Deep Gaussian Process Surrogates for Computer Experiments

Robert B. Gramacy with **Annie Sauer** and Dave Higdon and Andy Cooper

Virginia Tech Department of Statistics

January 2023

Where are we going?

1. Deep Gaussian Processes
   - Why?
   - What?
   - How?
2. Active Learning
   - Why?
   - What?
   - How?
3. Vecchia Approximation
   - Why?
   - What?
   - How?

Surrogates as statistical models of computer experiments

Surrogates are meta-models of computer experiments.

Surrogates are used to make **predictions** with appropriate **uncertainty quantification (UQ)**.

As simulations become more complex, surrogate models must keep up.

"Shallow" Gaussian process (GP) surrogates

The typical surrogate model is a GP

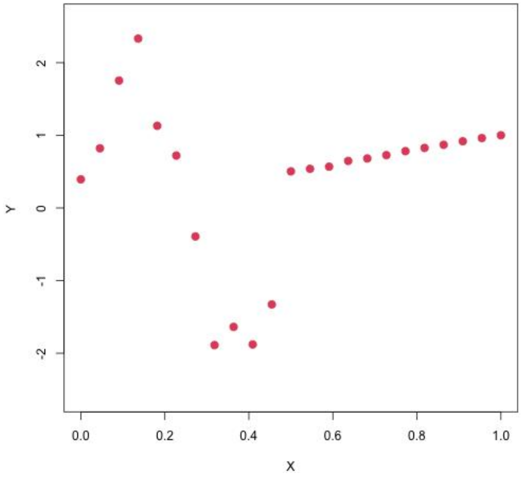
- nonlinear
- nonparametric (mostly)
- adept at uncertainty quantification

A GP assumes a MVN prior

$$Y \sim \mathcal{N}(0, \Sigma(X))$$

All of the "work" is in the covariance

$$\Sigma(X)^{ij} = \tau^2 \left(k \left(\frac{||x_i - x_j||^2}{\theta} \right) + g\mathbb{I}_{i=j} \right)$$



Higdon (2002), Virtual Library of Simulation Experiments (VLSE)

"Shallow" Gaussian process (GP) surrogates

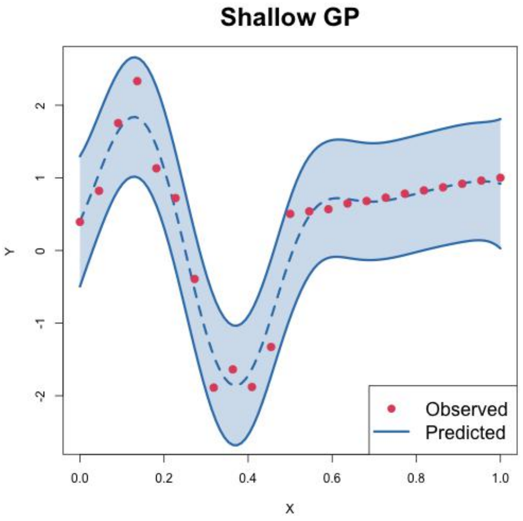
Conditioned on observed data (X, Y) and hyperparameter settings, posterior predictions at locations $\mathcal{X}$ follow

$$Y(\mathcal{X}) \mid X, Y \sim \mathcal{N}(\mu^\star, \Sigma^\star)$$

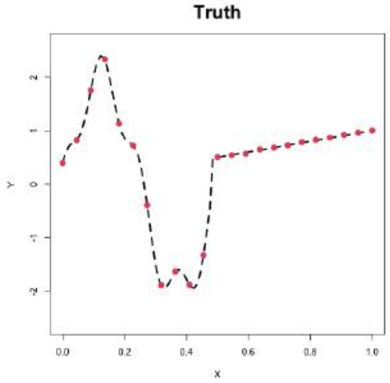
where

$$\mu^\star = \Sigma(\mathcal{X}, X)\Sigma(X)^{-1}Y$$
$$\Sigma^\star = \Sigma(\mathcal{X}) - \Sigma(\mathcal{X}, X)\Sigma(X)^{-1}\Sigma(X, \mathcal{X})$$

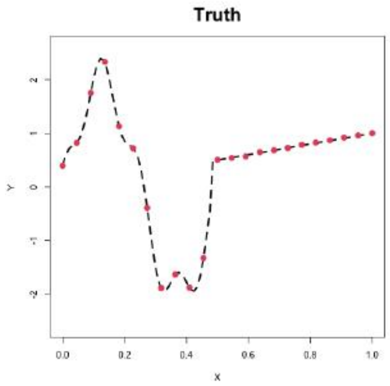
Hyperparameters may be estimated through MLE or sampled through MCMC.

