Yale Senior Thesis 2020

S&DS 492

Applications of Bayesian Inference for the Parameterization of Mixture Models of Neuronal Dendrite Morphogenesis

Daniel Fridman
Advisor: Prof. Jonathan Howard
DUS: Prof. Sekhar Tatikonda

April 30 2020

1 Introduction

1.1 Neuronal Dendrite Morphogenesis

Neurons are extraordinarily complex and sophisticated biological systems whose morphological structure and dynamics allow them to efficiently process signals and form the circuitry of the brain. Dendrites, which branch out of the neuron's cell body, play a crucial role in receiving and integrating input signals from neighboring neurons. A neuron's specific dendrite morphology and patterning plays an important role in determining which signals the neuron receives and how it processes them. Understanding dendrite morphology and dynamics as well as the underlying mechanisms driving dendritic development has important implications for elucidating normal neural and brain development. Furthermore, identifying defects in dendrite development may enhance our understanding of the cellular basis of various neurological and neurodevelopmental disorders such as schizophrenia, Down's syndrome, and autism.

Over the past several decades, studies on *Drosophila melanogaster* (fruit fly) neurons have revealed a broad range of genetic, molecular, and biophysical mechanisms contributing to dendrite morphogenesis (1). As a result of these mechanisms, different neurons develop distinct dendrite morphologies including different dendrite sizes, branching patterns, and area coverage (dendritic field). These structural differences allow certain neurons to carry out distinct physiological functions within the neural circuitry of the brain. In particular, four distinct classes of dendritic arborization neurons have been identified in *D. melanogaster* (1).

A variety of physiological requirements determine the sizes and shapes of dendrites in the four distinct neuron classes. These include the interplay between the metabolic costs of dendrite growth and the necessity of covering a specific area (receptive field) required for properly integrating its sensory or synaptic inputs. Additionally, dendrite branching patterns must not only cover the area of its inputs, but also be capable of sampling and processing them. Furthermore, dendrites must be sufficiently dynamic such that they can adjust in response to experience and changes in their biological environments.

Genetic, molecular, and biophysical factors control the mechanisms of dendrite morphogenesis and help neurons fulfill their physiological requirements. Transcription factors allow for expression of genes dictating dendrite size and complexity, branching patterns, and other morphological features. Cell surface receptors play similar roles in controlling dendrite development. Importantly, the cytoskeleton plays a crucial role in controlling dendrite dynamics through the assembly of actin and microtubules which form the physical structural basis for dendrite growth, branching, and dynamics. All together, these factors influence the the connections that neurons make and how they process signals, playing a significant role in neurological development, health, and disease.

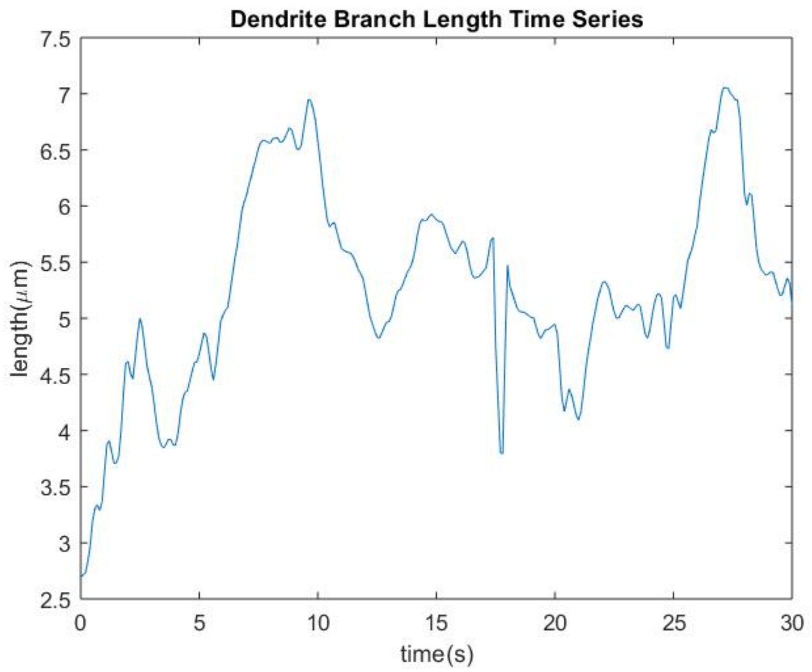


Figure 1: Dendrite branch time series for a single track showing length measurements in μm recorded every 0.1 s for 30 s. Experiment was repeated for 441 tracks. Experiments were performed using confocal microscopy for *D. melanogaster* neurons.

1.2 Modelling Dendrite Branch Dynamics

Studies in the Jonathan Howard lab have focused on dendrite dynamics and branching processes in Class IV dendritic arborization neurons of *D. melanogaster*. Using confocal microscopy, the Howard lab tracked the spatial and temporal dynamics of dendrite tips, recording a time series of tip lengths (Fig. 1). Each time series consists of a track of a single dendrite tip length for 30 seconds with 441 total tracks recorded.

From this data, all branch lengths across the 441 tracks were concatenated and the corresponding dendrite branch velocities were computed. A histogram of the raw velocity data is shown below (Fig. 2).

The Howard lab hypothesizes that class IV neuronal dendrites display 3

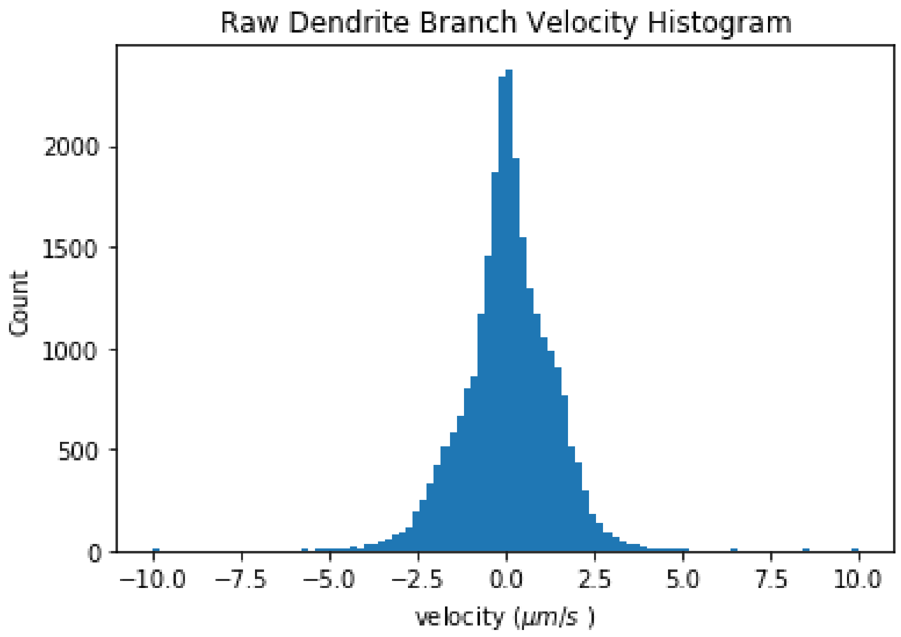


Figure 2: Raw dendrite branch velocity histogram

distinct states throughout their dynamic branching processes - growth, paused, and shrinking states. Furthermore, they hypothesize that the velocities of each state can be represented according to a unique velocity distribution which they hypothesize to be Gaussian for the paused state (with small positive and negative velocities near zero), log-Normal for the growing state (with only positive velocities), and negative (reversed) log-Normal for the shrinking state (with only negative velocities). As such, the dendrite branch velocity data can be modelled as a three-state log-N-Gauss-log-N mixture model with unique mean, variance, and weight parameters (eq. 1).

