State-Space Methods for Inferring Spike Trains from Calcium Imaging

Joshua Vogelstein

Johns Hopkins

April 23, 2009
What is our goal?

Inferring spike trains using only calcium imaging
Why is this a hard problem?

Many reasons...

1. Too many spike trains to search through them all
2. Noise is non-Gaussian
3. Observation are non-linear
4. Parameters are unknown
What are we going to do?

Our strategy

- Write down a **generative model**, explaining the causal relationship between spikes and movies
- Develop an algorithm to **invert** that model, to obtain spike trains and microcircuits from the movies
- **Test** our approach on real data
Outline

1. Introduction
2. General Methods
3. Simplifying our model
4. Fast non-negative spike inference (FANSI)
5. Particle-filter-smoother (PFS) spike inference
6. PFS results
7. Concluding thoughts
Generative Model is a state-space model

Our generative model

\[ n_t \sim \text{Poisson}(\lambda \Delta) \]

\[ \tau = C_t - C_{t-1} = -C_{t-1} + n_t Y_x, t = \alpha_x [C_t + \beta] + \sigma_x \epsilon_x, t, \epsilon_x, t \text{ iid } \sim N(0, 1) \]
Generative Model is a state-space model

Our generative model

\[ n_t \sim \text{Poisson}(\lambda \Delta) \]
Generative Model is a state-space model

Our generative model

\[ n_t \sim \text{Poisson}(\lambda\Delta) \]

\[ \tau \frac{C_t - C_{t-1}}{\Delta} = -C_{t-1} + n_t \]
Generative Model is a state-space model

Our generative model

\[ n_t \sim \text{Poisson}(\lambda \Delta) \]

\[ \tau \frac{C_t - C_{t-1}}{\Delta} = -C_{t-1} + n_t \]

\[ Y_{x,t} = \alpha_x [C_t + \beta] + \sigma Y \varepsilon_{x,t}, \quad \varepsilon_{x,t} \overset{iid}{\sim} \mathcal{N}(0, 1) \]
Generative Model is a state-space model

Our generative model

\[
\begin{align*}
n_t & \sim \text{Poisson}(\lambda \Delta) \\
\tau \frac{C_t - C_{t-1}}{\Delta} & = -C_{t-1} + n_t \\
Y_{x,t} & = \alpha_x [C_t + \beta] + \sigma_Y \varepsilon_{x,t}, \quad \varepsilon_{x,t} \overset{iid}{\sim} \mathcal{N}(0, 1)
\end{align*}
\]

General state-space formalism

\[
\begin{align*}
C_t & = \gamma C_{t-1} + n_t, & n_t & \sim \text{Poisson}(\lambda \Delta) \\
\vec{Y}_t & = \alpha C_t + \epsilon_t, & \epsilon_t & \sim \mathcal{N}(\alpha \beta, \sigma_Y^2 I)
\end{align*}
\]
Generative Model is a state-space model

General state-space formalism

\[ C_t = \gamma C_{t-1} + n_t, \quad n_t \sim \text{Poisson}(\lambda\Delta) \]
\[ \vec{Y}_t = \alpha C_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(\alpha\beta, \sigma_Y^2 I) \]

State-space distributions

- Transition distribution: \( P_\theta(C_t|C_{t-1}) \) is Linear-Poisson
- Observation distribution: \( P_\theta(\vec{Y}_t|C_t) \) is Linear-Gaussian
Generative Model for a single neuron

Some Notation

- $n = \{n_t\}_{t=0}^T$ is the spike train
- $\varepsilon \sim \mathcal{N}(\mu, \Sigma)$ means $\varepsilon$ is distributed according to a Gaussian with mean $\mu$ and covariance $\Sigma$
- $\vec{Y}_t = \{Y_{x,t}\}_{x=0}^P$ is the $t$-th image frame
- $\vec{Y} = \{\vec{Y}_t\}_{t=1}^T$ is the entire movie
- $\theta = \{\lambda, \tau, \sigma_c, \alpha, \beta, \sigma_Y\}$ is the set of model parameters
Generative Model for a single neuron

Simulation
Generative Model for a single neuron

Schematic

Spatially Filtered Fluorescence

Calcium

Spike Train

Time (sec)

Joshua Vogelstein (Johns Hopkins)

State-Space Calcium Imaging

April 23, 2009
Inverting the model

What does this even mean?

The model defines:

**Likelihood:**

\[ P_{\theta}(\vec{Y}|n) = N(\vec{Y}; \mu, \Sigma) \]

**Prior:**

\[ P_{\theta}(n) = \text{Poisson}(\lambda \Delta) \]

We want the **posterior**:

\[ P_{\theta}(n|\vec{Y}) \]

We know **Bayes Rule**:

\[ P_{\theta}(n|\vec{Y}) \propto P_{\theta}(\vec{Y}|n) P_{\theta}(n) \]

So, that should be no problem, right?
Inverting the model

What does this even mean?

- The model defines:
  - **Likelihood**: $P_{\theta}(\tilde{Y} | n) = \mathcal{N}(\tilde{Y}; \mu, \Sigma)$
  - **Prior**: $P_{\theta}(n) = \text{Poisson}(\lambda \Delta)$
Inverting the model

What does this even mean?

- The model defines:
  - Likelihood: \( P_\theta(\tilde{Y}|n) = \mathcal{N}(\tilde{Y}; \mu, \Sigma) \)
  - Prior: \( P_\theta(n) = \text{Poisson}(\lambda \Delta) \)
- We want the posterior: \( P_\theta(n|\tilde{Y}) \)
Inverting the model

What does this even mean?

- The model defines:
  - Likelihood: $P_{\theta}(\vec{Y}|n) = \mathcal{N}(\vec{Y}; \mu, \Sigma)$
  - Prior: $P_{\theta}(n) = \text{Poisson}(\lambda \Delta)$

- We want the posterior: $P_{\theta}(n|\vec{Y})$

- We know Bayes Rule: $P_{\theta}(n|\vec{Y}) \propto P_{\theta}(\vec{Y}|n)P_{\theta}(n)$
General Methods

Inverting the model

What does this even mean?

- The model defines:
  - Likelihood: \( P_\theta(\vec{Y}|n) = \mathcal{N}(\vec{Y}; \mu, \Sigma) \)
  - Prior: \( P_\theta(n) = \text{Poisson}(\lambda \Delta) \)

- We want the posterior: \( P_\theta(n|\vec{Y}) \)

- We know Bayes Rule: \( P_\theta(n|\vec{Y}) \propto P_\theta(\vec{Y}|n)P_\theta(n) \)

- So, that should be no problem, right?
Inverting the model

Problems
Inverting the model

Problems
We can compute $P_\theta(n|\vec{Y})$ for any individual $n$, but we may want a point estimate of this distribution.
Inverting the model

Problems

- We can compute $P_\theta(n|\vec{Y})$ for any individual $n$, but we may want a point estimate of this distribution.
- For instance, a desirable quantity might be $n_{MAP} = \arg\max_n P_\theta(n|\vec{Y})$. 
Inverting the model

Problems

- We can compute $P_\theta(n|\tilde{Y})$ for any individual $n$, but we may want a point estimate of this distribution.
- For instance, a desirable quantity might be $n_{MAP} = \arg\max_n P_\theta(n|\tilde{Y})$.
- Another point estimate of interest may be $n_{mean} = E[n|\tilde{Y}]$. 

