Prediction of a Connectome Based on Fluorescence Data

Guy W. Cole, Isaiah S. Morales, Austin Stone

May 7, 2014

Outline

Kaggle Challenge

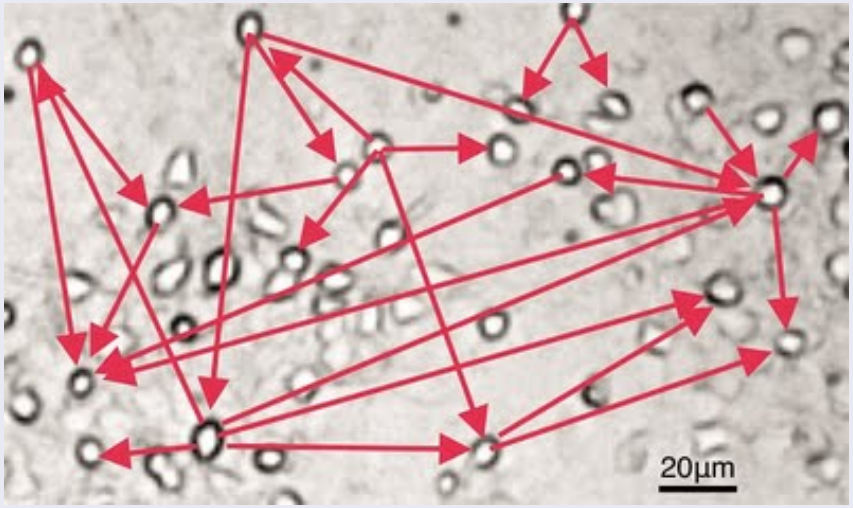
Our challenge

Kaggle.com is a website that sponsors machine learning challenges. A current challenge (with a 3000 dollar award) is to provide the most accurate connectome of a population of neurons given every neuron's location and every neuron's fluorescence data over a period of an hour.

Challenge Description

The goal

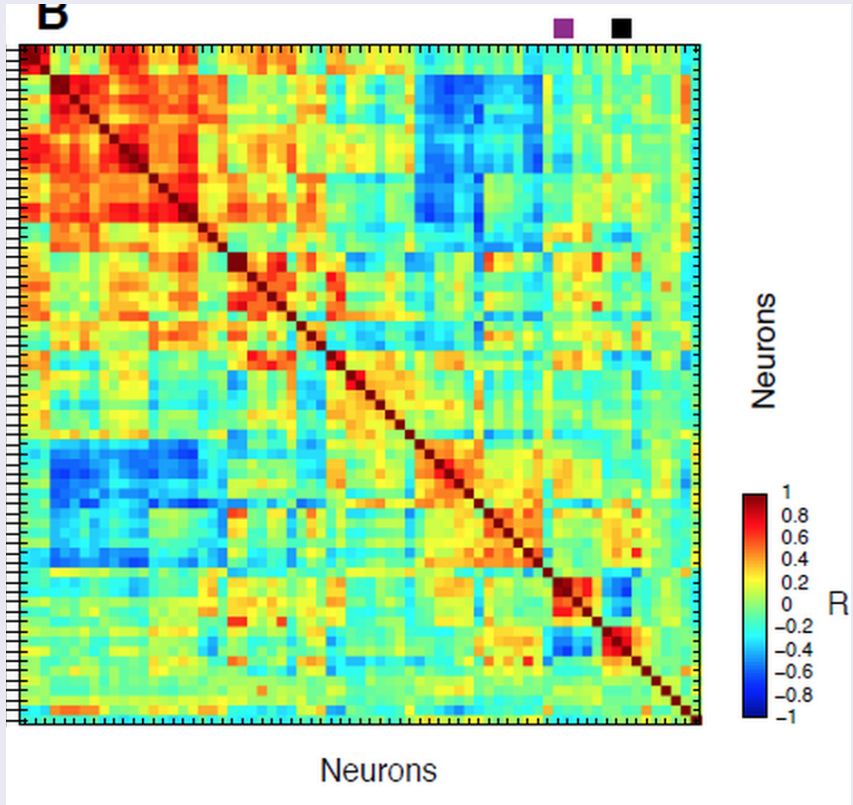
- It is easy to visually locate all neurons on a petri dish, but extremely intractable to determine which neurons have axons connecting them to other neurons.
- Thus, a current outstanding problem in neuroscience is to infer the connectome indirectly.



What information can we use to infer the connectome?