$$
\begin{aligned}
y_i \sim w_1 \frac{1}{(y>0)\sigma_1\sqrt{2\pi}} exp(-\frac{(ln((y>0)) - \mu_1)^2}{2\sigma_1^2}) + \\
w_2 \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y-\mu_2)^2}{2\sigma_2^2}) + \\
w_3 \frac{1}{|(y<0)|\sigma_3\sqrt{2\pi}} exp(-\frac{(ln(|(y<0)|) - \mu_3)^2}{2\sigma_3^2})
\end{aligned}
\tag{1}
$$

1.3 Applying Bayesian Inference for Model Parameterization

In this paper, I seek to determine the unknown μ, σ, and w parameters for the proposed log-N-Gauss-log-N mixture model. In order to accomplish this, I seek to apply Bayesian inference methods. In recent years, Bayesian inference has gained popularity for model parameterization as it provides a rigorous method for parameter inference. Through the application of Bayes rule, Bayesian inference allows for calculating posterior distributions for model parameters that can be updated with new data. Furthermore, in cases where models are too complex to analytically calculate posterior distributions, Markov Chain Monte Carlo (MCMC) methods have allowed for estimating posterior distributions by iteratively sampling from them. One such MCMC method is Gibbs sampling, which will be discussed in detail below. This paper seeks to apply the Gibbs sampling algorithm in order to determine parameter distributions for the unknown μ, σ, and w parameters and to assign each velocity measurement in our dataset as coming from a growing, paused, or shrinking state with a corresponding probability. The results of this model parameterization will allow for the assessment of our model's fit to the data and provide further insight into the dynamics of dendrite morphogenesis.

2 Background on Gibbs Sampling

In this section I will provide the necessary background on Gibbs sampling leading up to my implementation of the Gibbs sampling algorithm for paramaterizing a log-N-Gauss-log-N mixture model. I will begin with an overview of Bayesian inference, discuss conjugate priors, and conclude with a generalized description of the Gibbs sampling algorithm.

2.1 Bayesian Inference

In many diverse fields, scientists often use predictive models to explain and better understand complex natural processes. While certain models are deterministic, natural processes tend to have significant amounts of variation and noise and lend themselves to statistical models. While it may be relatively straightforward to state the probability of observing a certain set of data given that we know the model that generated it, it is significantly more difficult to state the probability of a hypothesized model's parameters given that we observe a certain set of data. Since the more common goal in research is to derive a model that adequately explains experimentally measurable or observable data, we are often faced with the latter task of estimating model parameters from the data. This task is known as statistical inference (2).

While traditionally, frequentist-based inference and maximum likelihood estimation (MLE) have been used to estimate model parameters, Bayesian inference has gained increasing popularity as a rigorous method for not only determining parameter point estimates, but also providing posterior distributions of the parameter space. The basis of Bayesian inference is the Bayes' rule. If we have a hypothesized model with parameters θ and observed or measured data D, we are able to use Bayes' rule to make the following inversion: $p(D|\theta) \longrightarrow p(\theta|D)$, using the following equation (2):

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} \tag{2}$$

where $p(D|\theta)$ is known as the *likelihood*, $p(\theta)$ is known as the *prior*, and $p(\theta|D)$ is known as the *posterior*. The *likelihood* represents the probability of generating a certain sample of data D, given that we know the model that generated our data and that our model's parameters equal θ. The *prior* represents an initial assumption about the distribution of our parameter space before seeing the data. The denominator on the right-hand side is known as the *marginal likelihood* and represents the probability of obtaining a certain set of data, assuming we have a defined likelihood and a prior. Finally, and most importantly, the *posterior* is the end goal of Bayesian inference and represents our updated distribution across the parameter space after seeing the data (2).

While Bayes' rule may often be applied to obtain an exact analytical solution of the posterior for simple models, such an analytical solution may not be achievable for more complex models with many parameters. Part of the issue stems from the fact that the *marginal likelihood* $p(D)$ requires an integral across the parameter space: $p(D) = \int_\theta p(D|\theta)p(\theta)d\theta$. However, for n model parameters, this quickly becomes an intractable high-dimensional integral: $p(D) = \int_{\theta_1} \cdots \int_{\theta_n} p(D|\theta_1, \ldots, \theta_n)p(\theta_1, \ldots, \theta_n)d\theta_1 \ldots d\theta_n$ (2).

Upon closer observation, however, it is apparent that $p(D)$ is independent of our parameters θ and thus serves as a normalizing factor for the numerator $p(D|\theta)p(D)$ in order to make the posterior $p(\theta|D)$ a valid probability distribution. Thus, we can avoid calculating the denominator by establishing the following proportionality (2):

$$p(\theta|D) \propto p(D|\theta)p(\theta) \tag{3}$$

2.2 Conjugate Priors

In certain special cases, an exact closed-form solution for the posterior can be calculated without having to calculate the integral for the marginal posterior. This can be achieved through a mathematically convenient choice for a prior which has nice properties when combined with the likelihood. More specifically, if a prior is chosen from a specified family of distributions such that the posterior will fall into the same family of distributions, it may be possible to obtain a closed-form solution for the posterior. These 'mathematically convenient' priors are known as *conjugate priors* (2).

In order to explain how conjugate priors work, it is easiest to use an example. Thus, I will use a biophysically relevant example relating to ion channel patch-clamp recordings in order to demonstrate the use of conjugate priors (4,5).

2.2.1 Example: Ion Channel Patch-Clamp Recording (3,4,5)

Most cells, including neurons, contain proteins called ion channels on their membranes which allow for ions to flow between the interior and exterior of the cell. These ion channels regulate the concentration of ions across the membrane by transitioning between open and closed states. It can be assumed that these transitions occur stochastically, according to Markovian dynamics. Furthermore, it can be assumed that the time an ion channel spends in any given state follows an exponential distribution, with short dwell-times being more likely than long dwell-times. An experiment can be carried out which tracks the time spent in each state and a histogram of dwell-times can be polotted, a simulation of which is shown in Fig. 3.

We first model the dwell-times as random variables from an exponential distribution, $y_i \sim \lambda e^{-\lambda y}$. For N samples, we thus form an exponential likeihood: $\prod_{i=1}^{N} \lambda e^{(-\lambda y_i)}$. Next, we seek to determine the time-scale parameter λ of our

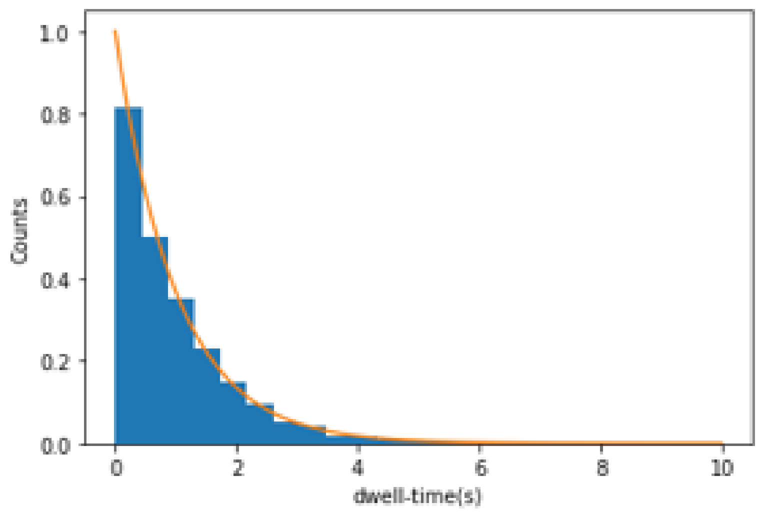


Figure 3: A simulation showing the dwell time of the ion channel in any given state distributed exponentially

model based on our data. We can formulate this problem in terms of Bayesian inference as follows:

$$p(\lambda|y) \propto (\prod_{i=1}^{N} \lambda e^{(-\lambda y_i)})p(\lambda) \tag{4}$$

