Stochastic Multilevel Emulation

Jack Kennedy
Kevin Wilson, Daniel Henderson
Newcastle University

19th February 2020

Stats4Grads
Durham University

Outline

Motivation

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
    - Optimisation

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
    - Optimisation
    - Uncertainty/Sensitivity analysis

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
  - Optimisation
  - Uncertainty/Sensitivity analysis
  - Calibration

Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
  - Optimisation
  - Uncertainty/Sensitivity analysis
  - Calibration
- All require many runs of the simulator

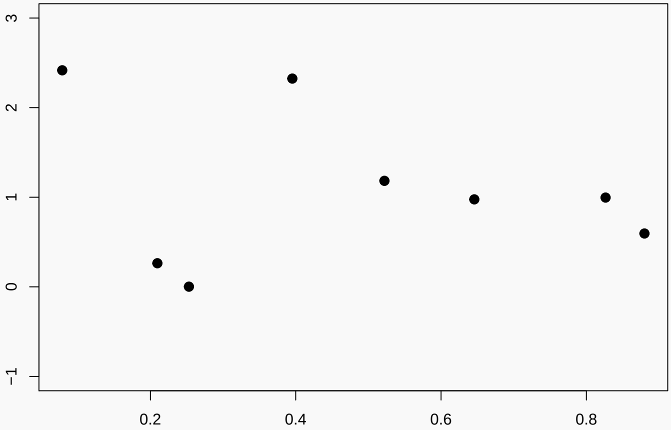
Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
  - Optimisation
  - Uncertainty/Sensitivity analysis
  - Calibration
- All require many runs of the simulator
- If model too expensive — replace it by a Gaussian Process emulator

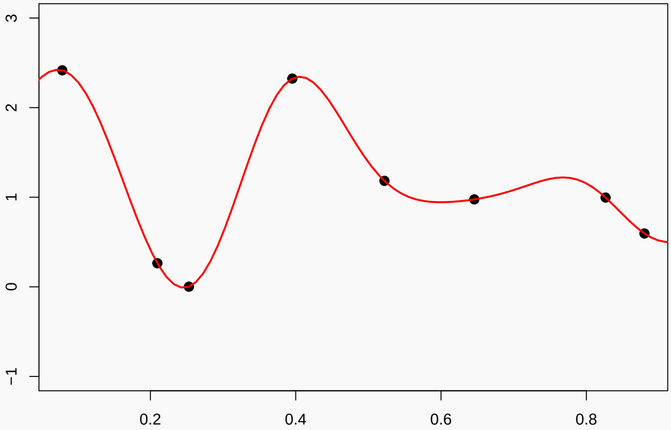
Motivation

- Computer models (simulators) used in all areas of science and engineering to replace infeasible physical experiments
- Common computer model tasks:
    - Optimisation
    - Uncertainty/Sensitivity analysis
    - Calibration
- All require many runs of the simulator
- If model too expensive — replace it by a Gaussian Process emulator
- Theory on emulation of deterministic models is well developed (although still active!)

