kCSD-python, a tool for reliable current source density estimation

Chaitanya Chintaluri[1,2,†], Marta Kowalska[1,†], Władysław Średniawa[1,3], Michał Czerwiński[1], Jakub M. Dzik[1], Joanna Jędrzejewska-Szmek[1], Daniel K. Wójcik[1,*]

1 Laboratory of Neuroinformatics, Department of Neurophysiology, Nencki Institute of Experimental Biology of Polish Academy of Sciences, 3 Pasteur Str., 02-093 Warsaw, Poland
2 Centre for Neural Circuits and Behaviour, Department of Physiology Anatomy and Genetics, University of Oxford, Oxford, UK
3 University of Warsaw, Faculty of Biology, Department of Zoology, 1 Miecznikowa Str., 02-096 Warsaw, Poland

*** d.wojcik@nencki.gov.pl**
† joint first authors

July 19, 2019

Abstract

Kernel Current Source Density (kCSD), which we introduced in 2012, is a kernel-based method to estimate current source density (CSD) from extracellular potentials recorded with arbitrarily placed electrodes. Estimating reconstruction errors in CSD has been an outstanding challenge. To address it, here we revisit kCSD and explore its mathematical underpinnings. First, we quantify the information that can be recovered from extracellular recordings for a given setup, by introducing eigensources — a set of basic CSD profiles, which form the basis of estimation space. Next, we investigate the effect of relative placement of basis sources and electrodes on the reconstruction fidelity. We show that the correct distribution of sources is crucial for the reconstruction, in particular, CSD reconstruction is possible even for badly misplaced electrodes. We also introduce L-curve, a new method for choosing reconstruction parameters, in addition to the previously used cross-validation. Finally, we propose two types of diagnostics of reconstruction veracity, error propagation map and reliability map. For any given setup, the error propagation map indicates how the electrode noise propagates to the reconstructed CSD and the reliability map illustrates the point-wise reliability of kCSD estimation. The kCSD method and the additional techniques introduced here are implemented in `kCSD-python`, a new Python package provided under an open license. kCSD-python's features and usage are highlighted with a jupyter notebook tutorial. This new tool can perform CSD estimations for 1D, 2D, and 3D electrode setups, assuming distributions of sources in a tissue, a slice, or in a single cell.

Introduction

Multisite recording of extracellular potential is a popular technique in neuroscience. The obtained potential reflects activity of underlying neural network and is directly related to the distribution of current sources along the active cells (current source density, CSD). The relation between the CSD and recorded potential, while occasionally contested [Bédard and Destexhe, 2011], overall is well established and trusted in experimental and analytical practice [Buzsáki et al., 2012, Einevoll et al., 2013, Gratiy et al., 2017]. Due to the long range of electric potential [Łęski et al., 2007, Hunt et al., 2011, Lindén et al., 2011, Łęski et al., 2013] it is useful to estimate the current sources. Several methods have been introduced to estimate current sources since 1950s [Pitts, 1952, Nicholson and Freeman, 1975, Pettersen et al., 2006, Łęski et al., 2007, 2011]. In 2012 we proposed a non-parametric method of CSD estimation which we called kernel CSD method (kCSD, [Potworowski et al., 2012]). Here we revisit this method with three aims in mind. First, to present a new Python based toolbox, which allows kCSD reconstruction of current sources for data from 1D setups (laminar probes and equivalent electrode distributions), 2D (planar MEA, multishaft silicon probes, Neuropixel or SiNAPS probes, etc), and 3D electrode setups (Utah arrays, multiple electrodes placed independently in space with controlled positions), where the sources are assumed to come from tissue (kCSD) or from single cells with known morphology (skCSD). Fig. 1 shows the different experimental scenarios for which this software is applicable. Second, using the new software, we investigate the properties of some mathematical structures arising in the context of practical experimental studies. Third, we introduce several new conceptual tools which will facilitate CSD estimation and knowledge extraction from the data.

Our plan is as follows. We first review the basic relations between the relevant physical quantities we study and we set up the computational framework. Then we study the central quantities involved in estimation of the sources from the measurements. We follow this with introduction of L-curve estimation of optimal reconstruction parameters and reliability maps for the study of reconstruction accuracy. Finally, we provide an overview of the new software package which was used for the presented analysis and illustrate the new concepts introduced here.

Current source density and local field potential

The extracellular potential that we measure is a consequence of ion motion in the tissue which is driven by ionic currents through the ion channels embedded in neuronal and glial membranes, as well as capacitive currents arising in response to potential gradients across the membrane. From the perspective of extracellular medium it seems as if the current was disappearing or appearing from inside a cell, which is why we talk about current sources and sinks. The distribution of these current sources is called the current source density (CSD) and its relation to the extracellular potential is given by the Poisson equation

$$C = -\nabla(\sigma \nabla V), \tag{1}$$

where C is the CSD, V is the extracellular potential, and σ — the conductivity tensor. Thus, if we knew the potential in the whole extracellular space, we could easily compute the CSD. On the other hand, knowing CSD in the whole space, we can compute the extracellular potential. Assuming isotropic and homogeneous tissue the Poisson equation reduces to

$$C = -\sigma \Delta V \tag{2}$$

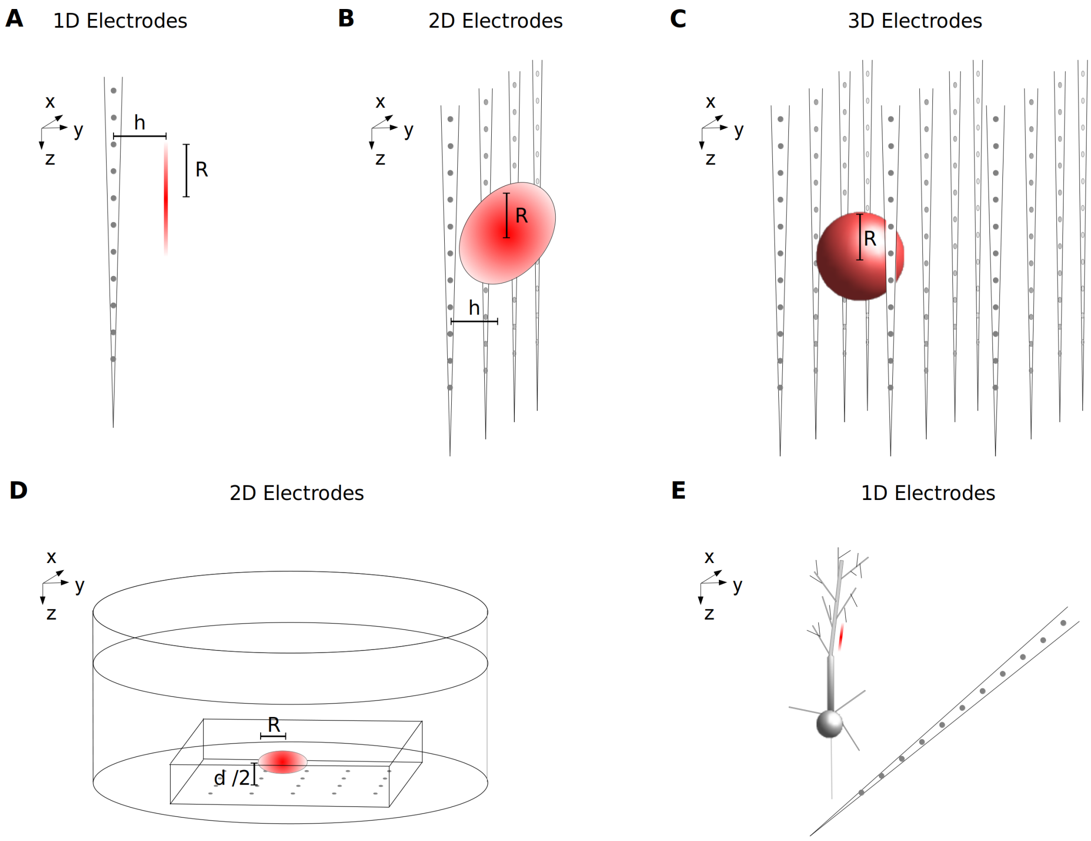

Figure 1: Overview of experimental contexts where `kCSD-python` is applicable. 1D setups such as A) laminar probes and equivalent; 2D setups, such as B) multishaft silicon probes, Neuropixel or SiNAPS probes, or D) planar MEA; 3D electrode setups, such as multiple multishaft silicon probes, Utah arrays, multiple electrodes placed independently in space with controlled positions, where the sources are assumed to come C) from tissue (kCSD) or E) from single cells with known morphology (skCSD). For description of parameters see Methods.

which can be easily solved:

$$V(\mathbf{x}) = \frac{1}{4\pi\sigma} \int d\mathbf{x}' \frac{C(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|}. \tag{3}$$

In more complex situations when σ depends on position and direction, and we have non-trivial boundary conditions, one must resort to numerical integration. This can be done Ness et al. [2015], Næss et al. [2017] but in the following we will assume constant scalar σ (homogeneous, isotropic medium). The reason is that more general models of tissue conductivity do not affect the following discussion and results but they would make presentation more cumbersome. In practice, they only affect the relation between the basis functions in the space of CSD and potential, $\tilde{b}_j$ and b_j in what follows. In the provided package this is one Python function that needs to be replaced.

Careful discussion of the meaning of the CSD and derivation of the relations between CSD and the potential can be found in Stevens [1966], Nicholson [1973], Gratiy et al. [2017]. Discussion of physiological sources of the extracellular potential can be found in the reviews by Buzsáki et al. [2012], Einevoll et al. [2013].

Kernel Current Source Density and prior estimation methods

Pitts [1952] observed that the Poisson equation relating the extracellular potential to the CSD can be used to estimate CSD from measurements. He used this observation to examine the distribution of currents and sinks in the spinal cord. His approach, which we call *traditional CSD method*, was based on direct numerical approximation to the second derivative in equation (2). Pitts' approach with small modifications was used for more than half a century and is still in use today. In 2006, Pettersen et al. [2006] proposed a model-based approach to estimate CSD. The *inverse CSD (iCSD) method* which they introduced for analysis of laminar recordings assumed a parametric model of CSD with the recordings used to compute model parameters. Some advantages of this approach were explicit inclusion of assumptions regarding physical properties of the tissue conductivity and the properties of sources in the directions not probed. Inverse CSD facilitates testing the influence of these uncontrolled factors on the estimated CSD. It was extended to 2D and 3D recordings [Łęski et al., 2007, 2011], however, it still required recordings on regular rectangular grids (although see Wójcik and Łęski [2010]) and did not compensate for the measurement noise. Potworowski et al. [2012] proposed how to overcome these limitations generalizing iCSD to the *kernel CSD (kCSD) method*. It was later studied from the general perspective of discrete inverse problems by Kropf and Shmuel [2016].

Kernel CSD estimation is conceptually a two-step procedure. First, we do a kernel interpolation of the measured potential which gives $V(x)$ in the whole space. This is obtained with the help of a symmetric kernel function, $K(\mathbf{x}, \mathbf{x}')$, so that

$$V^*(\mathbf{x}) = \sum_{j=1}^{N} \beta_j K(\mathbf{x}, \mathbf{x}_j),$$

where $\mathbf{x}_j$, $j = 1, \ldots, N$, are electrode positions. The regularized solution, which makes correction for noise, is obtained by minimizing prediction error

$$\mathrm{err}[V] = \sum_{i=1}^{N} (V(\mathbf{x}_i) - V_i)^2 + \lambda \|V(\mathbf{x})\|^2_{\mathcal{F}},$$

which gives

$$\beta = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{V},$$

where $\mathbf{V}$ is the vector of measured potentials, λ is regularization parameter, the norm $\|.\|^2_{\mathcal{F}}$ is discussed in the Methods section, and

$$\mathbf{K}_{i,j} \equiv K(\mathbf{x}_i, \mathbf{x}_j).$$

To simplify the notation and bring the discussion closer to our numerical implementation we shall consider estimation at an arbitrary discrete set of points $\mathbf{y}_i$, with $i = 1, \ldots, W$. Further, in matrix representations of operators we shall assume the variables taking the values of electrode positions, $\mathbf{x}_j$, $j = 1, \ldots, N$, or the values of estimation points, $\mathbf{y}_i$, $i = 1, \ldots, W$. In the latter case we shall underline the matrix. So a single underline means rows are numbered by estimation points, and columns are numbered by electrode positions. Double underline means that both rows and columns are numbered by estimation points. The variable which takes value at estimation points can usually be also read as a free variable. For example

$$\underline{\mathbf{K}} = \begin{bmatrix} K(\mathbf{y}_1, \mathbf{x}_1) & \ldots & K(\mathbf{y}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ K(\mathbf{y}_W, \mathbf{x}_1) & \ldots & K(\mathbf{y}_W, \mathbf{x}_N) \end{bmatrix},$$

or $\underline{\mathbf{K}} = [K(\mathbf{y}, \mathbf{x}_1), \ldots, K(\mathbf{y}, \mathbf{x}_N)]$. Usually these two viewpoints should be interchangeable, if not, specific use should be clear from the context. For example, when we discuss software implementation, only the first (discrete) view is applicable.

With this notation, our estimation of the potential in the whole space is

$$\underline{\mathbf{V}}^* = \underline{\mathbf{K}}(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{V}.$$

Once we estimate the potential we must shift the obtained solution to CSD space. This is easiest to understand in 3D where we can simply plug this solution into the Poisson equation (1), and compute CSD everywhere. In the general case this can be achieved with a second function, which we call cross-kernel, $\widetilde{\mathbf{K}}(\mathbf{x}, \mathbf{x}')$. With these functions the resulting CSD estimation is given by

$$\underline{\mathbf{C}}^* = \underline{\widetilde{\mathbf{K}}}(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{V}.$$

In practice, to identify relevant $\mathbf{K}$ and $\widetilde{\mathbf{K}}$, we introduce a large basis of CSD sources spanning the space of interest, $\tilde{b}_j(\mathbf{x})$, and corresponding basis in the potential space, $b_j(\mathbf{x})$, and construct our kernels from these basis functions [Potworowski et al., 2012]. We review the details of kCSD method in the Methods section.

The challenges of the method are how to construct $\mathbf{K}$ and $\widetilde{\mathbf{K}}$, how to select the relevant parameters, and reliability of the estimation. In the following we address all these issues conceptually and computationally with the provided open package, `kCSD-python`.

Results

In this paper we introduce the `kCSD-python` package, a novel implementation of the kernel Current Source Density method. It is distributed under an open license (3-Clause BSD License) and is available on GitHub (`https://github.com/Neuroinflab/kCSD-python`). The package contains a set of tools for kCSD analysis, for validation of the results of analysis, and extensive tutorials implemented in jupyter notebook to familiarize the user with its usage. It allows the user to analyze their own electrophysiological recordings or to explore the method with data generated *in silico*.

In this section we introduce several tools to facilitate conceptual and practical understanding of the reliability of CSD analysis. First, we discuss spectral decomposition of the kCSD and introduce the concept of eigensources. Then we study the effects of relative placement of the basis, the electrodes and sources on reconstruction fidelity. We show that when the basis is placed in the region containing the true sources, reconstruction can be performed even if the signal is picked away from the source. Next we discuss parameter selection focusing on the L-curve approach which we introduced in the package along the previous cross-validation approach. Finally, we introduce reliability maps as a heuristic tool to build intuition about the power of any given setup to resolve the CSD.

We close this section with an extensive tutorial overview of the `kCSD-python` package. Its goal is to show how to use this package to perform CSD analysis, how to apply the provided analysis tools, and to validate the results. We first consider a regular grid of ideal (noise-free) electrodes, where we compute the potentials from a known test source (the ground truth). We then use these potentials to reconstruct the sources which we compare with ground truth (Basic features). Then, we explore the effects of noise on the reconstruction and test the robustness of the method (Noisy electrodes). In the final part of the tutorial we look at how the errors in the estimation depend on the sources and the electrode configuration by testing the effects of broken electrodes on reconstruction (Broken electrodes).

Spectral decomposition for kCSD and regularization

Let us reconsider the construction of kCSD (see Methods). In kCSD we estimate CSD in space $\widetilde{\mathcal{F}}$ span by a large, M-dimensional basis, $\tilde{b}_i$. However, our experimental setup imposes constraints which force our model to an N-dimensional subspace where the estimation really takes place, with $N \ll M$. To understand the structure of this smaller space we can decompose the operator $\mathbf{K}$, eq. (25), acting on the measurements. We can take advantage of the symmetry and positivity of $\mathbf{K}$ matrix which guarantees existence of eigendecomposition

$$K = \sum_{j=1}^{N} \mu_j \mathbf{w}_j \mathbf{w}_j^T. \tag{4}$$

Then the kCSD reconstruction for a set of measurements $\mathbf{V}$, eq. (27), is

$$\begin{aligned} \underline{\mathbf{C}}^*[\mathbf{V}] &= \underline{\widetilde{\mathbf{K}}} \sum_{j=1}^{N} \frac{1}{\mu_j + \lambda} \mathbf{w}_j \mathbf{w}_j^T \mathbf{V} \\ &= \sum_{j=1}^{N} \frac{1}{\mu_j + \lambda} (\underline{\widetilde{\mathbf{K}}} \mathbf{w}_j)(\mathbf{w}_j^T \mathbf{V}). \end{aligned}$$

Since $\mathbf{w}_j$ are orthogonal we have

$$\underline{\mathbf{C}}^*[\mathbf{w}_j] = \frac{1}{\mu_j + \lambda} \underline{\widetilde{\mathbf{K}}} \mathbf{w}_j.$$

Thus $\mathbf{w}_j$ are the natural 'eigenmeasurements' corresponding to individual CSD profiles, $\underline{\widetilde{\mathbf{K}}} \mathbf{w}_j$, accessible to the given setup when specific basis $\tilde{b}_i$ is assumed. Moreover, it is easy to see that the CSD profiles

$$\underline{\mathbf{C}}_j = \underline{\widetilde{\mathbf{K}}} \mathbf{w}_j \tag{5}$$

actually form the basis of estimation space, we call them 'eigensources'. Since kCSD method is self-consistent, in the absence of noise, we see that the potential at the electrodes generated by $\underline{\mathbf{C}}_j$ is

$$\mathbf{V}[\underline{\mathbf{C}}_j] = \mathbf{K} \mathbf{w}_j = \mu_j \mathbf{w}_j.$$

It leads to reconstructed CSD

$$C^*[\mathbf{V}[\underline{\mathbf{C}}_j]] = \frac{\mu_j}{\mu_j + \lambda} \underline{\widetilde{\mathbf{K}}} \mathbf{w}_j \propto \underline{\mathbf{C}}_j$$

which is equal to $\underline{\mathbf{C}}_j$ for $\lambda = 0$.

A natural question appears as to what happens to the missing $M - N$ dimensions. The answer is that they are projected onto 0 (annihilated). It is possible to construct them explicitly. Starting with the basis of N eigensources we can expand it within $\widetilde{\mathcal{F}}$ by Gram-Schmidt orthogonalization. This construction breaks $\widetilde{\mathcal{F}}$ into two orthogonal subspaces, one of which spans all the sources which can be recovered with a given setup, the other contains all the sources which are annihilated.

Let us now investigate this decomposition in an example. Here we study spectral properties of the simplest 1D case with regularly distributed electrodes. We consider $M = 2^k, k = 0, 1, \ldots, 9$ basis sources distributed uniformly in $[0, 1]$ and the number of electrodes is $N = 12$. This means that $\mathbf{K}$ is 12×12 and there are 12 eigensources.

