Modern Statistical/Machine Learning Techniques for Bio/Neuro-imaging Applications

Ruoxi Sun

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2019

ABSTRACT

Modern Statistical/Machine Learning Techniques for Bio/Neuro-imaging applications

Ruoxi Sun

Developments in modern bio-imaging techniques have allowed the routine collection of a vast amount of data from various techniques. The challenges lie in how to build accurate and efficient models to draw conclusions from the data and facilitate scientific discoveries. Fortunately, recent advances in statistics, machine learning, and deep learning provide valuable tools. This thesis describes some of our efforts to build scalable Bayesian models for four bio-imaging applications: (1) Stochastic Optical Reconstruction Microscopy (STORM) Imaging, (2) particle tracking, (3) voltage smoothing, (4) detect color-labeled neurons in c elegans and assign identity to the detections.

Contents

Acknowledgements

I would like to express great gratitude to my advisor Liam Paninski. Without him, the thesis will not ever happen. I deeply admire his brilliance and diligence as a researcher; and kindness and humbleness as a human. During these years, he becomes my role model. It is a great fortunate that I can work with him. Here I would like to say thank you to Liam from the following perspectives.

Firstly, Liam opened the door to a fantastic research area to me. Under his guidance, I developed strong interests. I had lots of fun in every day of Ph.D. studies. I appreciate his intriguing, encouraging and helpful research conversations. Secondly, I appreciate his efficient mentorship, which brought up the best out of me. I joined his lab with an experimental biology background. He raised me up from zero to a Ph.D. candidate in computational side. Also, I appreciate his open-mindedness and willing to allow me to join his group. Thirdly, he supported me emotionally. He is very caring, understanding, and supportive. He is always by my side when I met obstacles or felt stuck/down. Lastly, I thank him for his generosity financially. I wish he continue his success in research, and publish lots of exciting work, and mentor more and more professors from his group! End with lyrics of "you raise me up": "You raise me up to more than I can be".

I would like to thank my very talented and supportive collaborators. Without their

help and understanding, the projects will not be done in such a joyful and efficient way. I had the fortune to work with Scott Linderman on voltage imaging. It is his intelligence, brilliance, and support on math and code that this project can be in shape in two months. I had the fortune to work with Erdem Varol, Gonzalo Mena and Amin Nejatbakhsh, and Eviatar Yemini. It is their creativity and hard working that this project can move forward smoothly. It is their understanding that supports me through the final stage of Ph.D.

In addition, I had the fortune to be surrounded by so many excellent faculty, postdoc and graduate students. I would like to thank Prof John Cunningham for discussions on my first project. I would like to thank Xinyi Deng for helpful discussions and company – sharing happiness and upsets. I would like to thank Xuexin Wei, for helpful discussions on Reinforcement learning project. I thank Pengcheng Zhou, Shreya Saxena, and many others.

I would like to thank Prof Dimitris Anastassiou to embrace me generously for the uncertain beginning stage of PhD studies, and later support me to pursue interests. I thank Prof Ron Prywes, Sarah Kim, Allison Ong, Anthony Cruz for supporting me administrative perspectives, saving me from logistics and allowing me to focus on research. I thank Zuckerman Institute for brand new and modern building, providing me with great working environment.

Last but not least, I would like to thank my parents for unconditional love and support me from ups and downs. I particularly would like to thank my grandmother, she listened to my happiness and worries during entire graduate studies, provided with very wise advice and believed in me always. This thesis is dedicated to her.

To my grandmother.

Ms. Yifang Guo

Introduction

Modern advances in bio-imaging and neuro-imaging techniques have opened tremendous opportunities to investigate scientific problems inaccessible before. The insights provided by the novel techniques have advanced scientific domains. The novel technologies improve upon existing ones in different perspectives. Some can provide higher resolution, such as super-resolution imaging; some offer more insightful representation of recording, such as voltage imaging of neurons, as oppose to traditional calcium imaging; others can be unprecedented technologies, such as NeuroPAL, which designs multicolor C. elegans strain that can be used to resolve unique neural identities in whole-brain images.

Despite that the novel technologies are exciting, statistical approaches are in great need to transform the rich imaging information into accessible and interpretive knowledge. The computational imaging community has adopted techniques from mathematics, statistics, machine learning, and deep learning to introduce novel methodologies targeting the improvement of our understanding of various new data.

This thesis describes statistical approaches for four novel biological or neural technologies and demonstrates their performances. Concretely, we propose scalable Bayesian approaches to either capture the relevant structure of the data (e.g. data generation process) or model the uncertainty. Our promising results demonstrate the potential of our

methods to advance bio/neuro-imaging analysis to go beyond the limit of its experimental capacity. We arrange the thesis as follows:

1. **Super resolution imaging, (Chapter 2)**

   Resolution of light microscope suffers from light diffraction– a single point on specimen leads to a blurry projection on the microscope. Super-resolution data, Stochastic Optical Reconstruction Microscopy (STORM), obtains high resolution by collecting hundreds or thousands of random subsets of the specimen image (stochastically activation of a random subset of fluorophores that are used to label the specimen). Traditionally, STORM analysis is performed by independently sparse deconvolution on each of the individual images to resolve the centers of blurry projections, and adding estimates of all frames. However, this approach is sub-optimal, as the sparse activations are not independent, and they share information. They are samples drawn from the same distribution – original specimen image. Therefore, a natural way to take advantage of shared information and the relevant structure of the data is a Bayesian framework. The objective function of the Bayesian framework mimics the data generation, which is from stochastic activation of subsets of fluorophores to noisy emissions. Then we use variational inference to resolve the center of fluorophores as well as the model parameters.

2. **Particle tracking data (Chapter 3)**

   Tracking objects is a classic problem with broad applications in science, machine learning, and computer vision. For example, in science, we track neurons, virus, or nano-particles on a microscope; in non-science, we track traffic or people. In our ap-

plication, particles are doing Brownian motion, and frequently running into each other (collisions). Particles appear and disappear at random time. Most existing methods are deterministic, which output a single best estimate, e.g., maximum likelihood estimator (MLE), of the true tracks. However, due to the limited resolution of the microscope, particles are indistinguishable in shape, leading to impossible to deterministically assign identities of particles after collisions (particles running into each other). For example, two particles move towards each other and then move apart. After collisions, we can have 50% chance to have two particles cross over and 50% chance to bounce back. Therefore, we propose a probabilistic approach, which infers the Bayesian posterior distribution of estimates of tracks given noisy observations using an iterative neural networks sampling approach. Drawing samples from the inferred posterior distribution, we are able to provide multiple valid solutions given the observed data.

3. **Voltage smoothing data (Chapter 4)**

   The subthreshold neuronal activity provides a rich representation of the properties of a single cell's physiology. The access to sub-threshold activity reveals the intrinsic biophysical properties, such as membrane and ion channel parameters; circuit-level properties (e.g. synaptic connectivity), and neural coding (e.g. receptive fields). Recent advances in optical voltage sensors have brought us closer to a critical goal in cellular neuroscience: imaging the full spatiotemporal voltage on a dendritic tree. However, current sensors and imaging approaches still face significant limitations in SNR and sampling frequency; therefore, statistical denoising methods remain critical for understanding single-trial spatiotemporal dendritic voltage dynamics.

Previous denoising approaches were either based on an inadequate linear voltage model or scaled poorly to large trees. Here we introduce a scalable fully Bayesian approach. We develop a generative nonlinear model that requires few parameters per dendritic compartment but is nonetheless flexible enough to sample realistic spatiotemporal data. The model captures potentially different dynamics in each compartment and leverages biophysical knowledge to constrain intra- and inter-compartmental dynamics. We obtain a full posterior distribution over spatiotemporal voltage via an efficient augmented block-Gibbs sampling algorithm. The nonlinear smoother model outperforms previously developed linear methods, and scales to much larger systems than previous methods based on sequential Monte Carlo approaches.

4. **NeuroPAL data (Chapter 5)**

   The major challenge in exploring neural circuitry is to identify neuronal patterns of activity, gene expression, and mutant effects. Our collaborators, Eviatar Yemini and Hobert Oliver developed a multicolor C. elegans strain, called the NeuroPAL (a Neuronal Polychromatic Atlas of Landmarks), to resolve unique neural identities in whole-brain images. An identical color map is shared among all NeuroPAL worms, permitting a complete, unambiguous determination of individual neuron names. NeuroPAL worms were applied to functional connectomics (gustatory/olfactory neurons) and molecular connectomics (metabotropic neurons). Here, we proposed a software for fully-automated neural identification for NeuroPAL.

Chapter 1

Super Resolution Microscope

Super-resolution microscopy methods have become essential tools in biology, opening up a variety of new questions that were previously inaccessible with standard light microscopy methods. In this paper we develop new Bayesian image processing methods that extend the reach of super-resolution microscopy even further. Our method couples variational inference techniques with a data summarization based on Laplace approximation to ensure computational scalability. Our formulation makes it straightforward to incorporate prior information about the underlying sample to further improve accuracy. The proposed method obtains dramatic resolution improvements over previous methods while retaining computational tractability.

This is a joint work with Evan Archer, Liam Paninski. This work was built on Evan Archer's previous work. This work was published: Ruoxi Sun, Evan Archer, Liam Paninski. Scalable variational inference for super resolution microscopy, International Conference on Artificial Intelligence and Statistics (AISTATS) 2017. `http://biorxiv.org/content/early/2016/11/19/081703`. Code is available: `https://github.com/SunRuoxi/vEM`

1.1 Introduction

Super-resolution microscopy techniques, such as STORM (Rust, Bates, and Zhuang 2006), PALM (Betzig et al. 2006), or fPALM (Hess, Girirajan, and Mason 2006) imaging have quickly become essential tools in biology. These methods overcome the light diffraction barrier of traditional microscopy, thus enabling researchers to ask questions previously considered inaccessible (as a measure of impact, developers of these methods were awarded the Nobel prize in Chemistry in 2014). Given a sample treated with a fluorescent dye, the basic strategy is to stochastically activate fluorophores at a low rate, guaranteeing that only a sparse subset are activated at a given time. By repeatedly imaging the sample we obtain a movie wherein each frame reflects a random, sparse set of fluorophore activations. Then we exploit the sparsity of activations within each frame to localize the positions of the activated fluorophores; aggregating a long sequence of such point localizations then yields a super-resolved image (Fig. 1.1).

Many methods have been proposed to expand upon this basic idea, focusing upon improving localization performance within each individual frame (Sage et al. 2015). For sparse recovery of a single frame, several modern techniques take a compressed sensing approach that exploits the true sparsity of the underlying fluorophore activations; these techniques result in a formulation as a sparse deconvolution problem (Min et al. 2014; Zhu et al. 2012), providing scalable, fairly accurate reconstructions.

The critical message of this paper is that such standard approaches are sub-optimal because each frame is reconstructed independently, thereby discarding information that should be shared across frames. Intuitively, given $N-1$ reconstructed frames, we should

have a good deal of prior information about the locations of fluorophores on the N-th frame, and ignoring this information will in general lead to highly suboptimal estimates. (This basic point has been made previously, e.g. by (Cox et al. 2012; Mukamel, Babcock, and Zhuang 2012); we will discuss this work further below.)

Here, we propose a scalable Bayesian approach that properly pools information across frames and can also incorporate prior information about the image, leading to dramatic resolution improvements over previous methods while retaining computational tractability.

1.2 Model

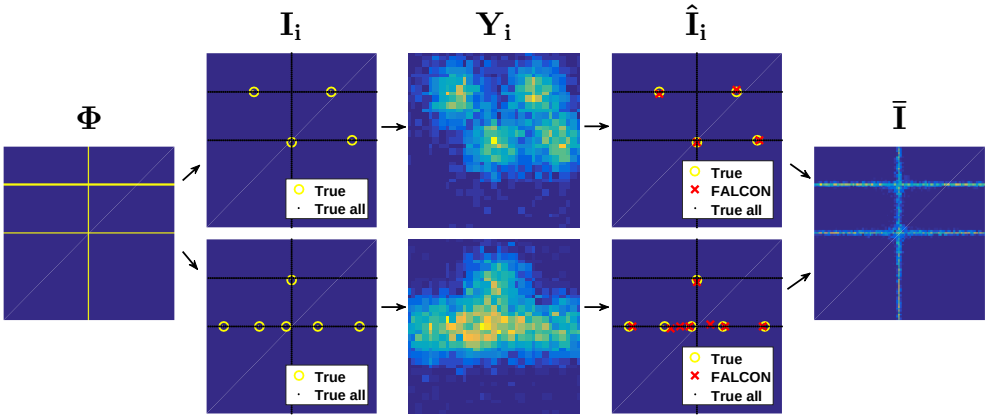

Figure 1.1: Overview of standard super-resolution microscopy. **Column 1**: The true fluorophore density matrix Φ. **Column 2**: I_i indicates the sparse subset of fluorophores (yellow circles) activated on frame i from Φ, which is also plotted as background (black dots); two independent sample frames shown here (top and bottom). **Column 3**: Y_i are the observed camera images on these two frames, formed by blurring and downsampling the corresponding I_i and adding Poisson noise. **Column 4**: $\hat{I}_i$ indicate the estimated locations of the active fluorophores on each frame, with the true I_i shown for comparison. The FALCON method (Min et al. 2014) was used to compute the estimates here; note that estimator performance decreases in regions where the "bumps" in Y_i overlap significantly. **Column 5**: The standard approach to estimate $p\Phi$ is to simply average over multiple inferred frames, $\bar{I} = \frac{1}{N}\sum_i^N \hat{I}_i$.

At each frame i we observe an $L \times L$ fluorescence image $Y_i \in \mathbb{R}_+^{L \times L}$, and collect the sequences of N observed frames into the movie $Y = \{Y_i : i \in \{1, \ldots, N\}\}$. We model each observed frame Y_i as a noisy, blurred, low-resolution image,

$$Y_i \sim \text{Poisson}(AI_i); \tag{1.1}$$

here A is a matrix implementing convolution with a known point-spread function (PSF), scaling by the mean photon emission rate per fluorophore, and spatial downsampling; the high-resolution image $I_i \in \mathbb{R}_+^{D \times D}$ is a sparse matrix, zero except at the locations of fluorophores activated on frame i. In this application $L < D$. Below we will use the sparse representation (m_i, F_i) for I_i: m_i denotes the number of active fluorophores in I_i and $F_i \in \mathbb{R}^{2m_i}$ denotes the vector of xy positions of these fluorophores. Note that multiple fluorophores can be active at the same location, so the entries of I_i are nonnegative integers; it is straightforward to extend our methods to the case that I_i can take arbitrary nonnegative real values, but we will suppress this case here for notational simplicity.

At each high-resolution pixel position (x, y), we model the activation of fluorophores by an inhomogeneous Poisson process with rate λ_{xy},

$$I_{i,xy} \sim \text{Poisson}(\lambda_{xy}), \tag{1.2}$$

$$\lambda_{xy} = p\Phi_{xy}, \tag{1.3}$$

and p is a scalar (typically under at least partial experimental control) that sets the fluorophore emission rate. The matrix $\Phi \in \mathbb{R}_+^{D \times D}$ specifies the density of fluorophores at

each pixel location, and is the main object we aim to estimate; since λ and Φ are related by a constant (p), we will develop the inference methods below in terms of λ, as this leads to slightly simpler algebra.

This model can be extended to 3D (Babcock, Sigal, and Zhuang 2012; Huang et al. 2008) and/or multispectral imaging (Bates et al. 2012), but for simplicity here we focus on 2D single-color imaging.

The above definitions lead to the joint probability distribution

$$P(Y, F, m, \lambda) \propto \prod_{i}^{N} \left\{ P(Y_i|F_i, m_i) P(F_i, m_i|\lambda) \right\} P(\lambda), \tag{1.4}$$

where m and F collect the N scalars m_i and vectors F_i, respectively; $P(\lambda)$ is a prior distribution on λ;

$$P(Y_i|F_i, m_i) = P(Y_i|I_i) = \prod_{x=1}^{L} \prod_{y=1}^{L} e^{-[AI_i]_{xy}} \frac{[AI_i]_{xy}^{Y_{i,xy}}}{Y_{i,xy}!}, \tag{1.5}$$

where we have used the equivalence between I_i and (m_i, F_i), and

$$P(F_i, m_i|\lambda) = P(F_i|\lambda, m_i) P(m_i|\lambda) \tag{1.6}$$

$$= \prod_{j=1}^{m_i} \left\{ \frac{\lambda_{F_i^j}}{\eta} \right\} \text{Poisson}(m_i|\eta), \tag{1.7}$$

where F_i^j denotes the xy position of the j-th active fluorophore in frame i, $\lambda_{F_i^j}$ is the value of the 2D function λ at location F_i^j, and we have abbreviated the normalizer $\eta = \sum_{x=1}^{D} \sum_{y=1}^{D} \lambda_{xy}$.

1.3 Inference

Now that the model and likelihoods have been defined, we can proceed to develop our estimator for the underlying fluorophore density image λ. We take a Bayesian approach, which requires that we approximate the posterior distributions of the unknown quantities (m_i, F_i) given the observed data Y_i. (Approximation methods are required here since this is a non-conjugate latent variable model; we cannot analytically integrate out the I_i variables.) A number of such approximation methods are available; for example, (Picardo et al. 2016) recently developed MCMC methods to perform Bayesian inference in a similar model. However, these methods do not scale to the cases of interest here, where the number of frames N and pixels (D^2 and L^2) are often quite large.

Therefore we have developed a variational expectation-maximization (vEM) (Blei, Kucukelbir, and McAuliffe 2016) approximate inference approach. As is standard, we need to choose a variational family of distributions q (these distributions will be used to approximate the true posterior), then write down the "evidence lower bound" (ELBO; this is a function of q and other model parameters), and then develop methods for tractably ascending the ELBO.

The most standard choice of q here (a fully factorized distribution over all latent variables, i.e., the activations I_i together with λ) does not lead to a scalable inference method, due to the very high dimensionality of $\{I_i\}$; in addition, this vanilla variational approximation is poor here because of strong posterior correlations between adjacent pixels in the I_i images. Instead, we exploit the sparse representation (m_i, F_i) for I_i: a more effective approach for approximating $p(F, m|Y, \lambda)$ was to use a simple point estimate for m

(discussed below) and then, conditionally on m, a factorized ("mean-field") approximation for $p(F|m, Y, \lambda)$. Thus we approximate

$$P(F|\lambda, Y, m) \approx q(F) = \prod_{i}^{N} \prod_{j}^{m_i} q_{ij}(F_i^j). \tag{1.8}$$

We have factorized across frames i and active fluorophores j within each frame; here each $q_{ij} \in \mathbb{R}_+^{D \times D}$ is a probability density on the $D \times D$ grid that summarizes our approximate posterior beliefs about the fluorophore location F_i^j. In practice each q_{ij} will be extremely sparse, with very compact support, as we will discuss further below (Fig 1.2E).

The ELBO is given by:

$$\mathcal{L}(\lambda, q(F)) = \int q(F) \ln \frac{P(F, \lambda, Y|m)}{q(F)} dF \tag{1.9}$$

Our goal is to maximize $\mathcal{L}(\lambda, q(F))$ with respect to the distributions q_{ij} and image λ.

We will use a coordinate-ascent approach in which we update one q_{ij} or λ at a time; as discussed below, after one more approximation each update step can be computed cheaply (and parallelizes easily), and empirically only a few coordinate sweeps are necessary for convergence to a local optimum.

Laplace Approximation

Computing each q_{ij} update directly requires the computation of an $L \times L$ sum over the observed data image Y_i and several $D \times D$ sums over the other factors $q_{ij'}$, and since we have to compute these updates repeatedly, it is important to reduce the computation time

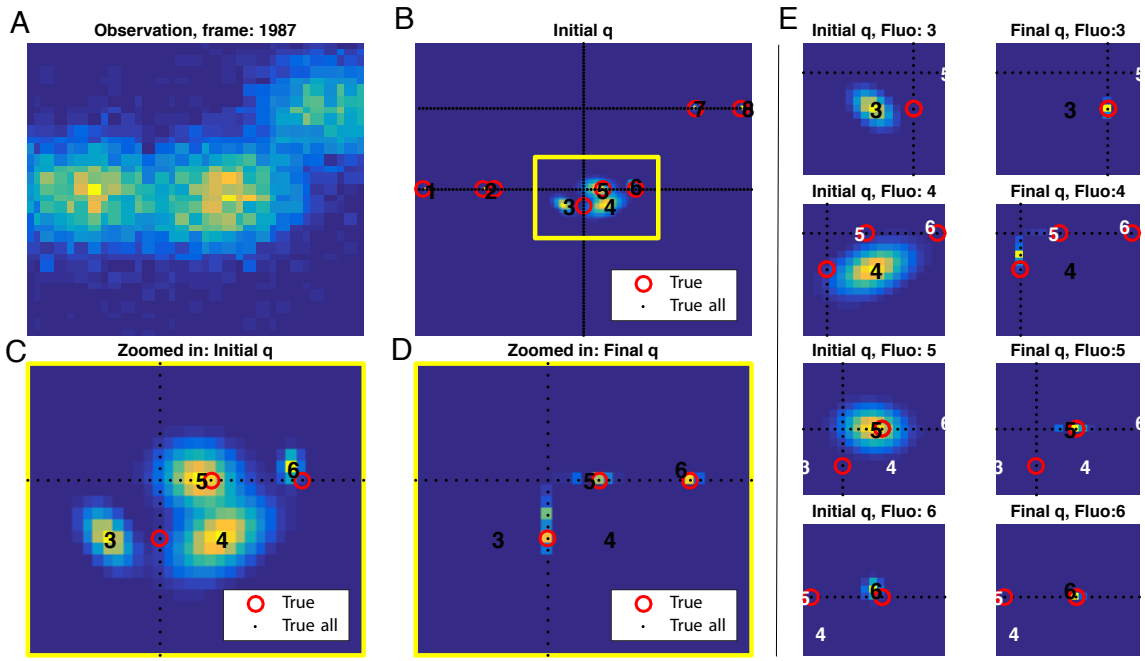

Figure 1.2: **Updating the factors q_{ij} in the E step**. **(A)** A single simulated observation frame Y_i. **(B)** 8 superimposed initial q_{ij} distributions for frame i, computed via Laplace approximations with means $\hat{F}_i^j$. True active fluorophores in I_i are labeled as red circles; true λ indicated by black dots; numbers indicate each $\hat{F}_i^j$ (ordering is arbitrary). Fluorophores 1,2,7, and 8 are relatively spatially isolated, with correspondingly large Fisher information (see supplementary Fig. 1.8) and so their initial q distributions are highly concentrated (and cannot even be seen beneath the red circles). In contrast, the closely overlapping PSF's of fluorophores 3,4,5, and 6 lead to broad initializations of q. **(C)** Zoom of yellow region outlined in **B**. **(D)** Final q_{ij}'s estimated by the vEM algorithm (same region as in panel **B**). Note that these have converged onto the region of positive λ (despite not having access to the ground truth λ), and the four original estimated fluorophore distributions have essentially converged near the 3 true active fluorophores in this region. **(E)** Further zoom showing details of each q_{ij} in **D**. The locations of other $\hat{F}_i^j$'s are indicated by white numbers. Left column: initial q_{ij}'s; right column: final q_{ij}'s. Again, the numbers indicate the $\hat{F}_i^j$ locations, which correspond to the peaks of the initial q_{ij}'s. Note the significant differences between the initial and final q_{ij}'s.

in this inner loop. We have found that we can effectively summarize the data in each frame by using a conditional Laplace approximation to the likelihood. Specifically, we

approximate

$$P(Y_i|F_i, m_i) \propto \mathcal{N}(F_i|\hat{F}_i, \hat{\Sigma}_i), \tag{1.10}$$

where the left hand side is the Poisson likelihood from eq. 1.5 and the right hand side denotes a multivariate normal density over $F_i \in R^{2m_i}$, with mean

$$\hat{F}_i = \arg\max_{F_i} P(Y_i|F_i, m_i) \tag{1.11}$$

and covariance inverse to the Fisher information J_i,

$$\hat{\Sigma}_i = [J_i]^{-1} = [-\nabla^2_{F_i} \ln P(Y_i|F_i, m_i)]|^{-1}_{F_i=\hat{F}_i}. \tag{1.12}$$

This Gaussian approximation to the Poisson likelihood is well-known to be accurate in the high-information regime where a sufficient number of photons are observed; see (Mukamel and Schnitzer 2012) for further discussion, and supplementary Fig. 1.8 for empirical evaluations of this approximation in the context of our simulations. (However, note that this Laplace approximation is not equivalent to assuming a Gaussian noise model with constant variance for Y_i; the Poisson noise model used here is significantly more accurate and consistent with the physics of shot noise.)

As we will see in the next subsection, this approximation allows us to replace the expensive sums noted above with evaluations of a much simpler $2m_i$-dimensional quadratic form. $\hat{F}_i$ and $\hat{\Sigma}_i$ serve as approximate sufficient statistics for Y_i, drastically reducing the size of the data that needs to be touched per iteration. In fact, the observed Fisher infor-

mation matrix J_i is sparse - if fluorophores j and j' are sufficiently distant (more than a couple PSF widths apart) then $J_{i,(j,j')} = 0$, and this can be used to further speed up the computation. In practice, we compute J_i via automatic differentiation ("Algorithm: ADiGator, a Toolbox for the Algorithmic Differentiation of Mathematical Functions in MATLAB Using Source Transformation via Operator Overloading") and locally optimize eq. 1.11 numerically using an efficient initializer discussed further below.

