Efficient Approximation of Covariance Matrix Inverse for Solving the Gaussian Process Maximum Likelihood Problem

Daniil Kononenko

June 1 2012

Table of contests

Table of contests

Problem statement

- Unknown function $f(x)$, $x \in R^n$, $f \in R$.
- Training set $\left\{(x_i, y_i = y(x_i))\right\}_{i=1}^{N}$.
- Observations $y(x_i) = f(x_i) + \varepsilon(x_i)$, $\varepsilon(x) \sim \mathcal{N}(0, \sigma^2)$.
- Let $f(x, w)$ — gaussian random field with parametric covariance function

$$K_0(x, x') = K_0(x, x'|\Theta).$$

- Covariance function of process $y(x)$

$$K(x, x'|\Theta, \sigma^2) = K_0(x, x'|\Theta) + \sigma^2\delta(x, x').$$

Posterior distribution

- Conditional distribution at new point

$$Law(y(x)|y(x_1), y(x_2), \ldots, y(x_n)) \sim \mathcal{N}(\hat{f}(x), \hat{\sigma}^2(x))).$$

- Mean

$$\hat{f}(x) = \mathbf{k}(x)\mathrm{K}^{-1}\mathbf{y}.$$

- Variance

$$\hat{\sigma}^2(x) = K(x,x) - \mathbf{k}(x)\mathrm{K}^{-1}\mathbf{k}(x)^T.$$

- Here $\mathbf{k}(x) = \{K(x, x_i)\}_{i=1}^N$, $\mathrm{K} = \{K(x_i, x_j)\}_{i,j=1}^N$, $\mathbf{y} = (y_1, \ldots, y_N)^T$.
- Unknown parameters are tuned using maximum likelihood method.

Table of contests

Likelihood

Likelihood:

$$l(\mathbf{a}) = \log p(\mathbf{y}|\mathbf{x}, \mathbf{a}) = -\frac{1}{2}\mathbf{y}^T \mathrm{K}^{-1}\mathbf{y} - \frac{1}{2}\log(\det(\mathrm{K})) - \frac{N}{2}\log 2\pi.$$

Derivatives w.r.t. hyperparameters:

$$\frac{\partial l}{\partial a_i} = -\mathrm{tr}\left(\mathrm{K}^{-1}\frac{\partial \mathrm{K}}{\partial a_i}\right) + \mathbf{y}^T \mathrm{K}^{-1}\frac{\partial \mathrm{K}}{\partial a_i}\mathrm{K}^{-1}\mathbf{y}.$$

Here $\mathbf{a}$ — vector of hyperparameters.
It is necessary to efficiently calculate values

$$\mathrm{K}^{-1}\mathbf{y},\ \mathrm{tr}\left(\mathrm{K}^{-1}\frac{\partial \mathrm{K}}{\partial a_i}\right),\ \log(\det(\mathrm{K})).$$

Cholesky decomposition

- Cholesky decomposition $K = U^T U$, where U is upper triangular matrix. It is time-consuming: $O(N^3)$ operations.
- Typical number of iterations for likelihood optimization using gradient method with multistart is $100 - 200$. If $N \gtrsim 10^3$ then learning becomes rather long, up to half an hour.
- Stochastic global optimization methods require much more iterations.

Table of contests

Product computation

Skilling (1993), Gibbs and MacKay (1997) suggest the following approach.

- Minimize $g(u) = \frac{1}{2}u^T \mathrm{K} u - u^T \mathbf{y}$. Exact minimum $u^* = \mathrm{K}^{-1}\mathbf{y}$.
- Quadratic function is minimized using some simple algorithm, for example conjugate gradients.
- Trace is approximated via Monte Carlo method $\operatorname{tr}\left(\mathrm{K}^{-1}\frac{\partial \mathrm{K}}{\partial a_i}\right) = E\left[d^T \mathrm{K}^{-1}\frac{\partial \mathrm{K}}{\partial a_i} d\right]$, $d \sim \mathrm{N}(0, 1)$, $\mathrm{K}^{-1}d$ is calculated using method above.