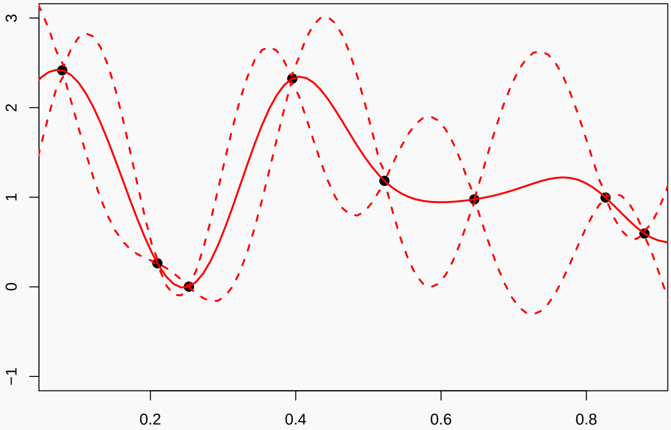
Motivation



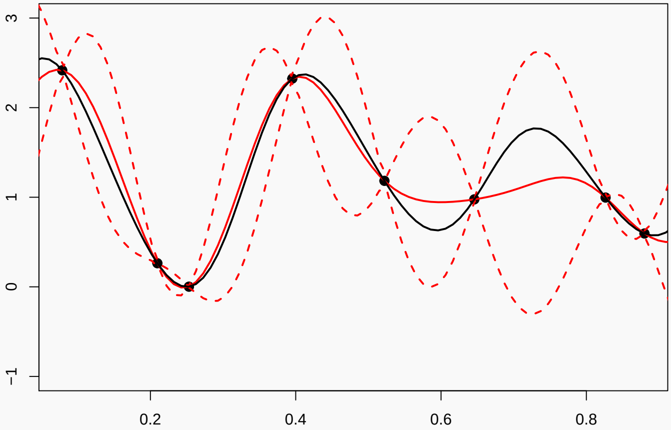
Motivation



Motivation



Motivation



Motivation

- Stochastic computer experiments are becoming more and more popular
    - Agent based models and population dynamics
    - Probabilistic forecasting
    - Catch all for scientific uncertainty

Motivation

- Stochastic computer experiments are becoming more and more popular
    - Agent based models and population dynamics
    - Probabilistic forecasting
    - Catch all for scientific uncertainty
- We can also emulate stochastic simulators
- But is it not easy!

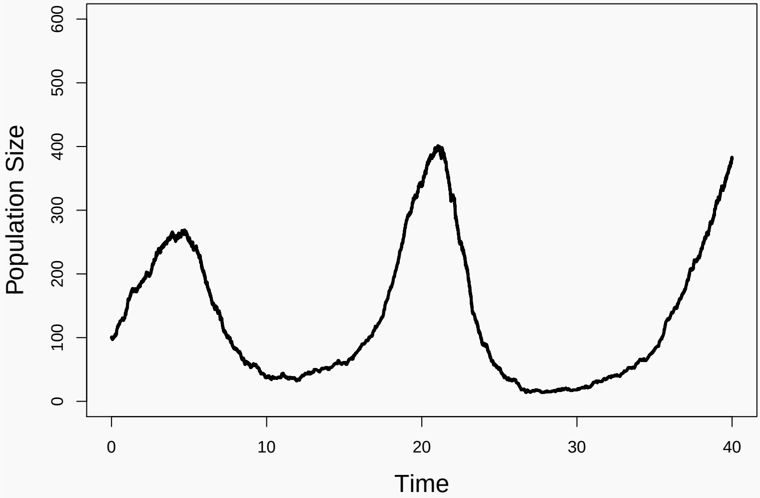
Figure 1: Random randy rabbits

Motivation

Some types of models can be thought as having dependence structure

- Dynamic simulators

Dynamic Simulator

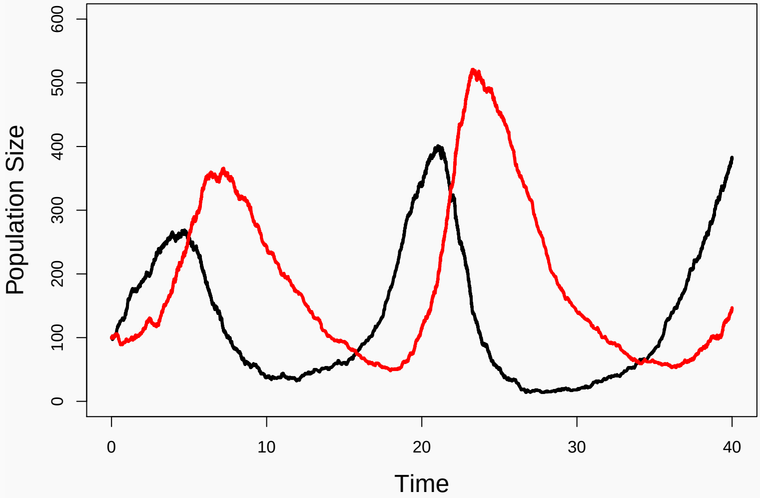


Motivation

Some types of models have a dependence structure

- Dynamic simulators
- Multiple output simulators

Multiple Output Simulator



Motivation

Some types of models have a dependence structure

- Dynamic simulators
- Multiple output simulators
- Different versions of the "same" simulator