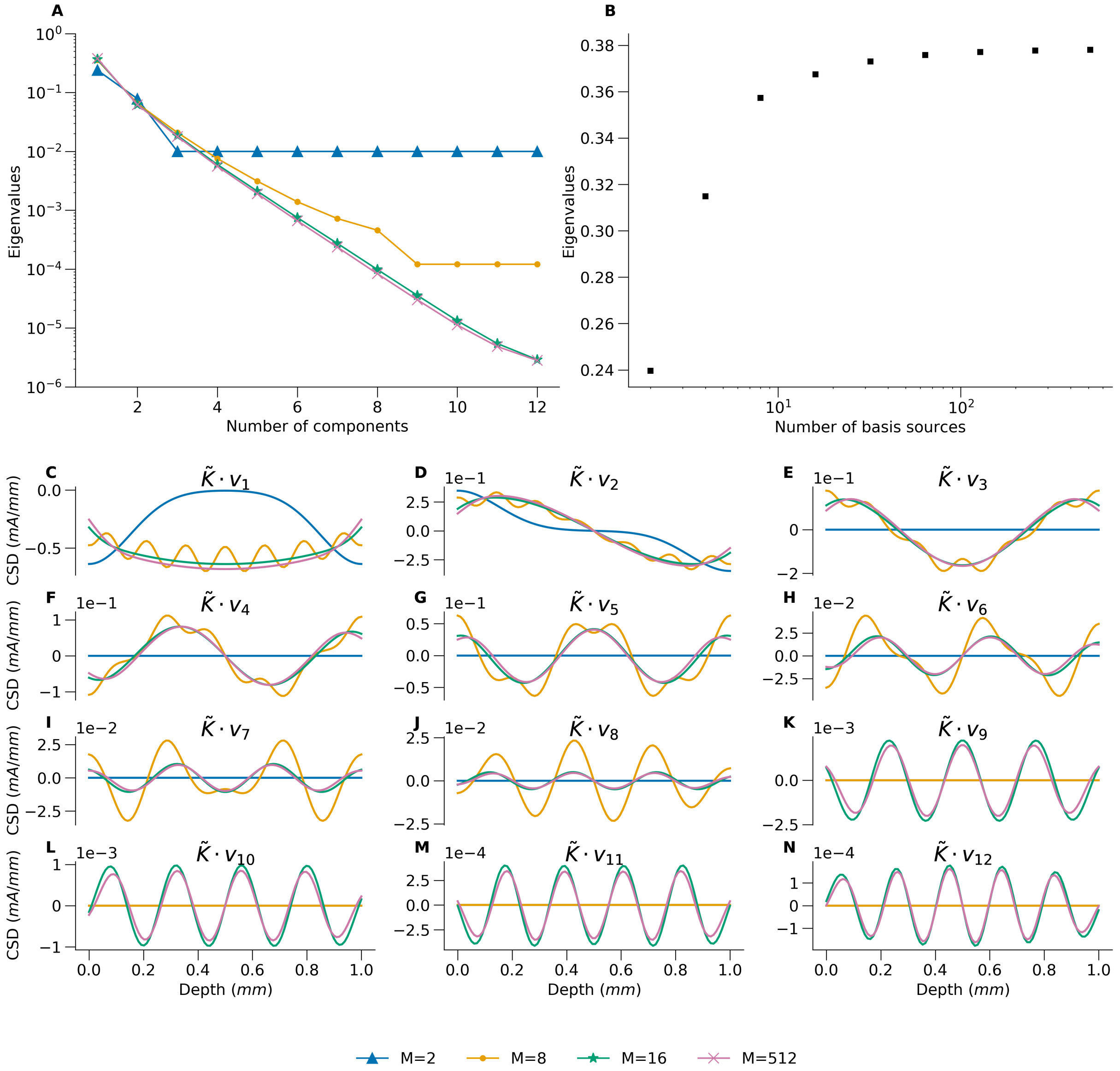

Figure 2: Demonstration of spectral properties of kCSD method for simple 1D case with 12 regularly distributed electrodes on the interval $[0, 1]$. A) Eigenvalues obtained from the decomposition of $\mathbf{K}$ for different number of basis sources (M=2, 8, 16, 512), B) Shows the first eigenvalue in the function of growing number of basis, C–N) Shows 12 eigensources (products of $\widetilde{\mathbf{K}}$ and all the eigenvectors), which produce the inverse solution in kCSD method. Different curves represent eigensources estimated for different M.

Fig. 2.A) shows the eigenspectrum of K for growing basis size, $M = 2, 8, 16, 512$. As we can see, the spectrum of $\mathbf{K}$ quickly stabilizes and already for $M = 16$ it is very close to that for $M = 512$, so it is almost asymptotic. Approach to asymptotic values is illustrated for the leading eigenvalue which is plotted as a function of M in Fig. 2.B). Fig. 2.C)–N) show all the 12 eigensources, $\underline{\mathbf{C}}_j = \widetilde{\underline{\mathbf{K}}}\mathbf{w}_j$, eq. 5, for the same basis. As we can see, in the degenerate cases of $M < N$ only M eigensources are non-trivial and we can see that they are approximations to asymptotic sources. Eigensources for $M = 16$ are very close to those for $M = 512$ so it seems to indicate that as long as the number of basis sources, M, is larger than the number of electrodes, N, for regular distribution of electrodes and basis sources covering the same region, the number of basis functions is not a limiting factor of the reconstruction. On the other hand, as we observed before, every reconstruction is a linear combination of the eigensources, thus the only things which can be reconstructed are linear combinations of the profiles shown in Fig. 2.C)–N).

Observe that for electrodes placed on a regular grid the set of eigensources forms a Fourier-like basis. This is in line with other results in discrete inverse problems Hansen [2010] although here we do not consider singular value decomposition of resolvent as is usually done (see e.g. Kropf and Shmuel [2016]). This brings to mind the sampling theorem Oppenheim et al. [1996], except here we measure one quantity (the potential) and reconstruct another (CSD). In the sampling theorem we reconstruct a continuous signal from discrete samples. This in particular shows that there would be a range of spatial frequencies within which reconstruction is efficient and beyond which it will increasingly fail. While this is a linear oversimplification it captures the intuition well. For an alternative view of this observation see Fig. 2 of Cserpan et al. [2017].

Note that we can perform similar studies for arbitrary setups. Once they are no longer regular, the basis of eigensources may deviate from Fourier arbitrarily. This can be easily studied with the `kCSD-python` package introduced here for any given setup used experimentally.

The effects of relative placement of basis, electrodes, and sources, on reconstruction fidelity

In CSD analysis it is usually assumed that the electrodes probe the region of interest well and that the sources to be reconstructed are within the space span by the electrodes. It has usually been our tacit assumption so far, in this and previous papers. However, this need not necessarily be so. First, in many experiments carried in the past and at present, the number of electrodes is small, we often probe complex extended structures with laminar probes, etc. Second, it is relatively common to record away from the source. For example, imagine *in vivo* study where we target a specific structure where activity is expected yet post-experimental histology shows the probe was misplaced. Can we learn anything of value about the sources of interest in such a case?

To address this question here we consider a 1D Gaussian source centered at 0.25, which is essentially nonzero on interval $[0, 0.5]$, where the interval $[0, 1]$ is a metaphor for the whole brain. We then consider nine cases with three different distributions of 12 electrodes on intervals $[0, 1]$ ('within the whole brain'), $[0, 0.5]$ (covering only the region where the source is non-zero), $[0.5, 1]$ (badly misplaced probe), and three different distributions of basis sources, spanning the intervals $[0, 1]$, $[0, 0.5]$, $[0.5, 1]$ (Fig. 3). The different placements of basis sources are a metaphor for the prior knowledge from expert insight of where we anticipate the sources. Here, the width of the basis source, R, was selected through cross-validation for the first case and used throughout, $\lambda = 0$ and no noise was assumed.

In Fig. 3, consider the first row, where the electrodes span 'the whole brain'. In that case, if we distribute the basis sources throughout the brain (left column, panel A) or through the region where the true sources were placed (middle column, panel B), the reconstruction is faithful. However, if we distribute the basis sources in the right half of 'the brain', away from the true sources (Right column, panel C), kCSD fails miserably trying to reconstruct the source from the measured potentials in the place where there is nothing. This is to be expected. Interestingly, when the electrodes are placed on the right half, completely outside the region containing the true sources, if we place the basis sources

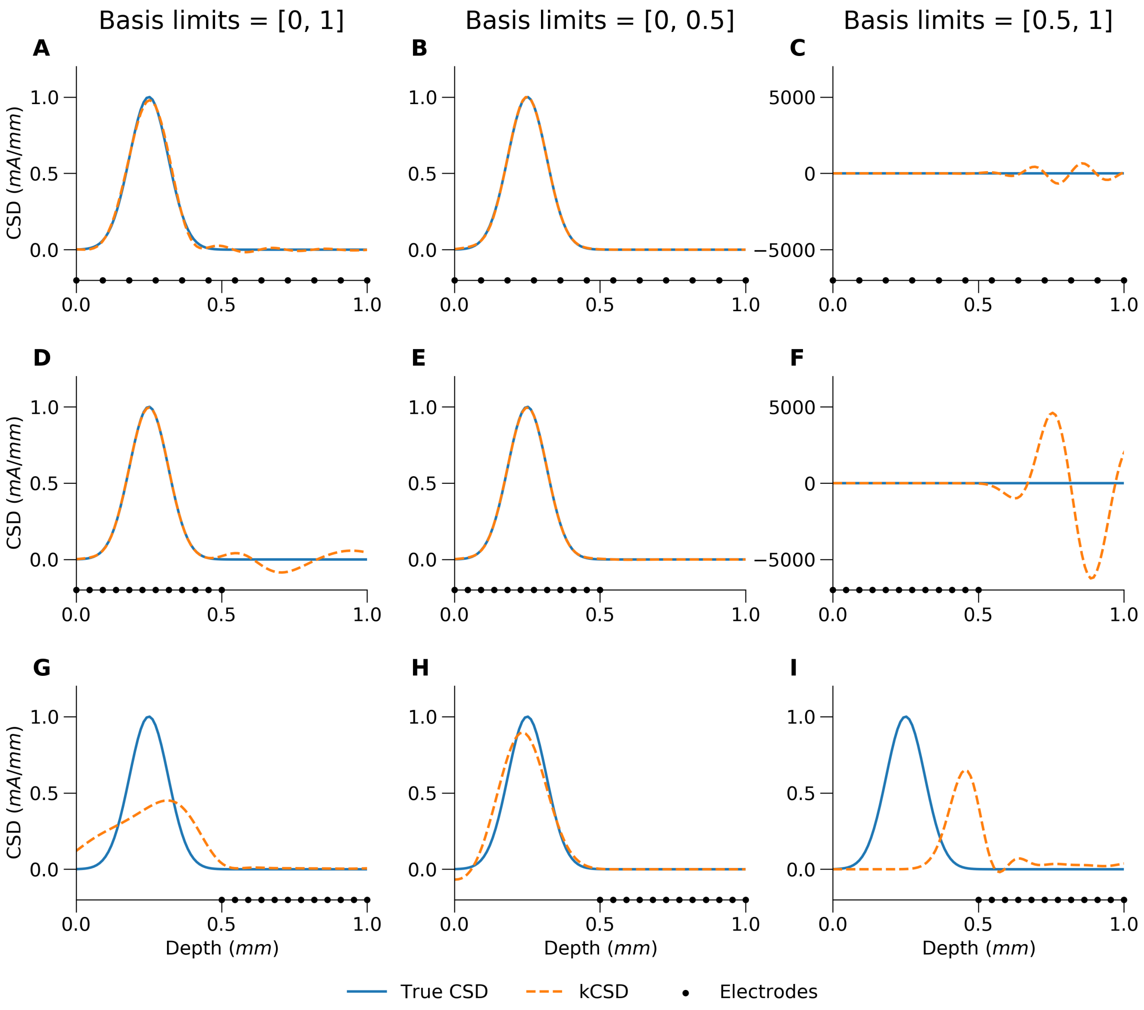

Figure 3: Demonstration of kCSD reconstruction for different relative placement of basis sources and electrodes for simple 1D Gaussian source in the absence of noise. Width of the basis source, R, was selected through cross-validation for the first case (A) and used throughout. Columns represent results for three different distributions of basis sources: left — centers of basis sources distributed uniformly on interval [0, 1], middle — interval [0, 0.5], right — interval [0.5, 1]. Rows represent different distribution of electrodes: top — electrodes span the interval [0, 1], middle — interval [0, 0.5], bottom — interval [0.5, 1]. Black dots represent electrode positions, continuous line — ground truth (TrueCSD), dashed line — CSD estimated with kCSD method.

so that they cover the region where the true sources are located the reconstruction, while misshapen, does indicate the location of sources (Panels G and H). Moreover, if we have expert knowledge which tells us to expect the source in the left half, and we accordingly place the basis sources only there, the reconstruction is actually remarkably faithful (Panel H). These results indicate that it is worthwhile to attempt reconstruction even in cases with misplaced electrodes, as long as we know relative distances between the electrodes and we have reasons to speculate the source being reconstructed to be in a restricted region.

Since noise-free data are not realistic it is interesting to investigate how noise affects these results. In a real life scenario we would always tune the parameters R, λ from data using cross-validation or L-curve method (next section). Fig. 4 shows reconstructions in the same nine cases as before for data which have been contaminated with additive noise. We show results obtained when optimal parameters where selected with cross-validation or with L-curve method in every case. As we can see, even though the results degrade when

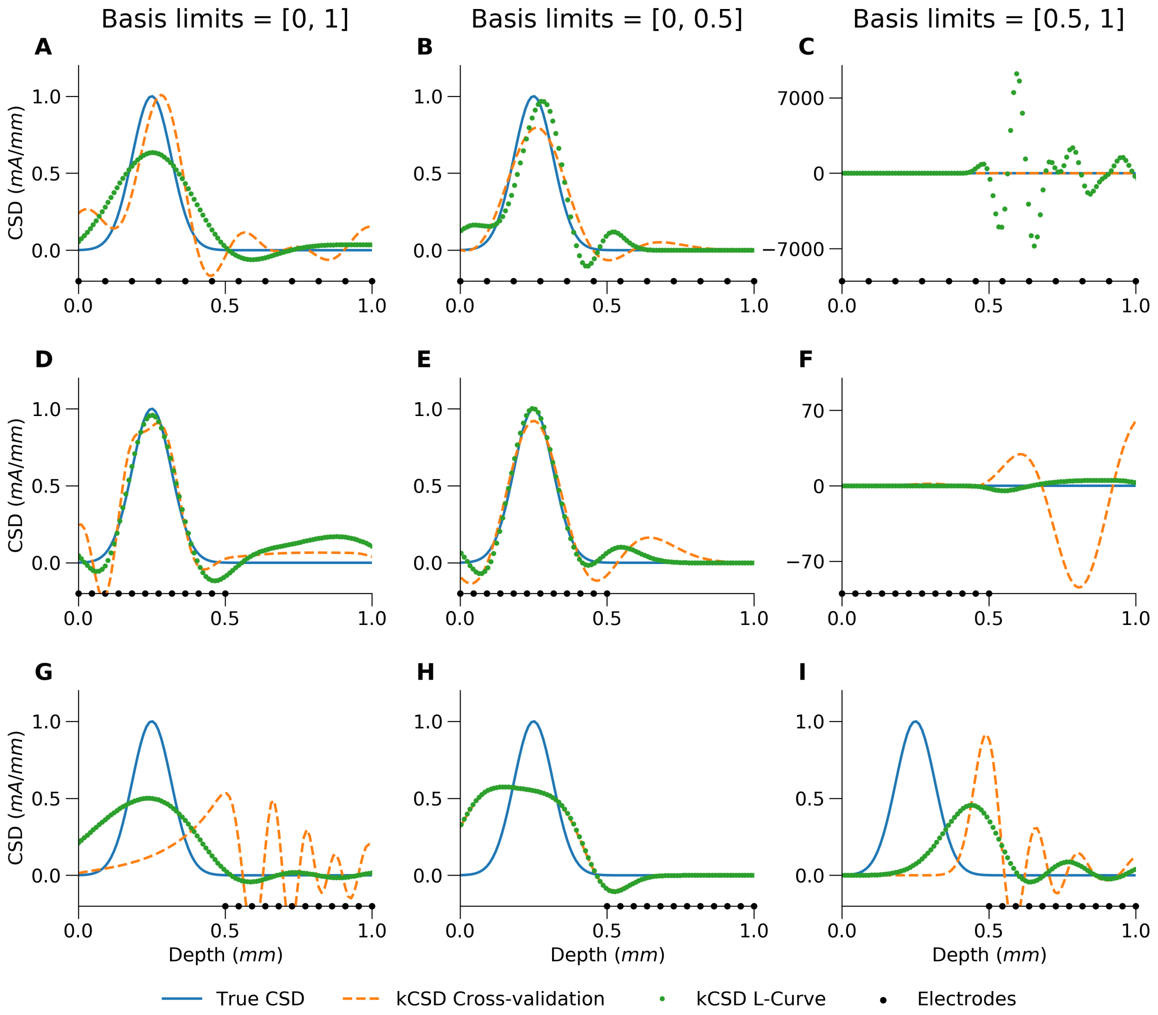

Figure 4: Demonstration of kCSD reconstruction for different relative placement of basis sources and electrodes for simple 1D Gaussian source. Here we assume measurement noise, thus cross-validation and L-curve methods were used to select regularization solution parameters R, λ. Columns and rows — see previous figure. Black dots represent electrode positions, continuous line — ground truth (TrueCSD), dashed line — CSD estimated with kCSD method, dotted line — L-curve estimation.

compared with the noise-free data, in all cases regularization improves the reconstruction, often providing useful information. Interestingly, in the case where we record outside the region of interest but where the basis spans 'the whole brain', L-curve gives a reasonable estimate while cross-validation fails. Note that this is accidental. Our tests show that in difficult cases one or the other approach to parameter selection may give better results.

Parameter selection

An important part of kCSD estimation is selection of parameters, in particular the regularization parameter, λ, but also the width of the basis source, R. Previously we proposed to use cross-validation. Here we also apply L-curve approach [Hansen, 2010, Kropf and Shmuel, 2016] for regularization. Both these methods are implemented in the provided Python package.

Cross-validation To select parameters using cross-validation [Potworowski et al., 2012] we consider a range of parameter values, $\lambda \in [\lambda_0, \lambda_1]$. For any test value λ we select an electrode $i = 1, \ldots, N$ and ignore it. With eq. (24) we build a model from remaining measurements, $V_\lambda^{/i}(\mathbf{x})$, and use it to predict the value at the ignored electrode, $V_\lambda^{/i}(\mathbf{x}_i)$. Here

$$V_\lambda^{/i}(\mathbf{x}) = \sum_{j \neq i} \beta_j^{/i,\lambda} K(\mathbf{x}, \mathbf{x}_j),$$

where the minimizing vector

$$\beta^{/i,\lambda} = (\mathbf{K}^{/i} + \lambda \mathbf{I}^{/i})^{-1} \mathbf{V}^{/i},$$

and where $^{/i}$ means i-th column and row are removed from the given matrix.

