Bayesian Image Recovery for Dendritic Structures Under Low Signal-to-Noise Conditions

Geoffrey Fudenberg and Liam Paninski

***Abstract*—Experimental research seeking to quantify neuronal structure constantly contends with restrictions on image resolution and variability. In particular, experimentalists often need to analyze images with very low signal-to-noise ratio (SNR). In many experiments, dye toxicity scales with the light intensity; this leads experimentalists to reduce image SNR in order to preserve the viability of the specimen. In this paper, we present a Bayesian approach for estimating the neuronal shape given low-SNR observations. This Bayesian framework has two major advantages. First, the method effectively incorporates known facts about 1) the image formation process, including blur and the Poisson nature of image noise at low intensities, and 2) dendritic shape, including the fact that dendrites are simply-connected geometric structures with smooth boundaries. Second, we may employ standard Markov chain Monte Carlo techniques for quantifying the posterior uncertainty in our estimate of the dendritic shape. We describe an efficient computational implementation of these methods and demonstrate the algorithm's performance on simulated noisy two-photon laser-scanning microscopy images.**

***Index Terms*—Bayes procedures, image restoration, Monte Carlo methods.**

I. Introduction

In neuroscience, what we see often limits what we know. Improving imaging capabilities holds great promise for innovative experimental work. For example, quantitative analysis of dendritic spine morphology has the potential to teach us a great deal about synaptic transmission [1], [2] and long-term synaptic plasticity [3]. The opportunities for insight into the function of dendritic spines have already spurred computational work on automated analysis of dendritic spine morphology [4]–[8].

Clearly, quantitative experiments can directly benefit from improved imaging procedures. In lieu of designing a new imaging apparatus, this paper focuses on methods for making better use of currently obtainable data, especially within a sparse low signal-to-noise (SNR) regime. As a key example, we focus on algorithms for recovering two-photon laser scanning microscopy (TPLSM) images. While TPLSM offers increased optical resolution without increased phototoxicity as compared with conventional confocal microscopy [9], phototoxicity persists within the focal slice and limits attainable resolution: to avoid damaging the sample, imaging at long time scales or high intensities is impossible. Since photon detection during imaging can be modeled as a Poisson process [10], in which the signal-to-noise varies directly with the total number of photons absorbed, these biophysical limitations on the maximal intensity force us to deal with low-SNR images.

In general, the image recovery problem may be posed as follows: we make noisy, blurred observations of some underlying "true" neural shape $S_{\rm true}$—schematically

$$I_{\rm obs} = S_{\rm true} * w + \text{noise}$$

where $I_{\rm obs}$ denotes the observed image data, and $*$ denotes a convolution by w, the point-spread function (or PSF)—and our goal is to recover the true input image as faithfully as possible. Of course, exact recovery of this true input image is rarely feasible, and so we also need to know how uncertain we are about our estimate: in a sense, we would like to put "errorbars" around our estimate $\hat{S}$.

Many groups have contributed to the general problem of restoring noisy blurred images using maximum likelihood (ML) approaches [11], [12]. For the case of Poisson noise, Richardson and Lucy [13], [14] independently introduced iterative deconvolution algorithms. Significant research has been devoted to developing Bayesian statistical machinery for use in astrophysical [15] and medical imaging problems, especially positron emission tomography (PET) [16], [17].

We build upon this extensive previous imaging literature by incorporating important prior information about dendrites. In particular, we know that dendrites are simply connected geometrical structures, with fairly smooth boundaries. We may further restrict our attention to binary images: a given pixel may be either inside or outside the dendrite.[1] Furthermore, we focus our attention on cases where it is possible to generate an initial binary image with the correct topology. Thus, conceptually, to recover the true neural shape we need only search over the space of topologically equivalent binary images with smooth boundaries (of course, we do not attempt a brute-force search over this space, which would be computationally impractical). By combining the statistical model for image degradation with this *a priori* information, we can apply powerful likelihood-based tools from Bayesian statistics to the problem of optimally recovering dendritic shape (including spine size, etc.). In particular, we develop efficient Markov chain Monte Carlo (MCMC)

Manuscript received September 12, 2007; revised October 06, 2008. Current version published February 11, 2009. This work was supported in part by a Rabi Scholarship, in part by a Goldwater Scholarship, in part by a National Science Foundation CAREER award, and by in part by an Alfred P. Sloan Research Fellowship. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Gaudenz Danuser.

The authors are with the Department of Statistics and Center for Theoretical Neuroscience, Columbia University, New York, NY 10027 USA (e-mail: geoff.fudenberg@gmail.com; liam@stat.columbia.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2008.2010212

[1]Note that this is equivalent to the assumption that neuronal fluorescence is uniform inside the neuron, i.e., the dye concentration equilibrates completely. This assumption has been debated in the literature [18], [19] but provides a reasonable starting point for the analysis presented here.

methods for computing the optimal estimate of the underlying neuronal shape, while at the same time quantifying our uncertainty about this estimate.

II. Blurred Poisson Image Formation Model

We begin by describing our model of how the observed image depends on the true underlying neuronal shape. Define $I_{\rm obs}(s,t,u)$ to be the photon count observed in the (s,t,u)th pixel, and $S_{\rm true}(x,y,z)$ to be the true underlying (unobserved) neuronal shape: $S_{\rm true}(x,y,z)$ is either one or zero, depending on whether the location (x,y,z) is inside or outside of the neuron, respectively. (Note that the pixelization of $S_{\rm true}$ is user-defined—we can attempt to reconstruct the neuronal shape at any resolution we desire—while the pixelization of $I_{\rm obs}$ is set by the imaging apparatus. Therefore, the observed pixelization may generally be coarser than that of $S_{\rm true}$.)

The observed photon counts per pixel $I_{\rm obs}(s,t,u)$ are assumed to be discretized observations of a Poisson process; the rate of each individual Poisson count at pixel (s,t,u) is given by the convolution

$$\lambda(s,t,u) = \int_{s-ds/2}^{s+ds/2}\int_{t-dt/2}^{t+dt/2}\int_{u-du/2}^{u+du/2}[\lambda_{\rm out}+(\lambda_{\rm in}-\lambda_{\rm out}) S_{\rm true}(x,y,z)]w(s-x,t-y,u-z)dxdydz \quad (1)$$

where $w(x,y,z)$ is the point-spread (blur) function, or PSF, and $\lambda_{\rm out}$ represents the baseline fluorescence outside of the neuron and $\lambda_{\rm in}$ represents the internal fluorescence; typically, $\lambda_{\rm in}$ is greater than $\lambda_{\rm out}$. A reasonable definition of the signal-to-noise ratio here is

$$\mathrm{SNR} = \frac{\lambda_{\rm in}-\lambda_{\rm out}}{\sqrt{\lambda_{\rm out}}}$$

this is the difference between the mean fluorescence inside and outside the neuron, normalized by the standard deviation of the baseline (extracellular) photon count in a pixel of size one.

The image space $\mathcal{S}$ is the space of all binary simply connected shapes; that is, all possible shapes S such that $S(x,y,z)$ is either one or zero (inside or outside the neuron, respectively) and such that the interior (the collection of pixels (x,y,z) satisfying $S(x,y,z)=1$) is simply connected: we may connect any two "inside" pixels by a (possibly curved) continuous path which lies entirely inside the neuron. Thus, we implicitly assume that the pixelization (x,y,z) is sufficiently fine that no pixels are partially inside and partially outside the neuron. We further assume (as discussed in the introduction above) that the fluorescent dye is fully equilibrated in the sample: there are no spatial fluctuations in $\lambda_{\rm in}$ (caused, e.g., by fluctuations in internal dye concentration) or $\lambda_{\rm out}$.

Thus, the model is fully specified by the parameters $(S,\lambda_{\rm in},\lambda_{\rm out},w)$. This model is fairly standard in the imaging literature, and may be extended in a straightforward manner in a number of natural ways: non-Poisson noise, nonhomogeneous blurring, nonhomogeneous baseline fluorescence, etc. However, for simplicity we will only treat the homogeneous Poisson case here. We will further assume the parameters $(\lambda_{\rm in},\lambda_{\rm out},w(\cdot,\cdot))$ to be known properties of the fluorescent dye and imaging apparatus, respectively (though of course we may in general attempt to simultaneously infer these parameters, too; e.g., Fig. 5). Finally, we will restrict our attention to the 2-D case for illustrative purposes.

III. Computing the Likelihood

Now that our model is defined properly, our task is to quantify the posterior distribution $p(S\,|\,I_{\rm obs},\lambda_{\rm in},\lambda_{\rm out},w)$ over neuronal shapes S given the observed data $I_{\rm obs}$ and the imaging parameters $(\lambda_{\rm in},\lambda_{\rm out},w)$. By Bayes' rule, we know that we can express this posterior as a normalized product of two terms

$$p(S\,|\,I_{\rm obs},\lambda_{\rm in},\lambda_{\rm out},w)\propto p(I_{\rm obs}\,|\,S,\lambda_{\rm in},\lambda_{\rm out},w)p(S)$$

where the first term is the likelihood of observing $I_{\rm obs}$ given S and the second term is the prior probability of S.