Our goal is to select an appropriate prior, $p(\lambda)$, such that we can obtain a closed form posterior for the time-scale parameter, $p(\lambda|y)$. The conjugate prior to an exponential likelihood is the Gamma distribution:

$$p(\lambda) \sim Gamma(\lambda|\alpha, \beta) = \frac{\beta^{\alpha}}{\Gamma(a)} \lambda^{\alpha-1} e^{-\lambda b} \tag{5}$$

With the following set of steps we can see how the Gamma prior conveniently combines with the exponential likelihood:

$$
\begin{aligned}
p(\lambda|y) &\propto (\prod_{i=1}^{N} \lambda e^{(-\lambda y_i)}) \times \frac{\beta^\alpha}{\Gamma(a)} \lambda^{\alpha-1} e^{-\lambda\beta} \\
&\propto (\lambda^N e^{-\lambda \sum_{i=1}^{N} y_i}) \times (\frac{\beta^\alpha}{\Gamma(a)} \lambda^{\alpha-1} e^{-\lambda\beta}) \\
&\propto (\lambda^N e^{-\lambda \sum_{i=1}^{N} y_i})(\lambda^{\alpha-1} e^{-\lambda\beta}) \\
&= \lambda^{\alpha+N-1} e^{-\lambda(\sum_{i=1}^{N} y_i+\beta)}
\end{aligned} \tag{6}
$$

We observe that the simplified solution above follows the same form as the Gamma distribution, but with new hyperparameters, updated according to our data. Thus, we obtain the final closed-form solution for our posterior:

$$
p(\lambda|y) \sim \Gamma(\lambda|\alpha', \beta') \text{ s.t. } \alpha' = \alpha + N \text{ and } \beta' = \sum_{i=1}^{N} y_i + \beta \tag{7}
$$

Thus, using the idea of conjugate priors, we are able to solve for the posterior distribution of the time-scale parameter for our exponential model, dependent on our data of ion channel dwell-times.

2.3 Gibbs Sampling Overview

While conjugate priors can be used for simple models, more complex models with many parameters may have no convenient conjugate prior. Thus, it may not be possible to obtain exact closed-form solutions for the posterior. Nonetheless, posteriors can be estimated using dependent sampling methods referred to as Markov Chain Monte Carlo (MCMC). The idea of MCMC sampling is that the posterior can be sampled from and given enough samples, an approximation to the true posterior can be obtained.

One type of MCMC algorithm is known as *Gibbs sampling.* The idea of Gibbs sampling is that while it may not be possible to obtain a closed-form solution for the multi-parameter posterior, it may be possible to obtain closed-form posteriors for single model parameters conditioned on the other parameters. Thus, each parameter can be sampled from, dependent on the other parameters and the data. Sampling for multiple iterations and updating the parameter values for every iteration, the posterior for each parameter in the original posterior can be recreated (essentially returning a cross-section of each dimension in the multi-dimensional posterior) (2).

2.3.1 Generalized Gibbs Sampling Algorithm

As a generalized example of the Gibbs sampling procedure, we can imagine that we have a model with N unknown parameters, $\Theta = (\theta_1, \theta_2, \ldots, \theta_N)$ associated with a model that we've hypothesized for our data. We also assume that we have an observed dataset, D. Our goal is to estimate the N-dimensional posterior, $p(\theta_1, \theta_2, \ldots, \theta_N|D)$. While we may be unable to obtain a closed-form solution for this posterior, we may instead be able to obtain closed form solutions for the conditional posteriors of each of the parameters:
$p(\theta_1|\theta_2, \ldots, \theta_N, D), p(\theta_2|\theta_1, \theta_3, \ldots, \theta_N, D), \ldots, p(\theta_N|\theta_1, \ldots, \theta_{N-1}, D)$.
We can then apply the Gibbs sampling algorithm to sample from each of the conditional posteriors and estimate the N-dimensional posterior according to Algorithm 1 below (5).

2.3.2 Gibbs Sampling Example for Simple Gaussian Model

As a specific example of the Gibbs sampling procedure, we will look at a Gaussian model with two unknown parameters, μ and σ. Assume that our data is generated from a Gaussian distribution, $y_i \sim \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y-\mu)^2}{2\sigma^2})$. For N samples, we thus have a Gaussian likelihood. We seek to determine a 2-dimensional posterior, $p(\mu, \sigma|y)$. However, we assume that we cannot obtain a closed-form analytical solution for this posterior. Nonetheless, using the idea of

Algorithm 1 Generalized Gibbs Sampling Algorithm

```
1: Initialize at a random starting point $(\theta_1^0, \theta_2^0, \ldots, \theta_N^0)$
2: **for** t in t_iterations **do**
3:     randomize the order of sampling from conditional posteriors
4:     sample each parameter using the most recently sampled parameter
       values, i.e. for an order of $(\theta_1, \theta_2, \ldots, \theta_N)$, update as:
5:      $\theta_1^t \sim p(\theta_1^t | \theta_2^{t-1}, \ldots, \theta_N^{t-1}, D)$
6:     $\theta_2^t \sim p(\theta_2^t | \theta_1^t, \theta_3^{t-1}, \ldots, \theta_N^{t-1}, D)$
7:     ...
8:     $\theta_N^t \sim p(\theta_N^t | \theta_1^t, \ldots, \theta_N^t, D)$
9:     append $(\theta_1^t, \ldots, \theta_N^t)$ to $[\theta_1\_vals, \ldots, \theta_N\_vals]$
10: **end for**
11: **return** $[\theta_1\_vals, \ldots, \theta_N\_vals]$
```

conjugate priors, we can determine the closed-form solutions for both the μ and σ parameters conditioned on the other parameter and our data.

It has been shown that the following priors are conjugate to the Gaussian likelihood (6):

$$\begin{aligned} \mu|\tau &\sim N(\mu_0, n_0\tau) \\ \tau &\sim Gamma(\alpha, \beta) \end{aligned} \tag{8}$$

where $\tau = \frac{1}{\sigma^2}$. And the corresponding posteriors can be derived from the priors above (6):

$$\begin{aligned} \tau|y &\sim Gamma\left(\alpha + n/2, \beta + \frac{1}{2}\sum(y_i - \bar{y})^2 + \frac{nn_0}{2(n+n_0)}(\bar{y} - \mu_0)^2\right) \\ \mu|\tau, y &\sim N\left(\frac{n\tau}{n\tau + n_0\tau}\bar{y} + \frac{n_0\tau}{n\tau + n_0\tau}\mu_0, n\tau + n_0\tau\right) \end{aligned} \tag{9}$$

We can thus determine the posterior $p(\mu, \sigma|y)$ by using Gibbs sampling to iteratively sample from the μ and σ conditional posteriors, respectively, and updating our parameter values as described in algorithm 2 below:

Algorithm 2 Gibbs Sampling Algorithm for Simple Gaussian Model

```
1: Initialize at a random starting point $(\mu^0, \sigma^0)$
2: for t in t_iterations do
3:     sample each parameter using the most recently sampled parameter values, update as:
4:     $\tau^t|y \sim \Gamma\left(\alpha + n/2, \beta + \frac{1}{2}\sum(y_i - \bar{y})^2 + \frac{nn_0}{2(n+n_0)}(\bar{y} - \mu_0)^2\right)$
5:     $\mu^t|\tau^t, y \sim N\left(\frac{n\tau^t}{n\tau^t + n_0\tau^t}\bar{y} + \frac{n_0\tau^t}{n\tau^t + n_0\tau^t}\mu_0, n\tau^t + n_0\tau^t\right)$
6:     append $(\mu^t, \sqrt{\frac{1}{\tau^t}})$ to $[\mu\_vals, \sigma\_vals]$
7: end for
8: return $[\mu\_vals, \sigma\_vals]$
```

3 Implementation and Application to Dendrite Morphogenesis

In this section I will describe the implementation of the Gibbs sampling algorithm for the 3-component log-N-Gauss-log-N mixture model used to model dendrite branch velocity distributions. I will first discuss the methods employed in applying Gibbs sampling to mixture models and then discuss the specifics of my implementation.

