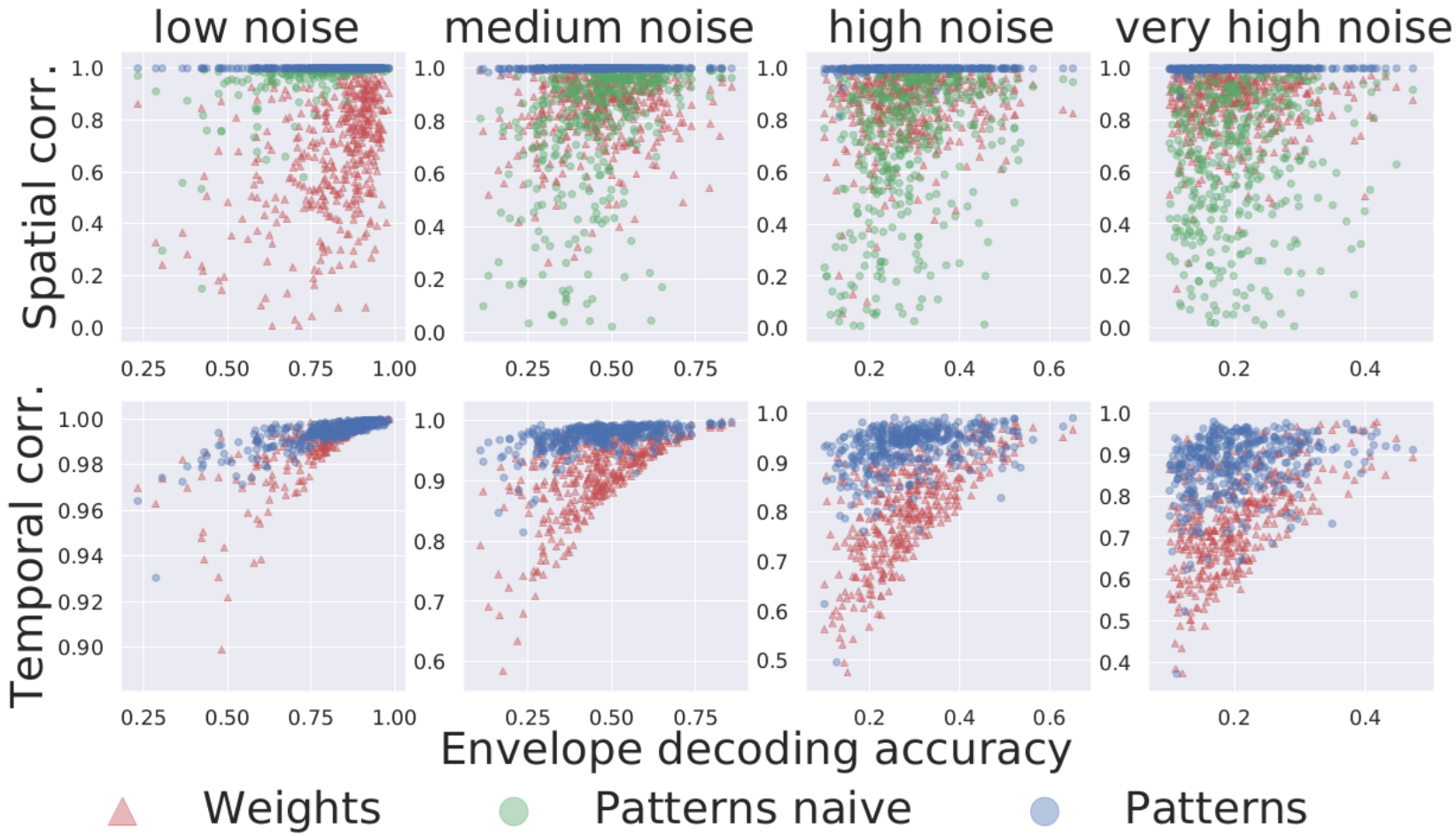
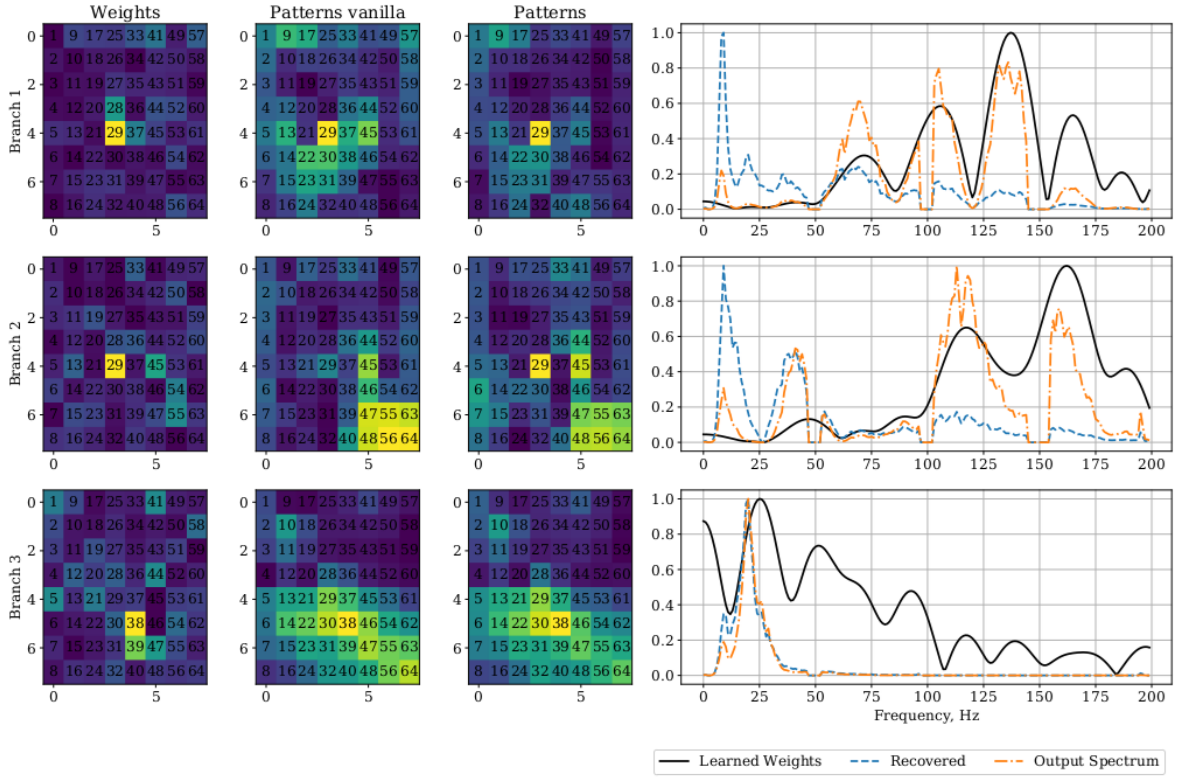


Figure 8: Monte-arlo simulations. Point coordinates reflect the achieved at each Monte Carlo trial envelope decoding accuracy (x-axis) and correlation coefficient with the true pattern (y-axis). Each point of a specific color corresponds to a single Monte Carlo trial and codes for a method used to compute patterns. *Weights* - direct weights interpretation, *Patterns naive* - spatial patterns interpretation without taking branch specific temporal filters into account, *Patterns* - the proposed method

Application to ECoG data



	Thumb	Index	Ring	Little
Subject 1	0,48	**0,79**	0,61	0,32
Subject 2	0,73	0,55	0,78	**0,79**

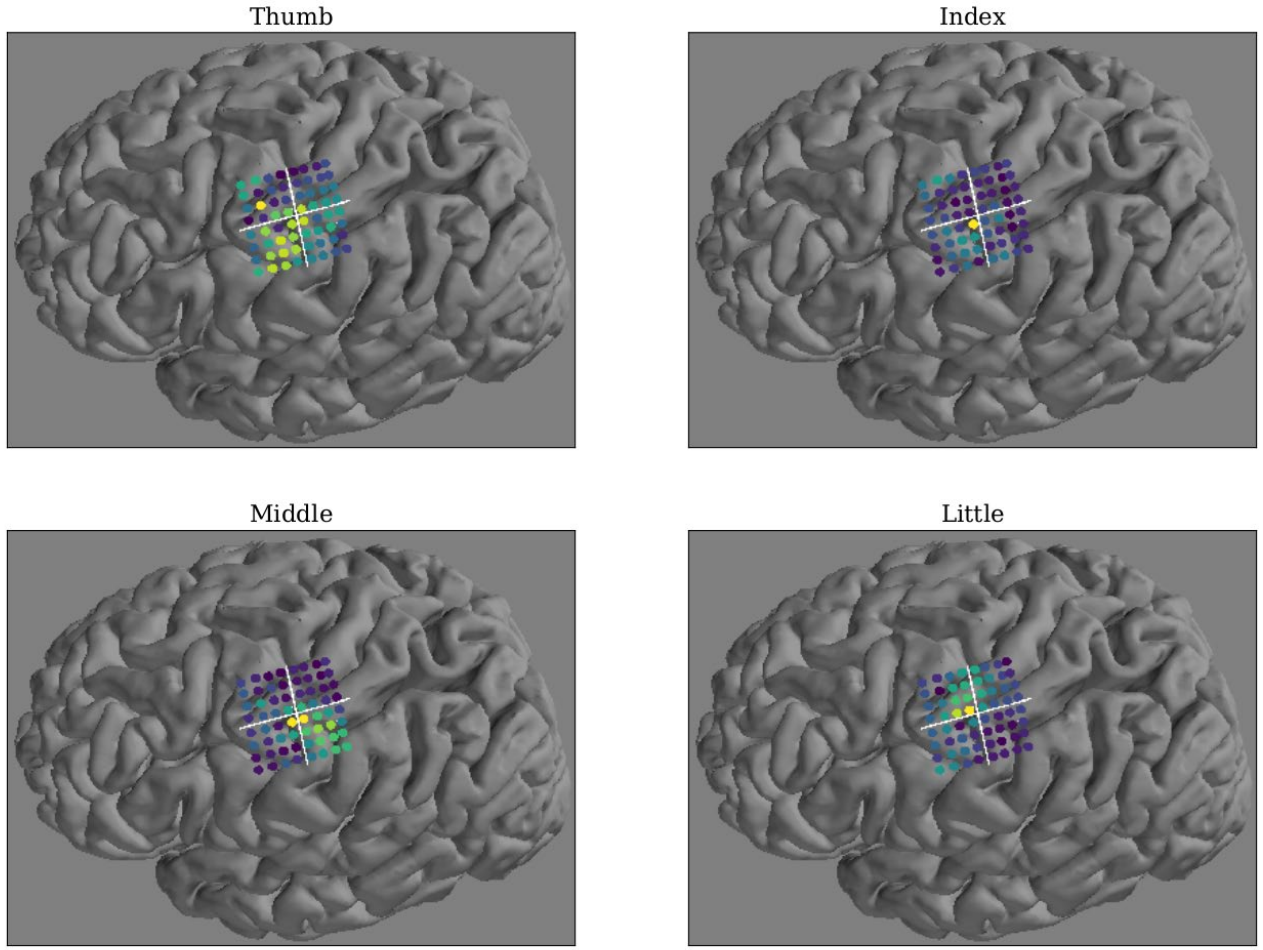
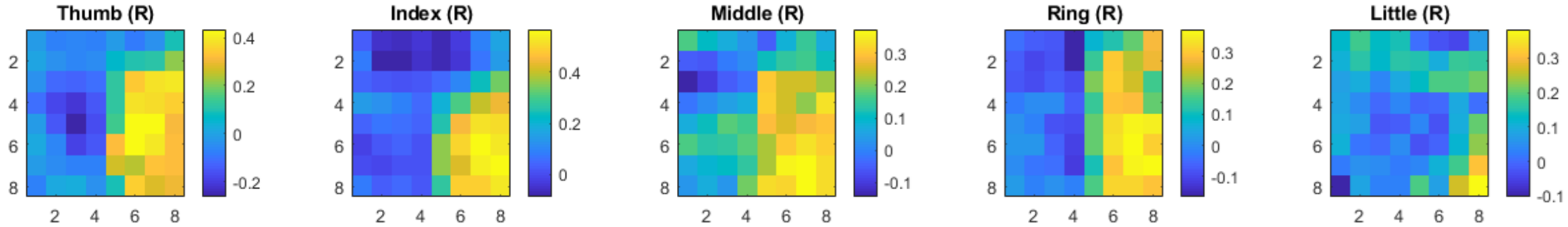
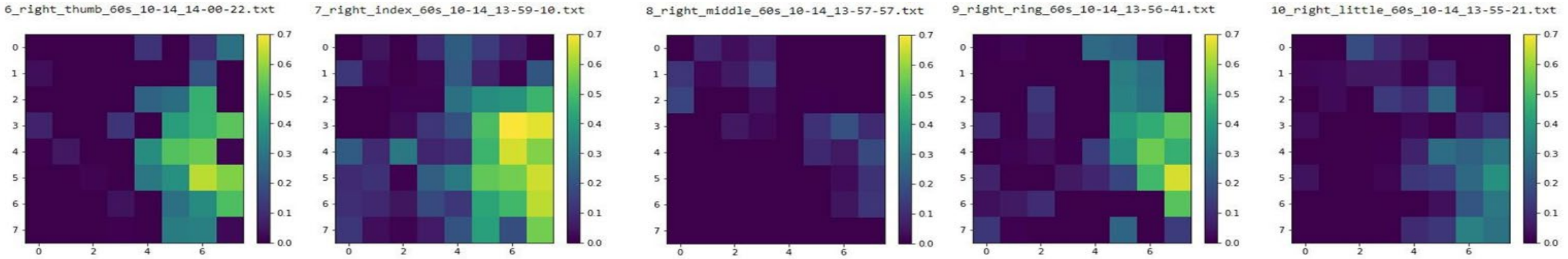