We will begin by examining the likelihood. Our model states that the photon counts in each individual pixel constitute independent Poisson random variables. Thus, the likelihood of observing a count of n_i in the ith pixel is given by the Poisson distribution with rate λ_i

$$\mathrm{Poiss}(n_i\,|\,\lambda_i)=\frac{e^{-\lambda_i}\lambda_i^{n_i}}{n_i!}\quad n_i=0,1,2,\ldots$$

where λ_i is calcuated via (1). Now, to obtain the likelihood of the observed array $\{n_i\}$ of counts (where i indexes every pixel in $I_{\rm obs}(s,t,u)$), we simply form the product

$$p(I_{\rm obs}\,|\,S,\lambda_{\rm in},\lambda_{\rm out},w)=\prod_i\frac{e^{-\lambda_i}\lambda_i^{n_i}}{n_i!}.$$

It is convenient to work with the log-likelihood instead

$$\log p(I_{\rm obs}\,|\,S,\lambda_{\rm in},\lambda_{\rm out},w)=\sum_i(n_i\log\lambda_i-\lambda_i)+\mathrm{const.} \quad (2)$$

where the constant does not depend on S, and may, therefore, be ignored.

IV. Maximum Likelihood Estimation

Before we describe methods for computing the posterior $p(S\,|\,I_{\rm obs})$, it is worth examining the somewhat simpler problem of computing the maximum likelihood estimate for S

$$\hat{S}_{\rm MLE}=\arg\max_{S\in\mathcal{S}}p(I_{\rm obs}\,|\,S)$$

(we have suppressed the dependence of the likelihood function $p(I_{\rm obs}\,|\,S)$ on the imaging parameters $(\lambda_{\rm in},\lambda_{\rm out},w)$ for simplicity).

To compute $\hat{S}_{\rm MLE}$, we must search over the image space $\mathcal{S}$. Of course, this space is far too large to search directly; thus, some kind of local search algorithm is necessary. We will describe the simplest version of this local search, in which we iteratively change the state of one pixel at a time in our current estimate of the true underlying neuronal shape S. It is important to note that $\mathcal{S}$ is connected: any two simply connected shapes

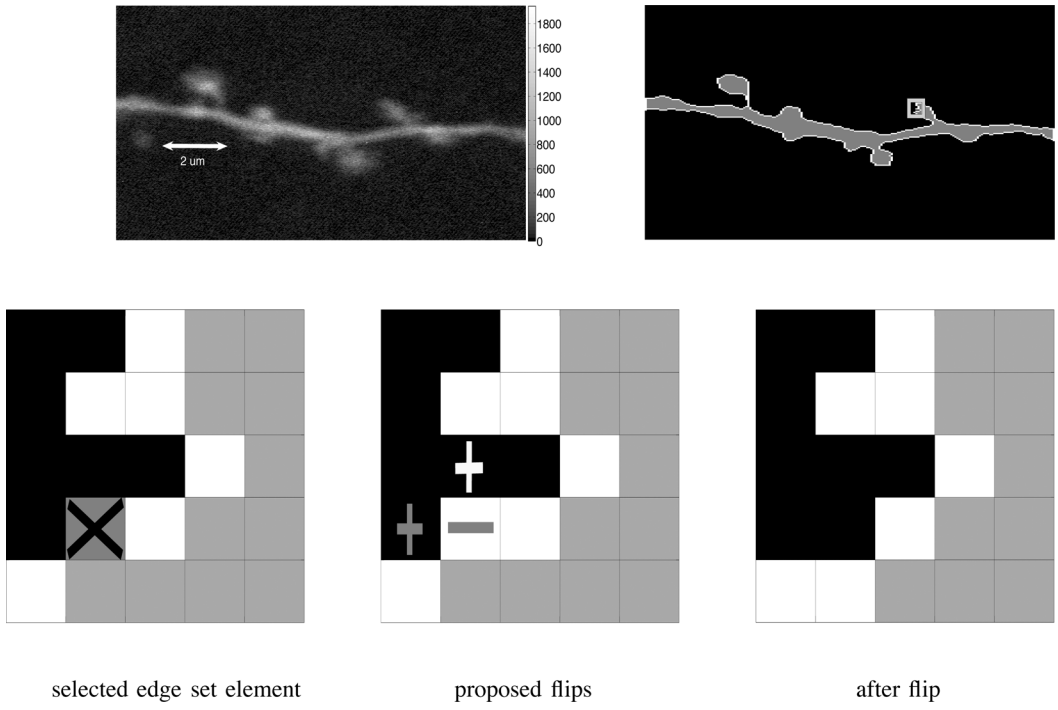

Fig. 1. Illustration of iterative pixel-flipping algorithm. **Top Left**: Raw sample image of a dendrite segment (courtesy of R. Araya and R. Yuste); colorbar indicates the observed counts per pixel. Image taken at 30 pixels per micron. **Top Right**: A sample S, showing interior (gray and white), exterior (black), and boundary set (white) pixels. The rectangle encloses the area around the pixels displayed in the lower three panels. **Lower Panels**: These three panels illustrate a local area of the neuron before and after flipping a selected edge set pixel from inside to outside the neuron. **Left**: We begin by randomly choosing a pixel in the boundary set (dark grey pixel marked with black X). **Middle**: Negative $(-)$ signs mark possible removals in the neighborhood of the selected pixel; positive $(+)$ signs show possible flips which would add exterior neighbors. Gray symbols are the allowed flips (these do not violate simple connectivity), while the white symbol marks a forbidden flip: adding this pixel would create a hole in S, violating simple connectivity. If a proposed flip is accepted, elements of the edge set must be updated. **Right**: The edge set has been updated upon removing the allowed gray $(-)$ pixel.

are topologically equivalent, in the sense that we can continuously deform one shape into another (no "tearing" or "gluing" is allowed). In this setting, continuous deformations are constructed by composing a sequence of single pixel flips which do not "tear" S (break it up into two disconnected pieces) or "glue" S on itself (form a loop of the dendritic shape onto itself). However, while the underlying space $\mathcal{S}$ is connected in this sense, the likelihood function typically has many local optima, and, therefore, local search techniques must be supplemented with a randomized or multistart strategy (e.g., simulated annealing) in order to find the global optimum. We provide details of the local search strategy in the following subsections. For information on our particular initialization procedure, see the Appendix.

A. Boundary Set

On each iteration, we choose a pixel randomly and decide whether or not to "flip" it (i.e., add the pixel to the interior set if it is currently outside, or remove the pixel if it is inside). Before deciding whether or not to flip a given pixel, we must first test whether flipping the pixel will preserve the simply connected structure of S. Clearly, we are not allowed to flip pixels which are not located on the boundary of S, since flipping a pixel which is not touching S will create a disconnected shape (the current S plus a disconnected pixel), and flipping a pixel which is on the interior of S but not on the boundary will form a hole, therefore breaking the simple connectivity of S. Thus, we may restrict our attention to the boundary set of S: the set of all pixels which are in the interior of S but which contact at least one pixel on the exterior. For concreteness, we say that a pixel contacts another if they are direct neighbors either horizontally or vertically (i.e., diagonal contacts are not considered). An example of an edge pixel flip is shown in Fig. 1.

After each accepted flip, we must update the boundary set, by removing any pixels in the interior which are now not touching the exterior and adding any interior pixels which are now on the boundary. As we discuss below, this is a local computation: we do not need to rescan the entire image after each accepted flip; instead, we need only check the four neighbors of the flipped pixel.

B. Simple Connectivity Constraint

Each possible flip is checked to ensure maintenance of the simple connectivity constraint. While simple connectivity is a global topological constraint, we can enforce it locally in two dimensions [20]. Mathematically, this is possible by forbidding topological changes: flips which either cut a region in two (tearing) or join two unconnected regions together (gluing). We may check these constraints in a computationally efficient, local manner by requiring the number of connected regions in a 3×3 neighborhood of the proposed pixel (i.e., the eight horizontal, vertical, and diagonal neighbors) to remain constant when we flip the pixel from the neuron.

This check on the connectivity may be performed using, e.g., the bwlabel.m function in Matlab, which serves to count all the

connected regions in a given image. However, it is inefficient to perform this check every time we want to flip a pixel. Instead, we cache 3×3 patches that we have tested previously, along with whether the flip was accepted or not. Now for each new patch that we need to test, we need only check to see if we have checked this patch before, and if so, whether it led to a valid flip.

In the data we present, we assume only one neuron is labeled. This has an important correspondence with experiments, as some techniques label single neurons, while others typically label many cells. However, our algorithm maintains the topology of initialization—if two neuronal structures started out next to each other, but were separate, then the algorithm would keep them separate. Again, this behavior relies on the fact that by rejecting changes in local topology, we maintain global topology. If the algorithm was used to recover an area with two nearby neurons, and was initialized with separated shapes, it might incorrectly assign a spine to the wrong neuron—leading to a less favorable recovery—but the two structures would remain separate.

C. Local Updating of the Loglikelihood

Once we have decided that a given pixel flip is acceptable (i.e., it maintains simple connectivity of S), we need to decide whether the flip will increase the likelihood. In a strict ascent algorithm, we will only accept a flip if it increases the likelihood function, while in a simulated annealing (randomized) algorithm, the probability of accepting a flip increases as a function of the ratio between the likelihood of S after and before the flip. In either case, we need to compare the postflip and preflip likelihood.

