A comparison of neural networks for approximate Bayesian inference

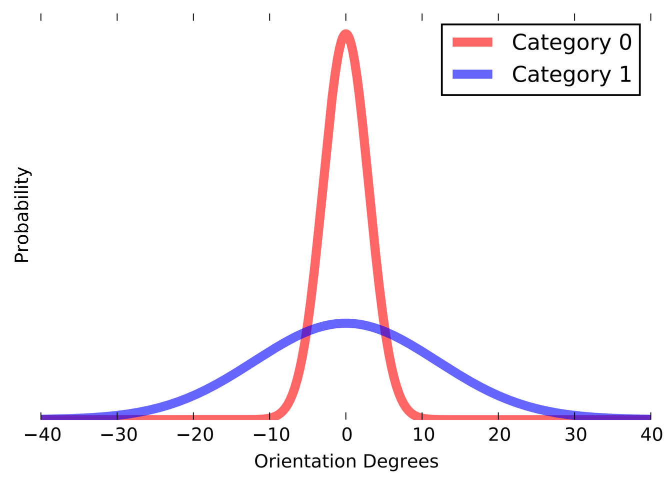
David Halpern, Emin Orhan and Wei Ji Ma

NYU

Lots of behavioral evidence that the brain represents and computes with uncertainty
But how do neurons do this?

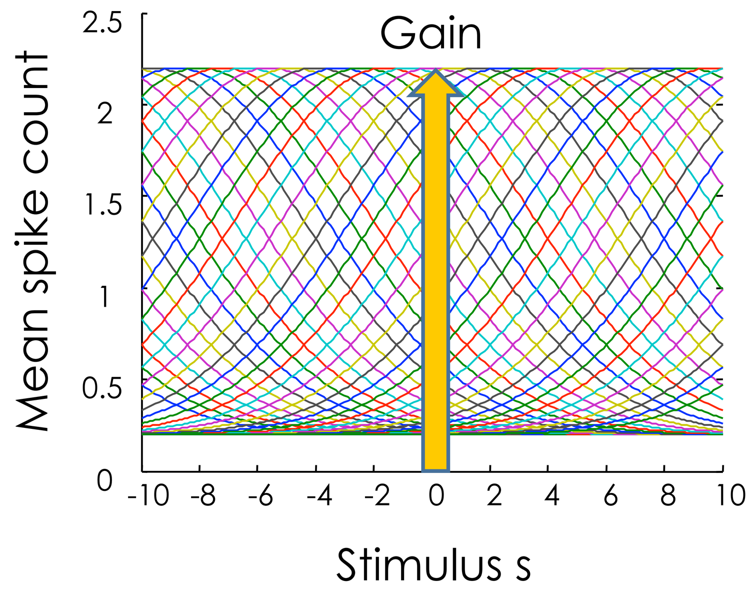
Task

Categorize an orientation drawn from one of two overlapping distributions
Optimality requires subjects to keep track of uncertainty **from trial to trial**
People do that to some extent (Qamar et al. 2013)



Neural encoding model

Neurons have Gaussian tuning curves around a preferred orientation $f_1(s),\ldots,f_N(s)$ and Poisson variability



Optimal inference in model

$$P(C,s|\mathbf{r}) = \left(\prod_i \frac{f_i(s)^{r_i}}{r_i!}\right)\sqrt{\frac{\tau_C}{2\pi}}e^{\frac{-\tau_C s^2}{2}}$$

C is the category, s is the stimulus
r is the neural response and τ_C is the precision of the category distribution

Variational Inference

Assume that the posterior is factorizable, i.e.

$$P(C,s|\mathbf{r}) \approx Q(C|\mathbf{r})Q(s|\mathbf{r})$$

We then get variational distributions

$$Q(C|\mathbf{r}) \sim \mathrm{Bernoulli}(p) \quad Q(s|\mathbf{r}) \sim \mathcal{N}(\eta,\tau)$$

Interpret as three-neuron dynamical system (as in Beck et al. 2012)

$$\frac{dp}{dt} = 1 - p\left(1+\sqrt{\frac{\tau_0}{\tau_1}}e^{-\frac{((\frac{\eta}{\tau})^2+\frac{1}{\tau})\Delta\tau}{2}}\right)$$

