Of course the main interest of an EFN is to learn intractable exponential families. The normal family is the ubiquitous prior for real valued parameters, but it does not match well with the nonnegativity requirements of the intensity measure required of certain distributions, most notably the Poisson. Log Gaussian Cox Processes have been used numerous times in machine learning, and all have required attention to approximate inference in this fundamentally nonconjugate model; furthermore, very many of these examples have been used to analyze the latent firing intensity of neural spike train data \cite{cunningham2008fast,cunningham2008inferring,adams2009tractable,gao2016linear}.

Here, we trained an EFN to learn the 20-dimensional log-normal Poisson posterior inference family. This gives us a model of the posterior distribution for a given prior covariance, and some chosen spiking responses. We demonstrate the utility of such a model on responses of neurons in primary visual cortex of anesthetized macaques to drift grating stimuli at 6.25 Hz \cite{smith2008spatial}. We can compare the posterior distribution learned with variational inference (red) for a given neuron’s response, to the posterior distribution we get with immediate look-up after training an EFN (blue) (Fig. 4A-B). These posteriors are very similar, and neither fits the data better than the other.

Training an EFN understandably takes more time than an NF (Fig. 4C), but once the EFN is trained we have immediate posterior inference lookup. If we have a target level of approximation (ELBO target) we can determine when it is faster to get posterior inference on a number of datasets by training an EFN and then using the immediate lookup feature or by running variational inference independently for each distribution. By computing the amount of computational time it takes to reach the ELBO target on average for both EFN and NF, and then counting how many datasets it would take to learn with NF before eclipsing the training time for the EFN. This results in a decision boundary (Fig. 4D), where an EFN is more computationally efficient for running posterior inference, and we have infinite computational savings for each additional dataset.