Inferring the Connectome

- In wet labs, we can monitor the activity of neurons by placing a fluorescence meter on each neuron.
- The hope is to be able to reconstruct the connectome using correlations in this fluorescence data.



Stetter 2012

- To test the accuracy of a predicted connectome, one needs to know the actual connectome.
- Since determining the actual connectome is such a hard problem and consequently very few actual network connectomes are known, the Kaggle challenge uses data simulated from a virtual population of neurons in which the connectome is known via the method outlined in Stetter, 2012[1]

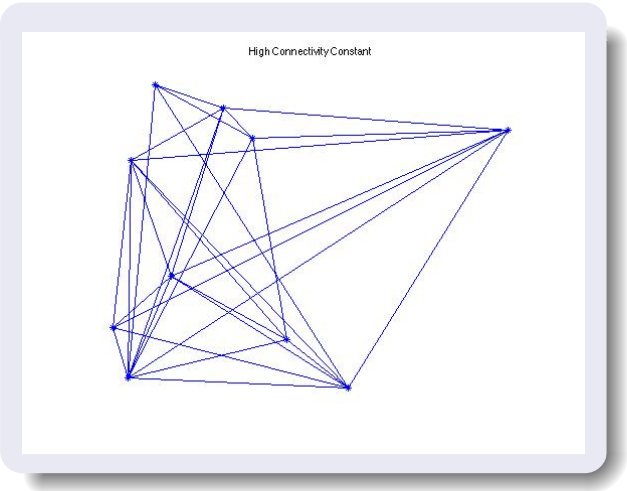
[1] Stetter O, Battaglia D, Soriano J, Geisel T (2012) Model-Free Reconstruction of Excitatory Neuronal Connectivity from Calcium Imaging Signals. PLoS Comput Biol 8(8): e1002653. doi:10.1371/journal.pcbi.1002653

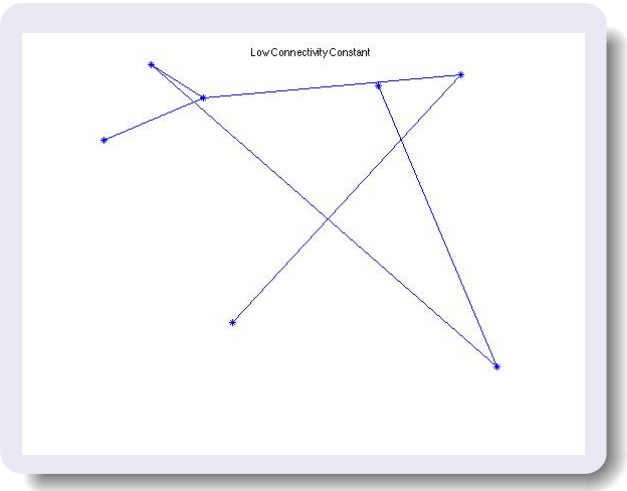
Latent Variables taken from the Stetter Paper

Connectivity Constant, p

The setter paper defines several different latent variables that we incorporated into our simulation.

- Connectivity constant, p, is the chance that two arbitrary neurons are connected to each other.
- This plays into our model in that it affects the prior distribution of what we believe the connectome is.