We repeat this for all the electrodes $i = 1, \ldots, N$ and compare predictions from the remaining electrodes against actual measurements:

$$\text{prediction error}(\lambda) = \sqrt{\sum_{i=1}^{N} (V_\lambda^{/i}(\mathbf{x}_i) - V_i)^2}. \tag{6}$$

For final analysis, λ giving minimum prediction error is selected. It is worth checking if the global minimum is also a local minimum. If the λ selected is one of the limiting values this may indicate that extending the range of λ might result is more optimal result or that the problem is ill-conditioned, for example too noisy, etc, and we are either underfitting or overfitting, as we discuss below for the L-curve.

L-curve Consider the error function, Eq. (23), which we minimize to get the regularized solution, $V_\lambda = \underline{\mathbf{K}}\beta_\lambda$. It is a sum of two terms we are simultaneously minimizing, prediction error

$$\varrho_\lambda = \sum_{i=1}^{N} (V_\lambda(\mathbf{x}_i) - V_i)^2, \tag{7}$$

and the norm of the model

$$\eta_\lambda = \|V_\lambda(\mathbf{x})\|^2_{\mathcal{F}} = |\beta_\lambda^T \mathbf{K} \beta_\lambda|, \tag{8}$$

weighted with λ. Taking $\lambda = 0$ is equivalent to assuming noise-free data. In this case we are fitting the model to the data, in practice, overfitting. On the other hand, taking large λ means assuming very noisy data, in practice ignoring measurements, which results in a flat underfitted solution. Between these extremes there is usually a solution such that if we decrease λ, the prediction error, ϱ, slightly decreases, while the norm of the model, η, increases fast, or the opposite, Fig. 5D.

This is apparent when the prediction error and the norm of the model are plotted in the log-log scale. This curve follows the shape of the letter L, hence the name L-curve [Hansen, 1992]. Several methods have been proposed to measure the curvature of the L-curve and to identify optimal parameters [Hansen et al., 2007]. In `kCSD-python`, we have implemented the triangle area method proposed by Castellanos et al. [2002].

To illustrate this method in the context of CSD reconstructions, we study an example of 1D dipolar current source with a split negative pole (See Fig. 5C, True CSD, red dashed line). We compute the potential at 32 electrodes (Fig. 5A, C, black dots) with additive noise at every electrode. Notice that if we reliedwanted to interpret the recorded potential directly (Fig. 5A, red dots) it is difficult to discern the split negative pole. Fig. 5D shows the estimated curvature for our example as a function of λ. The optimal value of λ is found by maximizing the curvature of the log-log plot of η versus ϱ, Fig. 5B. The red dot in Fig. 5B, D, indicates the ideal λ parameter for this setup obtained through the L-curve method.

Several methods have been proposed to measure the curvature of the L-curve and identify optimal parameters [Hansen et al., 2007]. We adopted the triangle area method proposed by Castellanos et al. [2002]. To distinguish between convex and concave plot, clockwise directed triangle area is measured as negative. Fig. 5D shows this estimated curvature for our example as a function of λ.

Fig. 6 compares results of parameter tuning and reconstruction by cross-validation and L-curve method. The top panel shows the estimation error between the reconstruction and the ground truth (known model data) for the two approaches as a function of increasing measurement noise simulated. The bottom panel shows which λ is being selected by both methods as the noise is increasing. The mean and the errors were obtained for 10 different realizations of noise. As we can see, the two methods give consistent results for low noise. For increasing noise, in this case, the L-curve tends to indicate lower λ values which here gives a slightly bigger error. However, it is computationally faster.

Reconstruction accuracy

With kCSD procedure one can easily estimate optimal CSD consistent with the obtained data. However, so far we have not discussed estimation of errors on the reconstruction. Since the errors may be due to a number of factors — the procedure itself, measurement noise, incorrect assumptions — one should consider several approaches to this challenge.

First, to understand the effects of the selected basis sources and setup, one may consider the estimation operator $\underline{\widetilde{\mathbf{K}}}(\mathbf{K} + \lambda)^{-1}$ and the space of solutions it spans. As we discussed above, this space is given by the eigensources, eq. (5). The orthogonal complement of this space in the original estimation space, $\tilde{\mathcal{F}}$, is not accessible to the kCSD method. The study of eigensources facilitates understanding which CSD features can be reconstructed and which are inaccessible.

Second, to consider the impact of the measurement noise on the reconstruction, for any specific recording consider the following model-based procedure. Reconstruct CSD from data with optimal parameters. Compute potential from estimated CSD. Add random noise to each computed potential. The noise could be estimated from data, either as a measure of fluctuations on a given electrode for a running signal, or from variability of evoked potentials. Then, for any realization of noise, compute estimation of CSD. The pool of estimated CSD gives estimation of the error at any given point where estimation is made.

This computation can be much simplified by taking advantage of the linearity of the resolvent, $\mathbf{E} = \underline{\widetilde{\mathbf{K}}}(\mathbf{K}+\lambda\mathbf{I})^{-1}$. Then, the i-th column ($\mathbf{E}_i$) represents contribution of unitary change of i-th measured potential (the i-th element of the vector $\mathbf{V}$) to the estimated CSD ($\underline{\mathbf{C}}^*$). As the contribution is proportional to the change, the column can be considered an *Error Propagation Map* for i-th measurement (Fig. 7.A). Note that these vectors (the

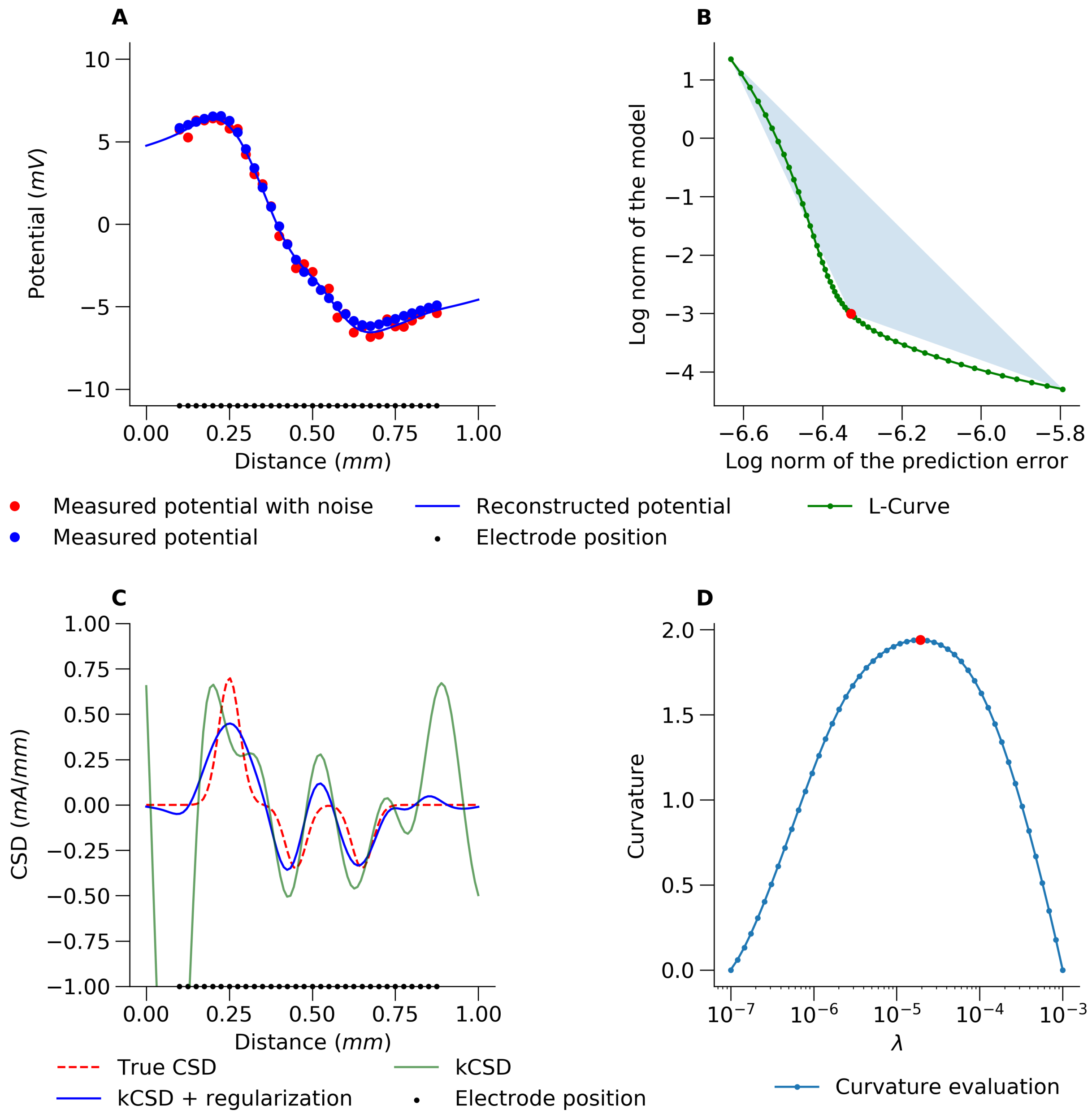

Figure 5: An example of the L-curve method for estimating kCSD parameters. A) The red points represent the potential used for CSD reconstruction. The black points show the electrode positions. The ground truth is shown in panel C with the red dashed curve. The measurement was simulated by adding small random noise to all the electrodes (32 values taken from a uniform distribution). The blue line shows a kernel interpolation of the potential which is the first step of kCSD method. B) L-curve plot for a single R parameter. The apex of the L-curve is numerically computed from the oriented area of directed triangles connecting the point on the L-curve with its two ends. C) Comparison of the true CSD and kCSD reconstruction for parameters obtained with L-curve regularization. D) Etimation of L-curve curvature with triangle method (see the Methods).

columns of resolvent, $\mathbf{E}_i$) also happen to form another basis of the solution space, an alternative to the basis of eigensources.

If ε_i is an error of i-th measurement, then its contribution to $\underline{\mathbf{C}}^*$ is $\varepsilon_i \mathbf{E}_i$. Moreover, if

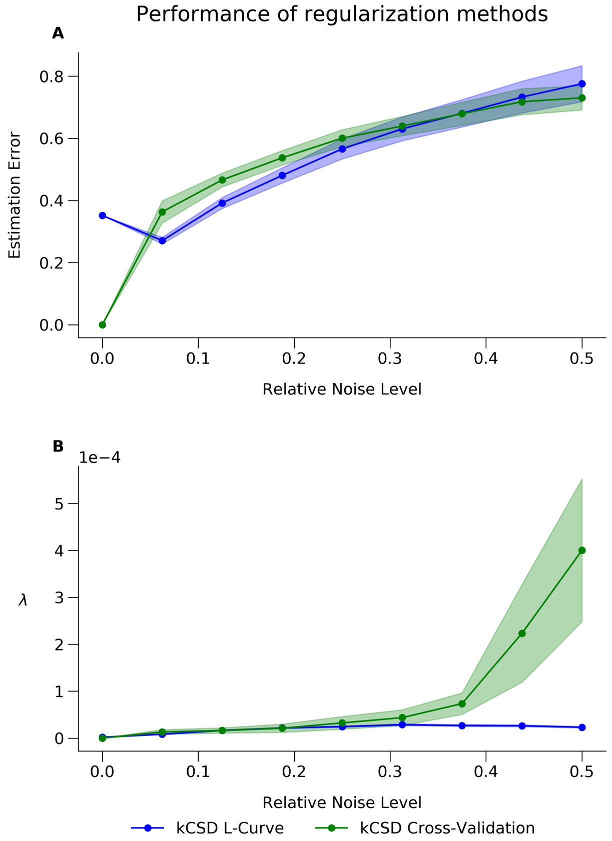

Figure 6: Comparison of parameter selection using cross-validation and L-curve method with the addition of noise A)The estimation error value is calculated as root mean square error between normalized true CSD and estimated CSD (normalized to maximum value) obtained either by using L-curve (blue) or by using cross-validation (green) B) The corresponding ideal λ parameter selected. For both these of methods λ increases with the added noise. Blue and green ribbons indicate standard deviation of the RMS around mean λ computed from 10 different noise realizations.

the measurement errors follow multivariate normal 321

$$\varepsilon \sim \mathcal{N}(0, \Sigma_V),$$

then 322

$$\mathbf{V} \sim \mathcal{N}(\mathbf{V}_{exact}, \Sigma_V),$$

and the estimated CSD also follows multivariate normal 323

$$\underline{\mathbf{C}}^* \sim \mathcal{N}(\mathbf{E}\mathbf{V}_{exact}, \mathbf{E}\Sigma_V\mathbf{E}^T).$$

The diagonal of $\mathbf{E}\Sigma_V\mathbf{E}^T$ represents a *map of CSD measurement uncertainty* (uncertainty 324
attributed to the noise in the measurement, Fig. 7.B[1]. 325

Third, one can study reconstruction accuracy for a meaningful family of test functions. 326
This could be the Fourier modes for rectangular regions or a collection of Gaussian test 327
functions, centered in different places, of single or multiple radii. For each of these test 328

[1] In the special case when ε_i are mutually independent and of equal variance σ^2, the map of CSD measurement uncertainty can be calculated as a diagonal of $\mathrm{Var}[\underline{\mathbf{C}}^*] = \mathbf{E}\mathbf{E}^T\sigma^2$.

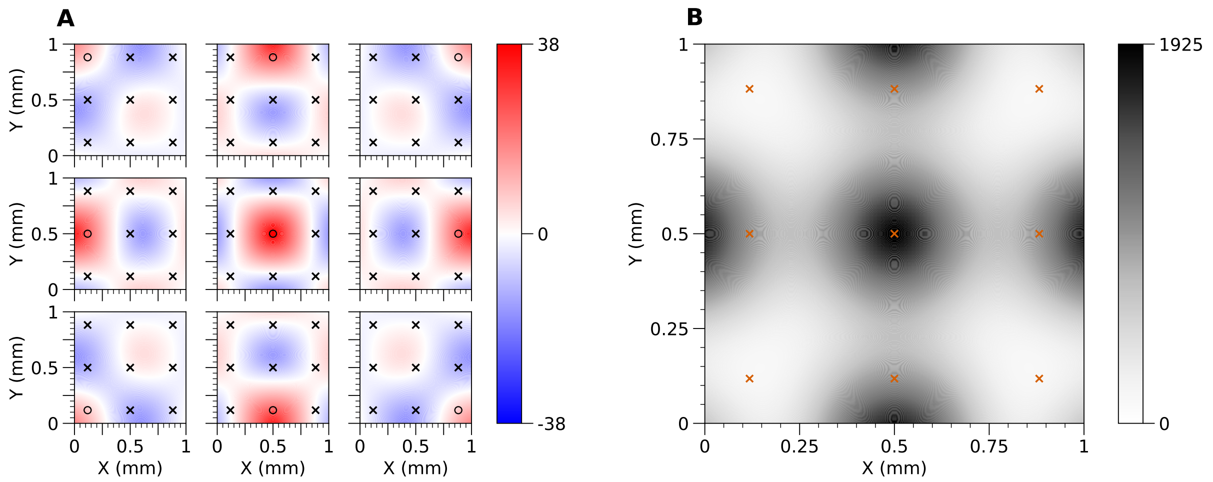

Figure 7: A) Error propagation maps for 3×3 regular grid of electrodes. Every panel represents the contribution of the potential measured at the corresponding electrode marked with a black circle ($\circ$) to the reconstructed CSD. Every other electrode is marked with a black cross ($\times$). B) Map of CSD measurement uncertainty for 3×3 regular grid of electrodes. The CSD measurement uncertainty is represented by variance of the CSD reconstruction caused by the uncertainty in measurement of the potentials. It is assumed that measurement errors for electrodes are mutually independent and follow standard normal distribution ($\varepsilon_i \sim \mathcal{N}(0, 1)$). Location of electrodes is marked with red crosses ($\times$).

functions one would compute the potential, perform reconstruction, and compare the results with the original at every point. Finally, one could average this information over multiple different test sources computing a single *Reliability Map*, which we now introduce.

Reliability maps Assume the standard kCSD setup, that is a region $\mathcal{R} \subset \mathbb{R}^n$ where we want to estimate the sources, set of electrode positions, $\mathbf{x}_i$, and perhaps additional information, such as morphology for skCSD Cserpan et al. [2017]. We now want to characterize predictive power of the combination of our setup and our selected basis, $\tilde{b}_i$. To do this we select a family of test functions, $C^i(\mathbf{x})$, for example Gaussian test functions, centered in different places, of multiple radii, or products of Fourier modes, etc. Then, for each C^i we compute $V^i = \mathcal{A}C^i$ by forward modeling, generating a surrogate dataset. Next, we apply the standard kCSD reconstruction procedure obtaining estimation of the tested ground truth, $\tilde{C}^i$. We can then compute reconstruction error using point-wise modification of Relative Difference Measure (RDM) proposed by [Meijs et al., 1988]:

$$err(\mathbf{x}) = \left| \frac{\tilde{C}^i(\mathbf{x})}{\|\tilde{C}^i\|} - \frac{C^i(\mathbf{x})}{\|C^i\|} \right| * \frac{\|C^i\|}{\max_{\mathbf{x}\in\mathcal{R}} |C^i|}, \tag{9}$$

where $i = 1, 2, \ldots$ enumerates different ground truth profiles. A simple measure of reconstruction accuracy is then given by the average over these profiles:

$$Reliability(\mathbf{x}) := \langle err_i(\mathbf{x}) \rangle_{i\in[1,M]} = \frac{\sum_{i=1}^{M} err_i(\mathbf{x})}{M}. \tag{10}$$

Fig. 8 shows example reliability map for the case of 10x10 electrode distribution. The class of functions used were the families of small and large sources (see Methods). We used eight mirror symmetries of the grid in computation.

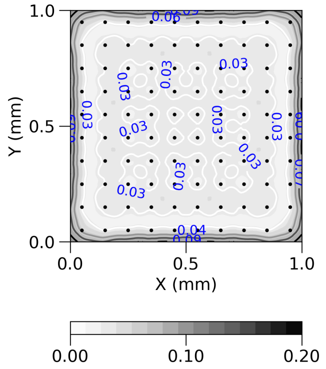

Figure 8: Reliability map created according to formula (9) and (10) for 10x10 regular grid of electrodes with noise-free symmetrized data. Black dots represent locations of contacts used in the study. Values on the map can be interpreted as follows: the closer to 0, the higher reconstruction accuracy might be achieved for a given measurement condition.

We can use reliability map as another source of information about the precision of reconstruction, which is shown in Fig. 9. In A) we show some dipolar source which is used

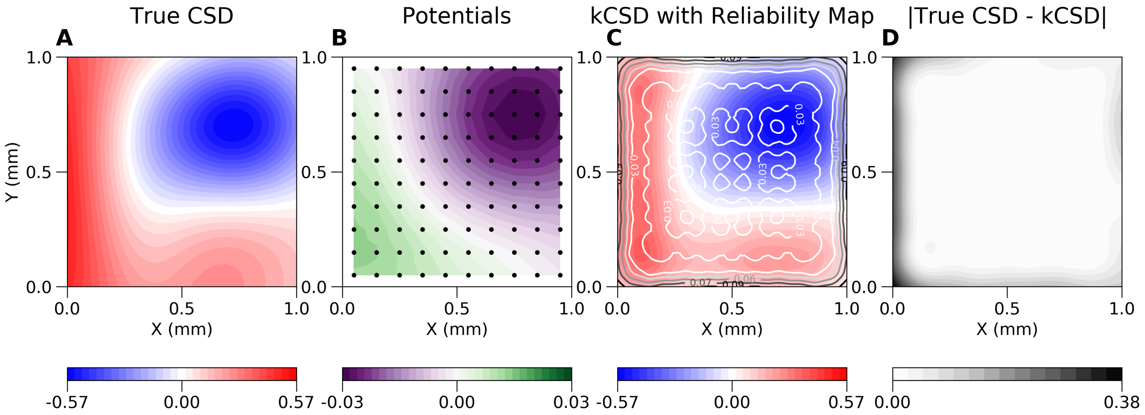

Figure 9: Example use of reliability maps. A) Example dipolar source (ground truth) which is used to compute the potential on a grid of electrodes shown in B). C) shows reconstructed sources superimposed on reliability map. D) shows the difference between the ground truth and the reconstruction.

to compute the potential on a grid of electrodes shown in B). Fig. 9.C) shows reconstructed sources superimposed on reliability map. Panel D) shows the difference between the ground truth and reconstruction. Note that plots such as shown in panel A) and D) are feasible only for simulated or model data, where we know actual sources and use them to validate the method. On the other hand, plots shown in panel B and C represent what can be routinely computed for experimental data.