Computing the log-likelihood (2) requires that we compute λ_i (1) and then perform a sum over all pixels i. However, if the point-spread function $w(\cdot)$ has finite support, then updating the log-likelihood requires just a simple (fast) local computation. To see this, we expand the log-likelihood ratio

$$\begin{aligned}
&\log \frac{p(I_{\text{obs}} \mid S_{\text{after}})}{p(I_{\text{obs}} \mid S_{\text{before}})} \\
&\quad = \log p(I_{\text{obs}} \mid S_{\text{after}}) - \log p(I_{\text{obs}} \mid S_{\text{before}}) \\
&\quad = \sum_i \left(n_i \log \lambda_i^{(\text{after})} - \lambda_i^{(\text{after})} \right) \\
&\qquad - \sum_i \left(n_i \log \lambda_i^{(\text{before})} - \lambda_i^{(\text{before})} \right) \\
&\quad = \sum_i \left[n_i \left(\log \lambda_i^{(\text{after})} - \log \lambda_i^{(\text{before})} \right) \right. \\
&\qquad \left. - \left(\lambda_i^{(\text{after})} - \lambda_i^{(\text{before})} \right) \right] \\
&\quad = \sum_j \left[n_j \left(\log \lambda_j^{(\text{after})} - \log \lambda_j^{(\text{before})} \right) \right. \\
&\qquad \left. - \left(\lambda_j^{(\text{after})} - \lambda_j^{(\text{before})} \right) \right] \\
&\quad = \sum_j n_j \left(\log \lambda_j^{(\text{after})} - \log \lambda_j^{(\text{before})} \right) \\
&\qquad - \sum_j \left(\lambda_j^{(\text{after})} - \lambda_j^{(\text{before})} \right)
\end{aligned}$$

where the sum over j is only over those pixels for which $\lambda_j^{(\text{after})} \neq \lambda_j^{(\text{before})}$. In many applications, $w(\cdot)$ is only a few pixels wide, and, by linearity of convolution, changing one pixel in S will only affect λ over a few pixels. The sum over j will be much smaller (and faster to compute) than the sum over i. Furthermore, the second sum in the last line above is in fact a constant, again by linearity of convolution

$$\sum_j \left(\lambda_j^{(\text{after})} - \lambda_j^{(\text{before})} \right) = \pm(\lambda_{\text{in}} - \lambda_{\text{out}})W$$

where W is the space integral of the point spread function $w(\cdot)$ and the $\pm$ is taken to be positive when the pixel is flipped in (added to the interior set) and negative when the pixel is flipped out (removed from the interior). So, finally, we obtain

$$\begin{aligned}
&\log \frac{p(I_{\text{obs}} \mid S_{\text{after}})}{p(I_{\text{obs}} \mid S_{\text{before}})} \\
&\quad = \sum_j n_j \left(\log \lambda_j^{(\text{after})} - \log \lambda_j^{(\text{before})} \right) \pm (\lambda_{\text{in}} - \lambda_{\text{out}})W.
\end{aligned}$$

Finally, note that we only need to compute λ_j for pixels such that n_j is positive. We achieve this by employing sparse matrices to store the location of positive n_j for a tested pixel as we sample from S. Thus, updates for the loglikelihood may be computed quite quickly.

D. Direct Maximum Likelihood Performs Poorly

The performance of the maximum likelihood estimator $\hat{S}_{\text{MLE}}$ is illustrated in Fig. 2. We started with the binary test image S_{true} on the left, then convolved this image with an isotropic Gaussian point-spread function $w(\cdot)$ (the standard deviation of this Gaussian was taken to be 3 pixels here), then sampled photon counts $n(x, y)$ from the blurred Poisson model as described above. The images of $\hat{S}_{\text{MLE}}$ are arranged opposite data with increasing SNR below the "true" shape. The top row uses $\lambda_{\text{in}} = 2\lambda_{\text{out}}$, the middle row uses $\lambda_{\text{in}} = 5\lambda_{\text{out}}$, and the bottom row uses $\lambda_{\text{in}} = 500\lambda_{\text{out}}$. For each set of data we computed $\hat{S}_{\text{MLE}}$ by direct ascent methods, but similar results are observed when various forms of simulated annealing are employed.

The main result evident here is that directly computing $\hat{S}_{\text{MLE}}$ leads to rather poor image recovery in the low-SNR regime. Roughly speaking, the MLE tries to include all pixels where many photons have been detected and to exclude all pixels in which no photons are detected; under the simple-connectivity constraint, this behavior leads to the undesirable "tendril" (high perimeter-to-area ratio) structures seen in Fig. 2. This behavior is in fact evident over a fairly wide range of SNR regimes; only when the SNR is high enough does the MLE become a viable estimator for S.

V. Penalized Maximum Likelihood Leads to Better Image Reconstruction

Given our prior knowledge about the relative smoothness of neuronal edges, we know that the overly "hairy" recovered neuronal shape shown in Fig. 2 is inaccurate. We would like to build this *a priori* information directly into our estimator. One direct and fairly classical way to do this is to maximize a penalized likelihood instead of maximizing the likelihood directly:

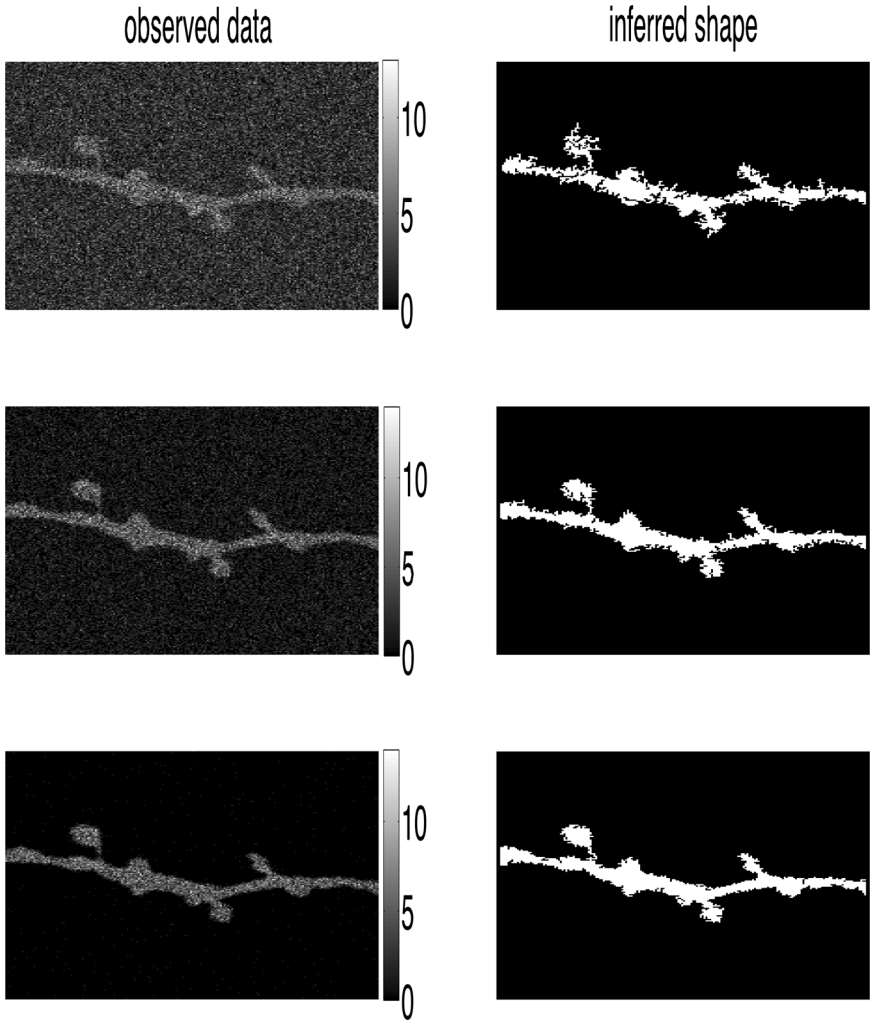

Fig. 2. Unpenalized MLE gives an overly "stringy" estimate of the underlying neuron shape. This figure shows how MLE performance scales with increasing SNR. The left column displays $I_{\rm obs}$ (simulated) with increasing SNR generated by the blurred Poisson model and the right column shows the corresponding $\hat{S}_{\rm MLE}$. The top row uses $\lambda_{\rm in} = 2\lambda_{\rm out}$, the middle row uses $\lambda_{\rm in} = 5\lambda_{\rm out}$, and the bottom row uses $\lambda_{\rm in} = 500\lambda_{\rm out}$. With low-SNR the unpenalized MLE clearly fails to accurately recover the original image; note the undesirable "tendril" (high perimeter per area) structures. As SNR increases, however, the data more adequately constrains recovery and $\hat{S}_{\rm MLE}$ approaches $S_{\rm true}$.

we may obtain a smoother reconstruction by penalizing flip proposals which make the edge more jagged. Thus, we maximize the penalized loglikelihood

$$\log p(I_{\rm obs} \,|\, S) - Q(S) \tag{3}$$

instead of just the loglikelihood $\log p(I_{\rm obs} \,|\, S)$. We want to choose the penalty function $Q(S)$ so that $Q(S)$ becomes larger as S becomes more jagged. However, we constrain ourselves to penalty functions Q which may be updated through strictly local computations so that our iterative algorithm (which requires many pixel flips) remains computationally tractable. As usual, this penalized likelihood has a natural Bayesian interpretation: if our prior distribution on images is of the form

$$p(S) \propto e^{-Q(S)}$$

then the penalized loglikelihood is just the log-posterior under this prior, and the maximum penalized likelihood estimator is equivalent to the maximum *a posteriori* estimator.