By Joshua Vogelstein (Johns Hopkins)
We can compute $P_\theta(n|\vec{Y})$ for any individual $n$, but we may want a point estimate of this distribution.

For instance, a desirable quantity might be $n_{MAP} = \arg\max_n P_\theta(n|\vec{Y})$.

Another point estimate of interest may be $n_{mean} = E[n|\vec{Y}]$.

Both of these point estimates require having all possible spike trains.
General Methods

Inverting the model

Problems

- We can compute $P_\theta(n|\vec{Y})$ for any individual $n$, but we may want a point estimate of this distribution.
- For instance, a desirable quantity might be $n_{MAP} = \arg\max_n P_\theta(n|\vec{Y})$.
- Another point estimate of interest may be $n_{mean} = E[n|\vec{Y}]$.
- Both of these point estimates require having all possible spike trains.
- Because $n \sim \text{Poisson}(\lambda \Delta)$:
  - there are an infinite number of possible spike trains.
  - we lack the calculus to integrate over Poisson.
What to do? Approximate!

Two general options

- **Simplify** the assumptions to get something tractable
- **Monte Carlo** sample from the model, to approximate the distributions
Outline

1. Introduction
2. General Methods
3. Simplifying our model
4. Fast non-negative spike inference (FANSI)
5. Particle-filter-smoother (PFS) spike inference
6. PFS results
7. Concluding thoughts
How can we simplify?

Simplification steps

1. Explicitly **state** the all assumptions of our model
2. Determine **which** ones are giving us trouble
3. Try approximating them
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps instantaneously after each spike
- Calcium decays mono-exponentially
- Calcium jumps the same size with each spike
- There are no other sources of fluctuations in calcium
- Observations are a linear-Gaussian function of calcium

Which of these assumptions screws us?

Poisson spikes!
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps instantaneously after each spike
- Calcium decays mono-exponentially
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
- Calcium decays *mono-exponentially*
- Calcium jumps the *same size* with each spike
Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
- Calcium decays *mono-exponentially*
- Calcium jumps the *same size* with each spike
- There are no *other sources* of fluctuations in calcium
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
- Calcium decays *mono-exponentially*
- Calcium jumps the *same size* with each spike
- There are no *other sources* of fluctuations in calcium
- Observations are a *linear-Gaussian* function of calcium
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
- Calcium decays *mono-exponentially*
- Calcium jumps the *same size* with each spike
- There are no *other sources* of fluctuations in calcium
- Observations are a *linear-Gaussian* function of calcium

Which of these assumptions screws us?
How can we simplify our model?

Stating our assumptions

- Spikes are Poisson
- Calcium jumps *instantaneously* after each spike
- Calcium decays *mono-exponentially*
- Calcium jumps the *same size* with each spike
- There are no *other sources* of fluctuations in calcium
- Observations are a *linear-Gaussian* function of calcium

Which of these assumptions screws us?

Poisson spikes!
How can we approximate Poisson spikes?

Some reasonable options

1. As $\Delta \to 0$, Poisson $\to$ Bernoulli
2. When Poisson rate is low, Poisson $\approx$ Exponential
3. When Poisson rate is high, Poisson $\approx$ Gaussian
How can we approximate Poisson spikes?

Some reasonable options

1. As $\Delta \to 0$, Poisson $\to$ Bernoulli
How can we approximate Poisson spikes?

Some reasonable options

1. As $\Delta \to 0$, Poisson $\to$ Bernoulli
2. When Poisson rate is low, Poisson $\approx$ Exponential
How can we approximate Poisson spikes?

Some reasonable options

1. As $\Delta \to 0$, Poisson $\to$ Bernoulli
2. When Poisson rate is low, Poisson $\approx$ Exponential
3. When Poisson rate is high, Poisson $\approx$ Gaussian
How can we approximate Poisson spikes?

Some reasonable options

1. As $\Delta \to 0$, Poisson $\to$ Bernoulli
2. When Poisson rate is low, Poisson $\approx$ Exponential
3. When Poisson rate is high, Poisson $\approx$ Gaussian
Rigorously stating the problem to solve

\[ n_{MAP} = \arg\max_n P_\theta(n|\vec{Y}) = \arg\max_n P_\theta(\vec{Y}|n)P_\theta(n) \]
A closer look at the problem at hand

Rigorously stating the problem to solve

\[
\mathbf{n}_{MAP} = \arg\max_{\mathbf{n}} P_\theta(\mathbf{n} | \mathbf{Y}) = \arg\max_{\mathbf{n}} P_\theta(\mathbf{Y} | \mathbf{n}) P_\theta(\mathbf{n})
\]

\[
= \arg\max_{\mathbf{n}_t \forall t} \prod_{t=1}^{T} P_\theta(Y_t | n_t) P_\theta(n_t)
\]
A closer look at the problem at hand

Rigorously stating the problem to solve

$$n_{MAP} = \arg\max_n P_\theta(n|\vec{Y}) = \arg\max_n P_\theta(\vec{Y}|n)P_\theta(n)$$

$$= \arg\max_{n_t \forall t} \prod_{t=1}^{T} P_\theta(Y_t|n_t)P_\theta(n_t)$$

$$= \arg\max_{n_t \forall t} \sum_{t=1}^{T} \log P_\theta(Y_t|C_t) + \log P_\theta(n_t)$$
Rigorously stating the problem to solve

\[ n_{MAP} = \arg\max_n P_\theta(n|\vec{Y}) = \arg\max_n P_\theta(\vec{Y}|n)P_\theta(n) \]

\[ = \arg\max_{n_t \forall t} \prod_{t=1}^T P_\theta(Y_t|n_t)P_\theta(n_t) \]

\[ = \arg\max_{n_t \forall t} \sum_{t=1}^T \log P_\theta(Y_t|C_t) + \log P_\theta(n_t) \]

\[ = \arg\max_{n_t \forall t} \sum_{t=1}^T \log \mathcal{N}(\vec{Y}_t - \alpha C_t - 1\beta, \sigma_Y^2) + \log P_\theta(n_t) \]
A closer look at the problem at hand