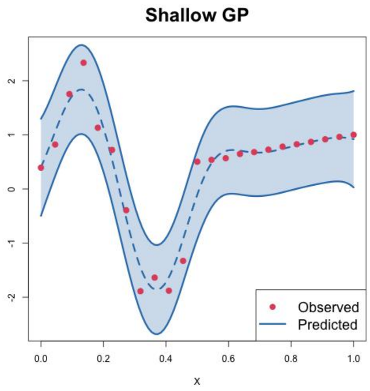


"Shallow" GP surrogates are limited by stationarity

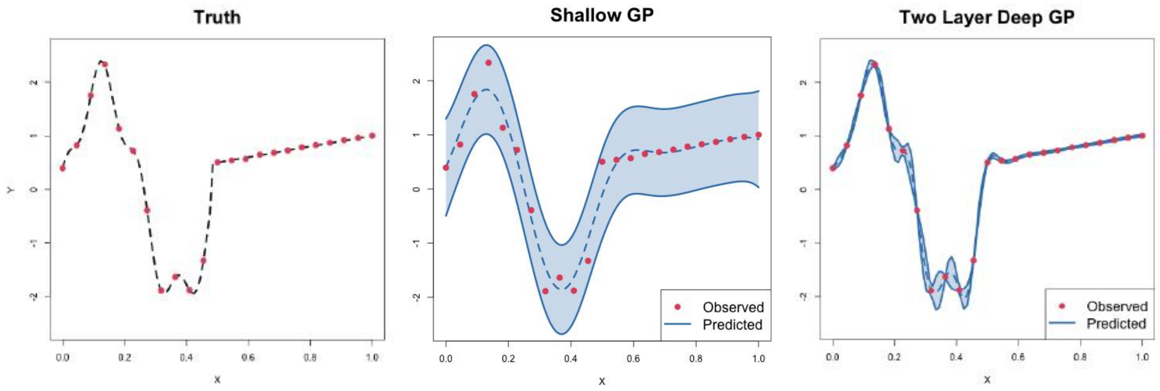


"Shallow" GP surrogates are limited by stationarity

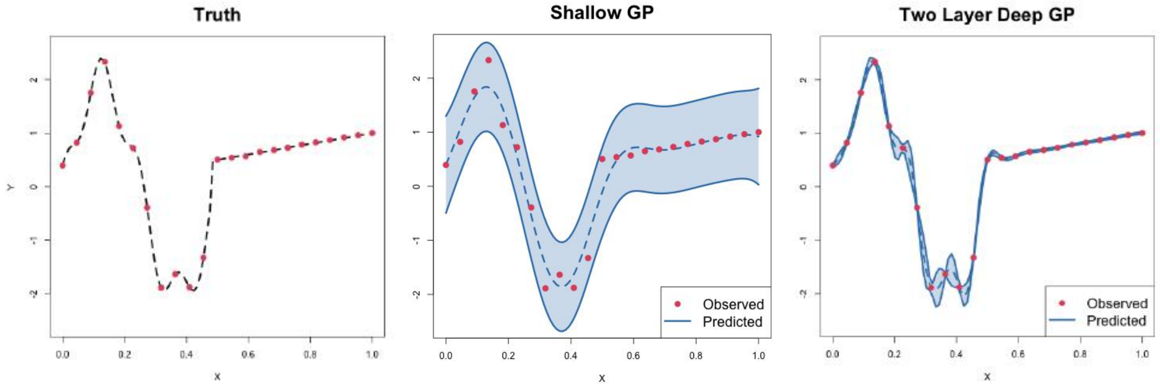




“Shallow” GP surrogates are limited by stationarity



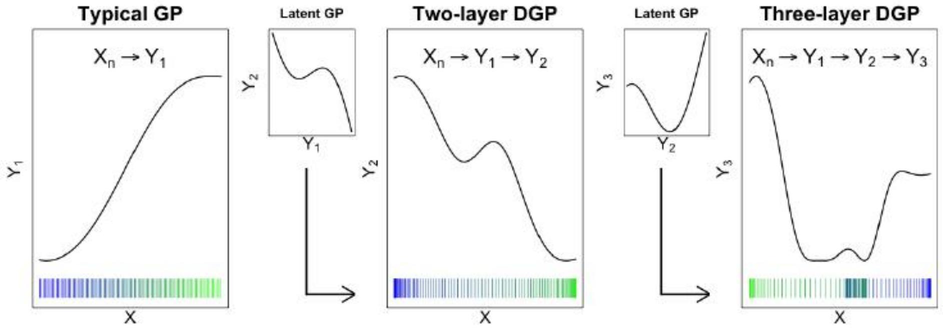
"Shallow" GP surrogates are limited by stationarity



Approaches to modeling non-stationarity

- Non-stationary kernels (Paciorek & Schervish, 2003; Higdon et al., 1999)
- Partition/Local GPs (Gramacy & Lee, 2007; Gramacy & Apley, 2015)
- Deep GPs (Damianou & Lawrence, 2012; Schmidt & O'Hagan, 2003)

DGPs are functional compositions of GPs



$$Y_1 \sim \mathcal{N}(0, \Sigma(X)) \quad \longrightarrow \quad Y_2 \sim \mathcal{N}(0, \Sigma(Y_1)) \quad \longrightarrow \quad Y_3 \sim \mathcal{N}(0, \Sigma(Y_2))$$

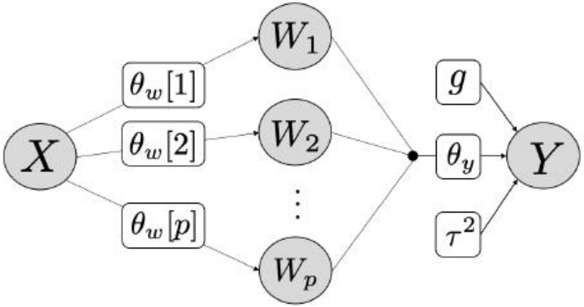
Intermediate Gaussian layers are unobserved/latent

We represent a two-layer DGP prior as

$$Y \mid W \sim \mathcal{N}\left(0, \Sigma(W)\right)$$
$$W_k \overset{\text{ind}}{\sim} \mathcal{N}\left(0, \Sigma(X)\right) \quad \forall \;\; k = 1, \ldots, p.$$

Posterior inference requires

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid W)\mathcal{L}(W \mid X)\; dW$$



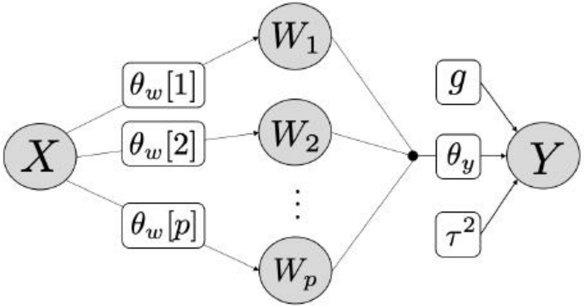
Intermediate Gaussian layers are unobserved/latent

We represent a two-layer DGP prior as

$$Y \mid W \sim \mathcal{N}(0, \Sigma(W))$$
$$W_k \overset{\text{ind}}{\sim} \mathcal{N}(0, \Sigma(X)) \quad \forall \;\; k = 1, \ldots, p.$$

Posterior inference requires

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid W)\mathcal{L}(W \mid X)\; dW$$



To encourage identifiability and parsimony, we impose

- Unit scale and noise-free latent W
- Conditional independence among nodes of W
- Isotropic length scales (single θ for all dimensions of X and W)

Direct posterior inference for DGPs is intractible

Direct posterior inference is intractible due to the latent layer W.

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid W)\mathcal{L}(W \mid X)\, dW$$

Direct posterior inference for DGPs is intractible

Direct posterior inference is intractible due to the latent layer W.

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid W)\mathcal{L}(W \mid X)\, dW$$

Methods for approximate DGP inference:

- Variational inference (Damianou & Lawrence, 2012; Salimbeni & Deisenroth, 2017; Marmin & Filippone, 2022)
- Expectation propogation (Bui et al., 2016)
- Hamiltonian Monte Carlo sampling (Havasi et al., 2018)

Direct posterior inference for DGPs is intractible

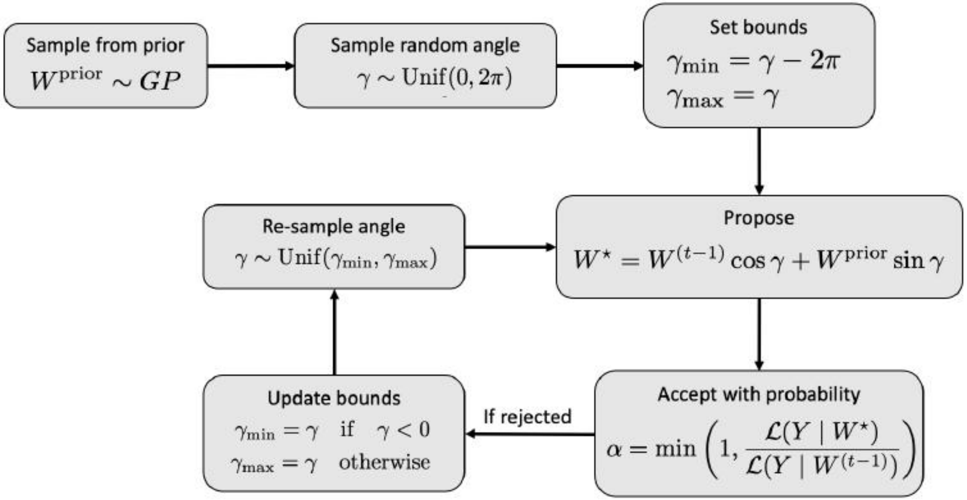
Direct posterior inference is intractible due to the latent layer W.

$$\mathcal{L}(Y \mid X) \propto \int \mathcal{L}(Y \mid W)\mathcal{L}(W \mid X)\, dW$$

To prioritize UQ, we embrace a fully-Bayesian MCMC inferential scheme.