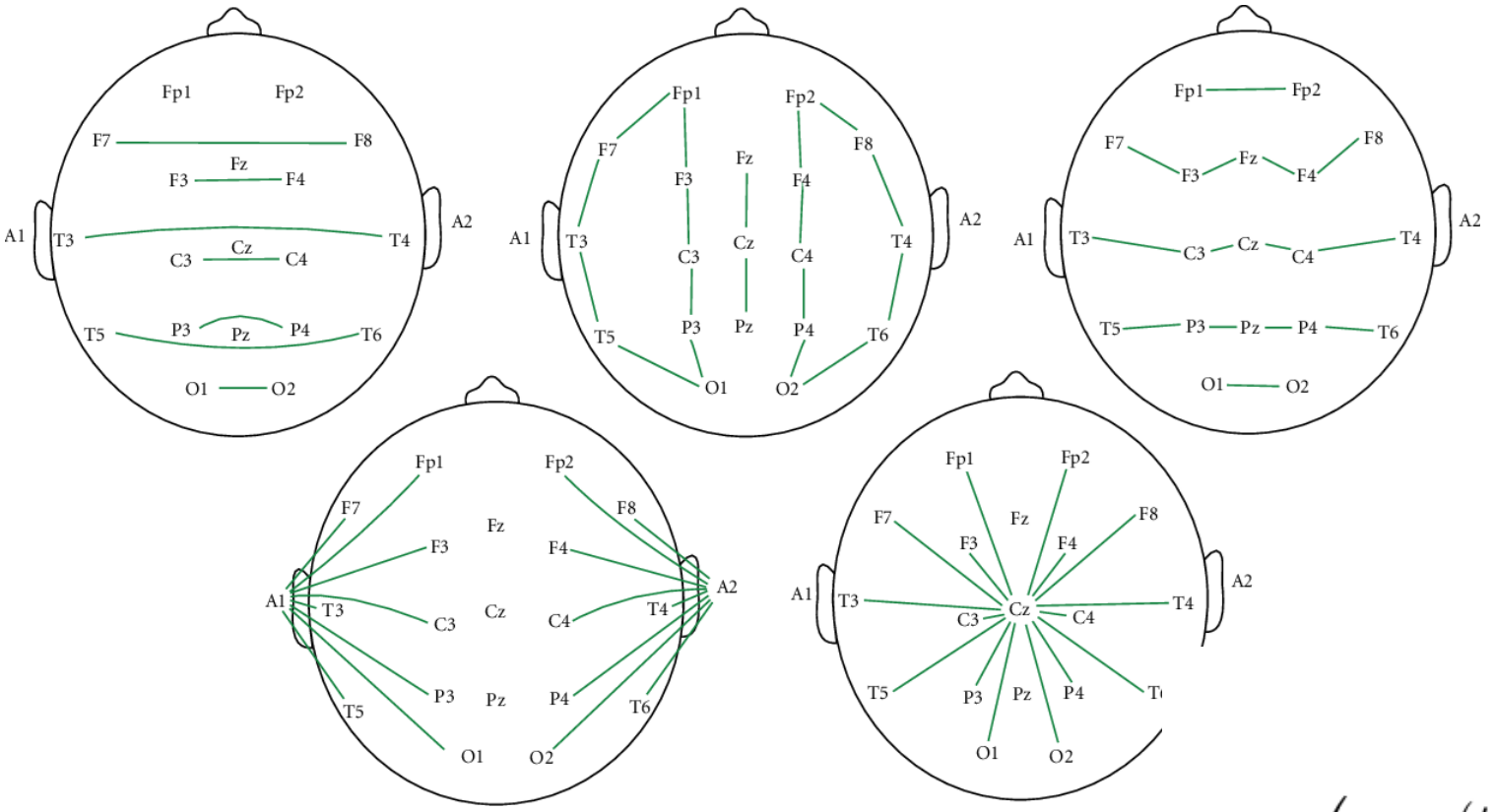
Figure 11: Spatial patterns for all available fingers mapped onto cortex

Presurgical or intraoperative passive maping of motor cortex (regression vs advaced archs)





EEG montage is just a matrix multiplication



We want to be invariant to the choice of montage. Indeed, things that are going in the brain should NOT depend on the way we measure them!

Mathematically, a montage is simply a

matrix multiplication:

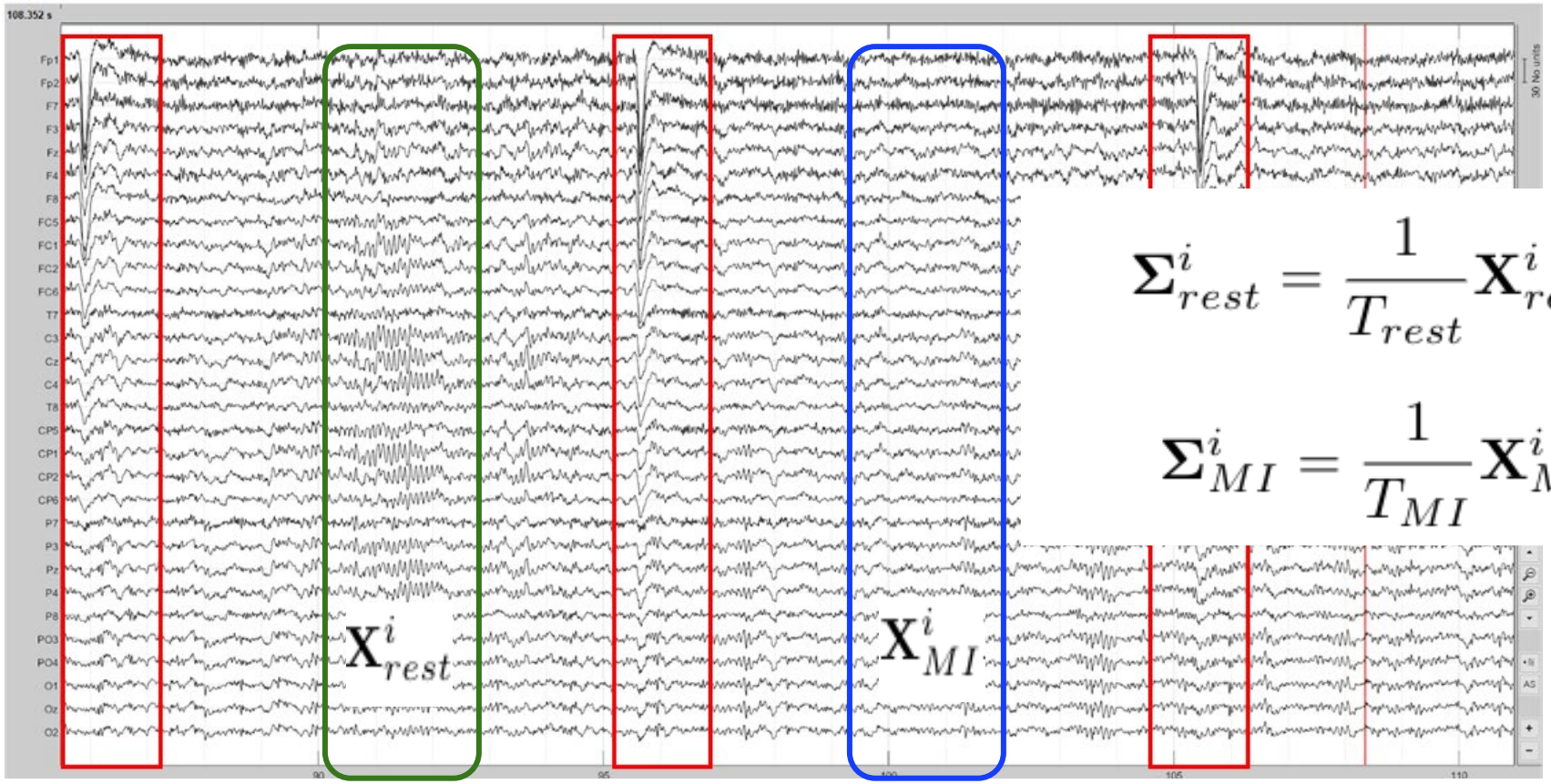
$$\mathbf{x}_m(t) = \mathbf{M}\mathbf{x}(t)$$

$$\begin{pmatrix} x_{m1}(t) \\ x_{m2}(t) \\ \dots \\ x_{mn}(t) \end{pmatrix} = \begin{pmatrix} 1 & -1 & \dots & 0 \\ 0 & 1 & \dots & -1 \\ \dots & \dots & \dots & \dots 0 \\ \dots & -1 & 1 & \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{pmatrix}$$

Use covariance matrices as features

Rest, i-th trial

Motor Imagery, i-th trial



$$\mathbf{\Sigma}^i_{rest} = \frac{1}{T_{rest}} \mathbf{X}^i_{rest} \mathbf{X}^{iT}_{rest}, \quad i = 1, \ldots, N_{rest}$$

$$\mathbf{\Sigma}^i_{MI} = \frac{1}{T_{MI}} \mathbf{X}^i_{MI} \mathbf{X}^{iT}_{MI}, \quad i = 1, \ldots, N_{MI}$$

How do we compare them?

$$\mathbf{\Sigma}^i_{rest} = \frac{1}{T_{rest}} \mathbf{X}^i_{rest} \mathbf{X}^{iT}_{rest}, \quad i = 1, \ldots, N_{rest}$$

$$\mathbf{\Sigma}^i_{MI} = \frac{1}{T_{MI}} \mathbf{X}^i_{MI} \mathbf{X}^{iT}_{MI}, \quad i = 1, \ldots, N_{MI}$$

Is Frobenius norm good enough?

$$||\mathbf{\Sigma}_{MI} - \mathbf{\Sigma}_{rest}||_F = \sqrt{\sum_{i=1}^{N} \sum_{i=1}^{N} (\mathbf{\Sigma}_{MI}[i,j] - \mathbf{\Sigma}_{rest}[i,j])^2}$$

No!

