Appendix A

Efficient evidence optimization for hyperparameters

For efficient optimization of hyper parameters, we decompose the posterior moments $(f_{\text{map}}, \Sigma)$ into terms that depend on $\phi$ 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 $\phi$. 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 $m = H^{-1}(\Sigma^{-1}f_{\text{map}} - K^{-1}\mu_f)$ 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 $\phi_i$, numerically maximize the posterior and form the Laplace approximation $\mathcal{N}(f_{\text{map}}, \Sigma_i)$; (2) compute the Gaussian “potential” $\mathcal{N}(m_i, H_i)$ underlying the likelihood, given the current values of $(f_{\text{map}}, \Sigma_i, \phi_i)$, as described above; (3) Find $\phi_{i+1}$ by maximizing the log-evidence, which is:

$$
E(\phi) = r^T \log(g(f_{\text{map}})) - 1^T g(f_{\text{map}}) - \frac{1}{2} \log |KH_i + I| - \frac{1}{2} (f_{\text{map}} - \mu_f)^T K^{-1} (f_{\text{map}} - \mu_f).
$$

(1)
where $f_{\text{map}}$ and $\Sigma$ are updated using $H_i$ and $m_i$ obtained in step (2), i.e. $f_{\text{map}} = \Sigma(H_i m_i + K^{-1} \mu _t)$ and $\Sigma = (H_i + K^{-1})^{-1}$. Note that this significantly expedites evidence optimization since we do not have to numerically optimize $f_{\text{map}}$ for each $\phi$.

Appendix B

Gauss-Hermite quadrature to compute Gaussian integrals

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

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

(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

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

(3)

and the weights are defined by

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

(4)

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

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

(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:

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

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

$$\mathbb{V}[\lambda] = \mathbb{E}^2[\lambda] - \mathbb{E}[\lambda^2].$$

(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 $\lambda$, which can be done using a pre-computed lookup table.

**input:** posterior mean and covariance $\mathcal{N}(f^*|\mu_t, \Lambda_t)$,

compute total posterior variance of $\lambda$ and choose a next stimulus

for $i=1:M$ do
  grid points $\{x^*_i\}_{i=1}^M$ for representing the posterior over $f$

for $j=1:N$ do
  candidate points $\{x'_j\}_{j=1}^N$

$\Pi(i,j) := \sigma^2_t(i) - \frac{J\mu'(i)\Lambda^2_t(i,j)}{1 + J\mu'(i)\sigma^2_t(j)}$, update the posterior variance of $f$

end for

$\nabla(\lambda|D_t, r', x'_j) := \text{GHQuad}(\mu_t(i), \Pi(i,j))$, using eq. 6

$x_{t+1} = \arg\min\{x'_j\}_{j=1}^N \sum_{i=1}^M \nabla(\lambda|D_t, r', x'_j)$, select a stimulus $x_{t+1}$

return a new stimulus to present $x_{t+1}$ at time $t + 1$.

References