Note: Many simulators will fall into at least one of these categories

Motivation

- Many (stochastic) simulators can be made arbitrarily complex
  - Time step or grid coarseness
  - Model topology
  - Additional scientific understanding

Motivation

- Many (stochastic) simulators can be made arbitrarily complex
  - Time step or grid coarseness
  - Model topology
  - Additional scientific understanding
- These cheaper models will be correlated with a more expensive version but more accurate version
- There is an established literature on correlated deterministic models
- I want to utilise *cheap stochastic simulators* in a similar way

Emulation of Stochastic Simulators

Emulation Via Heteroscedastic Gaussian Processes

- The current daddy of stochastic emulation is HetGP

$$\eta(\cdot) \sim \mathcal{GP}\{m(\cdot), C(\cdot,\cdot) + \lambda^2(\cdot)\mathrm{I}\}$$

$$\log(\lambda^2(\cdot)) \sim \mathcal{GP}\{m_\lambda(\cdot), C_\lambda(\cdot,\cdot) + \omega^2\mathrm{I}\}$$

Emulation Via Heteroscedastic Gaussian Processes

- The current daddy of stochastic emulation is HetGP

$$\eta(\cdot) \sim \mathcal{GP}\{m(\cdot), C(\cdot,\cdot) + \lambda^2(\cdot)\mathrm{I}\}$$
$$\log(\lambda^2(\cdot)) \sim \mathcal{GP}\{m_\lambda(\cdot), C_\lambda(\cdot,\cdot) + \omega^2\mathrm{I}\}$$

- Doubly non-parametric
- Gives us a joint (Bayesian) model for the mean and variance
- Allows us to learn about simulator stochasticity without replication
- Incredibly flexible approach to emulation of stochastic simulators

Prediction with HetGP

- Posterior predictive with HetGP is fairly straightforward
- Observe data $\mathcal{D} = (y, \boldsymbol{x})$

Prediction with HetGP

- Posterior predictive with HetGP is fairly straightforward
- Observe data $\mathcal{D} = (y, \boldsymbol{x})$
- Infer hyperparameters Θ in your favourite way

$$\eta(\boldsymbol{x}^*)|\mathcal{D}, \Theta \sim \mathcal{N}\{m^*(\boldsymbol{x}^*), C^*(\boldsymbol{x}^*, \boldsymbol{x}^*) + \lambda^{*2}(\boldsymbol{x}^*)\}$$

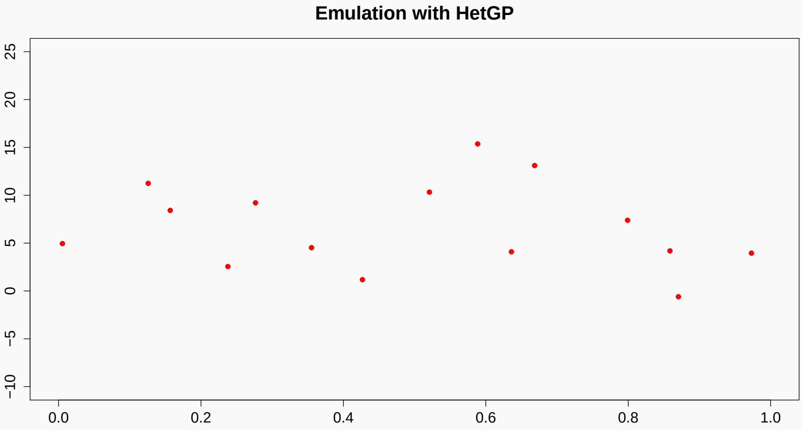
Prediction with HetGP

- Posterior predictive with HetGP is fairly straightforward
- Observe data $\mathcal{D} = (y, \boldsymbol{x})$
- Infer hyperparameters Θ in your favourite way