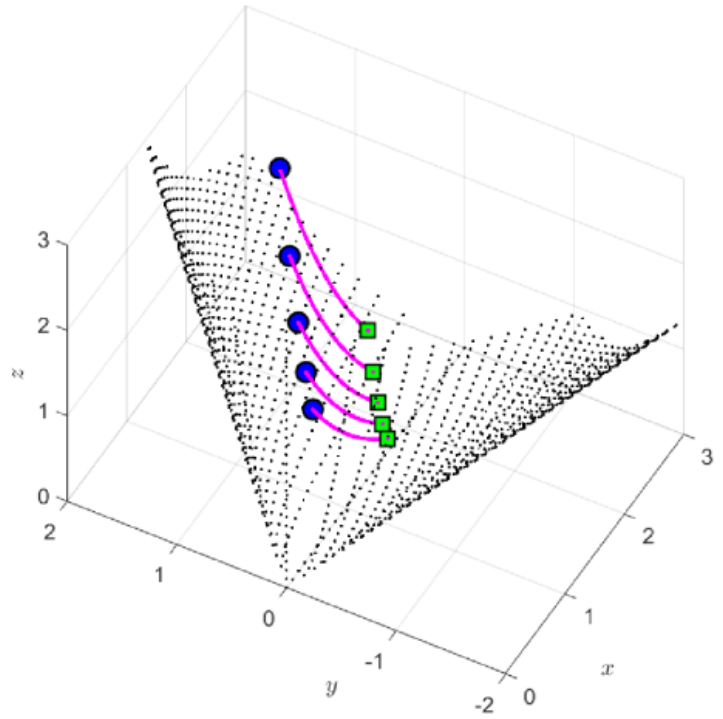


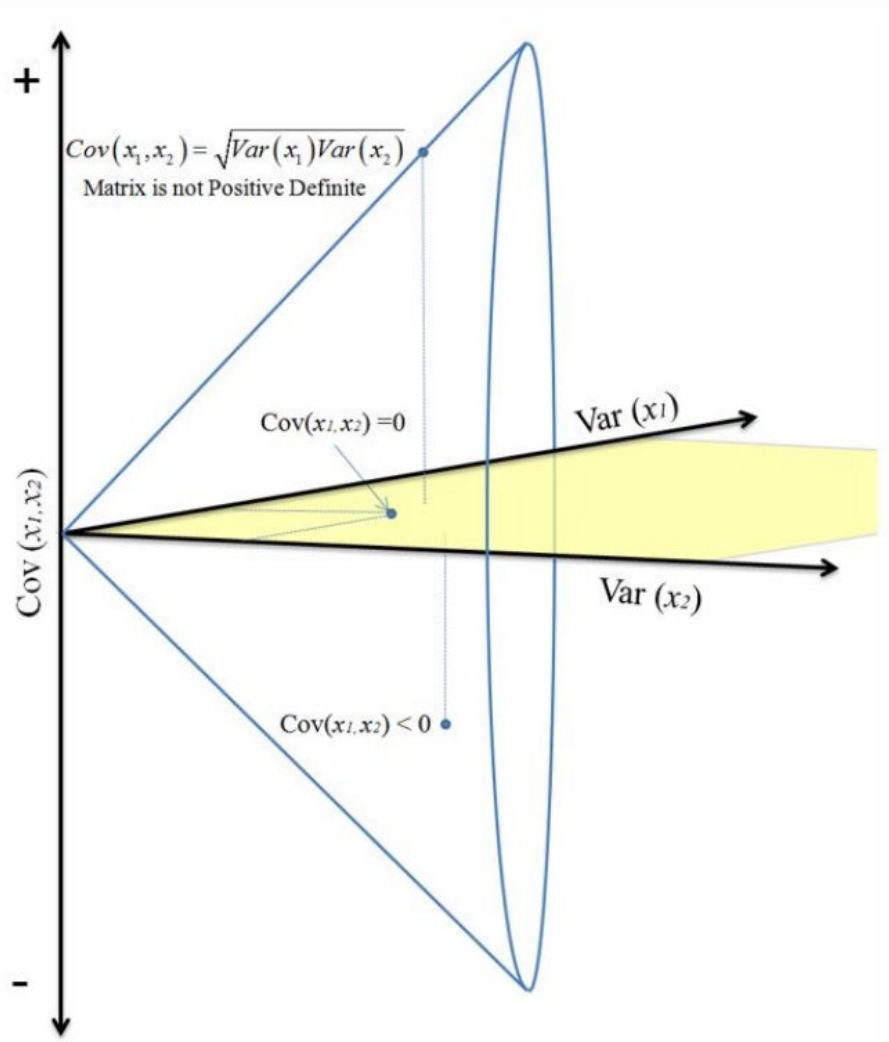
Fig. 1. The cone manifold of 2×2 SPD matrices. The black dots mark the boundary of the cone (i.e., matrices with eigenvalue zero). Each magenta curve is the geodesic between pairs of matrices (blue circles and green squares). All the geodesic curves are of the same length (i.e., the Riemannian distance between all the pairs is equal).

Covariance matrix is symmetric positive definite (SPD)

$$\begin{bmatrix} var(x) & cov(x,y) \\ cov(x,y) & var(y) \end{bmatrix}$$

$$det = var(x)var(y) - cov(x,y)^2 > 0$$

$$\left| \frac{cov(x,y)^2}{var(x)var(y)} \right| = |correlation\ coefficient| < 1$$



Geodesics is the shortest path on the MF connecting two points

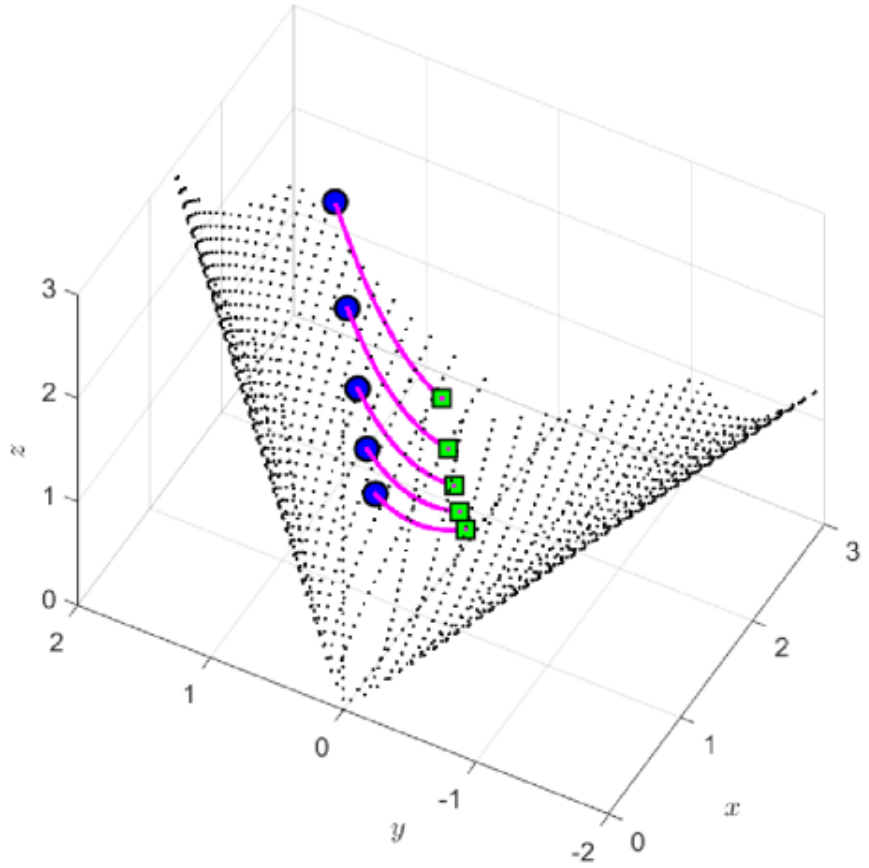
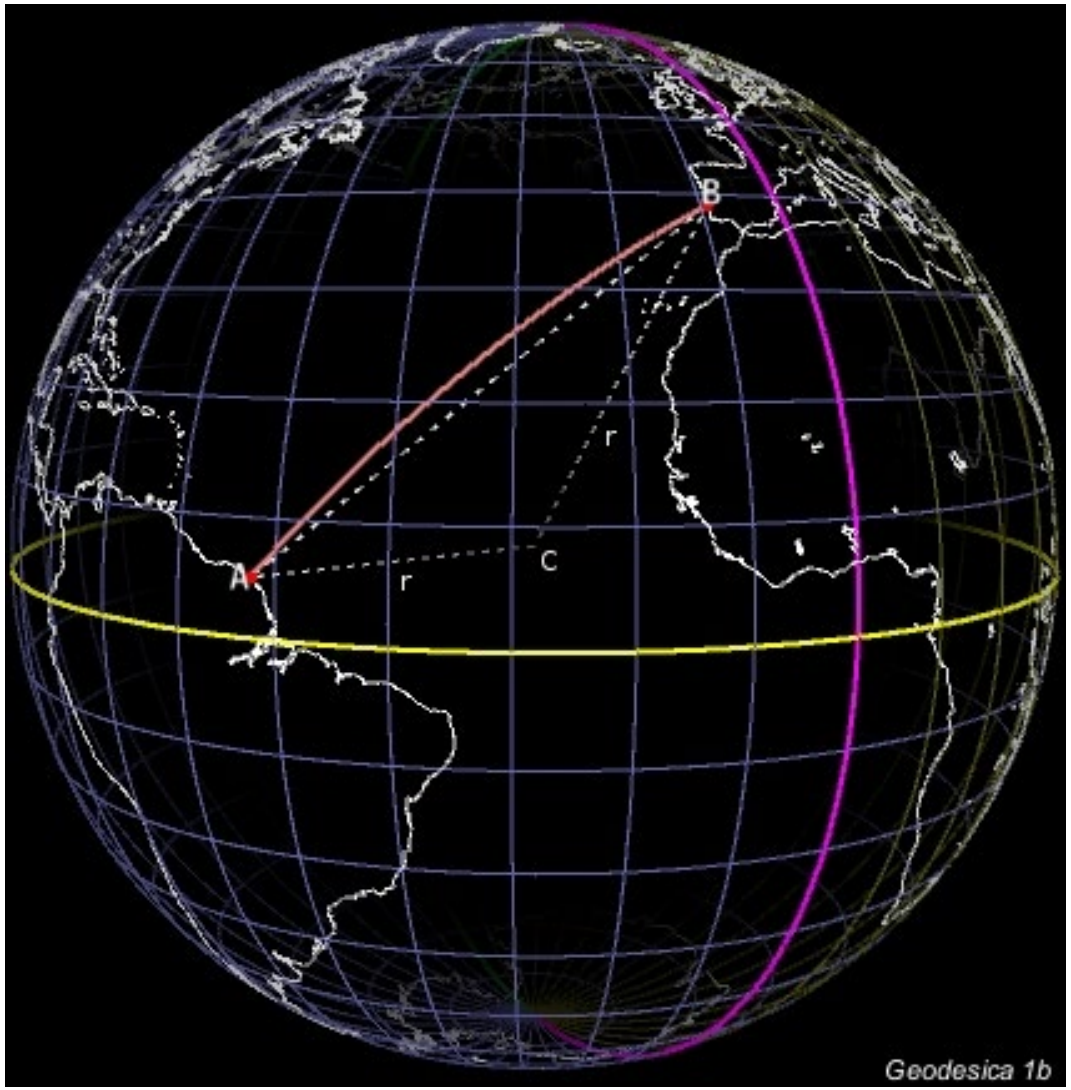
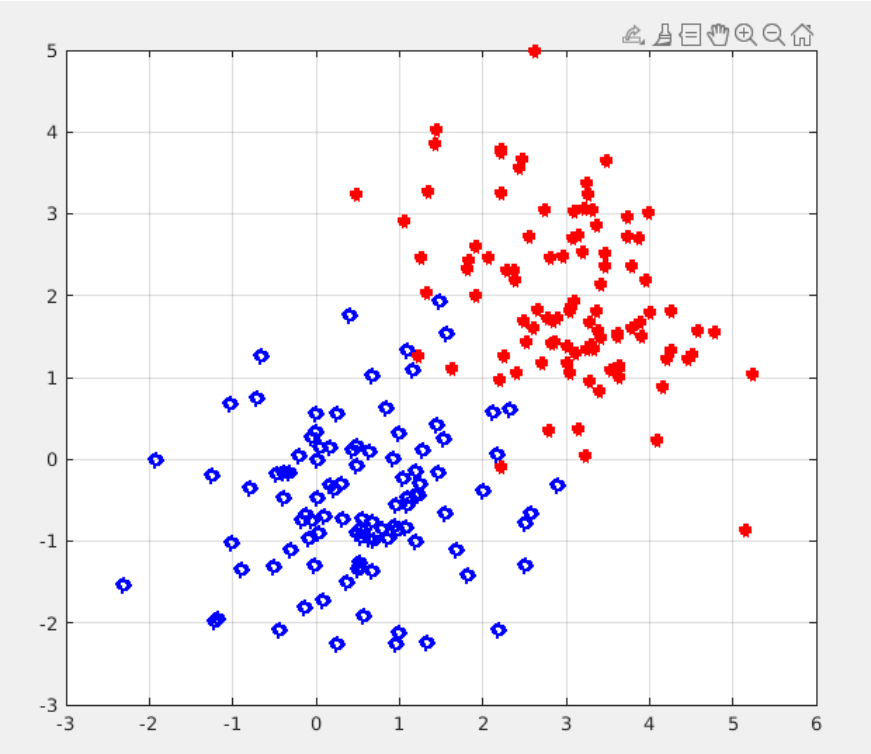


Fig. 1. The cone manifold of 2×2 SPD matrices. The black dots mark the boundary of the cone (i.e., matrices with eigenvalue zero). Each magenta curve is the geodesic between pairs of matrices (blue circles and green squares). All the geodesic curves are of the same length (i.e., the Riemannian distance between all the pairs is equal).