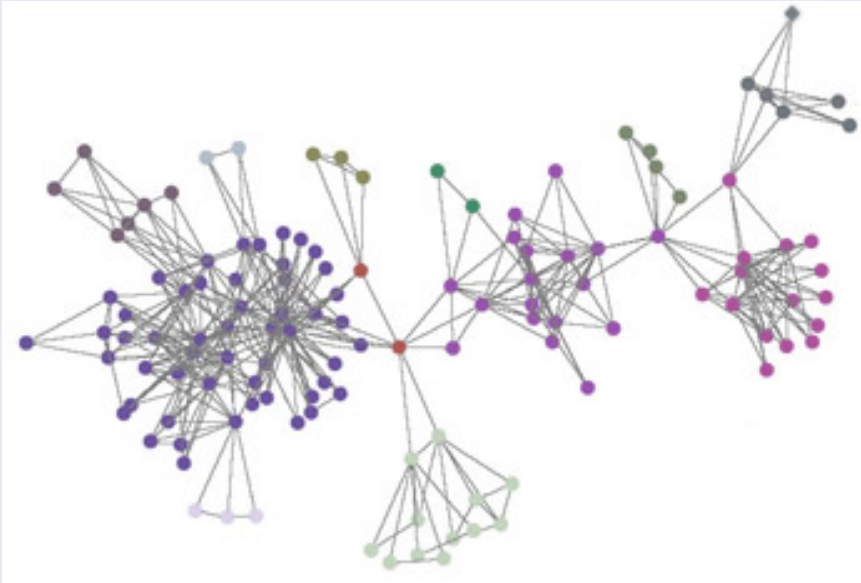


Latent Variables taken from the Stetter Paper

Clustering Coefficient

$E_v/(K_v(K_v-1)/2)$

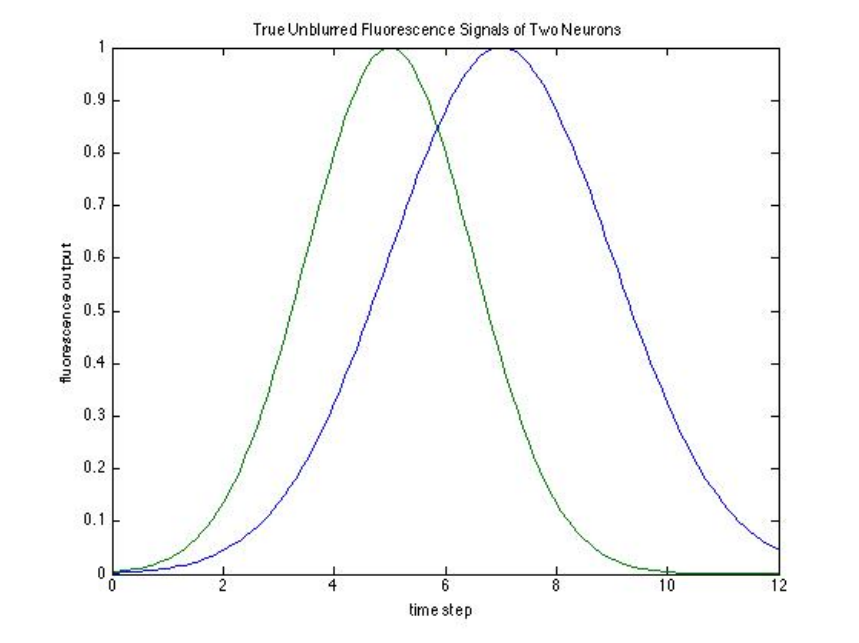
- The clustering coefficient is a measure of how tightly the graph "clusters."
- This plays into our model in that it affects the prior distribution of what we believe the connectome is.



Problems with Fluorescence Data

Fluorescence Blurring

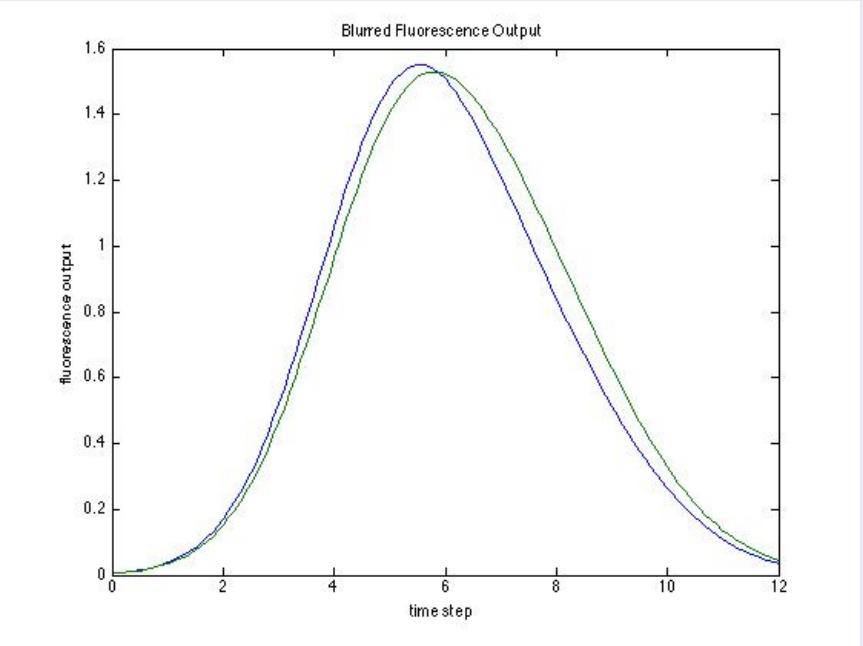
A neuron's fluorescence values effect the fluorescence of other neurons even if they aren't connected due light scattering effects. To the right is an example of an unblurred, "true" fluorescence output from two different neurons.



