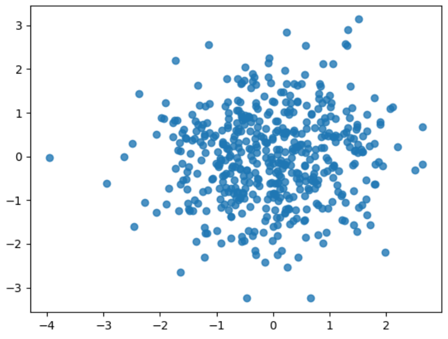
Progress Report
Model Setup
April 16, 2018

Contents

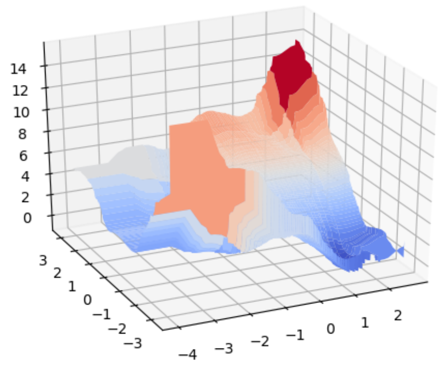
Simulated Data

First generate data from a spatial gaussian process with 100 locations, we sample spatial locations (x, y) iid from standard normal, and assume the pollutant z follow below Gaussian Process:

$$z(x, y) \overset{iid}{\sim} N(f(x, y), \sigma^2 = 0.1)$$
$$f(x, y) = 0.2x + 0.5y + \sqrt{x^2 + y^2 + 5 * cos(x * y)} + sin(x) + cos(x) + log\Big(exp(x * y) + exp(x)\Big)$$



(a) Sampled saptial location for monitoring sites (standardized)



(b) Average pollutant surface over space (standardized)

We then generate prediction for $z(x, y)$ from 10 base GP models [Duvenaud et al., 2013]:

1. Linear,
2. Polynomial, degree 2, 3, 4
3. Gaussian RBF, with ARD
4. Matérn $\frac{1}{2}$, $\frac{3}{2}$, $\frac{5}{2}$ with ARD

5. MLP, with ARD.
   Equivalent to a 2-layer network with Gaussian CDF activation function and infinite hidden units:

$$k(x,y) = \sigma^2 \frac{2}{\pi} \text{asin} \left(\frac{\sigma_w^2 x^\top y + \sigma_k^2}{\sqrt{\sigma_w^2 x^\top x + \sigma_k^2 + 1}\sqrt{\sigma_w^2 y^\top y + \sigma_k^2 + 1}} \right)$$

6. Spectral Mixture [Wilson and Adams, 2013]

Model Overview

Input: data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, and prediction from base models $\{\hat{\mathbf{y}}_k\}_{k=1}^K$.
Step 1: Learn model-specific GP from model prediction (i.e. the pre-ensemble processing step):

$$y(\mathbf{x}) \sim \mathcal{GP}_k(\hat{y}_k(\mathbf{x}), cov(z_k)(\mathbf{x}))$$

Step 2: Learn ensemble weights $\{w_k(\mathbf{x})\}_{k=1}^K$

$$\begin{aligned} \mathbf{w} &\sim Dirichlet(exp(\mathbf{w}')) \qquad \text{where } \mathbf{w} = \{w_k\}_{k=1}^K, \mathbf{w}' = \{w_k'\}_{k=1}^K \\ w_k'(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0},\ k_{w,k}(\mathbf{x}, \mathbf{x}') + \sigma_k^2) \end{aligned}$$

Step 3: Generate ensemble predictive distribution:

$$y \sim \sum_{k=1}^K w_k(\mathbf{x}) * \mathcal{GP}_k\Big(\hat{y}_k(\mathbf{x}),\ cov(z_k)(\mathbf{x})\Big)$$

Below sections describe the model derivation for each step in detail. For the purpose of debugging and future model improvement, for each step, I will also (1) highlight model assumption and (2) propose test cases.

Step 1: Learn Model-specific GP

This step learn a standard GP for each model-specific residual process $\hat{\boldsymbol{\epsilon}}_k = \mathbf{y} - \hat{\mathbf{y}}_k(\mathbf{x})$.

Assumption 1 (A1):
Covariance structure for all residual processes $\boldsymbol{\epsilon}_k$ can be well captured by a common kernel family $k_\epsilon(\boldsymbol{\theta})$.