Riemannian manifold of positive definite symmetric matrices

Ignore the means! Are these two 2-D processes identical?



How to measure and say that two SPD matrices are equal? How to naturally measure the distance between them?

SPD matrices naturally stem as elements of the PDF of a multivariate Gaussian random process. Assume zero mean…

$$f_{\mathbf{X}}(x_1,\ldots,x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}}$$

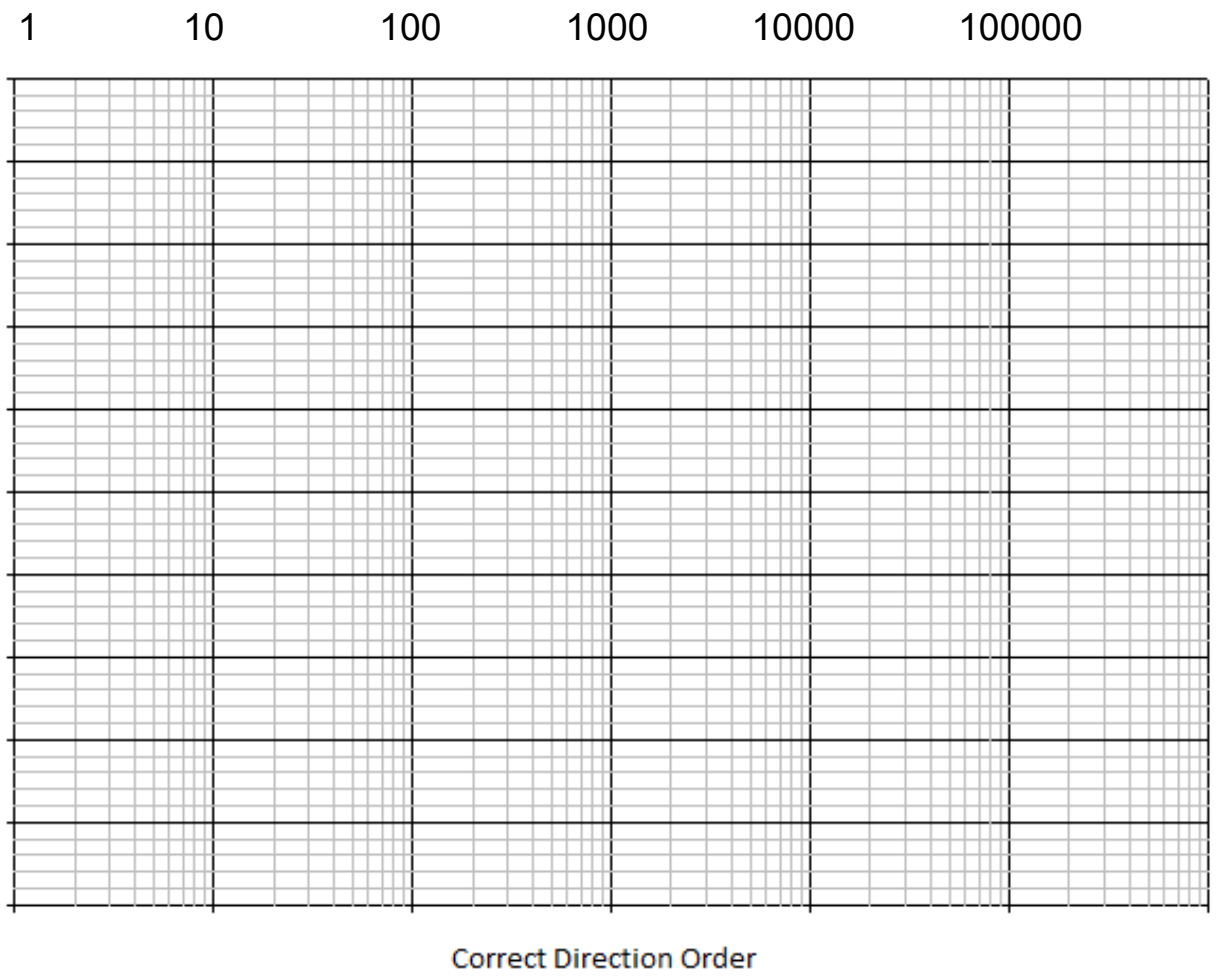
Remember that you can whiten the random process by multiplying its samples with the inverse matrix square root of its covariance matrix

$$\mathbf{z} = \boldsymbol{\Sigma}_{\mathbf{x}}^{-1/2}\mathbf{x}$$

What if you use the covariance matrix from another process and see how “white” the result is?

$$\mathbf{z}_y = \boldsymbol{\Sigma}_{\mathbf{y}}^{-1/2}\mathbf{x} \quad \boldsymbol{\Sigma}_{\mathbf{z}_y} = E\{\mathbf{z}_y\mathbf{z}_y^T\} = \boldsymbol{\Sigma}_y^{-1/2}\boldsymbol{\Sigma}_x\boldsymbol{\Sigma}_y^{-1/2} == \mathbf{I}?$$

Scale invariance? Use logarithm!



When plotted on logarithmic paper

$$|kX1 - kX2|| = ||X1 - X2||$$

Indeed:

$$|| \log(kX1) - \log(kX2)|| = ||\log(k) + \log(X1) - \log(k) - \log(X2)\}\} = ||\log(X1) - \log(X2)|| = ||\log(X1/X2)||$$

Lesson: The invariance to scaling is achieved when using log-ratio like metrics

Natural distance

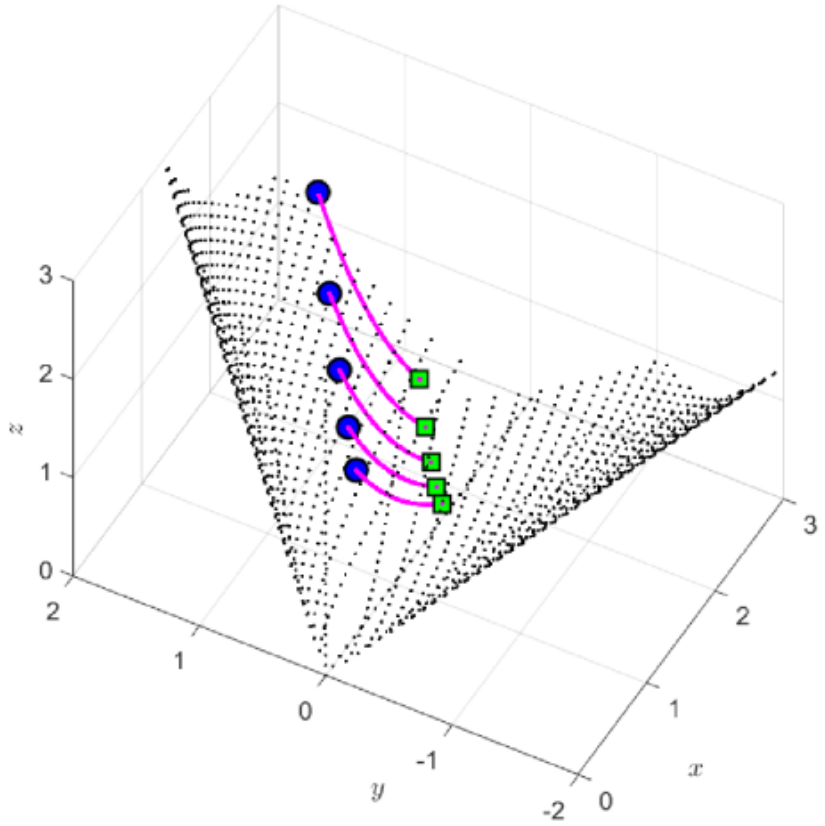


Fig. 1. The cone manifold of 2×2 SPD matrices. The black dots mark the boundary of the cone (i.e., matrices with eigenvalue zero). Each magenta curve is the geodesic between pairs of matrices (blue circles and green squares). All the geodesic curves are of the same length (i.e., the Riemannian distance between all the pairs is equal).

How do we measure that $\boldsymbol{\Sigma}_{\mathbf{z}_y} = E\{\mathbf{z}_y \mathbf{z}_y^T\} = \boldsymbol{\Sigma}_y^{-1/2} \boldsymbol{\Sigma}_x \boldsymbol{\Sigma}_y^{-1/2} == \mathbf{I}?$

Remember scale invariance? Use logarithm!

$$dist(\boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_x) = trace\left(log\left(\boldsymbol{\Sigma}_y^{-1/2} \boldsymbol{\Sigma}_x \boldsymbol{\Sigma}_y^{-1/2}\right)\right) = \sum_{i=1}^{N} log\left(\lambda_i\left(\boldsymbol{\Sigma}_y^{-1/2} \boldsymbol{\Sigma}_x \boldsymbol{\Sigma}_y^{-1/2}\right)\right)$$

It is clearly invariant with respect to multiplying data by some full rank matrix

It can be proven to be equal to the length of a geodesics connecting two matrices

Geometric mean for SPD matrices

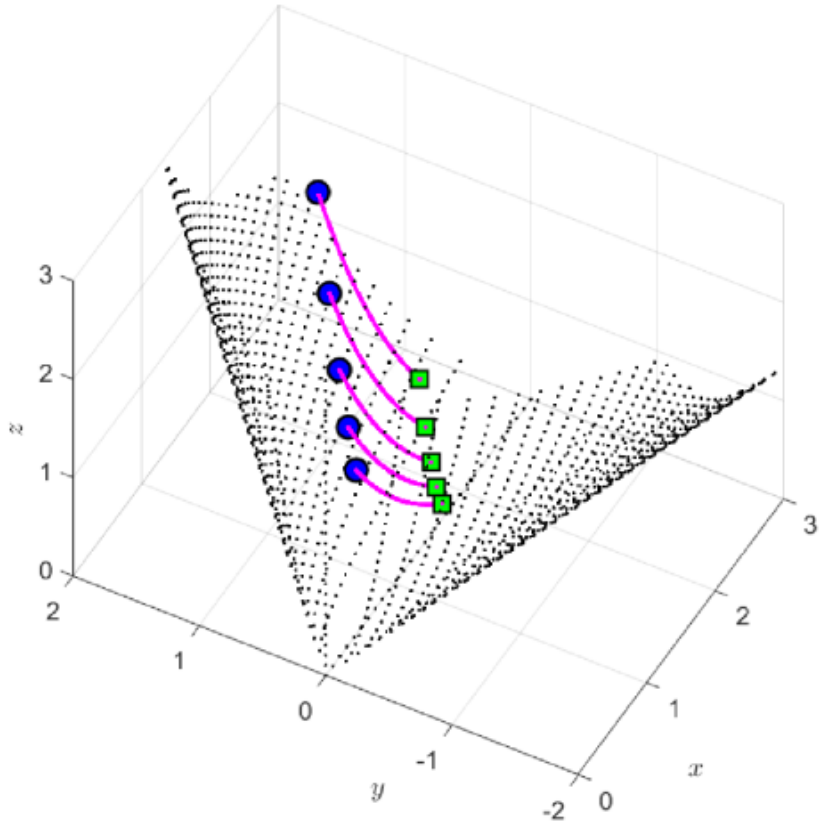


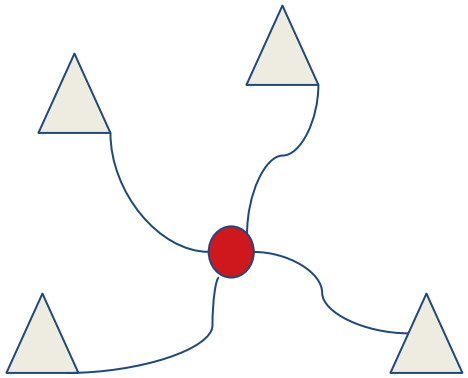
Fig. 1. The cone manifold of 2×2 SPD matrices. The black dots mark the boundary of the cone (i.e., matrices with eigenvalue zero). Each magenta curve is the geodesic between pairs of matrices (blue circles and green squares). All the geodesic curves are of the same length (i.e., the Riemannian distance between all the pairs is equal).