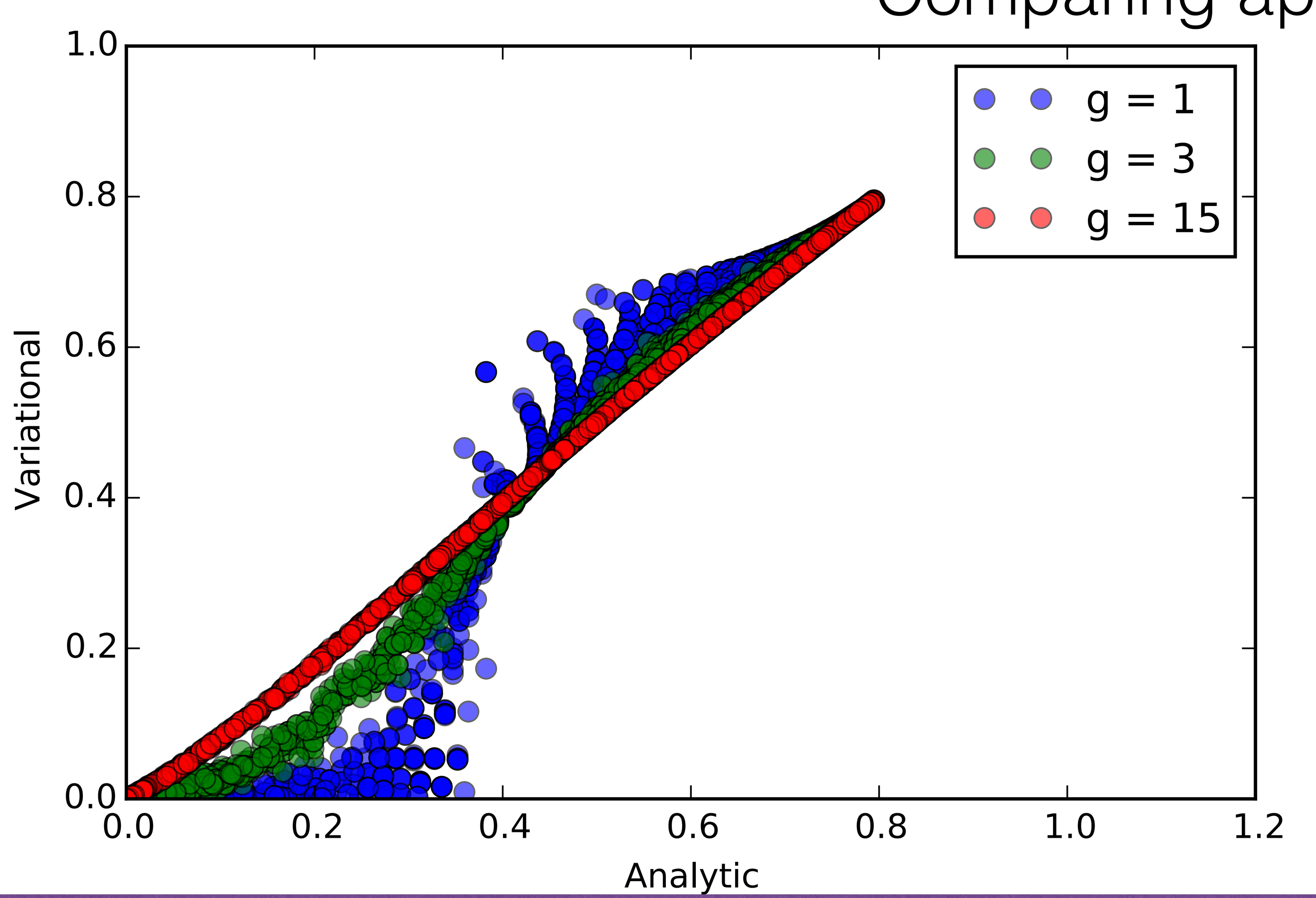
$$\eta = \sum_i \frac{r_i}{\sigma_{\mathrm{tc}}^2}s_i^{\mathrm{pref}}$$