Denote $\mathbf{K}_\epsilon$ as the kernel matrix generated by k_ϵ, then GP model adopts below prior:

$$\boldsymbol{\epsilon}_k | \boldsymbol{\theta}_{\epsilon,k} \sim \mathcal{GP}_k\Big(0,\ \mathbf{K}_\epsilon(\boldsymbol{\theta}_{\epsilon,k}) + \sigma_k^2 \mathbf{I}\Big)$$

where $\boldsymbol{\theta}_{\epsilon,k}$ are the kernel parameters with ARD prior.
Correspondingly, if denote $\mathbf{K}_{\epsilon,k} = \mathbf{K}_\epsilon(\boldsymbol{\theta}_{\epsilon,k})$, the predictive posterior is [Rasmussen and Williams, 2006]:

$$\boldsymbol{\epsilon}_k | \boldsymbol{\theta}_{\epsilon,k}, \hat{\boldsymbol{\epsilon}}_k \sim \mathcal{GP}_k\Big(\mathbf{K}_{\epsilon,k}(\mathbf{K}_{\epsilon,k} + \sigma_k^2 \mathbf{I})\boldsymbol{\epsilon}_k,\ \mathbf{K}_{\epsilon,k} - \mathbf{K}_{\epsilon,k}(\mathbf{K}_{\epsilon,k} + \sigma_k^2 \mathbf{I})^{-1}\mathbf{K}_{\epsilon,k}\Big)$$

therefore, the model-specific predictive posterior for observation $\mathbf{y} = \boldsymbol{\epsilon}_k + \hat{\mathbf{y}}_k$ as:

$$\mathbf{y} \sim \mathcal{GP}_k\Big(\hat{\mathbf{y}}_k + \mathbf{K}_{\epsilon,k}(\mathbf{K}_{\epsilon,k} + \sigma_k^2 \mathbf{I})\boldsymbol{\epsilon}_k,\ \mathbf{K}_{\epsilon,k} - \mathbf{K}_{\epsilon,k}(\mathbf{K}_{\epsilon,k} + \sigma_k^2 \mathbf{I})^{-1}\mathbf{K}_{\epsilon,k}\Big)$$

Test Case (T1):
Rationale: By [Krogh and Vedelsby, 1994], a simple bias-variance trade-off derivation shows that the cross-validation error of overall ensemble can be written as

$$\epsilon_{ens} = E(\epsilon_k) - Var(\epsilon_k)$$

Therefore we need to examine $E(\epsilon_k)$ and $Var(\epsilon_k)$ of model-specific GPs and make sure they give similar (or better) ensemble performance compared to that of the original model-specific predicts.

1. Examine $E(\epsilon_k)$ of the model-specific GP, compare to that of the original model predicts.
2. Examine $Cov(\epsilon_k)$ of model-specific GPs, compare to that of the original model predicts.

(Compared to that of the original prediction, we expect decreased $E(\epsilon_k)$ and decreased $Var(\epsilon_k)$ for model-specific GPs, because these model-specific GPs are essentially estimates from a two-model GAMs estimated using backfitting).

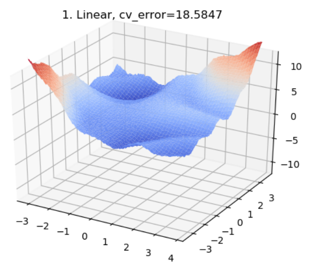
In current implementation, we use RBF for k_ϵ.

Step 1 Experiment
First visualize residual process from individual model:

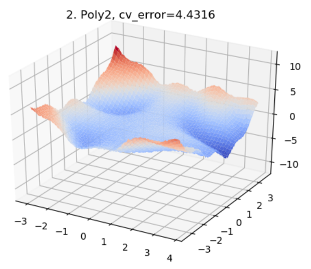
Model	Linear	Poly 2	Poly 3	Poly 4	RBF	Matern 1/2	Matern 3/2	Matern 5/2	MLP
CV, Original	18.5847	4.4316	4.5581	4.9085	4.7633	4.7373	4.7784	4.7644	4.7614
CV, GP	4.5211	4.5217	4.5217	4.5216	4.5217	4.5192	4.5208	4.5212	4.5217

Table 1: CV Performance for pre-processing step

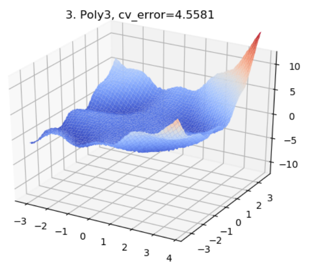
Variance among original predictors: 1.4226, Variance among GP predictors: 6.2087×10^{-3}.