$$dist(\boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_x) = trace\left(log\left(\boldsymbol{\Sigma}_y^{-1/2}\boldsymbol{\Sigma}_x\boldsymbol{\Sigma}_y^{-1/2}\right)\right) = \sum_{i=1}^{N} log\left(\lambda_i\left(\boldsymbol{\Sigma}_y^{-1/2}\boldsymbol{\Sigma}_x\boldsymbol{\Sigma}_y^{-1/2}\right)\right)$$

C. *Riemannian mean*

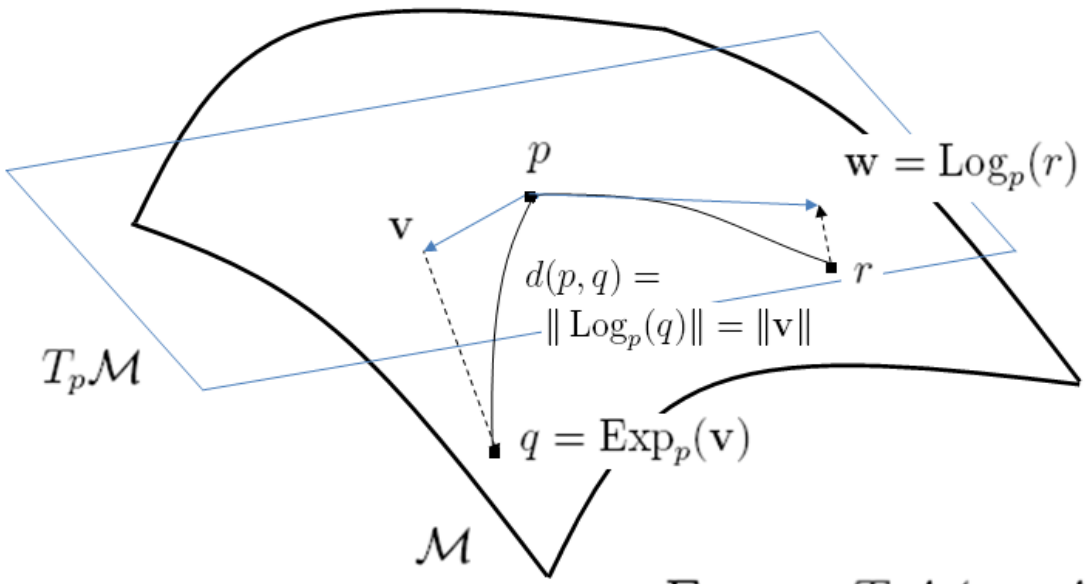
The Riemannian mean $\overline{\boldsymbol{P}}$ of a set $\{\boldsymbol{P}_i | \boldsymbol{P}_i \in \mathcal{M}\}$ is defined using the Fréchet mean:

$$\overline{\boldsymbol{P}} \triangleq \arg\min_{\boldsymbol{P}\in\mathcal{M}} \sum_i d_R^2(\boldsymbol{P}, \boldsymbol{P}_i). \tag{4}$$



Geometric mean for SPD matrices

$$dist(\boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_x) = trace\left(log\left(\boldsymbol{\Sigma}_y^{-1/2}\boldsymbol{\Sigma}_x\boldsymbol{\Sigma}_y^{-1/2}\right)\right) = \sum_{i=1}^{N} log\left(\lambda_i\left(\boldsymbol{\Sigma}_y^{-1/2}\boldsymbol{\Sigma}_x\boldsymbol{\Sigma}_y^{-1/2}\right)\right)$$



$$\mathrm{Exp}_p : \; T_p\mathcal{M} \rightarrow \mathcal{M}$$
$$\mathrm{Log}_p : \; \mathcal{M} \rightarrow T_p\mathcal{M}$$

APPENDIX F
RIEMANNIAN MEAN ALGORITHM

Algorithm 2 Riemannian mean for SPD matrices as presented in [6]

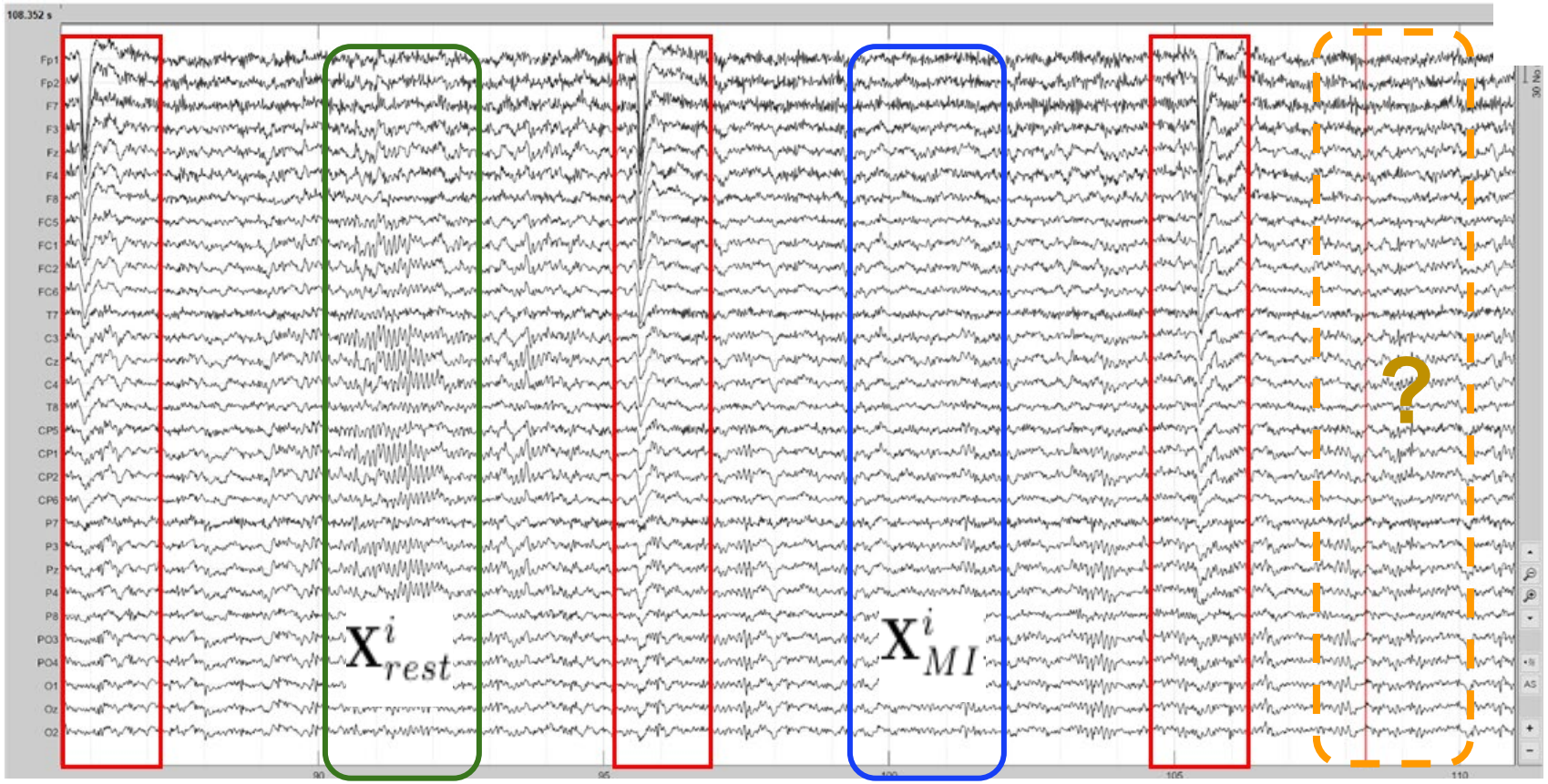
Input: a set of SPD matrices $\{\boldsymbol{P}_i \in \mathcal{M}\}_{i=1}^{N}$.
Output: the Riemannian mean matrix $\overline{\boldsymbol{P}}$.

1) Compute the initial term $\overline{\boldsymbol{P}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{P}_i$
2) **do**
   a) Compute the Euclidean mean in the tangent space: $\overline{\boldsymbol{S}} = \frac{1}{N}\sum_{i=1}^{N}\mathrm{Log}_{\overline{P}}(\boldsymbol{P}_i)$
   b) Update $\overline{\boldsymbol{P}} = \mathrm{Exp}_{\overline{P}}(\overline{\boldsymbol{S}})$
   c) **while** $\|\overline{\boldsymbol{S}}\|_F > \epsilon$ where $\|\cdot\|_F$ is the Frobenius norm.

Making decisions in the Riemannian classifier of SPDs

Rest, i-th trial

Motor Imagery, i-th trial



$$\mathbf{\Sigma}^i_{rest} = \frac{1}{T_{rest}} \mathbf{X}^i_{rest} \mathbf{X}^{iT}_{rest}, \quad i = 1, \ldots, N_{rest}$$

$$\mathbf{\Sigma}^i_{MI} = \frac{1}{T_{MI}} \mathbf{X}^i_{MI} \mathbf{X}^{iT}_{MI}, \quad i = 1, \ldots, N_{MI}$$

$$\bar{\mathbf{\Sigma}}_{MI} = E\{\mathbf{\Sigma}_{MI}\}$$

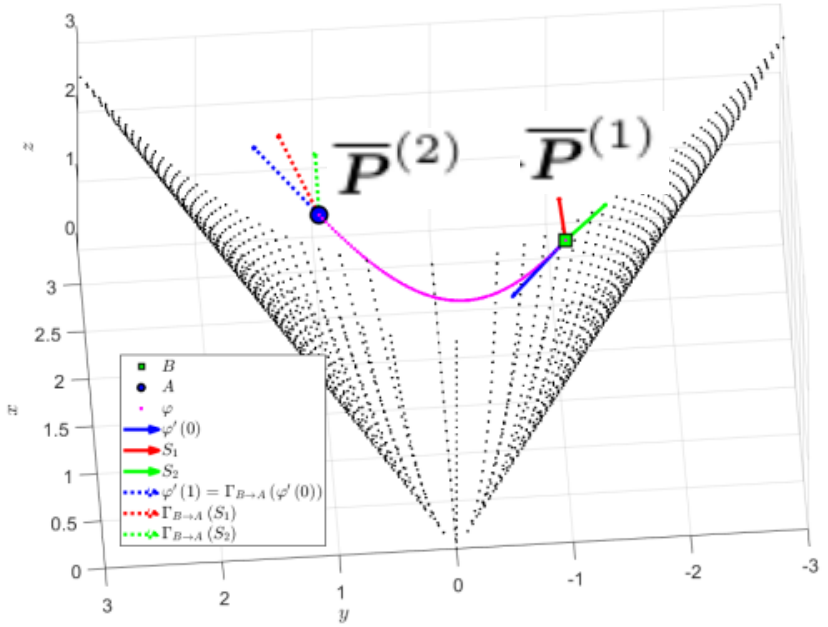
$$\bar{\mathbf{\Sigma}}_{rest} = E\{\mathbf{\Sigma}_{rest}\}$$

$$\mathbf{\Sigma}_? \rightarrow dist(\mathbf{\Sigma}_?, \mathbf{\Sigma}_{rest}) < dist(\mathbf{\Sigma}_?, \mathbf{\Sigma}_{MI}) \ : \ rest \ : \ MI$$