Rigorously stating the problem to solve

\[ n_{MAP} = \arg\max_n P_\theta(n|\vec{Y}) = \arg\max_n P_\theta(\vec{Y}|n)P_\theta(n) \]

\[ = \arg\max_{n_t \forall t} \prod_{t=1}^{T} P_\theta(Y_t|n_t)P_\theta(n_t) \]

\[ = \arg\max_{n_t \forall t} \sum_{t=1}^{T} \log P_\theta(Y_t|C_t) + \log P_\theta(n_t) \]

\[ = \arg\max_{n_t \forall t} \sum_{t=1}^{T} \log \mathcal{N}(\vec{Y}_t - \alpha C_t - 1\beta, \sigma^2_Y I) + \log P_\theta(n_t) \]

\[ = \arg\max_{n_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma^2_Y} \left\| \vec{Y}_t - \alpha C_t - 1\beta \right\|^2 + \log P_\theta(n_t) \]
Rewriting the prior in terms of $C_t$

\[ \tau \frac{C_t - C_{t-1}}{\Delta} = -C_{t-1} + n_{t-1} \]

\[ C_t = \gamma C_{t-1} + n_{t-1} \]

\[ n_{t-1} = (C_t - \gamma C_{t-1}) \]

\[ n = MC \]
Writing the prior in terms of $C_t$

Rewriting the calcium dynamics

$$\tau \frac{C_t - C_{t-1}}{\Delta} = -C_{t-1} + n_{t-1}$$

$$C_t = \gamma C_{t-1} + n_{t-1}$$

$$n_{t-1} = (C_t - \gamma C_{t-1})$$

$$n = MC$$

Rewriting the optimization in terms of $C_t$

$$C_{MAP} = \arg\max_{C_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \tilde{Y}_t - \alpha C_t - 1\beta \right\|^2 + \log P_\theta(C_t - \gamma C_{t-1})$$
A closer look at the Gaussian approximation

Gaussian approximation of $P_{\theta}(n)$

$C_{MAP} = \text{argmax}_{C_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \tilde{Y}_t - \alpha C_t - 1_\beta \right\|^2 + \log P_{\theta}(n_t)$

$\approx \text{argmax}_{C_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \tilde{Y}_t - \alpha C_t - 1_\beta \right\|^2 + \log \mathcal{N}(n_t; \lambda \Delta, \lambda \Delta)$

$= \text{argmax}_{C_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \tilde{Y}_t - \alpha C_t - 1_\beta \right\|^2 + \frac{-1}{2(\lambda \Delta)^2} \left( C_t - \gamma C_{t-1} - \lambda \Delta \right)^2$
Simplifying our model

Gaussian approximation

A closer look at the Gaussian approximation

Gaussian approximation of $P_{\theta}(n)$

$$C_{MAP} = \arg\max_{C_t \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \vec{Y}_t - \alpha C_t - \beta \right\|^2 + \frac{-1}{2(\lambda \Delta)^2} (C_t - \gamma C_{t-1} - \lambda \Delta)^2$$

Thoughts on the Gaussian approximation

- We have a \textit{quadratic} problem
- It is, in fact, a \textit{Wiener} filter
- We can solve this in $O(T)$
- Spikes are \textit{not} constrained to be non-negative
- Spikes are \textit{not} constrained to be integers
A closer look at the Gaussian approximation

Wiener filter for slow and fast firing rate simulated neurons

Simplifying our model
1. Introduction
2. General Methods
3. Simplifying our model
4. Fast non-negative spike inference (FANSI)
5. Particle-filter-smoother (PFS) spike inference
6. PFS results
7. Concluding thoughts
A closer look at the exponential approximation

Exponential approximation of $P_\theta(n)$

$$C_{\text{MAP}} = \arg\max_{C_t - \gamma C_{t-1} \geq 0 \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \| \tilde{Y}_t - \alpha C_t - 1\beta \|^2 + \log P_\theta(n_t)$$

$$\approx \arg\max_{C_t - \gamma C_{t-1} \geq 0 \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \| \tilde{Y}_t - \alpha C_t - 1\beta \|^2 + \log \text{Exp}(n_t; \lambda \Delta)$$

$$= \arg\max_{C_t - \gamma C_{t-1} \geq 0 \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \| \tilde{Y}_t - \alpha C_t - 1\beta \|^2 - \lambda \Delta(C_t - \gamma C_{t-1})$$
A closer look at the exponential approximation

Exponential approximation of $P_\theta(n)$

$$C_{MAP} = \arg \max_{C_t - \gamma C_{t-1} \geq 0, \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma_Y^2} \left\| \bar{Y}_t - \alpha C_t - 1\beta \right\|^2 - \lambda \Delta (C_t - \gamma C_{t-1})$$

Thoughts on the exponential approximation

- We have a **concave** (but not quadratic) problem
- We can solve this in $O(T)$
- Spikes are constrained to be non-negative
- Spikes are not constrained to be integers
A closer look at the exponential approximation

**FANSI for slow and fast firing rate simulated neurons**

**Slow Firing Rate**

Fluorescence

**Fast Firing Rate**

Fluorescence

**Fast Filter**

Time (sec)
FANSI algorithm for solving non-negative state-space problems

**Direct** method: optimize $C_{MAP}$ directly

$$
C_{MAP} = \arg \max_{C_t - \gamma C_{t-1} \geq 0 \ \forall t} \sum_{t=1}^{T} \frac{-1}{2\sigma^2_Y} \left\| \tilde{Y}_t - \alpha C_t - 1 \beta \right\|^2 - \lambda \Delta (C_t - \gamma C_{t-1})
$$

$$
= \arg \max_{C_t - \gamma C_{t-1} \geq 0 \ \forall t} \frac{-1}{2\sigma^2_Y} \left\| \tilde{Y}_t - \alpha C_t - 1 \beta \right\|^2 - \lambda \Delta (MC)'1
$$

**Barrier** method: optimize $C_z$ iteratively

$$
C_z = \arg \max_{C_t \ \forall t} \frac{-1}{2\sigma^2_Y} \left\| \tilde{Y}_t - \alpha C_t - 1 \beta \right\|^2 - \lambda \Delta (MC)'1 + z \log (MC)'1
$$
Finding $C_z$

It’s concave, so we just use Newton-Raphson

\[
\mathcal{L}_z = \frac{-1}{2\sigma_Y^2} \left\| \tilde{Y}_t - \alpha C_t - 1\beta \right\|^2 - \lambda \Delta (MC)'1 + z \log (MC)'1
\]

\[
g = \frac{\alpha}{\sigma_Y^2} (\tilde{Y} - \alpha C' - 1\beta) + \lambda \Delta M'1 - zM'(MC_z)^{-1}
\]

\[
\tilde{H} = \frac{\alpha'\alpha}{\sigma_Y^2} + zM'(MC_z)^{-2}M
\]

$C_z \leftarrow C_z + sd$

$\tilde{H}d = g \Rightarrow d = \tilde{H}\backslash g$

Some thoughts

- Because $M$ is bidiagonal, $\tilde{H}$ is tridiagonal
- This means that we can use Gaussian elimination
Estimating the parameters

Pseudo-EM

- EM requires **sufficient statistics** that we don’t compute
- However, we can **estimate**

\[
\hat{\theta} = \arg\max_{\theta \in \Theta} \int P(\tilde{Y}|n, \theta)P(n|\theta)dn \approx \arg\max_{\theta \in \Theta} P(\tilde{Y}|\hat{n}, \theta)P(\hat{n}|\theta)
\]

- We **iterate** finding \(\hat{n}\) and \(\hat{\theta}\)
- The likelihood for the parameters is **concave**
- In practice, we can always find good parameters, without modifying the initial values, with relatively little data
Using our FANSI filter