Fig. 3 illustrates the behavior of a penalized MLE on the data shown in the middle row of Fig. 2. We found the following $Q(S)$ to be a simple, effective penalizer here:

$$Q(S) = \alpha_1 Q_1(S) + \alpha_2 Q_2(S)$$

where

$$Q_1(S) = \text{number of exterior pixels neighboring a member of the boundary set of } S$$
$$Q_2(S) = \text{number of elements in the boundary set of } S$$

and α_1 and α_2 are adjustable parameters. In this case, both $Q_1(S)$ and $Q_2(S)$ serve to measure the roughness of the boundary of S, and may both be computed efficiently via simple local computations (specifically, counting the number of pixels in or near the boundary set of S); we found empirically that a combination of these two penalties leads to better recovery of the true shape S, since $Q_1(S)$ alone favors rectangular, blocky edges, while $Q_2(S)$ selects for diagonal edges.

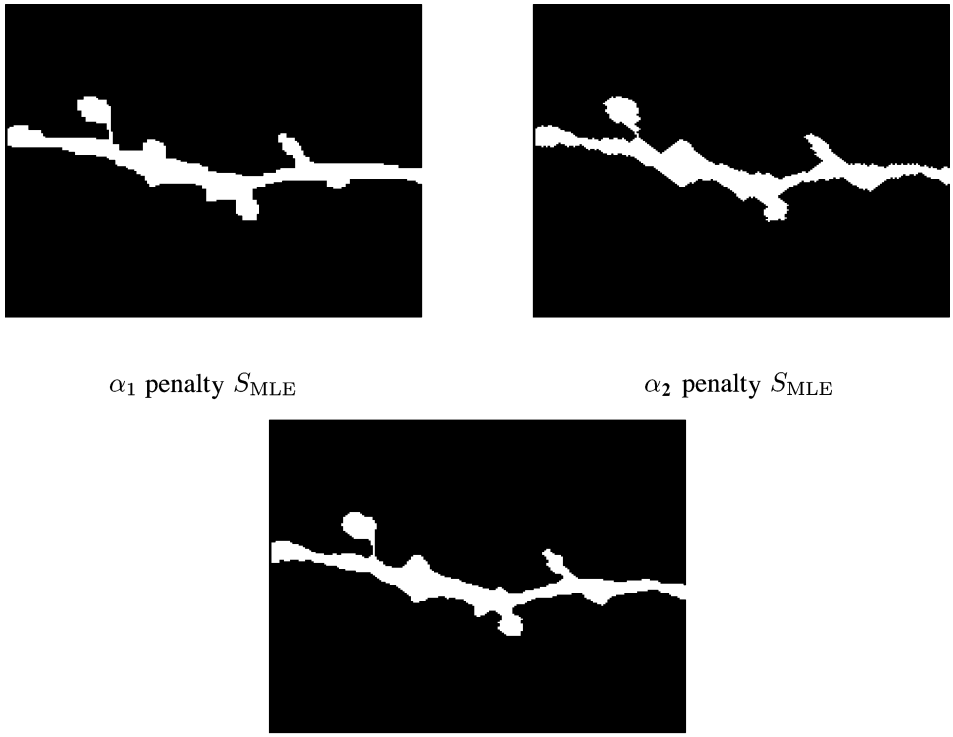

Fig. 3. Illustration of the edge-length penalties $Q_1(S)$ and $Q_2(S)$. Each panel displays the penalized MLE, computed with $\alpha_2 = 0$ (left) and $\alpha_1 = 0$ (right), given the observed data shown in the middle panel ($\lambda_{\mathrm{in}} = 5\lambda_{\mathrm{out}}$) in Fig. 2. In each case, the penalized MLE eliminates the "tendrils" seen in the unpenalized MLE; however, employing $Q_1(S)$ or $Q_2(S)$ alone leads to somewhat blocky reconstructed images. By combining penalties (i.e., setting both α_1 and α_1 positive), we can more optimally recover a neuronal shape.

A. *Fitting the Smoothing Penalty Parameters via Cross-Validation*

The definition of the penalty function $Q(S)$ above poses the obvious, yet nontrivial, problem: How do we optimally choose the penalty weights α_1, α_2? A satisfactory analytic treatment of this problem has proven elusive; any definition of the "optimal" penalty value would depend upon many parameters (e.g., the signal-to-noise ratio of the observed data, the smoothness of the true underlying image, etc.). Moreover, direct numerical minimization of the penalized loglikelihood (3) as a function of (S, α_1, α_2) clearly leads to the unpenalized maximum likelihood solution (i.e., α_1 and α_2 are just set to zero), which is clearly not the solution we are looking for (cf. Fig. 2). Thus, we took a cross-validation approach to optimizing the penalty values here.

Cross-validation refers to the technique of splitting the observed data into two components, the "training" set and the "held-out" (or "test") set. Fitting is performed on the training data, and then the performance of the estimator (under various values of the penalization parameters) is tested on the held-out data. Using cross-validation, we then choose the penalization parameters which perform best on the held-out test data. The best-performing parameters are those for which the penalized MLE maximizes the (unpenalized) likelihood of the test data.

In the interest of minimizing imaging damage to the sample, we would like to be able to fit the penalty parameters given a single image. Thus, we randomly hold out a fraction of the observed *pixels* in a single image. This entails a straightforward change in our loglikelihood function: instead of computing the sum over all pixels i in expression (2), we compute the sum over all *observed* pixels i' instead—we simply discard the held-out test pixels from the sum.

Thus, for each candidate value of the penalization parameters (α_1, α_2), we compute

$$\hat{S}_{\alpha_1,\alpha_2} = \arg\max_{S\in\mathcal{S}} \sum_{i'} \left(n_{i'} \log \lambda_{i'} - \lambda_{i'}\right) - \alpha_1 Q_1(S) - \alpha_2 Q_2(S)$$

where the fluorescence rate at the i'th pixel given S, $\lambda_{i'}$, is computed as described above. Then to choose the best (α_1, α_2), we compare the loglikelihood each $\hat{S}_{\alpha_1,\alpha_2}$ assigns to the test data

$$\sum_{j\in\mathrm{held-out}} (n_j \log \lambda_j - \lambda_j)$$

where the sum is over all the held-out test pixels and now λ_j is computed given the shape $\hat{S}_{\alpha_1,\alpha_2}$ computed using the training data only.

Applying this method returns nonzero values for penalty parameters (α_1, α_2) since—as shown in Fig. 2—naïve MLE reconstruction at low-SNR is underconstrained by the data. Imposing the edge penalty constrains image recovery and increases the loglikelihood of the the held-out test data. The success of our cross-validation approach is shown in Fig. 4: We see that the cross-validated test loglikelihood does a good job of choosing penalty values which lead to the best-recovered neuronal shapes S for known $\lambda_{\mathrm{in}} = 5\lambda_{\mathrm{out}}$. The recovery using parameters chosen by cross-validation incorrectly classifies