Parallel Transport on the Cone Manifold of SPD Matrices for Domain Adaptation

Or Yair, *Student Member, IEEE,* Mirela Ben-Chen, and Ronen Talmon, *Member, IEEE*

Domain adaptation via parallel transport over the SPD manifold

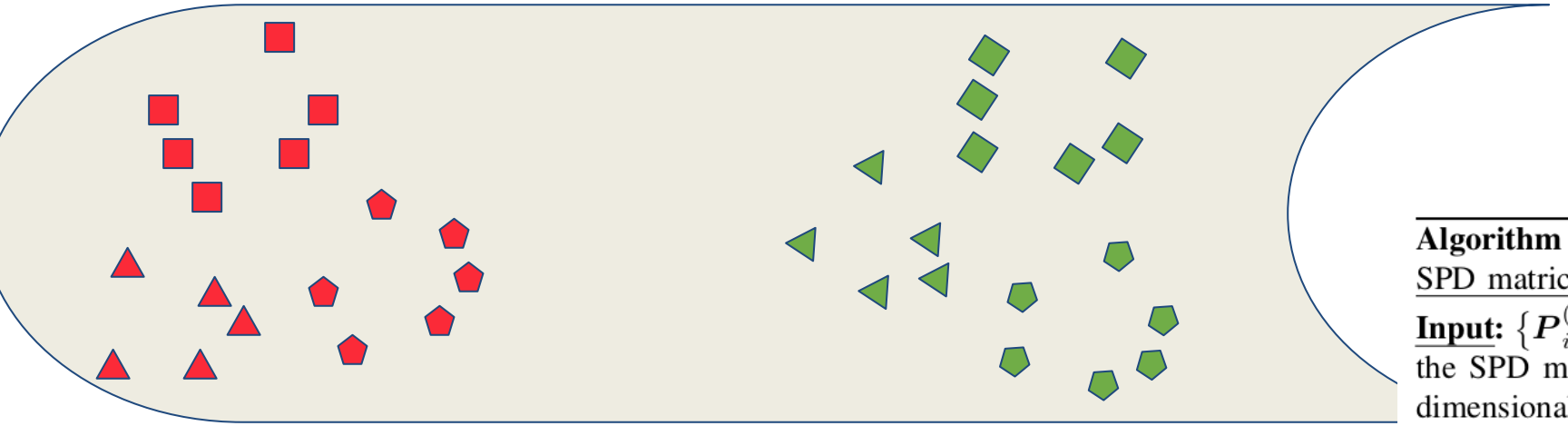


$$\Psi\left(\boldsymbol{P}_i^{(2)}\right) = \Gamma_{\overline{\boldsymbol{P}}^{(2)}\to\overline{\boldsymbol{P}}^{(1)}}\left(\boldsymbol{P}_i^{(2)}\right) = \boldsymbol{E}\boldsymbol{P}_i^{(2)}\boldsymbol{E}^T$$

$$\boldsymbol{E} \triangleq \left(\overline{\boldsymbol{P}}^{(1)}\left(\overline{\boldsymbol{P}}^{(2)}\right)^{-1}\right)^{\frac{1}{2}}$$

Fig. 2. Illustration of the PT on the SPD manifold. $\boldsymbol{A}$ and $\boldsymbol{B}$ are two SPD matrices, and φ is the unique geodesic between them. We plot three vectors in $\mathcal{T}_{\boldsymbol{B}}\mathcal{M}$: $\varphi'(0)$, S_1 and S_2 along with their corresponding parallel transported vectors to $\mathcal{T}_{\boldsymbol{A}}\mathcal{M}$ using $\Gamma_{\boldsymbol{B}\to\boldsymbol{A}}$.

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

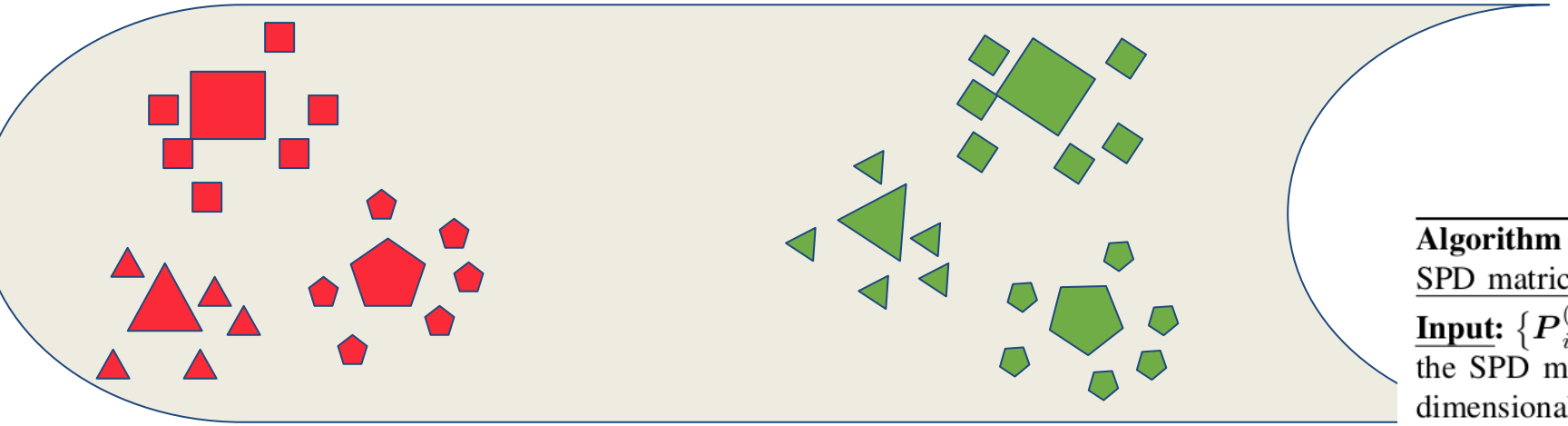
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

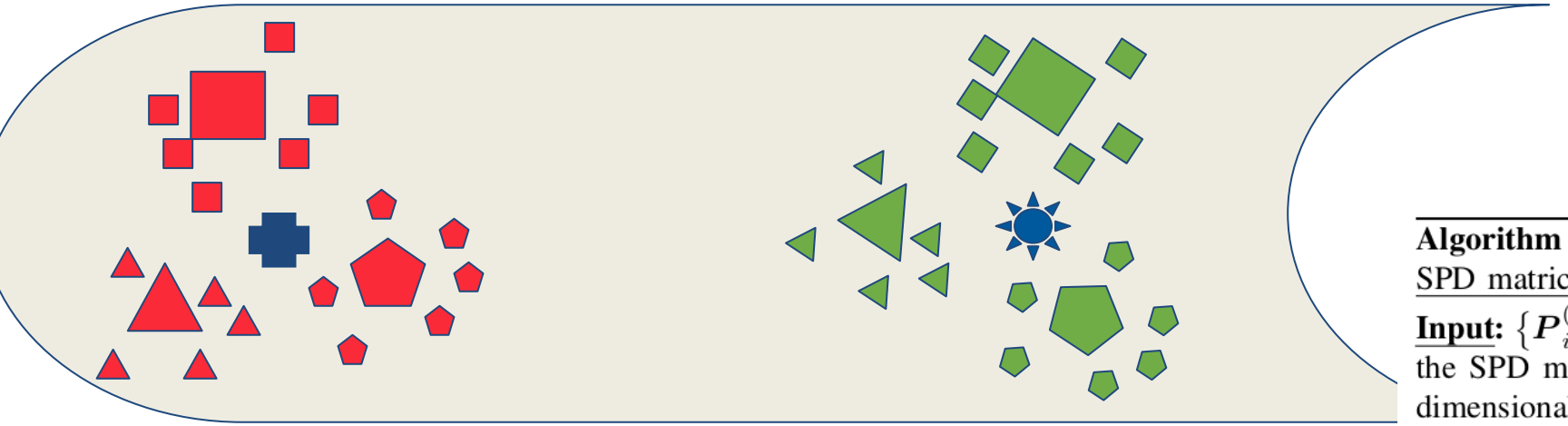
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

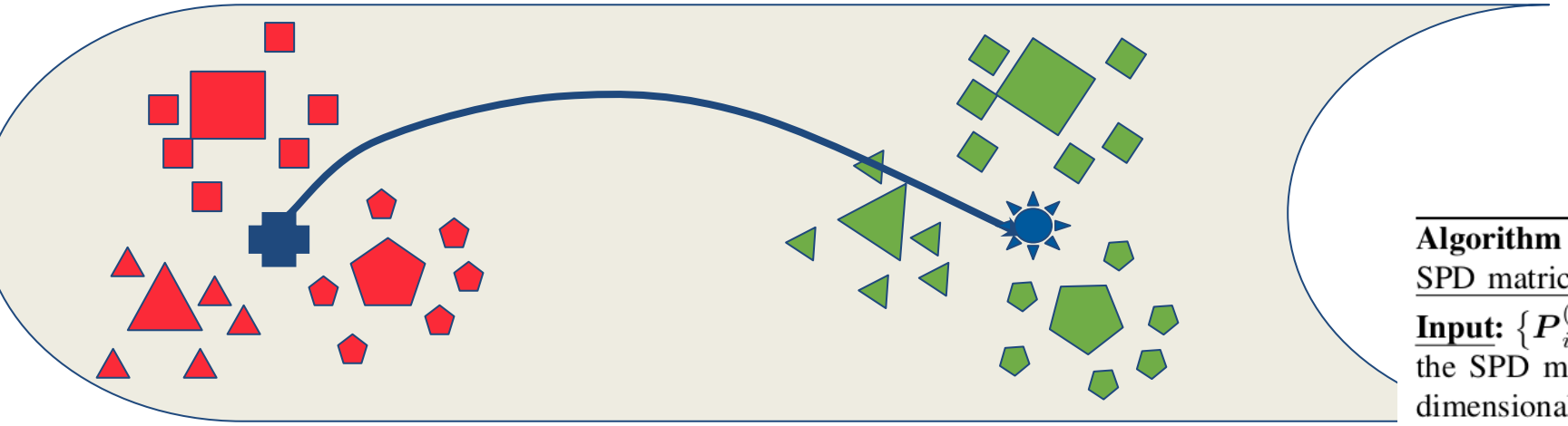
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