3.1 Gibbs sampling for mixture models (5)

Mixture models contain multiple component distributions and thus require parameters to be sampled for each component in order to estimate the posterior. In order to accomplish this, a trick known as *data augmentation* is used which adds a new latent indicator variable to the data to label which component each data point was likely drawn from. For a k-component mixture model, we would have k potential categories for each indicator variable: $cat_i \in (1, 2, \ldots, k)$. Additionally, we assume that in total our mixture model contains (D+k) parameters representing D parameters from all the components of the model and k weight parameters associated with each of the k components. With the inclusion of latent variables, the posterior (originally with D+k parameters) now contains N additional parameters indicating the category of each data point: $p(\theta_1, \ldots, \theta_D, w_1, \ldots, w_k, cat_1, \ldots, cat_N|y)$. These latent variables will be marginalized out in the process of Gibbs sampling, but are included to simplify the sampling procedure.

After including the latent indicator variables, the following conditional posteriors need to be computed in order to apply the Gibbs sampling procedure:

$$\begin{aligned} p(\theta_1|\ldots) \propto p(y|\ldots)p(\theta_1), \ldots, p(\theta_N|\ldots) \propto p(y|\ldots)p(\theta_N) \\ p(w_1|\ldots) \propto p(y|\ldots)p(w_1), \ldots, p(w_k|\ldots) \propto p(y|\ldots)p(w_k) \\ p(cat_1|\ldots) \propto p(y|\ldots)p(cat_1), \ldots, p(cat_N|\ldots) \propto p(y|\ldots)p(cat_N) \end{aligned} \tag{10}$$

The way this can be achieved is by using the idea of conjugate priors to find an appropriate prior to each of the likelihoods and thus obtain a closed-form conditional posterior for each parameter. Then, the conditional posteriors for each of the parameters can be sampled from and updated iteratively.

The posterior $p(\theta_i|...)$ can be computed using the conjugate prior to the likelihood of whichever distribution our k-th component of the model comes from. For example, if one of our model components comes from an exponential distribution, we would use a Gamma prior and its corresponding posterior as shown in section 2.2.1. Likewise, if one of our model components comes from a Gaussian distribution, we would use a $N - \Gamma^{-1}$ prior and its corresponding posterior as shown in section 2.3.2. The posterior for the k-th component, however, would be conditioned on the data assigned to the k-th component rather than the full dataset.

Next, in order to assign each data point to one of k components, we need to sample cat_i from k components with probability equal to the posterior probability of cat_i coming from each of k components, $p(cat_i = 1|\ldots), \ldots, p(cat_i = k|\ldots)$. This posterior probability can be expressed as follows:

$$\begin{aligned} p(cat_i = j|\ldots) &\propto p(y_i|cat_i = j, \ldots)p(cat_i = j) \\ &\propto p(y_i|cat_i = j, \ldots) * w_j \end{aligned} \tag{11}$$

As shown above, the posterior probability that data point i is assigned to category j is proportional to the likelihood of data point i being drawn from the j-th model component times the weight of the j-th component.

Each data point in the dataset is then assigned to one of k possible categories according to a categorical distribution with corresponding probabilities:

$$cat_i \sim Categorical(cat_i|p_1, \ldots, p_k) \tag{12}$$

where $p_1 = p(cat_i = 1|\ldots), \ldots, p_k = p(cat_i = k|\ldots)$. The categorical distribution is an extension of the Bernoulli distribution to k dimensions and can be thought of as doing a k-dimensional coin flip with corresponding probabilities.

The final parameters for which we need to determine a conditional posterior are the weight parameters w for each of the k model components. It's important to realize that the weight w_j essentially represents the probability of sampling from the j-th component and thus (in order to ensure a valid probability distribution) the weights in the mixture model need to sum to 1, $w_1 + w_2 + \ldots + w_k = 1$.

Using the conjugacy between a categorical likelihood and the Dirichlet prior, we can obtain a closed form for the joint posterior for all k weight parameters as follows:

$$\begin{aligned} p(w_1, \ldots, w_k | \ldots) &\propto L(cat_i | \ldots) p(w_1, \ldots, w_k) \\ &\propto L(cat_i | \ldots) * Dir(w_1, \ldots, w_k | \alpha_1, \ldots, \alpha_k) \\ &= L(cat_i | \ldots) * \frac{\Gamma(\sum_{j=1}^{k} \alpha_j)}{\prod_{j=1}^{k} \Gamma(\alpha_j)} \prod_{j=1}^{k} w_j^{\alpha_j - 1} \\ &\propto Dir(w_1, \ldots, w_k | n(cat_1) + \alpha_1, \ldots, n(cat_k) + \alpha_k) \end{aligned} \tag{13}$$

where $n(cat_j)$ represents the number of elements assigned to category j.

With the steps above, we have derived the conditional posteriors for all of our model parameters and can now apply the Gibbs sampling algorithm to estimate the posterior of any mixture model whose likelihoods of its individual components have conjugate priors (i.e. for which $p(\theta_i | \ldots)$ can be solved).

In the following section we will apply the steps shown in section 3.1 as well as the posterior for a Gaussian likelihood stated in section 2.3.2 to implement the Gibbs sampling algorithm for a 3-component log-N-Gauss-log-N mixture model.

3.2 Gibbs Sampling for 3-component log-N-Gaussian-log-N Mixture Model

3.2.1 Defining our Mixture Model

As stated in section 1.2, we hypothesize that dendrite branches display growing, paused, and shrinking states. As paused state velocities may take on small positive or small negative values near zero, we hypothesize that the paused state follows a Gaussian velocity distribution. Furthermore, since it only makes sense for a growing state to contain positive velocity values, we model the growing state as a log-Normal distribution as the log-Normal distribution may not take on negative values. Likewise, since a shrinking state should only contain negative velocity values, we model the shrinking state as a negative (reflected) log-Normal distribution.

Compiling our hypotheses about the growing, paused, and shrinking states of dendrite branches, we model dendrite branch velocity distributions as a 3-component log-N-Gaussian-log-N mixture model containing 9 unknown mean, variance, and weight parameters that we seek to determine (i.e. $\mu_{growing}$, μ_{paused}, $\mu_{shrinking}$; $\sigma_{growing}$, σ_{paused}, $\sigma_{shrinking}$; and $w_{growing}$, w_{paused}, $w_{shrinking}$).

As shown in section 1.2, the model can be represented as follows:

$$\begin{aligned} y_i \sim w_1 \frac{1}{(y>0)\sigma_1\sqrt{2\pi}} exp(-\frac{(ln((y>0)) - \mu_1)^2}{2\sigma_1^2})+ \\ w_2 \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y-\mu_2)^2}{2\sigma_2^2})+ \\ w_3 \frac{1}{|(y<0)|\sigma_3\sqrt{2\pi}} exp(-\frac{(ln(|(y<0)|) - \mu_3)^2}{2\sigma_3^2}) \end{aligned} \tag{14}$$

3.2.2 Deriving Conditional Posterior Distributions

In this section I will derive the conditional posterior parameter distributions for the μ and σ parameters of the 3-component log-N-Gaussian-log-N mixture model used to model the velocity distribution of neuronal dendrite branches. The conditional posteriors can then be sampled from using a Gibbs sampling procedure in order to estimate the true posterior parameter distributions for the 9 unknown parameters.