- Metropolis-Hastings sampling of covariance hyperparameters
- **Elliptical slice sampling of latent Gaussian layers** (Murray et al., 2010)
- Iteration in a Gibbs scheme

Elliptical slice sampling provides efficient mixing


Sample from prior
$W^{\text{prior}} \sim GP$
Sample random angle
$\gamma \sim \text{Unif}(0, 2\pi)$
Set bounds
$\gamma_{\min} = \gamma - 2\pi$
$\gamma_{\max} = \gamma$
Propose
$W^\star = W^{(t-1)} \cos\gamma + W^{\text{prior}} \sin\gamma$
Accept with probability
$\alpha = \min\left(1, \frac{\mathcal{L}(Y \mid W^\star)}{\mathcal{L}(Y \mid W^{(t-1)})}\right)$
If rejected
Update bounds
$\gamma_{\min} = \gamma$ if $\gamma < 0$
$\gamma_{\max} = \gamma$ otherwise
Re-sample angle
$\gamma \sim \text{Unif}(\gamma_{\min}, \gamma_{\max})$

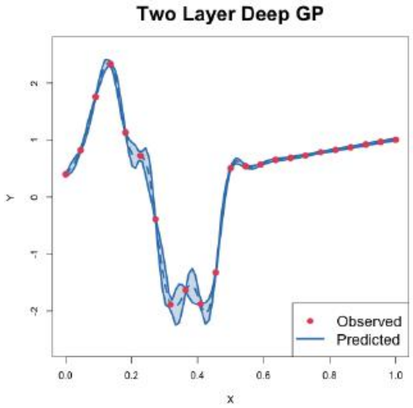
Also used for stochastic imputation (Ming et al., 2021)

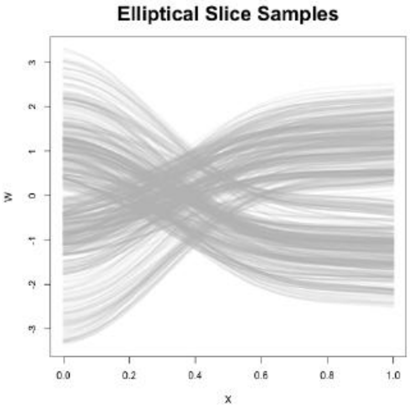
Elliptical slice sampling provides efficient mixing

```
R> library(deepgp)
R> fit <- fit_two_layer(x, y, nmcmc = 10000)
R> fit <- trim(fit, 5000, 5)
R> fit <- predict(fit, x_pred)
```

Elliptical slice sampling provides efficient mixing

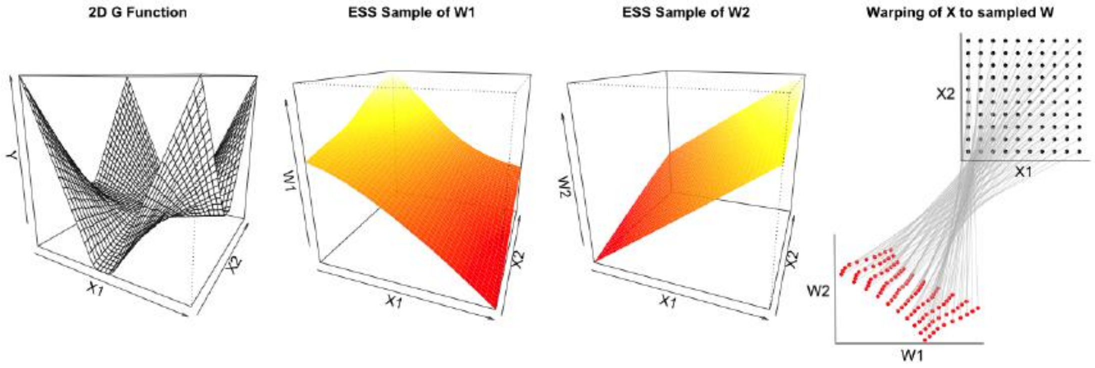
```
R> library(deepgp)
R> fit <- fit_two_layer(x, y, nmcmc = 10000)
R> fit <- trim(fit, 5000, 5)
R> fit <- predict(fit, x_pred)
```





Preview of DGP predictive prowess

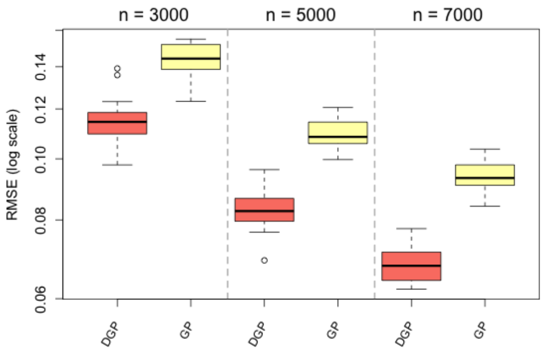
2-dimensional G-function

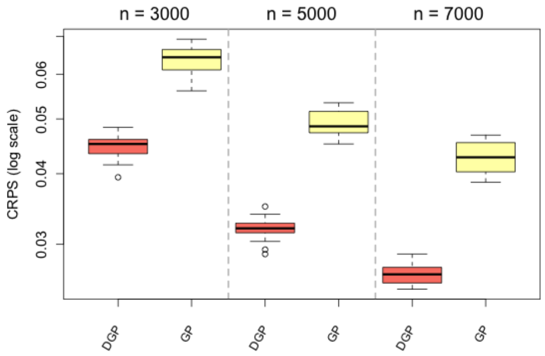


Marrel et al. (2009), VLSE

Preview of DGP predictive prowess

4-dimensional G-function (20 reps)





- RMSE = root mean squared error
- CRPS = continuous rank probability score (Gneiting & Raftery, 2007)

Deep Gaussian processes - Summary

- Why?
  - Non-stationary flexibility while maintaining the predictive prowess and uncertainty quantification of “shallow” GPs
- What?
  - Functional compositions of Gaussian layers
  - Intermediate layers are latent/unobserved
- How?
  - Bayesian MCMC hinging on elliptical slice sampling of latent layers
  - Implementation in the `deepgp` package

