Multi-task Gaussian Processes with Task-Dependent Kernels

1. Group Members

Greg Benton; gwb67@cornell.edu
David Kent; dk576@cornell.edu

2. General Outline

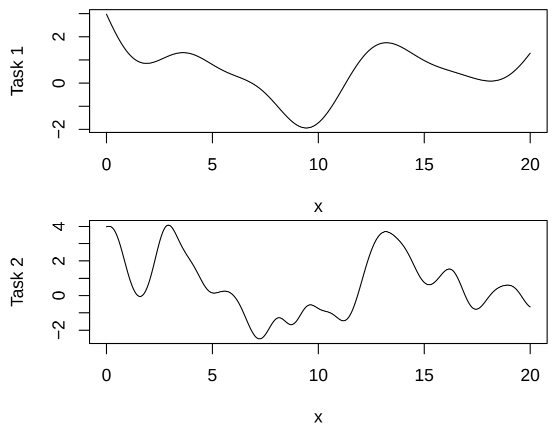
So far in the course we have looked at Gaussian processes that map from $\mathbb{R}^d$ into $\mathbb{R}$. There are, however, real world demands for processes for which the output is in $\mathbb{R}^q$ for $q > 1$. This is typically referred to as multi-task learning, where each of the q output dimensions corresponds to some task we wish to learn.

This has been approached in a couple distinct ways over the past few years and has been recently implemented in GPyTorch in the style of (Bonilla et al., 2008), where the covariance between $f_l(\mathbf{x})$ and $f_k(\mathbf{x}')$ is given by $K^f_{lk} k^x(\mathbf{x}, \mathbf{x}')$, where k^x is a correlation function shared by all q tasks, and K^f is a $q \times q$ matrix of the inter-task similarities. The drawback here is that each of the tasks relies on the same underlying kernel. In some applications, this is a reasonable simplification. The color channels of an image, for example, are likely to have similar covariance structures.

In this project we will extend the functionality of GPyTorch's multi-task learning to allow for the learning of correlated tasks that require distinct kernel parameters for each task. Using the RBF kernel as a starting point for its flexibility and ubiquity we will begin by implementing a method for modeling q tasks, each using an RBF kernel with distinct parameters.

We will follow (Melkumyan & Ramos, 2011a). The covariance between $f_l(\mathbf{x})$ and $f_k(\mathbf{x}')$ is given by some generic kernel function $k(\mathbf{x}, \mathbf{x}')$, but we take advantage of the fact that when we impose certain kernel functions on tasks l and k, the resulting kernel matrix K is a $qd \times qd$ block matrix with known form. For example, when tasks l and k are both given RBF kernels, the diagonal matrices K_{ll} and K_{kk} are identical to the single-task scenario, and the cross-covariances K_{lk} have an analytic form.

By the midterm report we will have a full implementation of multi-task Gaussian processes with distinct kernel parameters for each task implemented on top of GPyTorch. Validation at this stage will occur using synthetically generated data. Shown below are two time-series over the same domain generated by correlated Gaussian processes using RBF kernels with different length-scale parameters. Synthetically generated data like this will provide a baseline for measuring progress by the midterm report.



After validating on synthetic data, we will apply this implementation to real-world data and compare accuracy against the existing GPyTorch methods by comparing mean squared error on withheld data. An example of a scenario of interest in which our implementation would be useful is in the modeling of closed-end funds (CEFs). Typically the market price of a CEF is subject to much more volatility than the net asset value of the fund, so although these two objects are highly correlated we expect the individual covariance structure of the CEF's market price and the net asset value to be different.

After the full implementation and testing with the RBF kernel, we will implement other kernels which have known analytic cross-covariance, including the Matérn kernel with $\nu = 3/2$, and the "Sparse" kernel mentioned in (Melkumyan & Ramos, 2011a).

The planned division of labor is for Dave to spend the next few days building a small "toy" implementation of the methods described in (Bonilla et al., 2008) and (Melkumyan & Ramos, 2011a) to anchor our understanding and guide a more robust version in the future. During this time Greg will get a more rigorous understanding of the GPyTorch library in an effort to map out what code specifically will need to be written for the project and plan the structure of the architecture that will be built. Following this we will spend 7-10 days working collaboratively on the

GPyTorch version of the multi-kernel model. This gives us about one week to ten more days to work on validation and error checking on synthetic data, as well as assembling the midterm report.

After this, depending on the specific scheduling of our presentation we will work on developing the presentation and testing on real world data. If we are able to accelerate our timeline in a meaningful way we will use the extra time to add more kernels to the multi-kernel implementation for increased functionality.

3. Papers Read

References

Bonilla, Edwin V, Chai, Kian M, and Williams, Christopher. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pp. 153–160, 2008.

Melkumyan, Arman and Ramos, Fabio. Multi-kernel gaussian processes. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, pp. 1408, 2011a.

Melkumyan, Arman and Ramos, Fabio. Multi-kernel gaussian processes. Technical Report ACFR-TR-2011-002, Australian Center for Field Robotics, July 2011b.

Titsias, Michalis K and Lázaro-Gredilla, Miguel. Spike and slab variational inference for multi-task and multiple kernel learning. In *Advances in neural information processing systems*, pp. 2339–2347, 2011.

Wilson, Andrew Gordon, Knowles, David A, and Ghahramani, Zoubin. Gaussian process regression networks. *arXiv preprint arXiv:1110.4411*, 2011.