Problems with Fluorescence Data

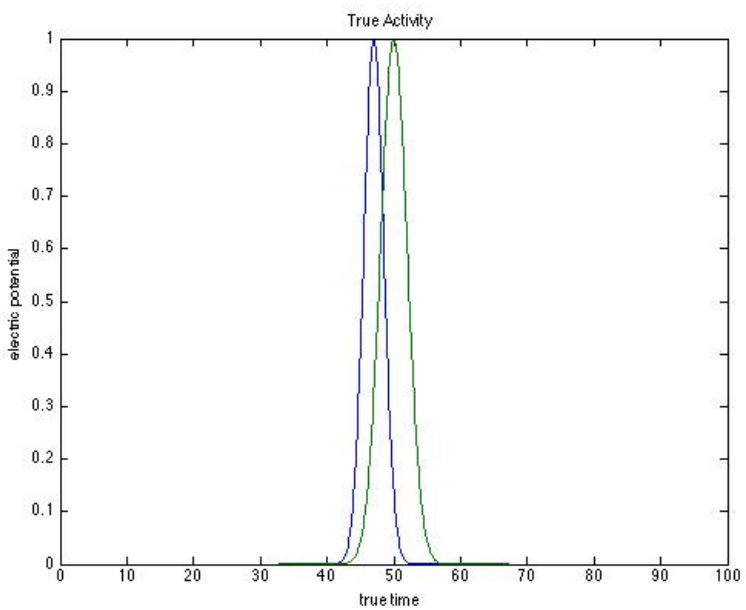
Fluorescence Blurring

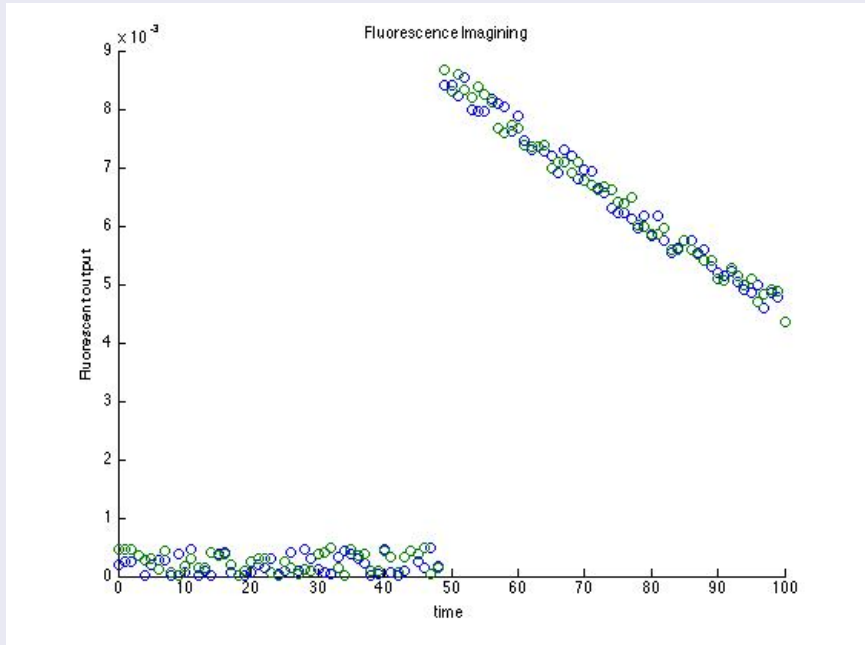
Blurring linearly combines the true fluorescence value of a neuron with the fluorescence values of its neighbors. This causes nearby neural fluorescence data to look more similar than it should.



Problems with Fluorescence Data

The fluorescence data is sampled in intervals of 20 ms and decays very slowly. The rate of sampling is too slow to infer exactly when spikes happen, and which spikes caused other spikes.





Data simulation methods

- Stetter 2012 devised a system of differential equations to simulate the fluorescence of interconnected neurons based on a connectome.
- A connectome was simualted randomly, then connections were altered systematically to create the desired clustering coefficient.
- Fluorescence values were then blurred to reflect actual systems.
- The blurred data than had Gaussian noise added, resulting in reported fluorescence values outside [0,1].

General Outline

- Set up a Bayesian hierarchical model of the (unspecified) system of differential equations.
- Use penalization (LASSO and/or Ridge) and "spike and slab" priors to differentiate connection status.
- Use Markov Chain Monte Carlo (MCMC) (and possibly other methods) to find optimal parameter solutions to the model.
- Report back MAP and Expectation settings of the Connectome.