Another interesting question is the effect of broken or missing electrodes on the reconstruction. Formally one can attempt kCSD reconstruction from a single signal but it is naive to expect much insight this way. It is thus natural to ask what information can be obtained from a given setup and what we lose when part of it becomes inaccessible.

Fig. 10 shows the effect of removing electrodes on the reconstruction. Fig. 10.A shows average error of kCSD method across many random ground truth sources for a regular grid of 10x10 electrodes. Fig. 10.B to D show the increase of average reconstruction error as we remove 5 (B), 10 (C) and 20 (D) contacts. To emphasize the errors we show the difference between the reliability map for the broken grid minus the original one. Note the different scales in plots B–D versus A. The consecutive rows show similar results when only small sources were used (E–H), or only large sources were used (I–L). Random sources in Fig. 10.A are both small and large sources (see Methods). This shows, among others, as we explained, that the reliability maps depend on the test function space, however, we feel they are more inuitive to understand than the individual eigensources spanning the solution space. 362 363 364 365 366 367 368

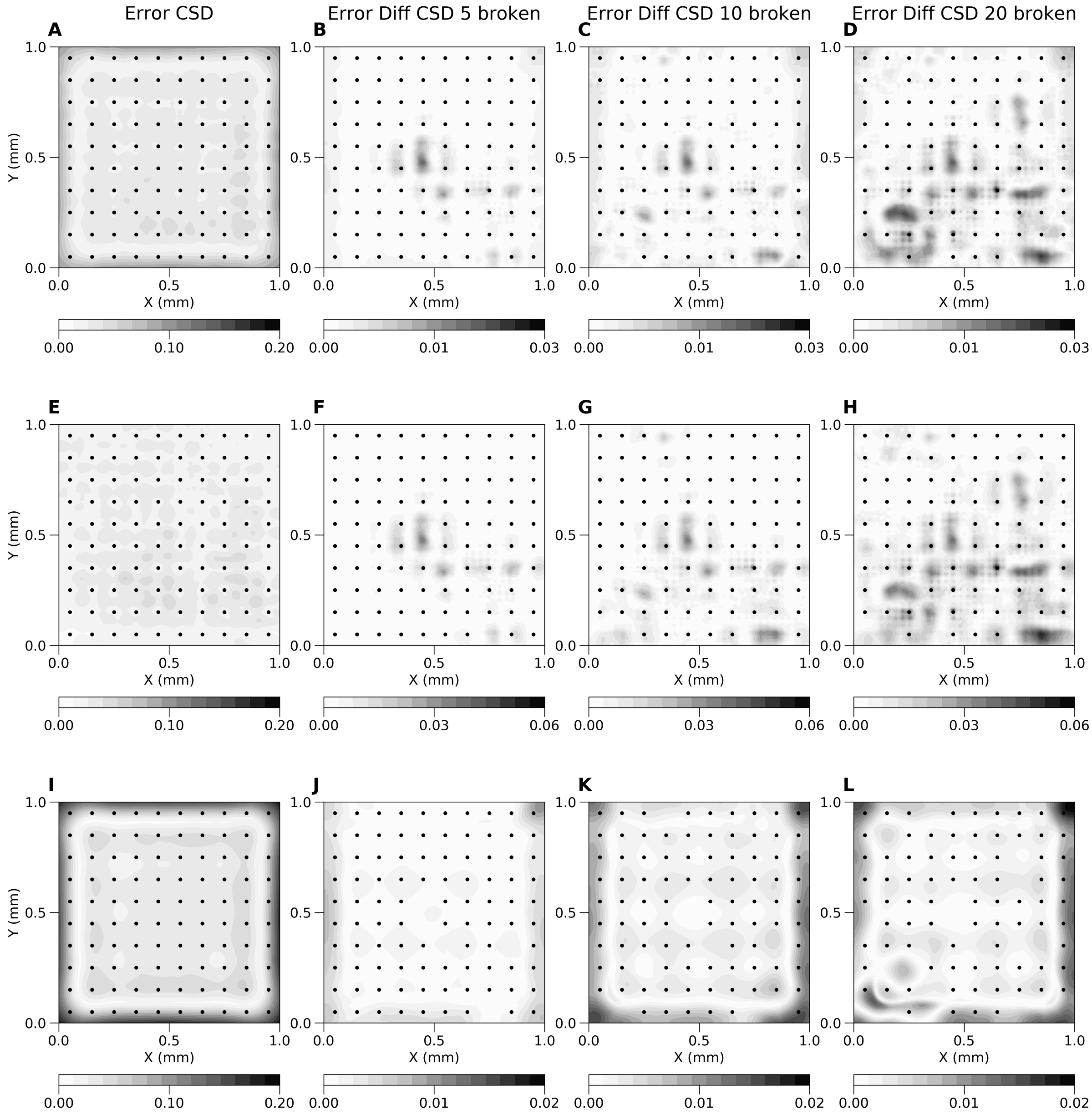

Figure 10: Average error (eq. 9) of kCSD method across random small and large (A), only small (E) and only large (I) sources for regular 10x10 electrodes grid and the same grid with broken 5 (B, F, J), 10 (C, G, K) and 20 (D, H, L) contacts. Plots (B, C, D, F, G, H, J, K, L) show difference between average error for regular grid and grid with broken contacts. Estimation was made in noise free scenario, R parameter selected in cross-validation. Black dots represent locations of contacts used in the study.

kCSD-python package tutorial

In this section we first illustrate the use of kCSD package for CSD reconstruction in the simplest case of a regular 2D square grid. This is a simplified version of a slice on a microelectrode array [Ness et al., 2015], or a planar silicone probe within the brain, where we assume constant conductivity in the whole space. In the following sections we show how we validate our methods and what kind of diagnostics we find useful in the analysis of experimental data. This tutorial is available as a jupyter notebook and can also be accessed through a web-browser without installation. For more details, see 'Overview of kCSD-python package' in the Discussion.

Basic features

We start with the basic CSD estimation on a regular grid. First, we define a region of interest. Then, using predefined test functions for the current sources, we place a ground truth current source in this region. We define the distribution of electrodes. Assuming ideal electrodes, we compute the potential generated by the selected current sources as measured at the electrodes. Given these potentials and the electrode locations we estimate the current source density using kCSD. As a final step, we perform cross-validation to avoid overfitting. Since we know the ground truth used to generate the potentials that were used in the kCSD estimation, we can compare the ground truth to the estimate and see the reconstruction accuracy.

Defining region of interest

```
In [1]: %matplotlib inline
        import numpy as np
        csd_at = np.mgrid[0.:1.:100j,
                          0.:1.:100j]
        csd_x, csd_y = csd_at
```

We define the region of interest between `0` and `1` in the `xy` plane with a resolution of `100` points in each dimension. We will assume the distance is given in mm, so we want to perform a reconstruction on a square patch of $1mm^2$ size.

Setting up the ground truth The `kCSD-python` library provides functions to generate test sources which can be imported from the `csd_profile` module. Here we use the `gauss_2d_small` function to generate two-dimensional Gaussian sources which are small in the scale set by the interelectrode distance. The other implemented option for two-dimensional test sources is the `gauss_2d_large` function. To generate the exact same sources in each run we must invoke this function using the same random seed which is stored in the `seed` variable. For simplicity, these current sources are static and do not change with time. We visualize the current sources as a heatmap.

```
In [2]: from kcsd import csd_profile as CSD
        CSD_PROFILE = CSD.gauss_2d_small
        true_csd = CSD_PROFILE(csd_at, seed=15)
```

The code below displays this test source as the True CSD. For convenience we define this as a function `make_plot`. The output for this code is shown in Fig. 11A.

```
In [3]: import matplotlib.pyplot as plt
        import matplotlib.cm as cm

        def make_plot(xx, yy, zz, title=’True CSD’, cmap=cm.bwr):
            fig = plt.figure(figsize=(7, 7))
            ax = plt.subplot(111)
            ax.set_aspect(’equal’)
            t_max = np.max(np.abs(zz))
            levels = np.linspace(-1 * t_max, t_max, 32)
            im = ax.contourf(xx, yy, zz, levels=levels, cmap=cmap)
            ax.set_xlabel(’X (mm)’)
            ax.set_ylabel(’Y (mm)’)
            ax.set_title(title)
            ticks = np.linspace(-1 * t_max, t_max, 3, endpoint=True)
            plt.colorbar(im, orientation=’horizontal’, format=’%.2f’, ticks=ticks)
            return ax
        make_plot(csd_x, csd_y, true_csd, title=’True CSD’, cmap=cm.bwr)

Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8d08dd74a8>
```

Place electrodes We now define the virtual electrodes within the region of interest. We place them between `0.05` *mm* and `0.95` *mm* of the region of interest, with a resolution of `10` (as indicated by `10j` in `mgrid`) in each dimensions, totalling to `100` electrodes. Notice that the electrodes do not span the entire region of interest. Although in this example the electrodes are distributed on a regular grid, this is not required by the kCSD method as it can handle arbitrary distributions of electrodes.

```
In [4]: ele_x, ele_y = np.mgrid[0.05: 0.95: 10j,
                                0.05: 0.95: 10j]
        ele_pos = np.vstack((ele_x.flatten(), ele_y.flatten())).T
```

Compute potential To obtain the potential, `pots`, at the given electrode positions due to the current sources that were placed in the previous steps we use the function `forward_method`. We assume the sources are localized within a slab of tissue of thickness 2h on top the MEA (See Łęski et al. [2011], Ness et al. [2015] and Methods). We also assume infinite homogeneous medium of conductivity `sigma` equal to 1 S/m. Finally, we assume that the electrodes are ideal, point-size and noise-free.

```
In [5]: from scipy.integrate import simps

        def integrate_2d(csd_at, true_csd, ele_pos, h, csd_lims):
            csd_x, csd_y = csd_at
            xlin = csd_lims[0]
            ylin = csd_lims[1]
            Ny = ylin.shape[0]
```

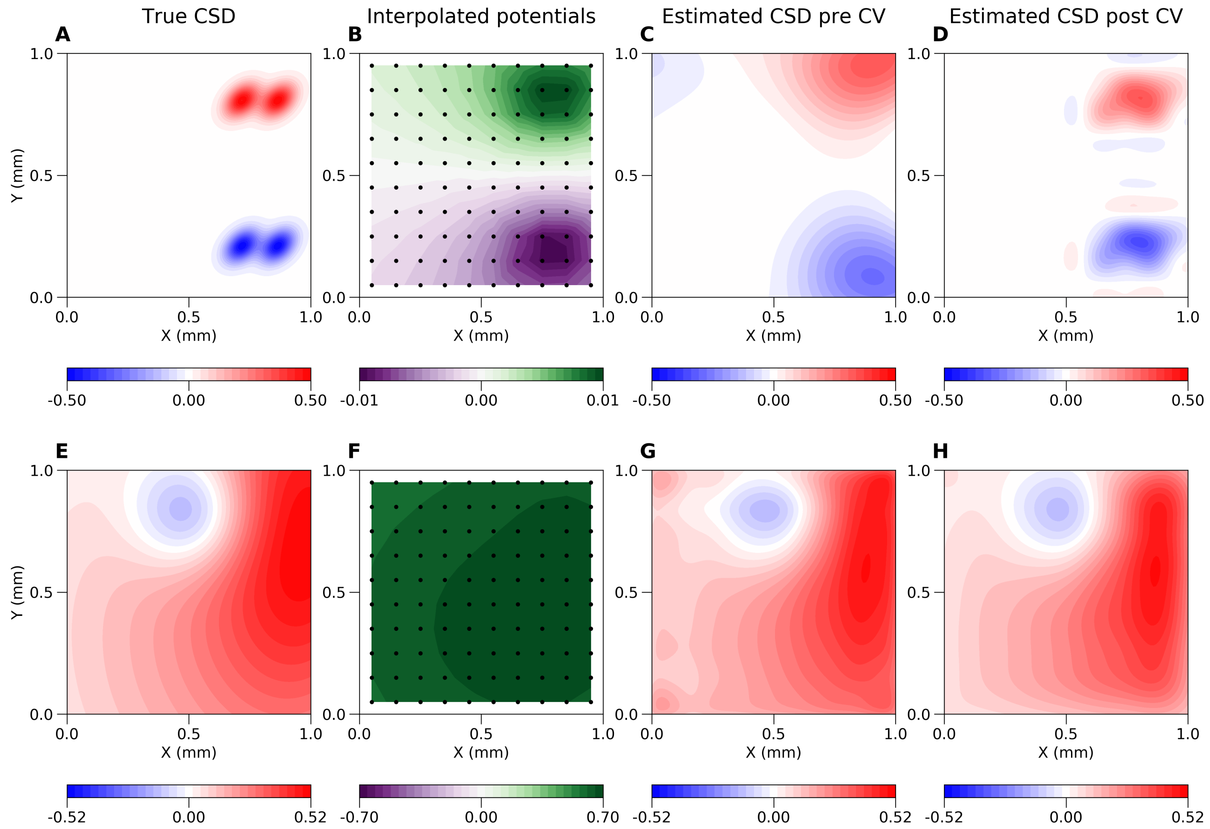

Figure 11: Basic features tutorial A) Shows the ground truth (True CSD), here, two-dimensional small Gaussian current sources, for the CSD seed of 15. B) The interpolated potentials generated by this current source are shown, the electrodes are displayed as black dots. C) CSD estimated with kCSD using the potentials from the electrode positions, without cross-validation. D) Same as C but cross-validation was used. E-H) Analogous to A–D, except large Gaussian current sources for seed 6 were used.

```
    m = np.sqrt((ele_pos[0] - csd_x)**2 + (ele_pos[1] - csd_y)**2)
    m[m < 0.0000001] = 0.0000001
    y = np.arcsinh(2 * h / m) * true_csd
    integral_1D = np.zeros(Ny)
    for i in range(Ny):
        integral_1D[i] = simps(y[:, i], ylin)
    integral = simps(integral_1D, xlin)
    return integral

def forward_method(ele_pos, csd_at, true_csd):
    pots = np.zeros(ele_pos.shape[0])
    xlin = csd_at[0, :, 0]
    ylin = csd_at[1, 0, :]
    h = 50.  # distance between the electrode plane and midslice
    sigma = 1.0 # S/m
    for ii in range(ele_pos.shape[0]):
        pots[ii] = integrate_2d(csd_at, true_csd,
                                [ele_pos[ii][0], ele_pos[ii][1]], h,
                                [xlin, ylin])
    pots /= 2 * np.pi * sigma
    return pots

pots = forward_method(ele_pos, csd_at, true_csd)
```

To visualize the potential, we interpolate the hundred values computed at the electrodes positions with `interpolate.griddata` function. Note that the kCSD estimation uses only the potential recorded at the electrode positions. To distinguish between the potentials and CSD plots we use different colormaps. The electrodes are marked with dots in this plot. The output from this step is shown in Fig. 11B.

```
In [6]: from scipy.interpolate import griddata

        def grid(x, y, z):
            x = x.flatten()
            y = y.flatten()
            z = z.flatten()
            xi, yi = np.mgrid[min(x):max(x):np.complex(0, 100),
                              min(y):max(y):np.complex(0, 100)]
            zi = griddata((x, y), z, (xi, yi), method='linear')
            return xi, yi, zi

        pot_X, pot_Y, pot_Z = grid(ele_pos[:, 0], ele_pos[:, 1], pots)
        ax = make_plot(pot_X, pot_Y, pot_Z, title='Interpolated potentials',
                       cmap=cm.PRGn)
        ax.scatter(ele_pos[:, 0], ele_pos[:, 1], 10, c='k')

Out[6]: <matplotlib.collections.PathCollection at 0x7f8d05851198>
```

kCSD method Here we illustrate the most basic estimation of CSD with the `kcsd` library. Since our example is two dimensional the relevant method is `KCSD2D`. For convenience we encapsulate the actual method call with parameters being set inside a function `do_kcsd`. We first set `h` and `sigma` parameters of the forward model. Then we restrict the potentials to the first time point of the recording. For typical experimental data the shape of this matrix would be $N_{ele} \times N_{time}$, where N_{ele} is the number of electrodes and N_{time} is the total number of recorded time points. Next, we call the `KCSD2D` class with the relevant parameters. The only required parameters are the electrode positions, `ele_pos`, and the potentials they see, `pots`. We can also provide here the parameters for the forward model, `h` and `sigma`. We define a rectangular region of estimation by setting the values `xmin`, `xmax` and `ymin`, `ymax`. The number of basis functions, `n_src_init` is set to `1000`, basis functions are of the type `gauss`, and the width of the Gaussian basis source `R_init` is set to be `1`. Finally, estimated CSD is stored as `est_csd`.

```
In [7]: from kcsd import KCSD2D

        def do_kcsd(ele_pos, pots):
            h = 50.  # distance between the electrode plane and the midslice
            sigma = 1.0 # S/m
```

```
    pots = pots.reshape((len(ele_pos), 1)) # first time point
    k = KCSD2D(ele_pos, pots, h=h, sigma=sigma,
               xmin=0.0, xmax=1.0,
               ymin=0.0, ymax=1.0,
               n_src_init=1000, src_type='gauss', R_init=1.)
    return k

k = do_kcsd(ele_pos, pots)
est_csd = k.values('CSD')
```

Estimated current sources are shown in Fig. 11C. Compare this to the True CSD obtained before, Fig. 11A. Observe that the estimation is not very faithful. This is caused by the ground truth varying significantly in the scale of a single inter-electrode distance. In the next step we will use cross-validation to select better reconstruction parameters.

```
In [8]: make_plot(k.estm_x, k.estm_y, est_csd[:, :, 0],
                  title='Estimated CSD without CV', cmap=cm.bwr) # First time point

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8cfd447dd8>
```

Cross validation Leave-one-out cross-validation is performed with a single line command. In this procedure we scan a range of R values which set the size of the Gaussian basis functions and the regularization parameter λ values. At the end of this step we obtain the optimal parameters that would correct for overfitting. The function outputs the progress of the cross-validation step and displays the optimal candidates in the last line. Alternatively, one could use the L-curve method to find these optimal parameters. Fig. 11D shows the kCSD reconstruction obtained after cross-validation. We find that this estimation of the current sources resembles the True CSD better.

```
In [9]: k.cross_validate(Rs=np.linspace(0.01, 0.15, 15))
        est_csd = k.values('CSD')

  No lambda given, using defaults
  Cross validating R (all lambda) : 0.01
  Cross validating R (all lambda) : 0.02
  ...
  Cross validating R (all lambda) : 0.15
  R, lambda : 0.11 1.46779926762e-06
```

```
In [10]: make_plot(k.estm_x, k.estm_y, est_csd[:, :, 0],
                   title='Estimated CSD with CV', cmap=cm.bwr) # First time point

Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8cfd2707f0>
```

Noisy electrodes

Until now we assumed noise-free data, however, experimental data are always noisy. In this section we investigate how noise affects the kCSD estimation. We first show how to compute the reliability map which we introduced before, Eq. (10). Then we discuss reproducible generation of noisy data with varying noise amplitude. Finally, we study the error in the reconstruction as a function of changing noise level.