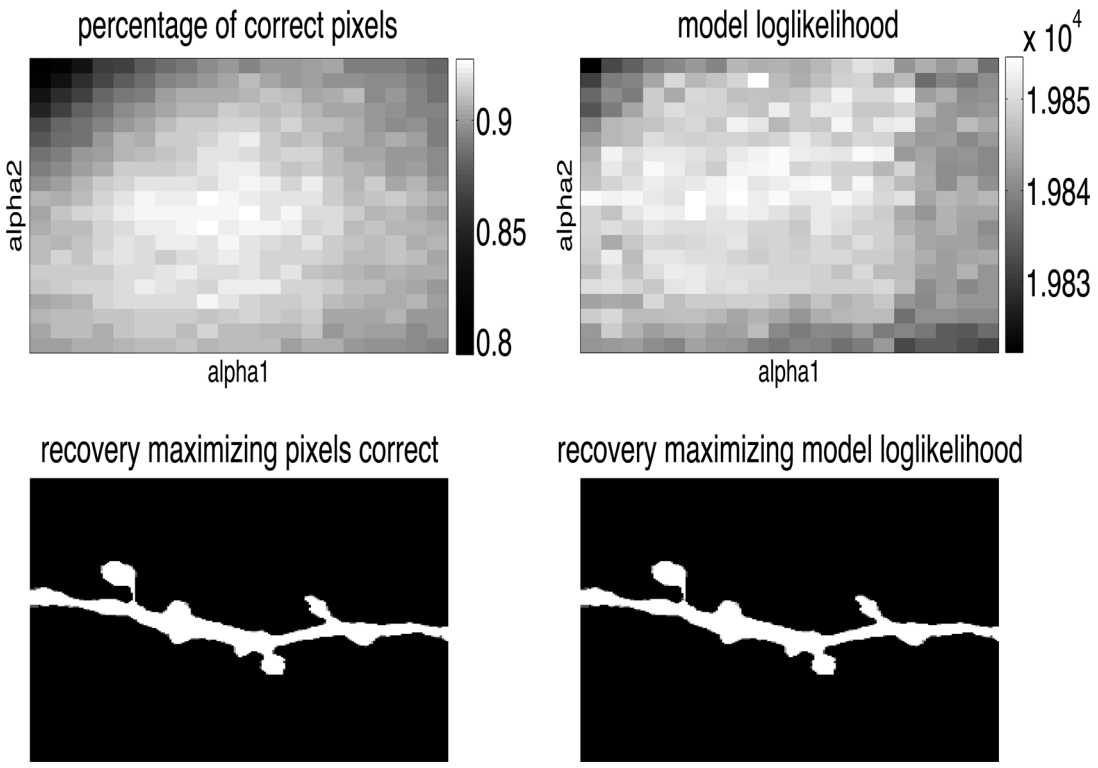

Fig. 4. Illustration of the performance of cross-validation in selecting the optimal penalization parameters (α_1, α_2) for data shown in middle panel of Fig. 2 ($\lambda_{\text{in}} = 5\lambda_{\text{out}}$). **Top**: Comparison of cross-validated test loglikelihood surface (right) versus percentage of pixels which were correctly classified as being inside or outside of the true neuron S_{true} (left), as a function of (α_1, α_2). In this case, the percent correctly classified is calculated as $(1 - (\text{erroneously classified pixels}/\text{number of pixels in } S_{\text{true}}))$. The two surfaces match each other qualitatively, indicating that cross-validation is an effective technique for selecting (α_1, α_2). **Bottom Left**: Recovered shape $\hat{S}_{\alpha_1,\alpha_2}$ which maximizes the number of recovered pixels. White and black indicate pixels where the two agree, light grey denotes misses, and dark grey shows false positives. **Bottom Right**: $\hat{S}_{\alpha_1,\alpha_2}$ selected by using cross-validation to fit (α_1, α_2). The two estimated shapes are comparable recoveries of the true shape, the respective percentages of erroneously classified pixels divided by number of pixels in S_{true} are 7.00% and 7.36%. The comparable fidelity of these recoveries again show that cross-validation is an effective technique for selecting (α_1, α_2).

only a slightly higher number of pixels than the parameters which best recovered the original image; the respective percentages of erroneously classified pixels divided by number of pixels in S_{true} are 7.00% and 7.36%.

Running the algorithm on similar simulated data as in Fig. 4, we are able to simultaneously estimate the penalty parameters (α_1, α_2) and fluorescence levels $(\lambda_{\text{in}}, \lambda_{\text{out}})$. Of course, this full search covers a larger parameter space and is, therefore, slower. Fig. 5 shows the loglikelihood surface as a function of λ_{in} and λ_{out} at optimal values of α_1, α_2. We see the surface is fairly well behaved; we may fit fluorescence levels in addition to penalty values for (α_1, α_2). Cross-validation, then, can robustly determine favorable image recovery parameters.

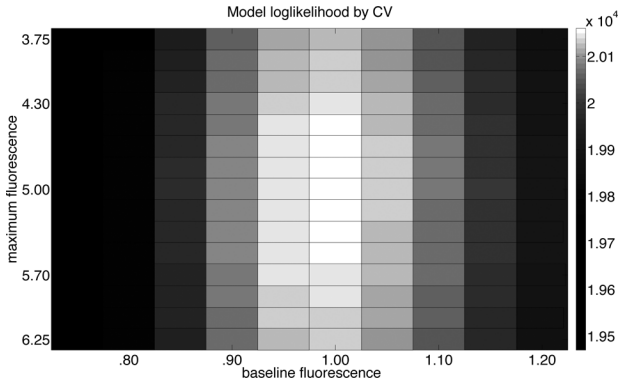

Fig. 5. Cross validation can be used to simultaneously determine optimal penalty parameters (α_1, α_2) and fluorescence levels $(\lambda_{\text{in}}, \lambda_{\text{out}})$; here we show the loglikelihood surface as a function of λ_{in} and λ_{out} at optimal values of α_1, α_2. As with Fig. 4, this figure illustrates performance using same-SNR data ($\lambda_{\text{in}} = 5\lambda_{\text{out}}$) as in the middle panel of Fig. 2. λ_{in} vary respectively from .75 to 1.25 times and .75 to 1.20 times the true values used to generate the data. The maximal loglikelihood value exactly coincides with fluorescence levels used for data generation. Note that our estimate for λ_{out} is more strongly constrained by the data, since there are many more pixels (i.e., more information) outside the dendrite than inside.

VI. Markov Chain Monte Carlo Methods for Sampling From $p(S \mid I_{\text{obs}})$

Now we may finally turn to the primary goal of this paper, which is to develop fully Bayesian methods for quantifying our knowledge of the neuronal shape S given the observed image data I_{obs}. Up until now, we have discussed techniques for computing estimators $\hat{S}$ which (locally) maximize the posterior $p(S \mid I_{\text{obs}})$. However, it is equally important to quantify our uncertainty in these estimates. One standard way to do this is to draw samples $S^{(i)}$ from the posterior $p(S \mid I_{\text{obs}})$, and then to quantify the variability of these samples. Directly sampling from these posteriors is infeasible, but standard Markov Chain Monte Carlo [21] methods may be employed easily.

We implemented a simple Gibbs sampler [22] here. The idea is that we iteratively choose a pixel i randomly and sample

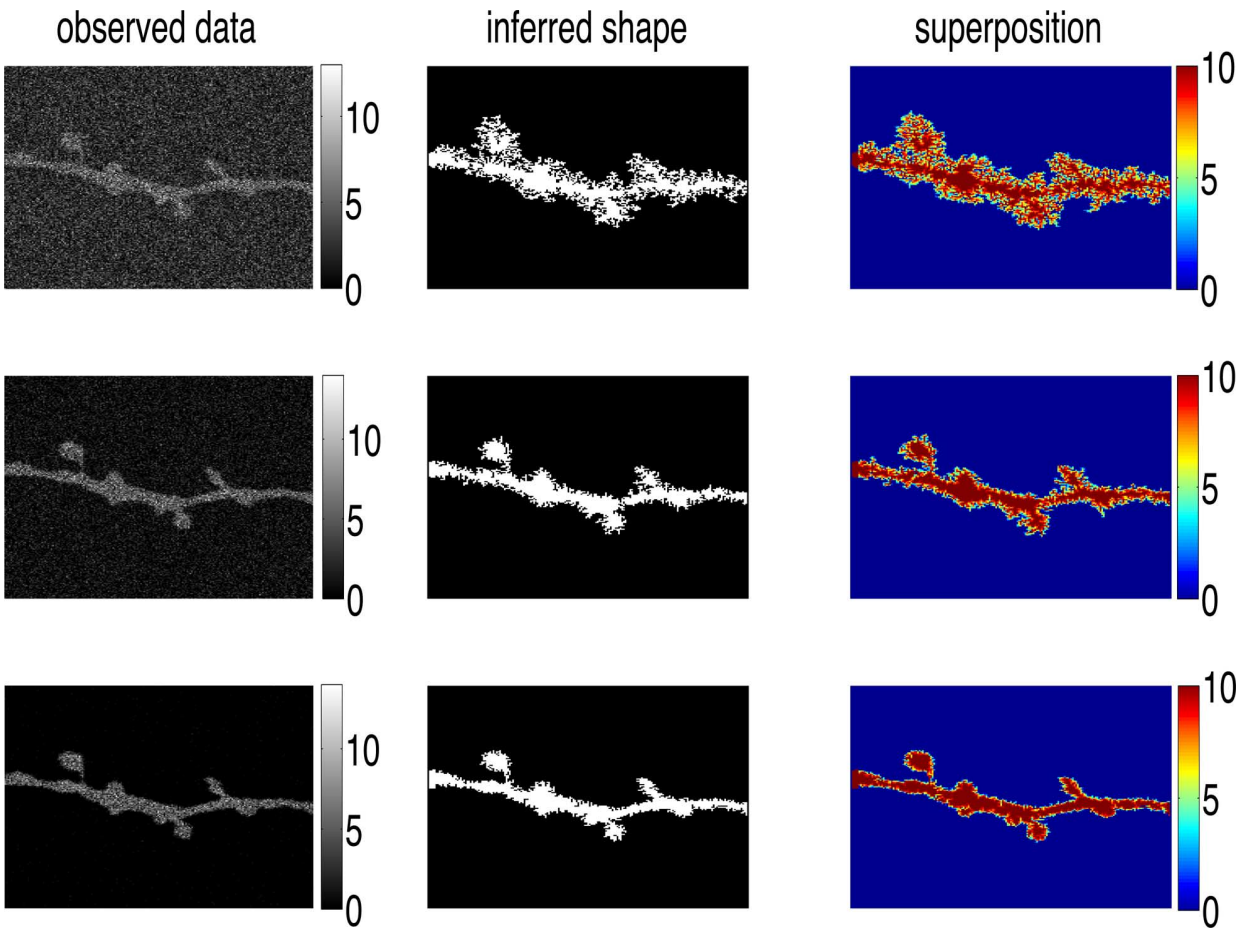

Fig. 6. This figure displays how unpenalized MCMC sampling scales with increasing SNR. From the center and right columns, we gain a sense of the variability of samples, which decreases with increasing SNR. **Left Column**: $I_{\rm obs}$ (simulated) generated by the blurred Poisson model (same data as in Fig. 2). **Center Column**: A random sample S drawn during Gibbs sampling. **Right Column**: Pictures which show average shape of samples drawn using Gibbs sampling. Each image represents the square root of the superposition of 100 samples. Colorbar represents the percentage of samples that have a given pixel on the interior, red(10) means the pixel was on the neuronal interior on every sample, blue(0) means the pixel was on the exterior in every sample. Samples were drawn once every 200 proposals from the Markov chain after an initialization (burn-in) period of 5e3 proposals.