Discussion of method

Advantages:

- Accurate and fast method of approximating $\mathrm{K}^{-1}\mathbf{y}$.
- Simple termination conditions.

Drawbacks:

- Accurate trace estimator needs significant number of seeds, for each seed estimating $\mathrm{K}^{-1}d$ needs several N^2 operations, so calculation time is significant.
- It is not clear what to do with the determinant term.

Table of contests

Quasi-Newton approximation

Zhang and Leithead (2007) suggest an alternative algorithm.

- Quadratic function $g(u) = \frac{1}{2}u^T \mathrm{K} u - u^T \mathbf{y}$ is minimized using Quasi-Newton method.
- Thus, the sequence of approximations of inverse matrix K^{-1} is built using BFGS formula

$$
\begin{aligned}
&u_{j+1} = u_j - \eta_j \mathrm{K}_j^{-1} \bigtriangledown g(u_j), \\
&\mathrm{K}_{j+1}^{-1} = \mathrm{K}_j^{-1} + \triangle \mathrm{K}_j^{-1}, \\
&\triangle \mathrm{K}_j^{-1} = \left(1 + \frac{q_j^T \mathrm{K}_j^{-1} q_j}{q_j^T p_j}\right) \frac{p_j p_j^T}{p_j^T q_j} - \frac{p_j q_j^T \mathrm{K}_j^{-1} + \mathrm{K}_j^{-1} q_j p_j^T}{q_j^T p_j},
\end{aligned}
$$

here $p_j = u_{j+1} - u_j$, $q_j = \bigtriangledown g(u_{j+1}) - \bigtriangledown g(u_j)$.

Initialization and termination

- Initial approximation K_0^{-1} is the value derived in the previous point, where likelihood was calculated (with previous values of hyperparameters).
- Each iteration is $O(N^2)$ operations.
- Maximum number of iterations is some constant J, so, the whole method of calculating inverse covariance matrix is $O(N^2)$.
- Stopping criterion

$$|| \bigtriangledown g(u_j)||_\infty \leq \frac{\varepsilon}{N}.$$

Quality criterion and scaling

- Quality criterion

$$\frac{|\mathrm{tr}(\tilde{\mathrm{K}}^{-1}\mathrm{K}) - N|}{N} < \varepsilon_1,$$

  $\tilde{\mathrm{K}}^{-1}$ — approximation to the inverse matrix.
- Restart technique.
- Random scaling:

$$s = \mathrm{K}_j^{-1}(i,:) \times \mathrm{K}(:,i), i \text{ — random coordinate,}$$
$$\mathrm{K}_j^{-1} := \frac{\mathrm{K}_j^{-1}}{s}, j \text{ — number of iteration.}$$

Table of contests

Determinant approximation

Determinant approximation:

$$\log(\det(\mathrm{K})) = \mathrm{tr}(\log(K)),$$
$$\log(\mathrm{K}) = -\sum_{i=1}^{\infty}\left(\frac{\mathrm{K}^i}{i}\right) \text{ — matrix logarithm.}$$

Two-level scheme of approximation:

- Some finite number of terms in series: $\log(\mathrm{K}) = -\sum_{i=1}^{L}\left(\frac{\mathrm{K}^i}{i}\right)$
- Each element $\mathrm{tr}(\mathrm{K}^i)$ is approximated via Monte Carlo method.
- Authors suggest several compensation schemes for errors at both levels.

Discussion of method

Advantages:

- Simultaneous approximation of $\mathrm{K}^{-1}\mathbf{y}$ and K^{-1}.
- Quality control for trace computations.

Drawbacks:

- Complicated and rather long scheme of determinant approximation.
- No quality control for determinant term.