**in vivo data**

Fluorescence Projection

Fast Filter

Time (sec)

Joshua Vogelstein (Johns Hopkins) State-Space Calcium Imaging April 23, 2009 28 / 78
Fast non-negative spike inference (FANSI)

- Due to state-space nature, requires only $O(T)$
- Outperforms Wiener filter for both fast and slow spiking neurons
- Works for in vivo data
- Parameter estimation is simple and straightforward (and unsupervised)
Relaxing various assumptions, but staying within the state-space framework

- Time varying rate — more flexible prior
- Slow rise time — add another $C$ with different dynamics
- Poisson observations — likelihood maintains concavity (unlike Poisson dynamics)
- Optimal thresholding — initialize an integer programming algorithm with $\hat{n}$
- Multiple neurons — even overlapping spatial filters is ok
Desirata not easily incorporated into our FANSI framework

- Non-Poisson spiking — eg, spike history dependence
- Non-linear observations — fluorescence saturates
- Errorbars — FANSI only provides a MAP estimate (which is not conducive to a Laplace approximation)
- Coupling between neurons — an extension of spike history dependence
- Better parameter estimation — requires sufficient stats not available from FANSI
So, what can we do, to achieve these desirata?

Remember this slide?

- **Simplify** the assumptions to get something tractable
- **Monte Carlo** sample from the model, to approximate the distributions

We already did the simplify option, let’s try the Monte Carlo option.
Remember the state-space formalism?

General state-space formalism

\[ C_t = \gamma C_{t-1} + n_t, \]
\[ \vec{Y}_t = \alpha C_t + \epsilon_t, \]
\[ n_t \sim \text{Possion}(\lambda \Delta) \]
\[ \epsilon_t \sim \mathcal{N}\left(\alpha \beta, \sigma_Y^2 \mathbf{I}\right) \]
Remember the state-space formalism?

**General state-space formalism**

\[
C_t = \gamma C_{t-1} + n_t, \quad n_t \sim \text{Poisson}(\lambda \Delta)
\]

\[
\vec{Y}_t = \alpha C_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(\alpha \beta, \sigma^2 Y)
\]

**So, what does sampling mean here?**

- 
- 

Joshua Vogelstein (Johns Hopkins) State-Space Calcium Imaging April 23, 2009 34 / 78
Remember the state-space formalism?

**General state-space formalism**

\[
C_t = \gamma C_{t-1} + n_t, \quad n_t \sim \text{Poisson}(\lambda \Delta)
\]

\[
\vec{Y}_t = \alpha C_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(\alpha \beta, \sigma^2_Y I)
\]

**So, what does **sampling** mean here?**

- Sample spike trains!
Particle-filter-smoother (PFS) spike inference

Remember the state-space formalism?

General state-space formalism

\[ C_t = \gamma C_{t-1} + n_t, \quad n_t \sim \text{Poisson}(\lambda \Delta) \]
\[ \vec{Y}_t = \alpha C_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(\beta \beta, \sigma^2 \mathbf{I}) \]

So, what does **sampling** mean here?

- Sample spike trains!
- How do we do it?
What are some ways we can sample spike trains?

Three general approaches:

- Naïve — too slow: \( > 2^T \) possible spike trains
- Markov Chain Monte Carlo (MCMC) — too difficult: because space of spike trains is so non-convex
- Sequential Monte Carlo (or particle filter; SMC) — just right!
What are some ways we can sample spike trains?

Three general approaches

- Naïve — too slow: $> 2^T$ possible spike trains
- Markov Chain Monte Carlo (MCMC) — too difficult: because space of spike trains is so non-convex
What are some ways we can sample spike trains?

Three general approaches

- Naïve — too slow: $> 2^T$ possible spike trains
- Markov Chain Monte Carlo (MCMC) — too difficult: because space of spike trains is so non-convex
- Sequential Monte Carlo (or particle filter; SMC) — just right!
What are some ways we can sample spike trains?

Three general approaches

- Naïve — too slow: $> 2^T$ possible spike trains
- Markov Chain Monte Carlo (MCMC) — too difficult: because space of spike trains is so non-convex
- Sequential Monte Carlo (or particle filter; SMC) — just right!

What’s the big idea?

- Let’s say we have a state-space model
- and we want to infer some statistics about the hidden state
- what do we do?
**What are some ways we can sample spike trains?**

### Three general approaches

1. Naïve — too slow: $> 2^T$ possible spike trains
2. Markov Chain Monte Carlo (MCMC) — too difficult: because space of spike trains is so non-convex
3. Sequential Monte Carlo (or particle filter; SMC) — just right!

### What’s the big idea?

1. Let’s say we have a state-space model
2. and we want to infer some statistics about the hidden state
3. what do we do?
4. **forward-backward** algorithm
Forward backward algorithms

Two familiar special cases
Forward backward algorithms

Two familiar special cases

- Discrete state-space models (HMM) — remember the Baum-Welch algorithm?
Forward backward algorithms

Two familiar special cases

- Discrete state-space models (HMM) — remember the Baum-Welch algorithm?
- Linear-Gaussian dynamics — remember the Kalman filter-smooth?
Forward recursion

- First compute the probability of the hidden state, given all previous observations, $P_\theta(H_t | \vec{Y}_{0:t})$
- This is called the filter (or forward) distribution
Forward backward algorithms — how do they work again?

Forward recursion
- First compute the probability of the hidden state, given all previous observations, $P_{\theta}(H_t|\vec{Y}_{0:t})$
- This is called the filter (or forward) distribution

Backward recursion
- Then compute the probability of the hidden state, given all observations (both past and future), $P_{\theta}(H_t|\vec{Y}_{0:T})$
- This is called the smooth (or backward) distribution
What are these recursions?

**Forward recursion**

\[
P_{\theta}(H_t | \vec{Y}_{0:t}) = \frac{1}{Z} P_{\theta}(\vec{Y}_t | H_t) \int P_{\theta}(H_t | H_{t-1}) P_{\theta}(H_{t-1} | \vec{Y}_{0:t-1}) dH_{t-1}
\]
What are these recursions?

**Forward recursion**

\[
P_\theta(H_t|\vec{Y}_{0:t}) = \frac{1}{Z} P_\theta(\vec{Y}_t|H_t) \int P_\theta(H_t|H_{t-1}) P_\theta(H_{t-1}|\vec{Y}_{0:t-1}) dH_{t-1}
\]

**Backward recursion**

\[
P_\theta(H_t, H_{t-1}|\vec{Y}) = P_\theta(H_t|\vec{Y}) \frac{P_\theta(H_t|H_{t-1}) P_\theta(H_{t-1}|\vec{Y}_{0:t-1})}{\int P_\theta(H_t|H_{t-1}) P_\theta(H_{t-1}|\vec{Y}_{0:t-1}) dH_{t-1}}
\]

\[
P_\theta(H_{t-1}|\vec{Y}) = \int P_\theta(H_t, H_{t-1}|\vec{Y}) dH_t
\]
Consider the integral

$$\int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|\vec{Y}_{0:t-1})dH_{t-1}$$

Under which circumstances can we analytically evaluate that integral?