Statistical models are only as good as their data

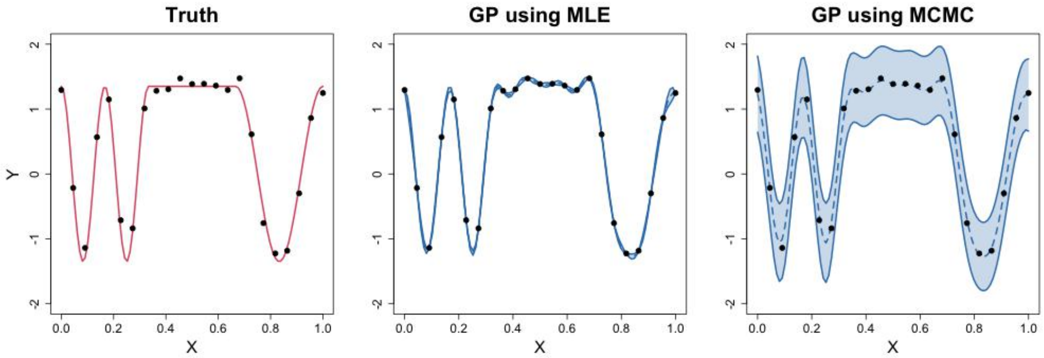
While a DGP has the flexibility to *address* non-stationarity, the data must *reveal* it.

- Strategically choose input configurations to maximize learning from a limited budget (Sauer, Gramacy & Higdon, 2022; Gramacy, Sauer, & Wycoff, 2022).

Statistical models are only as good as their data

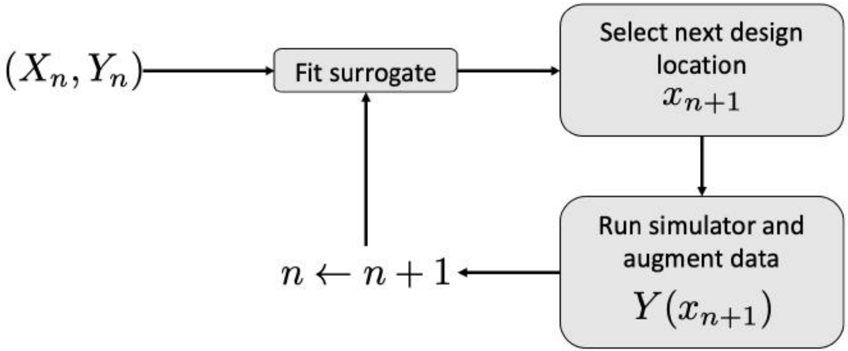
While a DGP has the flexibility to *address* non-stationarity, the data must *reveal* it.

- Strategically choose input configurations to maximize learning from a limited budget (Sauer, Gramacy & Higdon, 2022; Gramacy, Sauer, & Wycoff, 2022).



Active Learning/Sequential Design through greedy acquisitions

When computational costs are high, we may make the most of a stringent simulation budget through greedy acquisition: sequential design.


(X_n, Y_n)
Fit surrogate
Select next design location x_{n+1}
Run simulator and augment data $Y(x_{n+1})$
$n \leftarrow n+1$

Active Learning/Sequential Design through greedy acquisitions

$$Y(x) \mid X_n, Y_n \sim \mathcal{N}\left(\mu(x), \sigma^2(x)\right) \quad \text{for} \quad \begin{aligned} \mu(x) &= \Sigma(x, X_n)\Sigma(X_n)^{-1} Y_n \\ \sigma^2(x) &= \Sigma(x) - \Sigma(x, X_n)\Sigma(X_n)^{-1}\Sigma(X_n, x) \end{aligned}$$

Given augmented inputs $X_{n+1} = \{X_n, x_{n+1}\}$, the variance becomes

$$\sigma^2_{n+1}(x) = \Sigma(x) - \Sigma(x, X_{n+1})\Sigma(X_{n+1})^{-1}\Sigma(X_{n+1}, x)$$

Active Learning/Sequential Design through greedy acquisitions

$$Y(x) \mid X_n, Y_n \sim \mathcal{N}\left(\mu(x), \sigma^2(x)\right) \quad \text{for} \quad \begin{array}{rl} \mu(x) & = \Sigma(x, X_n)\Sigma(X_n)^{-1}Y_n \\ \sigma^2(x) & = \Sigma(x) - \Sigma(x, X_n)\Sigma(X_n)^{-1}\Sigma(X_n, x) \end{array}$$

Given augmented inputs $X_{n+1} = \{X_n, x_{n+1}\}$, the variance becomes

$$\sigma^2_{n+1}(x) = \Sigma(x) - \Sigma(x, X_{n+1})\Sigma(X_{n+1})^{-1}\Sigma(X_{n+1}, x)$$

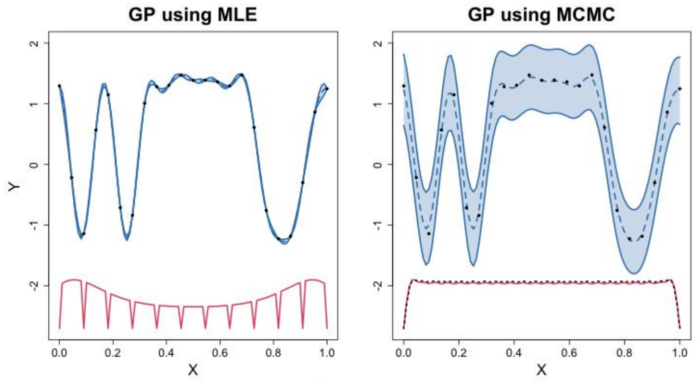
We choose acquisitions to minimize the posterior predictive variance.

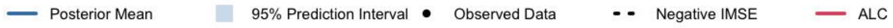
$$x_{n+1} = \underset{x_{n+1}}{\text{argmin}}\ \mathrm{IMSE}(x_{n+1}) \quad \text{where} \quad \mathrm{IMSE}(x_{n+1}) = \int \sigma^2_{n+1}(x)dx$$

For faster computation, we also utilize the sum approximation (Cohn, 1994).