It is first important to note that any data distributed according to a log-Normal or negative log-Normal distribution can be transformed into a Gaussian distribution through a log transformation:

$$\begin{aligned} y \sim \text{log-Normal}(\mu, \sigma^2) \rightarrow ln(y) \sim \mathcal{N}(\mu, \sigma^2) \\ y \sim \text{Negative log-Normal}(\mu, \sigma^2) \rightarrow ln(|y|) \sim \mathcal{N}(\mu, \sigma^2) \end{aligned} \tag{15}$$

Then, assuming the data is either generated from a Gaussian distribution or can be transformed to follow a Gaussian distribution with parameters μ and τ, a Gaussian likelihood can be used for each component of the mixture model as follows:

$$\begin{aligned} y_{paused}|\mu_{paused},\tau_{pasued} &\sim N(\mu_{paused},\tau_{paused}) \\ ln(y_{growing})|\mu_{growing},\tau_{growing} &\sim N(\mu_{growing},\tau_{growing}) \\ ln(|y_{shrinking}|)|\mu_{shrinking},\tau_{shrinking} &\sim N(\mu_{shrinking},\tau_{shrinking}) \end{aligned} \tag{16}$$

where $\tau = \frac{1}{\sigma^2}$.

Furthermore, as shown in section 2.3.2, the following priors are conjugate to a Gaussian likelihood:

$$\begin{aligned} \mu|\tau &\sim N(\mu_0, n_0\tau) \\ \tau &\sim Gamma(\alpha,\beta) \end{aligned} \tag{17}$$

and, as also shown in section 2.3.2, the posterior takes the following form:

$$\begin{aligned} \tau|y &\sim Gamma\left(\alpha+n/2,\beta+\frac{1}{2}\sum(y_i-\bar{y})^2+\frac{nn_0}{2(n+n_0)}(\bar{y}-\mu_0)^2\right) \\ \mu|\tau,y &\sim N\left(\frac{n\tau}{n\tau+n_0\tau}\bar{y}+\frac{n_0\tau}{n\tau+n_0\tau}\mu_0, n\tau+n_0\tau\right) \end{aligned} \tag{18}$$

Given a Gaussian distributed dataset with unknown parameters μ and σ, the N-Γ^{-1} distribution can be sampled from to generate an approximation of the posterior parameter distributions according to algorithm 3:

3.2.3 Defining the Gibbs Sampling Algorithm

Compiling all the work shown in sections 3.2.1 and 3.2.2, the Gibbs sampling algorithm can be defined according to algorithms 3 and 4.

Algorithm 3 Sampling from the N-Γ^{-1} posterior

```
1: Initialization:
2: $\mu_0 = mean(\texttt{data})$
3: $n_0 = 2$
4: $\alpha = length(\texttt{data})$
5: $\beta = \frac{1}{2}\sum(\texttt{data} - \mu_0)^2$
6: $A = length(\texttt{data})$

7: for i in n_samples do
8:     $\alpha' \leftarrow \alpha + A/2$
9:     $\beta' \leftarrow \beta + \frac{1}{2}\sum(\texttt{data}_i - mean(\texttt{data}))^2 + \frac{A * n_0}{2(A+n_0)}(mean(\texttt{data}) - \mu_0)^2$
10:    $\tau|\texttt{data} \sim \Gamma(\alpha', \beta')$
11:    $\sigma^2_{posterior} \leftarrow \frac{1}{\tau}$
12:    append $\sigma^2_{posterior}$ to $\sigma\_vals$ list

13:    $\mu \leftarrow \frac{A}{A+n_0}\bar{x} + \frac{n_0}{A+n_0}\mu_0$
14:    $\sigma^2 \leftarrow \frac{1}{A\tau + n_0\tau}$
15:    $\mu_{posterior}|\tau, \texttt{data} \sim \mathcal{N}(\mu, \sigma^2)$
16:    append $\mu_{posterior}$ to $\mu\_vals$ list
17: end for
18: return $\mu\_vals$, $\sigma\_vals$
```

4 Results

In order to paramaterize the dataset of dendrite branch velocities (Fig. 2) using the 3-component log-N-Gauss-log-N mixture model (eq. 14), the Gibbs sampling algorithm (Algorithm 4) was applied on both simulated and real datasets and the results are described below.

4.1 Effects of Gibbs Sampling Initialization on posterior predictions

In order to verify that the Gibbs sampling algorithm successfully converged to the true posteriors, we tested the algorithm's performance on a simulated dataset with known parameter values. A dataset was simulated according to the 3-component log-N-Gauss-log-N mixture model with true μ, σ, and w parameters set to parameter values previously determined by the Howard lab

Algorithm 4 Gibbs Sampling for log-N-Gauss-log-N Mixture Model

```
1: for iteration t = $1, 2, \ldots$,N do
2:     if t=1 then
3:         Use Otsu initialization, ...
4:         Compute k1 and k2 thresholds to threshold 3 categories
5:         Growing_Set $\leftarrow data > k2$
6:         Shrinking_Set $\leftarrow data < k1$
7:         Paused_Set $\leftarrow k1 < data < k2$
8:     else
9:         Compute probabilities for growing, shrinking, or paused categories
           based on $\mu$, $\sigma$, and $w$ parameters sampled at iteration $t-1$
10:        $p1 \leftarrow w^{t-1}_{growing} * log\text{-}N(\texttt{data\_set > 0}, \mu^{t-1}_{growing}, \sigma^{t-1}_{growing})$
11:        $p2 \leftarrow w^{t-1}_{paused} * N(\texttt{data\_set}, \mu^{t-1}_{paused}, \sigma^{t-1}_{paused})$
12:        $p3 \leftarrow w^{t-1}_{shrinking} * log\text{-}N(abs(\texttt{data\_set} < 0), \mu^{t-1}_{shrinking}, \sigma^{t-1}_{shrinking})$
13:        *Fill p1 and p3 with zero's for $p(\texttt{data\_set} < 0)$ and
           $p(\texttt{data\_set} > 0)$, respectively
14:        Assign each point in data_set as coming from growing, paused, or
           shrinking category with prob. p1,p2,p3, respectively
15:        for i in length(data_set) do
16:            $cat_i \sim Categorical(p1[i], p2[i], p3[i])$
17:        end for
18:        Reassign categories based on categorical samples, ...
19:        Growing_Set $\leftarrow data[cat_i == 1]$
20:        Paused_Set $\leftarrow data[cat_i == 2]$
21:        Shrinking_Set $\leftarrow data[cat_i == 3]$
22:    end if
23:    Sample $\mu$ and $\sigma$ parameter values according to normal-inverse-gamma
       posterior (*See Algorithm 3)
24:    $[\mu^t_{grow}, \sigma^t_{grow}] \sim N\text{-}\Gamma^{-1}(log(\texttt{Growing\_Set}))$
25:    $[\mu^t_{pause}, \sigma^t_{pause}] \sim N\text{-}\Gamma^{-1}(\texttt{Paused\_Set})$
26:    $[\mu^t_{shrink}, \sigma^t_{shrink}] \sim N\text{-}\Gamma^{-1}(log(abs(\texttt{Shrinking\_Set})))$
27:    Sample weight $w$ parameter values according to Dirichlet posterior
28:    $[w^t_{grow}, w^t_{pause}, w^t_{shrink}] \sim Dirichlet(len(\texttt{Growing\_Set}) + \alpha_1,$
       $len(\texttt{Paused\_Set}) + \alpha_2, len(\texttt{Shrinking\_Set}) + \alpha_3)$
29:    *Repeat lines 8-28 for t = 2,...,N
30: end for
```