from $p(S_i \mid I_{\rm obs}, \alpha_1, \alpha_2, \{S_j\}_{i \neq j})$, the distribution of the states of the pixel S_i given the observed data $I_{\rm obs}$, the parameters (α_1, α_2) which specify our prior $p(S) \propto e^{-Q(S)}$, and the states $\{S_j\}_{i \neq j}$ of every other pixel in S. It is well known that this iterative sampling strategy leads to samples from the full posterior $p(S \mid I_{\rm obs}, \alpha_1, \alpha_2)$, although these samples will not be independent; see, e.g., [21] for further details and background on Gibbs sampling.

Thus, we only need to discuss the problem of sampling from $p(S_i \mid I_{\rm obs}, \alpha_1, \alpha_2, \{S_j\}_{i \neq j})$; this turns out to be quite easy given the methods we have already developed. We simply write out

$$\begin{aligned} &p(S_i \mid I_{\rm obs}, \alpha_1, \alpha_2, \{S_j\}_{i \neq j}) \\ &\quad \propto p(S_i, I_{\rm obs}, \{S_j\}_{i \neq j} \mid \alpha_1, \alpha_2) \\ &\quad = p(S_i, \{S_j\}_{i \neq j} \mid \alpha_1, \alpha_2) p(I_{\rm obs} \mid S_i, \{S_j\}_{i \neq j}) \\ &\quad = 1(S \text{ is simply connected})\, e^{-Q(S_i, \{S_j\}_{i \neq j})} \\ &\qquad \times p(I_{\rm obs} \mid S_i, \{S_j\}_{i \neq j}) \end{aligned}$$

where the first term in the last line is one or zero depending on whether the shape S made of the pixels $(S_i, \{S_j\}_{i \neq j})$ is simply connected. Luckily, we have already developed methods for efficiently checking simple connectivity and for computing and updating the penalty $Q(S)$ and the likelihood $p(I_{\rm obs} \mid S)$, one pixel S_i at a time. Thus, we see that sampling from the posterior $p(S \mid I_{\rm obs})$ via the Gibbs approach requires no more code than what we have already written to maximize $p(S \mid I_{\rm obs})$.

The behavior of the Gibbs sampler is illustrated in Figs. 6 and 7 (code to be made publicly available at http://www.stat.columbia.edu/~liam). With very low SNR, image recovery is not particularly constrained by the data. Fig. 6 shows the results of unpenalized sampling for increasing SNR (this is equivalent to sampling from the posterior given a "flat" prior, i.e., $p(S) = 1$ for all simply connected S). At the lowest SNR $\log p(I_{\rm obs} \mid S)$ displayed (top row), samples from the unpenalized Markov chain shows high variability; in fact the Gibbs chain does not equilibrate around $S_{\rm true}$. This analysis, therefore, indicates that the chain is relatively unconstrained by the data. If we increase the strength of the penalty, by raising α_1 and α_2, we can constrain image recovery more effectively (Fig. 7). The top row shows what happens when we consider penalized Gibbs sampling for the same lowest-SNR data. In this case, $\log p(I_{\rm obs} \mid S)$ equilibrates fairly quickly. Furthermore, sample shapes show the effect of the constraint and are much less variable than without a penalty. Thus, given appropriate prior information on dendritic shape, we can decrease the uncertainty

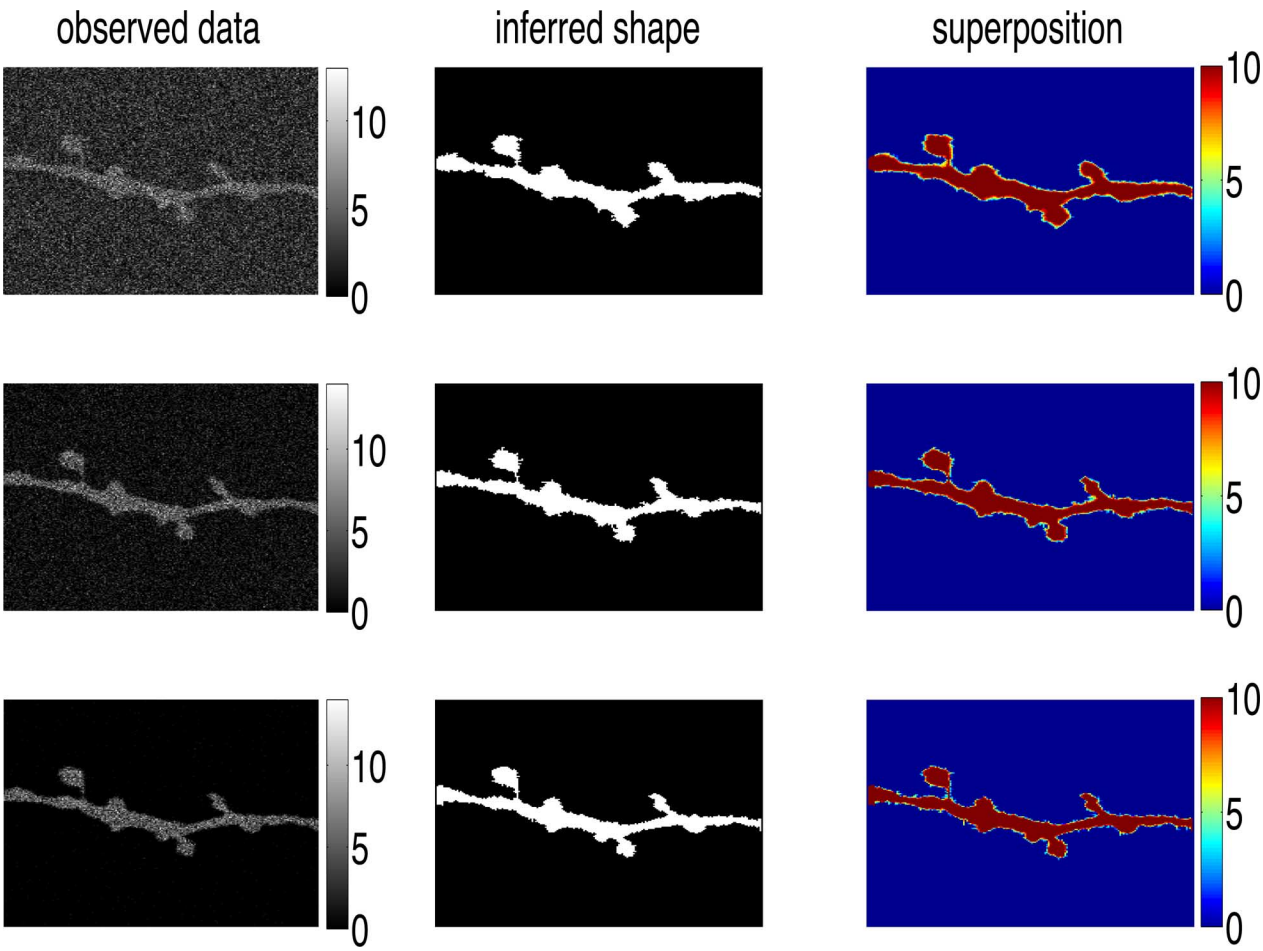

Fig. 7. This figure shows how penalized MCMC sampling scales with increasing SNR. From the center and right columns, we see that imposing a penalty term ($\alpha_1 = .2, \alpha_2 = 2$ here) constrains the variability of samples drawn using the Gibbs sampler. **Left Column**: I_{obs} (simulated) generated by the blurred Poisson model (same data as in Fig. 2). **Center Column**: A random sample S drawn during Gibbs sampling. **Right Column**: Pictures which show average shape of samples drawn using Gibbs sampling. Conventions as in Fig. 6. Note that penalization leads to much less variability—especially at low SNR—than is evident in Fig. 6.

in our recovered image, as expected. (In both figures, top rows have a ratio λ_{in}:λ_{out} of 2:1; the penalized Gibbs chain equilibrates in about 5e3 steps, or 45 s. All computations performed on a Pentium 4 CPU 3.40-GHz laptop with 1-GB RAM).