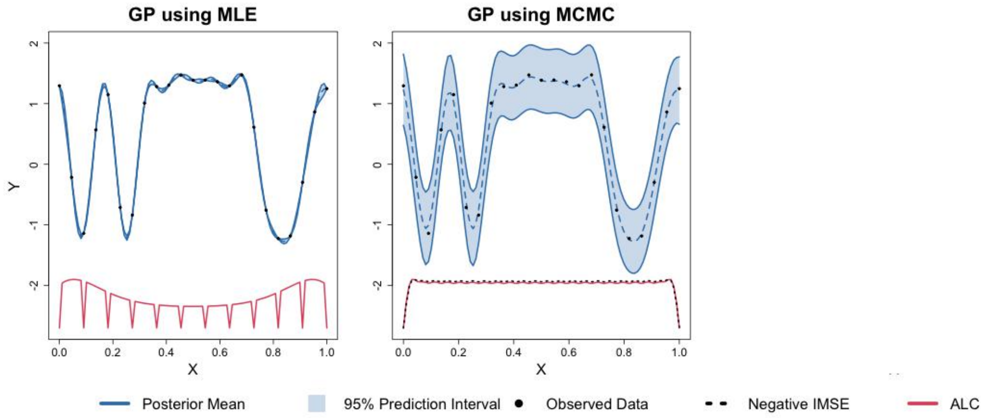
$$x_{n+1} = \underset{x_{n+1}}{\text{argmax}}\ \mathrm{ALC}(x_{n+1}) \quad \text{where} \quad \mathrm{ALC}(x_{n+1}) \propto -\sum_{x \in X_{ref}} \sigma^2_{n+1}(x)$$

Active Learning/Sequential Design through greedy acquisitions





Active Learning/Sequential Design through greedy acquisitions

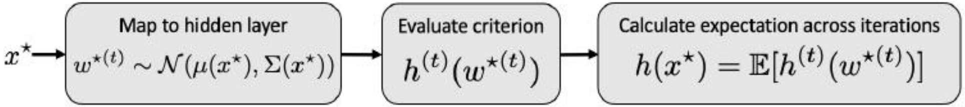


If the surrogate is stationary, sequential designs will end up "space-filling."

Active Learning for DGPs

Novel inputs $x^\star$ are mapped to hidden layer $w^{\star(t)}$ using typical GP prediction.

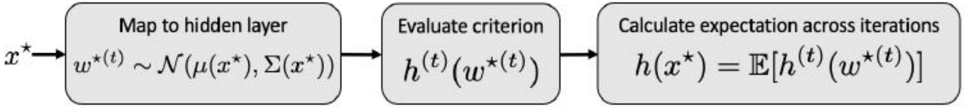
Criteria (IMSE/ALC) are calculated for $w^{\star(t)}$ and averaged across iterations.


$x^\star$
Map to hidden layer
$w^{\star(t)} \sim \mathcal{N}(\mu(x^\star), \Sigma(x^\star))$
Evaluate criterion
$h^{(t)}(w^{\star(t)})$
Calculate expectation across iterations
$h(x^\star) = \mathbb{E}[h^{(t)}(w^{\star(t)})]$

Active Learning for DGPs

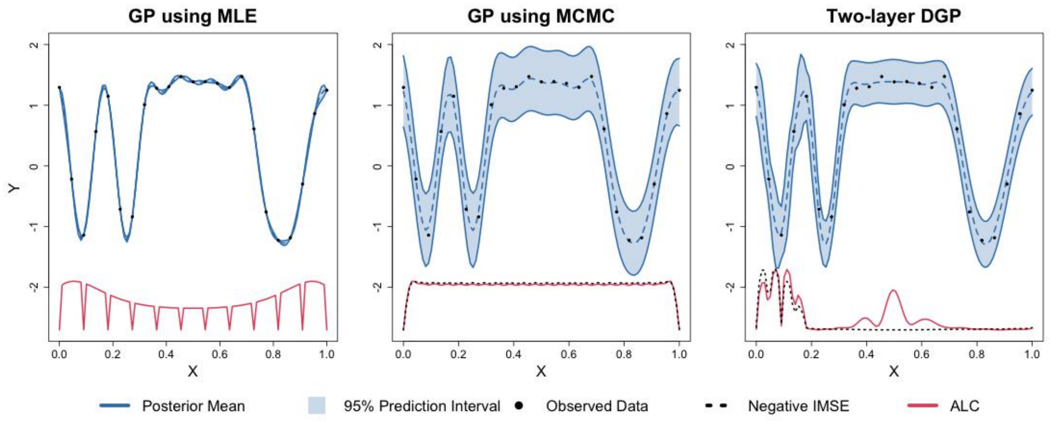
Novel inputs $x^\star$ are mapped to hidden layer $w^{\star(t)}$ using typical GP prediction.

Criteria (IMSE/ALC) are calculated for $w^{\star(t)}$ and averaged across iterations.


Map to hidden layer
$w^{\star(t)} \sim \mathcal{N}(\mu(x^\star), \Sigma(x^\star))$
Evaluate criterion
$h^{(t)}(w^{\star(t)})$
Calculate expectation across iterations
$h(x^\star) = \mathbb{E}[h^{(t)}(w^{\star(t)})]$
$x^\star$

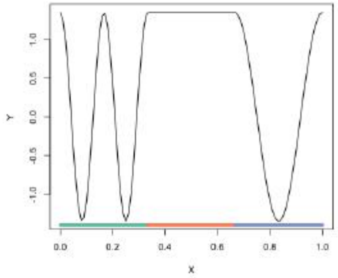
```
R> fit <- fit_two_layer(x, y)
R> imse <- IMSE(fit, x_candidates)
R> alc <- ALC(fit, x_candidates)
```

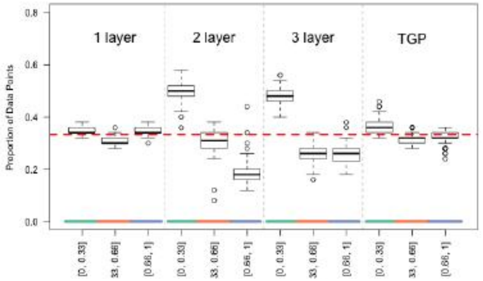
DGPs depart from space filling and outperform on RMSE/SCORE

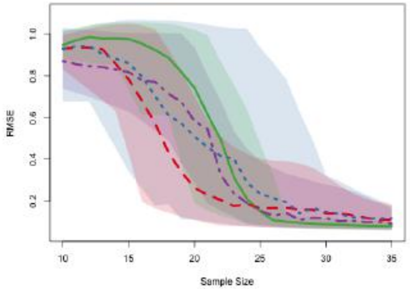


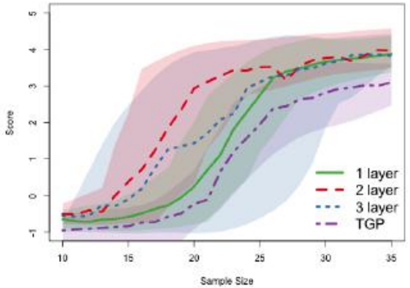
▸ plot of ESS samples

DGPs depart from space filling and outperform on RMSE/SCORE



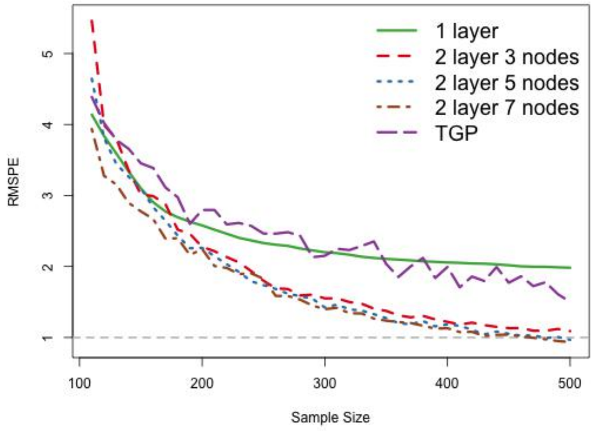


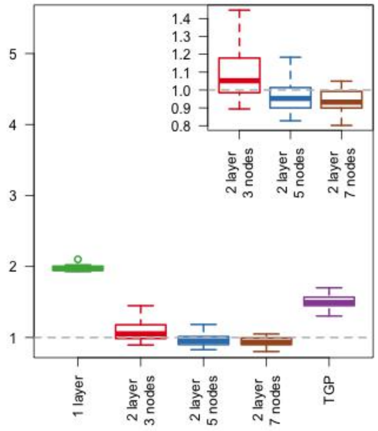




Satellite drag computer simulation

- *Test Particle Monte Carlo (TPM)* simulator developed at LANL (Sun et al., 2019)
- Inputs: 7 configuration variables, satellite mesh, atmospheric composition
- Goal: RMSPE below 1% **starting on a restricted domain**