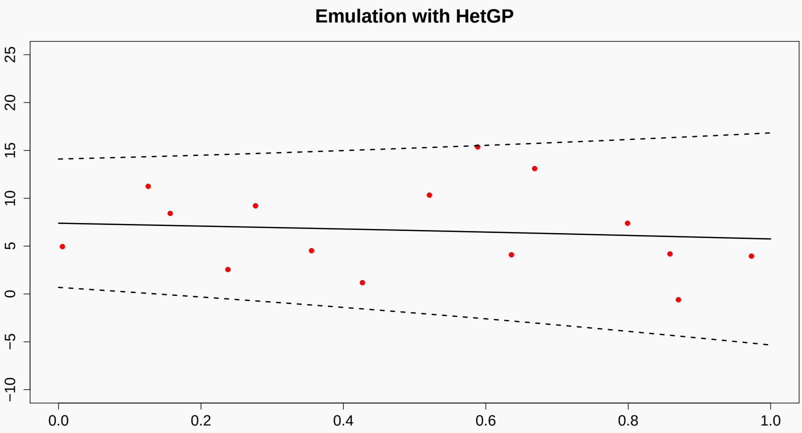
$$\eta(\boldsymbol{x}^*) | \mathcal{D}, \Theta \sim \mathcal{N}\{m^*(\boldsymbol{x}^*), C^*(\boldsymbol{x}^*, \boldsymbol{x}^*) + \lambda^{*2}(\boldsymbol{x}^*)\}$$

- Only tricky task is matrix inversion

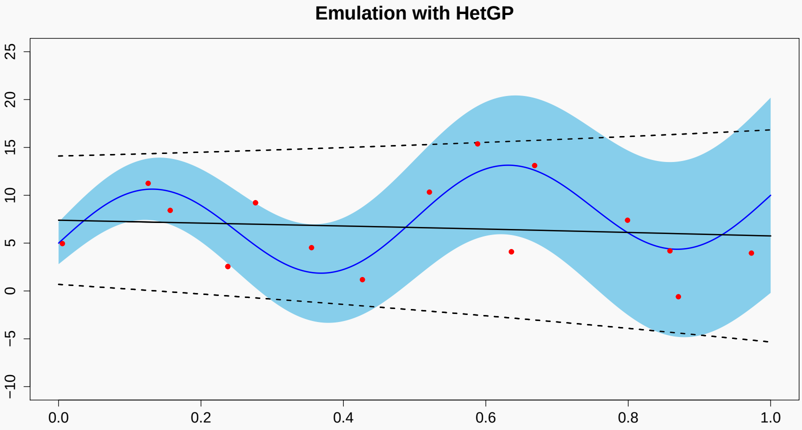
Simple Example



Simple Example



Simple Example

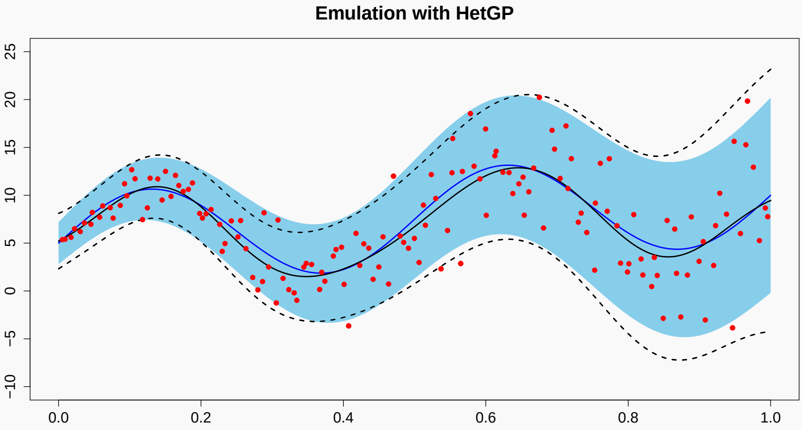


Periodic behaviour interpreted as noise!

How to fix HetGP?