The Laplace approximation also provides a convenient initialization for the q_{ij}'s: we simply set each q_{ij} to be the marginal (Gaussian) density of F_i^j in eq. 1.10, with q_{ij} set to zero for all pixels sufficiently distant from $\hat{F}_i^j$.

Variational EM Algorithm

Now we can put the pieces together and derive our vEM algorithm. The first step is to expand the ELBO eq.1.9, plugging in our factorized q, the Laplace approximation eq.1.10, and the likelihood eq.1.7 to arrive at

$$\begin{aligned}\ln P(F, \lambda, Y|m) \approx \sum_i^N \Big\{ & \ln \mathcal{N}(F_i|\hat{F}_i, \hat{\Sigma}_i) \\ & + \ln \prod_j^{m_i} \frac{\lambda_{F_i^j}}{\eta} + \ln \text{Poisson}(m_i|\eta) \Big\} \\ & + \ln P(\lambda) + const. \end{aligned} \tag{1.13}$$

The vEM algorithm alternates between an E step (in which we optimize the ELBO wrt each q_{ij}, with λ and all the other q's held fixed), and an M step (in which we optimize the

ELBO wrt λ with all the q's held fixed).

M step:

$$\hat{\lambda} = \arg\max_{\lambda} \mathcal{L}(\lambda, q(F)) \tag{1.14}$$

$$= \arg\max_{\lambda} \ \mathbb{E}_q[\ln P(F, \lambda, Y|m)] \tag{1.15}$$

$$= \arg\max_{\lambda} \sum_{i}^{N} \sum_{j=1}^{\hat{m}_i} q_{ij} \odot \ln \lambda - N\eta + \ln p(\lambda), \tag{1.16}$$

with $\odot$ denoting pointwise multiplication. If we use a flat prior for λ, the $p(\lambda)$ term can be dropped, and if we abbreviate $Q = \sum_i^N \sum_j^{\hat{m}_i} q_{ij}$, we have the solution $\hat{\lambda} = Q/N$. (Recall that $\lambda, \hat{\lambda}$, and $Q \in \mathbb{R}_+^{D\times D}$.) This is a natural generalization of the MLE for a discretized inhomogeneous Poisson process.

E step:

$$q_{ij} = \arg\max_{q_{ij}} \mathcal{L}(\hat{\lambda}, q) \tag{1.17}$$

$$\propto \exp\left\{\mathbb{E}_{q_{\backslash ij}}\left[\ln P(F, \hat{\lambda}, Y|m)\right]\right\} \tag{1.18}$$

$$\propto \exp\left\{\mathbb{E}_{q_{\backslash ij}}\left[\sum_{n}^{N} -\frac{1}{2}(F_n - \hat{F}_n)^T J_n (F_n - \hat{F}_n)\right.\right.$$

$$\left.\left. + \sum_{n}^{N}\sum_{k=1}^{m_n} \ln \hat{\lambda}_{F_n^k}\right]\right\} \tag{1.19}$$

$$\propto \hat{\lambda}_{F_i^j} \exp\left\{ -\frac{1}{2}(F_i^j - \hat{F}_i^j)^T J_i^{jj} (F_i^j - \hat{F}_i^j)\right.$$

$$\left. - \sum_{k\neq j}^{m_i} (F_i^j - \hat{F}_i^j)^T J_i^{jk} (\mu_{ik} - \hat{F}_i^k)\right\}; \tag{1.20}$$

here we have abbreviated $q_{\backslash ij} = q/q_{ij}$, J_i^{jk} is the 2×2 block of J_i corresponding to

fluorophores j and k, and $\mu_{ik} = E_{q_{ik}} F_i^k$. Note that in the end, due to the Laplace approximation, the q_{ik}'s only enter the update above via their means, and that the updated q_{ij} is simply proportional to a Gaussian factor multiplied by $\hat{\lambda}$.

Finally, note that the effective support of each q_{ij} tends to shrink compared to the initialization (and this increasing sparsity can be readily exploited computationally); this makes sense, because our initialization (from the Laplace approximation) is based only on the likelihood of a single frame Y_i — when we incorporate the information from other frames (via $\hat{\lambda}$) the approximate posterior q_{ij} tends to become more concentrated. See Fig. 1.2 for an illustration.

Extensions and further details

In the developments above we have deferred several questions. How do we estimate m_i, the number of active fluorophores in each frame? How do we initialize the optimization problem eq.1.11 in the Laplace approximation for the likelihood? How do we make use of prior information $P(\lambda)$ in the M-step?

For the first two tasks mentioned above we exploit pre-existing solutions. Specifically, we have found that the FALCON (Min et al. 2014) method provides fairly good preliminary estimates of both the number and the location of fluorophores in each frame i; the former is used as m_i and the latter are used to initialize the optimization in eq. 1.11.

One of the major benefits of a Bayesian approach is that we can easily incorporate prior information about parameters of interest - in this case, λ. In principle it is possible to incorporate various sources of prior information about λ, but here we restrict our attention to the simplest case: in many cases the true underlying λ is known to be sparse,

and we can exploit this fact to improve our estimates significantly. (Note that this sparsity constraint on λ is in addition to the fact that the images I_i are sparse, a fact that we have already exploited repeatedly. Also note that standard super-resolution approaches exploit the sparsity of each I_i — but since they simply average over the estimated $\hat{I}_i$ to obtain $\hat{\lambda}$, previous approaches have not attempted to further exploit the sparsity of λ.) An effective and computationally trivial approach is to apply the standard L1 "soft threshold" operator (Bach et al. 2011) to Q in the M-step (eq.1.16):

$$\hat{\lambda} = S_c(Q)/N = S_c\left[\sum_i^N \sum_j^{m_i} q_{ij}(F_i^j)\right]/N \tag{1.21}$$

where we define $S_c[x] = \max(x - c, 0)$; the threshold c can easily be chosen to achieve an a desired level of sparsity (typically set by prior knowledge, though cross-validation could be used here instead).

When active fluorophores are well-isolated in the image (i.e., the "bumps" corresponding to each active fluorophore are sufficiently distinguishable) then FALCON's estimates are typically accurate, and the corresponding entries of the Fisher information matrix J_i are large. However, when the bumps overlap then the Fisher information can decrease significantly (see supplementary Fig. 1.8 for an illustration) and the accuracy of m_i and the nearby fluorophore estimates $\hat{F}_i^j$ decrease. In this case we can achieve significantly improved accuracy by exploiting information from other frames, via the estimated $\hat{\lambda}$.

Thus the full algorithm proceeds as follows. To initialize we run FALCON on each frame and compute the Laplace approximation, then run vEM (restricting attention to the $\sim 50\%$ of frames on which fluorophore activation was sparsest, to improve localization

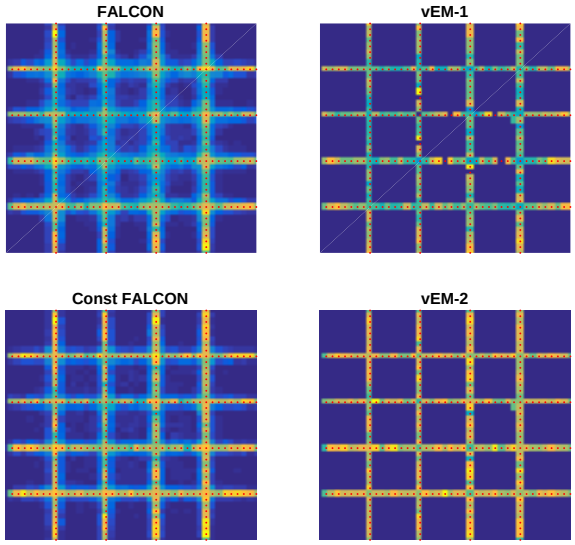

Figure 1.3: **Illustrating the algorithm steps on a simulated example**. Upper left: FALCON estimate given 5000 frames of data. Upper right: output after first run of vEM, based on the 2000 sparsest frames. This estimate is used to constrain FALCON to obtain a more accurate preliminary estimate $\hat{F}$ and $\hat{m}$, with all 5000 frames (lower left), and our final estimate using all 5000 frames after a second vEM run is shown in the lower right. The grid shape of the true underlying simulated image (red dots) is recovered essentially perfectly in the lower right; estimation noise (averaged over frames) blurs the true grid shape significantly in the left panels.

accuracy). Then we rerun FALCON incorporating information from the preliminary $\hat{\lambda}$ estimate to improve the estimates $\hat{F}$ and $\hat{m}$. FALCON uses an L1-penalized regression approach to obtain preliminary estimates of I_i from Y_i; it is straightforward to include a weighted L1 term where the weight is inversely proportional to $\hat{\lambda}$ to encourage the FALCON output to localize near regions of high $\hat{\lambda}$ (and to eliminate some spurious location estimates). Then we can use the resulting updated $\hat{\lambda}$-constrained FALCON estimates of the fluorophore locations to re-initialize eq. 1.11 on the subset of frames where the preliminary FALCON and vEM results disagree (updating these m_i as well), and proceed with further vEM iterations. This procedure can in principle be iterated, though we find in

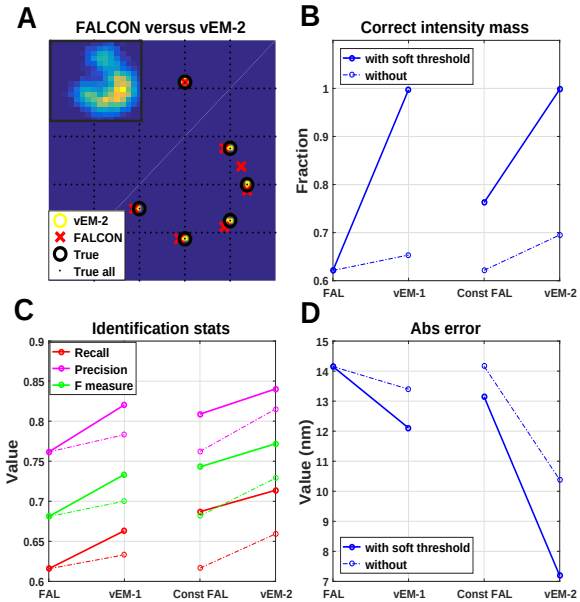

Figure 1.4: **Evaluating performance of each algorithm step**. **(A)** Estimates of active fluorophore locations in a single frame. Dotted black line indicates location of remaining (inactive) fluorophores. Note that vEM-2 estimates are more accurate than FALCON estimates. Algorithm steps and all simulation details follow Fig. 1.3, except in the vEM-1 step we compute results over all 5000 frames (not just 2000 frames, as in Fig. 1.3 upper right), for apples-to-apples comparison against the vEM-2 results here. Inset: observed data Y_i for this frame. **(B)** The percentage of $\hat{\lambda}$ contained within the true support after each algorithm step. Note that vEM leads to significant improvements over FALCON; applying soft-thresholding in the M-step also provides significant improvements. **(C)** Recall, Precision and F-measure of identified fluorophores (solid: soft-thresholded; dashed: no soft-thresholding), and **(D)** mean absolute error of fluorophore location estimates. In both cases, similar trends as in **(B)** are visible.

practice that one outer iteration typically suffices. In the inner loop, we found that just 5 vEM iterations were sufficient. See Fig. 1.3 for an illustration of each algorithm step's output.

1.4 Results

Figures 1.3-1.6 detail simulated comparisons between FALCON, a state-of-the-art super-resolution algorithm (Min et al. 2014), and the vEM algorithm developed here. The simulated image was a simple grid pattern; full simulation details are given in the Appendix. In Fig. 1.3 it is clear that the variational EM algorithm recovers the true grid support in this simulated example more accurately than does the FALCON algorithm. Fig. 1.4 quantifies the performance of the new proposed algorithm following each step illustrated in Fig. 1.3. Specifically, we examine the proportion of fluorophores whose estimated positions were recovered on the correct support of the true underlying grid image (panel B); the frame-by-frame precision and recall (and F measure, defined as the harmonic mean of precision and recall) of individual fluorophore estimates (panel C); and the frame-by-frame absolute error of individual fluorophore estimates (panel D). In each panel, we see that vEM leads to significant improvements over FALCON; applying soft-thresholding in the M-step also leads to significant improvements.

Figure 1.5 quantifies these results further, and adds comparisons to other competitive algorithms in the literature. Again the conclusion is that the vEM approach provides significantly more accurate estimates at little computational cost. Supplementary Figures 1.9 and 1.10 in the appendix show that this conclusion holds fairly uniformly over a wide range of PSF widths and average fluorophore densities, respectively.

Figure 1.6 provides a visual summary of one of the critical points of this paper: as the number of observed frames N increases, the vEM estimator continues to improve, and by $N = 5000$ is able to recover the true support of the underlying grid image with

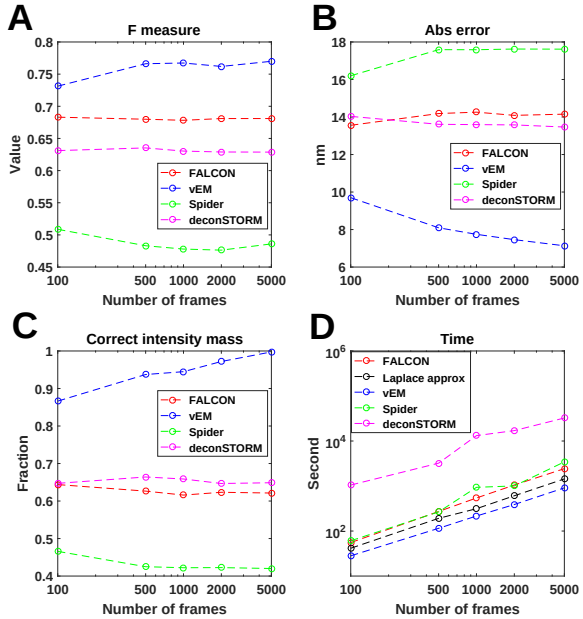

Figure 1.5: **Evaluation of vEM in comparison with FALCON Min et al. 2014, deconSTORM Mukamel, Babcock, and Zhuang 2012, and SPIDER Hugelier et al. 2016 as a function of the number of observed frames** N. **A-C**: F-measure, mean absolute fluorophore estimation error, and fraction of fluorophore mass recovered correctly on the true underlying grid computed as in Fig. 1.4. The vEM approach outperforms the other state-of-the-art algorithms on all of these metrics. **D**: Computational time of each algorithm step. Our full algorithm runs FALCON (red curve), computes the Laplace approximation (black curve), then iterates vEM to convergence (blue curve), then repeats the whole process on at least a subset of frames, so overall speed is $\sim$ 2x slower than FALCON overall. The deconSTORM algorithm is relatively much slower here.

almost perfect accuracy. FALCON, on the other hand (as well as other approaches that estimate each frame independently), outputs estimates that appear blurry, due to noise in the estimated fluorophore locations, averaged over many frames — and this effective blur (and resulting loss of resolution) does not decrease asymptotically as N increases, since unlike vEM, FALCON does not exploit information from the $(N-1)$ other frames to improve estimation of individual frames.

Finally, Figure 1.7 shows a comparison of FALCON vs vEM applied to real data (see

Appendix for full details). In this case the ground truth image is not available for comparison, but nonetheless the results are consistent with the simulated results described above: vEM leads to a sharper, better-resolved image than does FALCON.

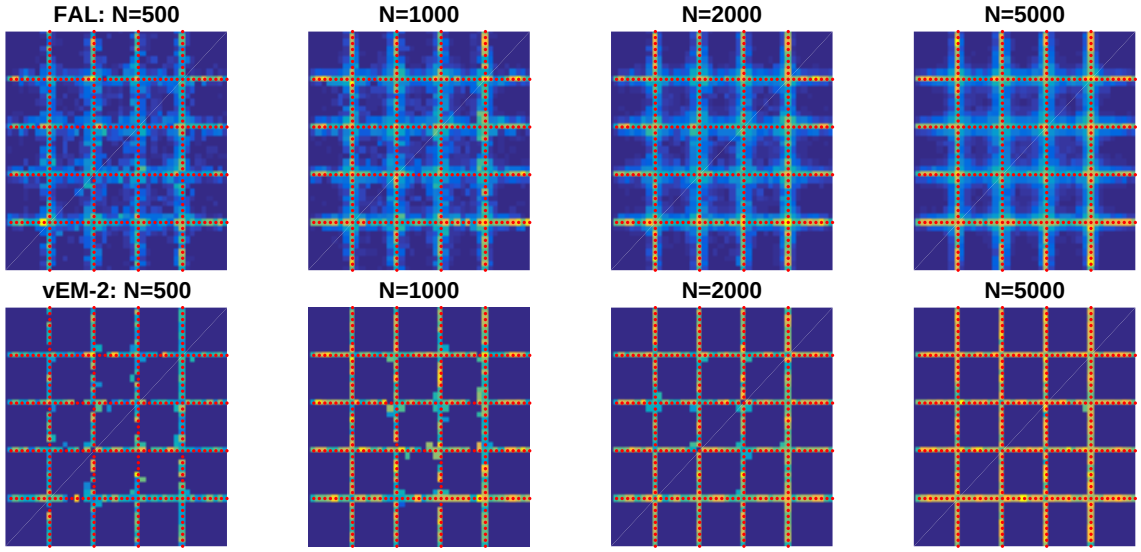

Figure 1.6: **Estimates with varying number of observed frames N**. Estimated $\hat{\lambda}$ images output by FALCON (upper panels) and vEM (lower panels) with 500, 1000, 2000, and 5000 simulated frames (with imaging parameters such as PSF width and fluorophore density p held fixed over all frames). Red dots indicate the ground truth grid. The grid recovery accuracy of the vEM algorithm continues to improve with N — recovering the underlying grid structure nearly perfectly when $N = 5000$ — but the estimation noise-induced blur in the FALCON estimate does not decrease with N.

1.5 Discussion

We have introduced scalable Bayesian methods for improved estimation in super-resolution microscopy. By further extending the reach of these critical imaging methods, our approach can significantly impact a variety of biological applications. The hybrid vEM / Laplace-approximation / sparse-representation approach developed here is more generally applicable in other hierarchical sparse signal model applications (Picardo et al.

2016). Our methods exploit the insight that sharing information across image frames significantly improves accuracy — and this effect grows more powerful as the number of frames N increases.

Similar points have appeared previously in the super-resolution microscopy literature, notably in (Mukamel, Babcock, and Zhuang 2012) and (Cox et al. 2012). The methods introduced in (ibid.) are seldom used in practice on large-scale imaging data, due to prohibitive computational expense. The vEM methods we have introduced here are much more scalable (Fig. 1.5D); indeed, we were unable to obtain good results from the method used in (ibid.) in a reasonable amount of computational time (> 1 day) and so we did not show comparisons against this method here (see appendix for further discussion).

The deconSTORM method described in (Mukamel, Babcock, and Zhuang 2012) (see also Fig. 1.5) attempts to improve upon simple Richardson-Lucy deconvolution by incorporating local information about the survival of active fluorophores from one frame i to the next $(i + 1)$. Our approach is orthogonal: we share information between frames I_i *globally*, through $\hat{\lambda}$. As we discuss in the appendix ("Markov model"), the vEM framework extends easily to handle these local correlations between fluorophores at frames i and $i + 1$. We observed that although incorporating these local correlations can slightly improve the recovery of individual fluorophores (Fig.1.11), the local Markov model does not significantly qualitatively improve the accuracy of the final estimated $\hat{\lambda}$ (Fig.1.12).

A final interesting and important direction for future work would be to extend some of the methods developed here to the case where the fluorophores are moving from frame to frame, in the context of single-particle tracking experiments (Shen and Andersson 2009).

vEM code is available here: `https://github.com/SunRuoxi/vEM`

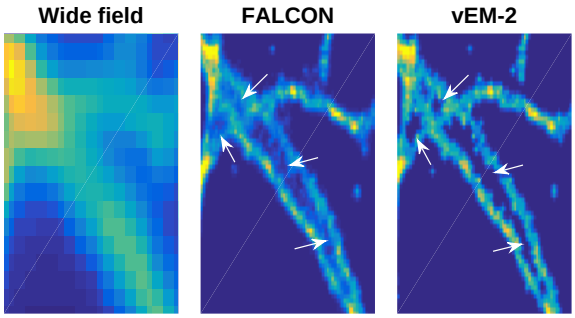

Figure 1.7: **Analysis of real tubulin image data**. Final resolved images output by FALCON and vEM-2 with 5000 frames. Wide field image, with all fluorophores turned on simultaneously, is given in first panel. Note that FALCON image is blurrier than the vEM-2 image, especially in areas of high fluorophore density, e.g., where multiple tubulin branches are close together, as noted by white arrows.

Details of algorithmic comparisons

All parameters of the methods compared here are tuned for best performance.

FALCON (Min et al. 2014): We set the sparsity parameter (the ℓ_1 weights) κ to 3. The threshold above which the support is defined is set as 10% of the maximum intensity.

SPIDER (Hugelier et al. 2016): We set the sparsity parameter, the weights of ℓ_0 regulation, $\kappa = 250$.

deconSTORM (Mukamel, Babcock, and Zhuang 2012): In our simulations accuracy continued to improve even after 5000 iterations, so we used 5000 iterations in our comparisons. Note computation time is proportional to the number of iterations, so this method could be sped up at the cost of some accuracy.

3B (Cox et al. 2012): As mentioned in the Discussion, we were not able to obtain reasonable results using this method, even after > 1 day of computation, for data sets with e.g. 2000 observed frames. In personal communications with the developers of this method, it was emphasized that this approach is better suited for smaller images and smaller values of N, since the speed of this method decreases more or less with the square of the size of the PSF (in pixels), the size of the image area being analyzed, and the number of frames observed. Therefore we did not pursue further quantitative comparisons against this method.

Evaluation: Identified fluorophores are defined as estimates within some fixed distance from the true fluorophores. The cutoff radius we used was 50 nm. FALCON returns a list of fluorophore locations, whereas vEM, deconSTORM, and SPIDER return images of the estimated fluorophore density. To quantify fluorophore estimate accuracy for these methods we thresholded these images and took local weighted averages to obtain the estimated fluorophore locations.

Experimental details

Simulation: To validate our analysis method, we simulated grid data on a 32-by-32 pixel map with an image pixel size of 100 nm. The final resolved image sits on a 3x finer grid with super resolution pixel size of 33 nm. Unless stated otherwise, each frame has an emission rate of 0.04, corresponding to an average molecule density of 6.8 μm^{-2}. The average photon number is 1,000 per fluorophore with PSF width 150 nm in standard deviation or 353 nm in FWHM. To test the performance of our method under different practical situ-

ations, we varied critical parameters, including the number of frames, molecule density, and PSF width. In those simulations, we replace the above parameters with a range stated in the corresponding figure caption and keep other parameters unchanged.

Real data: We reconstruct a patch of tubulins on a 32-by-32 pixel image map with 5000 real experimental frames. The final resolved image sits on a 4x finer grid. We modeled the PSF as a Gaussian blur with width of 183 nm in standard deviation; this parameter was estimated by fitting observations of non-overlapping fluorophores to a 2D Gaussian function, following Small and Stahlheber 2014. The dataset is provided as Tubulin ConjAL647 on the Single Molecule Localization Microscope website Sage et al. 2015.

1.6 Markov model

In the main text, we focus on a model in which fluorophores become active according to a Poisson process with rate λ. The active fluorophores in one frame are conditionally independent from those in other frames, given λ. In this section, we incorporate the phenomenon that some fluorophores do not quench immediately, i.e. there is a probability that an active fluorophore will remain active in the following frames. Thus the active fluorophores in one frame consist of two groups: "newborn" fluorophores that activate from the dark state with rate λ (the same as before), plus fluorophores remaining active from the previous frame, each with probability α. Under this assumption, the active fluorophores in one frame are dependent on those in the frame before and after, so we denote the new model as the "Markov model" and the original model as the "non-Markov model."