Active Learning for DGPs - Summary

- Why?
  - When computer simulations are expensive, the "budget" of evaluations is limited
- What?
  - Sequential selection of inputs using greedy acquisition criteria
  - IMSE or ALC (see Gramacy, Sauer, & Wycoff, 2022 for Expected Improvement)
- How?
  - Map inputs through hidden layers and evaluate criterion on mapped values
  - Sequential selections depart from space-filling and focus on regions of interest

Statistical models are only as good as their data

While a DGP has the flexibility to *address* non-stationarity, the data must *reveal* it.

- Strategically choose input configurations to maximize learning from a limited budget (Sauer, Gramacy & Higdon, 2022; Gramacy, Sauer & Wycoff, 2022).

Statistical models are only as good as their data

While a DGP has the flexibility to *address* non-stationarity, the data must *reveal* it.

- Strategically choose input configurations to maximize learning from a limited budget (Sauer, Gramacy & Higdon, 2022; Gramacy, Sauer & Wycoff, 2022).
- Deploy a space filling design that is large enough to pick up on changes in the response surface (Sauer, Cooper & Gramacy, 2022).

Statistical models are only as good as their data

While a DGP has the flexibility to *address* non-stationarity, the data must *reveal* it.

- Strategically choose input configurations to maximize learning from a limited budget (Sauer, Gramacy & Higdon, 2022; Gramacy, Sauer & Wycoff, 2022).
- Deploy a space filling design that is large enough to pick up on changes in the response surface (Sauer, Cooper & Gramacy, 2022).

Large datasets present computational bottlenecks for GP inference ($\mathcal{O}(n^3)$).

$$\mathcal{L}(Y \mid X) \propto |\Sigma(X)|^{-1/2} \exp\left(-\frac{1}{2} Y^\top \Sigma(X)^{-1} Y\right)$$

These are compounded in a Bayesian DGP setting.

Inducing points are popular, but not effective

Competing implementations for DGP inference ...

- Variational inference (Damianou & Lawrence, 2012; Salimbeni & Deisenroth, 2017; Marmin & Filippone, 2022)
- Expectation propogation (Bui et al., 2016)
- Hamiltonian Monte Carlo sampling (Havasi et al., 2018)

All (but one) use **inducing point** approximations to handle large data sizes (Snelson & Ghahramani, 2006; Banerjee et al., 2008):

- observe covariance through fixed set of "knots" which are tricky to place and result in blurry predictions (Garton et al., 2020; Wu et al., 2022).

Marmin & Filippone (2022) utilize **random feature expansions**.

Vecchia approximation from conditional distributions

Any joint distribution may be represented as a product of conditional distributions, i.e.

$$f(y_3, y_2, y_1) = f(y_3 \mid y_2, y_1) f(y_2 \mid y_1) f(y_1).$$

Vecchia approximation from conditional distributions

Any joint distribution may be represented as a product of conditional distributions, i.e.

$$f(y_3, y_2, y_1) = f(y_3 \mid y_2, y_1) f(y_2 \mid y_1) f(y_1).$$

In general,

$$\mathcal{L}(Y) = \prod_{i=1}^{n} \mathcal{L}\left(y_i \mid Y_{c(i)}\right) \quad \text{for} \quad c_0 = \emptyset \quad \text{and} \quad c_i = \{1, 2, \ldots, i-1\} \;\; \forall \;\; i = 2, \ldots, n.$$

The Vecchia approximation (Vecchia, 1988) instead takes the subset

$$c_i \subset \{1, 2, \ldots, i-1\} \quad \text{of size} \quad |c_i| = \min(m, i-1).$$

Vecchia approximation of GPs

In a typical "shallow" GP setting we have

$$\mathcal{L}(Y) = \prod_{i=1}^{n} \mathcal{L}\left(y_i \mid Y_{c(i)}\right),$$

where

$$\mathcal{L}(y_i \mid Y_{c(i)}) \sim \mathcal{N}_1(\mu_i(X), \sigma_i^2(X)) \quad \text{for} \quad \begin{aligned} B_i(X) &= \Sigma(x_i, X_{c(i)})\Sigma(X_{c(i)})^{-1} \\ \mu_i(X) &= B_i(X)Y_{c(i)} \\ \sigma_i^2(X) &= \Sigma(x_i) - B_i(X)\Sigma(X_{c(i)}, x_i). \end{aligned}$$

This converts an $\mathcal{O}(n^3)$ computation into n-many $\mathcal{O}(m^3)$ computations.

Stein et al., 2004; Datta et al., 2016; Stroud et al., 2017; Finley et al., 2019; Katzfuss & Guinness 2020, 2021

Vecchia approximation induces sparsity in precision matrix

The Cholesky decomposition of the precision matrix is **sparse**.

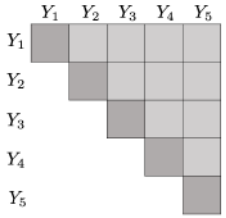
$$Y \sim \mathcal{N}\left(0, \Sigma = Q^{-1} = (UU^\top)^{-1}\right)$$

The upper triangular U matrix has closed-form

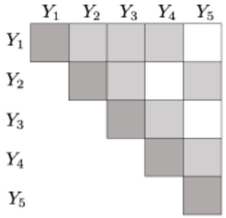
$$U^{ji} = \begin{cases} \frac{1}{\sigma_i(X)} & i = j \\ -\frac{1}{\sigma_i(X)} B_i(X)[\#j \in c(i)] & j \in c(i) \\ 0 & \text{otherwise} \end{cases}$$

whose entries may be populated **in parallel**.