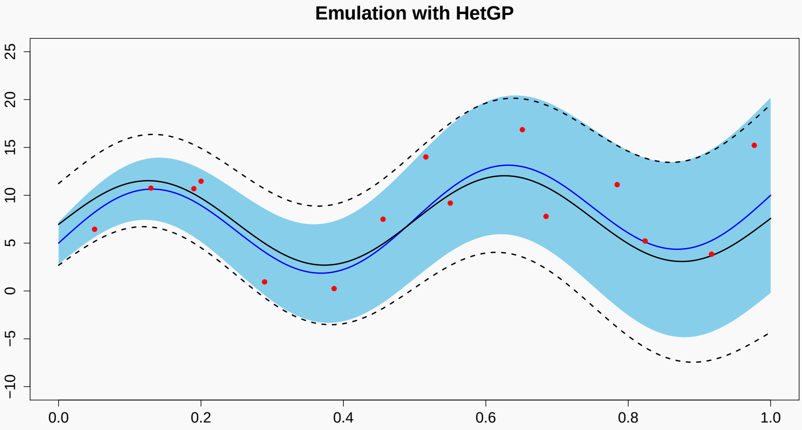
- Currently approaches to “good” emulation with HetGP:
  - Obtaining a massive training budget
  - Careful choice of mean function

HetGP



We can bully HetGP into giving us a good emulator

HetGP



Giving HetGP a “good” mean solves the problem too

Emulation with HetGP

- Currently people deal with this problem by either:
    - Careful choice of mean function

Emulation with HetGP

- Currently people deal with this problem by either:
  - Careful choice of mean function – but lack of knowledge in mean function is why we conduct such experiments!

Emulation with HetGP

- Currently people deal with this problem by either:
  - Careful choice of mean function – but lack of knowledge in mean function is why we conduct such experiments!
  - Obtaining a massive training budget

Emulation with HetGP

- Currently people deal with this problem by either:
  - Careful choice of mean function – but lack of knowledge in mean function is why we conduct such experiments!
  - Obtaining a massive training budget – but the model is expensive!

Emulation with HetGP

- Currently people deal with this problem by either:
    - Careful choice of mean function – but lack of knowledge in mean function is why we conduct such experiments!
    - Obtaining a massive training budget – but the model is expensive!
- More structured approach; use cheap approximations to build a mean function

Emulation with HetGP

- Currently people deal with this problem by either:
  - Careful choice of mean function – but lack of knowledge in mean function is why we conduct such experiments!
  - Obtaining a massive training budget – but the model is expensive!
- More structured approach; use cheap approximations to build a mean function
- Goal: improved emulation of stochastic simulators at fixed training budget

Deterministic Multilevel Emulators

Multilevel Emulators for Deterministic Simulators

- Proposed by Kennedy & O'Hagan (2000) [1]
- Suppose we have "levels" of a simulator η^ℓ where $\ell \in \{1, 2, \ldots, L\}$

[1] Kennedy, M.C. and O'Hagan, A., 2000. Predicting the output from a complex computer code when fast approximations are available. Biometrika, 87(1), pp.1-13.

Multilevel Emulators for Deterministic Simulators

- Proposed by Kennedy & O'Hagan (2000) [1]
- Suppose we have "levels" of a simulator η^ℓ where $\ell \in \{1, 2, \ldots, L\}$
- Each level is more expensive than the last
- Most popular approach is an autoregressive model

$$\eta^1(\cdot) \sim \mathcal{GP}\{m_1(\cdot), C_1(\cdot, \cdot)\}$$
$$\eta^\ell(\boldsymbol{x}) = \rho_\ell \eta^{\ell-1}(\boldsymbol{x}) + \delta^\ell(\boldsymbol{x})$$

- Model δ^ℓ as a GP $\implies$ η^ℓ also a GP

[1] Kennedy, M.C. and O'Hagan, A., 2000. Predicting the output from a complex computer code when fast approximations are available. Biometrika, 87(1), pp.1-13.