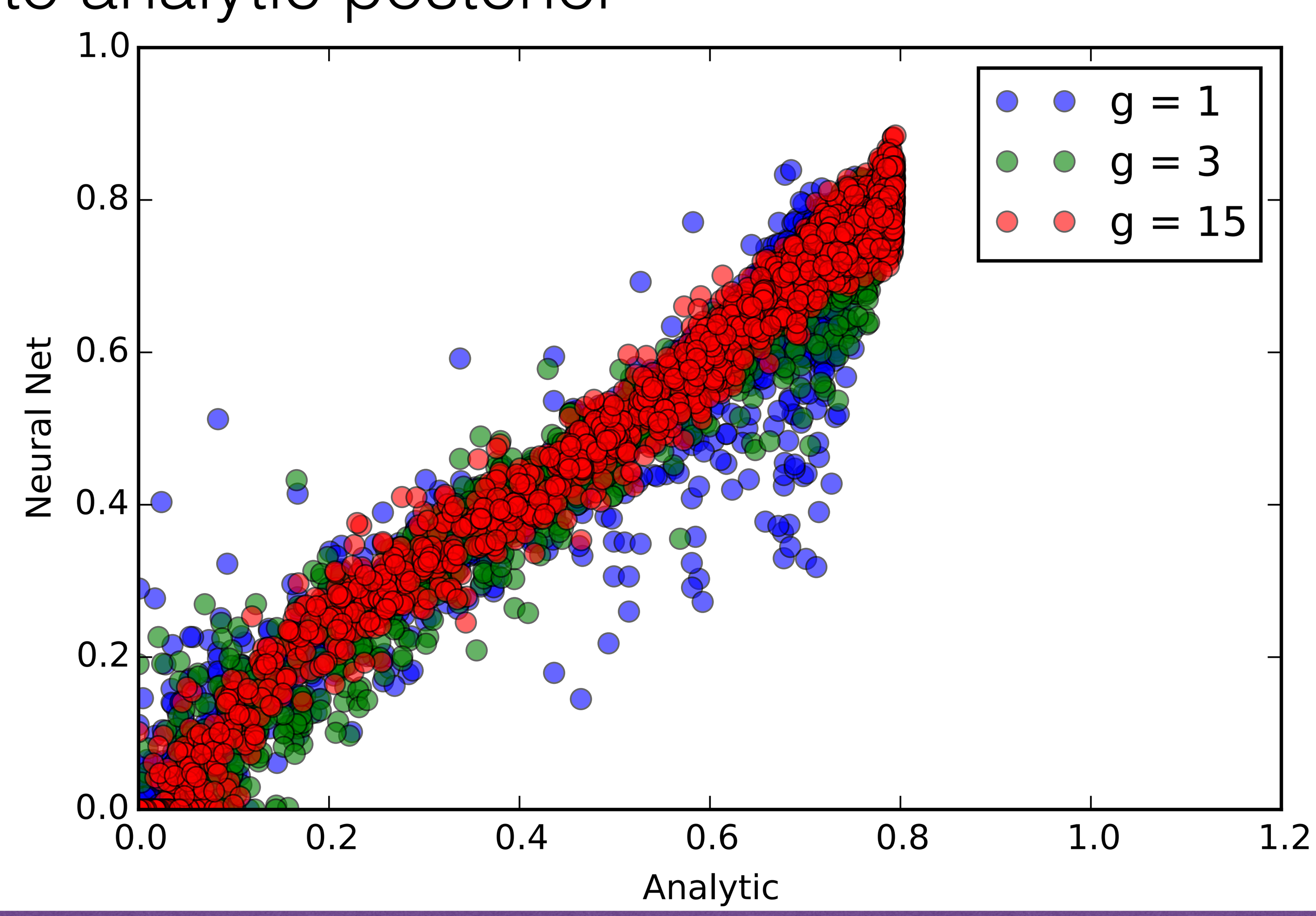
$$\frac{d\tau}{dt} = -\tau + \sum_i \frac{r_i}{\sigma_{\mathrm{tc}}^2} + (\tau_0 - p\Delta\tau)$$