- When $H_t$ takes finite possible values (i.e., for HMM)
Consider the integral
\[
\int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|\vec{Y}_{0:t-1})dH_{t-1}
\]

Under which circumstances can we analytically evaluate that integral?

- When \( H_t \) takes finite possible values (ie, for HMM)
- When both distributions are Gaussian, as the product of Gaussians are... Gaussian
Consider the integral

\[ \int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|\vec{Y}_{0:t-1})dH_{t-1} \]

What can we do to approximate this integral?
Consider the integral

$$\int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|Y_{0:t-1})dH_{t-1}$$

What can we do to approximate this integral?

- Discretize on a grid
Consider the integral

\[
\int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|\tilde{Y}_{0:t-1})dH_{t-1}
\]

What can we do to approximate this integral?

- Discretize on a grid
- Approximation distributions as Gaussians (Laplace approximation)
Consider the integral

\[ \int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}|\tilde{Y}_{0:t-1})dH_{t-1} \] 

What can we do to approximate this integral?

- Discretize on a grid
- Approximation distributions as Gaussians (Laplace approximation)
- Sample!
Approximating $P_\theta(H_t|\vec{Y}_{0:t})$

Approximate with a histogram

$$P_\theta(H_t|\vec{Y}_{0:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta \left( H_t - H_t^{(i)} \right)$$

- $N$ particles at each time step
- $w_t^{(i)}$ indicates the weight (likelihood) particle $i$ at time $t$
- $H_t^{(i)}$ indicates the position of particle $i$ at time $t$
- Collectively, they comprise the approximation to our distribution
Approximating $P_{\theta}(H_t | \vec{Y}_{0:t})$

Approximate with a histogram

$$P_{\theta}(H_t | \vec{Y}_{0:t}) \approx \sum_{i=1}^{N} w_{t}^{(i)} \delta \left( H_t - H_{t}^{(i)} \right)$$
Substituting this approximation into the integral

Given samples, what do we do?

\[
P_{\theta}(H_t | \tilde{Y}_{0:t}) = \frac{1}{Z} P_{\theta}(\tilde{Y}_t | H_t) \int P_{\theta}(H_t | H_{t-1}) P_{\theta}(H_{t-1} | \tilde{Y}_{0:t-1}) dH_{t-1}
\]

\[
\tilde{w}_t^{(i)} = P_{\theta}\left(\tilde{Y}_t | H_t^{(i)}\right) \sum_{j=1}^{N} P_{\theta}\left(H_t^{(i)} | H_{t-1}^{(j)}\right) w_t^{(j)}
\]

\[
\approx P_{\theta}\left(\tilde{Y}_t | H_t^{(i)}\right) P_{\theta}\left(H_t^{(i)} | H_{t-1}^{(i)}\right) w_t^{(i)}
\]

\[
w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}
\]
But how do we actually get the samples?

Importance sampling

\[ \tilde{w}_t^{(i)} \approx \frac{P_\theta \left( \tilde{Y}_t | H_t^{(i)} \right) P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)} \right) w_{t-1}^{(i)}}{q \left( H_t^{(i)} \right)} \]

- \( q(\cdot) \) is called the proposal (or importance) distribution (or density)
- \( q(\cdot) \) can depend on anything in the past, ie, \( \left\{ H_{t-1}^{(j)} \right\}_{j=1}^{N} \) and \( \tilde{Y}_{0:t} \)
- we have various “standard” options
Particle-filter-smoother (PFS) spike inference

The forward recursion

Typical proposal distribution options

Transition distribution sampler

If \( q(\cdot) = P_\theta \left( \mathbf{H}_t^{(i)} | \mathbf{H}_{t-1}^{(i)} \right) \), then the computation of the forward distribution is straightforward:

\[
\tilde{w}_t^{(i)} = P_\theta \left( \tilde{\mathbf{Y}}_t | \mathbf{H}_t^{(i)} \right) w_{t-1}^{(i)}
\]
Typical proposal distribution options

Transition distribution sampler

If $q(\cdot) = P_{\theta} \left( H^{(i)}_t | H^{(i)}_{t-1} \right)$, then the computation of the forward distribution is straightforward:

$$\tilde{w}^{(i)}_t = P_{\theta} \left( \tilde{Y}_t | H^{(i)}_t \right) w^{(i)}_{t-1}$$

One step ahead sampler

- $q(\cdot) = P_{\theta} \left( H^{(i)}_t | H^{(i)}_{t-1}, \tilde{Y}_t \right) \propto P_{\theta} \left( H^{(i)}_t | H^{(i)}_{t-1} \right) P_{\theta} \left( \tilde{Y}_t | H^{(i)}_t \right)$
- more efficient than the transition distribution, as it also considers observations
- Computing the forward distribution no longer simplifies
- It is sometimes difficult to sample from
Typical proposal distribution options

One step ahead sampler

- $q(\cdot) = P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)}, \vec{Y}_t \right) \propto P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)} \right) P_\theta \left( \vec{Y}_t | H_t^{(i)} \right)$
- more efficient than the transition distribution, as it also considers observations
- Computing the forward distribution no longer simplifies
- It is sometimes difficult to sample from

Optimal sampler

- $q(\cdot) = P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)}, \vec{Y}_{t:T} \right) \propto P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)} \right) P_\theta \left( \vec{Y}_{t:T} | H_t^{(i)} \right)$
- Uses all available information
- Sometimes (approximately) possible using a backwards recursion
The procedure so far

For each time step, for each particle:

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights
The procedure so far

For each time step, for each particle

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights

Anybody see a problem?
The procedure so far

For each time step, for each particle

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights

Anybody see a problem?

- Weights **degenerate**
The procedure so far

For each time step, for each particle

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights

Anybody see a problem?

- Weights degenerate
- Draw an example on the board
Particle-filter-smoother (PFS) spike inference

The forward recursion

Anybody see a solution

Resample

Sample particles (with replacement) according to their weights

If we do it too often, we reduce to our particle diversity

If we do it too infrequently, the weights degenerate

Thus, it is standard to resample when the effective number of particles is too small:

\[ \hat{N}_{\text{eff}} = N \sum_{i=1}^{N} \left( w(i) \right)^2 \]

The threshold is typically taken to be \( N/2 \)

Goal: expectation of resampled distribution should be equal to original distribution

Given this constraint, we want to minimize variance
Anybody see a solution

Resample

- Sample particles (with replacement) according to their weights
- If we do it too often, we reduce to our particle diversity
- If we do it too infrequently, the weights degenerate
- Thus, it is standard to resample when the effective number of particles is too small:

\[
\hat{N}_{\text{eff}} = \sum_{i=1}^{N} \left( w_t^{(i)} \right)^2
\]

- The threshold is typically taken to be \( N/2 \)
- Goal: expectation of resampled distribution should be equal to original distribution
- Given this constraint, we want to minimize variance
Resampling schemes

Multinomial resampling

- Draw particle $i$ with probability $w_t^{(i)}$
- This is the simplest strategy, but not the best
Resampling schemes