If we increase SNR instead of altering the penalty, the Gibbs sampler also coverges nicely to a stationary state and $\log p(I_{\text{obs}} \mid S)$ stabilizes quickly. The bottom row of Fig. 6 shows the lower sample variability and the better approximation of the true shape, indicating that the data adequately constrains recovery. Combined with the penalized sampling results, the results of increased SNR indicate that the Gibbs sampler can be used as a valuable tool to determine whether a dendritic image is in fact adequately constrained by the observed data (the higher SNR samples with ratios λ_{in}:λ_{out} of 5:1 and 500:1 equilibrate after around 7.5e3 steps, or 30 s, with or without the penalty; as in the case of lowest SNR we considered, sample shape and variability are visibly affected by the constraint).

VII. Discussion

We have introduced Bayesian methods for quantifying neuronal shape given low-SNR image observations. Our key insights are that: 1) neurons are simply connected topological structures, and this geometric constraint may be easily incorporated in algorithms for determining neural shape; 2) direct maximum likelihood estimation leads to poor recovery of the neuronal shape, while simple penalization methods perform much more effectively; and 3) Markov chain Monte Carlo techniques allow us to quantify the uncertainty in our estimates of the neuronal shape and can be used to determine whether this shape is actually constrained by the data. These aspects of our method make it a potentially useful front end for software whose goals include identifying spines or tracking neuronal branching, such as NeuronStudio (Computational Neurobiology and Imaging Center, Mount Sinai School of Medicine; http://www.mssm.edu/cnic/tools.html) or IMARIS (Bitplane AG. Zurich, Switzerland; http://www.bitplane.com/go/products/imaris).

Other groups have examined priors on roughness for imaging work. Good's roughness penalty [23] has received attention for use in biological imaging applications including tomography [24], confocal microscopy [25] and optical sectioning microscopy [26]. Good's penalty is a kind of weighted Laplacian penalty; there are close connections between the edge-based penalty introduced here and these Laplacian penalties, since the Laplacian serves to effectively detect edges in the case of the binary shapes S of interest here.

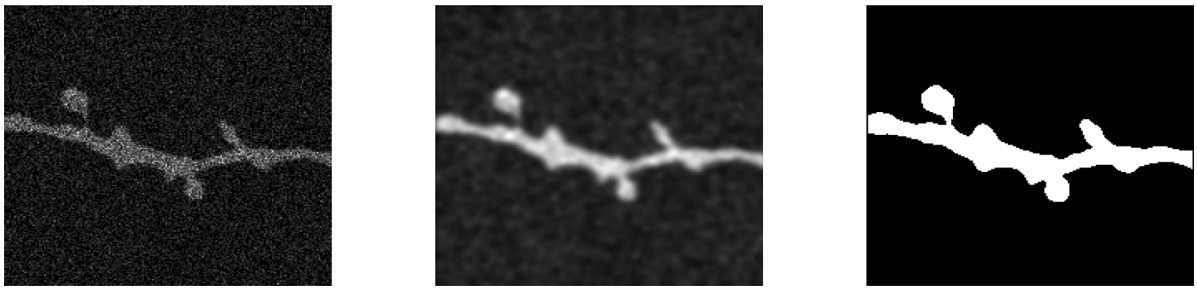

Fig. 8. This figure shows how we obtain a starting S from the observed data I_{obs}. **Left**: Observed data as in Fig. 2 (with $\lambda_{\mathrm{in}} = 5\lambda_{\mathrm{out}}$). **Center**: Smoothed data, η_{data}. **Right**: $S_{\mathrm{threshold}}$, obtained by thresholding η_{data}.

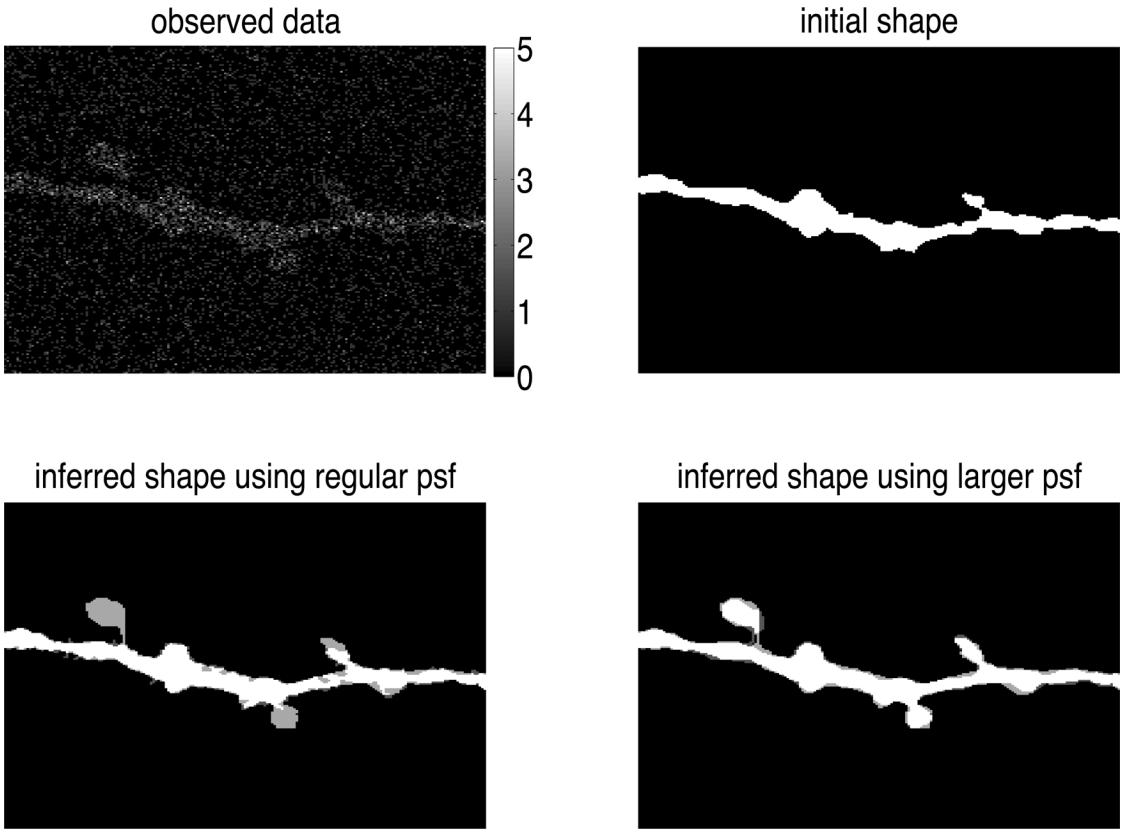

Fig. 9. Demonstration of the utility of prior optimization with an unphysically large psf for recovering images with even lower SNR. **Top Left**: Observed data using $(\lambda_{\mathrm{in}} = 5\lambda_{\mathrm{out}})$ but with λ_{in} at $(1/6)$ of its value in Fig. 2 etc. **Top Right**: Starting shape from thresholding. Poor starting points, such as this one, become increasingly common as SNR decreases and the data becomes more sparse. Optimization using an unphysically large psf can improve image recovery. **Bottom Left**: Comparison between $\hat{S}_{\mathrm{MLE}}$ and S_{true} after using the 'physical' psf, w, used to generate data. White and black indicate pixels where the two agree, light grey denotes misses, and dark grey shows false positives. **Bottom Right**: Comparison between $\hat{S}_{\mathrm{MLE}}$ and S_{true} after prior optimization with an unphysically large psf (4x spread). In the second case, the recovery is clearly more successful and $\hat{S}_{\mathrm{MLE}}$ much better approximates S_{true}. In particular, note how the top left spine, which recovery using the 'physical' psf misses entirely, is present after optimization with the larger psf.

A major direction for future work is to incorporate more "realism" into our priors. One attractive possibility has recently been developed by [27]: the idea is to develop priors which are truly "neuronal," instead of the simple edge-based priors we have used here. This could potentially lead to much more accurate recovery of the underlying neuronal shapes. In particular, we would like to use different priors for different types of neuronal types, which might differ in the roughness of the dendritic membrane, the number, size, and shape of dendritic spines, etc. The challenge here will be to incorporate this more detailed prior knowledge of neuronal shape while maintaining the local nature of the computations described here. Future work could also attempt to relax the assumption that neuronal fluorescence can be modeled as a step function, and account for subtleties related to dye distribution in spines and dendrites. Another important track involves translating our 2-D work into 3-D. Most critically, we must adapt methods from digital topology to efficiently enforce the more complicated 3-d simple connectivity constraint [28], [20].

Finally, we should emphasize that the Gibbs sampling methods described here, while simple to code and understand, are far from optimal in terms of computational speed. In fact, the problem of efficiently sampling from a binary Markov random field remains a topic of ongoing interest in applied

statistics, computer vision, and statistical physics. This body of work has a great potential for adaptation to our particular circumstances. Instead of flipping a single graph vertex, ideas like graph clustering and graph relabeling can be used to split, merge, and regroup chunks of the graph [29]. This drastically speeds convergence when adjacent graph vertices are strongly coupled [30], as they are for neuronal structures. We hope in the future to apply these advanced sampling techniques to the analysis of spine geometry in real neuronal data.