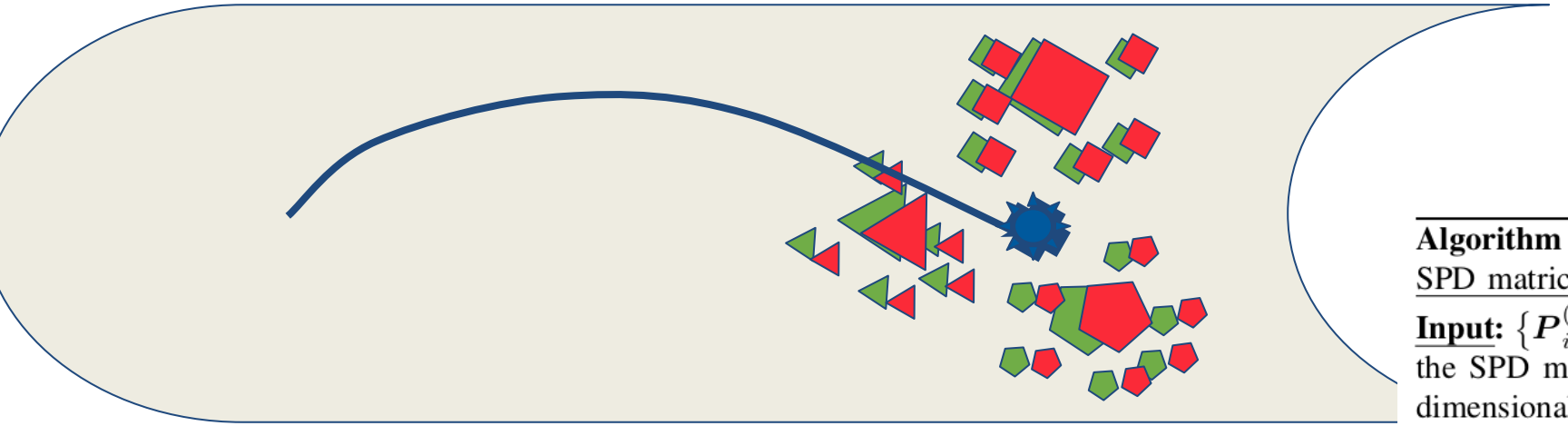
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

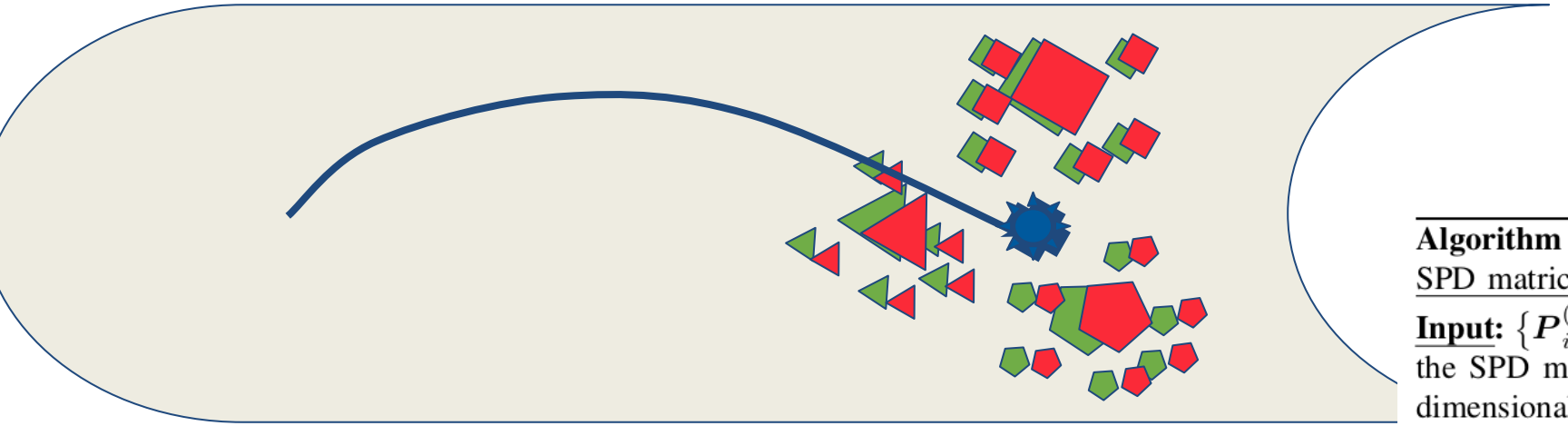
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

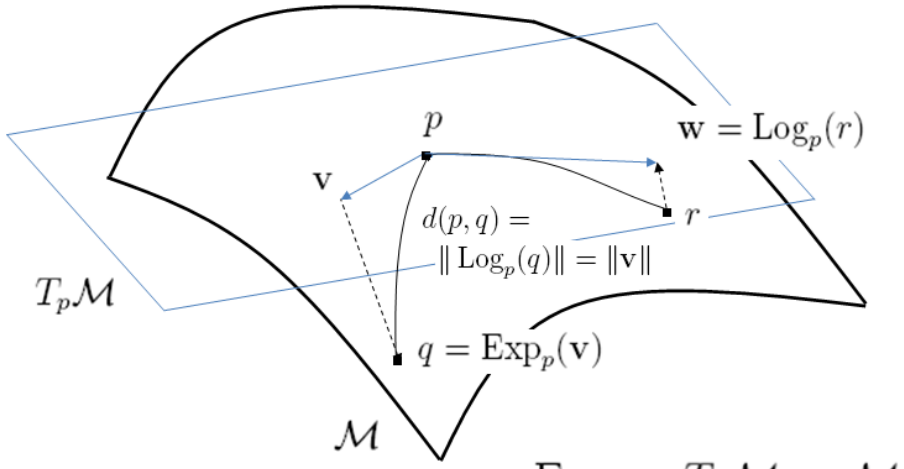
$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold





$$\mathrm{Exp}_p : \ T_p\mathcal{M} \to \mathcal{M}$$
$$\mathrm{Log}_p : \ \mathcal{M} \to T_p\mathcal{M}$$

Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

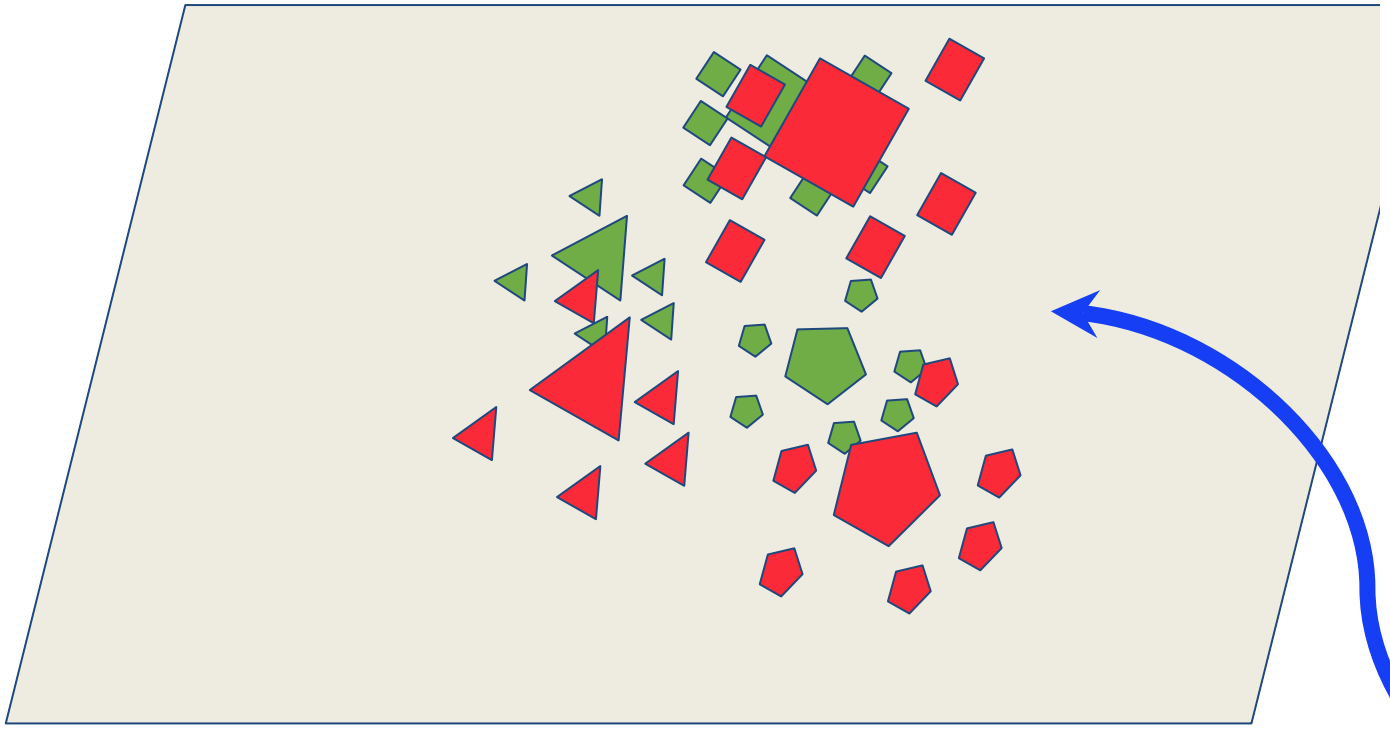
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

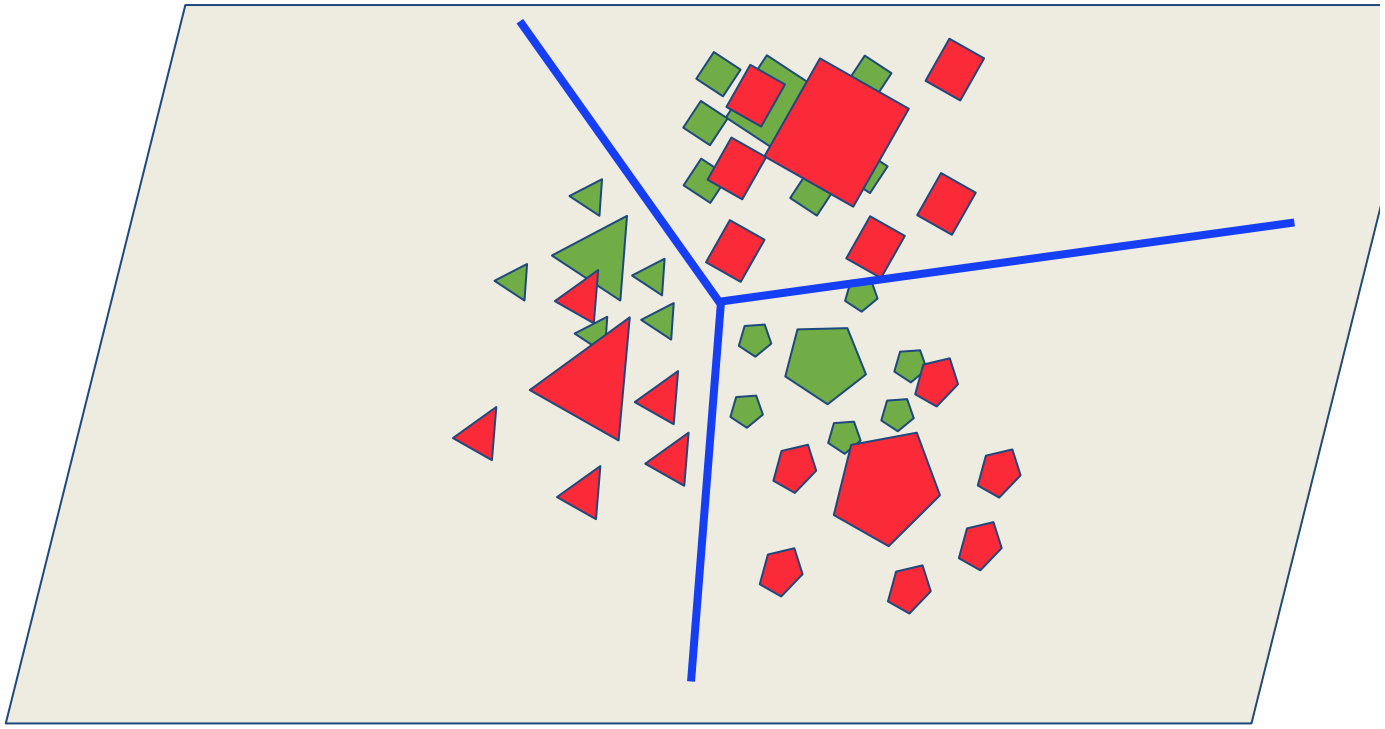
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

Domain adaptation via parallel transport over the SPD manifold



Algorithm 1 Domain adaptation using Parallel Transport for SPD matrices

Input: $\{\boldsymbol{P}_i^{(1)}\}_{i=1}^{N_1}, \{\boldsymbol{P}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\boldsymbol{P}_i^{(K)}\}_{i=1}^{N_K}$ where $\boldsymbol{P}_i^{(k)}$ is the SPD matrix associated with the i-th element (e.g., high-dimensional time-series) in the k-th subset.