Multinomial resampling

- Draw particle $i$ with probability $w_t^{(i)}$
- This is the simplest strategy, but not the best

Stratified resampling

- Discretize $(0, 1]$ into $N$ equal partitions
- Sample once from each partition
- Call `cumsum` on the weights to generate a cumulative sum of weights
- Each time a sample falls into the interval for a particular particle, keep that particle
- This approach has a lower conditional variance than multinomial sampling
Particle-filter-smoother (PFS) spike inference

Putting the whole forward recursion (ie, particle filter) together

For each time step, for each particle

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights
4. Stratified resample, if necessary
Putting the whole forward recursion (ie, particle filter) together

For each time step, for each particle

1. Sample from the proposal distribution
2. Update weights
3. Normalize weights
4. Stratified resample, if necessary

Demo the procedure on the board
Remember the backward recursion

\[ P_\theta(H_t, H_{t-1} | \vec{Y}) = P_\theta(H_t | \vec{Y}) \frac{P_\theta(H_t | H_{t-1}) P_\theta(H_{t-1} | \vec{Y}_{0:t-1})}{\int P_\theta(H_t | H_{t-1}) P_\theta(H_{t-1} | \vec{Y}_{0:t-1}) dH_{t-1}} \]

\[ P_\theta(H_{t-1} | \vec{Y}) = \int P_\theta(H_t, H_{t-1} | \vec{Y}) dH_t \]
Remember the backward recursion

\[ P_\theta(H_t, H_{t-1} | \vec{Y}) = P_\theta(H_t | \vec{Y}) \frac{P_\theta(H_t | H_{t-1}) P_\theta(H_{t-1} | \vec{Y}_{0:t-1})}{\int P_\theta(H_t | H_{t-1}) P_\theta(H_{t-1} | \vec{Y}_{0:t-1}) dH_{t-1}} \]

\[ P_\theta(H_{t-1} | \vec{Y}) = \int P_\theta(H_t, H_{t-1} | \vec{Y}) dH_t \]

Substituting the results from our forward recursion, obtaining a particle-filter-smoother (PFS)

\[ P_\theta \left( H^{(i)}_t, H^{(j)}_{t-1} | \vec{Y} \right) = P_\theta \left( H^{(i)}_t | \vec{Y} \right) \frac{P_\theta \left( H^{(i)}_t | H^{(j)}_{t-1} \right) P_\theta \left( H^{(j)}_{t-1} | \vec{Y}_{0:t-1} \right)}{\sum_{j=1}^{N} P_\theta \left( H^{(i)}_t | H^{(j)}_{t-1} \right) P_\theta \left( H^{(j)}_{t-1} | \vec{Y}_{0:t-1} \right)} \]

\[ P_\theta \left( H^{(j)}_{t-1} | \vec{Y} \right) = \sum_{i=1}^{N} P_\theta \left( H^{(i)}_t, H^{(j)}_{t-1} | \vec{Y} \right) \]
Remember the backward recursion

Backward recursion

\[
P_\theta(H_t, H_{t-1}| \vec{Y}) = P_\theta(H_t| \vec{Y}) \frac{P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}| \vec{Y}_{0:t-1})}{\int P_\theta(H_t|H_{t-1})P_\theta(H_{t-1}| \vec{Y}_{0:t-1}) dH_{t-1}}
\]

\[
P_\theta(H_{t-1}| \vec{Y}) = \int P_\theta(H_t, H_{t-1}| \vec{Y}) dH_t
\]

Some thoughts on the backwards recursion

- Plug and chug (ie, does not require computing any new distributions)
- Scales with \( O(N^2 \times T) \) because we need the probability of going from any \( N \) particles at time \( t-1 \), to any \( N \) particles at time \( t \)
- For applications like ours, can really refine our estimates (as most of the information comes from after the spike)
The whole PFS algorithm for non-analytic state-space models

For $t = 1, \ldots, T$, for each particle
- Sample from the proposal distribution
- Update weights
- Normalize weights
- Stratified resample, if necessary

For $t = T, \ldots, 1$, for each particle
- Update backward distribution

Things to choose
- Proposal
- Number of particles
- Resampling details
Our model (slightly modified)

\[
\begin{align*}
n_t & \sim B(\lambda \Delta) \\
C_t & = \gamma C_{t-1} + A n_t + \sigma c \varepsilon_t, \\
\vec{Y}_t & = \alpha C_t + \epsilon_t,
\end{align*}
\]

\[
\varepsilon_t \sim N(0, 1)
\]

The necessary distributions

\[
P_\theta(C_t|C_{t-1}, n_t) \sim \begin{cases} 
N(\gamma C_{t-1} + A n_t, \sigma^2_c) & \text{if } n_t = 1 \\
N(\gamma C_{t-1}, \sigma^2_c) & \text{if } n_t = 0
\end{cases}
\]

\[
P_\theta(\vec{Y}_t|C_t) \sim N(\alpha C_t + \alpha \beta, \sigma^2_Y I)
\]
One step ahead sampler for our model

Defining $q(\cdot)$

$$q \left( \{C, n\}^{(i)}_t \right) = P_\theta \left( \tilde{Y}_t | C^{(i)}_t \right) P_\theta \left( C^{(i)}_t | \{C_{t-1}, n_t\}^{(i)} \right) P_\theta \left( n_t^{(i)} \right)$$

Procedure

- First sample spikes, by integrating out $C_t$
- Then, given spikes, sample $C_t$
Sampling from $q\left(n_t^{(i)}\right)$ by integrating out $C_t$

Solving the integral

\[
q\left(n_t^{(i)}\right) = \int q\left(\{C, n\}_t^{(i)}\right) dC_t^{(i)} \\
= P_\theta\left(n_t^{(i)}\right) \int P_\theta\left(\vec{Y}_t|C_t^{(i)}\right) P_\theta\left(C_t^{(i)}|\{C_{t-1}, n_t\}^{(i)}\right) dC_t^{(i)} \\
= P_\theta\left(n_t^{(i)}\right) \int \mathcal{N}\left(C_t^{(i)}; \mu_1, \sigma_1^2\right) \mathcal{N}\left(C_t^{(i)}; \mu_2(n_t^{(i)}), \sigma_2^2\right) dC_t^{(i)} \\
= P_\theta\left(n_t^{(i)}\right) \mathcal{N}\left(\vec{Y}_t; \mu_3(\{C_{t-1}, n_t\}^{(i)}), \sigma_3^2\right)
\]

So, we compute the probability of $n_t^{(i)} = 0$ and 1 by plugging those values into the above equation, and then we sample from that distribution.
Sampling from \( q \left( C_t^{(i)} \right) \)

\[
q \left( C_t^{(i)} \right) = \frac{1}{Z} P_\theta \left( \vec{Y}_t | C_t^{(i)} \right) P_\theta \left( C_t^{(i)} | \{ C_{t-1}, n_t \}^{(i)} \right)
\]

\[
= \frac{1}{Z} \mathcal{N} \left( C_t^{(i)}; \mu_4(\vec{Y}_t), \sigma_4^2 \right) \mathcal{N} \left( C_t^{(i)}; \mu_5(\{ C_{t-1}, n_t \}^{(i)}), \sigma_5^2 \right)
\]

\[
= \mathcal{N} \left( C_t^{(i)}, \mu_6(\vec{Y}_t, \{ C_{t-1}, n_t \}^{(i)}), \sigma_6^2 \right)
\]
Particle-filter-smoother (PFS) spike inference