Design of ML emulators

- We run the cheap model(s) many more times than the expensive model(s)
- Computationally convenient to have nested design: $X^{\ell} \subseteq X^{\ell-1}$
- If a non-nested design; data imputation mimics a nested design
- We can build designs from scratch and/or use data we have lying around

Picture of Design Matrix

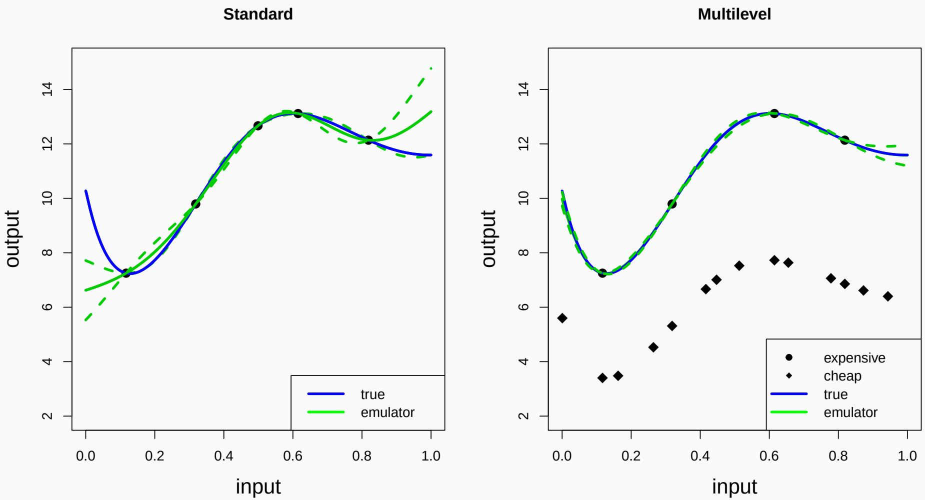
$$\begin{pmatrix} h^C(x_1^C) & 0 \\ h^C(x_2^C) & 0 \\ \vdots & \vdots \\ h^C(x_{N_C}^C) & 0 \\ \rho h^C(x_1^E) & h^E(x_1^E) \\ \rho h^C(x_2^E) & h^E(x_2^E) \\ \vdots & \vdots \\ \rho h^C(x_{N_E}^E) & h^E(x_{N_E}^E) \end{pmatrix} = \begin{pmatrix} h^C(X^C) & 0 \\ \rho h^C(X^E) & h^E(X^E) \end{pmatrix}$$

ML Emulator Likelihood

$$\begin{pmatrix} Y^C \\ Y^E \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} h^C(X^C) & 0 \\ \rho h^C(X^E) & h^E(X^E) \end{pmatrix}\begin{pmatrix} \beta^C \\ \beta^E \end{pmatrix}, \begin{pmatrix} \mathrm{Var}(Y^C) & \mathrm{Cov}(Y^C, Y^E) \\ \mathrm{Cov}(Y^E, Y^C) & \mathrm{Var}(Y^E) \end{pmatrix}\right)$$

$$\begin{aligned}
\mathrm{Var}(Y^C)_{ij} &= \sigma_C^2 \exp\left\{-\sum (d_{ij}^k/\theta_k^C)^2\right\} \\
\mathrm{Var}(Y^E)_{ij} &= \rho^2\sigma_C^2 \exp\left\{-\sum (d_{ij}^k/\theta_k^C)^2\right\} + \sigma_E^2 \exp\left\{-\sum (d_{ij}^k/\theta_k^E)^2\right\} \\
\mathrm{Cov}(Y^C, Y^E)_{ij} &= \rho\sigma_C^2 \exp\left\{-\sum (d_{ij}^k/\theta_k^C)^2\right\}
\end{aligned}$$