Table of contests

Quasi-Newton determinant updates

- Matrix determinant lemma:

$$\det(A + uv^T) = \det(A)(1 + v^T A^{-1} u).$$

- DFP update formula instead of BFGS:

$$\triangle \mathrm{K}_j^{-1} = \frac{p_j p_j^T}{q_j^T p_j} - \frac{\mathrm{K}_j^{-1} q_j q_j^T (\mathrm{K}_j^{-1})^T}{q_j^T \mathrm{K}_j^{-1} q_j},$$
$$\det(\mathrm{K}_{j+1}^{-1}) = \det(\mathrm{K}_j^{-1}) \frac{q_j^T p_j}{q_j^T \mathrm{K}_j^{-1} q_j}.$$

- At each iteration current approximation to determinant is exactly the determinant of current approximation to inverse matrix.

Further extensions

- More efficient scaling technique:

$$m = \mathrm{mean}(\mathrm{diag}(\mathrm{K}_j^{-1} \times \mathrm{K})),$$
$$\mathrm{K}_j^{-1} := \frac{\mathrm{K}_j^{-1}}{m}.$$

- If norm of gradient increases on successive iterations, the restart is made.
- Initial approximation K_0^{-1} is the value derived at one of the previous points, the closest one to the current point in the sense of special metric. Metric represents the l_2-norm difference between diagonals of covariance matrixes.

Table of contests

Computational experiments

- Large set of 211 functions, typical for testing optimization tasks.
- Dimensionality of functions — from 2 to 8.
- For each function training sets of 320 and 1000 points were generated.
- Comparison of MSE and time of training.
- MSE is calculated on large test sets of 10000 points.
- Results — Dolan-More curves.

Dolan-More curves

- T tasks, A algorithms.
- e_{ta} — MSE or time of training of a-th algorithm on t-th task.
- $\tilde{e}_t = \min_a e_{ta}$.

$$p_a(h) = \frac{\#\{t : e_{ta} < h\tilde{e}_t\}}{T}$$

- If the curve is higher, then algorithm is better.
- $p_a(1)$ — part of tasks, on which algorithm a is the best.