The Markov model incorporates the positions of active fluorophores in neighboring

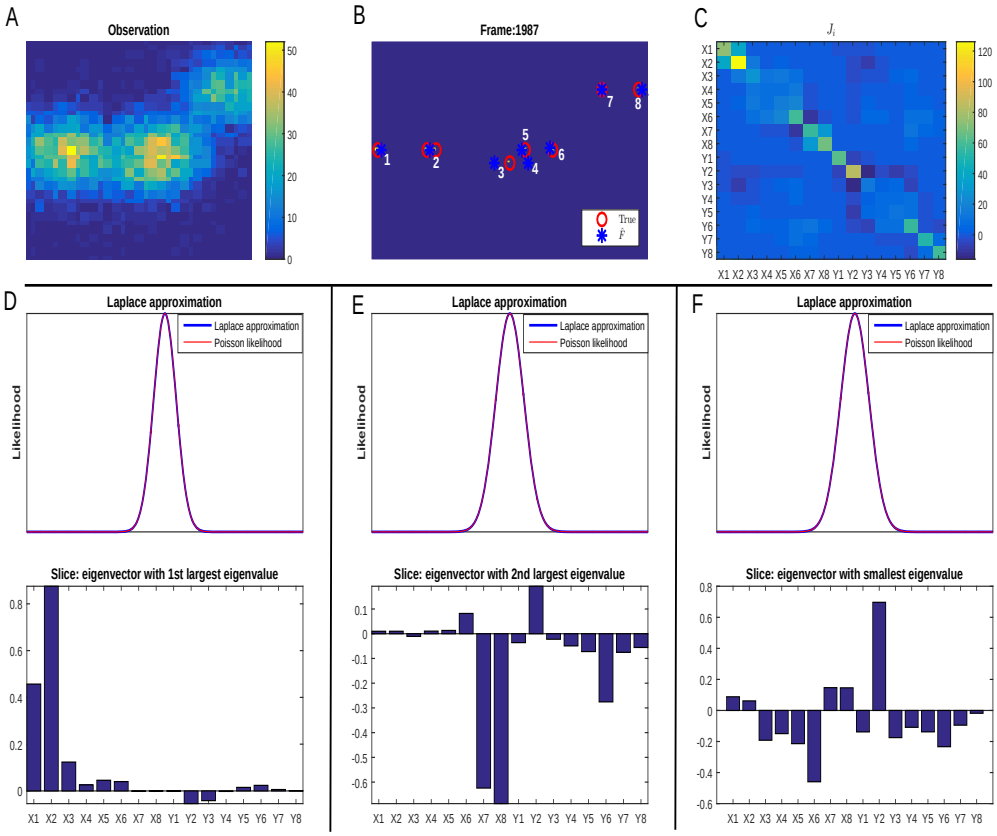

Figure 1.8: **Accuracy of the Laplace approximation**. We use the same frame as in Figure 1.2 in the main text as an illustration. **(A)** Y_i. **(B)** Red circles are true positions; blue stars are $\hat{F}_i^j$. **(C)** Fisher information matrix J_i. Xj indicates x coordinates of fluorophore j, and Yj the y coordinates. FALCON inferred 8 fluorophores in this frame, so we have 16 total coordinates. Note that the blocks of J_i corresponding to fluorophores 3,4,5,6 have smaller values, indicating reduced estimation accuracy due to overlapping PSF bumps. Panels **(D)**, **(E)**, **(F)** indicate the accuracy of the Laplace approximation for the Poisson likelihood. Since the likelihood w.r.t. F_i is a 16-dimensional function, we can only display slices of this function. We choose slices in the direction of eigenvectors of J_i — the principal components of the Laplace approximation. (Directions appear in bottom panels.) The red (Poisson likelihood) and blue (Laplace approximation) curves align very closely in each of the three directional slices shown here.

frames $i+1$ and $i-1$, which can potentially be useful to help pinpoint the positions of active fluorophores in each frame i. Thus we would expect the Markov model to outperform the non-Markov in localizing individual fluorophores. (Similar neighboring-frame

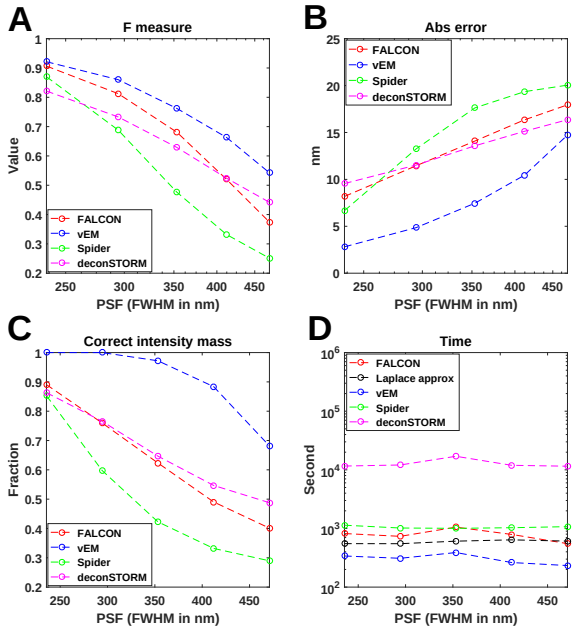

Figure 1.9: **Evaluation of vEM in comparison with FALCON Min et al. 2014, deconSTORM Mukamel, Babcock, and Zhuang 2012, and SPIDER Hugelier et al. 2016 as a function of PSF width**. Panel layout as in Fig. 1.5. 2000 frames were used here.

effects are incorporated in Mukamel, Babcock, and Zhuang 2012 and Cox et al. 2012.) In this section, we will first introduce the Markov model, and then show results comparing the effectiveness of the Markov model versus the non-Markov model.

Model

We begin by writing down the Markov model for the time series of activations of fluorophores $I_{:,xy}$ at location xy across all N frames:

$$p(I_{:,xy}|\lambda_{xy}, \alpha) = p(I_{1,xy}) \prod_{i>1}^{N} p(I_{i,xy}|I_{i-1,xy}, \lambda_{xy}, \alpha) \tag{1.22}$$

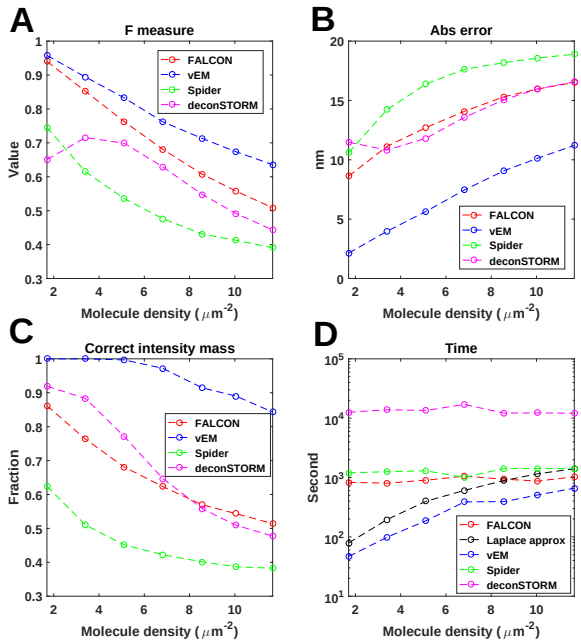

Figure 1.10: **Evaluation of vEM in comparison with FALCON Min et al. 2014, deconSTORM Mukamel, Babcock, and Zhuang 2012, and SPIDER Hugelier et al. 2016 as a function of fluorophore density** p. Panel layout as in Figures 1.5 and 1.9. We found that deconSTORM tends to have an inflated false positive rate (i.e., lower precision) for small values of p in panel (A). Also note that vEM is relatively cheaper computationally for small p, where there are fewer fluorophores to iterate over in the Laplace approximation and vEM steps. 2000 frames were used here.

(As usual, fluorophores in different locations xy activate conditionally independently given λ.) The transition matrix is given by

$$\begin{cases} P(I_{i,xy} = 1 | I_{i-1,xy} = 0, \lambda, \alpha) = \lambda \\ P(I_{i,xy} = 0 | I_{i-1,xy} = 0, \lambda, \alpha) = 1 - \lambda \\ P(I_{i,xy} = 1 | I_{i-1,xy} = 1, \lambda, \alpha) = \alpha + \lambda \\ P(I_{i,xy} = 0 | I_{i-1,xy} = 1, \lambda, \alpha) = 1 - \alpha - \lambda, \end{cases} \tag{1.23}$$

where α is the probability that an active fluorophore remains active in the next frame, and λ is defined in eq. 1.2, 1.3; note that the probability of a new fluorophore activating in any frame is typically fairly low (to guarantee that each image I_i is sparse), so $\lambda \ll 1$, while the probability of remaining active may be non-negligible. We are most interested here in the case that α is significantly greater than 0, implying $\lambda \ll \alpha$. Finally, note that we use a Bernoulli emission model here instead of the Poisson emission model used in the main text (eq. 1.5); this simplifies the derivations below. Of course the Poisson and Bernoulli models are identical in the limit of small λ.

After the Laplace approximation our full approximate loglikelihood is

$$\begin{aligned}\ln p(I, Y|\lambda, \alpha) = \sum_{i=1}^{N} \Bigg\{ \sum_{x}^{D} \sum_{y}^{D} \ln p(I_{i,xy}|I_{i-1,xy}, \lambda_{xy}, \alpha) \\ + \ln \mathcal{N}(F_i|\hat{F}_i, \hat{\Sigma}_i) \Bigg\}\end{aligned} \tag{1.24}$$

where in eq. 1.24 we define I_0 as all zeros.

Variational EM Algorithm

Now we proceed as before and maximize the ELBO (eq. 1.9) to obtain the E and M steps. We will assume that α is known. In reality, of course, α is unknown and needs to be estimated along with λ. It is straightforward to derive EM iterations for α as well, but we do not pursue this here. Instead, in the Results section we will simulate data from the Markov model and estimate λ using a known value of α, to give the Markov approach the best possible chance of improving over the results of the non-Markov model.

M step

$$\hat{\lambda} = \arg\max_{\lambda} \mathcal{L}(\lambda, q(F)) \tag{1.25}$$

$$= \arg\max_{\lambda} \ \mathbb{E}_q[P(I, Y|m, \lambda, \alpha)] \tag{1.26}$$

$$= \arg\max_{\lambda} \ \mathbb{E}_q[\sum_{i}^{N} \ln p(I_i|I_{i-1}, \lambda, \alpha)] \tag{1.27}$$

$$= \arg\max_{\lambda} \ \mathbb{E}_q[\sum_{i}^{N} (1 - I_{i-1}) \ln \left[\lambda^{I_i}(1-\lambda)^{1-I_i}\right]$$

$$+ (I_{i-1}) \ln \left[(\alpha+\lambda)^{I_i}(1-\alpha-\lambda)^{1-I_i}\right]]; \tag{1.28}$$

eq. 1.28 combines the cases in eq. 1.23 and uses the property that I_i can only take values of 0 or 1. We have dropped the pixel subscript xy to simplify notation; the maximization problem is separable over locations xy and therefore we can optimize for each pixel independently in parallel.

Setting the derivative w.r.t. λ to zero, we obtain

$$\frac{Q_{1,N}}{\hat{\lambda}} - \frac{Q'_{1,N}}{\hat{\lambda}} - \frac{N - Q_{1,N}}{1-\hat{\lambda}} + \frac{Q_{0,N-1} - Q'_{1,N}}{1-\hat{\lambda}}$$
$$+ \frac{Q'_{1,N}}{\alpha+\hat{\lambda}} - \frac{Q_{0,N-1} - Q'_{1,N}}{1-\alpha-\hat{\lambda}} = 0, \tag{1.29}$$

where we define $Q \in R_+^{D\times D}$ and $Q' \in R_+^{D\times D}$:

$$Q_{S,N} = \sum_{i=S}^{N} E(I_i) = \sum_{i=S}^{N} \sum_{j}^{m_i} q_{ij}(F_{ij}) \tag{1.30}$$

$$Q'_{S,N} = \sum_{i=S}^{N} E(I_i) \odot E(I_{(i-1)}) \tag{1.31}$$

$$= \sum_{i=S}^{N} \left\{ \sum_{j=1}^{m_i} q_{ij}(F_{ij}) \right\} \odot \left\{ \sum_{k=1}^{m_{(i-1)}} q_{ik}(F_{ik}) \right\}.$$

$$\tag{1.32}$$

Now set

$$k_1 = Q_{1,N} - Q'_{1,N} \tag{1.33}$$

$$k_2 = N - Q_{1,N} - Q_{0,N-1} + Q'_{1,N} \tag{1.34}$$

$$k_3 = \frac{Q'_{1,N}}{\alpha + \hat{\lambda}} - \frac{Q_{0,N-1} - Q'_{1,N}}{1 - \alpha - \hat{\lambda}} \tag{1.35}$$

$$\approx \frac{Q'_{1,N}}{\alpha} - \frac{Q_{0,N-1} - Q'_{1,N}}{1 - \alpha} \tag{1.36}$$

where in eq. 1.36, we used $\lambda << \alpha$. Therefore, eq. 1.29 becomes

$$\frac{k_1}{\hat{\lambda}} - \frac{k_2}{1 - \hat{\lambda}} + k_3 = 0, \tag{1.37}$$

which reduces to a simple quadratic equation in $\hat{\lambda}$. We select the valid solution $\hat{\lambda} \in (0, 1)$.

Similarly as in the non-Markov model, we can perform soft-thresholding on the obtained value to increase the sparsity of $\hat{\lambda}$.

Finally, note that if we set $\alpha = 0$, eq. 1.29 becomes

$$\frac{Q}{\hat{\lambda}} - \frac{N-Q}{1-\hat{\lambda}} = 0; \tag{1.38}$$

this leads to $\hat{\lambda} = Q/N$, which corresponds to the M step in the non-Markov model (description under eq. 1.16), as desired. **E step**:

$$\begin{aligned}
q_{ij}(F_i^j) &= \arg\max_{q_{ij}} \mathcal{L}(\hat{\lambda}, q(F)) && (1.39)\\
&\propto \exp\left\{\mathbb{E}_{q_{\setminus ij}}\left[\ln P(I, Y|m, \hat{\lambda}, \alpha)\right]\right\} && (1.40)\\
&\propto \exp\left\{\mathbb{E}_{q_{\setminus ij}}\Bigg[\ln p(I_i|I_{i-1}, \hat{\lambda}, \alpha)\right.\\
&\qquad + \ln p(I_{i+1}|I_i, \hat{\lambda}, \alpha)\\
&\qquad \left. + \ln \mathcal{N}(F_i|\hat{F}_i, \hat{\Sigma}_i)\Bigg]\right\} && (1.41)\\
&\propto \exp\Bigg\{
\end{aligned}$$

$$\underbrace{+\left(\mathbb{E}_{q_{\setminus ij}}\left[I_{i-1}\right] \ln\left[\left(\frac{1-\hat{\lambda}}{\hat{\lambda}}\right)\left(\frac{\alpha+\hat{\lambda}}{1-\alpha-\hat{\lambda}}\right)\right] + \ln\frac{\hat{\lambda}}{1-\hat{\lambda}}\right)_{F_i^j}}_{\text{effect of } I_{i-1}} \tag{1.42}$$

$$\underbrace{+\left(\mathbb{E}_{q_{\setminus ij}}\left[I_{i+1}\right] \ln\left[\left(\frac{1-\hat{\lambda}}{\hat{\lambda}}\right)\left(\frac{\alpha+\hat{\lambda}}{1-\alpha-\hat{\lambda}}\right)\right] + \ln\frac{1-\alpha-\hat{\lambda}}{1-\hat{\lambda}}\right)_{F_i^j}}_{\text{effect of } I_{i+1}} \tag{1.43}$$

$$\underbrace{-\frac{1}{2}(F_i^j - \hat{F}_i^j)^T J_i^{jj}(F_i^j - \hat{F}_i^j) - \sum_{k\neq j}^{m_i}(F_i^j - \hat{F}_i^j)^T J_i^{jk}(\mu_{ik} - \hat{F}_i^k)}_{\text{Laplace approx}} \tag{1.44}$$

$$\Bigg\},$$

where the operator $()_{F_i^j}$ is the value of the 2D function of the variable in the parentheses at location F_i^j.

Note that if $\alpha = 0$ and $\lambda \ll 1$, then

$$eq.1.42 + eq.1.43 \approx \ln \lambda;$$

therefore, we recover the E step in the non-Markov model (eq.1.20), as desired.

Results

To quantify the benefits of including the Markov terms in the model, we generate data from the Markov model with a range of parameters and perform inference with both the Markov and non-Markov models. Note that the non-Markov model is mis-specified in these simulations, while the Markov model is given the "unfair" advantage of knowing the true value of α. Nonetheless, somewhat surprisingly, our basic conclusion is that incorporating the Markov effects has only a small effect on inference performance. Fig. 1.11 shows that the estimation accuracy on individual fluorophores is improved modestly if the Markov terms are included, once α is sufficiently large. However, in Fig. 1.12 we see that the Markov terms lead to negligible improvement in the overall estimate of λ, which is the main object of interest in many super-resolution imaging studies. Thus we conclude that the "local" information encoded by the Markov terms in the model is mostly redundant with the "global" information encoded by our estimate $\hat{\lambda}$, which is shared across frames to improve our estimate of each I_i.

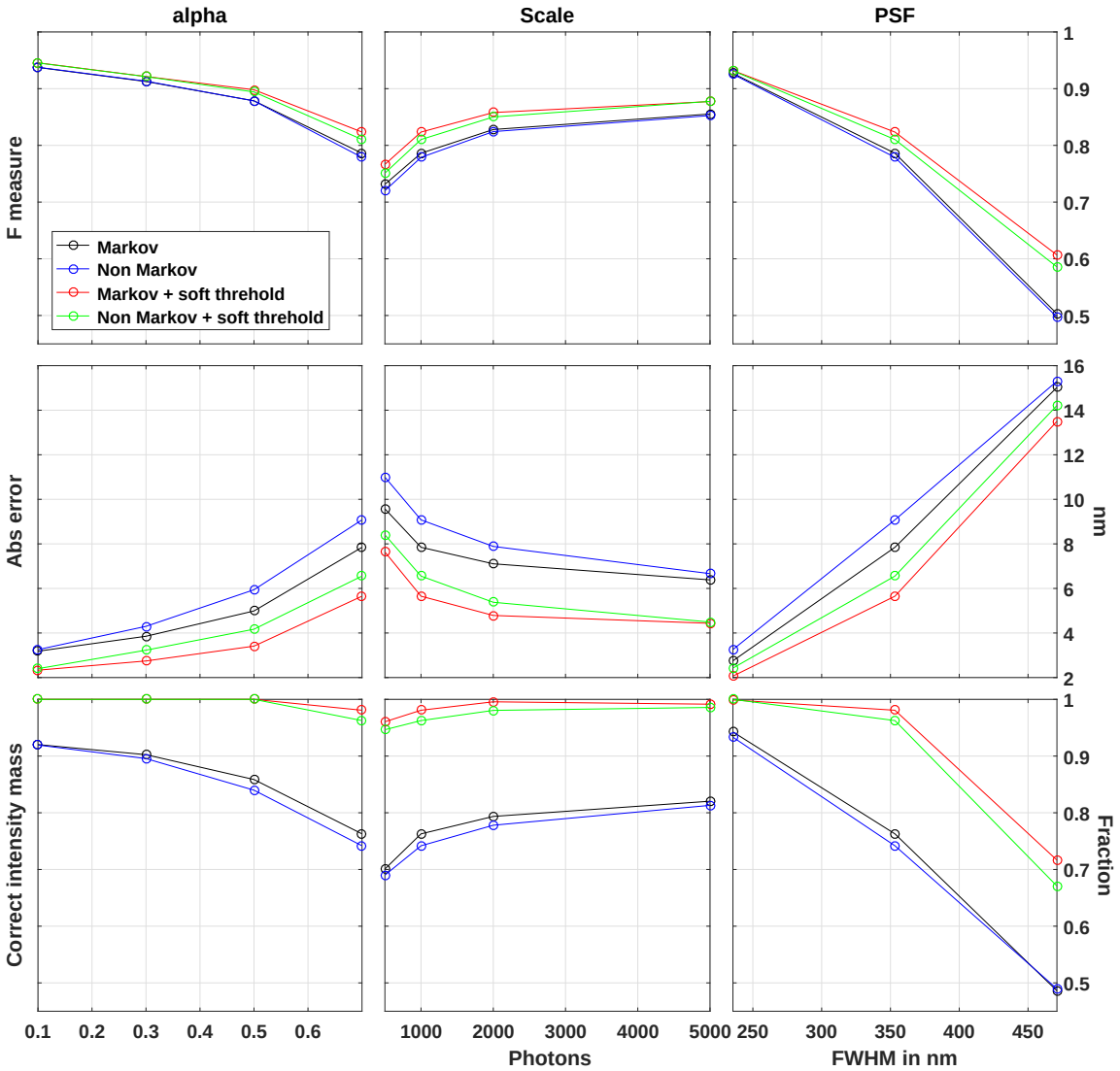

Figure 1.11: **Evaluation of Markov and non-Markov models as a function of α (First column), *Scale* (Second column), and *PSF* (Third column), with and without soft thresholding**. The performance is quantified using the same measures as in the main text. *Scale* is the average number of photons per active fluorophore. We use $N = 2000$ frames. Emission rate p is 0.01. When not indicated otherwise, α, *Scale*, and *PSF* are set to be 0.7, 1000 photons, and 353.25 nm respectively. Note that this is rather large value of α; as seen in the left panel, smaller differences between the Markov and non-Markov models are seen when smaller values of α are used.

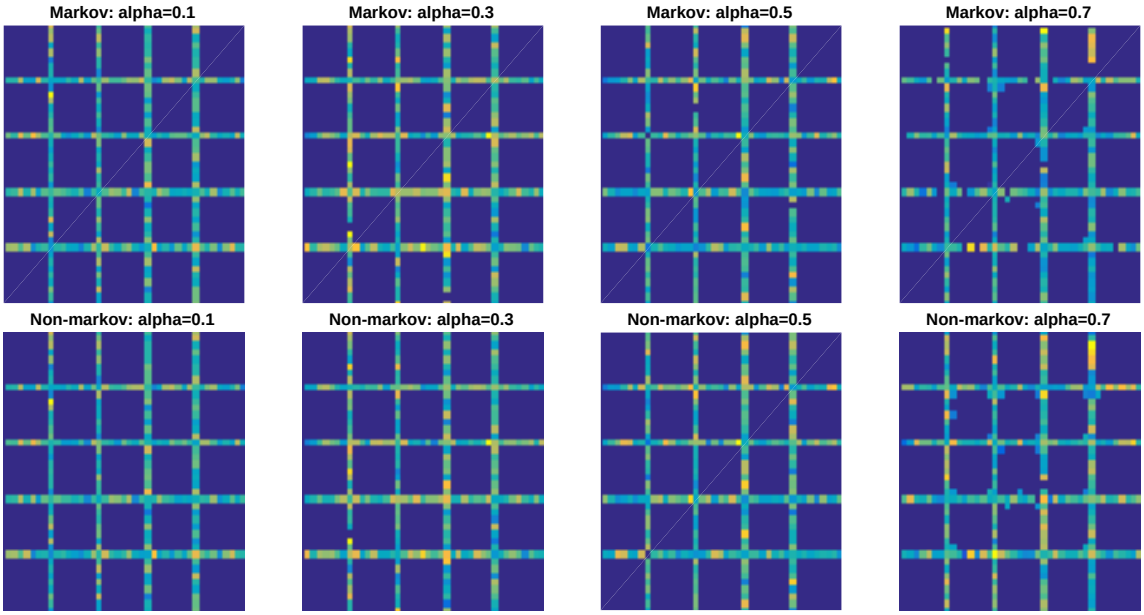

Figure 1.12: **Estimates of Markov and non-Markov model as a function of α with soft-thresholding.**. The final resolved images of Fig. 1.11 (First column). The test image was the same grid used in the main text. Recall that α influences the number of active fluorophores (with large α corresponding to more persistent fluorophores and therefore higher fluorophore density); the average fluorophore densities in the four columns shown here are 1.90, 2.55, 3.44, and 5.75μm^{-2} (left to right). We use $N = 2000$ frames. Emission rate p is 0.01, with an average of 1000 photons per fluorophore. PSF is 353.25 nm. Note that no major differences are seen between the Markov and non-Markov estimates. Similar results were obtained over a wide range of parameters (not shown).

Chapter 2

Particle Tracking

Many important datasets in physics, chemistry, and biology consist of noisy sequences of images of multiple moving overlapping particles. In many cases, the observed particles are indistinguishable, leading to unavoidable uncertainty about nearby particles' identities. Exact Bayesian inference is intractable in this setting, and previous approximate Bayesian methods scale poorly. Non-Bayesian approaches that output a single "best" estimate of the particle tracks (thus discarding important uncertainty information) are therefore dominant in practice. Here we propose a flexible and scalable amortized approach for Bayesian inference on this task. We introduce a novel neural network method to approximate the (intractable) filter-backward-sample-forward algorithm for Bayesian inference in this setting. By varying the simulated training data for the network, we can perform inference on a wide variety of data types. This approach is therefore highly flexible and improves on the state of the art in terms of accuracy; provides uncertainty estimates about the particle locations and identities; and has a test run-time that scales linearly as a function of the data length and number of particles, thus enabling Bayesian inference in arbitrarily large particle tracking datasets.

This work was published: Ruoxi Sun, Liam Paninski. Scalable approximate Bayesian inference for particle tracking data, The International Conference on Machine Learning

(ICML) 2018. https://www.biorxiv.org/content/10.1101/276253v1. Code is available: https://github.com/SunRuoxi/Single_Particle_Tracking

2.1 Introduction

In many biological and physical experiments it is necessary to track the movement of many isolated particles in a video datastream. This is an essential task in biomedical research, for example, to reveal the biophysical properties of both the imaged particles (e.g., single molecules) and the biological substrate (e.g., cell membrane) that the particles are traversing. Effective particle tracking algorithms have wide applications in both fundamental and applied biology, and more generally in chemistry and physical applications.

Previous scalable approaches to this task have largely involved non-Bayesian methods aiming at estimating a single "best" path of the underlying particles. However, in many applications particles have indistinguishable shapes under light microscopic resolution. This leads to a fundamental non-identifiability: if two particles pass close by each other ("meet") then it is impossible to deterministically link the pre-meeting paths with the correct post-meeting paths (see Figure 2.1 below for an illustration). This motivates a Bayesian approach for assigning posterior probabilities over all the possible sets of particle paths consistent with the observed data.

Formally, at each timestep we observe a noisy, blurry image recording the particles' current positions. In the simplest case, we can cast the tracking task in a factorial hidden Markov Model (HMM) framework, where each particle evolves according to a Markov process and thus multiple HMMs (one per particle) jointly determine the observed image

data. The classic HMM inference approach is the forward-backward algorithm (Rabiner 1990), but the complexity of forward-backward scales superlinearly with the number of particles here.

In this work, we propose an amortized inference approach utilizing a specialized recurrent neural network architecture to approximate the posterior particle transition densities inferred by forward-backward. After network training, posterior inference can be performed very quickly: given a new video dataset, the network outputs the conditional particle initialization and transition densities, and then we can simply sample forward from the resulting Markov chain to draw samples from the posterior particle paths.

We apply the method to simulated and real data. We show that the method robustly performs approximate Bayesian inference on the observed data, and provides more accurate results than competing methods that output just a single "best" path. Our approach is much more scalable than previously proposed Bayesian approaches, scaling linearly in the number of frames and in the number of observed pixels.