PFS algorithm for our model

Weighting the samples

Plug in the proposal distributions

\[
\tilde{w}_t^{(i)} = w_{t-1}^{(i)} P_\theta \left( \vec{Y}_t | C_t^{(i)} \right) \frac{P_\theta \left( C_t^{(i)} \{C_{t-1}, n_t\}^{(i)} \right) P_\theta \left( n_t^{(i)} \right)}{q \left( n_t^{(i)} \right) q \left( C_t^{(i)} \right)}
\]
Estimating the parameters in our model

The PFS algorithm provides the sufficient statistics for all the parameters

- Each parameter depends on either $P_\theta(H_t | \tilde{Y})$ or $P_\theta(H_t, H_{t-1} | \tilde{Y})$, both of which we have from the forward-backward algorithm
- Parameters are jointly concave, given the inferred hidden distributions
- In practice, a small number of iterations is required to converge to reasonable answers
Some thoughts on our PFS algorithm

Comparison with FANSI

- Should be more accurate than FANSI, as the approximation is better
- Provides errorbars (FANSI doesn’t)
- Parameters can be estimated better than FANSI
- Requires $O(N^2 T)$ (FANSI only requires $O(T)$
- Can be further generalized
Main result using our Particle-Filter-Smoother (PFS)

Inferring a spike train from noisy observations

Simulated Spike Train

Simulated Calcium

Simulated Fluorescence

Wiener Filter

Linear Observation PFS Spike Inference
Incorporating nonlinear observations

Modifying the model

\[ C_t = \gamma C_{t-1} + C_b + An_t + \sigma_c \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim \mathcal{N}(0, 1) \]

\[ \vec{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi} S(C_t) + \eta \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \mathbf{I}) \]

\[ S(C_t) = \frac{C_t^{nd}}{C_t^{nd} + k_d} \]
Incorporating nonlinear observations

Modifying the model

\[ C_t = \gamma C_{t-1} + C_b + A n_t + \sigma_c \sqrt{\Delta \varepsilon_t}, \quad \varepsilon_t \sim \mathcal{N}(0,1) \]
\[ \vec{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi S(C_t)} + \eta \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0,\mathbf{I}) \]
\[ S(C_t) = \frac{C_t^{nd}}{C_t^{nd} + k_d} \]

What must change?

Observation distribution, \( P_{\theta}(\vec{Y}_t | C_t) \)

Proposal distribution, \( q(\cdot) \)
Incorporating nonlinear observations

Modifying the model

\[ C_t = \gamma C_{t-1} + C_b + An_t + \sigma_c \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim \mathcal{N}(0, 1) \]

\[ \vec{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi} S(C_t) + \eta \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, 1) \]

\[ S(C_t) = \frac{C_t^{nd}}{C_t^{nd} + k_d} \]

What must change?

- Observation distribution, \( P_\theta(\vec{Y}_t | C_t) \)
Incorporating nonlinear observations

Modifying the model

\[ C_t = \gamma C_{t-1} + C_b + A n_t + \sigma_c \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim \mathcal{N}(0, 1) \]
\[ \hat{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi S(C_t)} + \eta \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I) \]
\[ S(C_t) = \frac{C_t^{nd}}{C_t^{nd} + k_d} \]

What must change?

- Observation distribution, \( P_\theta(\hat{Y}_t | C_t) \)
- Proposal distribution, \( q(\cdot) \)
How do we modify the relevant distributions

Observation distribution

\[ P_\theta(\mathbf{Y}_t | C_t) = \mathcal{N} \left( \mathbf{Y}_t; \mu_7(S(C_t)), \sigma_7(S(C_t))^2 \right) \]

- We want a Gaussian function of \( C_t \)
- We don’t have it, but we compute a Laplacian approximation
How do we modify the relevant distributions

**Observation distribution**

\[ P_{\theta}(\vec{Y}_t | C_t) = \mathcal{N}(\vec{Y}_t; \mu_7(S(C_t)), \sigma_7(S(C_t))^2) \]

- We want a Gaussian function of \( C_t \)
- We don’t have it, but we compute a Laplacian approximation

**Laplace**

![Normalized Likelihood vs. [Ca^{2+}]](image)
Nonlinear observation

in silico data

- Simulated Spike Train
- Simulated Calcium
- Simulated Fluorescence
- Wiener Filter
- Nonlinear Observation PFS Spike Inference
- Nonlinear Observation PFS $[\text{Ca}^{2+}]$ Inference

Time (sec)
Nonlinear observation

in vitro bursts

- in vitro Fluorescence
- Wiener Filter
- Nonlinear Observation PFS Spike Inference
- Nonlinear Observation PFS $[\text{Ca}^{2+}]$ Inference

Time (sec)
Nonlinear observation

**in vitro spike train**

- **in vitro Fluorescence**
- **Wiener Filter**
- **Nonlinear Observation PFS Spike Inference**
- **Nonlinear Observation PFS [Ca^{2+}] Inference**

Time (sec)

Joshua Vogelstein (Johns Hopkins)
Intermittent observations

Generalizing the model

\[ \mathbf{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi S(C_t) + \eta \epsilon_t}, \quad \forall t/d \in \mathbb{Z} \]
Generalizing the model

\[ \bar{Y}_t = \alpha [S(C_t) + \beta] + \sqrt{\xi S(C_t) + \eta \epsilon_t}, \quad \forall t/d \in \mathbb{Z} \]

Generalizing the observation distribution

\[ P_\theta(\bar{Y}_t | C_t) = \begin{cases} 
\mathcal{N}(\alpha [S(C_t) + \beta], \xi S(C_t) + \eta) & \text{if } t/d \in \mathbb{Z} \\
1 & \text{otherwise} 
\end{cases} \]
One observation ahead sampler

Let $\nu$ be the time step of the next observation

$$q \left( H_t^{(i)} \right) = P_\theta \left( \tilde{Y}_\nu | H_t^{(i)} \right) P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)} \right)$$

- We can start at time $\nu$ and use the nonlinear observation sampler
- Then, we can recursively step backward, using the standard backward recursion
- Note, however, that the number of possible spike trains is $2^{\nu-t}$
- Thus, if $d \gg 1$, we have a mixture with too many components
Approximating $P_\theta (\vec{Y}_v | H^{(i)}_t)$

Reduce from $2^{v-t}$ to $v - t + 1$

Plug in our approximate observation distribution and sample $q (H^{(i)}_t) = \hat{P}_\theta (\vec{Y}_v | H^{(i)}_t) P_\theta (H^{(i)}_t | H^{(i)}_{t-1})$
Approximating $P_\theta (\vec{Y}_v | H_t^{(i)})$

