Supplementary Material

Bayesian active learning of neural firing rate maps with transformed Gaussian process priors
Mijung Park, J. Patrick Weller, Gregory D. Horwitz, & Jonathan W. Pillow (2014)
Neural Computation 26(8):1519-1541.

Appendix A

Efficient evidence optimization for hyperparameters

For efficient optimization of hyper parameters, we decompose the posterior moments $(\mathbf{f_{map}}, \Sigma)$ into terms that depend on ϕ and terms that do not via a Gaussian approximation to the likelihood. The logic here is that a Gaussian posterior and prior imply a likelihood function proportional to a Gaussian, which in turn allows prior and posterior moments to be computed analytically for each ϕ. This trick is similar to that of the EP algorithm [1]: we divide a Gaussian component out of the Gaussian posterior and approximate the remainder as Gaussian. The resulting moments are $H = \Sigma^{-1} - K^{-1}$ for the likelihood inverse-covariance (which is the Hessian of log-likelihood), and $\mathbf{m} = H^{-1}(\Sigma^{-1}\mathbf{f_{map}} - K^{-1}\boldsymbol{\mu}_\mathbf{f})$ for the likelihood mean, which comes from the standard formula for the product of two Gaussians.

The algorithm for evidence optimization proceeds as follows: **(1)** given the current hyperparameters ϕ_i, numerically maximize the posterior and form the Laplace approximation $\mathcal{N}(\mathbf{f_{map}}_i, \Sigma_i)$; **(2)** compute the Gaussian "potential" $\mathcal{N}(\mathbf{m}_i, H_i)$ underlying the likelihood, given the current values of $(\mathbf{f_{map}}_i, \Sigma_i, \phi_i)$, as described above; **(3)** Find ϕ_{i+1} by maximizing the log-evidence, which is:

$$\mathcal{E}(\phi) = \mathbf{r}^T \log(g(\mathbf{f_{map}})) - \mathbf{1}^T g(\mathbf{f_{map}}) - \frac{1}{2}\log|KH_i + I| - \frac{1}{2}(\mathbf{f_{map}} - \boldsymbol{\mu}_\mathbf{f})^T K^{-1}(\mathbf{f_{map}} - \boldsymbol{\mu}_\mathbf{f}), \tag{1}$$

where $\mathbf{f_{map}}$ and Σ are updated using H_i and $\mathbf{m}_i$ obtained in step **(2)**, i.e. $\mathbf{f_{map}} = \Sigma(H_i\mathbf{m}_i + K^{-1}\boldsymbol{\mu}_\mathbf{f})$ and $\Sigma = (H_i + K^{-1})^{-1}$. Note that this significantly expedites evidence optimization since we do not have to numerically optimize $\mathbf{f_{map}}$ for each ϕ.

Appendix B

Gauss-Hermite quadrature to compute Gaussian integrals

Gauss-Hermite quadrature is to approximate the value of integrals:

$$\int \exp(-x^2)\; t(x)dx \approx \sum_{i=1}^{n} w_i\; t(x_i), \tag{2}$$

where n is the number of sample points used and x_i are the roots of the Hermite polynomial $H_n(x_i)$, which is given by

$$H_n(x_i) = (-1)^n \exp(x_i^2)\frac{\partial^n}{\partial x_i^n}\exp(-x_i^2), \tag{3}$$

and the weights are defined by

$$w_i = \frac{2^{n-1}n!\sqrt{\pi}}{n^2[H_{n-1}(x_i)]^2}. \tag{4}$$

Gauss-Hermite quadrature for a Gaussian random variable $f \sim \mathcal{N}(\mu, \sigma^2)$ is given by,

$$\mathbb{E}[t(f)] = \int \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(f-\mu)^2}{2\sigma^2}\right)\; t(f)\; df \approx \frac{1}{\sqrt{\pi}}\sum_{i=1}^{n} w_i\; t(\mu + \sqrt{2}\sigma f_i). \tag{5}$$

We can use Gauss-Hermite quadrature to transform the variance of $f \sim \mathcal{N}(\mu, \sigma^2)$ into the variance of $\lambda = g(f)$ as below:

$$\begin{aligned}
\mathbb{E}[\lambda] &= \int \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(f-\mu)^2}{2\sigma^2}\right)\; g(f)\; df \approx \frac{1}{\sqrt{\pi}}\sum_{i=1}^{n} w_i\; g(\mu + \sqrt{2}\sigma f_i), \\
\mathbb{E}[\lambda^2] &= \int \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(f-\mu)^2}{2\sigma^2}\right)\; g^2(f)\; df \approx \frac{1}{\sqrt{\pi}}\sum_{i=1}^{n} w_i\; g^2(\mu + \sqrt{2}\sigma f_i), \\
\mathbb{V}[\lambda] &= \mathbb{E}^2[\lambda] - \mathbb{E}[\lambda^2].
\end{aligned} \tag{6}$$

Appendix C

Pseudocode for varmin learning

This pseudocode selects a new stimulus for the next trial given the current posterior mean and covariance of f. In pseudocode, we use the Gauss-Hermite quadrature for the transformation of the variance from f to λ, which can be done using a pre-computed lookup table.

```
input: posterior mean and covariance $\mathcal{N}(\mathbf{f}^*|\boldsymbol{\mu}_t, \Lambda_t)$,
compute total posterior variance of $\lambda$ and choose a next stimulus
for i=1:M do        grid points $\{\mathbf{x}_i^*\}_{i=1}^M$ for representing the posterior over $f$
for j=1:N do        candidate points $\{\mathbf{x}'_j\}_{j=1}^N$
$\Pi(i,j) := \sigma_t^2(i) - \frac{\mathcal{J}_{\boldsymbol{\mu}'(j)}\Lambda_t^2(i,j)}{1+\mathcal{J}_{\boldsymbol{\mu}'(j)}\sigma_t^2(j)}$,        update the posterior variance of $f$
end for
end for
$\mathbb{V}(\lambda_i|\mathcal{D}_t, r', \mathbf{x}'_j) :=$ GHQuad$(\boldsymbol{\mu}_t(i), \Pi(i,j))$,    using eq. 6
$\mathbf{x}_{t+1} = \arg\min_{\{\mathbf{x}'_j\}_{j=1}^N} \sum_{i=1}^M \mathbb{V}(\lambda_i|\mathcal{D}_t, r', \mathbf{x}'_j)$,    select a stimulus $\mathbf{x}_{t+1}$
return a new stimulus to present $\mathbf{x}_{t+1}$ at time $t+1$.
```

References

[1] T. P. Minka. Expectation propagation for approximate bayesian inference. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.