Questions and answers - Auralee Edelen Lecture 2

The following questions were submitted through Google Form. Some may have been answered in the Q&A session already. Nevertheless, we request our lecturers to provide written answers here for the benefit of those who could not attend that session. Thank you!

Slide 9. (1) How much does the representation power of gaussian process depend not only on the parameters for a given kernel, but also on the choices among the different kinds of kernels available? (2) In how many dimensions do you usually perform the regression?

A: (1) It matters quite a lot. For example, many applications use Matern kernel instead of RBF because it is better at learning more rapidly-varying "spikier" functions wrt changes in inputs, and save RBF for when they are expecting very smooth, slowly-varying function.

For some cases in accelerators, from basic beam optics we expect specific function responses (e.g. changing a quadrupole gives a concave up parabolic response in beam size); if we restrict the kernel to take that type of form, it can help us learn the exact response from less data but also means if the behavior deviates a lot from the ideal expected optics behavior we are less able to model those unexpected components.

(2) For GPs, we've used up to about 24 or so variable settings on the accelerator in the regression/optimization; one downside of using GPs is the N^3 scaling with data. To regress on more variables, generally we need more data. For neural networks in contrast we have used a hundred or so scalar variables plus image data in regression.

Slide not specified. One point that I understood is important for you is uncertainty estimation in the regression task. What are the advantages of GP compared to a Bayesian neural network (BNN) to perform the regression task? You mention little data that I think is one of the points (you need a lot of them to train a BNN). Are there other important aspects for which you prefer GP in your application?

The advantage of using a GP is primarily that we can get generally good predictions/uncertainty estimates for optimization with very little data (we don't necessarily need high accuracy of predictions, but we do need the predictions and uncertainty estimates to be well-behaved when used with optimization). This is useful

when learning on-the-fly as we sample the system during optimization. GPs also give quite well-behaved uncertainty estimates to use with BO.

BNNs in contrast are very useful in cases where we already have a large corpus of data for the problem. They can be a little tricky to initialize for a particular problem to get reliable uncertainty estimates, especially when trying to traverse OOD parts of the parameter space in the context of BO. The problem then also becomes how best to update the model during optimization. One of the areas of research being explored right now is exactly this for accelerators – using BNNs to give prior mean information for a GP, or fine-tuning just a small output layer of a BNN. There are similar ideas in deep kernel learning for BO.