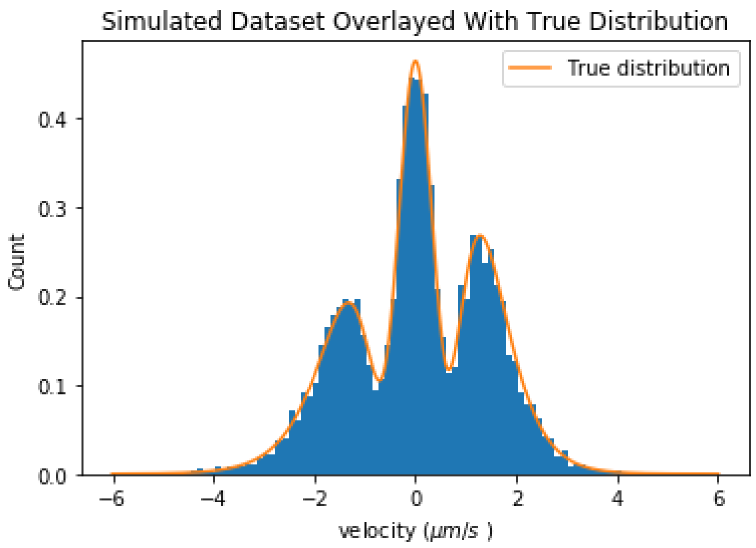


Figure 4: A simulated dataset (blue histogram) of dendrite branch velocities according to a 3-component log-N-Gauss-log-N mixture Model (shown as orange distribution). True parameter values were set to values previously obtained by the Howard lab using least squares fitting to fit a log-N-Gauss-log-N mixture model to dendrite branch velocity data. True parameter values were set as: [$true\_\mu_{growing} = 0.3873$, $true\_\mu_{paused} = 0$, $true\_\mu_{shrinking} = 0.4369$, $true\_\sigma_{growing} = 0.3624$, $true\_\sigma_{paused} = 0.3387$, $true\_\sigma_{shrinking} = 0.3918$, $true\_w_{growing} = 0.3351$, $true\_w_{paused} = 0.3939$, $true\_w_{shrinking} = 0.271$]

for the dendrite branch velocity dataset (Fig. 4).

The Gibbs sampling algorithm was initialized with random initialization, assigning each data point in the dataset to a growing, shrinking, or paused state with equal probability, with the restriction that only positive values could be assigned to a growing state and only negative values could be assigned to a shrinking state. As shown in Figure 5 below, Gibbs sampling with random initialization failed to accurately recover the true parameters. Note that only the μ posteriors are shown, but the algorithm failed to recover σ and w posteriors as well.

Upon examination of the fitted distribution using the parameter means of the posteriors recovered by Gibbs sampling (Fig. 6), it is apparent that

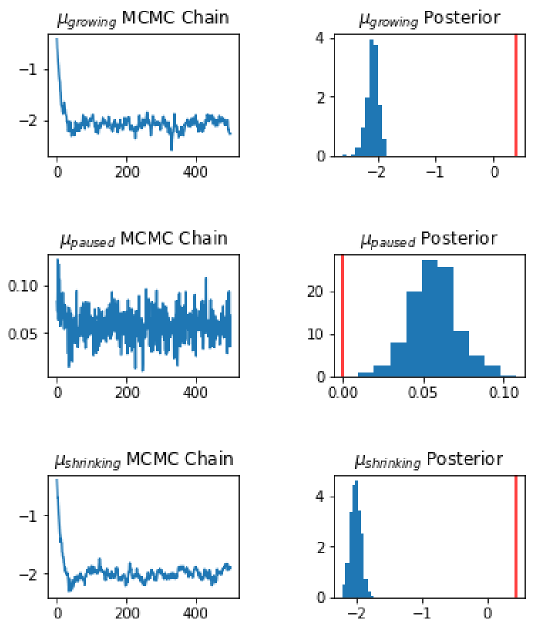


Figure 5: MCMC chain and posterior for μ parameter using simulated dataset and random initialization. Red line represents true parameter value.

random initialization assigns many large negative and large positive values to the Gaussian paused state, causing difficulties for the algorithm to converge and causing it to falsely converge to a wide Gaussian (large σ_{paused} (not shown)). Additonally, the algorithm converges to mean weights of about 0.91 for the Gaussian paused state and only about 0.046 and 0.042 for the log-Normal growing and shrinking states, respectively (posteriors not shown). Thus, it can be concluded that random initialization causes the algorithm to fit a wide Gaussian around the entire dataset, mostly disregarding the other components of the mixture model.

To address the issue of initialization, a technique called Otsu's method was employed to better initialize the categories of the data. Otsu's method is a technique used in image processing for image thresholding. The idea of Otsu's method is to maximize the inter-class variance between any multi-class dataset

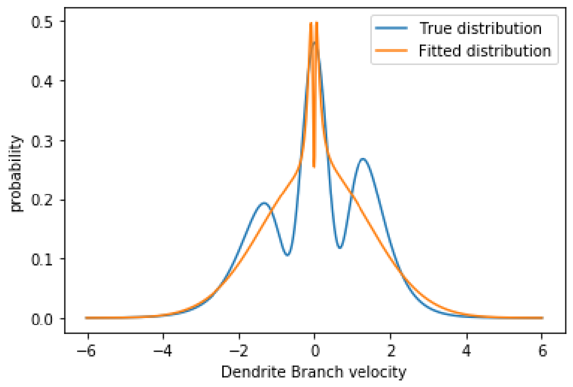


Figure 6: True model distribution (shown in blue) overlayed with the distribution obtained by Gibbs sampling (shown in orange). Parameter estimates were obtained by taking the mean of the posteriors obtained by Gibbs sampling.

(7). In our case, Otsu's method was implemented to threshold our dataset into 3 categories (Fig. 7).

The Gibbs sampling algorithm was then initialized with growing, paused, or shrinking data categorizations according to Otsu thresholding as shown in algorithm 4. In theory, this procedure would initialize the data to be much closer to the true data categories, placing the Gibbs sampler much closer to the true posteriors in posterior parameter space, allowing for easier convergence.

Running the Gibbs sampling algorithm for 500 iterations using Otsu's initialization successfully recovered the true parameters as shown in Figures 8-10.

Taking the mean of each of the parameter's posterior estimates from Gibbs sampling and plotting the fitted distribution overlayed with the true distribution shows that Gibbs sampling with Otsu initializaiton is successfully able to recover the true distribution and its parameters (Fig. 11).

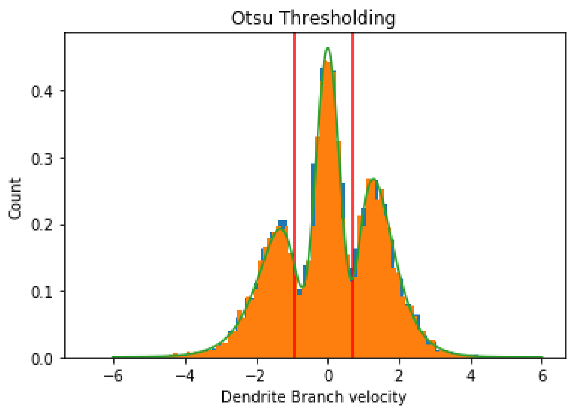
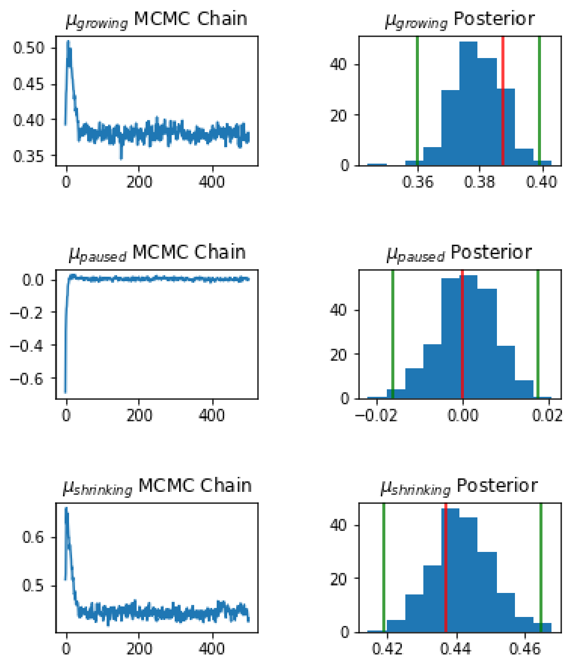
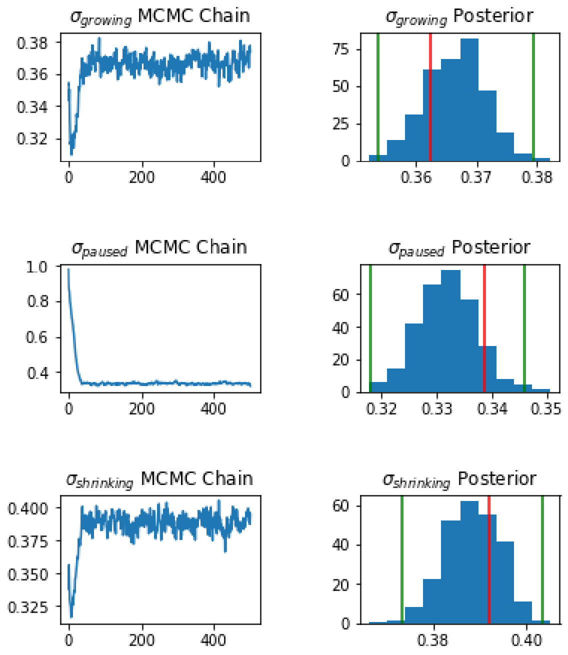