Reconstruction quality measure To assess the estimation quality we measure the point-wise difference between the true sources and the sources reconstructed with the kcsd. We define a function `point_errors` which takes the `true_csd` and the `estimated_csd` as the inputs, normalizes them individually, and computes the Frobenius norm of their difference.

```
In [11]: def point_errors(true_csd, est_csd):
             true_csd_r = true_csd.reshape(true_csd.size, 1)
             est_csd_r = est_csd.reshape(est_csd.size, 1)
             epsilon = np.linalg.norm(true_csd_r)/np.max(abs(true_csd_r))
             err_r = abs(est_csd_r/(np.linalg.norm(est_csd_r)) -
                         true_csd_r/(np.linalg.norm(true_csd_r)))
             err_r *= epsilon
             err = err_r.reshape(true_csd.shape)
             return err

         error_ideal = point_errors(true_csd, est_csd)
```

We visualize this difference as before, except we use greyscale colormap to display the intensity of the reconstruction error. For convenience we define the plotting in a function called `make_error_plot`. The output from this step is shown in Fig. 12A.

```
In [12]: def make_error_plot(xx, yy, error, title='Error CSD'):
             fig = plt.figure(figsize=(7, 7))
             ax = plt.subplot(111)
             ax.set_aspect('equal')
             t_max = np.max(np.abs(error))
             levels = np.linspace(0, t_max, 32)
             im = ax.contourf(xx, yy, error, levels=levels, cmap=cm.Greys)
             ax.set_xlabel('X (mm)')
             ax.set_ylabel('Y (mm)')
             ax.set_title(title)
             ticks = np.linspace(0, t_max, 3, endpoint=True)
             plt.colorbar(im, orientation='horizontal', format='%.2f', ticks=ticks)
             return ax

         make_error_plot(k.estm_x, k.estm_y, error_ideal,
                         title='Error CSD, no noise')

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8cfd1bac50>
```

Noise definition To study resilience of the reconstruction against the noise in a controlled way we seed the random number generator in the function `add_noise`. We consider normally distributed noise with the mean and standard deviation set by reference to the recorded potentials.

```
In [13]: def add_noise(pots, noise_level=0, noise_seed=23):
             rstate = np.random.RandomState(noise_seed)
             noise = noise_level*0.01*rstate.normal(np.mean(pots),
```

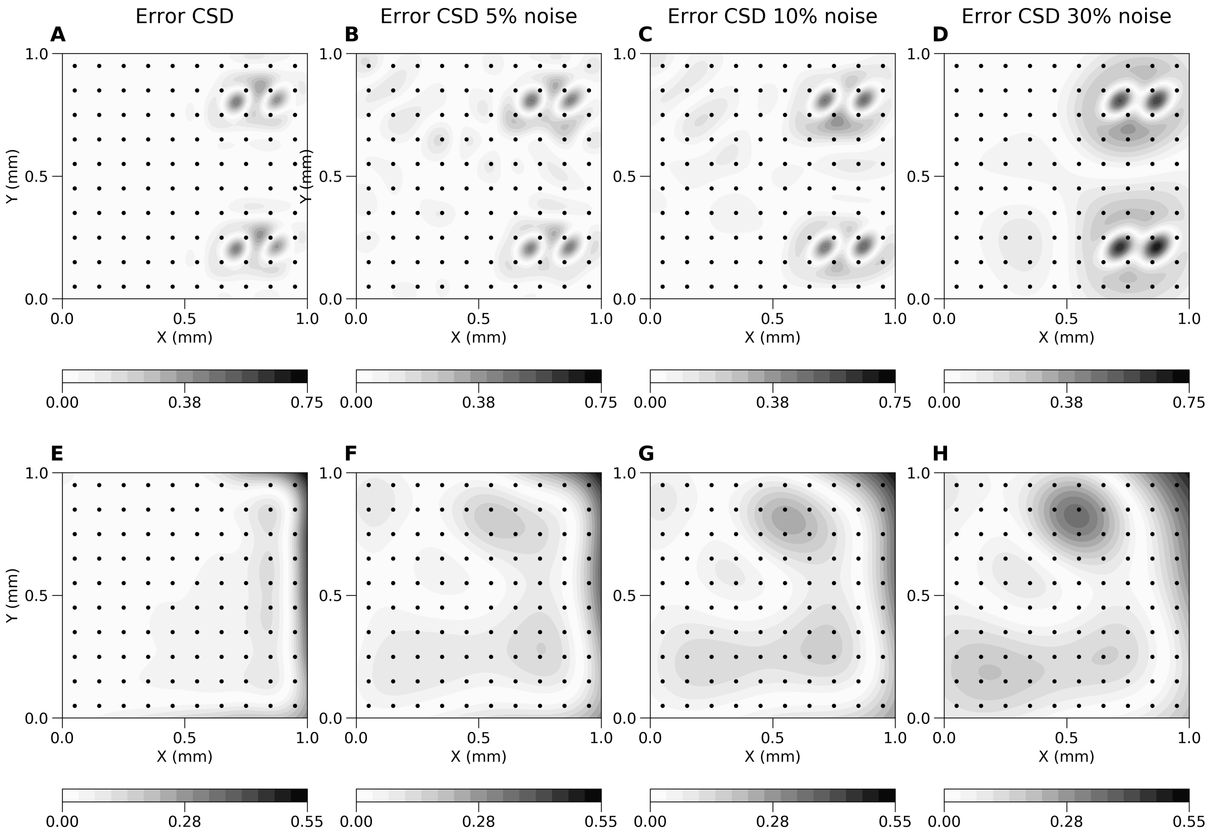

Figure 12: Noisy electrodes. A) The error between the True CSD and the estimation obtained with kCSD for a 2 dimensional small Gaussian current source, using the csd seed of 15. The electrodes in this case are assumed to be noise-free. B, C, D) Same as A, however, noise is added to the recorded potentials, whose magnitude is 5%, 10%, or 30%, respectively. E-H) Analogous to A–D, except in this case large Gaussian sources with seed 6 were used.

```
                                         np.std(pots),
                                         size=(len(pots), 1))
        return pots + noise.reshape(pots.shape)

    pots_noise = add_noise(pots, noise_level=15, noise_seed=23)
```

Source reconstruction from noisy data With these tools we can study the effects of noise on the reconstruction. We now generate noise for a given noise level between 0 and 100, add it to the simulated potential, and estimate CSD from these noisy potentials. We can then use the error plots to compare the reconstruction with the True CSD. Notice that the parameters giving best reconstruction obtained for noisy data in general will be different from those obtained for clean potentials to compensate for noise.

```
In [14]: k_noise = do_kcsd(ele_pos, pots_noise)
         k_noise.cross_validate(Rs=np.linspace(0.01, 0.15, 15))
         estm_csd_noise = k_noise.values('CSD')
         error_noise = point_errors(true_csd, estm_csd_noise)
```

```
No lambda given, using defaults
Cross validating R (all lambda) : 0.01
Cross validating R (all lambda) : 0.02
...
Cross validating R (all lambda) : 0.15
R, lambda : 0.01 0.00110069417125
```

We can display this error with the `make_error_plot` plotting function which we defined earlier. Changing the `noise_level` and the `noise_seed` affects the reconstruction, but the error depends also on the sources, so changing the True CSD type to a `gauss_2d_large` or changing `csd_seed` will lead to different results. This is illustrated in Fig. 12A–D for small Gaussian sources, and Fig. 12E–H for large Gaussian sources, with varying noise levels. The actual ground truth and reconstructions are shown in Fig. 11.

```
In [15]: make_error_plot(k_noise.estm_x, k_noise.estm_y, error_noise,
                         title='Error CSD, with noise')
```

Broken electrodes

It often happens that one needs to discard recordings from a subset of setup. This can happen when some electrodes are used for stimulation and cannot be used for recording, or for data managing purposes the bandwidth limitations may require a compromise between sampling rates and the number of electrodes being monitored simultaneously, or electrode may break down or get too noisy and their signals must be discarded. In this tutorial we discuss how to handle such cases and how to investigate the incurred errors in reconstruction. We first show how we remove recordings from selected (broken) electrodes from considered data. Then we calculate the estimation error for a given source for data from a damaged setup. Finally, we compute the average error across many sources from incomplete data. Note that kCSD reconstruction is designed to work with arbitrary electrode setups. Removing specific electrodes does not change the situation significantly. We focus here on broken electrodes as we see it is a common enough situation in practice that it deserves a consideration. We want to show how one can gain intuition regarding ways and places in which reconstruction may go wrong, when we slightly disturb a setup we are familiar with.

Remove broken electrodes To test the effects of removed electrodes on reconstruction from a given setup we simulate this with a function `remove_electrodes` that takes all the electrode positions for this setup and the number of electrodes that are to be removed. In this example we remove the electrodes randomly. Like we did previously, to facilitate reproducibility we also pass a `broken_seed` variable, so that at each subsequent run the same electrodes are discarded. By changing this seed we select a different set of electrodes for removal.

```
In [16]: def remove_electrodes(ele_pos, num_broken, broken_seed=42):
             rstate = np.random.RandomState(broken_seed)
             rmv = rstate.choice(ele_pos.shape[0], num_broken, replace=False)
             ele_pos = np.delete(ele_pos, rmv, 0)
             return ele_pos

         ele_pos_new = remove_electrodes(ele_pos, 5)  # Discard 5 electrodes of 100
```

Error in estimation with broken electrodes After removing the broken electrodes we compute the estimation error to gauge the effect of electrode removal on reconstruction. Here, a fuction `calculate_error` takes a `csd_seed` as an input, which selects a specific ground truth source, and all the remaining electrode positions, `ele_pos`. The function computes the True CSD for a `gauss_2d_small` type source, computes the potential at these electrode locations, performes kcsd estimation from these data, and computes the error in the estimation of the true csd.

```
In [17]: def calculate_error(csd_seed, ele_pos):
             true_csd = CSD_PROFILE(csd_at, seed=csd_seed)
             pots = forward_method(ele_pos, csd_at, true_csd)
             k = do_kcsd(ele_pos, pots)
             k.cross_validate(Rs=np.linspace(0.01, 0.15, 15))
             err = point_errors(true_csd, k.values('CSD'))
             return k, err

         k, err = calculate_error(csd_seed=15, ele_pos=ele_pos_new)
```

```
No lambda given, using defaults
Cross validating R (all lambda) : 0.01
Cross validating R (all lambda) : 0.02
...
Cross validating R (all lambda) : 0.15
R, lambda : 0.14 1.33352143216e-05
```

Below (Fig. 13) we plot these errors. We also display the electrodes which were used in the kcsd estimation.

```
In [18]: ax = make_error_plot(k.estm_x, k.estm_y, err,
                              title='Error CSD, 5 broken electrodes')
         ax.scatter(ele_pos_new[:, 0], ele_pos_new[:, 1], 10, c='k')
```

Average error for multiple sources As we can see, the estimation error depends on the test current sources used. To better understand the effects of the setup we compute the average error across multiple sources. As an example here we show this for two seeds. In principle, any type and number of sources may be tested, as we showed before in analysis of reliability maps. This step is computationally expensive, however, it would normally be carried out only once for a given electrode design configuration. We believe this approach offers useful diagnostics and builds intuition regarding the estimation power for the given setup.

```
In [19]: seed_list = range(15, 17)
         error_list = []
         for csd_seed in seed_list:
             k, err = calculate_error(csd_seed=csd_seed, ele_pos=ele_pos_new)
             error_list.append(err)
         avg_error = sum(error_list) / len(error_list)
```

```
No lambda given, using defaults
Cross validating R (all lambda) : 0.01
```

```
...
Cross validating R (all lambda) : 0.15
R, lambda : 0.14 1.33352143216e-05
No lambda given, using defaults
Cross validating R (all lambda) : 0.01
...
Cross validating R (all lambda) : 0.15
R, lambda : 0.04 3.48070058843e-15
```

In Fig. 13A–D we show this for the case of 0, 5, 10 and 20 broken electrodes, when the average error for 100 small Gaussian sources was considered. In Fig. 13E–H we show the same for large Gaussian sources. 514 515 516

```
In [23]: ax = make_error_plot(k.estm_x, k.estm_y, avg_error,
                              title='Average Error, 5 broken electrodes')
         ax.scatter(ele_pos_new[:, 0], ele_pos_new[:, 1], 10, c='k')
```

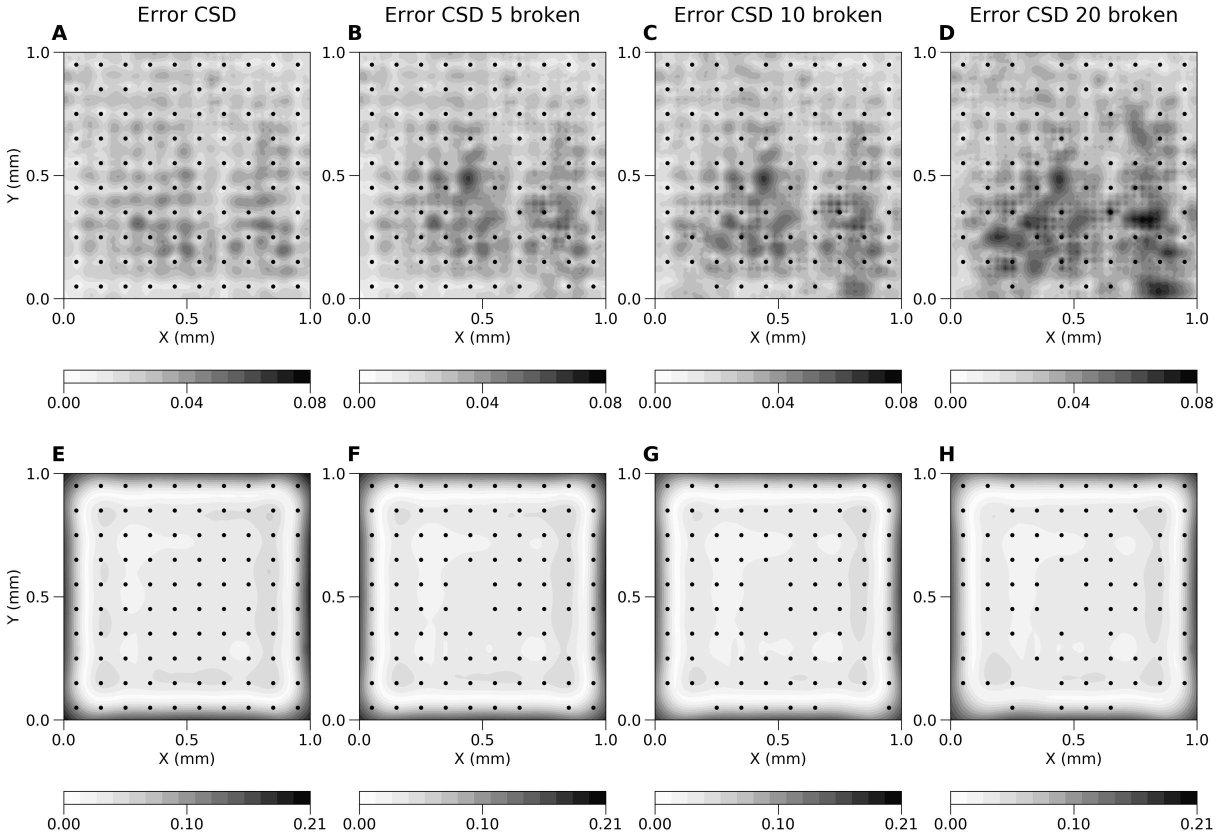

Figure 13: Broken electrodes. A) Shows the average error between the True CSD and the CSD estimated with kcsd for 100 random small Gaussian current sources. B) The same average error as in A, except in this case 5 electrodes were discarded in the estimation. Likewise for C and D, where 10 and 20 electrodes out of the 100 were considered broken. E-H) analogous to A–D, except in this case we show the averages for 100 large Gaussian current sources.

Discussion

In the present work we returned to the kernel Current Source Density method introduced by [Potworowski et al., 2012] for two reasons. First, to introduce a new Python package for kCSD estimation. In the Results section we provided a brief tutorial to the new package and an overview of its main functions. All the figures in this paper showing CSD and LFP were computed with this package and source files are provided. Second, to discuss some mathematical properties of the kCSD method, especially in view of what information it is possible to extract from sparse sampling of the potential and the limitations of the kCSD procedure. In this section we discuss several issues related to CSD analysis in general and kCSD in particular.

To LFP or to CSD? Extracellular potentials provide valuable insight about the functioning of the brain. Thanks to recent advances in multielectrode design and growing availability of sophisticated recording probes we can monitor the electric fields in the nervous system at thousands of sites, often simultaneously. One may wonder if this increased resolution makes CSD analysis unnecessary. In our view, as we have discussed many times, it is not so. The long range nature of electric potential means that even a single source contributes to every recording. Thus in principle we should always expect strong correlation between neighboring sites. However, if the separation between the electrodes becomes substantial, on the order of millimeters, the level of correlation between the recordings on different electrodes will decrease. This is because each electrode effectively picks up signals from a different composition of sources. Even if some are shared they are overshadowed by others which may lead to small interelectrode correlations. Still, our experience shows that significant correlations can be observed in the range of several millimeters [Łęski et al., 2007, Hunt et al., 2011] which is consistent with literature [Lindén et al., 2011, Łęski et al., 2013].Fundamentally, the LFP profile is different from CSD profile, and may significantly distort or hide features of importance. For example, Fig. 5.C shows the source composed of three gaussians, while direct inspection of LFP indicates a simple dipole.

In view of these facts we argue that it is always beneficial to attempt kCSD analysis. The caveat is not to believe the reconstructed CSD blindly but always interpret it against known anatomical and physiological knowledge supported by the tools such as provided in the present work (eigensources, reliability maps, etc). In the worst case, for a very small number of electrodes, while the reconstructed CSD will not be a good representation of the true sources, nevertheless, the kCSD procedure will still have the sharpening or deblurring properties and can be thought of as another decomposition method, such as PCA or ICA, simply using physical properties of electric field propagation for signal separation rather than orthogonalization or entropy maximization (or others for other decomposition methods). To use kCSD in this way we would estimate CSD at the positions of the electrodes only since this gives as many values as recorded, does not pretend to introduce new knowledge, but may correct for noise and better localize independent signals. This could be combined with other decomposition methods if desired which may give more physiologically interpretable results [Łęski et al., 2010, Głąbska et al., 2014].