Reduce from $2^{v-t}$ to $v - t + 1$

Plug in our approximate observation distribution and sample

$$q \left( H_t^{(i)} \right) = \hat{P}_\theta \left( \vec{Y}_v | H_t^{(i)} \right) P_\theta \left( H_t^{(i)} | H_{t-1}^{(i)} \right)$$
One observation ahead sampler improves performance

One observation ahead sampler

- Weighted Spike History
- Spike Train
- Calcium Concentration

Prior sampler

Inferred Distributions

Time (sec)
Calcium Concentration
Inferred Distributions

One observation ahead sampler

Particles
Inferred Distributions

Time (sec)
Calcium Concentration

Particles
Inferred Distributions

Time (sec)
Array of results varying noise and intermittency

One observation ahead sampler

Increasing Frame Rate

Increasing Observation Noise

Time (sec)
Incorporating stimulus and spike history dependence

Generalizing the model

\[ P(\theta(n_t)) = B(n_t; f(b + kx_t + \omega h_t)) \]

\[ h_t = \gamma h_{t-1} + n_{t-1} + \sigma h \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim N(0, 1) \]

Constraints on dynamics

- \( f(\cdot) \) must be convex and log-concave for likelihood to be concave (same as GLM)
- Dynamics on \( h_t \) must be Markovian to fit into SMC framework
Generalizing the model

\[ P_\theta(n_t) = \mathcal{B}(n_t; f(b + kx_t + \omega h_t)) \]

\[ h_t = \gamma_h h_{t-1} + n_{t-1} + \sigma_h \sqrt{\Delta \varepsilon_t}, \quad \varepsilon_t \sim \mathcal{N}(0, 1) \]
Incorporating stimulus and spike history dependence

Generalizing the model

\[ P_{\theta}(n_t) = B(n_t; f(b + kx_t + \omega h_t)) \]

\[ h_t = \gamma_h h_{t-1} + n_{t-1} + \sigma_h \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim \mathcal{N}(0, 1) \]

Constraints on dynamics

Constraints on dynamics:
- \( f(\cdot) \) must be convex and log-concave for likelihood to be concave (same as GLM)
- Dynamics on \( h_t \) must be Markovian to fit into SMC framework
Incorporating stimulus and spike history dependence

**Generalizing the model**

\[
P_\theta(n_t) = B(n_t; f(b + k x_t + \omega h_t))
\]

\[
h_t = \gamma_h h_{t-1} + n_{t-1} + \sigma_h \sqrt{\Delta \epsilon_t}, \quad \epsilon_t \sim N(0, 1)
\]

**Constraints on dynamics**

- \( f(\cdot) \) must be **convex and log-concave** for likelihood to be concave (same as GLM)
Incorporating stimulus and spike history dependence

Generalizing the model

\[
P_{\theta}(n_t) = B(n_t; f(b + k x_t + \omega h_t))
\]

\[
h_t = \gamma_h h_{t-1} + n_{t-1} + \sigma_h \sqrt{\Delta} \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, 1)
\]

Constraints on dynamics

- \( f(\cdot) \) must be **convex and log-concave** for likelihood to be concave (same as GLM)
- Dynamics on \( h_t \) must be **Markovian** to fit into SMC framework
Incorporating stimulus and spike history dependence

Modified transition distribution

\[ P_{\theta}(H_t|H_{t-1}) = P_{\theta}(C_t|C_{t-1}, n_t)P_{\theta}(n_t|h_t)P_{\theta}(h_t|h_{t-1}, n_{t-1}) \]

Modified sampler

1. Sample \( h_t \) using transition distribution
2. Sample \( n_t \) using one observation ahead sampler (with modified transition distribution)
3. Sample \( C_t \) using one observation ahead sampler (with modified transition distribution)
Schematic showing using GLM improves inference

Simulated data

- **External Stimulus**
- **Simulated Spike History**
- **Probability of Spiking**
- **Simulated Spike Train**
- **Simulated Calcium**
- **Simulated Fluorescence**
- **Superresolution PFS Spike Inference**
- **GLM PFS Spike Inference**

Time (sec): 0, 0.5, 1, 1.5

Joshua Vogelstein (Johns Hopkins)
State-Space Calcium Imaging
April 23, 2009
Using GLM improves inference in real data as well

**in vitro data**

- **Stimulus**
- **in vitro Spikes**
- **in vitro Fluorescence**
- **Wiener Filter**
- **Superresolution**
- **PFS Spike Inference**
- **GLM PFS Spike Inference**

Time (sec): 1 2 3 4 5
Advantages over FANSI

- Inference is better
## Summary of PFS results so far

### Advantages over FANSI

- Inference is better
- Naturally get errorbars on estimates
Summary of PFS results so far

Advantages over FANSI

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
Summary of PFS results so far

**Advantages** over FANSI

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
- Incorporate nonlinear observations more cleanly
- Can be generalized further

Slower
- More complicated
Summary of PFS results so far

**Advantages over FANSI**

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
- Incorporate nonlinear observations more cleanly
- Incorporate intermittent sampling more cleanly
Summary of PFS results so far

### Advantages over FANSI

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
- Incorporate nonlinear observations more cleanly
- Incorporate intermittent sampling more cleanly
- Incorporate spike history dependence more cleanly
Summary of PFS results so far

**Advantages over FANSI**

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
- Incorporate nonlinear observations more cleanly
- Incorporate intermittent sampling more cleanly
- Incorporate spike history dependence more cleanly
- Can be generalized further
Summary of PFS results so far

**Advantages** over FANSI

- Inference is better
- Naturally get errorbars on estimates
- Parameter estimation is better
- Incorporate nonlinear observations more cleanly
- Incorporate intermittent sampling more cleanly
- Incorporate spike history dependence more cleanly
- Can be generalized further

**Disadvantages** relative to FANSI

- Slower
- More complicated
Further generalizations

PFS can be generalized to incorporate

- Slow rise time
- Poisson (instead of Bernoulli) spiking
- Background drift
- Poisson observations
- Cross-coupling between neurons
Outline

1. Introduction
2. General Methods
3. Simplifying our model
4. Fast non-negative spike inference (FANSI)
5. Particle-filter-smoother (PFS) spike inference
6. PFS results
7. Concluding thoughts
## Two algorithms for inferring spikes from calcium movies

- **FANSI** is fast, can be performed **online**
- **FANSI** outperforms Wiener filter, so should be the **default** deconvolution algorithm
- **FANSI** can **initialize** parameters for **PFS**
- **PFS** is slower, but still fast (about **real time**)
- **PFS** further refines results, by **sampling**
- **PFS** can incorporate **nonlinear/intermittent** observations
- **PFS** can incorporate spike history terms (including **cross-coupling terms**)

---

**Joshua Vogelstein (Johns Hopkins)**

State-Space Calcium Imaging

April 23, 2009
Acknowledgments

- Financial support: Eric D. Young
- Theoretical support: Liam Paninski’s group (Baktash Babadi)
- in vitro support: Rafael Yuste’s group (Bendon O. Watson, Adam M. Packer)
- in vivo support: Karel Svoboda’s group (Takashi Sato)
- Emotional support: family, friends, the earth, the universe, etc.