Figure 7: Simulated dataset thresholded into 3 categories using Otsu's method. Thresholds are k1 = -0.919 and k2 = 0.714



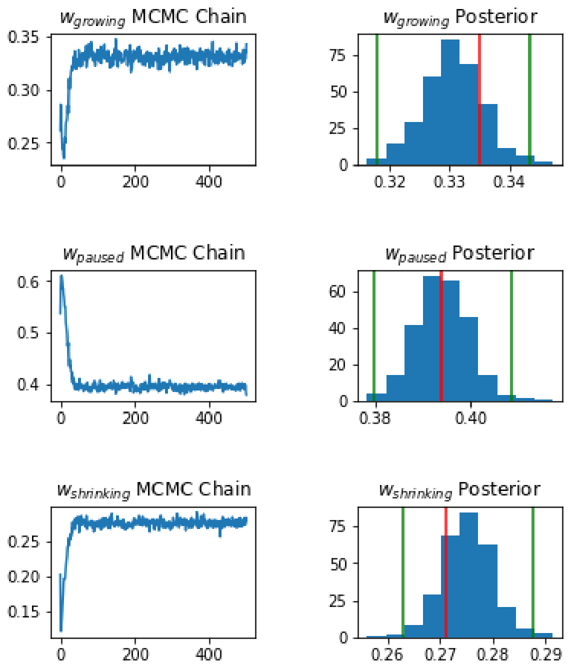
Parameter	99% conf int	std dev
$\mu_{growing}$	(0.3598, 0.3991)	0.0076
μ_{paused}	(-0.0163, 0.0178)	0.0066
$\mu_{shrinking}$	(0.419, 0.4647)	0.0088

Figure 8: MCMC chain and posterior for μ parameter using simulated dataset and Otsu initialization. Red line represents true parameter values. Green lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.



Parameter	99% conf int	std dev
$\sigma_{growing}$	(0.3537, 0.3794)	0.005
σ_{paused}	(0.3179, 0.3459)	0.0054
$\sigma_{shrinking}$	(0.3733, 0.4036)	0.0059

Figure 9: MCMC chain and posterior for σ parameter using simulated dataset and Otsu initialization. Red line represents true parameter values. Green lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.



Parameter	99% conf int	std dev
$w_{growing}$	(0.3179, 0.3434)	0.0049
w_{paused}	(0.3797, 0.4084)	0.0055
$w_{shrinking}$	(0.2629, 0.2876)	0.0048

Figure 10: MCMC chain and posterior for w parameter using simulated dataset and Otsu initialization. Red line represents true parameter values. Green lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.

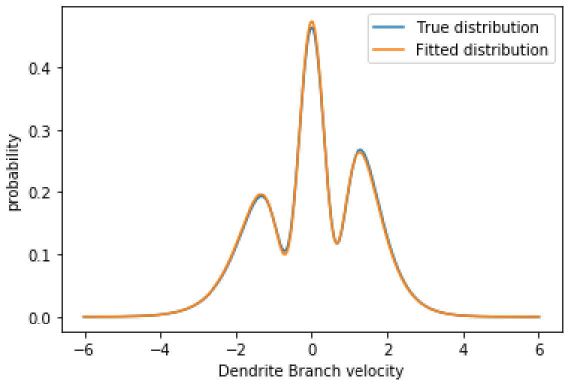
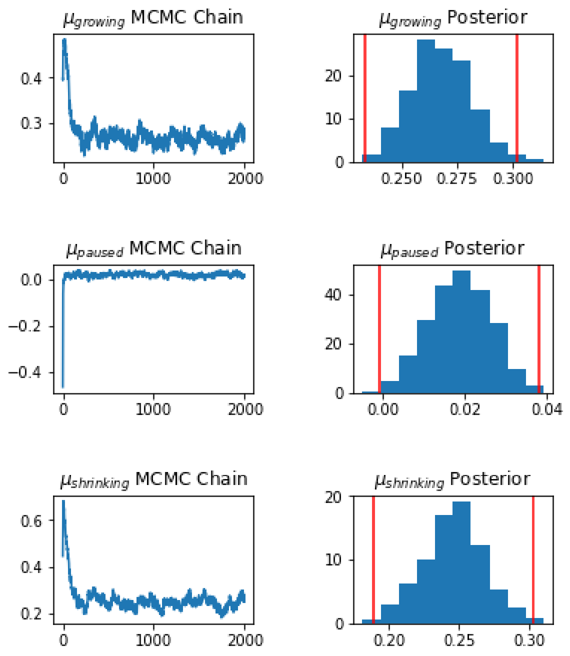


Figure 11: The true model distribution (shown in blue) overlayed with the distribution obtained by Gibbs sampling (shown in orange). Gibbs sampling with Otsu's initialization successfully recovers the true distribution.

4.2 Parameterization of experimentally obtained dendrite branch velocity distribution

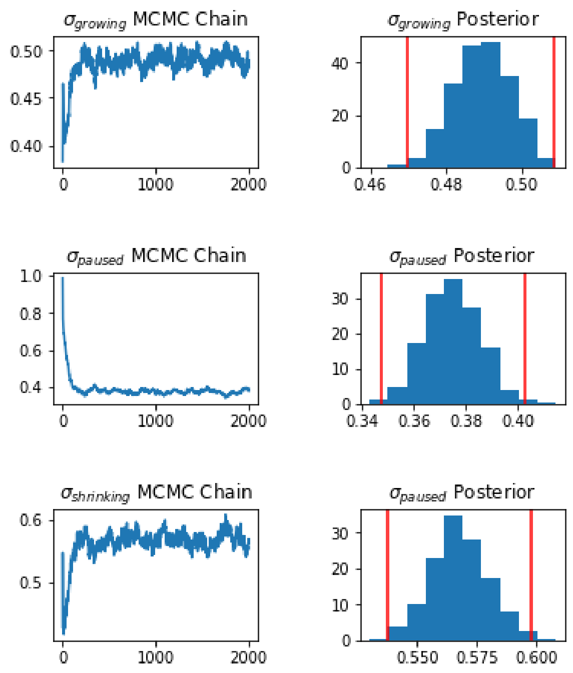
After successfully recovering the true parameters for the simulated model, I returned to my original goal of paramaterizing the experimental dataset of neuronal dendrite branch velocities (Fig. 2). As stated previously, the Howard lab hypothesizes that dendrite branch velocity distributions follow a log-N-Gauss-log-N model. Thus, I applied the Gibbs sampling algorithm (algorithm 4) with Otsu initializiation to the experimentally measured dataset and the posterior estimates with 99% confidence intervals are shown in Figures 12-14.