Approaches to CSD estimation. Several procedures for CSD estimation were introduced over the years and are still in use today. The first approach, which probably still dominates today, was introduced by Walter Pitts in 1952 [Pitts, 1952] and gained popularity after Nicholson and Freeman adopted it for laminar recordings [Nicholson and Freeman, 1975]. This was a direct numerical approximation to computation of the second derivative in the Poisson equation (1). Only minor improvements were introduced over the years to stabilize estimation [Rappelsberger et al., 1981] or handle boundaries [Vaknin et al., 1988]. The first major conceptual change was introduced by Pettersen et al. [2006] who introduced model-based estimation of the sources. Their idea was to assume a parametric model of sources, for example, spline interpolated CSD between electrode positions, and using forward modeling to connect measured potentials to model parameters. This model-based approach was generalized by Potworowski et al. [2012] who proposed a non-parametric kernel Current Source Density method which is the focus of the present work.

Apart from these main approaches several variants were proposed. For example, one may interpolate the potential first before applying traditional CSD approach, or the opposite, interpolate traditionally estimated CSD. Although in some cases the obtained results may look close to those obtained with kCSD, we do prefer kernel CSD approach due to the underlying theory which facilitates computation of estimation errors but also yields a unified framework for handling underlying assumptions, noisy data and irregular electrode distributions. In our view approaches combining ad hoc interpolation with numerical derivatives conceptually and computationally are less convincing to iCSD and kCSD and we would not recommend them.

Models of tissue. Throughout this work we assumed purely ohmic character of the sources. This has been debated in recent years [Bédard and Destexhe, 2011, Riera et al., 2012, Gratiy et al., 2017] and it is true that more complex biophysical models of the tissue, taking into account frequency dependent conductivity or diffusion currents, would influence the practice of source reconstruction or its interpretation. However, the available data indicate that in the range of frequencies of physiological interest these effects are small. While one should keep eyes open on the new data as they become available and keep in mind the different possible sources which may affect the reconstruction or interpretation, we believe that the traditional view of ohmic tissue is an adequate basis for typical experimental situations and going beyond that would probably require additional dedicated measurement for the experiment at hand which may not always be feasible. For example, as we discussed in [Ness et al., 2015], the specimen variability of the cortical conductivity in the rat is much bigger than the variability between different directions within a given rat [Goto et al., 2010]. This means that unless we have conductivity measurements for our specific rat we are probably making smaller error assuming isotropic conductivity than taking different values from literature. We feel there is not enough data to justify inclusion of more complex terms in the standard CSD analysis to be applied throughout the brains and species.

In this manuscript we also assumed constant conductivity. We are convinced this is a reasonable approximation for typical depth recordings. In general, however, this approximation needs to be justified or alternative models of tissue need to be considered. In principle, the kCSD method can be applied for a variety of tissue models as long as the basis potentials can be computed from the basis sources while incorporating the geometric and conductivity changes.

For example, Ness et al. [2015] considered a cortical slice placed on a microelectrode array (MEA) in which they included the geometry of the slice and modeled saline-slice interface with changing conductivity in the forward model. They found that Method of Images (MoI) gives a good approximation to the full solution obtained using finite-element model (FEM). This approximation was incorporated within the kCSD method as

MoIkCSD variant and is available in the `kCSD-python` package.

It is possible to generalize kCSD to reconstruct sources from recordings of multiple electrical modalities — LFP, ECoG, EEG. In this case one needs to include the head geometry and the changing tissue properties within the forward model and in the kCSD method. The anisotropic (white matter tracts) and inhomogeneous (varying between skull, cerebro-spinal fluid, gray matter and white matter) electrical conductivity changes can be approximated using data obtained with imaging techniques such as MRI, CT or DTI. Such sophisticated head models require numerical solutions such as finite element modeling (FEM) to compute the basis potentials from the basis sources. We are currently working on this approach to make it generic for any animal head and to eventually utilize it as a source localization method for human data, for example, to localize foci of pharmacologically intractable epilepsy seizures in humans. We call this extension kernel Electrical Source Imaging (kESI).

High density microelectrode recordings. One of the trends clearly observed in modern neurotechnology is the drive towards increasing the number of sensors and their density [Buzsáki, 2004, Berdondini et al., 2005, Frey et al., 2009, Hottowy et al., 2012, Jun et al., 2017, Angotzi et al., 2019], for *in vitro* and *in vivo* studies. While it seems that a better resolution for recording spiking activity of multiple cells is the main goal, also more precise stimulation and field potentials monitoring are targeted [Hottowy et al., 2012, Ferrea et al., 2012, Bakkum et al., 2013]. Such massive high density data from thousands of electrodes should greatly increase insight into the studied systems and significantly improve results of CSD reconstructions. There are two obstacles to fully benefit from kCSD analysis of data from these new systems. First, kCSD involves inversion of the kernel matrix which is quadratic in the number of electrodes. Combined with cross-validation the necessary matrix operations quickly become overwhelming. This can be mitigated in a number of ways, by subsampling the data, approximate inversions, and by switching from cross-validation to L-curve method, but the challenge remains. This is the easy problem. The difficult problem is physical. As we move away from a source its contribution to the recorded potential goes down. In consequence, since the present version of kCSD uses all recordings to estimate every source, when using remote signals to estimate local source, we obtain mainly contributions from noise. In effect we get a very reliable estimation of sources varying slowly in space but the sources changing fast in space are treated as noise and silenced by the regularization.

To take full advantage of these data a new approach must be developed. We are currently working on a multiscale approach which we call kCSDHD. The idea is to perform reconstructions in small windows in multiple scales to optimally reconstruct multiscale features of the source distribution and the challenge is to efficiently and correctly stitch them together. This will be reported in the future.

Parameter selection. In the Results section we discussed our strategy for data-based parameter selection using cross-validation or L-curve. Often, we need to tune not just λ but also other parameters. For example, for Gaussian basis sources we may need to decide on the width of the Gaussian used, R. To obtain the optimal set of parameters in that case we compute the curvature of the L-curve or the cross-validation error for some ranges of parameters considered and select parameters corresponding to the maximum curvature / minimum error in the parameter space. This is a simplification of the proposition by Belge et al. [2002] which in practice we found very effective.

As an example, in Fig. 14 we show results of such a scan for the problem shown in

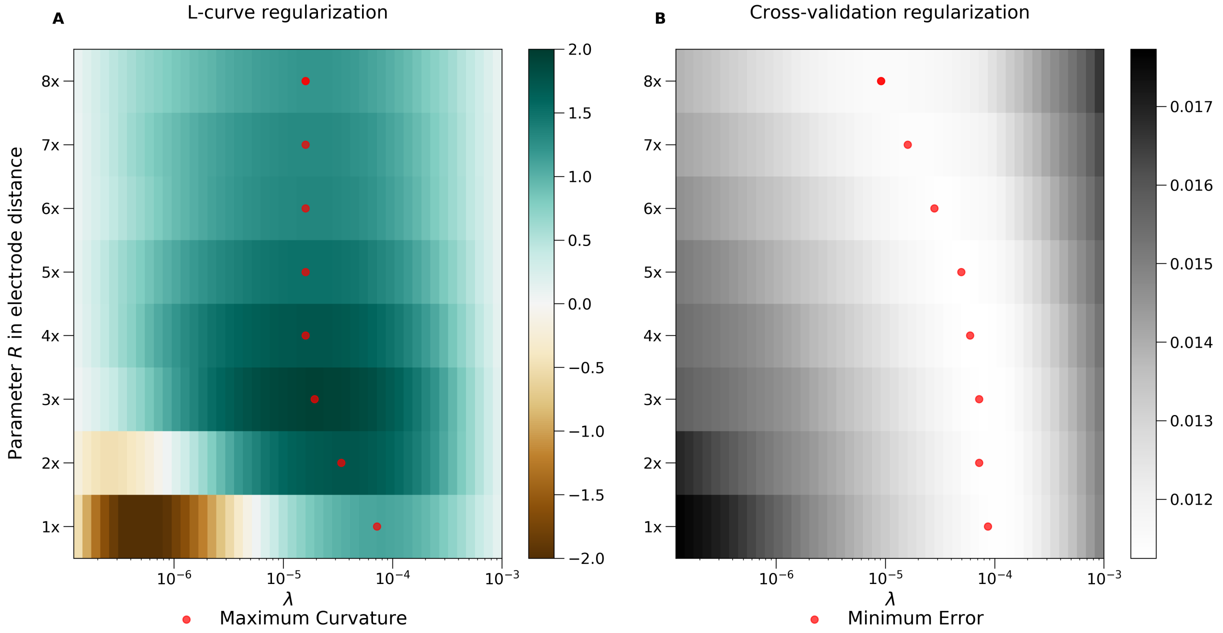

Figure 14: L-curve curvature (left) and CV-error (right) for the problem studied in Fig. 5. Observe that in both cases there are ranges of promising candidate parameter pairs, R, λ, which can give good reconstruction given the measured data. Red dots shows local extrema for each value of R fixed. See text for discussion of this effect.

Fig. 5. The range of λ to be considered can be set by hand but by default we base it on the eigenvalues of K. The smallest λ is set as the minimum eigenvalue of K which here was around 1e-10. We set maximum λ at standard deviation of the eigenvalues, which here was around 1e-3. The range of R values studied was from the minimum interelectrode distance to half the maximum interelectrode distance. Note that for very inhomogeneous distributions of electrodes this approach may be inadequate.

What we find is that apart from a global minimum in R, λ space there is a range of R values fixing which we can find optimal $\lambda(R)$ which leads to very close curvatures / CV-errors / estimation results. What happens is that within some limits we may achieve similar smoothing effects changing either λ or R. Bigger λ means more smoothing, but bigger R means broader basis functions and effectively also smoother reconstruction space. This is why the CV-error and curvature landscapes are relatively flat, or have these marked valleys observed in Fig. 14. This effect supports robustness of the kCSD approach.

The sources of error in kCSD estimation and how to deal with them. Kernel CSD method assumes a set of electrode positions and a corresponding set of recordings. Additionally, single cell kCSD requires morphology of the cell which contributed to the recordings and its position relative to the electrodes. Each of these may be subject to errors.

We assume that the electrode positions are known precisely. This is a justified assumption in case of multishaft silicon probes or integrated CMOS-MEA but not necessarily when multiple laminar probes are placed independently within the brain or for many other scenarios. We do not provide dedicated tools to study the effects of misplaced electrodes on the reconstructed CSD, however, this can be achieved easily with the provided package if needed. The location of the cell with respect to the electrodes is much more questionable, especially in 3D. Nevertheless, the necessary data to perform skCSD are too scarce to start addressing these issues.

On the other hand we do assume that the recordings are noisy and we use regularization to counteract the effects of noise. We have no mechanism to differentiate between electrodes with varying degrees of noise to compensate this differently. However, we observed that for cases with very bad electrodes, similar results are obtained for analysis of complete data and for analysis of partial data with bad electrodes removed from analysis. The difference was in λ selected which was larger when broken electrodes were included in the analysis. Depending on situation, if there is a big difference in the noise visible in different channels, an optimal strategy may be to discard the noisy data and perform reconstruction from the good channels only, which kCSD permits.

The main limitation of the method itself lies in the character of any inverse problem. Here it means that there is an infinite number of possible CSD distributions each consistent with the recorded potential. It is thus necessary to impose conditions which allow unique reconstruction and this is what every variant of CSD method is about. In kCSD this condition is minimization of regularized prediction error. In practical terms one may think of the function space in which we are making the reconstruction. This space is span by the eigensources we discussed before. We feel it is useful to consider both this space as well as its complement, that is the set of CSD functions whose contribution to every potential is zero. This can facilitate understanding of which features of the underlying sources can be recovered and which are inaccessible to the given setup. While for the most common regular setups, such as rectangular or hexagonal MEA grids or multishaft probes, intuitions from Fourier analysis largely carry over, in less regular cases this quickly becomes non-obvious.

To facilitate intuition building in the provided toolbox we include tools to compute the eigensources for a given setup. We also proposed here reliability maps, heuristic tools to build intuition regarding which parts of the reconstructed CSD can be trusted and which seem doubtful. These reliability maps are built around specific test ground truth distributions and some default parameters facilitating validation for any given setup are provided, due to the open source nature of the provided toolbox, more complex analysis is possible if the setup or experimental context require that.

Overview of kCSD-python package

This paper introduces the `kCSD-python` package — a new implementation of the kernel Current Source Density method [Potworowski et al., 2012] and its two recent variants ([Ness et al., 2015] and [Cserpan et al., 2017]). It is open source and available under the modified BSD License (3-Clause BSD) on GitHub (https://github.com/Neuroinflab/kCSD-python). It has been designed using test driven development and utilizes the continuous integration provided by Travis CI. It supports Python 2.7 and 3.7 versions and has a bare minimal library requirements (`numpy`, `scipy` and `matplotlib`). It can be installed using the Python package installer (`pip`) or using Anaconda python package environment (`conda`).

To facilitate uptake of this resource, the package comes with two extensive tutorials implemented in jupyter notebook. These tutorials allow users to test different configurations of current sources and electrodes to see the method in action. These provisions make the advantages and limitations of this method transparent to its users. Furthermore, these tutorials can be accessed without any installation on a web browser via Binder [Project

Jupyter et al., 2018]. It is extensively documented (https://kcsd-python.readthedocs.io) and includes all the necessary scripts to generate the figures in this manuscript.

Methods

Review of Kernel Current Source Density estimation

Basis functions. Here we repeat the key steps in the construction of kCSD estimation framework [Potworowski et al., 2012] to introduce the notation and establish the basic notions.

We first construct a pair of related function spaces in which we perform the estimation, space of sources $\tilde{\mathcal{F}}$ and space of potentials $\mathcal{F}$,

$$\begin{aligned}\tilde{\mathcal{F}} &= \left\{C(\mathbf{x}) = \alpha_1\tilde{b}_1(\mathbf{x}) + ... + \alpha_M\tilde{b}_M(\mathbf{x}) : \tilde{b}_i : M_C \longrightarrow \mathbb{R}\right\}, \qquad (11)\\ \mathcal{F} &= \{V(\mathbf{x}) = \alpha_1 b_1(\mathbf{x}) + ... + \alpha_M b_M(\mathbf{x}) \,:\, b_i : M_V \longrightarrow \mathbb{R}\}. \qquad (12)\end{aligned}$$

We select the basis source functions $\tilde{b}_i$ so that they are convenient to work with, such as step functions or gaussians, with support over regions which are most natural for the problem at hand. For example, when reconstructing the distribution of current sources along a single cell from a set of recordings with a planar microelectrode array, M_C is the neuronal morphology, which we take to be locally 1D set embedded in real 3D space, while M_V would be the 2D plane defined by the MEA.

The potential basis functions, b_i, are defined as the potential generated by $\tilde{b}_i$, so that $b_i = \mathcal{A}\tilde{b}_i$, where $\mathcal{A} : \tilde{\mathcal{F}} \mapsto \mathcal{F}$. Specific form of $\mathcal{A}$ operator depends on the problem at hand, the dimensionality of space in which estimation is desired, as well as on physical models of the medium, such as tissue conductivity, slice or brain geometry, etc. [Pettersen et al., 2006, Łęski et al., 2007, 2011, Ness et al., 2015, Cserpan et al., 2017]. In the simplest case of infinite, homogeneous and isotropic tissue in 3D we have

$$\begin{aligned}b_i(x,y,z) &= \mathcal{A}_3\tilde{b}_i(x,y,z)\\ &= \frac{1}{4\pi\sigma}\int dx'\int dy'\int dz'\,\frac{\tilde{b}_i(x',y',z')}{\sqrt{(x-x')^2+(y-y')^2+(z-z')^2}}. \qquad (13)\end{aligned}$$

In general, we can consider arbitrary conductivity and geometry of the tissue which may force us to use approximate numerical methods, such as finite element schemes. For example, Ness et al. [2015] show an application of kCSD for a slice of finite thickness and specific geometry, as well as a method of images approximation for kCSD for typical slices on multielectrode arrays (recordings far from the boundary, slice much thinner than its planar extent).

In the past we considered CSD reconstruction for recordings from 1D, 2D and 3D setups under assumption of infinite tissue of constant conductivity [Potworowski et al., 2012], we used method of images to improve reconstruction for slices of finite thickness on MEA under medium of different conductivity (ACSF, [Ness et al., 2015]) and we considered reconstruction of sources along single cells when we have reasons to trust the recorded signal to come from a specific cell of known morphology [Cserpan et al., 2017]. All these variants are implemented in the present code. Fig. 1 shows these scenarios.

For laminar probes, Fig. 1.A), following [Pettersen et al., 2006], we assumed elementary current sources contributing to the potential of the form $\tilde{b}_i(z)H(x,y)$. Here $\tilde{b}_i(z)$ is the one-dimensional basis source (we usually assume a Gaussian of width R). Since information beyond the electrode axis is unavailable we assume rotational symmetry around z. We usually assume $H(x, y)$ a step function on a disk of radius h:

$$H(x,y) = \begin{cases} 1 & x^2+y^2 \le h^2, \\ 0 & \text{otherwise}. \end{cases}$$

This can be integrated yielding 1D potential basis functions of the form

$$b_i(z) = \frac{1}{2\sigma} \int dz' \left(\sqrt{(z-z')^2+h^2} - |z-z'| \right) \tilde{b}_i(z'). \tag{14}$$

For planar setups, Fig. 1.B) [Łęski et al., 2011], we usually assume Gaussian basis sources $\tilde{b}_i(x, y)$, physically contributing to the potential with $\tilde{b}_i(x, y)H(z)$, where

$$H(z) = \begin{cases} 1 & -h \le z \le h \\ 0 & \text{otherwise}. \end{cases}$$

This can be integrated to give the potential in the electrode plane:

$$b_i(x,y) = \frac{1}{2\pi\sigma} \int dx' \int dy' \operatorname{arsinh} \left(\frac{2h}{\sqrt{(x-x')^2+(y-y')^2}} \right) \tilde{b}_i(x',y'). \tag{15}$$

This approach give two parameters describing the CSD basis functions, the width of the relevant Gaussian, R, and the thickness of contributing layer in 2D case or radius of circular sheath in 1D case (h). Note that if we assume above $H(x, y)$ and $H(z)$ to be Gaussian as well with the same width, in all three dimensionalities the individual contributions are spherically symmetrical Gaussians. Therefore, the same 3D approach can be used. Further, it can be integrated to yield potential in a closed form. Indeed, from eq. (13), taking

$$\tilde{b}_j(\bar{x}) = \frac{1}{(\sqrt{2\pi}R)^3} \exp \frac{-(\bar{x}-\bar{x}_j)^2}{2R^2}$$

we can show that

$$b_j(\bar{x}) = \frac{1}{4\pi\sigma|\bar{x}-\bar{x}_j|} \operatorname{erf} \left(\frac{|\bar{x}-\bar{x}_j|}{\sqrt{2}R} \right)$$

where

$$\operatorname{erf}(r) = \frac{2}{\sqrt{\pi}} \int_0^r e^{-t^2} dt.$$

This is also implemented in the present code.

kCSD framework. We can think of the whole set of the potential basis functions $b_i(\mathbf{x})$ as features representing $\mathbf{x}$ in a M-dimensional space through related embeddings

$$\begin{aligned} \Phi(\mathbf{x}) &= [b_1(\mathbf{x}), \ldots, b_M(\mathbf{x})]^T, \\ \tilde{\Phi}(\mathbf{x}) &= [\tilde{b}_1(\mathbf{x}), \ldots, \tilde{b}_M(\mathbf{x})]^T. \end{aligned}$$

Let us introduce a kernel function in $\mathcal{F}$ through

$$K(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}') = \sum_{i=1}^{M} b_i(\mathbf{x}) b_i(\mathbf{x}'). \tag{16}$$

This kernel turns $\mathcal{F}$ into a reproducing kernel Hilbert space (RKHS, Aronszajn [1950]) whose properties we discussed in Potworowski et al. [2012]. In particular, we can show that all potential profiles admissible by our construction can be written as linear combinations of multiple kernels fixed with one leg at different points:

$$\mathcal{F} = \left\{ \sum_{i=1}^{l} \beta_i K(\mathbf{x}_i, \mathbf{x}) : l \in \mathbb{N}, \mathbf{x}_i, \mathbf{x} \in M_V, \beta_i \in \mathbb{R}, i = 1, ..., l \right\}. \tag{17}$$

We introduce the inner product of functions in $\mathcal{F}$, $f(\mathbf{x}) = \sum_{i=1}^{l} f_i K(\mathbf{y}_i, \mathbf{x})$, $g(\mathbf{x}) = \sum_{j=1}^{m} g_j K(\mathbf{z}_j, \mathbf{x})$, as

$$\langle f, g \rangle_{\mathcal{F}} = \sum_{i=1}^{l} \sum_{j=1}^{m} f_i g_j K(\mathbf{y}_i, \mathbf{z}_j). \tag{18}$$

Using this inner product we define the norm in $\mathcal{F}$ by $\|f\|^2_{\mathcal{F}} = \langle f, f \rangle_{\mathcal{F}}$, and we induce a norm in $\tilde{\mathcal{F}}$ by

$$\|\tilde{f}\|^2_{\tilde{\mathcal{F}}} := \|\mathcal{A}\tilde{f}\|^2_{\mathcal{F}} = \|f\|^2_{\mathcal{F}} = \sum_{i=1}^{M} f_i^2. \tag{19}$$

Note that we have now two representations of every function in $\mathcal{F}$, as a sum of kernels or a sum of basis elements

$$V(\mathbf{x}) = \sum_{j=1}^{l} \beta_j K(\mathbf{x}_j, \mathbf{x}) = \sum_{i=1}^{M} \alpha_i b_i(\mathbf{x}), \tag{20}$$

where

$$\alpha_i = \sum_{j=1}^{l} \beta_j b_i(\mathbf{x}_j), \tag{21}$$

where $\mathbf{x}_j$ are some positions in space.