2.2 Model

To set the stage we describe the simplest concrete model for particle tracking data; we will generalize this model below. We have J indistinguishable particles: each particle j appears at some time t_j^{appear} and disappears at some later time $t_j^{disappear}$. The particles move according to independent Gauss-Markov processes, with no interactions between particles. On each frame t we observe a blurred noisy sum of the particles that are visible at time t. The observation likelihood depends on the details of the experimental setup;

the most common model is the Gaussian blur + Poisson noise model:

$$
\begin{aligned}
Y(t,x) &\sim Poisson[\lambda(t,x) + \lambda_0] \\
\lambda(t,x) &= \sum_j G[x - s_j(t)],
\end{aligned}
$$

where $Y(t,x)$ denotes the image data observed at pixel x at time t, λ_0 is a background "dark noise" Poisson intensity, $G[.]$ is a Gaussian point spread function (psf), $s_j(t)$ represents the location of particle j at time t, and the sum is over all particles that are alive at time t.

The model described above is a factorial HMM (Ghahramani and Jordan 1996). However, this simple model can be generalized significantly. There may be multiple distinguishable classes of particles that have different shapes or colors. In many datasets particles can interact: they might merge, collide, split, etc. Individual particles often move in a non-Markovian manner (e.g., switching between several different latent dynamical modes). There may be strong dependencies between the motion of different particles, due e.g. to substrate motion. Finally, the observation noise may be highly non-Poisson, with correlations and strong inhomogeneities across the field of view. Thus it is critical to develop flexible inference approaches that do not depend on strong factorial HMM assumptions.

2.3 Related work on particle tracking

The literature on particle tracking methods is vast, and dates back to early physics studies of Brownian motion in fluids; see e.g.(Manzo and Garcia-Parajo 2015) for a review, and (Chenouard et al. 2014) for a quantitative comparison of many algorithms. We will not attempt to review all of these methods here, but note that many algorithms split the tracking problem into a "detect" followed by a "link" step. The "detect" step outputs estimated particle locations given each image Y_t. Various nonlinear filtering, thresholding, deconvolution, and neural network approaches have been employed for this task (ibid.). Most such detection algorithms take just single frames Y_t as input, and therefore they do not integrate useful information across multiple frames to perform detection; (Newby et al. 2017) is a recent counterexample that demonstrates that better performance can be achieved if multiple frames Y_t are utilized in the detection step.

The "link" step then attempts to fuse these detected locations, to estimate the tracks that each visible particle took over the length of the observed movie. This linkage step is solved by some matching algorithm; see e.g. (Jaqaman et al. 2008) for an influential example of this approach, and (Chenouard et al. 2014; Turner, Bottone, and Avasarala 2014; Wilson et al. 2016) for discussion of some other linking methods.

As we emphasized in the introduction, deterministic detection and linking approaches are statistically suboptimal, since they ignore the irreducible uncertainty of the tracking problem that results when two or more visibly indistinguishable particles pass closer than a fraction of a psf-width of each other. Ignoring this uncertainty leads to non-robust results, in which tiny changes to the data can lead to discontinuous changes in the estimated

particle tracks. Moreover, it is clear that the linkage and detection should not be separated: if we know the tracks of particles at times $(1 : t - 1)$ and $(t + 1 : T)$, then we have very strong prior information about the locations of particles at time t, and ignoring this useful prior information will lead to suboptimal results. (See e.g. Sun, Archer, and Paninski 2017, where similar points were made in the context of a related super-resolution application.)

Similar points have been made in the Bayesian signal processing literature; for example, sequential Monte Carlo (particle filtering) methods have been applied to perform probabilistic inference in this setting (Smal et al. 2008). These approaches have the advantage of a proper grounding in standard Bayesian computational methodology, but scale poorly in the number of visible particles.

Finally, there is also a very large literature on "multi-target tracking," e.g., tracking multiple people visible on security cameras. In this literature the different targets are typically distinguishable (e.g., different people visible on a camera will have different faces, gaits, clothing, etc.), whereas in this paper we focus on the case that the particles to be tracked are indistinguishable. Of course a middle ground exists in which particles have some distinguishing features but some posterior uncertainty about particle identity remains due to noisy or incomplete observations; however, to keep our presentation simple we focus exclusively on the most challenging fully-indistinguishable case here.

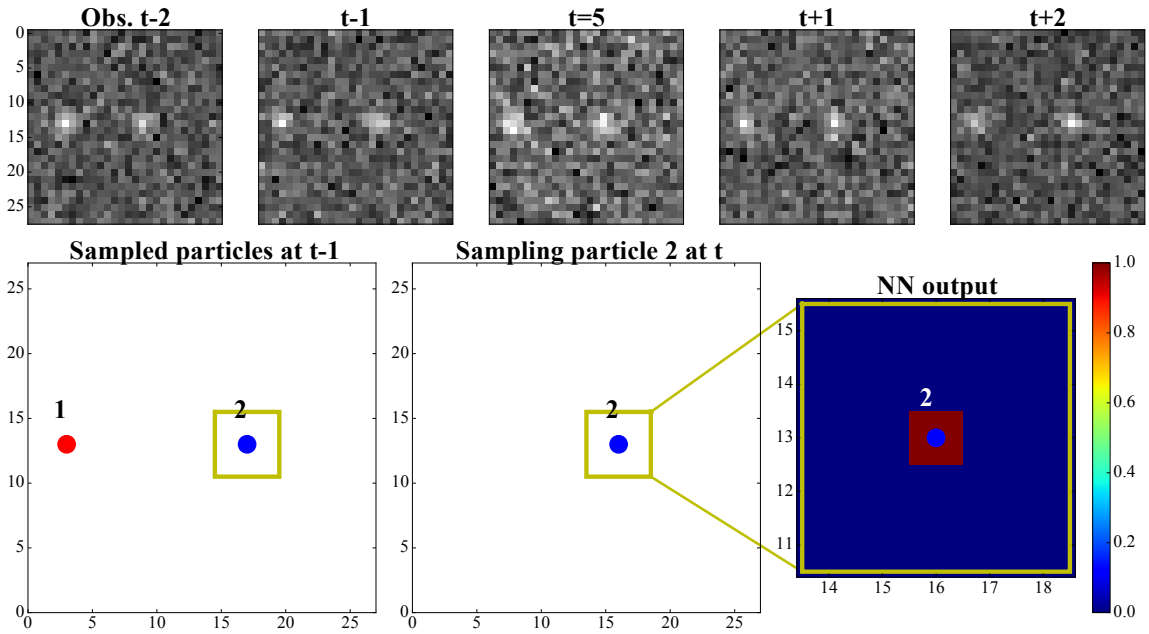

Figure 2.1: **Overview of the conditional transition density network**. Inputs to the network include the observed data $Y_{t-M:t+M}$, with $M = 2$ here (top), the locations of particles sampled at time $t-1$ (lower left), the particles that have been sampled so far at time t (lower middle), and the identity of the particle we are currently sampling (indicated by the yellow box). The network outputs the probability that the sampled particle survives to time t, and the conditional probability density of the particle's next position. See the sampling process video for further illustration of the network processing data. In this video, the particles are restricted to move in the horizontal direction only (to facilitate plotting of the results in the following section); different particles are marked by different colors. The lower right panel displays the probability map $p(s_t^i|q_{t-1}, \{q_t^j\}_{j<i}, Y)$ output by the network at each iteration.

2.4 Methods

Overview

Our conceptual starting point is the standard filter-backward-sample-forward algorithm for sampling from the posterior distribution $p(Q|Y)$ of the hidden state $Q = \{q_t\}$ of an HMM conditional on the observed data Y (Rabiner 1990). This algorithm has two steps: (1) combine the observed data Y with the prior distribution $p(Q)$ of the hidden Markov

state Q to obtain a new Markov chain $p(Q|Y)$, and (2) sample forward from this new Markov chain. Once (1) is complete, we can call (2) as often as we like to generate new sample paths from $p(Q|Y)$.

This approach is attractive in our setting because sampling forward from a Markov chain is a fast operation once the conditional initial and transition densities ($p(q_1|Y)$ and $p(q_t|Y, q_{t-1})$, respectively) are in hand, where the hidden state q_t is the configuration of the locations and identities of all of the particles alive at time t. Thus in principle we can simply run (2) repeatedly to compute probabilities of any quantity we care about (e.g., the probability that a particle is in location x at time t, or the probability that particle i in frame s should be linked with particle j in frame t).

Unfortunately, as emphasized above, computing (1) exactly is intractable in our context; thus we need to approximate the conditional initial and transition densities. Our strategy is to train neural networks to approximate these probabilities. This approach is highly flexible; given enough training data, we can handle a wide variety of non-standard data, well beyond the simplest Gaussian blur + Poisson noise factorial HMM described above, since the learned probabilities do not lean heavily on special assumptions about e.g. the noise model or the precise details of the graphical model underlying the data[1]. In turn, we can generate as much training data as we need by simulating ground truth particle tracks along with the resulting observed data videos Y.

It is convenient to split the network into three parts: the conditional transition density that governs how samples move from timestep t to $t+1$; the conditional birth density that

[1]The main assumption we make is that the posterior $p(Q|Y)$ can be well-approximated as Markovian, so that our resulting Markovian sampler can provide good approximations to true samples from the posterior. This assumption is reasonable in the majority of particle-tracking applications we have in mind.

governs the probability that a new particle appears at time t; and the conditional initial density that governs the positions of the particles at timestep 1. We describe each of these in turn below.

Conditional transition density network

This network is illustrated in Figure 2.1. The task of this network is to combine the observed data Y with the previous particle configuration q_{t-1} and to output probabilities that govern the particle configuration q_t in the next time step. This is a nontrivial task, since the dimensionality of q_t can be large and varies with time t as particles appear or disappear. Similarly, the observed image Y_t is often large (hundreds of pixels on a side), and in principle we need to observe multiple frames before and after time t to perform optimal inference.

Thus, for scalability, we break the problem up into a sequence of smaller pieces and work convolutionally. We begin by choosing a random ordering of the particles in q_{t-1}. Then, for each of these particles indexed by i, we input three types of data: (1) a local patch of the observed movie data (in a spatial neighborhood around the i-th particle location s^i_{t-1}, and in a temporal context of M frames before and after the current frame t; (2) a binary mask indicating the locations of the particles at time $t-1$ in the same spatial neighborhood as particle i; and (3) a second binary mask indicating the locations of the particles j that have been sampled at time t prior to sampling particle i[2]. The

[2]This input lets the network avoid placing two particles to explain a single observed bump in Y_t; if a previously-sampled particle j already explains the bump well, then the network will prefer to put particle i elsewhere. Also note that the input data Y and output probability maps don't need to have the same number of pixels (i.e., we could attempt to resolve the particle locations at higher spatial resolution than the observed data), but we have not pursued this direction in detail.

network is then trained to output a probability map $p(s_t^i | q_{t-1}, \{q_t^j\}_{j<i}, Y)$ indicating the likely location s_t^i, along with an auxiliary probability that the particle disappears (and is therefore no longer present at time t). Once these transition probabilities are learned, we can sample forward one particle and time-step at a time, as illustrated in the sampling process video and detailed in Algorithm 1; thus at test time inference scales linearly in the number of particles and time steps in the movie.

Note that we have slightly diverged from the vanilla filter-backward-sample-forward algorithm, which propagates information all the way back from the final observation Y_T to determine the state q_t. Instead, we exploit the fact that only a local context around time t is necessary to infer q_t, and thus we restrict our attention at time t to the local context $Y_{t-M:t+M}$. (We use $M = 2$ throughout this paper.)

Algorithm 2 Conditional sampling network

```
Initialization  S_1 = Initializer(Y_{1:M+1}, [])
for t = 2, 3, 4... do
    S_t = []
    for i in Permutation{S_{t-1}} do
        p_i = ConditionalProbability(Y_{t-M:t+M}, S_{t-1}, S_t, i)
        particle i disappears with prob. 1 - ∫ p_i
        otherwise i' is sampled from p_i
        Insert i' to S_t
    end
    N_t = NewBirth(Y_{t-M:t+M}, S_{t-1}, S_t) Insert N_t to S_t
    S_{t-1} = S_t
end
```

New birth networks and initialization

The network described above moves particles forward from timestep $t-1$ to t, and decides which particles should disappear at time t. However, new particles can enter the field of

view at any time, and therefore we need a method for adding new particles to q_t. Thus after running the update described above to q_t, we run a second convolutional network that takes the same inputs as above (i.e., the local context of q_{t-1}, q_t, and Y, now at each location in the image instead of just at the previously-sampled particle locations) and outputs the probability that a new particle is born at each location at time t. Then we iteratively sample from this density and update q_t until no further particles are added.

The same strategy can be used to initialize q_1; the only difference is that the inputs now don't include q_{t-1} or the context of Y prior to Y_1.

Network architecture and training

To handle the temporal and spatial dependencies in this data, we chose a combination of bi-directional 2D convolution LSTM and 3D convolutional layers; see Appendix. Overall, when the network is sampling forward, we can think of the resulting algorithm as a recurrent neural network (since the sampled output is then read back into the network to define the next state transition), with the somewhat non-standard feature that the network remains at timestep t for a random number of iterations (depending on how many particles need to be updated and how many particles are born at each timestep).

To train the network we generated simulated ground truth particle tracks q_t and corresponding observed movies Y. (We will discuss the training data in more detail in the following section.) Then we formed minibatches of training data, where each data sample included the inputs to the network (the local context of Y, q_{t-1}, and a random subset of q_t) along with the true particle location s_t^i, which served as the target output of the net-

work. We trained the network (using default learning rate settings in Keras) to minimize the binary cross-entropy between the target mask (zero except at s_t^i, or all zeros if all the particles in q_t were already sampled and no further particles should be added) and the network's output probability mask. Code is available here.

2.5 Results

One-dimensional example

We begin with a simple simulated experiment in which the particles are restricted to move in the horizontal direction only. This makes it easier to view and understand the results, by simply plotting the horizontal positions of the (true vs. inferred) particles as a function of time. The results are illustrated in Figure 2.2; the same data are shown in Figure 2.1 and the sampling process video.

In this example we see the appearance and disappearance of a couple particles, and two "meeting" events in which one particle overlaps significantly with another particle. Since in this example all the particles have identical shapes and are undergoing independent and identically distributed Brownian motions, there is no way to deterministically "link" particles before and after these meeting events; i.e., the "correct" linker here must output a probabilistic answer.

In panels 2-4 of Figure 2.2 we display three conditional sample paths drawn by our algorithm. Sample 0 (panel 2) recovers the ground truth accurately, and Sample 1 and 2 (panels 3 and 4) give different — but also valid — sets of tracks. Panel 5 shows an average

of 100 samples overlaid together, with the colors indicating relative probabilities of the chosen tracks. Note that at the beginning of the trial, where the two visible particles are well-isolated, the sampler essentially outputs a deterministic estimate, with all samples assigned to the left (red) or the right (blue). However, after the "meeting" near $t = 15$, the colors blend, indicating probabilistic assignment of tracks following this event, as desired.

For comparison, we also show the output of two existing particle tracking methods, both of which output deterministic particle identities. Our approach provides visibly more robust outputs on this example, with fewer dropped particle detections and false particle appearances or disappearances.

Two-dimensional example

Next we turn to a small-scale simulated two-dimensional example; the results are illustrated in the moving particles video, Figure 2.3, and the 3D view video. As in the previous one-dimensional example, we find that our proposed approach accurately detects the particle locations and appearance/disappearance times, and successfully assigns identities probabilistically following particle meetings.

Large scale examples and evaluation

To establish a more quantitative evaluation, we compared against two baseline methods: the popular Utrack approach (Jaqaman et al. 2008) and the method proposed in (Wilson et al. 2016), which performed well on the performance metrics established in the review / competition paper (Chenouard et al. 2014). We generated large-scale two-dimensional

simulated data whose parameters matched a pair of challenging datasets in (Chenouard et al. 2014), and then computed the suite of performance metrics (measuring various facets of detection accuracy, linking quality, etc.) introduced in the same paper (averaging over 100 draws from our sampler for each dataset). Results are shown in Table 1: we find that our proposed method outperforms the baselines on both datasets examined, on all the performance metrics computed here.

It is worth emphasizing that these performance metrics were designed for deterministic tracking algorithms, and therefore entirely miss one of the major advantages of our approach (the fact that it outputs not just a single "best" track estimate but instead estimates the posterior distribution over all tracks). How can we evaluate the quality of our approximation to the posterior here (and quantitatively compare between different algorithms that attempt to approximate this posterior)? One natural approach is to estimate the Kullback-Leibler divergence $D_{KL}[f(Q); p(Q|Y)]$ between our approximate posterior $f(Q)$ and the true posterior $p(Q|Y)$ on the state space Q given the observed data Y. Of course, this is not quite tractable, due to the intractability of $p(Q|Y)$, but we can estimate $D_{KL}[f(Q); p(Q|Y)]$ up to a constant in $f(Q)$ by sampling from $f(Q)$:

$$\begin{aligned} D_{KL}[f(Q); p(Q|Y)] &= E_{f(Q)} \log \frac{f(Q)}{p(Q|Y)} \\ = E_{f(Q)} \log \frac{f(Q)}{p(Q)p(Y|Q)} &+ const[f(Q)] \\ \approx \frac{1}{N} \sum_{i=1}^{N} \log \frac{f(Q^i)}{p(Q^i)p(Y|Q^i)} &+ const[f(Q)], \end{aligned}$$

where $\{Q^i)\}_{i=1:N}$ are N samples from $f(Q)$. Here $p(Q^i)$ and $p(Y|Q^i)$ can be evaluated

	SNR=5						
	Assign error	RMSE	COV	Alpha	Beta	JSC	JSC_θ
Network	$\mathbf{7.28 \pm 1.05}$	$\mathbf{0.08 \pm 0.01}$	$\mathbf{0.69 \pm 0.05}$	$\mathbf{0.82 \pm 0.06}$	$\mathbf{0.81 \pm 0.03}$	$\mathbf{0.68 \pm 0.06}$	$\mathbf{0.99 \pm 0.02}$
Utrack	8.88	0.189	0.574	0.778	0.748	0.523	0.769
Wilson	11.83	0.214	0.385	0.704	0.667	0.339	0.741
	SNR=4						
Network	$\mathbf{8.23 \pm 1.12}$	$\mathbf{0.07 \pm 0.01}$	$\mathbf{0.63 \pm 0.07}$	$\mathbf{0.76 \pm 0.03}$	$\mathbf{0.70 \pm 0.04}$	$\mathbf{0.55 \pm 0.08}$	$\mathbf{0.82 \pm 0.05}$
Utrack	12.78	0.10	0.32	0.62	0.59	0.29	0.68
Wilson	14.42	0.34	0.36	0.58	0.55	0.32	0.63

Table 1. Comparison of three particle tracking methods: our proposed approach ("network"), Utrack from (Jaqaman et al. 2008), and the method proposed in (Wilson et al. 2016). Bold indicates best performance; we find that the proposed network approach achieves the best performance over both datasets and all performance metrics computed here. RMSE: Root Mean Square Error; COV: Coverage; JSC: Jaccard similarity coefficient; all quantities are as defined in (Chenouard et al. 2014).

explicitly if the prior $p(Q)$ is e.g. Markovian; for our approach $f(Q^i)$ can also be evaluated directly since $f(Q)$ has an explicit Markov form. This provides us a method for scoring any Bayesian particle tracking algorithm for which we can explicitly evaluate the approximate posterior $f(Q)$. (We do not perform this scoring on the baselines examined here, since for any deterministic algorithm $f(Q)$ is a delta function, leading to an infinite Kullback-Leibler score if we treat q_t as a continuous random variable — i.e., the probabilistic approach trivially outperforms deterministic approaches.)

Real data example

Finally, we tested the performance of our algorithm on real data. The data are TIR-FM imaged clathrin-coated pits in a BSC1 cell (Jaqaman et al. 2008). We trained the network on simulated data whose parameters (signal-to-noise ratio, particle density and speed, psf width, etc.) were coarsely matched to the real data; see the comparison video for details. We plot three samples from our algorithm using different colors in Fig. 2.4 and the real data

video. While ground truth is unavailable in this case, by visual inspection the algorithm seems to effectively follow the particles in the video, without excessive oversegmentation of the tracks; the output here seems consistent with the behavior of the algorithm on the previous simulated datasets.

2.6 Discussion

Related machine learning work

In the introduction we emphasized the importance of the particle tracking problem; we believe that the more robust, accurate, and probabilistic tracking methods developed here will have a significant impact in a wide range of biological and physical applications.

More generally, from a machine learning point of view, the major novelty of our work is the incorporation of neural network methods to provide a flexible and scalable approximation of Bayesian inference via efficient sampling in a large graphical model.

Of course, interactions between Bayesian analysis and neural network methods comprise a very rich thread of research these days. The work of (Snell and Zemel 2017) is highly relevant: this paper describes a neural network approach to sample multiple segmentations that are consistent with an observed image, much as we use neural networks to sample multiple particle tracks that are consistent with an observed video.

As another example, variational autoencoders Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014 and variants thereof Fraccaro et al. 2017; Gao et al. 2016; Johnson et al. 2016; Krishnan, Shalit, and Sontag 2017 have become very popular recently

for performing inference in nonlinear HMMs. These methods are most effective when the latent state variable is low-dimensional. In the particle tracking problem the latent dynamical variable is very high-d (scaling with the number of particles) and more importantly the latent dimensionality is time-varying, as particles are born, die, merge, split, enter, or leave the focal plane. We are not aware of variational autoencoder approaches that would be easily applicable to the particle tracking problem.

Another related thread involves amortized inference using neural networks for sequential Monte Carlo; see e.g. Paige and Wood 2016. Again, it is not clear how well these methods would scale to the large-scale multiple-particle tracking problems of interest here.

Finally, our work is an example of a broad theme in the current image processing literature: start with "ground truth" images, then simulate observed data that can be generated as some kind of corruption of this ground truth, and then use this simulated data to train a neural network that can "denoise" (or super-resolve, or deblur, or infill, etc.) this corruption. A (highly non-exhaustive) list of recent examples includes: Parthasarathy et al. 2017, which applies this idea to approximate Bayesian decoding of neuronal spike train data; Yoon et al. 2017, to segmentation of three-dimensional neuronal images; and Weigert et al. 2017, to denoising of microscopy images.

Future work

At test time, as emphasized above, the inference approach proposed here is highly scalable, but the network training time is relatively slow (taking on the order of hours for the

experiments presented here). This is typical of "amortized inference" approaches: we pay with relatively long training times for fast test times. Thus our proposed approach is most valuable in settings where we have repeated experimental samples from a similar data regime (instead of training a new inference network for each new experimental dataset).

We have not expended serious effort optimizing over network architectures here; we could likely find lighter architectures that perform similarly, which would speed up both testing and training. Similarly, we could distill/compress the network to further speed up test times, if necessary e.g. for online experimental designs.

Similarly, we have not yet attempted to develop automated procedures for choosing parameters for generating training data. In practice we have found that these parameters (e.g., the amplitude, density, variance/speed of particles, plus noise levels, point-spread width, etc.) are fairly straightforward to choose, and the inference results are not highly sensitive to small misspecifications of these parameters (recall Figure 2.4 and the corresponding comparison video). It would be useful to develop a simple interface that would allow experimentalists to easily generate training data, followed by generation of a network trained to perform inference on their data.

An alternative approach would be to include data parameters as extra inputs for the network. Then in principle there would be no need to train a new network for each new type of data; instead we could perhaps just train a single big network on many different data types (with the corresponding data parameters included as inputs to the network) and then when presented with a new datatype we just provide the network with the required parameters and let it perform inference. This is an ambitious but important direction for

future work[3].

[3]Note that a slightly different philosophy is espoused in Newby et al. 2017, who trained a single deterministic network for particle detection that can be applied to a wide range of data, but without including any parameters describing the data generation mechanism as inputs to the network. This approach makes it easy for experimentalists to use the network (since no training or parameter estimation is required), but likely sacrifices some accuracy compared to a network that is provided information about the parameters governing the generation of the data. We hope to run more detailed comparisons of these approaches in the future.