Taking the means of each of the posterior parameter estimates and plotting the fitted distribution based on our mixture model (eq. 14) over a histogram of the dataset shows that the model and its estimated parameters is a good fit to the data (Fig. 15). Additionally, the data segmentation into growing, paused, and shrinking states obtained by the Gibbs sampler is showin in Figure 16.



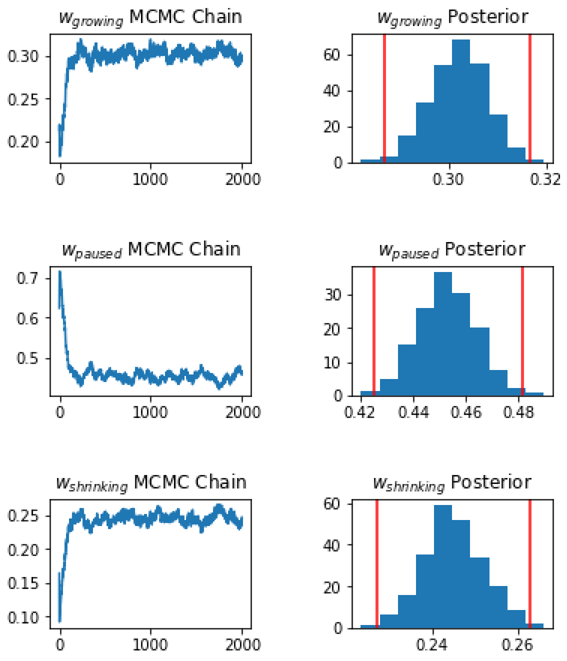
Parameter	99% conf int	std dev
$\mu_{growing}$	(0.233, 0.3021)	0.0134
μ_{paused}	(-0.0008, 0.0382)	0.0075
$\mu_{shrinking}$	(0.1889, 0.303)	0.0221

Figure 12: MCMC chain and posterior for μ parameter using experimentally measured dendrite branch velocity dataset and Otsu initialization. MCMC chain was run for 2000 iterations. Red lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.



Parameter	99% conf int	std dev
$\sigma_{growing}$	(0.4697, 0.5087)	0.0076
σ_{paused}	(0.3472, 0.4028)	0.0108
$\sigma_{shrinking}$	(0.5376, 0.5977)	0.0117

Figure 13: MCMC chain and posterior for σ parameter using experimentally measured dendrite branch velocity dataset and Otsu initialization. MCMC chain was run for 2000 iterations. Red lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.



Parameter	99% conf int	std dev
$w_{growing}$	(0.2866, 0.3167)	0.0058
w_{paused}	(0.425, 0.4816)	0.011
$w_{shrinking}$	(0.2271, 0.263)	0.007

Figure 14: MCMC chain and posterior for w parameter using experimentally measured dendrite branch velocity dataset and Otsu initialization. MCMC chain was run for 2000 iterations. Red lines represent 99% confidence intervals. Confidence intervals and standard deviations shown in table.

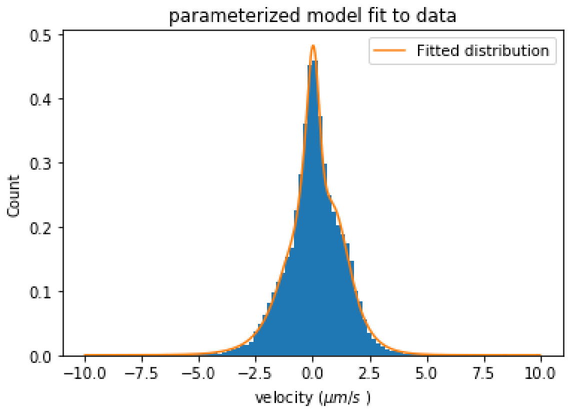


Figure 15: The dendrite branch velocity histogram (shown in blue) overlayed with the distribution obtained by Gibbs sampling (shown in orange). Gibbs sampling with Otsu's initialization recovers parameters that result in a good fit to the data.

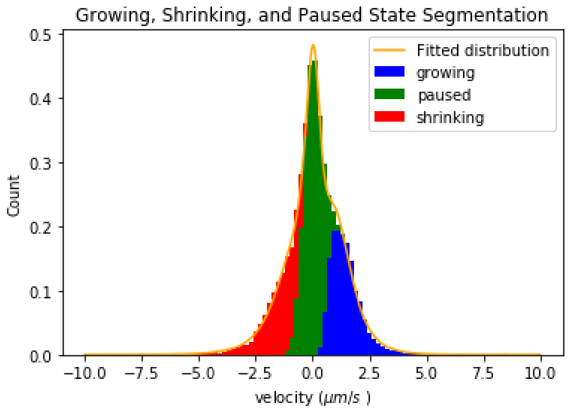


Figure 16: Results of final Gibbs sampling data segmentation into growing, paused, and shrinking states.

5 Conclusion

The results indicate that the Gibbs sampling algorithm can successfully be applied to parameterize mixture models including the model of dendrite branch velocity. However, it is important to note that initialization appears to play an important role in the success of the Gibbs sampler for this case. Further investigation into initialization and the shortcomings of Gibbs sampling algorithms for mixture models may be necessary.

The good fit of our distribution to the data (Fig. 15) and the reasonable segmentation (Fig. 16) further indicates that our choice of a 3-component log-N-Gauss-log-N mixture model accurately models the data. This supports the Howard lab's hypothesis that neuronal Class IV dendrites do indeed display distinguishable growing, paused, and shrinking states, providing further insight into dendrite morphogenesis and prompting further investigation of underlying biological mechanisms driving the 3 distinct dynamic states of neural dendrite development.

The results additionally provide a more rigorous means of quantifying model parameters with interpretable confidence intervals as well as a rigorous method for segmenting experimental data into growing, shrinking, and paused states with an associated probability. This can improve methods for modelling and simulating dendrite morphogenesis, improving our mechanistic understanding of neural processes.

Since healthy cognitive functioning as well as certain neurological diseases have been linked to proper or defective dendrite development, the results of this study may in the long-term help provide more insight into the importance of dendrite dynamics in proper neural development and how deviations in dendrite dynamics may contribute to the emergence of neurological disease.

6 References

(1) Jan, Lily et al., 2010. Branching Out: Mechanisms of Dendritic Arborization. *Nat Rev Neurosci*, Vol 11 pp. 316-328.

(2) Lambert, Ben. A Students Guide to Bayesian Statistics. *SAGE*, 2018.

(3) Barker, B.S. et al., 2017. Conn's Translational Neuroscience: Chapter 2 - Ion Channels. *Academic Press.* pp. 11-43.

(4) Fridman, Daniel. *Bayesian Inference for the Parameterization of Mixture Models with Biophysical Applications*, S&DS 480 Independent Study Report, Yale University. December 2019.

(5) Hines, K., 2015. A Primer on Bayesian Inference for Biophysical Systems. *Biophysical Journal*, Vol 108 pp. 2103-2113.

(6) Jordan, M. *The Conjugate Prior for the Normal Distribution*, lecture notes, Stat260: Bayesian Modeling and Inference, UC Berkeley, delivered February 8 2010.

(7) Otsu, Nobuyuki, 1979. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions and Systems*, Vol SMC-9 pp. 62-66.

7 Acknowledgements

I would like to thank Prof. Jonathan Howard for serving as my senior thesis advisor throughout this project and for providing me with the opportunity to work in his lab. I would also like to thank Olivier Trottier for directly supervising me throughout my project and for all of his help and guidance. Additionally, I would like to thank Sabya Sutradhar for providing the dendrite branch velocity data and advising me on the project as well as the entire Howard Lab. I would also like to thank Sekhar Tatikonda for serving as my DUS and the entire S&DS department for its support of my senior thesis project.