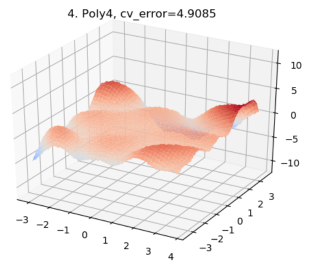
(a) Linear



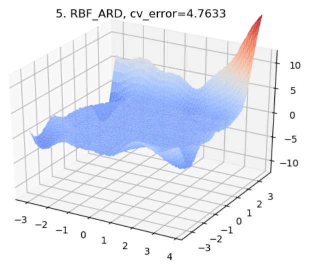
(b) Polynomial, Degree 2



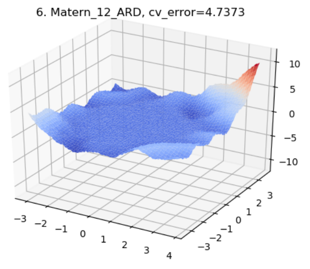
(c) Polynomial, Degree 3



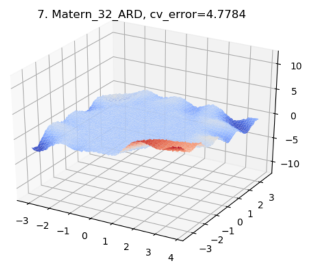
(d) Polynomial, Degree 4



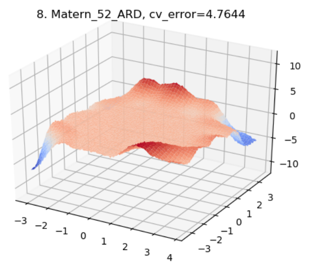
(e) RBF with ARD



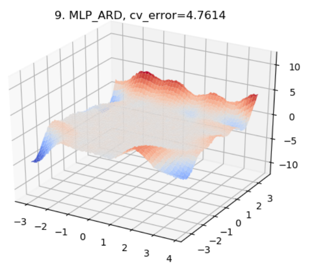
(f) Matén $\frac{1}{2}$ with ARD



(g) Matén $\frac{3}{2}$ with ARD



(h) Matén $\frac{5}{2}$ with ARD



(i) MLP with ARD

Figure 2: Residual Process from individual models

Step 2-3: Learn Ensemble Weight & Predictive Distribution

Based on discussion on 03/22, try a simplied ensemble:
Step 1: Take prediction as it is, don't learn model-specific GP:

$$\{\hat{y}_k(\mathbf{x})\}_{k=1}^K$$

Step 2: Learn ensemble weights $\{w_k(\mathbf{x})\}_{k=1}^K$
For representing the ensemble weights, we have two options here:

1. Dirichlet: $w_k \sim Dirichlet(exp(w'_k))$, or equivalently:

$$w_k = \frac{a_k}{\sum_{k=1}^K a_k} \qquad a_k \sim Gamma(exp(w'_k), 1)$$

2. Logistic Normal: $w_k = softmax(exp(w'_k + \epsilon))$, where $\epsilon \sim N(0,1)$ or equivalently:

$$w_k = \frac{a_k}{\sum_{k=1}^K a_k} \qquad a_k \sim logNormal(w'_k, 1)$$

   A more flexible, better behaved distribution compared to Dirichlet [Aitchison and Shen, 1980].

for both options, we have:

$$w'_k(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0},\ k_{w,k}(\mathbf{x}, \mathbf{x}') + \sigma_k^2)$$

Step 3: Generate ensemble predictive distribution:

$$y \sim N\Big(\sum_{k=1}^K w_k(\mathbf{x})\hat{y}_k(\mathbf{x}) + \epsilon(\mathbf{x}),\ \mathbf{I}\Big)$$
$$\epsilon(\mathbf{x}) \overset{iid}{\sim} N(0,1)$$

here $\epsilon(\mathbf{x})$ is a residual process that serves as "sculpting clay" that makes up for the ensemble, in case that in some space-time location where no base model predicts well.
Note:

1. For model-specific GP $w'_k(\mathbf{x})$, flexibility of these GPs will likely to have non-trivial influence on model behavior. For example, a very restrictive kernel (say, constant kernel) forces $w_k(\mathbf{x})$ to be a constant, in this case, the Step 3 becomes essentially Bayesian linear regression (which, depends on the sample size, may not be a bad choice). On the other hand, a very flexible kernel may lead to model overfit when sample size is small. Need to investigate this in simulation.