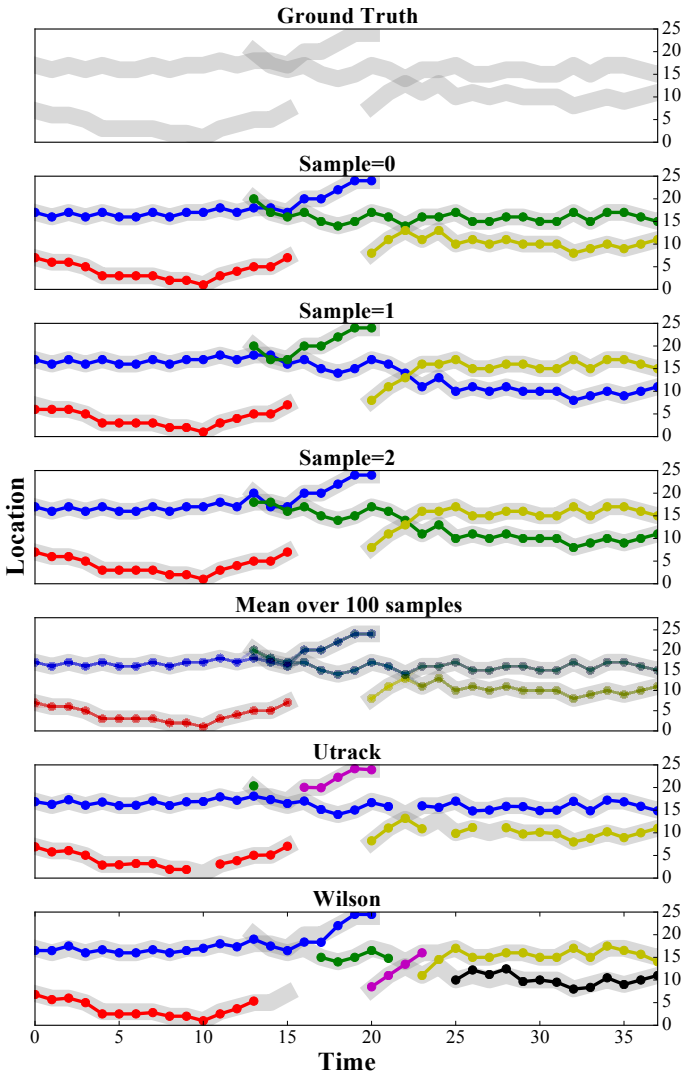

Figure 2.2: **One-dimensional simulated example**. We test the performance of the proposed algorithm on a simplified example where the particles are restrained to move in one dimension, to facilitate visualization. See the sampling process video for the raw data. **Top**: Ground truth tracks. New particles appear near $t = 13$ and $t = 20$; a particle disappears near $t = 20$; "meetings" between two particles occur near $t = 14$ and $t = 22$. **Panels 2-4**: sample tracks output by our proposed method. Colors indicate particle identity. Note that the detected locations track the ground truth locations and appearance/disappearance times accurately, and identity is assigned probabilistically following particle meetings, as desired. (The network output corresponding to Sample 0 is shown in Figure 2.1 and the sampling process video, with colors matched across the figures and video.) **Panel 5**: mean over 100 examples; the blended colors following particle meeting times indicate the relative probabilities of the identity assignments. **Bottom two panels**: output from two deterministic particle tracking methods, by (Jaqaman et al. 2008) and (Wilson et al. 2016), respectively. Several detection errors are visible in the output of these methods, leading to oversegmentation of the output tracks.

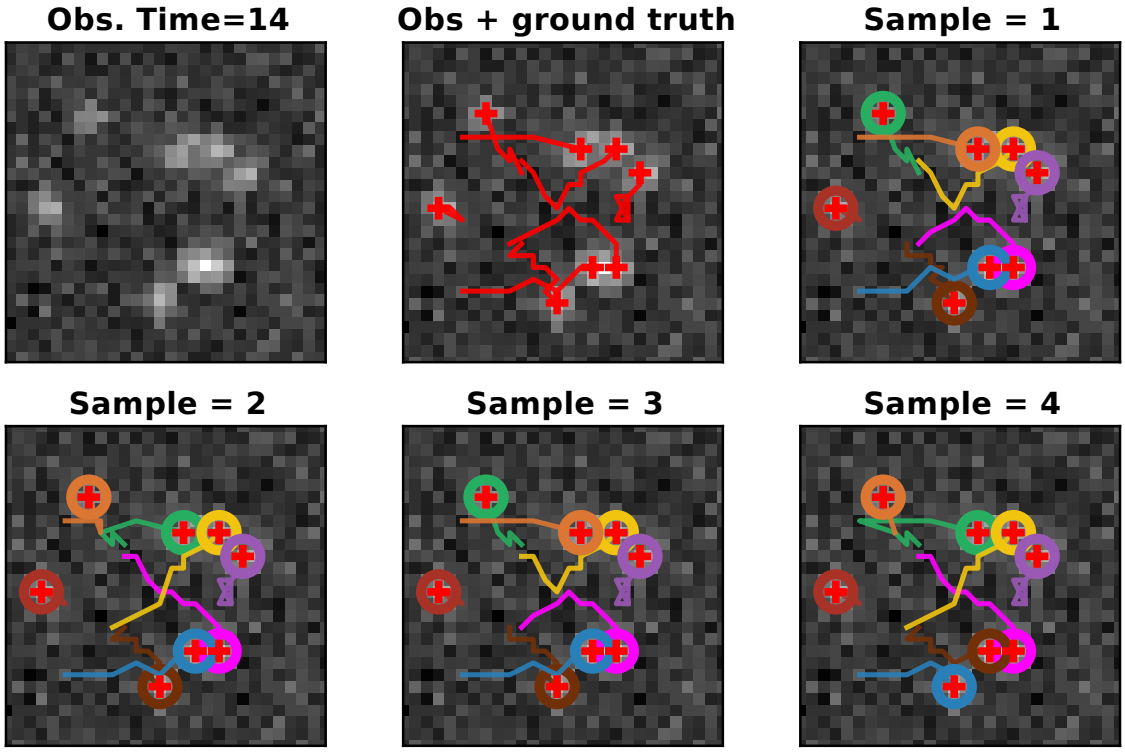

Figure 2.3: **Two dimensional example**. This figure displays a single frame of the moving particles video; view the video for further details. **First panel**: observed data Y_t. **Top middle**: a time-lapse trace of the ground truth particle tracks; plus marks the current particle position, and the tails mark the recent history. **Other panels**: four sample estimated particle tracks output by our proposed method. As can be seen in the moving particles video, as particles meet the particle identities are assigned probabilistically and the identities across different samples diverge, even if some identities were initially the same across some samples. (Colors of sampled tracks are assigned according to total proximity of each track to the ground truth tracks, each of which is assigned a random color.) We show another representation of these samples in the 3D view video; the first frame of this video shows the ground truth tracks and each remaining frame shows a single sample in 3d (two spatial dimensions and one time dimension), with colors matched to those shown in this figure and in the moving particles video; thick lines indicate ground truth tracks for comparison.

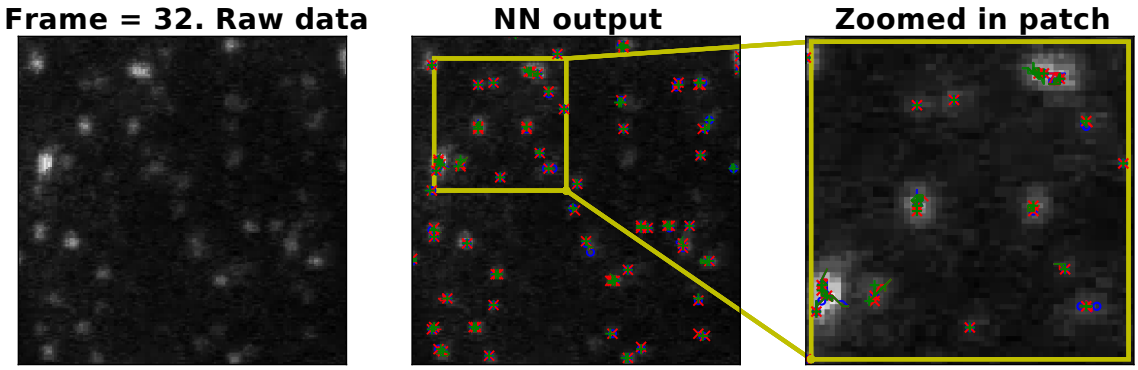

Figure 2.4: **Real data**. Performance on real data (TIR-FM imaged clathrin-coated pits in a BSC1 cell) (Jaqaman et al. 2008). **Left**: raw image sequence. **Middle**: raw image sequence overlaid with detection markers and tails indicating the recent location history. Colors indicate three different samples from our algorithm. **Right**: a zoomed in patch. Image size: 150×170 pixels, pixel size: 67 nm. For more details see the real data video.

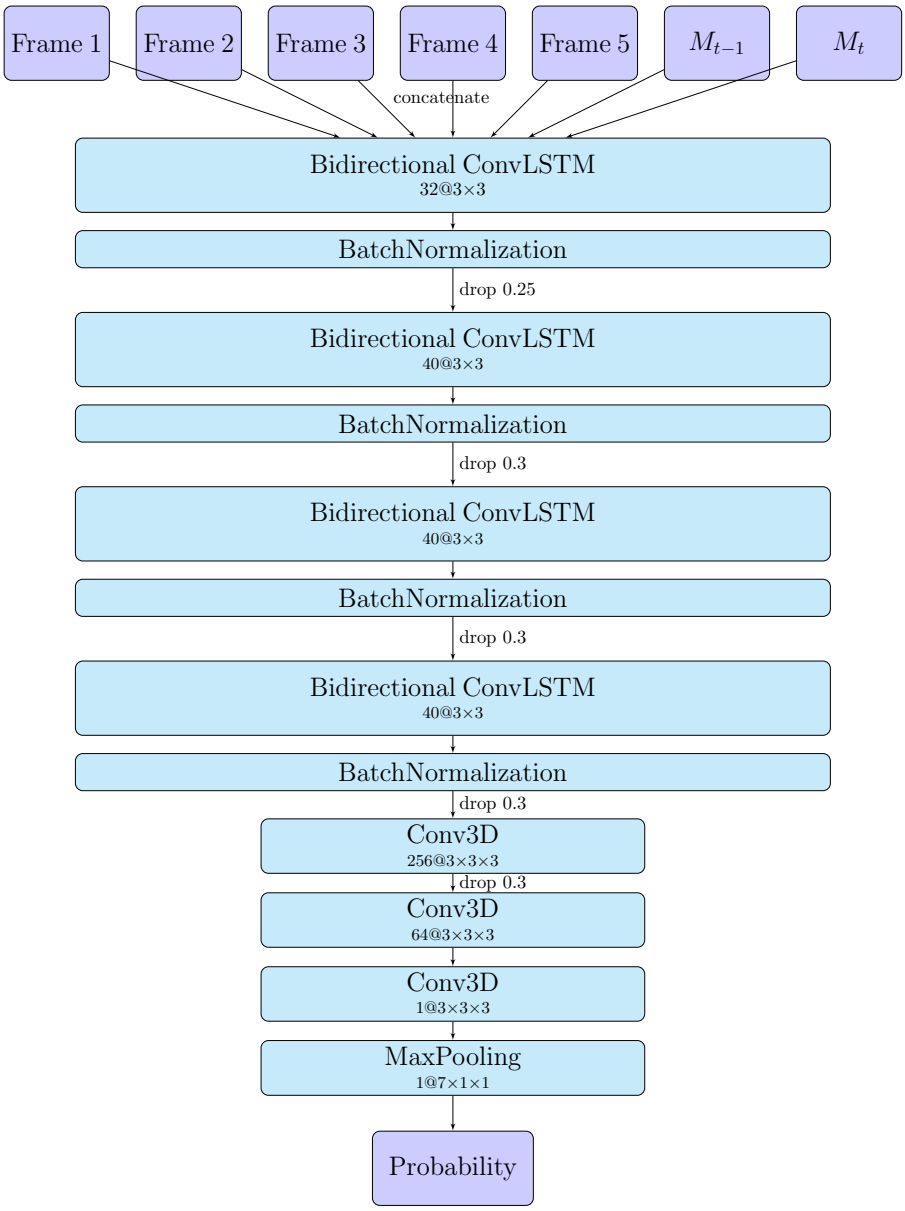

Figure 2.5: **Neural network architecture**. This is the architecture of the conditional transition density network described in section 2.4. We use Bidirectional Convolution LSTM and Convolution 3D Neural Networks as building blocks with notation '40@3x3' meaning 40 feature maps with kernel size 3 by 3. The input size of the networks is $7 \times patch\_size \times patch\_size$: 5 patches for $Y_{t-2:t+2}$, plus binary masks M_{t-1} and M_t encoding the positions of the particles at times $t-1$ and t respectively. The output probability map is $patch\_size \times patch\_size$. We used $patch\_size = 28$ here. The new birth network in section 2.4 uses a similar architecture.

Chapter 3

Voltage Smoothing

Recent advances in optical voltage sensors have brought us closer to a critical goal in cellular neuroscience: imaging the full spatiotemporal voltage on a dendritic tree. However, current sensors and imaging approaches still face significant limitations in SNR and sampling frequency; therefore statistical denoising methods remain critical for understanding single-trial spatiotemporal dendritic voltage dynamics. Previous denoising approaches were either based on an inadequate linear voltage model or scaled poorly to large trees. Here we introduce a scalable fully Bayesian approach. We develop a generative nonlinear model that requires few parameters per dendritic compartment but is nonetheless flexible enough to sample realistic spatiotemporal data. The model captures potentially different dynamics in each compartment and leverages biophysical knowledge to constrain intra- and inter-compartmental dynamics. We obtain a full posterior distribution over spatiotemporal voltage via an efficient augmented block-Gibbs sampling algorithm. The nonlinear smoother model outperforms previously developed linear methods, and scales to much larger systems than previous methods based on sequential Monte Carlo approaches.

This is joint work with Scott Linderman and Liam Paninski, and Ian. Scott Linderman contributed significantly to the project from all perspectives. Ian helped with quantitative

comparisons with other approaches.

This work was submitted to Conference on Neural Information Processing Systems (NeurIPS) 2019. Code is in preparation.

3.1 Introduction

Recent progress in the development of voltage indicators (Abdelfattah et al. 2018; Chavarha et al. 2018; Hochbaum et al. 2014; Kannan et al. 2018; Marshall et al. 2016; Miyazawa et al. 2018; Piatkevich et al. 2018) has brought us closer to a long-standing goal in cellular neuroscience: imaging the full spatiotemporal voltage on a dendritic tree. These recordings have the potential (pun not intended) to resolve fundamental questions about the computations performed by dendrites — questions that have remained open for more than a century (Cajal 1911; Stuart, Spruston, and Häusser 2016). Unfortunately, despite accelerating progress, currently available voltage indicators and imaging technologies provide data that is noisy and sparse in time and space. Our goal in this work is to take this noisy, sparse data and output Bayesian estimates, with uncertainty, of the spatiotemporal voltage on the tree, at arbitrary resolution.

A number of generic denoisers are available. For example, one previous approach is to run an independent spline smoother on the temporal trace from each pixel (Hochbaum et al. 2014). However, this approach ignores two critical features of the data. First, the data is highly spatiotemporally structured; thus, running a purely temporal smoother and ignoring spatial information (or vice versa) is suboptimal. Second, the smoothness of voltage data is highly inhomogeneous; for example, action potentials are much less

smooth than are subthreshold voltage dynamics, and it is suboptimal to enforce the same level of smoothness in these two very different regimes.

How can we exploit our strong priors, based on decades of biophysics research, about the highly-structured dynamics governing voltage on the dendritic tree? Our starting point is the cable equation (Tuckwell 1988), the partial differential equation that specifies the evolution of membrane potential in space and time. If we divide up the tree into N discrete compartments, then letting $V_t^{(n)}$ denote the voltage of compartment n at time t, we have

$$V_{t+\Delta t}^{(n)} \approx V_t^{(n)} + \frac{\Delta t}{C_n}\left[\sum_j I_t^{(n,j)} + \sum_{n'=1}^{N} g_{nn'} \cdot (V_t^{(n')} - V_t^{(n)})\right]. \tag{3.1}$$

Each compartment n has its own membrane capacitance C_n and internal currents $I_t^{(n,j)}$; j indexes membrane channel types, with $j = 0$ denoting the current driven by the membrane leak conductance. The currents through each channel type for $j > 0$ in turn depend on the local voltage $V_t^{(n)}$ plus auxiliary channel state variables with nonlinear, voltage-dependent dynamics. The coupling of voltage and channel state variables renders the intra-compartment dynamics highly nonlinear.

Additional current flows between compartments n and n' according to the conductance $g_{nn'} \geq 0$ and the voltage drop $V_t^{(n')} - V_t^{(n)}$. The conductances are undirected so that $g_{nn'} = g_{n'n}$. The symmetric matrix of conductances $G = \{g_{nn'}\} \in^{N \times N}$ specifies a weighted, undirected tree graph. Nonzero entries indicate the strength of connection between two physically coupled compartments: if $g_{nn'}$ is large then voltage differences between compartments n and n' are resolved quickly, i.e. their voltages become more

tightly coupled.

The HH model (Hodgkin and Huxley 1952) and its generalizations (Koch 2004) offer biophysically detailed models of voltage dynamics, but learning and inference pose significant challenges in the resulting high-dimensional nonlinear dynamical system. Two approaches have been pursued in the past. First, we can restrict attention to the subthreshold regime, where the dynamics can be approximated as linear. Invoking the central limit theorem leads to a Gaussian approximation on the current noise (due to the sum over j in Eq. 3.1), resulting in an overall linear-Gaussian model. If the observed data can in turn be modeled as a linear function of the voltage plus Gaussian noise, we are left with a classical Kalman filter model. (**?**) develops efficient methods to scale inference in this Kalman filter model to handle large trees. However, the resulting smoother doesn't handle spikes well — it either over-smooths spikes or under-smooths subthreshold voltages. The Kalman model suffers because it assumes voltage has one uniform smoothness level, and as already discussed, this assumption only makes sense in the subthreshold regime.

Alternatively, we can attempt to perform inference on noisy voltage recordings based on model (3.1) directly. There are many compartments, each with a voltage and a collection of channel state variables, leading to a very high-dimensional nonlinear dynamical system. For low-dimensional models, like single compartment models with few channel states, methods like SMC and ABC can be applied (Huys and Paninski 2009; Lueckmann et al. 2017; Meng, Kramer, and Eden 2011), but even with recent advances (Le et al. 2018; Maddison et al. 2017; Naesseth et al. 2018), inference in large scale biophysical models remains difficult. The learning problem (i.e. estimating the parameters governing the intra- and inter-compartment conductances) is even harder: inaccurate state inferences

lead to errors in parameter estimation, and poor parameter estimates lead to incorrect state inferences. Compounding all of this is model misspecification — critical parameters such as time constants and voltage-sensitivity functions vary across channels, and there are dozens of types of channels in real cells (Koch 2004)—and model identifiability — many channel combinations can produce similar dynamics (O'Leary et al. 2013). Thus performing inference on the multi-compartment biophysical model (3.1) directly seems intractable.

Below we propose an alternative approach, blending the cable equation model with general purpose statistical models of nonlinear dynamical systems, to enable efficient learning and inference of spatiotemporal voltage dynamics.

3.2 Model

Our basic strategy is as follows. For each compartment n, the biophysical model in (3.1) involves potentially dozens of channel types, each with their own state variables evolving according to nonlinear dynamics. But we only actually need the sum of their induced currents. We replace this sum with a simpler, low-dimensional effective model. We retain the basic spatial biophysical constraints on voltage dynamics (i.e., leaving the second term in model (3.1) as is), while approximating the nonlinear interactions between voltage and membrane channels with a more tractable model: a rSLDS. Figure 3.1 shows how each compartment is given its own discrete states, continuous states, and voltage, and how these variables interact to produce nonlinear spatiotemporal dynamics.

For each compartment n, we introduce $x_t^{(n)} \in^D$, a continuous latent state of dimen-

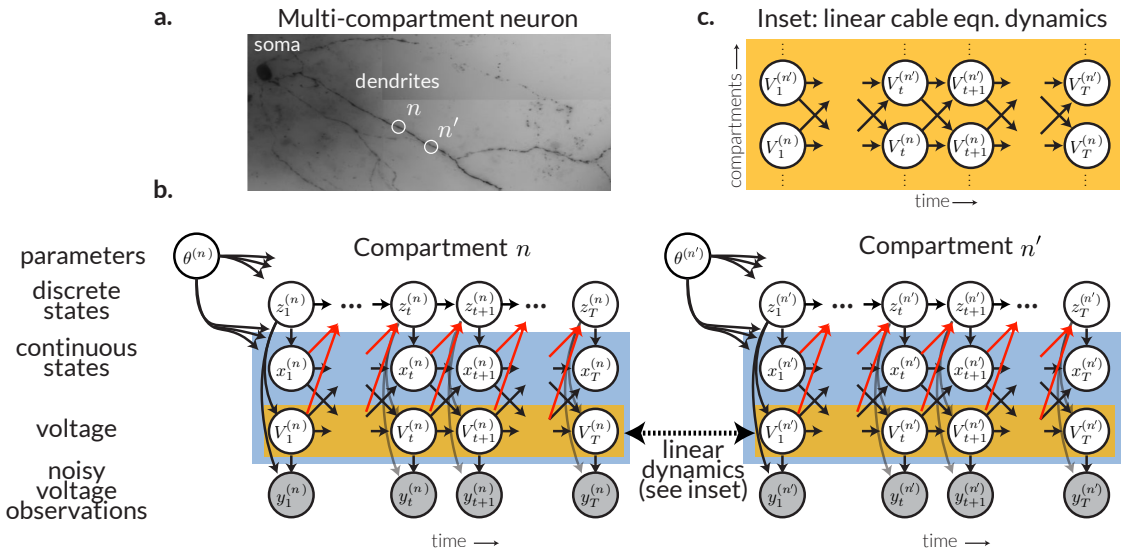

Figure 3.1: *Approximating the biophysical model of membrane potential dynamics with a tractable graphical model.* **a.** We study voltage dynamics on dendritic trees like the one shown here. **b.** Each compartment of the cell is approximated with a recurrent switching linear dynamical system, which has discrete latent states, continuous latent states, true (unobserved) voltage, and noisy voltage observations. The continuous states and voltage follow linear dynamics conditioned on the discrete states; marginalizing over the discrete states, we obtain nonlinear dynamics within each compartment. The red arrows denote the *recurrent* dependencies by which continuous states and voltage modulate discrete transition probabilities. **c.** Importantly, the inter-compartmental voltage dependencies are linear, as specified by the cable equation.

sion D ($D = 1$ in all the examples shown below, but higher-dimensional dynamics are possible). Let $z_t^{(n)} \in \{1, \ldots, K\}$ denote a corresponding discrete latent state. Given the discrete state, the voltage and continuous latent, $(V_t^{(n)}, x_t^{(n)})^\top$, together follow linear dy-

namics,

$$\left[\begin{pmatrix} V_{t+\Delta t}^{(n)} \\ x_{t+\Delta t}^{(n)} \end{pmatrix} \,\middle|\, z_t^{(n)} = k,\ \left\{V_t^{(n')}\right\}_{n' \neq n}\right]$$

$$= \begin{pmatrix} V_t^{(n)} \\ x_t^{(n)} \end{pmatrix} + \frac{\Delta t}{C_n}\left[\underbrace{A_k^{(n)}\begin{pmatrix} V_t^{(n)} \\ x_t^{(n)} \end{pmatrix} + b_k^{(n)}}_{\approx \sum_j I_t^{(n,j)}} + \begin{pmatrix} \sum_{n'=1}^{N} g_{nn'} \cdot \left(V_t^{(n')} - V_t^{(n)}\right) \\ 0 \end{pmatrix}\right].$$

The dynamics matrix $A_k^{(n)} \in^{(D+1)\times(D+1)}$ and the bias vector $b_k^{(n)} \in^{D+1}$ parameterize linear dynamical systems for each discrete state in each compartment. We further assume additive Gaussian dynamics noise for each compartment and discrete state with covariance $Q_k^{(n)}$. Importantly, the voltage dynamics retain the inter-compartmental linear terms from the cable equation, linking connected compartments in the dendritic tree (inset of Fig. 3.1). On the other hand, the nonlinear summed intra-compartment currents $\sum_j I_t^{(n,j)}$ are replaced with a collection of linear dynamics on voltage and continuous states; by switching between these discrete linear dynamics, we can approximate the nonlinear dynamics of the original model (since any sufficiently smooth function can be approximated with a piecewise-linear function).

To complete the dynamics model, we must specify the dynamics of the discrete states $z_t^{(n)}$. We use a rSLDS, allowing the discrete states to depend on the preceding volt-

age and continuous states,

$$p(z_{t+\Delta t}^{(n)} = k \mid V_t^{(n)}, x_t^{(n)}) \propto \exp\left\{ w_k^{(n)\top} \begin{pmatrix} V_t^{(n)} \\ x_t^{(n)} \end{pmatrix} + d_k^{(n)} \right\}. \tag{3.2}$$

The red arrows in Fig. 3.1 highlight these dependencies. The linear hyperplanes defined by $\{w_k^{(n)}, d_k^{(n)}\}_{k=1}^K$ define a weak partition of the space of voltages and continuous latent states. As the magnitude of the weight vectors increases, the partition becomes more and more deterministic. In the infinite limit, the discrete states are fully determined by the voltage and continuous states, and the switching linear dynamical system becomes a piecewise linear dynamical system (Sontag 1981). We find that these models admit tractable learning and inference algorithms, and that they can provide a good approximation to the nonlinear dynamics of membrane potential.

Finally, we observe noisy samples of the voltage $y_t^{(n)} \sim (V_t^{(n)}, {\sigma_k^{(n)}}^2)$ for each compartment, with state-dependent noise. Our goal is to learn the parameters of the multi-compartment rSLDS, $\Theta = \{\{\theta^{(n)}\}_{n=1}^N, G\}$, where $\theta^{(n)} = \{A_k^{(n)}, b_k^{(n)}, Q_k^{(n)}, w_k^{(n)}, d_k^{(n)}, \sigma_k^{(n)}\}_{k=1}^K$. Given the learned parameters, we seek a Bayesian estimate of the voltage given the noisy observations.

3.3 Bayesian learning and inference

The rSLDS inherits some of the computational advantages of the standard SLDS; namely, the conditional distribution of the discrete states given the continuous states is a chain-structured discrete graphical model, and most model parameters admit conjugate updates.