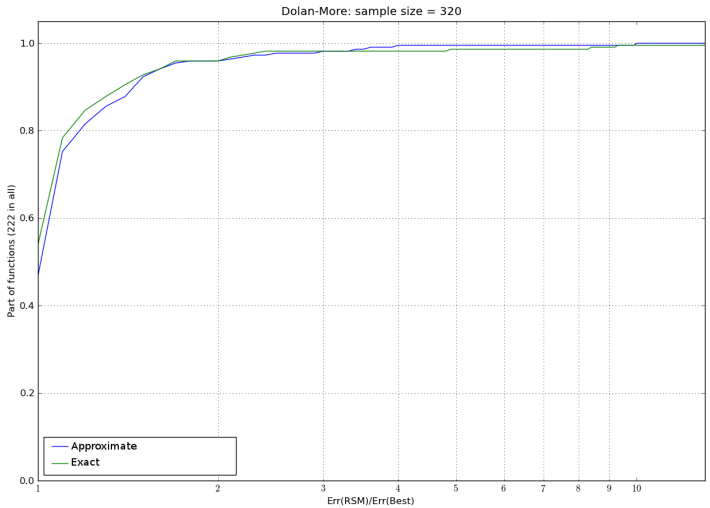

Figure: MSE, $N = 320$

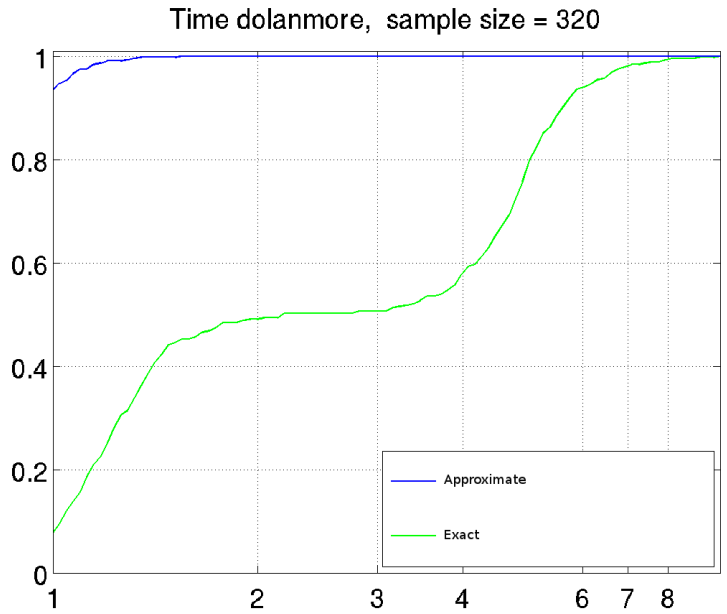

Figure: Time of training, $N = 320$

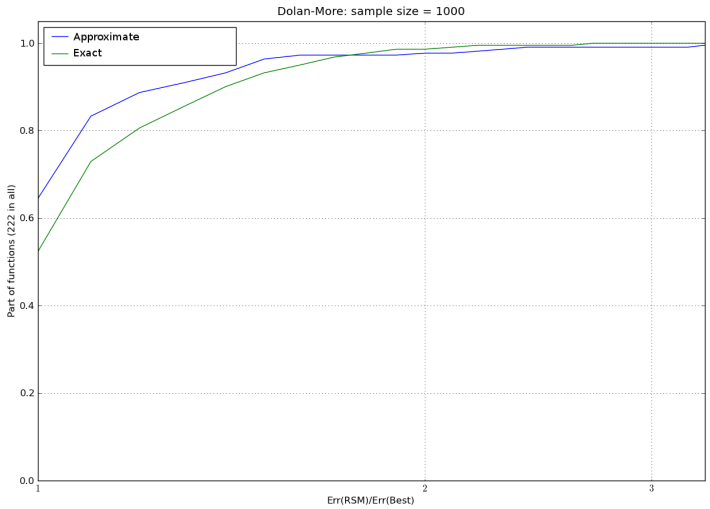

Figure: MSE, $N = 1000$

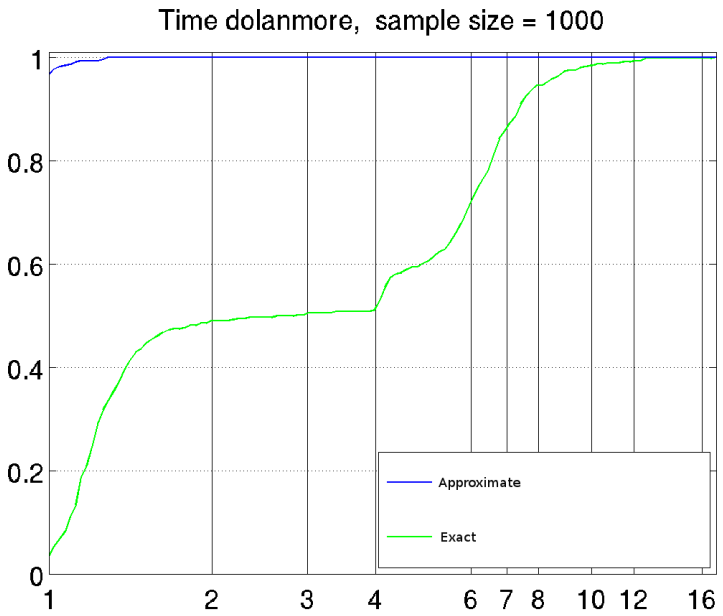

Figure: Time of training, $N = 1000$

Comparison

- Gibbs and Mackay — only ideas without full algorithm.
- Zhang and Leithead — 1 restart at $6-7$ iterations if $N = 1000$. But additional time for determinant approximation. Acceleration rate is less than 6.
- Proposed method: mean acceleration $4-6$ times.

Conclusion

Advantages:

- We avoid complicated schemes for determinant approximation.
- Simultaneous approximation of inverse matrix and its determinant.
- Quality control for determinant term.
- Experiment results: significant acceleration without loss of quality.

Thank you for attention.