2. For Step 2, both options consistutes valid construction of normalized random measure (need to verify if this statement is correct) over the candidate models. However, compare to Dirichlet, Logistic Normal is empirically much easier to sample both in terms of computation speed ($\times 30$ faster under NUTS) and more efficient in terms of sample quality (higher ESS). Question: how do we decide which option is better?

Posterior Visualization

Ensemble Weights

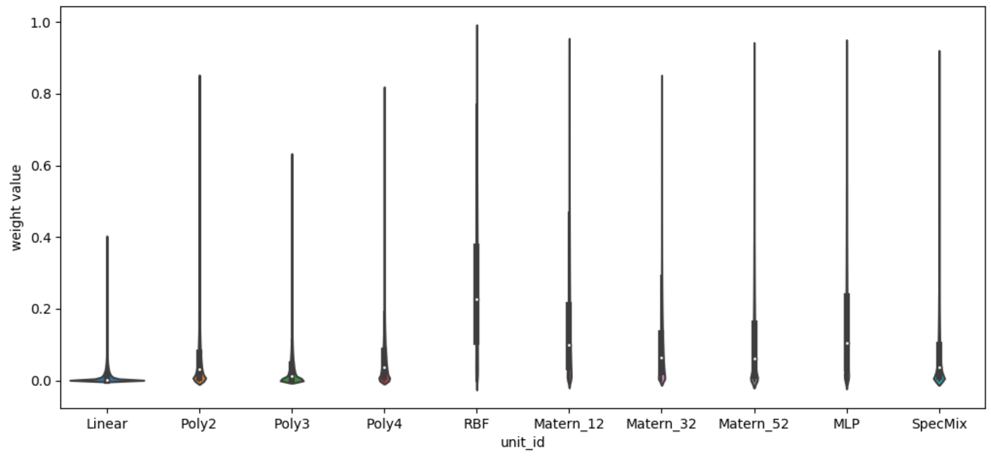


Figure 3: Dirichlet Representation

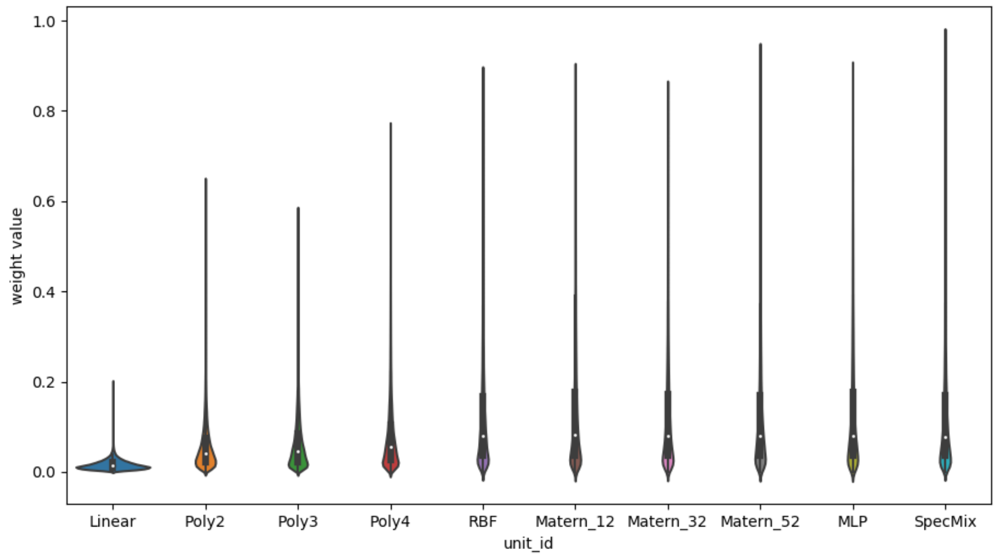


Figure 4: Logistic Normal Representation

Recall the prediction performance in first stage:

Linear	Poly 2	Poly 3	Poly 4	RBF	Matern 1/2	Matern 3/2	Matern 5/2	MLP	Spectral
19.8786	5.7871	5.8365	5.7879	5.5445	5.6745	6.0123	6.0106	5.9457	5.5445