One can see that in the two representations we have

$$\|V\|^2_{\mathcal{F}} = \sum_{i=1}^{l} \sum_{j=1}^{l} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{M} \alpha_i^2. \tag{22}$$

Source estimation with kCSD. Estimation of current sources with kCSD consists of two steps. The first is kernel interpolation of the potential, the second is changing the space from potential to sources. Conceptually, in the simplest case, this is equivalent to applying Laplacian or double derivative to the potential field obtained in the whole space. However, using our approach with double kernels, which take into account underlying physics and geometry of the studied system, it is possible to apply these ideas to more complex situations, e.g. slices of specific shape and conductivity profile Ness et al. [2015] or fields generated by individual cells Cserpan et al. [2017].

To estimate the potential in the whole space we minimize an error function

$$\text{err}[V] = \sum_{i=1}^{N} (V(\mathbf{x}_i) - V_i)^2 + \lambda \|V(\mathbf{x})\|^2_{\mathcal{F}}, \tag{23}$$

where the first term indicates proximity of our model to actual measurement, while the second constrains complexity of the model[2]. Using the representer theorem Kimeldorf and Wahba [1971] we can show that the solution is of the form

$$V^*(\mathbf{x}) = \sum_{j=1}^{N} \beta_j K(\mathbf{x}, \mathbf{x}_j), \tag{24}$$

where $\mathbf{x}_j$ are the N electrode positions. Minimum of (23) is obtained for

$$\beta = (\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{V},$$

where

$$\mathbf{K}_{i,j} \equiv K(\mathbf{x}_i, \mathbf{x}_j). \tag{25}$$

Now that we have the potential given by combination of kernels, Eq. (24), we can expand it in the original basis $b_i(\mathbf{x})$, Eq. (20). From that we obtain a consistent estimate of the CSD by lifting the model from the potential to the CSD representation:

$$C^*(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i \tilde{b}_i(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \sum_{i=1}^{M} b_i(\mathbf{x}_j)\tilde{b}_i(\mathbf{x}) = \sum_{j=1}^{N} \beta_j \widetilde{K}(\mathbf{x}, \mathbf{x}_j), \tag{26}$$

where we introduce the cross-kernel function[3]

$$\widetilde{K}(\mathbf{x}, \mathbf{y}) \equiv \sum_{i=1}^{M} \tilde{b}_i(\mathbf{x}) b_i(\mathbf{y}).$$

With this definition we can write

$$\underline{\mathbf{C}}^* = \underline{\widetilde{\mathbf{K}}}(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{V}. \tag{27}$$

Test sources used

In several tests and demonstrations in this paper we use two families of sources which we call small and large sources. They were defined by [Potworowski et al., 2012] in their appendix. They are implemented in functions `gauss_2d_small` and `gauss_2d_large` available in file `csd_profile.py`. Two pdf files with images showing both source families (100 sources each) are available as supplementary materials (`large_all.pdf` and `small_all.pdf`). These specific source families were selected to test reconstruction of easy (large) and difficult (small) cases while supporting reproducibility of computations and figures. Otherwise, there is nothing special about them.

Supplementary Materials

Supplementary Fig. 15 shows spectral properties of kCSD method for simple 2D case with 9 regularly distributed electrodes in the square $[0, 1] \times [0, 1]$. This corresponds conceptually to Fig. 2 for the setup from Fig. 7).

[2] Note a typo in eq. (4.2) in the original paper, which incorrectly states the error term as $\sum_i \beta_i^2$ while it should be $\|V\|_{\mathcal{F}}^2$, as given here by Eq. 22.

[3] Note that this definition replaces the two variables with respect to the original definition from [Potworowski et al., 2012] to avoid transposition in the matrix formulation below.

Supplementary Fig. 16 shows error propagation maps for 1D regular grid of 12 electrodes. This corresponds conceptually to Fig. 7 for the setup from Fig. 2).

Fig. 17 shows an example of 3D kCSD source reconstruction with the new code. Files `small_srcs_3D_all.pdf` and `large_srcs_3D_all.pdf` show 100 examples each of 3D kCSD reconstructions of small and large sources. Similarly, files `small_srcs_all.pdf` and `large_srcs_all.pdf` show each 100 examples each of 2D kCSD reconstructions of small and large sources. The files are available at http://bit.ly/kCSD-supplementary.

Fig. 18 shows an example of skCSD reconstruction which corresponds to Fig. 8 from Cserpan et al. [2017]. The simulation, reconstruction and visualization are all reimplemented in Python.

Acknowledgments

The Python implementation of kCSD was started by Grzegorz Parka during Google Summer of Code project through the International Neuroinformatics Coordinating Facility. Jan Mąka implemented the first Python version of skCSD class. The study received funding from the Polish National Science Centre's grants (2013/08/W/NZ4/00691) and (2015/17/B/ST7/04123). The authors declare no conflict of interest.

References

Gian Nicola Angotzi, Fabio Boi, Aziliz Lecomte, Ermanno Miele, Mario Malerba, Stefano Zucca, Antonino Casile, and Luca Berdondini. Sinaps: An implantable active pixel sensor cmos-probe for simultaneous large-scale neural recordings. *Biosensors & bioelectronics*, 126:355–364, February 2019. ISSN 1873-4235. doi: 10.1016/j.bios.2018.10.032.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

Giorgio A. Ascoli. Mobilizing the base of neuroscience data: the case of neuronal morphologies. *Nat Rev Neurosci*, 7(4):318–324, Apr 2006. doi: 10.1038/nrn1885. URL `http://dx.doi.org/10.1038/nrn1885`.

Douglas J. Bakkum, Urs Frey, Milos Radivojevic, Thomas L. Russell, Jan Müller, Michele Fiscella, Hirokazu Takahashi, and Andreas Hierlemann. Tracking axonal action potential propagation on a high-density microelectrode array across hundreds of sites. *Nat Commun*, 4:2181, 2013. doi: 10.1038/ncomms3181. URL `http://dx.doi.org/10.1038/ncomms3181`.

Claude Bédard and Alain Destexhe. Generalized theory for current-source-density analysis in brain tissue. *Phys Rev E Stat Nonlin Soft Matter Phys*, 84(4 Pt 1):041909, Oct 2011.

Murat Belge, Misha E. Kilmer, and Eric L. Miller. Efficient determination of multiple regularization parameters in a generalized L-curve framework. *Inverse Problems*, 18(4): 1161–1183, 2002. ISSN 02665611. doi: 10.1088/0266-5611/18/4/314.

L. Berdondini, P. D. van der Wal, O. Guenat, N. F. de Rooij, M. Koudelka-Hep, P. Seitz, R. Kaufmann, P. Metzler, N. Blanc, and S. Rohr. High-density electrode array for imaging in vitro electrophysiological activity. *Biosens Bioelectron*, 21(1):167–174, Jul

2005. doi: 10.1016/j.bios.2004.08.011. URL http://dx.doi.org/10.1016/j.bios.2004.08.011.

György Buzsáki. Large-scale recording of neuronal ensembles. *Nat. Neurosci.*, 7(5):446–451, May 2004. doi: 10.1038/nn1233. URL http://dx.doi.org/10.1038/nn1233.

György Buzsáki, Costas A Anastassiou, and Christof Koch. The origin of extracellular fields and currents — EEG, ECoG, LFP and spikes. *Nat Rev Neurosci*, 13(6):407–420, Jun 2012. doi: 10.1038/nrn3241. URL http://dx.doi.org/10.1038/nrn3241.

J. Longina Castellanos, Susana Gómez, and Valia Guerra. The triangle method for finding the corner of the L-curve. *Applied Numerical Mathematics*, 43(4):359–373, 2002. ISSN 01689274. doi: 10.1016/S0168-9274(01)00179-9.

Dorottya Cserpan, Domokos Meszena, Lucia Wittner, Kinga Toth, Istvan Ulbert, Zoltan Somogyvari, and Daniel Krzysztof Wójcik. Revealing the distribution of transmembrane currents along the dendritic tree of a neuron from extracellular recordings. *eLife*, 6: e29384, 2017. doi: 10.7554/eLife.29384. URL https://elifesciences.org/articles/29384.

Gaute T. Einevoll, Christoph Kayser, Nikos K. Logothetis, and Stefano Panzeri. Modelling and analysis of local field potentials for studying the function of cortical circuits. *Nat Rev Neurosci*, 14(11):770–785, Nov 2013. doi: 10.1038/nrn3599. URL http://dx.doi.org/10.1038/nrn3599.

E. Ferrea, A. Maccione, L. Medrihan, T. Nieus, D. Ghezzi, P. Baldelli, F. Benfenati, and L. Berdondini. Large-scale, high-resolution electrophysiological imaging of field potentials in brain slices with microelectronic multielectrode arrays. *Frontiers in Neural Circuits*, 6:80, 2012. doi: 10.3389/fncir.2012.00080. URL http://dx.doi.org/10.3389/fncir.2012.00080.

U. Frey, U. Egert, F. Heer, S. Hafizovic, and A. Hierlemann. Microelectronic system for high-resolution mapping of extracellular electric fields applied to brain slices. *Biosens Bioelectron*, 24(7):2191–2198, Mar 2009. doi: 10.1016/j.bios.2008.11.028. URL http://dx.doi.org/10.1016/j.bios.2008.11.028.

Takakuni Goto, Rieko Hatanaka, Takeshi Ogawa, Akira Sumiyoshi, Jorge Riera, and Ryuta Kawashima. An evaluation of the conductivity profile in the somatosensory barrel cortex of wistar rats. *Journal of neurophysiology*, 104:3388–3412, December 2010. ISSN 1522-1598. doi: 10.1152/jn.00122.2010.

Sergey L Gratiy, Geir Halnes, Daniel Denman, Michael J Hawrylycz, Christof Koch, Gaute T Einevoll, and Costas A Anastassiou. From maxwell's equations to the theory of current-source density analysis. *The European journal of neuroscience*, 45:1013–1023, April 2017. ISSN 1460-9568. doi: 10.1111/ejn.13534.

Helena Głąbska, Jan Potworowski, Szymon Łęski, and Daniel K Wójcik. Independent components of neural activity carry information on individual populations. *PloS one*, 9:e105071, 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0105071.

P. Hansen. *Discrete Inverse Problems*. Society for Industrial and Applied Mathematics, 2010. doi: 10.1137/1.9780898718836. URL http://epubs.siam.org/doi/abs/10.1137/1.9780898718836.

Per Christian Hansen. Analysis of Discrete Ill-Posed Problems by Means of the L-Curve. *SIAM Review*, 34(4):561–580, 1992. ISSN 0036-1445. doi: 10.1137/1034115. URL `http://epubs.siam.org/doi/10.1137/1034115`.

Per Christian Hansen, Toke Jensen, and Giuseppe Rodriguez. An adaptive pruning algorithm for the discrete L-curve criterion. *Journal of Computational and Applied Mathematics*, 198:483–492, 01 2007. doi: 10.1016/j.cam.2005.09.026.

Paweł Hottowy, Andrzej Skoczeń, Deborah E Gunning, Sergei Kachiguine, Keith Mathieson, Alexander Sher, Piotr Wiącek, Alan M Litke, and Władysław Dąbrowski. Properties and application of a multichannel integrated circuit for low-artifact, patterned electrical stimulation of neural tissue. *Journal of neural engineering*, 9(6):066005, 2012.

Mark J. Hunt, Monika Falinska, Szymon Łęski, Daniel K. Wójcik, and Stefan Kasicki. Differential effects produced by ketamine on oscillatory activity recorded in the rat hippocampus, dorsal striatum and nucleus accumbens. *J Psychopharmacol*, 25(6):808–821, Jun 2011. doi: 10.1177/0269881110362126. URL `http://dx.doi.org/10.1177/0269881110362126`.

James J Jun, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, Costas A Anastassiou, Alexandru Andrei, Çağatay Aydın, Mladen Barbic, Timothy J Blanche, Vincent Bonin, João Couto, Barundeb Dutta, Sergey L Gratiy, Diego A Gutnisky, Michael Häusser, Bill Karsh, Peter Ledochowitsch, Carolina Mora Lopez, Catalin Mitelut, Silke Musa, Michael Okun, Marius Pachitariu, Jan Putzeys, P Dylan Rich, Cyrille Rossant, Wei-Lung Sun, Karel Svoboda, Matteo Carandini, Kenneth D Harris, Christof Koch, John O'Keefe, and Timothy D Harris. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551: 232–236, November 2017. ISSN 1476-4687. doi: 10.1038/nature24636.

George Kimeldorf and Grace Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971. ISSN 0022-247X. doi: 10.1016/0022-247x(71)90184-3.

Jee-Hyun Kong, Daniel R. Fish, Rebecca L. Rockhill, and Richard H. Masland. Diversity of ganglion cells in the mouse retina: unsupervised morphological classification and its limits. *J Comp Neurol*, 489(3):293–310, Aug 2005. doi: 10.1002/cne.20631. URL `http://dx.doi.org/10.1002/cne.20631`.

Pascal Kropf and Amir Shmuel. 1d current source density (csd) estimation in inverse theory: A unified framework for higher-order spectral regularization of quadrature and expansion-type csd methods. *Neural computation*, 28:1305–1355, July 2016. ISSN 1530-888X. doi: 10.1162/NECO_a_00846.

M Kwan. Graphic programming using odd or even points. *Chinese Math*, 1:110, 1962.

Henrik Lindén, Espen Hagen, Szymon Łęski, Eivind S. Norheim, Klas H. Pettersen, and Gaute T. Einevoll. Lfpy: a tool for biophysical simulation of extracellular potentials generated by detailed model neurons. *Front Neuroinform*, 7:41, 2013. doi: 10.3389/fninf.2013.00041. URL `http://dx.doi.org/10.3389/fninf.2013.00041`.

Henrik Lindén, Tom Tetzlaff, Tobias C. Potjans, Klas H. Pettersen, Sonja Grün, Markus Diesmann, and Gaute T. Einevoll. Modeling the spatial reach of the LFP. *Neuron*, 72 (5):859–872, Dec 2011. doi: 10.1016/j.neuron.2011.11.006. URL `http://dx.doi.org/10.1016/j.neuron.2011.11.006`.

J. W. H. Meijs, M. J. Peters, H. B. K. Boom, and F. H. Lopes da Silva. Relative influence of model assumptions and measurement procedures in the analysis of the meg. *Medical and Biological Engineering and Computing*, 26(2):136–142, Mar 1988. ISSN 1741-0444. doi: 10.1007/BF02442255. URL `https://doi.org/10.1007/BF02442255`.

Torbjørn V Ness, Chaitanya Chintaluri, Jan Potworowski, Szymon Łęski, Helena Głąbska, Daniel K Wójcik, and Gaute T Einevoll. Modelling and analysis of electrical potentials recorded in microelectrode arrays (meas). *Neuroinformatics*, 13(4):403–426, 2015.

C. Nicholson. Theoretical analysis of field potentials in anisotropic ensembles of neuronal elements. *IEEE Trans Biomed Eng*, 20(4):278–288, Jul 1973. doi: 10.1109/TBME.1973.324192. URL `http://dx.doi.org/10.1109/TBME.1973.324192`.

C. Nicholson and J. A. Freeman. Theory of current source-density analysis and determination of conductivity tensor for anuran cerebellum. *J Neurophysiol*, 38(2):356–368, Mar 1975.

Solveig Næss, Chaitanya Chintaluri, Torbjørn V. Ness, Anders M. Dale, Gaute T. Einevoll, and Daniel K. Wójcik. Corrected Four-Sphere Head Model for EEG Signals. *Frontiers in human neuroscience*, 11:490, 2017. ISSN 1662-5161. doi: 10.3389/fnhum.2017.00490.

Alan V. Oppenheim, Alan S. Willsky, and S. Hamid Nawab. Signals & Systems. *Prentice-Hall, Inc., Upper Saddle River, NJ, USA*, 18:19–21, 1996.

Klas H Pettersen, Anna Devor, Istvan Ulbert, Anders M Dale, and Gaute T Einevoll. Current-source density estimation based on inversion of electrostatic forward solution: effects of finite extent of neuronal activity and conductivity discontinuities. *J Neurosci Methods*, 154(1-2):116–133, Jun 2006. doi: 10.1016/j.jneumeth.2005.12.005. URL `http://dx.doi.org/10.1016/j.jneumeth.2005.12.005`.

Walter Pitts. Investigations on synaptic transmission. In *Cybernetics, Trans. 9th Conf. Josiah Macy, New York*, pages 159–162, 1952.

Jan Potworowski, Wit Jakuczun, Szymon Łęski, and Daniel K Wójcik. Kernel current source density method. *Neural Comput*, 24(2):541–575, Feb 2012. doi: 10.1162/NECO_a_00236. URL `http://dx.doi.org/10.1162/NECO_a_00236`.

Project Jupyter, Matthias Bussonnier, Jessica Forde, Jeremy Freeman, Brian Granger, Tim Head, ChrisHoldgraf, Kyle Kelley, Gladys Nalvarte, Andrew Osheroff, M. Pacer, Yuvi Panda, Fernando Perez, Benjamin Ragan-Kelley, and Carol Willing. Binder 2.0 - Reproducible, Interactive, Sharable Environments for Science at Scale. In *Proceedings of the 17th Python in Science Conference.*, 2018. doi: 10.25080/Majora-4af1f417-011.