APPENDIX
INITIALIZATION OF S

In the main text, we chose not to give full details of the initialization of our estimate of the neuronal shape S, since our maximization and sampling methods may be initialized with any algorithm that extracts starting shapes with the correct topology from a noisy image. At low-SNR, generating topologically faithful initializations is itself a difficult problem which this particular paper does not aim to solve in full generality. Other authors consider this difficulty; for example, [8] presents a method employing adaptive thresholding which efficiently determiens local dendritic morphology. Furthermore, better initializations for S can reduce computation time and increase the accuracy of image recovery; thus, for completeness, we describe in this Appendix the method we used to extract an initial S from the data shown in this paper (Fig. 8).

To obtain the "true" neuronal shape shape analyzed here, we first thresholded the high-SNR empirical data shown in the top left of Fig. 1, then extracted the largest connected shape, filled in the resulting small observed holes, and smoothed the edges. To extract largest connected components, we first use Matlab's bwlabel.m function to determine connectivity, and then regionprops.m to label connected areas; to fill in holes, we used Matlab's imfill.m function. After using this shape to generate observed data $I_{\rm obs}$ according to our model, our procedure for initialization relies upon upon thresholding a smoothed version of this observed data. To choose the threshold value, we maximize the Poisson likelihood over a small set of candidates $S^{(k)}$. We generate candidate $S^{(k)}$ as follows: we first convolve the data, $n(x,y)$, with the psf to obtain a smoothed image $\eta_{\rm data}$. Next, we take a number of thresholds of $\eta_{\rm data}$, i.e., $\eta_{\rm data} > \gamma * {\rm mean}(\eta_{\rm data})$ where we vary γ. Each threshold produces a binary image (with all pixels greater than the threshold set to one). We then extract the largest simply connected component, $S_{\rm threshold}$. We choose the $S_{\rm threshold}$ maximizing the Poisson loglikelihood according to our image degradation model, $\log p(I_{\rm obs}\,|\,S)$, as our initial neuronal shape.

If the observed pixel counts are particularly sparse, and the neuron is particularly thin, this simple procedure can choose a poor starting point for our optimization. Sometimes this even leads to an intial guess with an incorrectly truncated neuron (Fig. 9, top right). As the average number of observed photons falling within an area of the neuron the size of the point-spread function drops to zero, the optimization easily becomes stuck in a local minima: the iterative algorithm cannot "see" enough data to flip the pixels that would move towards a globally optimal solution and, therefore, cannot recover from a faulty truncation. In this very low-SNR case, we have found that a coarse-to-fine strategy is effective: we perform the above thresholding procedure after smoothing with an unphysically wide psf. This leads to much more reasonable initializations for the algorithm in these very low-intensity cases (Fig. 9, bottom right).

ACKNOWLEDGMENT

The authors would like to thank R. Araya and R. Yuste for providing the example neural image used here and for helpful conversations about fluorescence imaging.

REFERENCES

[1] F. Engert and T. Bonhoeffer, "Dendritic spine changes associated with hippocampal long-term synaptic plasticity," *Nature*, vol. 399, pp. 66–70, 1999.

[2] R. Araya, J. Jiang, K. B. Eisenthal, and R. Yuste, "The spine neck filters membrane potentials," *PNAS*, vol. 103, no. 47, pp. 17961–17966, 2006.

[3] E. A. Nimchinsky, R. Yasuda, T. G. Oertner, and K. Svoboda, "The number of glutamate receptors opened by synaptic stimulation in single hippocampal spines," *J. Neurosci.*, vol. 24, pp. 2054–2064, Feb. 2004.

[4] A. Rodriguez, D. B. Ehlenberger, D. L. Dickstein, P. R. Hof, and S. L. Wearne, "Automated three-dimensional detection and shape classification of dendritic spines from fluorescence microscopy images," *PLoS ONE*, vol. 3, p. e1997, Apr. 2008.

[5] I. Y. Y. Koh, W. B. Lindquist, K. Zito, E. A. Nimchinsky, and K. Svoboda, "An image analysis algorithm for dendritic spines," *Neural Comput.*, vol. 14, no. 6, pp. 1283–1310, 2002.

[6] S. Farivar, "Cytoarchitecture of the Locust Olfactory System," Ph.D. dissertation, Caltech, 2005.

[7] X. Xu and S. Wong, "Optical microscopic image processing of dendritic spines morphology," *IEEE Signal Process. Mag.*, vol. 23, no. 7, pp. 132–135, Jul. 2006.

[8] J. Cheng, X. Zhou, E. Miller, R. Witt, J. Zhu, B. Sabatini, and S. Wong, "A novel computational approach for automatic dendrite spines detection in two-photon laser scan microscopy," *J. Neurosci. Meth.*, vol. 165, pp. 122–134, 2007.

[9] W. Denk, K. Delaney, A. Gelperin, D. Kleinfeld, B. Strowbridge, D. Tank, and R. Yuste, "Anatomical and functional imaging of neurons using 2-photon laser scanning microscopy," *Neurosci. Meth.*, vol. 54, no. 2, pp. 151–162, 1994.

[10] G. van Kempen, L. van Vliet, P. Verveer, and H. van der Voort, "A quantitative comparison of image restoration methods for confocal microscopy," *J. Microscopy*, vol. 185, no. 3, pp. 354–365, 1997.

[11] J. Conchello and J. McNally, "Fast regularization technique for expectation maximization algorithm for optical sectioning microscopy," *SPIE Proc.*, vol. 2655, pp. 199–208, 1996.

[12] J. Markham and J. Conchello, "Fast maximum-likelihood image-restoration algorithms for three-dimensional fluorescence microscopy," *J. Opt. Soc. Amer. A*, vol. 18, pp. 1062–1071, May 2001.

[13] W. H. Richardson, "Bayesian-based iterative method of image restoration," *J. Opt. Soc. Amer. A*, vol. 62, no. 1, pp. 55–59, 1972.

[14] L. B. Lucy, "An iterative technique for the rectification of observed distributions," *Astrophys. J.*, vol. 79, pp. 745–753, June 1974.

[15] R. Molina, "On the hierarchical Bayesian approach to image restoration: Applications to astronomical images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 11, pp. 1122–1128, Nov. 1994.

[16] Y. Vardi, L. A. Shepp, and L. Kaufman, "A statistical model for positron emission tomography," *J. Amer. Statist. Assoc.*, vol. 80, no. 389, pp. 8–20, 1985.

[17] G. Kontaxakis, L. Strauss, and G. Tzanakos, "An efficient implementation of the iterative MLEM image reconstruction algorithm for PET on a pentium PC platform," *J. Comput. Inf. Technol.*, vol. 7, no. 2, pp. 153–163, 1999.

[18] A. D. Simoni, F. Fernandes, and F. A. Edwards, "Spines and dendrites cannot be assumed to distribute dye evenly," *Trends Neurosci.*, vol. 27, pp. 15–16, 2004.

[19] K. Svoboda, "Do spines and dendrites distribute dye evenly?," *Trends Neurosci.*, vol. 27, pp. 445–446, 2004.

[20] T. Kong and A. Rosenfeld, "Digital topology: Introduction and survey," *Comput. Vis. Graph. Image Process.*, vol. 48, no. 3, pp. 357–393, 1989.
[21] C. Robert and G. Casella, *Monte Carlo Statistical Methods*. New York: Springer, 2005.
[22] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 11, pp. 721–741, Nov. 1984.
[23] I. J. Good and R. A. Gaskins, "Nonparametric roughness penalties for probability densities," *Biometrika*, vol. 58, no. 2, pp. 255–277, 1971.
[24] M. Miller and B. Roysam, "Bayesian image reconstruction for emission tomography incorporating Good's roughness prior on massively parallel processors," *Proc. Nat. Acad. Sci.*, vol. 88, pp. 3223–3227, 1991.
[25] P. J. Verveer and T. M. Jovin, "Image restoration based on Good's roughness penalty with application to fluorescence microscopy," *J. Opt. Soc. Amer. A*, vol. 15, pp. 1077–1083, 1998.
[26] S. Joshi and M. I. Miller, "Maximum a posteriori estimation with Good's roughness for optical sectioning microscopy," *J. Opt. Soc. Amer. A*, vol. 10, pp. 1078–1085, May 1993.
[27] G. Ascoli, J. Krichmar, S. Nasuto, and S. Senft, "Generation, description and storage of dendritic morphology data," *Philosoph. Trans.: Biol. Sci.*, vol. 356, pp. 1131–1145, 2001.
[28] C. Lohou and G. Bertrand, "A 3d 12-subiteration thinning algorithm based on p-simple points," *Discrete Appl. Math.*, vol. 139, no. 1–3, pp. 171–195, 2004.
[29] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.
[30] A. Barbu and S. C. Zhu, "Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1239–1253, Aug. 2006.

Geoffrey Fudenberg recieved the B.A. degree in physics from Columbia University, New York, in 2008. He is currently a graduate student in biophysics at Harvard University, Cambrige, MA.

Liam Paninski received the Ph.D. degree in neural science from New York University in 2003.

He is currently an Associate Professor in the Department of Statistics, Columbia University, New York.