Table 2: CV Performance of base models

Timeline

1. (March. Week 3-4)
   Initial implementation.
2. (March. Week 4 - April. Week 1)
   Initial experiment on simulated data. Investigating:
   (a) Sensitivity to model and kernel specification
   (b) Whether improved performance for overall ensemble, how it is tighed to the property of the base models. More specifically:
      i. Variance among model prediction
      ii. Number and bias of models (few strong model with good prediction for different aspect of the data, or large collection of weak models)
   (c) Identifiability of individual weights
3. (April. Week 2-3)
   More realistic experiment, gradually increase complexity of data-generation mechanism toward (using the mean-surface from QD or Itai's prediction)
4. (April. Week 4) Start experiment on real data

Direction of Improvement

Two idea for improvement:

1. **Step 1**, it should still be ok to use Gaussian process. Since under a zero-mean GP, the marginal distribution of $\epsilon(\mathbf{x}) = y(\mathbf{x}) - \hat{y}(\mathbf{x})$ is still zero-mean, i.e. $[\epsilon(\mathbf{x}_{obs}), \epsilon(\mathbf{x}^*)] \sim N(\mathbf{0}, \mathbf{K})$ for training data $\mathbf{x}_{obs}$ and testing data $\mathbf{x}^*$. It is just that the conditional posterior $\epsilon(\mathbf{x}^*)|\epsilon(\mathbf{x}_{obs})$ follows a non-zero mean Gaussian. However, since our focus in this step is just recovering a probability distribution from data, the focus should be learn hyperparameter of the kernel functions rather performing prediction. (However, still a good idea to make the kernel restrictive)

2. **Step 3**, model the residual process of overall ensemble $\boldsymbol{\epsilon}(\mathbf{x})$ using a (infinite) random feature ensemble [Rahimi and Recht, 2009] to enhance prediction. In this case, since support of model index k is countably infinite, we can put a Bayesian nonparametric prior on $w_k(\mathbf{x})$ (say, a GP-based dependent DP-Mixture Jara and Hanson [2011]), which makes $\sum_{k=1}^{\infty} w_k(\mathbf{x}) y_k(\mathbf{x})$ a weak-limit approximation of the Bayesian nonparametric integration over a space of spatial termporal processes.

References

David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure Discovery in Nonparametric Regression through Compositional Kernel Search. *arXiv:1302.4922 [cs, stat]*, February 2013. URL `http://arxiv.org/abs/1302.4922`. arXiv: 1302.4922.

Andrew Wilson and Ryan Adams. Gaussian Process Kernels for Pattern Discovery and Extrapolation. In *International Conference on Machine Learning*, pages 1067–1075, February 2013. URL `http://proceedings.mlr.press/v28/wilson13.html`.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. University Press Group Limited, January 2006. ISBN 978-0-262-18253-9. Google-Books-ID: vWtwQgAACAAJ.

Anders Krogh and Jesper Vedelsby. Neural Network Ensembles, Cross Validation and Active Learning. In *Proceedings of the 7th International Conference on Neural Information Processing Systems*, NIPS'94, pages 231–238, Cambridge, MA, USA, 1994. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2998687.2998716`.

J. Aitchison and S. M. Shen. Logistic-Normal Distributions: Some Properties and Uses. *Biometrika*, 67(2): 261–272, 1980. ISSN 0006-3444. doi: 10.2307/2335470. URL `http://www.jstor.org/stable/2335470`.

A. Jara and T. E. Hanson. A class of mixtures of dependent tail-free processes. *Biometrika*, 98(3):553–566, September 2011. ISSN 0006-3444. doi: 10.1093/biomet/asq082. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3398659/`.

Ephraim M. Hanks. Modeling Spatial Covariance Using the Limiting Distribution of Spatio-Temporal Random Walks. *Journal of the American Statistical Association*, 112(518):497–507, April 2017. ISSN 0162-1459. doi: 10.1080/01621459.2016.1224714. URL `https://doi.org/10.1080/01621459.2016.1224714`.

Ali Rahimi and Benjamin Recht. Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1313–1320. Curran Associates, Inc., 2009. URL `http://papers.nips.cc/paper/3495-weighted-sums-of-random-kitchen-sinks-replacing-minimization-with-randomization-in-learni`
`pdf`.