Output: $\{\tilde{\boldsymbol{S}}_i^{(1)}\}_{i=1}^{N_1}, \{\tilde{\boldsymbol{S}}_i^{(2)}\}_{i=1}^{N_2}, \ldots, \{\tilde{\boldsymbol{S}}_i^{(K)}\}_{i=1}^{N_K}$ where $\tilde{\boldsymbol{S}}_i^{(k)}$ is the new representation of $\boldsymbol{P}_i^{(k)}$ in a Euclidean space.

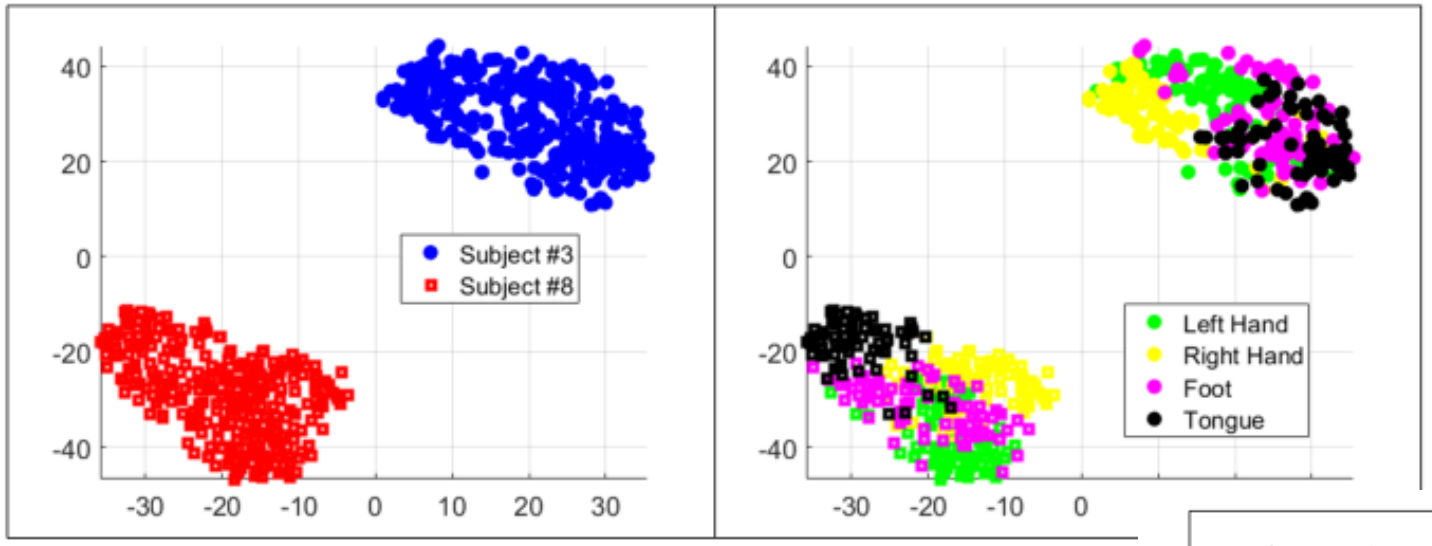
1) **For** each $k \in \{1, 2, \ldots, K\}$, compute $\overline{\boldsymbol{P}}^{(k)}$ the Riemannian mean of the subset $\{\boldsymbol{P}_i^{(k)}\}$.
2) Compute $\hat{\boldsymbol{P}}$, the Riemannian mean of $\{\overline{\boldsymbol{P}}^{(k)}\}_{k=1}^{K}$.
3) **For** all k and all i, apply Parallel Transport using (7):

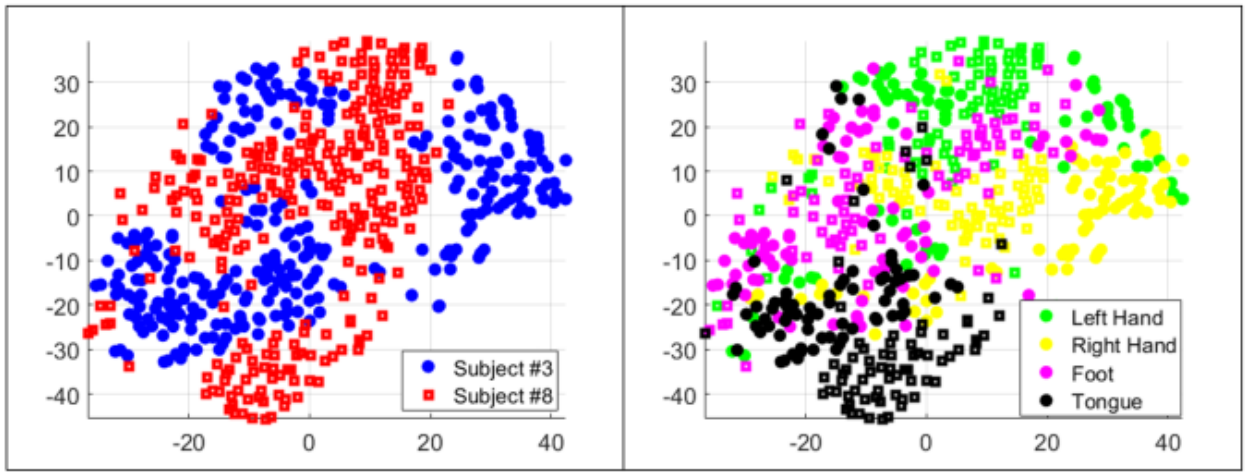
$$\boldsymbol{\Gamma}_i^{(k)} = \Gamma_{\overline{\boldsymbol{P}}^{(k)} \to \hat{\boldsymbol{P}}}\left(\boldsymbol{P}_i^{(k)}\right).$$

4) **For** all k and all i, project the transported matrix to the tangent space via:

$$\tilde{\boldsymbol{S}}_i^{(k)} = \log\left(\hat{\boldsymbol{P}}^{-\frac{1}{2}} \boldsymbol{\Gamma}_i^{(k)} \hat{\boldsymbol{P}}^{-\frac{1}{2}}\right).$$

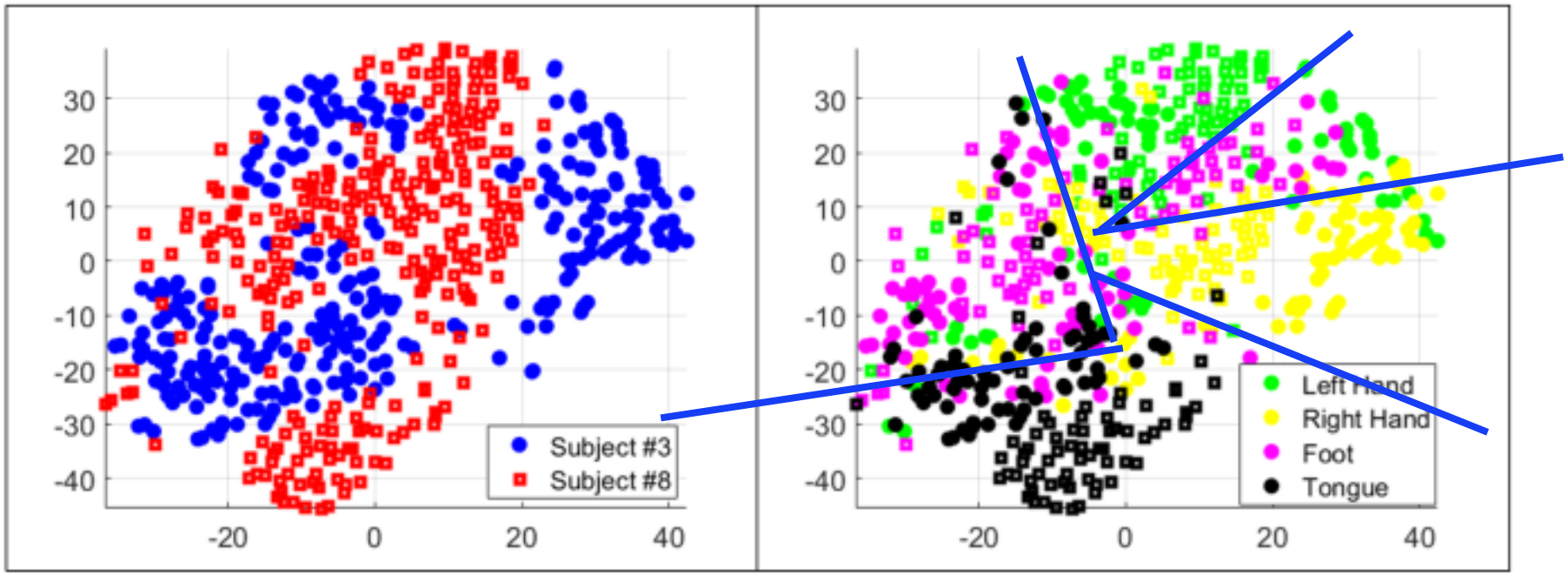
Domain adaptation via parallel transport over the SPD manifold





(c)

Domain adaptation via parallel transport over the SPD manifold



(c)

Domain adaptation via parallel transport over the SPD manifold

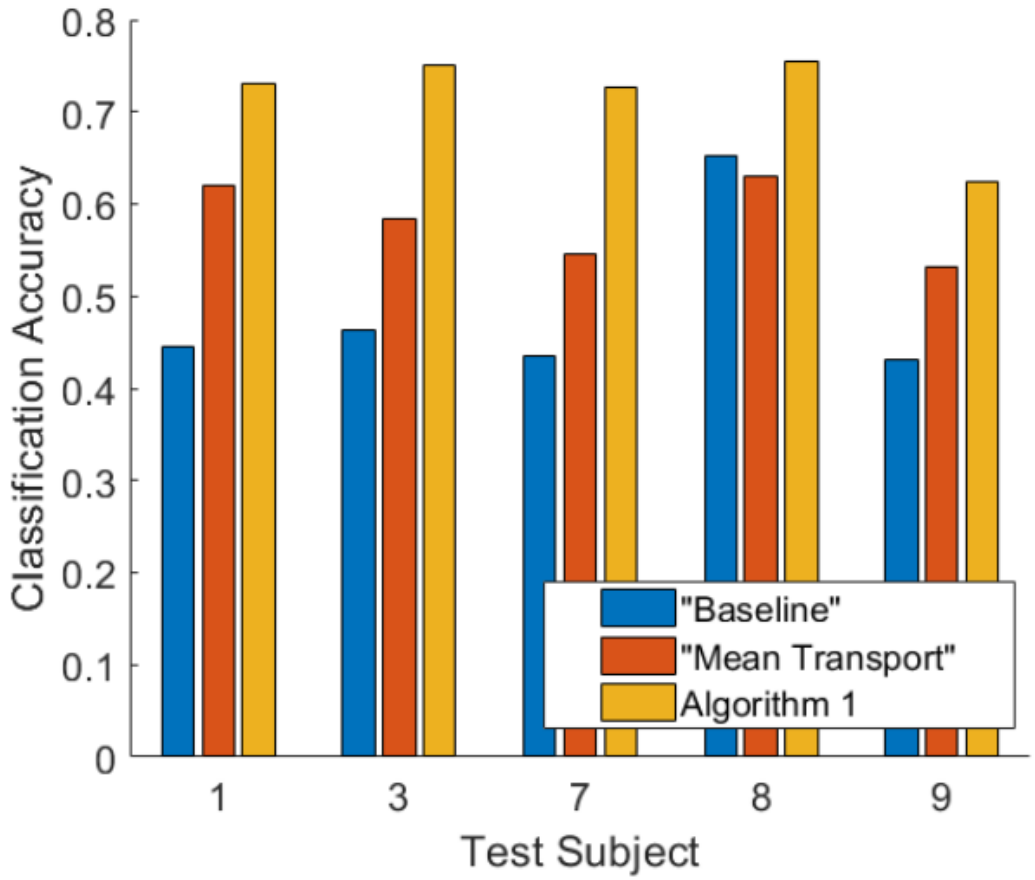


Fig. 7. The classification accuracy obtained by the three competing methods.