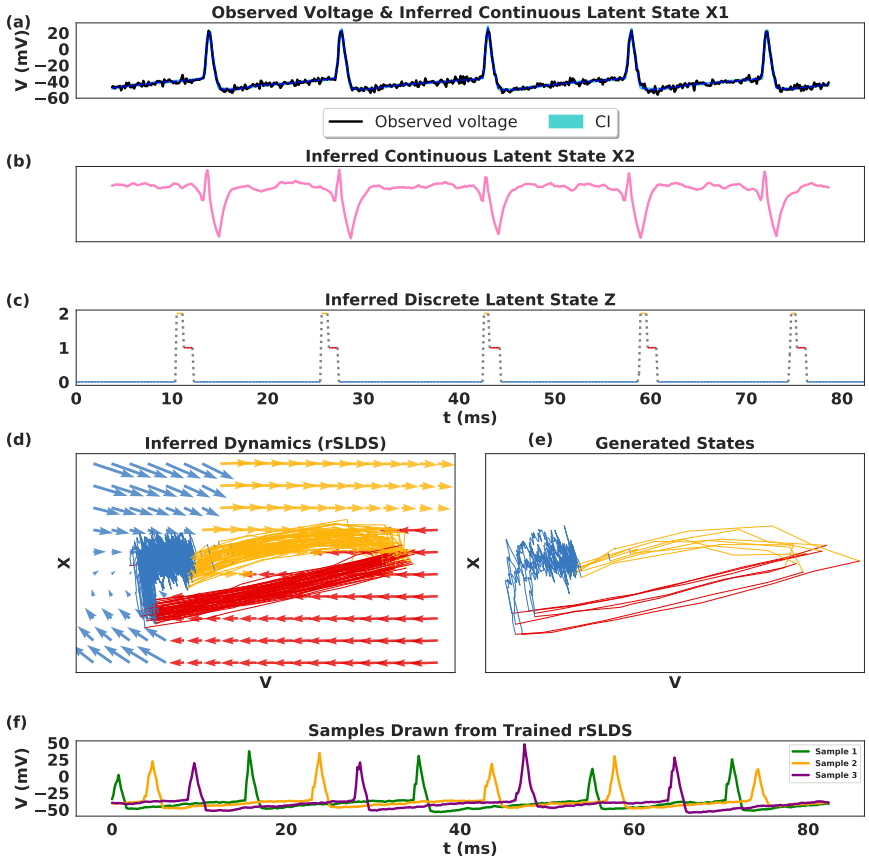

Figure 3.2: *A single-compartment, three-state rSLDS provides an adequate model of real somatic voltage responses.* (a) Observed voltage (formed by adding noise to the intracellular voltage from a real neuron) and the credible interval (CI) output by a the estimated rSLDS model. (b) Inferred continuous latent state $x(t)$ corresponding to this trial. (c) Inferred discrete latent state $z(t)$. (d) Inferred two-dimensional dynamics. The blue state (corresponding to $z = 0$ in (c)) is a fixed point at the rest potential; the yellow state corresponds to a fast depolarization (the upswing of the spike; $z = 2$) and the red state the hyperpolarization (the downswing of the spike; $z = 1$), followed by a return to the blue rest state. The thin traces indicate samples from $x(t)$ and $V(t)$ given the observed noisy voltage data. (e) Generative samples from the learned rSLDS; the difference between these traces and those shown in the previous panel is that these traces are generated using the learned rSLDS parameters without conditioning on the observed noisy voltages $\{y_t\}$, whereas in (d) we show samples from the posterior given $\{y_t\}$; the fact that the two sets of traces are similar is a useful check that the model fits have converged. (f) Voltages sampled from the rSLDS prior, corresponding to traces shown in (e). Note that the simple three-state two-dimensional rSLDS is able to learn to produce reasonably accurate spike shapes and firing rates.

However, the additional dependency in (3.2) breaks the linear Gaussian structure of the conditional distribution on voltage and continuous latent states.

Following previous work (Linderman et al. 2017; Nassar et al. 2019), we use Pólya-gamma augmentation (Polson, Scott, and Windle 2013) to render the model conjugate and amenable to an efficient Gibbs sampling algorithm. Briefly, we introduce augmentation variables $\omega_{tk}^{(n)}$ for each compartment, discrete state, and time bin. After augmentation, the voltage and continuous states are rendered conditionally linear and Gaussian; we sample them from their complete conditional with standard message passing routines. Moreover, the augmentation variables are conditionally independent of one another and Pólya-gamma distributed; we appeal to fast methods (Windle, Polson, and Scott 2014) for sampling these. The discrete states retain their chain-structured conditionals, just as in a hidden Markov model. Finally, the model parameters all admit conjugate Gaussian or matrix-normal inverse Wishart prior distributions as in (Linderman et al. 2017); we sample from their complete conditionals.

One algorithmic choice remains: how to update across compartments? Note that all of the voltages and continuous latent states are jointly Gaussian given the discrete states and the augmentation variables. Moreover, within single compartments, the variables follow a Gaussian chain-structured model (i.e., a Kalman smoother model). We leverage this structure to develop a block-Gibbs sampling algorithm that jointly updates each compartment's voltage and continuous latent trajectories simultaneously, given the discrete states, augmentation variables, and the voltages of neighboring compartments[1].

[1]In principle, all compartments' continuous latent states and voltages could be updated at once, given that they are conditionally jointly Gaussian; however, the computational cost would naively scale as $O(T(ND)^3)$, based on a Kalman forward-backward sweep. An important direction for future work would

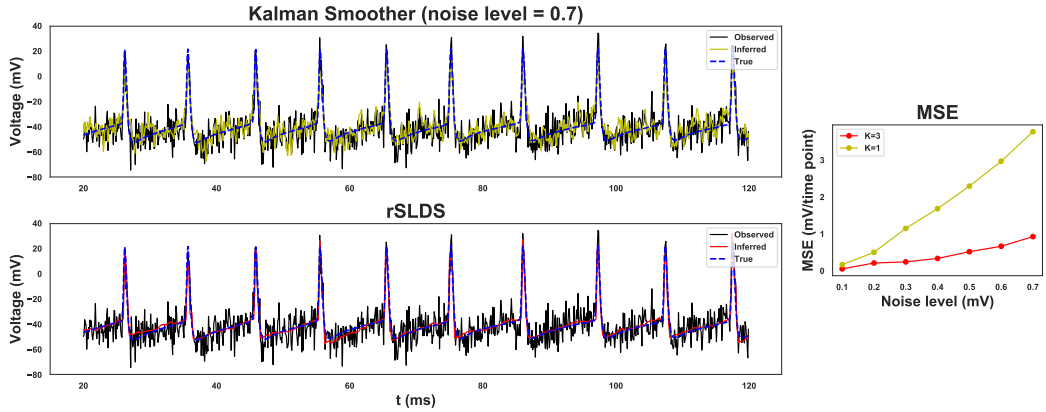

Figure 3.3: *Comparing the Kalman smoother against the rSLDS.* Left: example noisy data (constructed by adding noise to real voltage data) and output of the Kalman smoother (top) and rSLDS (bottom). Right: summary of MSE as a function of observation noise variance. The rSLDS outperforms the Kalman smoother, because the latter is a linear filter with a single homogeneous smoothness level, and therefore it must either underfit spikes or overfit subthreshold voltage (or both), whereas the rSLDS can enforce different levels of smoothness in different dynamical regimes (e.g., spiking versus non-spiking).

3.4 Results

A single-compartment rSLDS is a useful model of real somatic voltage

As a first test of the model and algorithm, we use intracellular voltage traces from the Allen Institute for Brain Science Cell Types Atlas (Brain Science 2015, cell id=464212183). The traces are recorded via patch clamp, a high SNR method that provides "ground truth" measurements. We added artificial white noise with a standard deviation of 5mV to these recordings in order to test the model's ability to both learn membrane potential dynamics and smooth noisy data. We fit the rSLDS to 100ms of data sampled at $\Delta t = 0.1$ms, for a total of 1000 time points. We used three discrete latent states ($K = 3$) and one additional latent dimension ($D = 1$); already, this simple model is sufficient to provide a surpris-

be to adapt the fast approximations proposed in (Paninski 2010) to implement this joint update more efficiently.

ingly good model of the observed data. Figure 3.2(a) shows the observed voltage (i.e. the ground truth plus white noise), and the 95% posterior credible intervals estimated with samples from the posterior under the estimated rSLDS model. The inferred discrete state sequence (c) shows how the rSLDS segments the voltage trajectory into periods of roughly linear accumulation (blue), spike rise (yellow) and spike fall (red). These periods are each approximated with linear dynamics in (V, x) space, as shown in Fig. 3.2(d). Moreover, simulating from the learned dynamics yields realistic trajectories in both latent space (e) and in voltage space, as shown in Fig 3.2(f).

Figure 3.3 compares the rSLDS to a simpler baseline method. Past work in voltage smoothing has utilized the Kalman smoother, based on an assumption of approximately linear dynamics in the subthreshold regime (Paninski 2010). Notably, the Kalman smoother is a special case of the rSLDS with $K = 1$ discrete state. We compare performance over a range of noise levels and find that the rSLDS significantly outperforms the standard Kalman smoother, due largely to the fact that the former can adapt to the drastically different smoothness of the voltage signal in the spiking versus the subthreshold regimes.

Spatiotemporal denoising with simulated data on real morphologies

The single compartment studies afford us ground truth electrical recordings, but obtaining simultaneous electrical recordings from a multiple compartments of a single cell is a significant technical challenge. Optical recordings of fluorescent voltage indicators offer multiple compartments, but at the cost of lower temporal resolution and significantly

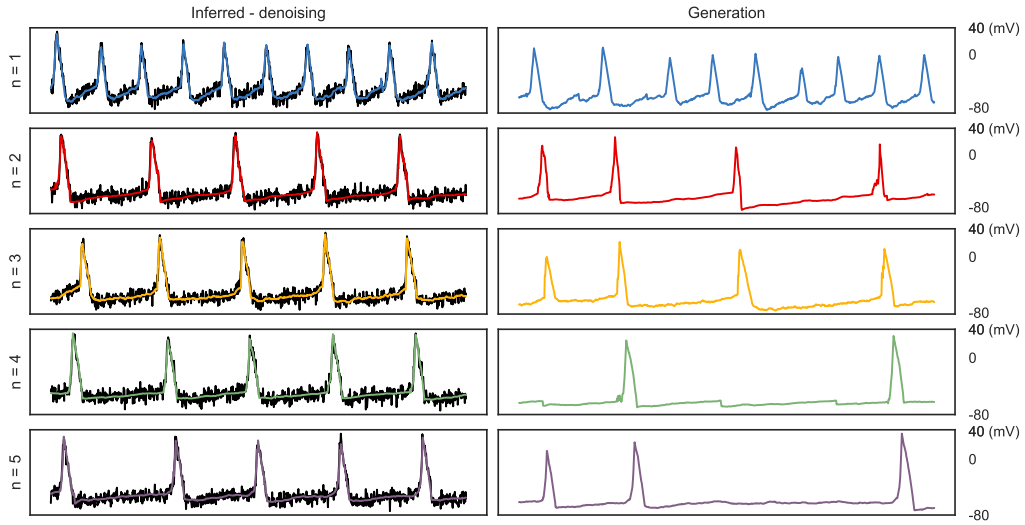

Figure 3.4: *Voltage smoothing in noisy recordings in a simulated dendritic branch.* The branch consists of five compartments connected in a chain. *Left:* The observed voltage (black line) and the inferred voltage (colored line) are shown for each compartment. *Right:* A sample from the learned multi-compartment rSLDS. The generated voltage traces show that the model has learned to reproduce the nonlinear dynamics of multi-compartment models, including the interactions between compartments that propagate spikes down the dendritic branch.

higher noise. To test our method's ability to denoise multi-compartment voltage traces, we simulate voltage traces using the simple HH model in lieu of ground truth electrical recordings[2]. We start by simulating a single dendritic branch and then move to a full dendritic tree, using real neuron morphologies.

Denoising a dendritic branch. We model a single dendritic branch as a five-compartment chain. We simulate membrane potential according to the classical HH model with voltage-gated sodium and potassium channels, a leak current, and currents from neighboring compartments in the chain. We inject a constant 35mA current into the

[2]Of course more detailed realistic simulations with multiple nonlinear channel types are possible; we leave this for future work. As we will see below, the simple HH model already provides a rich and interesting testbed simulated dataset for the methods presented here.

first compartment, and induced spikes propagate to downstream compartments according to the cable equation. However, we have tuned the inter-compartment conductances $g_{nn'}$ such that only every other spike propagates — a single spike in compartment 1 cannot depolarize compartment 2 enough to reach the spike threshold, but two spikes can. Moreover, we corrupt the voltage traces with additive Gaussian noise with standard deviation of 8mV. We test the multi-compartment rSLDS's ability to mimic the nonlinear dynamics of the true generative process and smooth the noisy voltage observations.

Figure 3.4 shows five compartments in the chain. In the left column we show the observed voltage traces (black) and the smoothed voltage (color) traces. Again, the model does an excellent job denoising the data, and is also able to learn a rich generative model of the spatiotemporal dynamics underlying the observed data. In particular, the model does not simply learn separate models for each compartment — it also learns interactions between compartments. This is evident in the traces that are generated by the model, as shown in the right column of Fig. 3.4. The first compartment shows a high firing rate, but only approximately every other spike propagates to downstream compartments, as in the real data. The generation is not perfect: some spikes fail to propagate (e.g. the third spike in compartment 3 does not propagate to compartment 4), and we see some spurious discrete state transitions (data not shown). Nevertheless, these generated samples indicate that the learned dynamical system captures the gross structure of multi-compartmental membrane dynamics. An accurate generative model offers a strong prior for smoothing spatiotemporal noisy voltage traces given by optical recordings.

Denoising a full dendritic tree. We have shown that the rSLDS can learn the dynamics of single compartments and dendritic branches (i.e. multi-compartment chains), but can these methods scale to full dendritic trees? We test these models on real three-dimensional dendritic tree (Brain Science 2015, cell id=464212183). (Note that this morphological data only includes the three-dimensional shape, not any voltage recordings.) Of the 2620 compartments in the original tree, we retain approximately every fifth compartment to create a tree with 519 compartments (Fig. 3.5f). We simulate the HH model on this tree to obtain 100ms of data at 10kHz temporal resolution. As above, we add Gaussian white noise (uncorrelated in space and time; standard deviation 25mV).

Fig. 3.5a shows the observed voltage across all spatial compartments for a single time point, and panel (b) shows the true underlying voltage. Supplementary Video 1 shows the voltage propagating in time through the tree. Panel (c) shows the voltage inferred by the multi-compartment rSLDS. The residual in panel (d) and the error in (e) show the difference between the inferred voltage and the observed and true voltage, respectively. There is a slight spatial correlation in the errors — specifically, the inferred voltage tends to slightly underestimate spike amplitude and overestimate the voltage during recovery — but the errors are generally small. Panel (g) shows the temporal estimates for the single compartment indicated by the green dot in (f). The posterior credible interval captures the true voltage, despite the high level of noise. Each spike corresponds to a canonical discrete state sequence, as in shown in (h). By transitioning between these discrete states, the piecewise linear dynamics aid the model-based decoding. Finally, panels (i-k) show both spatial and temporal dynamics of voltage for the dendritic branch highlighted in red in (f). We see the spikes propagating along the compartments in observed, true, and inferred

voltage. Again, the residuals (l) and errors (m) show small spatiotemporal correlations, but overall good recovery of the voltage from the noisy observations.

For comparison, previous particle filtering-based approaches to voltage smoothing (Huys and Paninski 2009) would need to infer the trajectories of ≈ 2000 state variables ($N = 500$ compartments with $D = 4$ dimensions each) in the simplest HH model of this data, and this model would not be able to adapt to modest changes in the dynamical parameters of the channels in each compartment. Scaling such methods to this number of state variables is a serious challenge, but the recurrent switching linear dynamical system approach makes this problem tractable.

3.5 Conclusion

The advent of new voltage imaging methods presents exciting opportunities to study computation in dendritic trees. We have developed new methods to smooth and denoise optical voltage traces in order to realize this potential. These methods incorporate biophysical knowledge in the form of constraints on the form of inter-compartmental dynamics, while allowing for effectively nonparametric learning of nonlinear intra-compartmental dynamics. We have illustrated the potential of these methods using semi-synthetic data based on real electrophysiological recordings of membrane potential and actual neuron morphologies. The results show promising progress toward the critical goal of denoising noisy spatiotemporal voltage measurements.

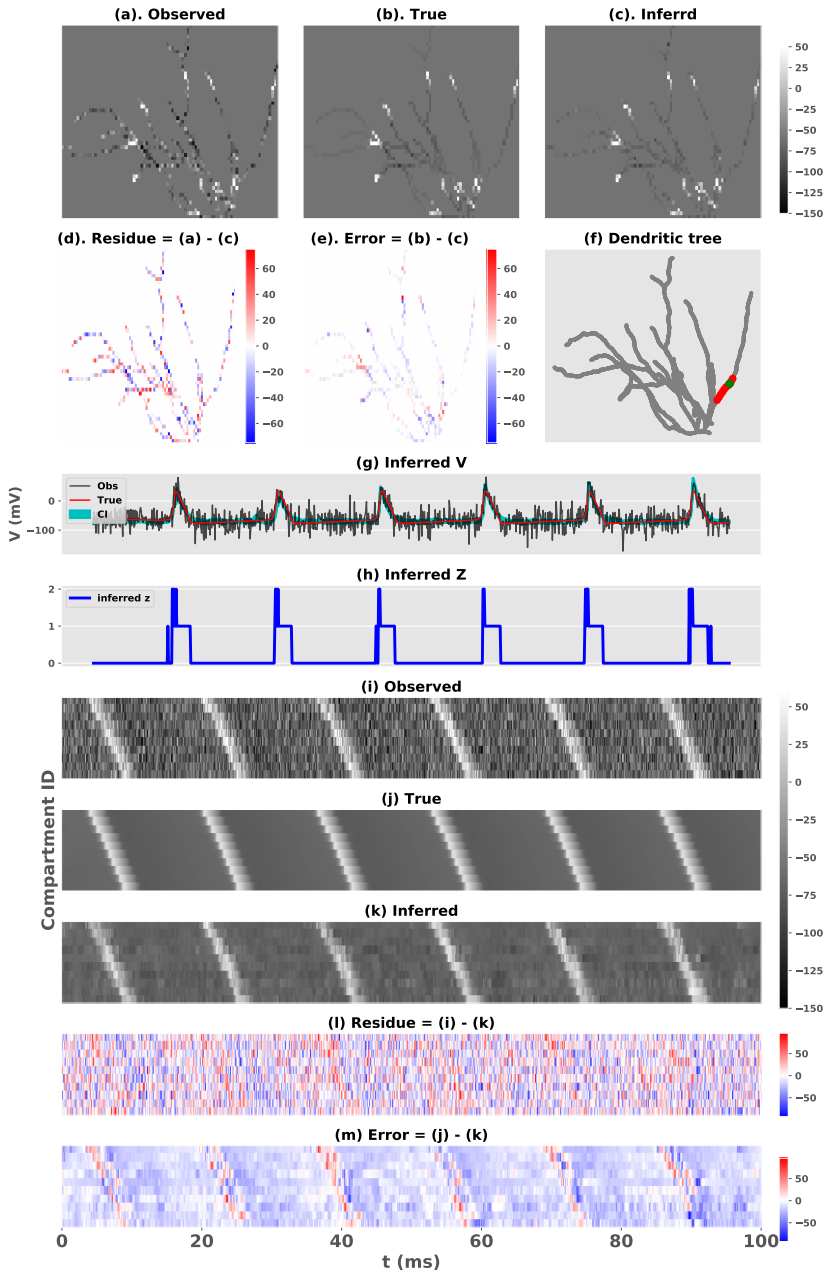

Figure 3.5: *Spatiotemporal denoising on a large simulated dendritic tree.* The 3D morphology of the tree is taken from a real cell in the Allen institute database. **(a)** Simulated voltage with additive Gaussian noise (25mV std.) on the dendritic tree. **(b)** True simulated voltage without noise. **(c)** Denoised estimate of voltage with the multi-compartment rSLDS; note the close match with (b). Colorbar shared between (a), (b), and (c). **(d)** Residual equals observed (a) minus inferred (c). **(e)** Error equals (b) minus (c). **(f)** Cartoon figure indicates the single compartment (green dot) and a segment of dendritic tree (red branch) shown in the following panels. **(g)** Noisy observation (black) and posterior credible interval (CI) of inferred voltage (cyan), and true voltage (red) corresponding to dot in (f). **(h)** The inferred discrete state of the compartment (0: subthreshold; 1: spike fall; 2: spike rise). **(i, j, k)** spatiotemporal representation of the noisy, true, and denoised voltage propagating up the branch shown in (f). **(l, m)** are the residual and error computed from (i, j, k).

Chapter 4

Computational Analysis for NeuroPAL

The development of experimental technology, NeuroPAL, is accomplished by our collaborator Eviatar Yemini and Oliver Hobert. The computational analysis is joint work with Erdem Varol, Gonzalo Mena, Amin Nejatbakhsh and Liam Paninski. This chapter of the thesis only includes the first three parts of the entire project to focus on my contribution and tell a relatively complete story. I led the first part – neuron filter (sec. 4.2). Amin and I worked closely together on neuron detection (sec 4.3). Erdem has led the development of (Sec 4.4).

This work is in preparation for journal submission. The full paper is included in Appendix.

4.1 Introduction

Identifying neuronal patterns of activity, gene expression, mutant effects remains a critical challenge when investigating neural circuitry. Despite rapid advances in whole brain imaging, current methods collapse sets of neurons activities into low dimension ensembles, prohibiting accurate analysis of individual neurons. In order to achieve single-neuron resolution functional analysis, our collaborators Eviatar Yemini and Oliver Hobert developed a multicolor C. elegans strain, called the NeuroPAL (a Neuronal Polychro-

matic Atlas of Landmarks). All NeuroPAL worms share an identical color map, permitting a complete, unambiguous determination of individual neuron names when using GCaMP/GFP/CFP/YFP reporters. The utility of NeuroPAL has been demonstrated in (1) functional connectomics; assisted by neural-activity sensor (GCaMP6s), NeuroPAL was able to delineate the neural circuitry activated by gustatory neurons or olfactory neurons in response to the corresponding stimulus respectively; (2) molecular connectomics; assisted by GFP-based reporters, NeuroPAL is able to identify expression for the whole family of metabotropic classical-neurotransmitter receptors.

The promising broad usage of NeuroPAL in neurobiology and the stereotyped designs for individual neurons (e.g. same color for same neuron) in NeuroPAL worms motivate and enable us to develop a novel set of algorithms, to automatically detect all neurons, match them to their unique named identities, and recover associated attributes such as location, neural activity or gene expression.

Overview of NeuroPAL

We describe a pipeline for semi-automated identification of neurons in *C. elegans* captured in three-dimensional color images of NeuroPAL strains. The proposed pipeline effectively comprises the three steps: (1) filter, (2) detect, and (3) identify, illustrated in the schematic in figure 4.1.

The first step in the pipeline consists of pre-filtering regions in the raw images that correspond to non-neuronal structures such as gut cells and lysosomes. This is followed by the neurons being detected from the filtered image using a greedy sparse reconstruc-

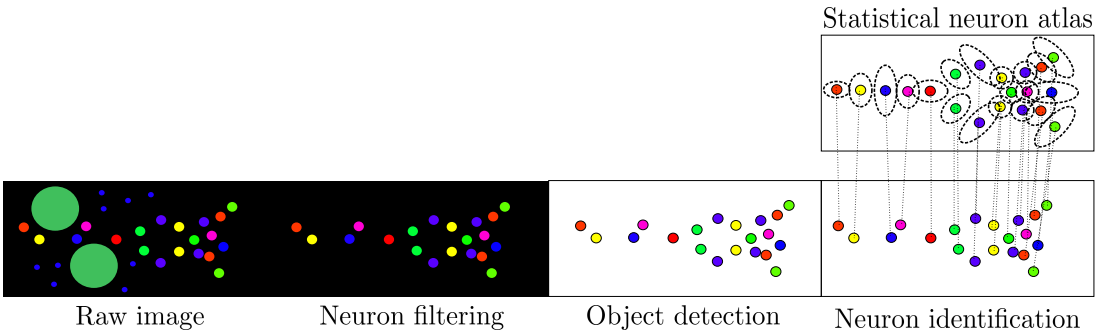

Figure 4.1: The schematic of the proposed *C. elegans* neuron detection and identification pipeline. From left to right: The raw image is supplied by the user and is then filtered to remove objects that are not neurons (i.e lysosomes, gut cells). From the filtered image, neurons are detected. The detected neurons are then aligned with a statistical atlas of neurons to yield their identities.

tion procedure consisting of a variant of matching pursuit (Bergeaud and Mallat 1995; Elad 2010; Mallat and Zhang 1993). Next, the color and positional features of the set of detected neurons are aligned with the features of a statistical atlas of neurons (Evangelidis and Horaud 2018) to yield a set of likelihoods for each neuron which are then used to compute probabilistic assignment of identities (Mena et al. 2018). Lastly, and optionally, the proposed model allows user supervision to refine the identifications.

The efficacy of the proposed pipeline is demonstrated in a leave-one-out cross-validation setting using 14 anterior and 39 posterior images of worms. The numerical results demonstrate the potential of our approach for practical deployment and public use.

This remainder of the manuscript is organized as follows. First, we introduce notation in Table 4.1. Section 4.2 describes the filters for removing non-neuronal objects from the image. In section 4.3, we describe the matching pursuit procedure for detecting neurons. In section 4.4, we describe the training model for obtaining statistical worm atlases.

The following sections are not included in this chapter, but are still stated here to keep

the project intact. In section 4.5, we show how to use the atlas to align and probabilistically identify neurons in out-of-sample worm images. In section 4.6, we demonstrate the empirical performance of our pipeline in cross-validation settings. Lastly, section 4.7 showcases some of the layouts and functions of the software package and graphical user interface.