P Rappelsberger, H Pockberger, and H Petsche. Current source density analysis: methods and application to simultaneously recorded field potentials of the rabbit's visual cortex. *Pflugers Archiv : European journal of physiology*, 389:159–170, January 1981. ISSN 0031-6768.

Jorge J. Riera, Takeshi Ogawa, Takakuni Goto, Akira Sumiyoshi, Hiroi Nonaka, Alan Evans, Hiroyoshi Miyakawa, and Ryuta Kawashima. Pitfalls in the dipolar model for the neocortical EEG sources. *Journal of neurophysiology*, 108:956–975, August 2012. ISSN 1522-1598. doi: 10.1152/jn.00098.2011.

Charles F Stevens. *Neurophysiology: A primer*. Wiley New York, 1966.

G Vaknin, P G DiScenna, and T J Teyler. A method for calculating current source density (csd) analysis without resorting to recording sites outside the sampling volume. *Journal of neuroscience methods*, 24:131–135, June 1988. ISSN 0165-0270.

Daniel K Wójcik and Szymon Łęski. Current source density reconstruction from incomplete data. *Neural Comput*, 22(1):48–60, Jan 2010. doi: 10.1162/neco.2009.07-08-831. URL `http://dx.doi.org/10.1162/neco.2009.07-08-831`.

Szymon Łęski, Daniel K Wójcik, Joanna Tereszczuk, Daniel A Swiejkowski, Ewa Kublik, and Andrzej Wróbel. Inverse Current-Source Density method in 3D: reconstruction fidelity, boundary effects, and influence of distant sources. *Neuroinformatics*, 5(4): 207–222, 2007. doi: 10.1007/s12021-007-9000-z. URL `http://dx.doi.org/10.1007/s12021-007-9000-z`.

Szymon Łęski, Ewa Kublik, Daniel A. Swiejkowski, Andrzej Wróbel, and Daniel K. Wójcik. Extracting functional components of neural dynamics with Independent Component Analysis and inverse Current Source Density. *Journal of computational neuroscience*, 29:459–473, December 2010. ISSN 1573-6873. doi: 10.1007/s10827-009-0203-1.

Szymon Łęski, Klas H Pettersen, Beth Tunstall, Gaute T Einevoll, John Gigg, and Daniel K Wójcik. Inverse Current Source Density method in two dimensions: Inferring neural activation from multielectrode recordings. *Neuroinformatics*, 9(4):401–425, Dec 2011. doi: 10.1007/s12021-011-9111-4. URL `http://dx.doi.org/10.1007/s12021-011-9111-4`.

Szymon Łęski, Henrik Lindén, Tom Tetzlaff, Klas H. Pettersen, and Gaute T. Einevoll. Frequency dependence of signal power and spatial reach of the local field potential. *PLoS Comput Biol*, 9(7):e1003137, Jul 2013. doi: 10.1371/journal.pcbi.1003137. URL `http://dx.doi.org/10.1371/journal.pcbi.1003137`.

S1 Fig

Eigensources in a 2D example.

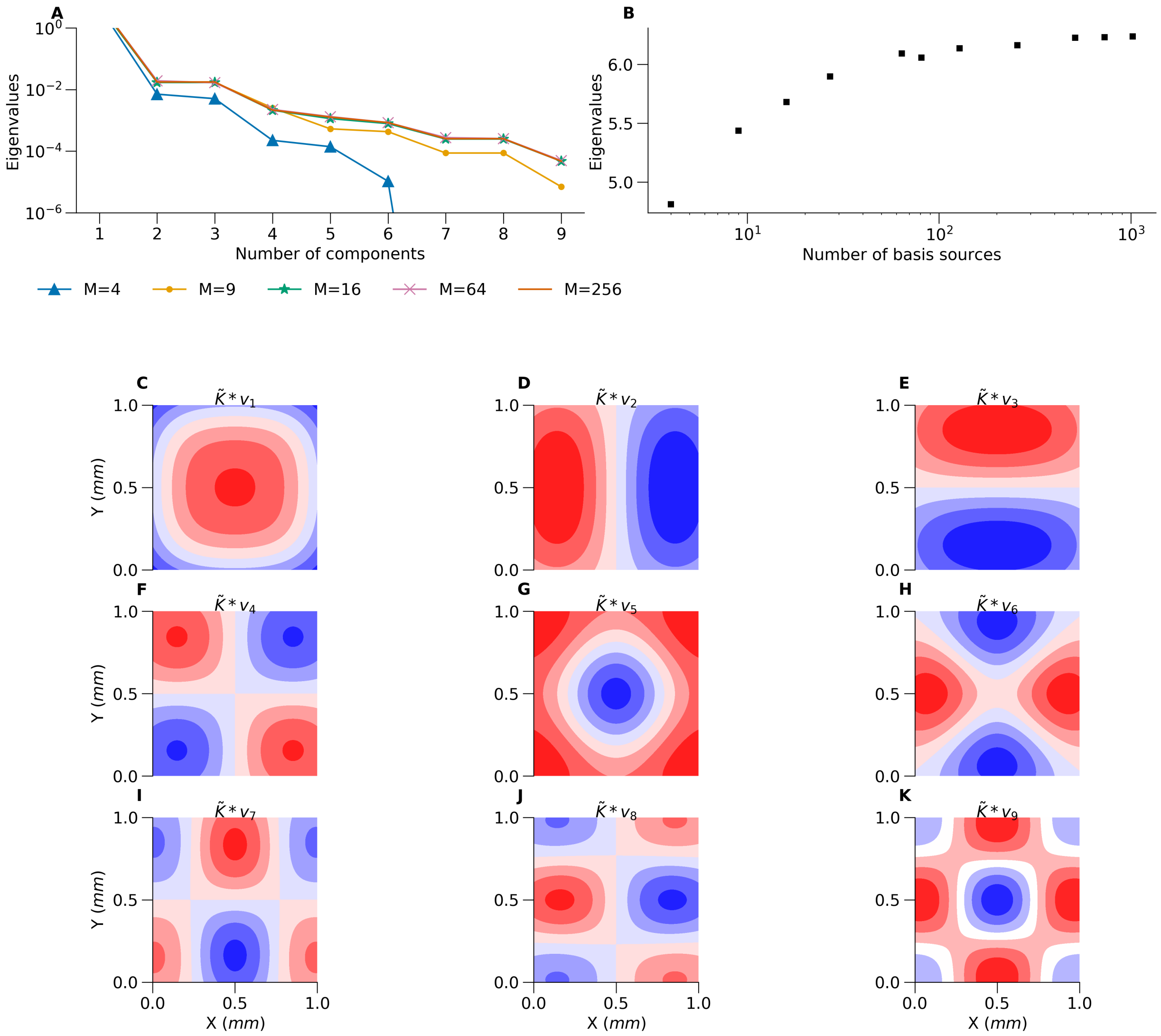

Figure 15: Demonstration of spectral properties of kCSD method for simple 2D case with 9 regularly distributed electrodes in the square $[0, 1] \times [0, 1]$ (from Fig. 7). A) Eigenvalues obtained from the decomposition of $\mathbf{K}$ for different number of basis sources (M=4, 9, 16, 64, 256), B) Shows the first eigenvalue in the function of growing number of basis, C–K) Shows 9 eigensources (products of $\widetilde{\mathbf{K}}$ and all the eigenvectors), which combine to produce the inverse solution in kCSD method.

S2 Fig 1020

Error propagation maps in a 1D example. 1021

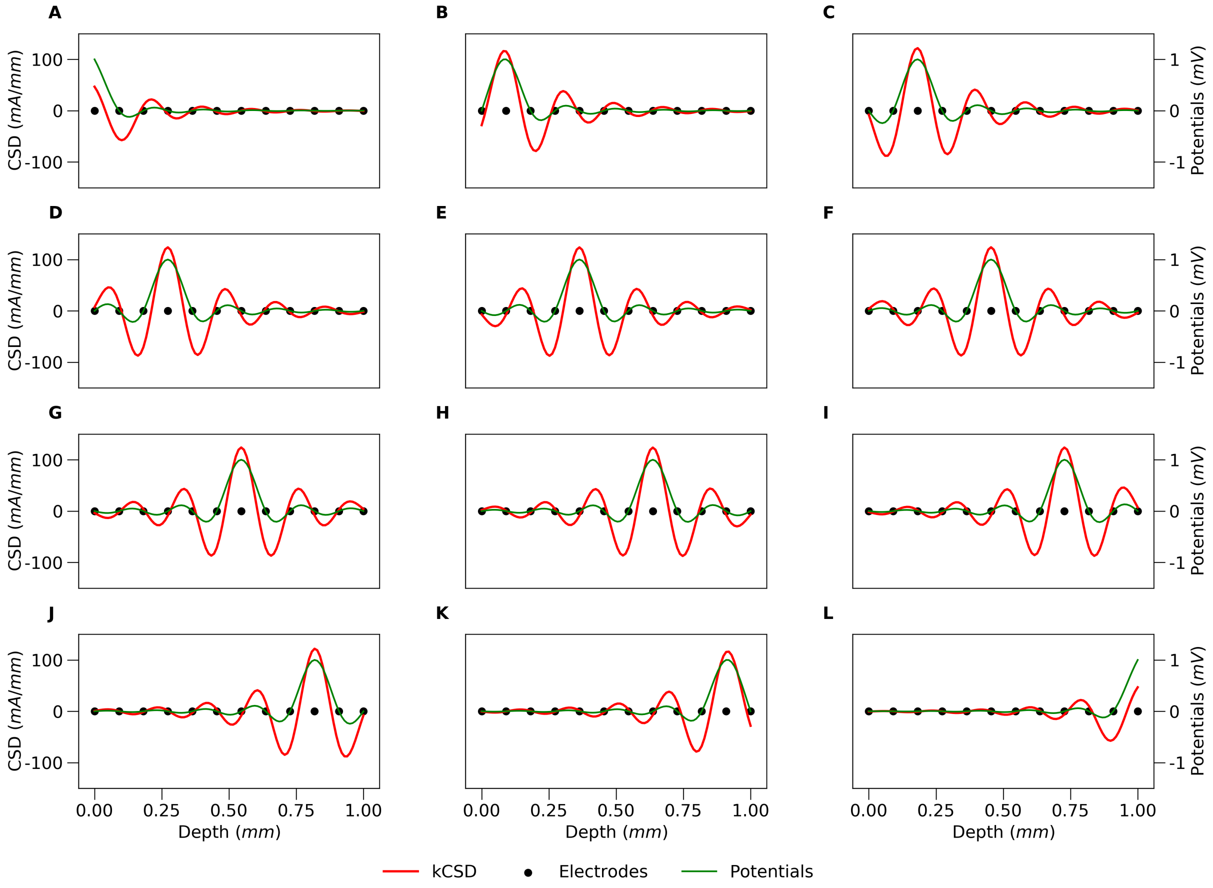

Figure 16: **Error propagation maps** for 1D regular grid of 12 electrodes (same as in Fig. 2). Every panel represents the CSD contribution (red line) of the potential measured at the corresponding electrode, for which the potential is 1 (green line).

S3 Fig 1022

An example of 3D source reconstruction. 1023

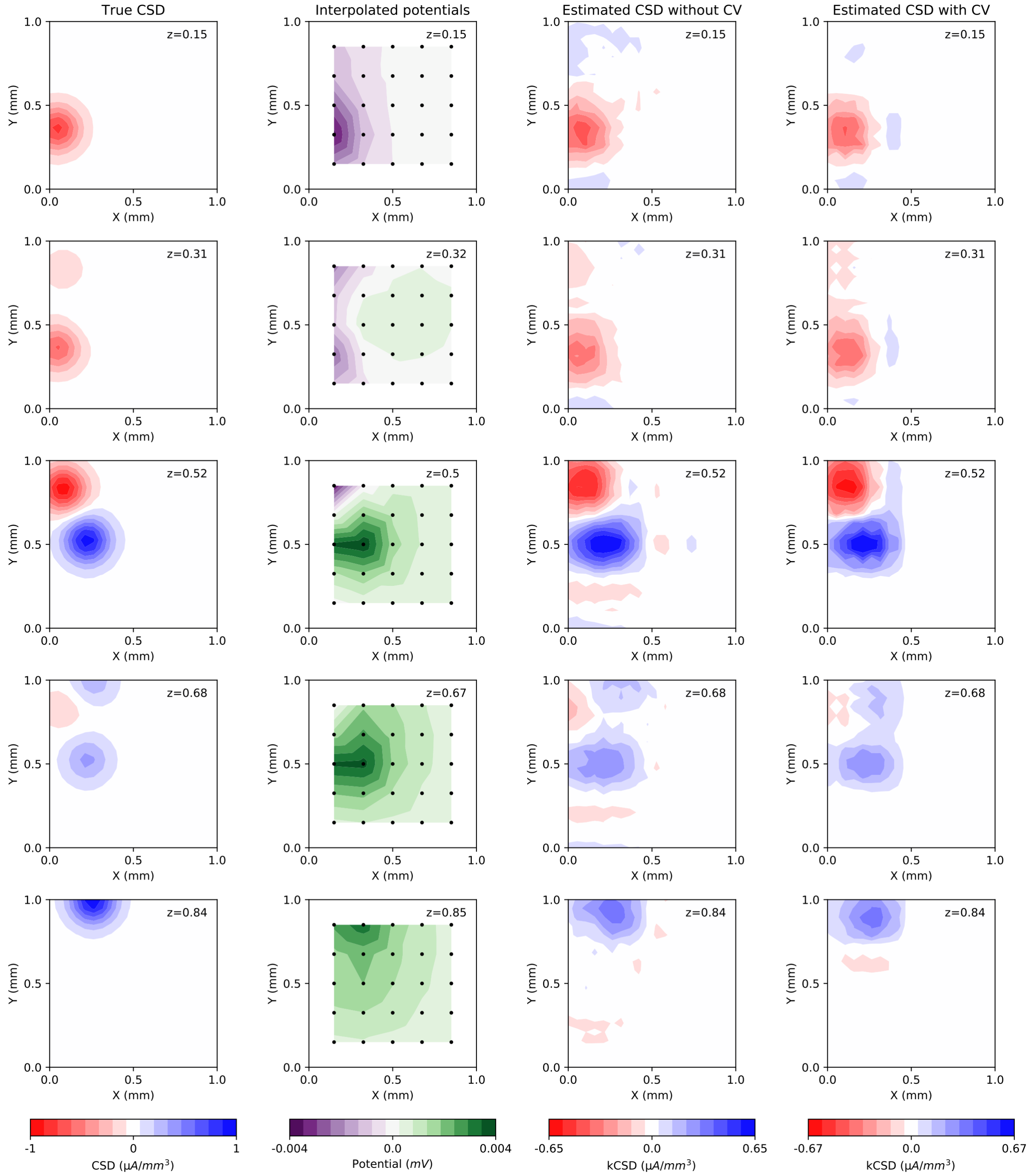

Figure 17: **An example of 3D kCSD source reconstruction**. Each column shows five consecutive parallel cuts through a box of size 1. A) Ground truth for the CSD seed of 16. B) Estimated potential; black dots indicate electrodes where potential is colected for further reconstruction. C) 3D kCSD reconstruction from the measured potentials, $\lambda = 0$. D) 3D kCSD reconstruction with cross-validation.

S4 Fig 1024

An example of skCSD source reconstruction. 1025

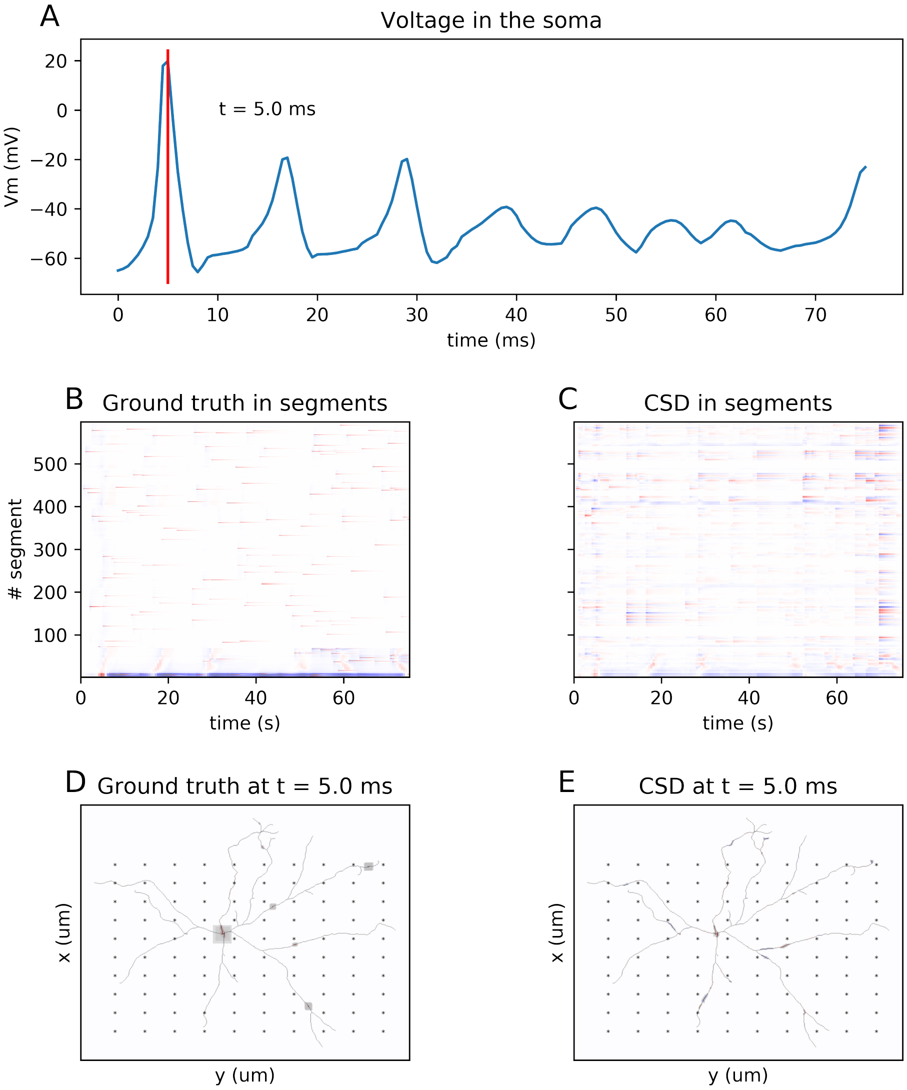

Figure 18: **An example of skCSD reconstruction** Cserpan et al. [2017] of somatic current injection together with random synaptic input patterns for a retinal ganglion cell model. A) Somatic membrane potential. B) Current density and its C) skCSD reconstruction in the segment space. Projection of D) ground truth and E) skCSD reconstruction on the neuron's morphology at 5 s of the simulation. We simulated a multi-compartmental model of a mouse retinal ganglion cell (morphology Kong et al. [2005] obtained from NeuroMorpho.Org Ascoli [2006]) with Hodgkin-Huxley sodium, potassium, and leakage channels in the soma (hh mechanism) in NEURON simulation environment. For calculation of the measured extracellular potentials we used LFPy package Lindén et al. [2013]. The model neuron was stimulated by an injection of oscillatory current to the soma (with frequency of 24.5 1/ms and amplitude of 3.6 nA) together with random synaptic inputs (weight of 0.04 μS) to the dendritic tree. The activity of the model neuron was measured by a rectangular grid of 100 electrodes (10 × 10, -400 $\mu m \times 400\mu m$). The figure corresponds to Fig. 8 from [Cserpan et al., 2017].