Multilevel Emulators for Deterministic Simulators



Stochastic Multilevel Emulators

Stochastic Multilevel Emulation

- Idea: use Kennedy & O'Hagan's autoregressive structure

Stochastic Multilevel Emulation

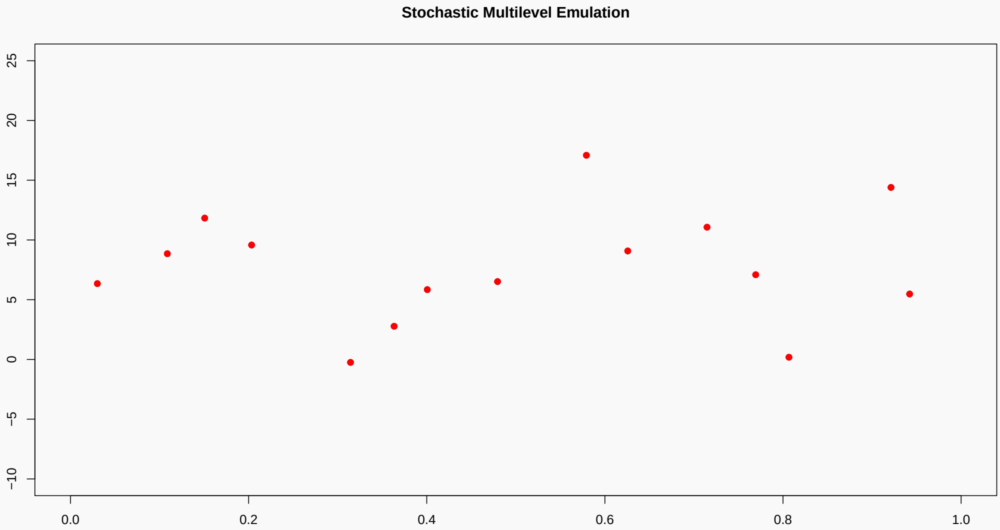
- Idea: use Kennedy & O'Hagan's autoregressive structure
- Two level setup:
  - $\eta^C(\cdot)$ - "cheap" simulator
  - $\eta^E(\cdot)$ - "expensive" simulator
  - $Z_C = \mathrm{E}(\eta^C)$

Stochastic Multilevel Emulation

- Idea: use Kennedy & O'Hagan's autoregressive structure
- Two level setup:
  - $\eta^C(\cdot)$ - "cheap" simulator
  - $\eta^E(\cdot)$ - "expensive" simulator
  - $Z_C = \mathrm{E}(\eta^C)$

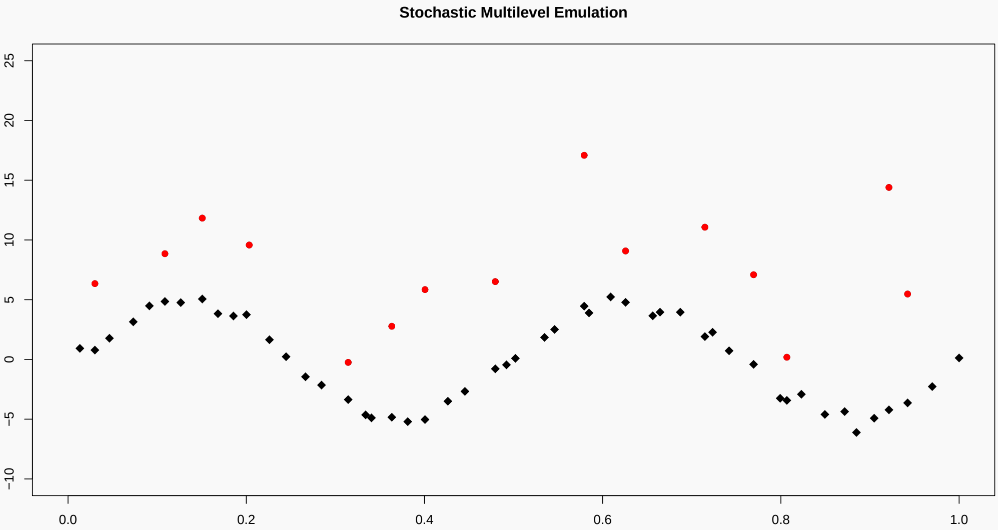
$$
\begin{aligned}
\eta^E(\cdot)|\rho, Z_C(\cdot) &= \rho Z_C(\cdot) + \delta(\cdot) \\
\eta^C(\cdot)|Z_C(\cdot) &= Z_C(\cdot) + \mathcal{N}\left\{0, \lambda_C^2 I_C\right\} \\
Z_C(\cdot) &\sim \mathcal{GP}\left\{m_C(\cdot), C_C(\cdot,\cdot)\right\} \\
\delta(\cdot)|\lambda_E^2(\cdot) &\sim \mathcal{GP}\left\{m_E(\cdot), C_E(\cdot,\cdot) + \lambda_E^2(\cdot) I\right\} \\
\log(\lambda_E^2(\cdot)) &\sim \mathcal{GP}\left\{m_\lambda(\cdot), C_\lambda(\cdot,\cdot) + \omega^2 I_E\right\}
\end{aligned}
$$