Statistical Model

$$C_{ij} \sim Bernoulli(\rho) \quad (1)$$

$$\beta_{ij} \sim N(0, (\tau_1 C_{ij} + \tau_0(1 - C_{ij}))I) \quad (2)$$

$$F_{i,t} = N(\begin{pmatrix} F_{;t-1} \\ F_{i,t-1}F_{;t-1} \end{pmatrix}^T \beta_i + \epsilon_{i,t}^{\mathbb{M}} \quad (3)$$

$$F_{i,t}^{bl} = A(\alpha, \lambda)F + \epsilon_{i,t}^{\mathbb{B}} \quad (4)$$

$$A_{ij} = \alpha e^{-\frac{1}{2}\frac{||x_i - x_j||_2^2}{\lambda^2}} + (1 - \alpha)\delta_{i=j} \quad (5)$$

Update Rules: C

$$p(C_{ij} = 1|\beta_j) = \frac{\rho N(\beta_{ij}|0, \tau_1)}{\rho N(\beta_{ij}|0, \tau_1) + (1-\rho)N(\beta_{ij}|0, \tau_0)} \quad (6)$$

$$odds(C_{ij} = 1|\beta_j) = \frac{p}{1-p} = \frac{\rho}{1-\rho}\frac{\tau_1}{\tau_0}e^{-\frac{1}{2}||\beta_{ij}||_2^2(\tau_1-\tau_0)} \quad (7)$$

$$\text{Stochastic} : \text{Sample } C_{ij} \sim p(C_{ij} = 1|\beta_j) \quad (8)$$

$$\text{MLE} : \text{Set } C_{ij} = 1_{p(C_{ij}=1|\beta_j)>\frac{1}{2}} \quad (9)$$

Update Rules: β

$$\hat{\beta}_i | C, F \sim N\left((\frac{1}{\sigma^2} X^T X + \tau_{C_i} I)^{-1} \frac{1}{\sigma^2} X^T F, (\frac{1}{\sigma^2} X^T X + \tau_{C_i} I)^{-1} \right)$$

- Can easily either sample $\hat{\beta}$ or just use expectation/MLE (center of the Gaussian)
- Effect of adding $\tau_{C_i} I$ is that filters of disconnected neurons end up being very close to 0, connected neurons end up close to unpenalized MLE. (That is, more shrinkage for disconnected neurons having any effect.)
- $\sigma^2 \tau = \frac{\sigma_{li}^2}{\sigma_{pr}^2} \approx$ signal-to-noise ratio
- We place a Gamma prior on $1/\sigma^2$ and update using Gibbs sampling of the Gamma posterior.

Update Rules: α, λ

- No easy way to optimize or conditionally sample.
- Return to random-walk metropolis: $\begin{matrix}\alpha' \\ \lambda'\end{matrix} = \begin{matrix}\alpha \\ \lambda\end{matrix} + \epsilon_{2x1}$
- We sample a series (of arbitrary size) or α', λ' pairs centered at α, λ and choose either:
- MLE: $\arg\max_{\alpha',\lambda'} p(F^{bl}|A(\alpha', \lambda')\hat{F}, \sigma^2)$
- Expectation: $(\alpha', \lambda') = \sum(\alpha', \lambda')p(F^{bl}|A(\alpha', \lambda')\hat{F}, \sigma^2)$
- Sample: $p(\alpha', \lambda') \propto \sum(\alpha', \lambda')p(F^{bl}|A(\alpha', \lambda')\hat{F}, \sigma^2)$

How we measured it

- Precision: $\frac{\text{Correct Connections}}{\text{Total Predicted Connections}} = \frac{TP}{FP+TP}$
- Recall: $\frac{\text{Correct Connections}}{\text{Total True Connections}} = \frac{TP}{FN+TP}$
- F1 score: $\frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}}$

Results: 100 neuron network

- Note, this network has 100 neurons, and a total of 1050 (10.5%) non-diagonal connections. We performed stochastic sampling with $\rho = 0.105$.

Method	Pre	Rec	F1
All connections	0.115	1.000	0.206
Random	0.115	0.182	0.141
Result (A)	0.130	0.376	0.193
Result (B)	0.325	0.125	0.181

- Result: We're beating random guessing, but not by a lot.

Conclusions

- A 2nd-order analysis that looks only 1 timestep back is doing a very poor job.
- Separating the problems of deblurring and estimating the connectome would be valuable.
- Higher-order and longer timesteps is trivial to add analytically, but computationally infeasible.
- Algorithms are highly parallelizable (note $\hat{\beta}_i \perp \hat{\beta}_i$) so may be a good approach in need of serious computing power.

Questions?

Questions?