Variable	Definition	Space	Section
I	Microscopy image in D voxels, C colors	$\mathbb{R}^{D\times C}$	4.2,4.3
m_k	center of kth detected neuron	$\mathbb{R}^3$	4.3,4.4
S_k	covariance of gaussian envelope of kth detected neuron	$\mathbb{S}^3_{++}$	4.3
π_k	color proportions of kth detected neuron	$\mathbb{R}^3$	4.3,4.4
μ_i	mean of position/color of neuron i	$\mathbb{R}^d$	4.4
$\boldsymbol{\Sigma}_i$	covariance of position/color of neuron i	$\mathbb{S}^d_{++}$	4.4
$\boldsymbol{\beta}_j$	Random transformation on worm j	$\mathbb{R}^{d\times d}$	4.4
$\boldsymbol{\beta}^0_j$	Random translation on worm j	$\mathbb{R}^d$	4.4
$\boldsymbol{P}_j$	Random permutation on worm j	$\mathcal{P}^{n\times n}$	4.4
$\boldsymbol{z}_{i,j}$	random draw of the stereotypical position/color of neuron i in worm j	$\mathbb{R}^d$	4.4
$\boldsymbol{x}_{i,j}$	randomly transformed neuron i for worm j	$\mathbb{R}^d$	4.4
$\boldsymbol{y}_{i,j}$	random permuted neuron i of worm j	$\mathbb{R}^d$	4.4
$\boldsymbol{L}$	Likelihood matrix of kth detection corresponding to the ith atlas neuron	$\mathbb{R}^{k\times n}$	4.4

Table 4.1: Notation

4.2 Step 1: Neuron filter

Due to the features of NeuroPAL staining, fluorescence images of the worms often contain bright non-neuronal components, such as gut cells or lysosomes. To reduce the number of false positives in detection, we remove these structures in a pre-filtering step (fig. 4.2).

Gut cells are usually larger-radius balls or hollow rings (larger than regular neuron nuclei), with a green color, whereas lysosomes are usually small balls (or an irregular shape) and blue in color. Therefore we filter these components by size and color: we

extract green and blue regions, and filter out those with size greater or less than the typical size of neuronal nuclei, respectively. For each color c, we create a binary mask to indicate the on-region for c. First, we lightly smooth each color channel individually (with a small 3d Gaussian filter). Then we generate a binary mask m_c to indicate regions with intensity for color c above the threshold (we use the 99th percentile as the threshold). Finally, we eliminate connected components in these masks that contain a total number of pixels above or below a user-defined threshold.

The overall algorithm for filtering gut cells and lysosomes is sketched in algorithm 3. The gut cell and lysosome removal subroutines are detailed below.

Algorithm 3 Neuron filter - Pseudocode

Input: Microscopy image: $\boldsymbol{I} \in \mathbb{R}^{D \times C}$

Apply a Gaussian smoother in each color channel

Mask pixels in each color channel whose intensities are in the 99th percentile

Compute connected components in masks

Eliminate connected components that are green and are larger than t' pixels (Gut elimination)

Eliminate connected components that are blue and are smaller than t'' pixels (Lysosome elimination)

return Filtered image: $\boldsymbol{I}_f \in \mathbb{R}^{D \times C}$

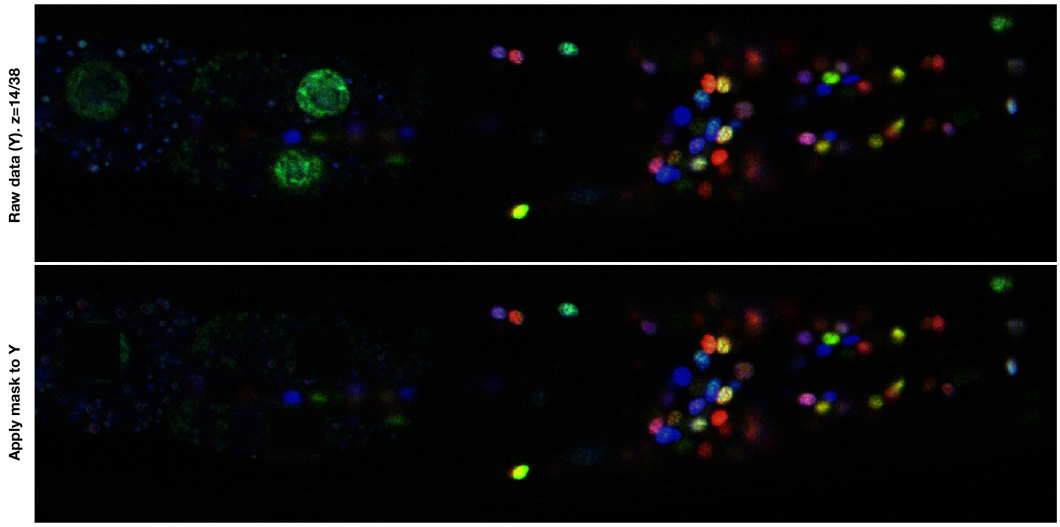

Figure 4.2: Filtered data (worm 38_YAaSP, ventral-dorsal view). Top: raw data (single z-slice). Bottom: filtered data, with with big green gut cells and small blue lysosomes largely removed. For all z slices, see Filter Video.

4.3 Step 2: Detecting neurons

Nuclear shape model

In NeuroPAL worms, neuronal nuclei are labeled with different colors. To detect these nuclei we greedily minimize the following objective:

$$\left\| \boldsymbol{I} - \sum_{k=1}^{K} f_{\theta_k} \right\|_2^2, \tag{4.1}$$

where $\boldsymbol{I} \in \mathbb{R}^{D \times C}$ represents the 3-dimensional color image of the worm (D voxels, C colors) and $f_{\theta_k}(\mathbf{v}) \in \mathbb{R}^{D \times C}$ is a function that approximates the multi-color shape of the neuron parametrized by θ_k; K is the number of cells visible in the field of view. We choose

f to be a Gaussian function:

$$f_{\theta_k}(\mathbf{v}, c) = b_k^c + \boldsymbol{\pi}_k^c (2\pi)^{-\frac{3}{2}} |\Sigma|^{-\frac{1}{2}} \exp\left[\frac{-(\mathbf{v}-_k)^T \boldsymbol{S}_k^{-1} (\mathbf{v}-_k)}{2}\right]; \tag{4.2}$$

here $_k$ is the spatial mean of the Gaussian bump, $\boldsymbol{S}_k$ controls the size and eccentricity of the Gaussian, $\boldsymbol{\pi}_k^c$ controls the brightness of the cell in the c-th color channel, and b_k^c is the brightness of the local background in color channel c, all for the cell indexed by k. We collect these parameters into

$$\theta_k = \{b_k, \pi_k,_k, \boldsymbol{S}_k\},$$

and impose constraints on each of these parameters:

$$\mathfrak{C} = \begin{cases} \boldsymbol{\pi}_{\text{min}} \leq \boldsymbol{\pi}_k \leq \boldsymbol{\pi}_{\text{max}} \\ b_{\text{min}} \leq b_k \leq b_{\text{max}} \\ {}_{\text{min}} \leq {}_k \leq {}_{\text{max}} \\ \sigma_{\text{min}} \leq \sigma(\boldsymbol{S}_k) \leq \sigma_{\text{max}}, \end{cases}$$

where the $\sigma(S)$ operator returns the singular values of the matrix S.

Optimization

To optimize this objective, we use a matching pursuit (MP) based strategy. In standard MP (?) we begin with a finite collection of filter functions f_k and then greedily add in the filter element k that leads to the largest reduction in the squared error (4.1). This greedy

optimization can be implemented efficiently using convolutions and simple subtractive updates of the objective function, until a maximal number K filters have been added to the model.

In the present context, the family of filters represented by the constraint set $\mathfrak{C}$ above is infinitely large, so we can not apply the standard MP approach directly. Instead, we choose an "average" Gaussian shape (i.e., with covariance $\boldsymbol{S}$ chosen near the "middle" of the constraint set), convolve the image $\boldsymbol{I}$ with this shape, and find local optima of the resulting function. (This is analogous to the first step of standard MP.) Then instead of subtracting this "average" shape away from around the peaks located in the first step, we locally optimize the parameter θ_k to fit the local shape of the image and then subtract this locally-optimized shape away. Then we iterate, adding a new locally-optimized shape to the model at each iteration k. (This algorithm can be parallelized by finding multiple local optima μ_k in the first step and then updating the corresponding parameters θ_k in parallel for locations μ_k that are sufficiently far apart.)

If we initialize $\rho_c = G^T \boldsymbol{I}_c$ (with $\boldsymbol{I}_c$ the 3d data volume in color channel c, and $G^T \boldsymbol{I}_c$ denoting convolution of $\boldsymbol{I}_c$ with the average Gaussian shape), then our approach can be summarized as in algorithm 4; see figure 4.3 for an illustration.

In the current optimization procedure we have two hyperparameters that should be specified before running the optimization. The first hyperparameter is the number of neurons (K). We found it more convenient to set K to a value larger than the estimated number of neurons in the image to prevent having false positives. This approach results in the detection of non-neuronal objects that can further be filtered out by the user. The second hyperparameter is the approximate size of a neuron in pixels which can be easily

read out from the image itself.

Algorithm 4 Neuron-like object detection

Input: Microscopy image: $\boldsymbol{I} \in \mathbb{R}^{D \times C}$

Initialize $\rho_c = G^T \boldsymbol{I}_c$ **for** $k = 1, \ldots, K$ **do**

end

Choose $\mu_k = \underset{x}{\operatorname{argmax}} \sum_c \rho_c(x)^2$ - regional maximum of smoothed image

Set $_{min}$ and $_{max}$ in the constraint set $\mathfrak{C}$ to enclose a small box near this initial μ_k, and then optimize

$$\theta_k = \underset{\theta_k \in \mathfrak{C}}{\operatorname{argmin}} \left\| \boldsymbol{I} - \sum_{j=1}^{k} f_{\theta_j}(X) \right\|^2$$

Compute residual image: $\rho_c \leftarrow \rho_c - G^T f_{\theta_k}$.

return $\theta_k = \{b_k, \boldsymbol{\pi}_{k,k}, \boldsymbol{S}_k\}$ which are the background mean, color proportions, cell centers and shape described by covariance.

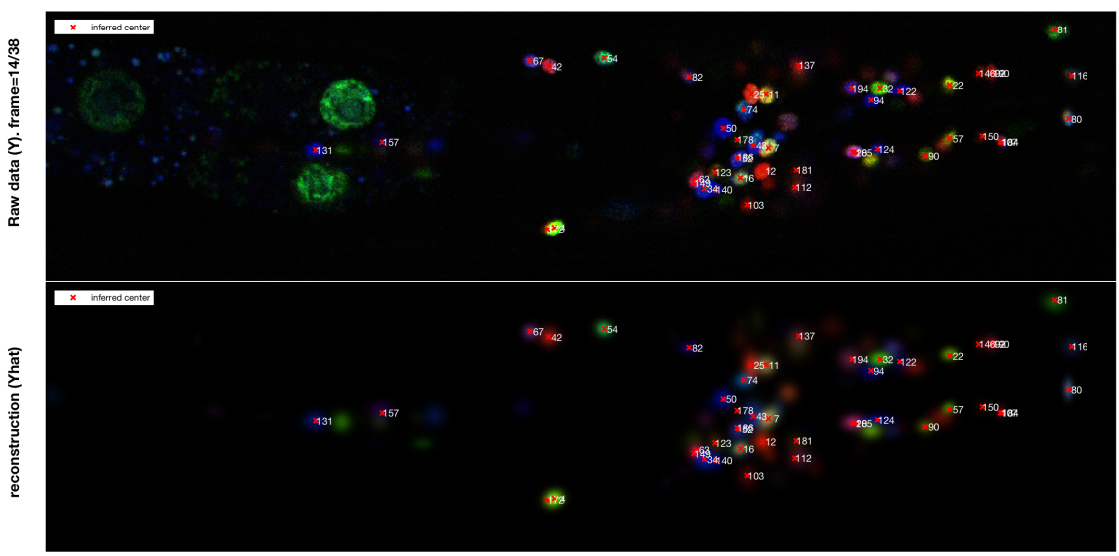

Figure 4.3: Detection step run on filtered data (same worm and z-slice as Fig. 4.2). Top: raw data $\boldsymbol{I}$. Bottom: reconstruction $\sum_k f_{\theta_k}$. Red crosses indicate inferred location of neurons; numbers indicate the order in which these locations were detected (typically brighter cells are detected first; note that some visible cells may be brighter on different z planes and may therefore not be labeled in this plane). For filtered data of all z slices, see full reconstruction video and residue video.

4.4 Step 3a: Statistical neuron atlas construction

Due to biological variability as well as variability in illumination and pose of the worm when imaged, the observed positions and colors of a given neuron will vary across imaged worms. This presents a significant challenge when attempting to obtain correspondences between worms to infer the identities of neurons. Therefore, to normalize the random variability that occurs across different worms, prior to identifying neurons in any given microscopy image, we estimate a statistical atlas of neuron positions and colors.

The approach we take resembles the joint expectation-maximization alignment of point sets technique of Evangelidis and Horaud 2018, with several important differences discussed below. The dataset we are modeling consists of a collection of point sets: each worm corresponds to one point set, with each point in the set corresponding to the position and color of a single detected neuron. We model each of these positions and colors as samples from a statistical atlas that is common across worms. Each neuron i has a corresponding mean and covariance in this atlas, denoted as $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$, respectively. After drawing all the positions and colors for a given worm j we apply a random affine transformation (parametrized by a matrix $\boldsymbol{\beta}_j$ and translation vector $\boldsymbol{\beta}_j^0$). Finally, since the order of neurons in each point set is arbitrary, we scramble the identities of the neurons with a random permutation, parameterized by a permutation matrix $\boldsymbol{P}_j$. This generative model is summarized in Figure 4.4.

ibid. infer the parameters of this generative model (i.e., the means and covariances of the statistical atlas, the random transformations, and the random permutations) in a completely unsupervised fashion using a three-way expectation maximization procedure.

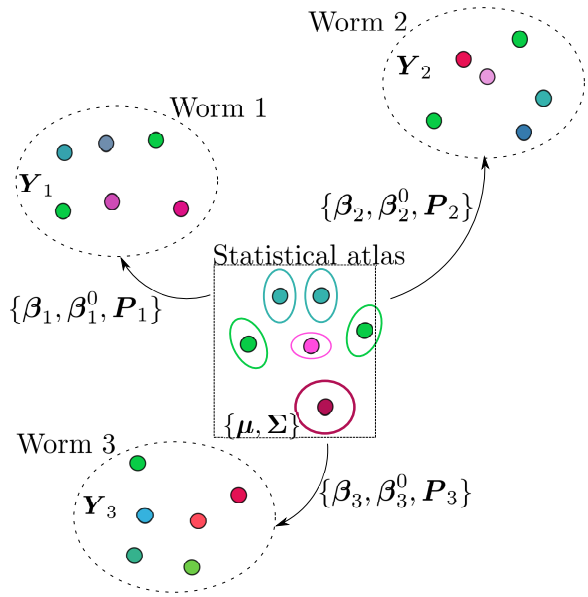

Figure 4.4: Schematic of the generative model of neuron position and color expression. First we draw a position and color for each neuron i from a distribution with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ (center box); then, to create the observed data $\boldsymbol{y}_{i,j}$ (the color and position of the i-th neuron of the j-th worm) we apply a random affine transformation $\boldsymbol{\beta}_j, \boldsymbol{\beta}_j^0$ to the positions and colors and a random permutation $\boldsymbol{P}_j$ to the identities (indicated with the arrows to the observed datasets Y_j for each worm j).

However, in our dataset, we have access to fully annotated neuron detections. We take advantage of this supervised data to simplify the inference problem.

Now we can describe our model in detail. Neural positions are three-dimensional, and there are three color channels in this dataset; therefore, if we use $\mathbf{y}_{i,j}$ to denote the appended position and color vector of the i-th neuron in worm j (as output by the detection step described in the previous section), then $\mathbf{y}_{i,j} \in \mathbb{R}^6$. Each of these observed $\mathbf{y}_{i,j}$ vectors has a corresponding latent vector $\boldsymbol{z}_{i,j}$ in the aligned atlas space. We model this latent vector as Gaussian,

$$\boldsymbol{z}_{i,j} \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{4.3}$$

with means $\boldsymbol{\mu}_i \in \mathbb{R}^6$ and covariances $\boldsymbol{\Sigma}_i \in \mathbb{S}_+^6$ that do not depend on the worm index j. We model the covariance $\boldsymbol{\Sigma}_i$ with block structure of the form $\boldsymbol{\Sigma}_i = \begin{bmatrix} \Sigma^i_{\text{position}} & \mathbf{0} \\ \mathbf{0} & \Sigma^i_{\text{color}} \end{bmatrix}$, since position and each color are independently varying.

Now the latent vectors $\boldsymbol{z}_{i,j}$ in the atlas space and observed data $\mathbf{y}_{i,j}$ extracted from the imaged worm j are connected by a worm-specific random affine transformation and permutation. We denote the intermediate affine-transformed variables as $\boldsymbol{x}_{i,j}$:

$$\boldsymbol{x}_{i,j} = \boldsymbol{z}_{i,j}\boldsymbol{\beta}_j + \boldsymbol{\beta}_j^0, \tag{4.4}$$

with $\boldsymbol{\beta}_j$ a 6×6 matrix (with a similar block structure as $\boldsymbol{\Sigma}_i$) and $\boldsymbol{\beta}_j^0 \in \mathbb{R}^6$. then we obtain $\mathbf{y}_{i,j}$ by scrambling the labels via the permutation p_j:

$$\boldsymbol{y}_{i,j} = \boldsymbol{x}_{p_j(i),j}. \tag{4.5}$$

Given these modelling assumptions, for a dataset of m worms and n_j detected neurons in each worm, we can express the likelihood of observation as:

$$P(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{P}, \boldsymbol{\beta}) = \tag{4.6}$$

$$\prod_{j=1}^{m}\prod_{i=1}^{n_j} \frac{1}{(2\pi)^{d/2}\det((\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T))^{1/2}} e^{-(1/2)(\boldsymbol{y}_{i,j}-\boldsymbol{\mu}_{\boldsymbol{P_{i,j}}}\boldsymbol{\beta}_j-\boldsymbol{\beta}_j^0)(\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T)^{-1}(\boldsymbol{y}_{i,j}-\boldsymbol{\mu}_{\boldsymbol{P_{i,j}}}\boldsymbol{\beta}_j-\boldsymbol{\beta}_j^0)^T} \tag{4.7}$$

which has the negative log likelihood:

$$
\begin{aligned}
&- L(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{P}, \boldsymbol{\beta}) = && (4.8)\\
&\sum_{j=1}^{m}\sum_{i=1}^{n_j}\frac{1}{2}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{\boldsymbol{p_{i,j}}}\boldsymbol{\beta}_j - \boldsymbol{\beta}_j^0)(\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T)^{-1}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{\boldsymbol{p_{i,j}}}\boldsymbol{\beta}_j - \boldsymbol{\beta}_j^0)^T) + \\
&\frac{1}{2}\log\det((\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T)) + C && (4.9)
\end{aligned}
$$

Since the term $\sum_j \sum_i (1/2)\log\det((\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T)$ is permutation invariant, we can write it as $\sum_j \sum_i (1/2)\log\det((\boldsymbol{\beta}_j\boldsymbol{\Sigma}_i\boldsymbol{\beta}_j^T)$.

Therefore, the maximum likelihood estimation (MLE) of parameters for our generative model involves optimizing the following objective:

$$
\begin{aligned}
&\underset{\boldsymbol{P},\boldsymbol{\beta},\boldsymbol{\mu},\boldsymbol{\Sigma}}{\text{minimize}} && (4.10)\\
&\sum_{j=1}^{m}\sum_{i=1}^{n_j}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{\boldsymbol{p_{i,j}}}\boldsymbol{\beta}_j - \boldsymbol{\beta}_j^0)(\boldsymbol{\beta}_j\boldsymbol{\Sigma}_{p_{i,j}}\boldsymbol{\beta}_j^T)^{-1}(\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{\boldsymbol{p_{i,j}}}\boldsymbol{\beta}_j - \boldsymbol{\beta}_j^0)^T) && (4.11)\\
&+\sum_{j=1}^{m}\sum_{i=1}^{n_j}\log\det((\boldsymbol{\beta}_j\boldsymbol{\Sigma}_i\boldsymbol{\beta}_j^T) && (4.12)
\end{aligned}
$$

Optimization

To infer the parameters of the generative model, we take an iterative approach since MLE is a tri-convex system with three blocks of variables: $\{\boldsymbol{P}\}, \{\boldsymbol{\beta}\}, \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$. This can be solved to a local optima by fixing $\boldsymbol{P}, \boldsymbol{\beta}$ and solving for $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ and then fixing $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ and iteratively solving for $\boldsymbol{P}, \boldsymbol{\beta}$ as is done in (Evangelidis and Horaud 2018).

Inference of the statistical atlas parameters $\mu, \boldsymbol{\Sigma}$

Let $\boldsymbol{P}_j \in \mathcal{P}^{n\times n}$ denote the permutation matrix, $\boldsymbol{Y}_j = [\mathbf{y}_{1,j}^T \dots \mathbf{y}_{n,j}^T]^T \in \mathbb{R}^{n\times d}$ denote the row stacked features of the neurons of the jth worm, and let $\boldsymbol{M} = [\boldsymbol{\mu}_1^T \dots \boldsymbol{\mu}_n^T] \in \mathbb{R}^{n\times d}$ denote the row stacked neuron means.

The generative model can be written in matrix form as:

$$\boldsymbol{Y}_j = \boldsymbol{P}_j \boldsymbol{M} \boldsymbol{\beta}_j + \mathbf{1}\boldsymbol{\beta}_j^0 + \boldsymbol{E} \tag{4.13}$$

where $\boldsymbol{E}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\beta}_j \boldsymbol{\Sigma}_{\boldsymbol{P}_{i,j}} \boldsymbol{\beta}_j^T)$ denotes the row stacked uncertainty terms. Since $\boldsymbol{P}_j^T \boldsymbol{P}_j = \mathbf{I}$ because $\boldsymbol{P}$ is a permutation matrix and assuming that $\boldsymbol{\beta}_j$ is a non-degenerate transformation, its inverse exists and can be used to write the system as:

$$\boldsymbol{P}_j^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \mathbf{1}\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} = \boldsymbol{M} + \boldsymbol{V} \tag{4.14}$$

where $\boldsymbol{V}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_i)$ is a term to quantify uncertainty.

This equation can be used to infer $\boldsymbol{M}$ and $\boldsymbol{\Sigma}$ by computing the first and second moments of $\boldsymbol{V}$:

$$\boldsymbol{M}^* = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{P}_j^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \mathbf{1}\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} \tag{4.15}$$

$$\boldsymbol{\Sigma}_i^* = \frac{1}{m} \sum_{j=1}^{m} (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i)^T (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i) \tag{4.16}$$

Inference of the transformation terms $\boldsymbol{\beta}, \boldsymbol{\beta}_0$

We can infer the transformation and translation terms $\boldsymbol{\beta}, \boldsymbol{\beta}_0$ if we view equation 4.14 as a weighted linear regression with a different goodness of fit criteria for each neuron quantified by a mahalanobis norm using the statistical atlas covariance $\boldsymbol{\Sigma}_i$.

The weighted regression can be posed as the following unconstrained minimization problem:

$$\underset{\boldsymbol{\beta}_j^{-1}, \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}}{\text{minimize}} \sum_{i=1}^{n_j} (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i)^T$$

which has the derivatives wrt to $\boldsymbol{\beta}_j^{-1}, \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}$ as

$$\frac{\partial}{\partial \boldsymbol{\beta}_j^{-1}} = \sum_{i=1}^{n_j} (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1} = 0$$
$$\frac{\partial}{\partial \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}} = \sum_{i=1}^{n_j} (\boldsymbol{P}_{j,i}^T \boldsymbol{X}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1} = 0$$

First we solve for $\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}$:

$$\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1*} = \left(\sum_{i=1}^{n} (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j \boldsymbol{\beta}_j^{-1} - \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1} \right) \left(\sum_{i=1}^{n} \boldsymbol{\Sigma}_i^{-1} \right)^{-1} \tag{4.17}$$

To analytically solve for $\boldsymbol{\beta}_j^{-1}$, we utilize the fact that $\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$ where $\text{vec}(\cdot)$ denotes vectorization operation and $\otimes$ denotes Kronecker product. This yields the

following relation:

$$\sum_{i=1}^{n}(\boldsymbol{S}_i^{-1} \otimes (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j))\text{vec}(\boldsymbol{\beta}_j^{-1}) = \sum_{i=1}^{n} \text{vec}((\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} + \boldsymbol{\mu}_i)\boldsymbol{\Sigma}_i^{-1})$$

which gives a vectorized solution for $\boldsymbol{\beta}_j^{-1}$:

$$\text{vec}(\boldsymbol{\beta}_j^{-1*}) = \left(\sum_{i=1}^{n}(\boldsymbol{\Sigma}_i^{-1} \otimes (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)) \right)^{-1} \left(\sum_{i=1}^{n} \text{vec}((\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1} + \boldsymbol{\mu}_i)\boldsymbol{\Sigma}_i^{-1}) \right) \tag{4.18}$$

Note that the expressions for the zero gradient solutions of $\boldsymbol{\beta}_j^{-1}$ involves $\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}$ and likewise, the zero gradient solution of $\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}$ involves $\boldsymbol{\beta}_j^{-1}$. This prohibits yielding a closed form solution for each term. Instead, we employ an iterative coordinate descent method to yield an optimum solution for $\boldsymbol{\beta}_j^{-1}$ and $\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}$. These optima can then be used to infer $\boldsymbol{\beta}_j^*$ and $\boldsymbol{\beta}_j^{0*}$. The affine transformation optimization procedure is outlined in algorithm 5.

