Let

, where .

Use AR(1) for

**Constraints:**

Easy one: Just let and be diagonal. (Now is free, yay!)

Maybe just let be upper/lower-trangular is enough? Why?

Letter

Description automatically generated with low confidence

Because the number of constraint parameters is , which is equivalent to the number of constraints in traditional constraint for **normal factor model**. **Try later.**

Using less constraints will lead to a faster convergence.

After doing Schur decomposition for

Therefore,

**Key derivation for clustering: Marginal distribution of .**

I no longer need to use auxiliary parameters.

To save notations, denote as , as , as and as . Further, denote . Then, I need to evaluate the marginal likelihood

**Way 1: use normal approximation (the Laplace approximation)**

That is,

, where is the Poisson density with mean .

In the Laplace approximation, , i.e., MAP and is the inverse negative hessian at . Since is usually super large, I can just use MLE to replace MAP, by doing Poisson regression. However, there are some drawbacks:

1. This is quite painful. If there are neurons and clusters at current step, I need to run Newton-Raphson times! (Maybe I can do OLS instead, i.e., )
2. It’s not quite robust, especially when generating a new set of parameters from prior. The inverse-hessian might be close to singular.

Another easy way is just set (evaluate the approximation at prior). Well, the approximation should be super bad…

**Way 2: approximate by gamma distribution**

Assume conditional independence: . Denote . Then, since

Then approximate as

So,

, where and

**Simulation 1: known label**

Here I use , but since now I include , the effective . So, there’s one more factor than the real



Trace for X



**ALERT: All the following clustering results are problematic, since I just found there are a lot of bugs in the normal approximation… Just show for reference…**

**Simulation 2: same setting as in simulation 1 but remove the label.**



OK, perfect. Let’s see what happens to a nastier simulation.

**Simulation 3: as in simulation 3 in the slides**

Spikes



The trace of clusters (use normal approximation by OLS):



Wow… Looks quite nasty, but when we check the clustering results for each neuron (at iteration 50 & 100), things are more meaningful.

|  |  |
| --- | --- |
| Iteration 50 | Iteration 100 (end of the chain) |
|  |  |

How about just use Poisson regression to estimate ? Better…

|  |  |
| --- | --- |
| trace | Iteration 100 (end of the chain) |
|  |  |

**Although, the results seem (just seem) not as good as previous, but I strongly prefer the current model:**

1. The new model is cleaner & more elegant. Moreover, it has a better interpretation with less constraints, i.e., mean part () + variance part (). The value of just controls how detailed we want to model the variance.
2. The convergence & fitting in terms of mean firing rate is (**so**) much faster, especially when doing the clustering.
3. The seemly perfectness (, **which is fake**) in the previous method is because the convergence of is slow, and this gives a huge space for to tune itself… After convergence, the auxiliary will never beat , since is observed for . So, after a few iterations, the algorithm will just stop searching. It will even stop at some totally irrational results sometimes.
4. The previous method is not robust… When change the variance of or change , the results will change a lot.
5. Maybe current imperfect performance comes from bad approximation of marginal likelihood.
6. The new method may just do a more detailed clustering: not only consider the waveform but also take amplitude into consideration. Maybe this is not true after I fix the approximation to marginal likelihood.

**Choose**

The number of latent factor should be optimized. Choose by using the shrinkage prior and adaptive Gibbs sampler. See if I can easily do that.