$m = n - 1$



$m = 2$



Katzfuss & Guinness (2021, Proposition 1)

GP tasks hinge on the sparse U matrix

Likelihood Evaluation

$$\log \mathcal{L}(Y) \propto \sum_{i=1}^{n} \log(U^{ii}) - \frac{1}{2} Y^\top U U^\top Y$$

Prior Samples

$$Y^\star = (U^\top)^{-1} z$$
$$z \sim \mathcal{N}(0, \mathbb{I})$$

Posterior Predictions

$$\mathcal{Y} \mid Y, X \sim \mathcal{N}(\mu^\star, \Sigma^\star)$$
$$\mu^\star = -(U_{\mathcal{X}}^\top)^{-1} U_{x,\mathcal{X}}^\top Y$$
$$\Sigma^\star = \left(U_{\mathcal{X}} U_{\mathcal{X}}^\top\right)^{-1}$$

Vecchia-approximated DGPs

Recall our “un-approximated” DGP model

$$Y \mid W \sim \mathcal{N}(0, \Sigma(W)) \qquad W_k \overset{\text{ind}}{\sim} \mathcal{N}(0, \Sigma(X)) \;\; \forall \;\; k = 1, \ldots, p.$$

Vecchia-approximated DGPs

Recall our “un-approximated” DGP model

$$Y \mid W \sim \mathcal{N}\left(0, \Sigma(W)\right) \qquad W_k \overset{\text{ind}}{\sim} \mathcal{N}\left(0, \Sigma(X)\right) \;\; \forall \;\; k = 1, \ldots, p.$$

In our DGP-Vecchia model, we impose a Vecchia approximation at each GP

$$Y \mid W \sim \mathcal{N}\left(0, (U_w U_w^\top)^{-1}\right) \qquad W_k \overset{\text{ind}}{\sim} \mathcal{N}_n\left(0, \left((U_x^{(k)})(U_x^{(k)})^\top\right)^{-1}\right) \;\; \forall \;\; k = 1, \ldots, p.$$

Vecchia-approximated DGPs

Recall our “un-approximated” DGP model

$$Y \mid W \sim \mathcal{N}\left(0, \Sigma(W)\right) \qquad W_k \overset{\text{ind}}{\sim} \mathcal{N}\left(0, \Sigma(X)\right) \;\; \forall \;\; k = 1, \ldots, p.$$

In our DGP-Vecchia model, we impose a Vecchia approximation at each GP

$$Y \mid W \sim \mathcal{N}\left(0, (U_w U_w^\top)^{-1}\right) \qquad W_k \overset{\text{ind}}{\sim} \mathcal{N}_n\left(0, \left((U_x^{(k)})(U_x^{(k)})^\top\right)^{-1}\right) \;\; \forall \;\; k = 1, \ldots, p.$$

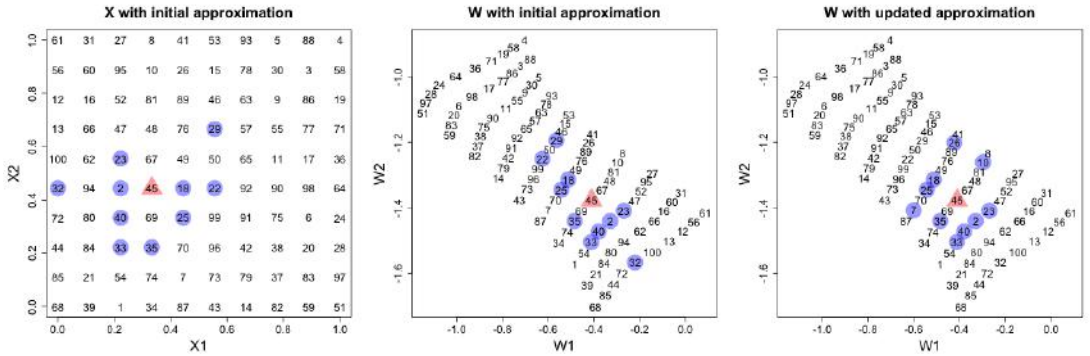
Within our DGP MCMC algorithm, we replace every (i) likelihood evaluation, (ii) prior sample, and (iii) GP prediction with its Vecchia-approximated counterpart.

```
R> fit <- fit_two_layer(x, y, vecchia = TRUE)
R> fit <- predict(fit, x_pred)
```

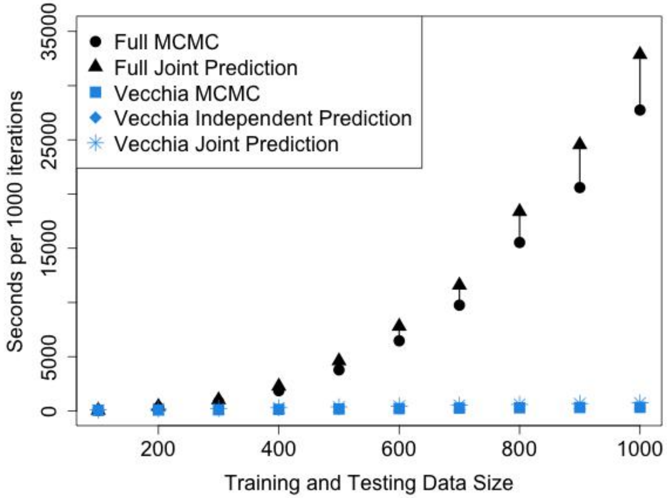
Ordering/conditioning specifications

We utilize

- Random orderings at each Gaussian layer (Guinness, 2018; Wu et al., 2022)
- Nearest-neighbor conditioning sets (Datta et al., 2016)
- Updating of conditioning sets based on learned latent layer warpings



Computation scales linearly

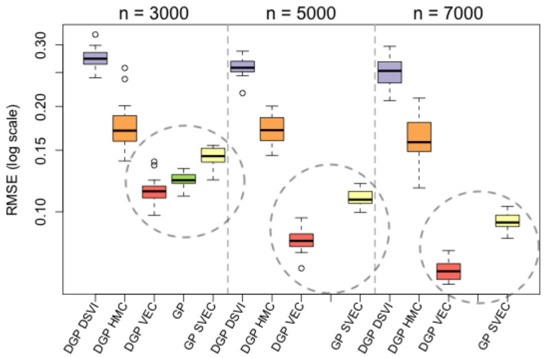


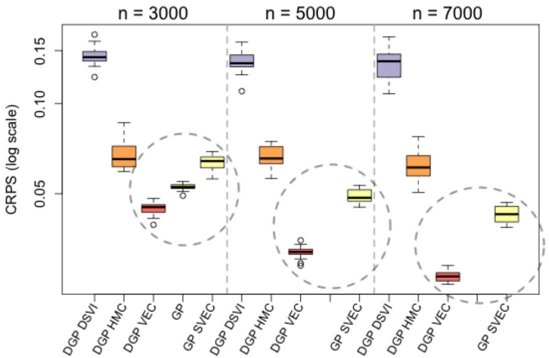
Deep and shallow competitors

- DGP DSVI: "doubly stochastic" VI (Salimbeni & Deisenroth, 2017)
  - utilizes inducing points
- DGP HMC: Hamiltonian Monte Carlo (Havasi et al., 2018)
  - utilizes inducing points
- DGP VEC: our Vecchia-approximated ESS (Sauer, Cooper, & Gramacy, 2022)
- GP: full un-approximated GP (when feasible)
- GP SVEC: Scaled Vecchia "shallow" GP (Katzfuss et al., 2020)