Neural Network

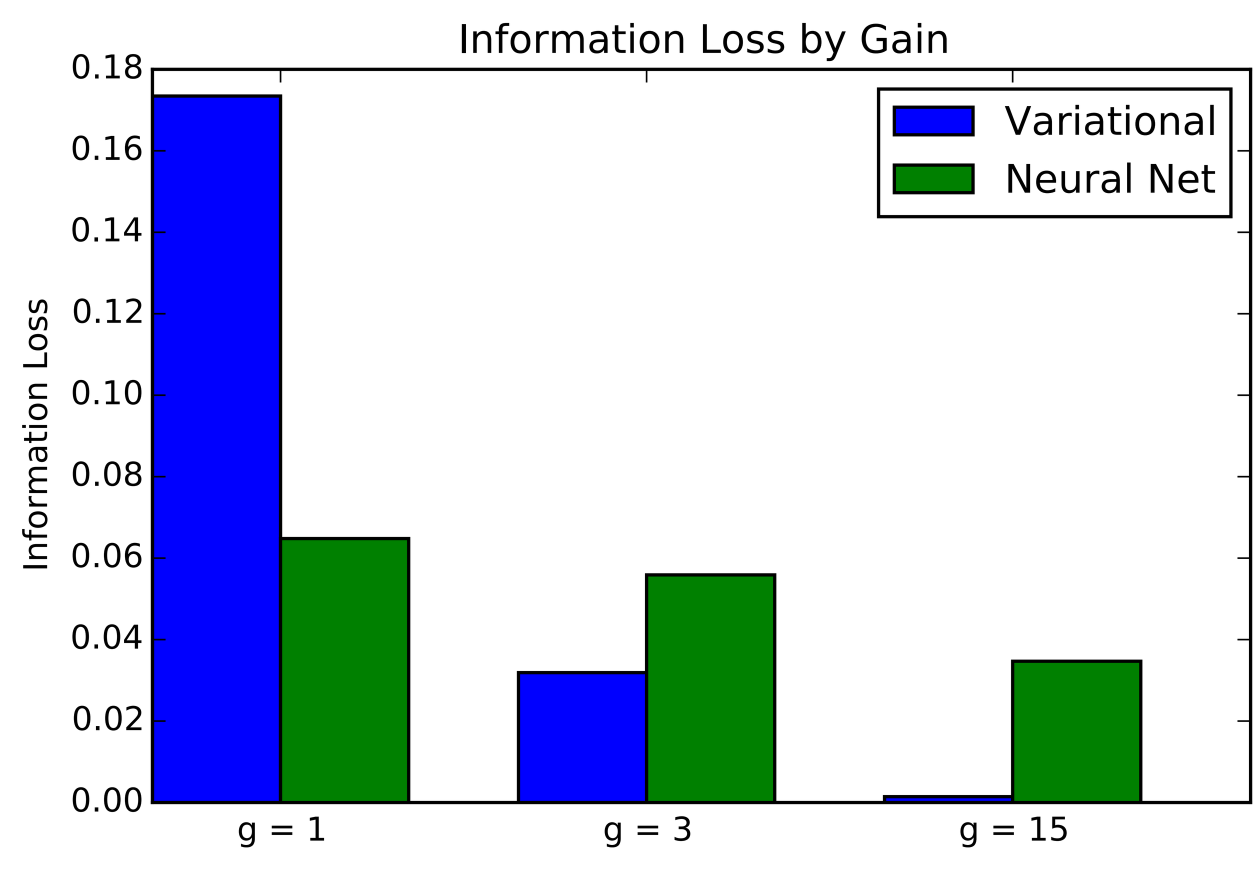
- One hidden layer
- Fully connected with Rectified Linear Unit activations
- 3 x (number of updates to convergence) hidden units in order to match the variational network
- Trained with 3000 randomly generated samples using stochastic gradient descent with mean squared error loss
- 100 epochs

Comparing approximations to analytic posterior





Conclusions



At higher gains, variational inference can be more efficient than a multilayer neural net

Future work:
Compare to a generic recurrent neural network
Use KL divergence as loss
Train on class labels