Permutation inference

Lastly, we can solve for the permutation $\boldsymbol{P}_j$ by setting up a $n \times n$ transport matrix $\boldsymbol{D}$ where

$$\boldsymbol{D}_{u,v} = (\boldsymbol{\mu}_u \boldsymbol{Y} \boldsymbol{\beta}_j + \boldsymbol{\beta}_j^0 - \boldsymbol{Y}_{j,v})\boldsymbol{\Sigma}_u^{-1}(\boldsymbol{\mu}_u \boldsymbol{Y} \boldsymbol{\beta}_j + \boldsymbol{\beta}_j^0 - \boldsymbol{Y}_{j,v})^T \tag{4.19}$$

and obtaining $\boldsymbol{P}_j$ through the linear assignment optimization solved through the Hungarian algorithm of Kuhn 1955 which minimizes the following objective:

Algorithm 5 Affine transformation optimization

Input: Statistical atlas parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}$, neuron correspondences for jth worm $\boldsymbol{P}_j$, ϵ convergence tolerance

Initalize: $[\boldsymbol{\beta}_j^{-1}]^0$ and $[\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}]^0$ randomly **while** *Not converged* **do**

end

$[\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}]^t \leftarrow \left(\sum_{i=1}^n (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j [\boldsymbol{\beta}_j^{-1}]^{t-1} - \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1} \right) \left(\sum_{i=1}^n \boldsymbol{\Sigma}_i^{-1} \right)^{-1}$

$[\boldsymbol{\beta}_j^{-1*}]^t \leftarrow \text{reshape}\left(\left(\sum_{i=1}^n (\boldsymbol{\Sigma}_i^{-1} \otimes (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T (\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)) \right)^{-1} \left(\sum_{i=1}^n \text{vec}((\boldsymbol{P}_{j,i}^T \boldsymbol{Y}_j)^T ([\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}]^t + \boldsymbol{\mu}_i) \boldsymbol{\Sigma}_i^{-1}) \right) \right)$

Check convergence $\|[\boldsymbol{\beta}_j^{-1}]^t - [\boldsymbol{\beta}_j^{-1}]^{t-1}\|_F \leq \epsilon$

return $\boldsymbol{\beta}_j^* = ([\boldsymbol{\beta}_j^{-1}]^t)^{-1}$, $\boldsymbol{\beta}_j^{0*} = [\boldsymbol{\beta}_j^0 \boldsymbol{\beta}_j^{-1}]^t \boldsymbol{\beta}_j^*$

$$\boldsymbol{P}_j^* = \arg\min_{p \in \mathcal{P}} \sum_{u,v} p_{u,v} \boldsymbol{D}_{u,v} \tag{4.20}$$

Statistical atlas of neuron positions and colors

If we have access to several annotated worms, meaning that we have access to both the detections and their correspondences to the identities of neurons, $\boldsymbol{P}$, we can infer the transformation terms $\{\boldsymbol{\beta}, \boldsymbol{\beta}_0\}$ as well as the parameters of the statistical atlas $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ using the procedure outlined in algorithm 6.

In words, algorithm 6 operates in the following way. First, the targeted inference parameters are initialized using the neuron centers and colors for a random worm. Then, the remaining worms are affinely aligned to the hypothetical atlas by solving the linear system for $\{\boldsymbol{\beta}_k, \boldsymbol{\beta}_j^0\}$ in equation 4.18. The means and covariances of the aligned neurons are

then used to update the atlas parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. This procedure is iteratively repeated until convergence. The resulting statistical atlas consisting of the mean stereotypical neuron positions and colors and their covariances can be seen in figures 4.5, 4.6.

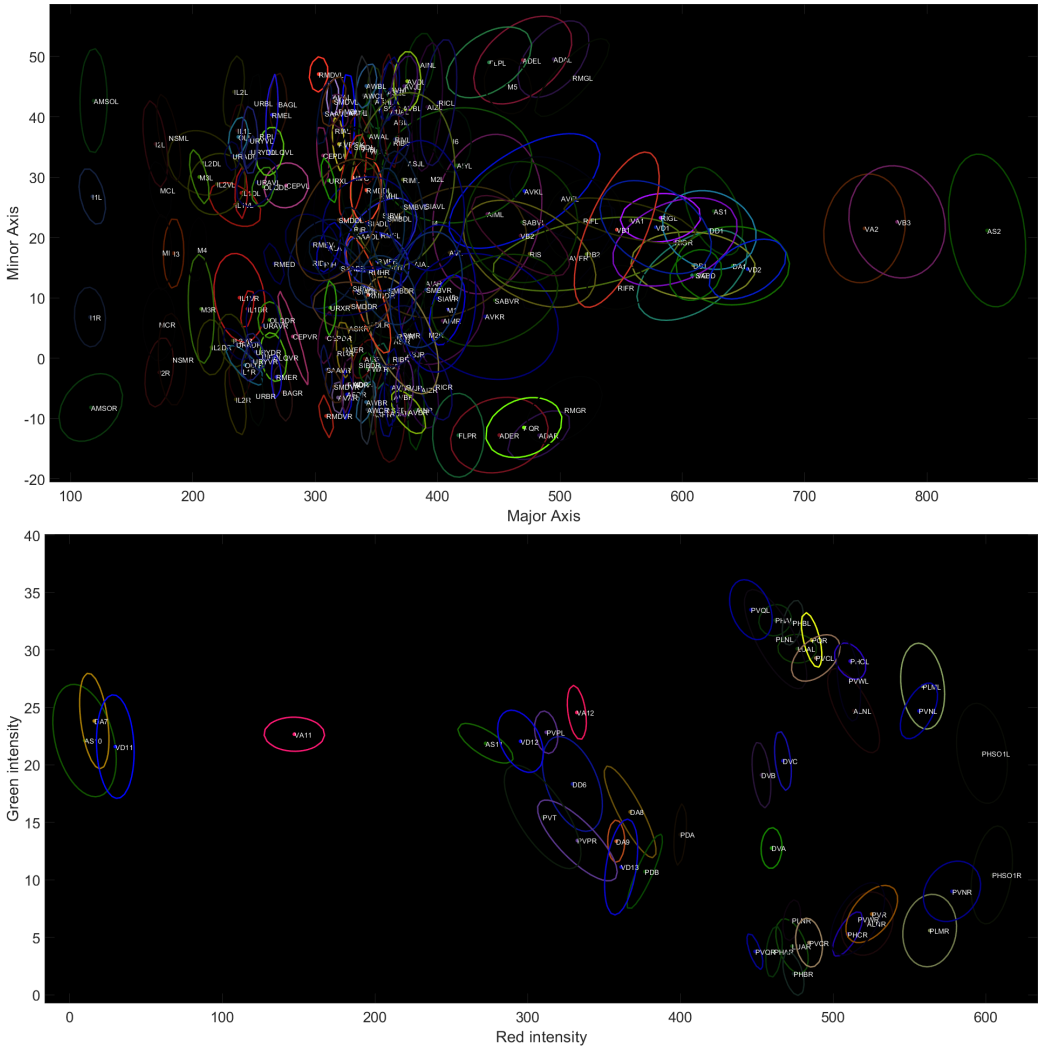

Figure 4.5: The statistical atlas of positions of *C. elegans* neurons of the head (top), and tail (bottom). The centers denote the mean positions while the ellipses denote the contours that delineate the half quantile. Note that this is 2D projection of a 3D atlas, therefore the x and y axes denote the major and minor axes of position in pixels.

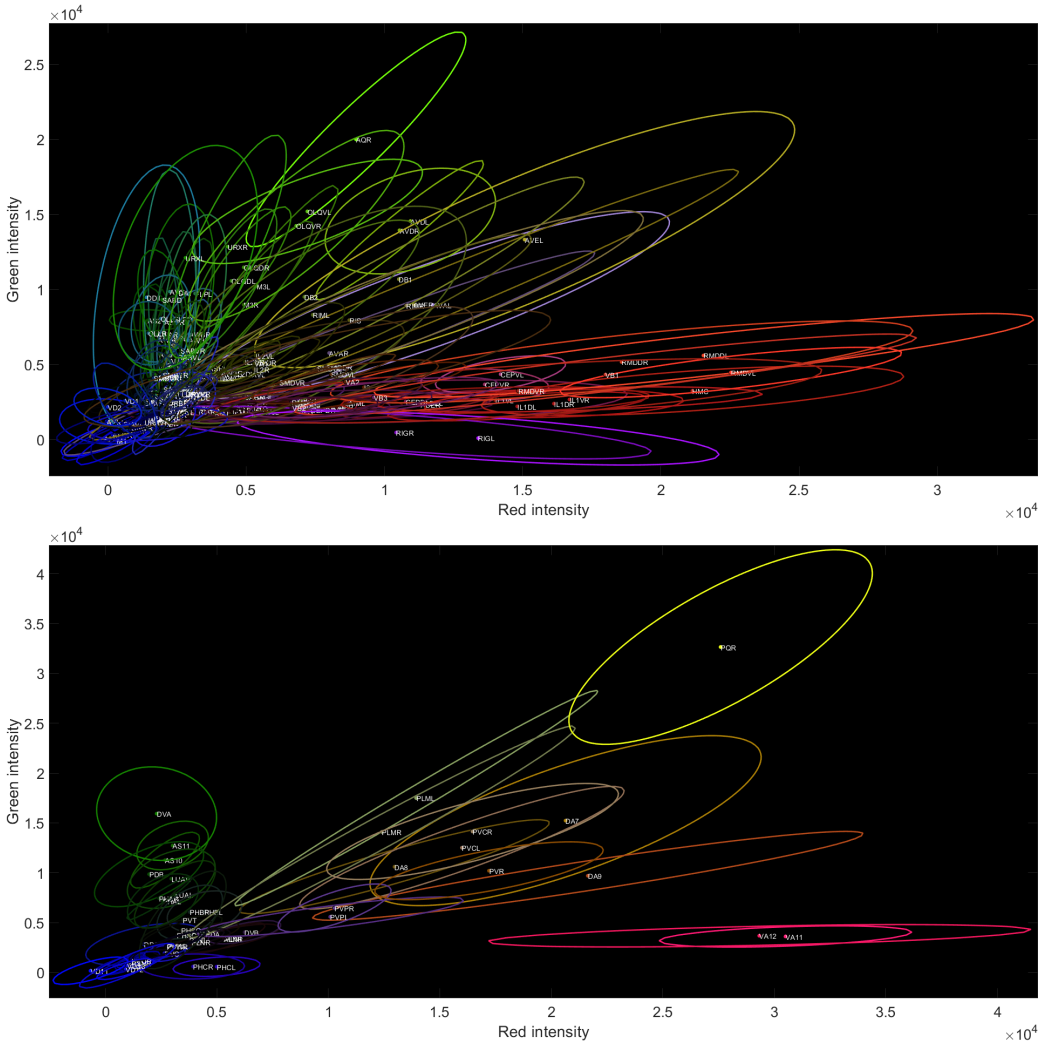

Figure 4.6: The statistical atlas of positions of *C. elegans* neurons of the head (top), and tail (bottom). The centers denote the mean positions while the ellipses denote the contours that delineate the half quantile. Note that this is 2D projection of a 3D atlas, therefore the x and y axes denote the major and minor axes of color intensities in arbitrary units.

Algorithm 6 Train statistical neuron atlas

Input: $\{\mathbf{y}_{i,j}\}$ (colors and positions) and $\{\boldsymbol{P}_{i,j}\}$ (neuron correspondences) for $j = 1, \ldots, m$ worms in a training set and $i = 1, \ldots, n$ neurons, ϵ (convergence tolerance)

Initalization

Select random worm $j \sim \text{Unif}[m]$

Set means as neuron centers of worm j: $\boldsymbol{\mu}_i^0 \leftarrow \mathbf{y}_{i,j}$ for $i = 1, \ldots, n$

Set covariances as identity: $\boldsymbol{\Sigma}_i^0 \leftarrow \boldsymbol{I}_6$ for $i = 1, \ldots, n$ **while** *Not converged* **do**

end

$t \leftarrow t + 1$ **for** *j=1,...,n* **do**

end

Solve alignment of jth worm to atlas $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ using equations 4.17 and 4.18

Update $\boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t$ using equations 4.15

Check convergence $\|\boldsymbol{\mu}^t - \boldsymbol{\mu}^{t-1}\|_F \leq \epsilon$ and $\|\boldsymbol{\Sigma}^t - \boldsymbol{\Sigma}^{t-1}\|_F \leq \epsilon$

return Statistical atlas of neuron colors and positions $\{\boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t\}$.

Conclusion

To conclude, we proposed and demonstrated the usage of Scalable Baysian models for four interesting biological or neural applications.

1. **Super Resolution Microscope**

   increased performances of more than 20% over state-of-art approaches

2. **Particle Tracking**

   proposed probabilistic-linking approach for tracking

3. **Voltage imaging denoising**

   modeled neuron voltage dynamics with non-linear dynamics

4. **NeuroPAL ID**

   proposed end-to-end approach to detect neurons and give their identity, and provide public-use software.

Leveraging on modern machine learning and deep learning techniques, robust statistical approaches and Bayesian models can enhance the performance of novel technology even further.

Bibliography

Abdelfattah, Ahmed S et al. (2018). "Bright and photostable chemigenetic indicators for extended in vivo voltage imaging." In: *bioRxiv*, p. 436840.

Babcock, Hazen, Yaron M Sigal, and Xiaowei Zhuang (2012). "A high-density 3D localization algorithm for stochastic optical reconstruction microscopy." In: *Optical Nanoscopy* 1.1, p. 1.

Bach, Francis et al. (2011). "Convex optimization with sparsity-inducing norms." In: *Optimization for Machine Learning* 5.

Bates, Mark et al. (2012). "Multicolor Super-Resolution Fluorescence Imaging via Multi-Parameter Fluorophore Detection." In: *ChemPhysChem* 13.1, pp. 99–107.

Bergeaud, Francois and Stephane Mallat (1995). "Matching pursuit of images." In: *Proceedings., International Conference on Image Processing*. Vol. 1. IEEE, pp. 53–56.

Betzig, Eric et al. (2006). "Imaging intracellular fluorescent proteins at nanometer resolution." In: *Science* 313.5793, pp. 1642–1645.

Blei, David M, Alp Kucukelbir, and Jon D McAuliffe (2016). "Variational inference: A review for statisticians." In: *arXiv preprint arXiv:1601.00670*.

Brain Science, Allen Institute for (2015). *Allen Cell Types Database*. URL: `https://celltypes.brain-map.org/`.

Cajal, S Ramon (1911). "Histologie du système nerveux de l'Homme et des vertébrés." In: *Maloine (Paris)* 2, pp. 891–942.

Chavarha, Mariya et al. (2018). "Fast two-photon volumetric imaging of an improved voltage indicator reveals electrical activity in deeply located neurons in the awake brain." In: *bioRxiv*, p. 445064.

Chenouard, Nicolas et al. (2014). "Objective comparison of particle tracking methods." In: *Nature methods* 11.3, p. 281.

Cox, Susan et al. (2012). "Bayesian localization microscopy reveals nanoscale podosome dynamics." In: *Nature methods* 9.2, pp. 195–200.

Elad, Michael (2010). "Sparse and redundant representations: from theory to applications in signal and image processing." In: *Springer New York*.

Evangelidis, Georgios Dimitrios and Radu Horaud (2018). "Joint alignment of multiple point sets with batch and incremental expectation-maximization." In: *IEEE transactions on pattern analysis and machine intelligence* 40.6, pp. 1397–1410.

Fraccaro, Marco et al. (2017). "A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning." In: *Advances in Neural Information Processing Systems*, pp. 3604–3613.

Gao, Yuanjun et al. (2016). "Linear dynamical neural population models through nonlinear embeddings." In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al., pp. 163–171.

Ghahramani, Zoubin and Michael I Jordan (1996). "Factorial hidden Markov models." In: *Advances in Neural Information Processing Systems*, pp. 472–478.

Hess, Samuel T, Thanu PK Girirajan, and Michael D Mason (2006). "Ultra-high resolution imaging by fluorescence photoactivation localization microscopy." In: *Biophysical journal* 91.11, pp. 4258–4272.

Hochbaum, Daniel R et al. (2014). "All-optical electrophysiology in mammalian neurons using engineered microbial rhodopsins." In: *Nature methods* 11.8, p. 825.

Hodgkin, Allan L and Andrew F Huxley (1952). "Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo." In: *The Journal of physiology* 116.4, pp. 449–472.

Huang, Bo et al. (2008). "Three-dimensional super-resolution imaging by stochastic optical reconstruction microscopy." In: *Science* 319.5864, pp. 810–813.

Hugelier, Siewert et al. (2016). "Sparse deconvolution of high-density super-resolution images." In: *Scientific reports* 6.

Huys, Quentin JM and Liam Paninski (2009). "Smoothing of, and parameter estimation from, noisy biophysical recordings." In: *PLoS computational biology* 5.5, e1000379.

Jaqaman, Khuloud et al. (2008). "Robust single-particle tracking in live-cell time-lapse sequences." In: *Nature methods* 5.8, p. 695.

Johnson, Matthew et al. (2016). "Composing graphical models with neural networks for structured representations and fast inference." In: *Advances in neural information processing systems*, pp. 2946–2954.

Kannan, Madhuvanthi et al. (2018). "Fast, in vivo voltage imaging using a red fluorescent indicator." In: *Nature methods* 15.12, p. 1108.

Kingma, Diederik P and Max Welling (2013). "Auto-encoding variational bayes." In: *arXiv preprint arXiv:1312.6114*.

Koch, Christof (2004). *Biophysics of computation: information processing in single neurons*. Oxford university press.

Krishnan, Rahul G, Uri Shalit, and David Sontag (2017). "Structured Inference Networks for Nonlinear State Space Models." In: *AAAI*, pp. 2101–2109.

Kuhn, Harold W (1955). "The Hungarian method for the assignment problem." In: *Naval research logistics quarterly* 2.1-2, pp. 83–97.

Le, Tuan Anh et al. (2018). "Auto-encoding sequential Monte Carlo." In: *International Conference on Learning Representations*.

Linderman, Scott W. et al. (2017). "Bayesian learning and inference in recurrent switching linear dynamical systems." In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Lueckmann, Jan-Matthis et al. (2017). "Flexible statistical inference for mechanistic models of neural dynamics." In: *Advances in Neural Information Processing Systems*, pp. 1289–1299.

Maddison, Chris J et al. (2017). "Filtering variational objectives." In: *Advances in Neural Information Processing Systems*, pp. 6573–6583.

Mallat, Stéphane G and Zhifeng Zhang (1993). "Matching pursuits with time-frequency dictionaries." In: *IEEE Transactions on signal processing* 41.12, pp. 3397–3415.

Manzo, Carlo and Maria F Garcia-Parajo (2015). "A review of progress in single particle tracking: from methods to biophysical insights." In: *Reports on progress in physics* 78.12, p. 124601.

Marshall, Jesse D et al. (2016). "Cell-type-specific optical recording of membrane voltage dynamics in freely moving mice." In: *Cell* 167.6, pp. 1650–1662.

Mena, Gonzalo et al. (2018). "Learning Latent Permutations with Gumbel-Sinkhorn Networks." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Byt3oJ-0W.

Meng, Liang, Mark A Kramer, and Uri T Eden (2011). "A sequential Monte Carlo approach to estimate biophysical neural models from spikes." In: *Journal of neural engineering* 8.6, p. 065006.

Min, Junhong et al. (2014). "FALCON: fast and unbiased reconstruction of high-density super-resolution microscopy data." In: *Scientific reports* 4.

Miyazawa, Hiroaki et al. (2018). "Optical interrogation of neuronal circuitry in zebrafish using genetically encoded voltage indicators." In: *Scientific reports* 8.1, p. 6048.

Mukamel, Eran A, Hazen Babcock, and Xiaowei Zhuang (2012). "Statistical deconvolution for superresolution fluorescence microscopy." In: *Biophysical journal* 102.10, pp. 2391–2400.

Mukamel, Eran A and Mark J Schnitzer (2012). "Unified resolution bounds for conventional and stochastic localization fluorescence microscopy." In: *Physical review letters* 109.16, p. 168102.

Naesseth, Christian et al. (2018). "Variational Sequential Monte Carlo." In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, pp. 968–977.

Nassar, Josue et al. (2019). "Tree-Structured Recurrent Switching Linear Dynamical Systems for Multi-Scale Modeling." In: *International Conference on Learning Representations*.

Newby, Jay M et al. (2017). "Deep neural networks automate detection for tracking of submicron scale particles in 2D and 3D." In: *arXiv preprint arXiv:1704.03009*.

O'Leary, Timothy et al. (2013). "Correlations in ion channel expression emerge from homeostatic tuning rules." In: *Proceedings of the National Academy of Sciences* 110.28, E2645–E2654.

Paige, Brooks and Frank Wood (2016). "Inference networks for sequential Monte Carlo in graphical models." In: *International Conference on Machine Learning*, pp. 3040–3049.

Paninski, Liam (2010). "Fast Kalman filtering on quasilinear dendritic trees." In: *Journal of computational neuroscience* 28.2, pp. 211–228.

Parthasarathy, Nikhil et al. (2017). "Neural Networks for Efficient Bayesian Decoding of Natural Images from Retinal Neurons." In: *Advances in Neural Information Processing Systems*, pp. 6437–6448.

Piatkevich, Kiryl D et al. (2018). "A robotic multidimensional directed evolution approach applied to fluorescent voltage reporters." In: *Nature chemical biology* 14.4, p. 352.

Picardo, Michel A et al. (2016). "Population-Level Representation of a Temporal Sequence Underlying Song Production in the Zebra Finch." In: *Neuron* 90.4, pp. 866–876.

Polson, Nicholas G, James G Scott, and Jesse Windle (2013). "Bayesian inference for logistic models using Pólya-gamma latent variables." In: *Journal of the American Statistical Association* 108.504, pp. 1339–1349.

Rabiner, Lawrence R. (1990). "Readings in Speech Recognition." In: ed. by Alex Waibel and Kai-Fu Lee. Morgan Kaufmann Publishers Inc. Chap. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267–296.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic back-propagation and approximate inference in deep generative models." In: *arXiv preprint arXiv:1401.4082*.

Rust, Michael J, Mark Bates, and Xiaowei Zhuang (2006). "Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM)." In: *Nature methods* 3.10, pp. 793–796.

Sage, Daniel et al. (2015). "Quantitative evaluation of software packages for single-molecule localization microscopy." In: *Nature methods*.

Shen, Zhaolong and Sean B Andersson (2009). "Tracking multiple fluorescent particles in two dimensions in a confocal microscope." In: *CDC*. Citeseer, pp. 6052–6057.

Smal, Ihor et al. (2008). "Particle filtering for multiple object tracking in dynamic fluorescence microscopy images: Application to microtubule growth analysis." In: *IEEE Transactions on Medical Imaging* 27.6, pp. 789–804.

Small, Alex and Shane Stahlheber (2014). "Fluorophore localization algorithms for super-resolution microscopy." In: *Nature methods* 11.3, pp. 267–279.

Snell, Jake and Richard S. Zemel (2017). "Stochastic Segmentation Trees for Multiple Ground Truths." In: *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*.

Sontag, Eduardo (1981). "Nonlinear regulation: The piecewise linear approach." In: *IEEE Transactions on Automatic Control* 26.2, pp. 346–358.

Stuart, Greg, Nelson Spruston, and Michael Häusser (2016). *Dendrites*. Oxford University Press.

Sun, Ruoxi, Evan Archer, and Liam Paninski (2017). "Scalable variational inference for super resolution microscopy." In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, pp. 1057–1065.

Tuckwell, Henry C (1988). *Introduction to theoretical neurobiology: volume 2, nonlinear and stochastic theories*. Vol. 8. Cambridge University Press.

Turner, Ryan D, Steven Bottone, and Bhargav Avasarala (2014). "A complete variational tracker." In: *Advances in Neural Information Processing Systems*, pp. 496–504.

Weigert, Martin et al. (2017). "Content-Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy." In: *bioRxiv*, p. 236463.

Weinstein, Matthew J and Anil V Rao. "Algorithm: ADiGator, a Toolbox for the Algorithmic Differentiation of Mathematical Functions in MATLAB Using Source Transformation via Operator Overloading." In:

Wilson, Rhodri S et al. (2016). "Automated single particle detection and tracking for large microscopy datasets." In: *Royal Society open science* 3.5, p. 160225.

Windle, Jesse, Nicholas G Polson, and James G Scott (2014). "Sampling Pólya-gamma random variates: alternate and approximate techniques." In: *arXiv preprint arXiv:1405.0506*.

Yoon, Young-Gyu et al. (2017). "Feasibility of 3D Reconstruction of Neural Morphology Using Expansion Microscopy and Barcode-Guided Agglomeration." In: *Frontiers in computational neuroscience* 11.

Zhu, Lei et al. (2012). "Faster STORM using compressed sensing." In: *Nature methods* 9.7, pp. 721–723.

Appendix

Here is the full paper for Chapter 4: Computational Analysis for NeuroPAL

"NeuroPAL: A Neuronal Polychromatic Atlas of Landmarks for Whole-Brain Imaging in C. elegans". Full paper link can be found: `https://www.biorxiv.org/content/10.1101/676312v1.abstract`