DGP-Vecchia outperforms both deep and shallow competitors

4-dimensional G-function (20 reps)

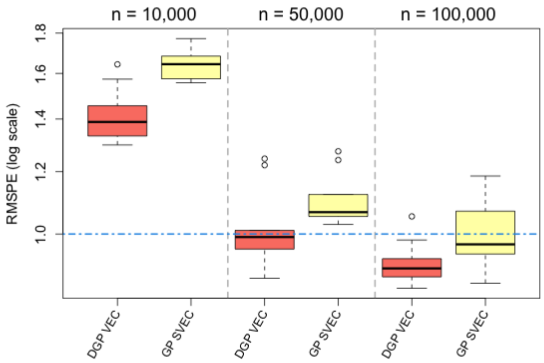


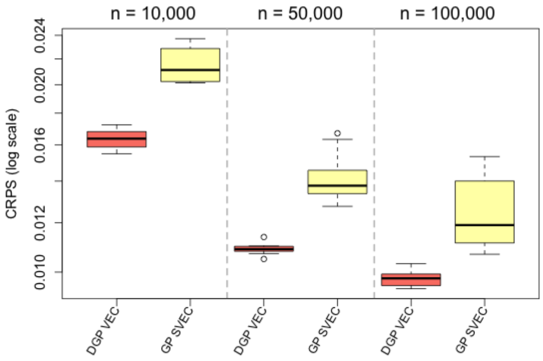


▸ simulation with noise ▸ larger scale simulation

Satellite drag computer simulation

- Same TPM simulator, **bigger data set/domain**
- Same Goal: RMSPE below 1%





DGP DSVI and DGP HMC omitted from figure with RMSPE's 30-35%

Vecchia approximation for DGPs - Summary

- Why?
  - Cubic computational bottlenecks, compounded in DGP MCMC
- What?
  - Imposing sparsity in the precision matrix (and its Cholesky decomposition)
  - Maintaining global scale
- How?
  - Same DGP MCMC scheme with Vecchia-approximation for each GP component
  - Random ordering at each layer
  - Nearest-neighbor conditioning, optionally adjusted based on learned latent layer

Thanks!

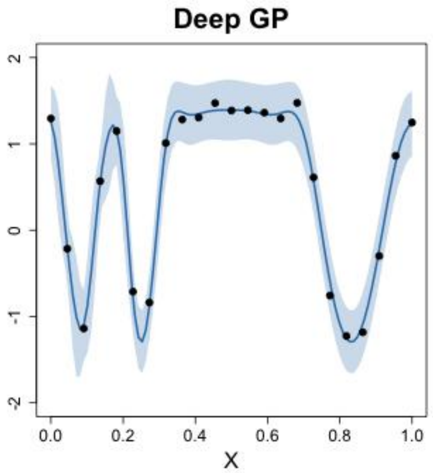
Everything you saw today is supported by

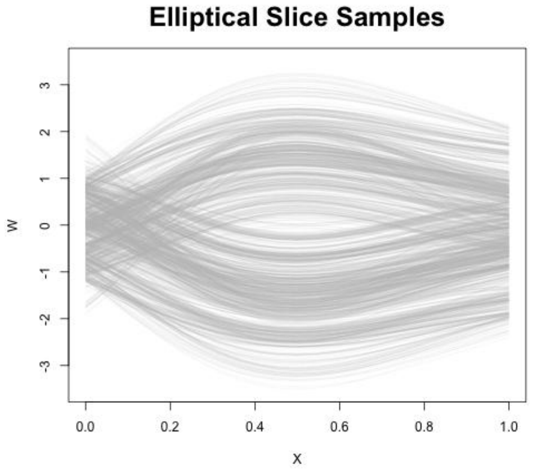
- deepgp for R on CRAN (Sauer, 2022)
- and a git repo of examples:

`https://bitbucket.org/gramacylab/deepgp-ex/`

Many thanks for your attention!

Elliptical slice samples for 1d piecewise function

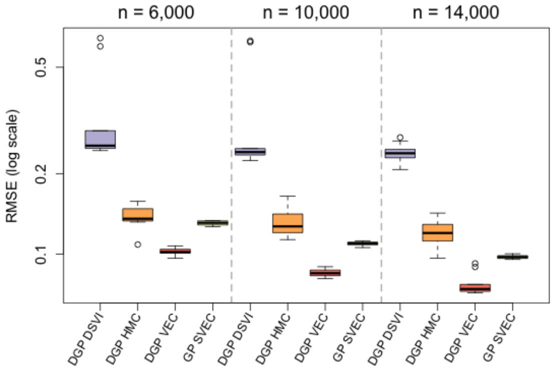


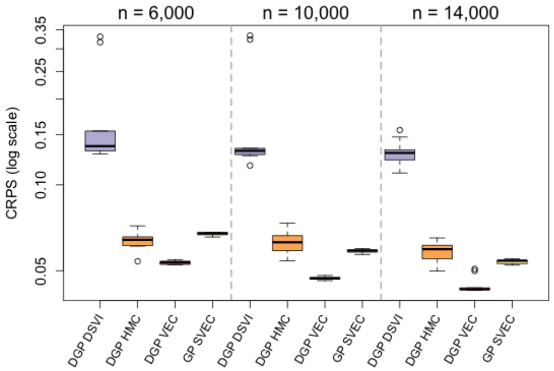


go back

Simulation with noise

4-dimensional G-function with white noise

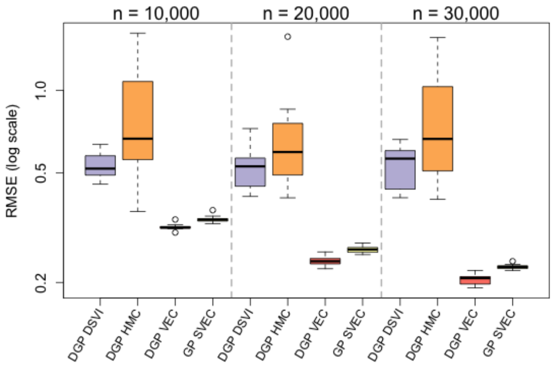


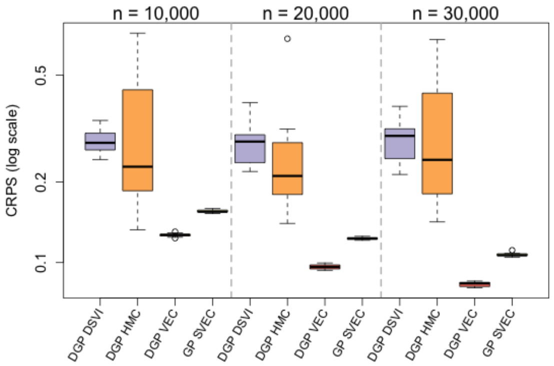


go back

Larger scale simulation

6-dimensional G-function





go back