Toy Example



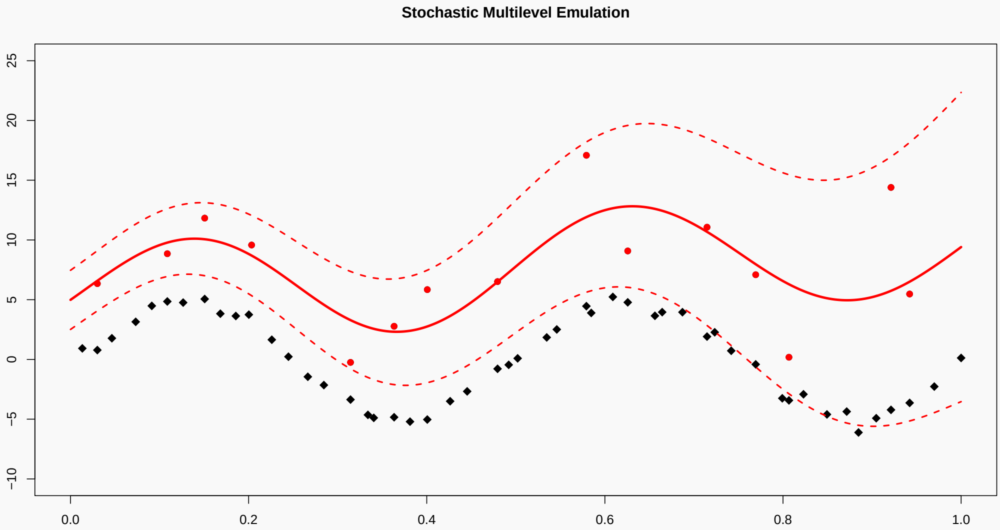
Stochastic Multilevel Emulation
25
20
15
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0

Toy Example



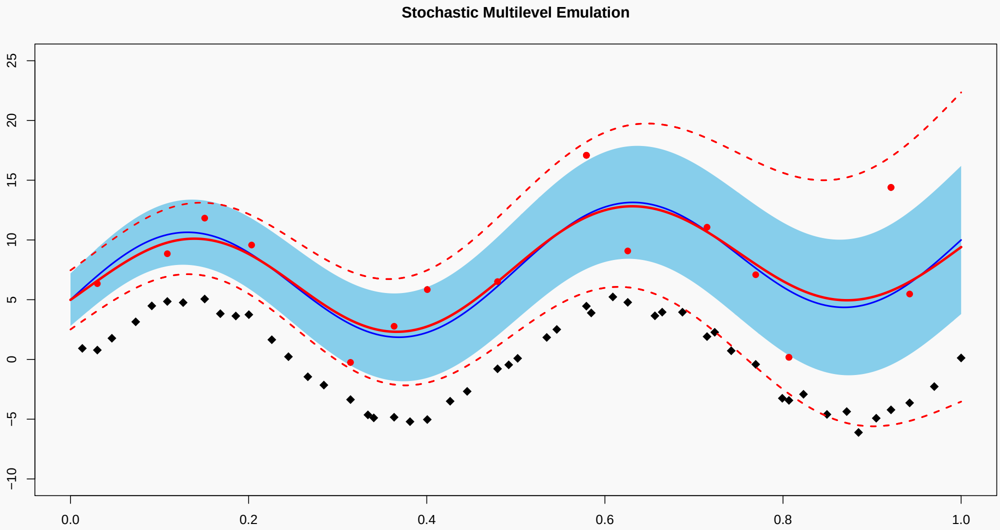
Stochastic Multilevel Emulation
25
20
15
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0

Toy Example



Stochastic Multilevel Emulation
25
20
15
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0

Toy Example



Stochastic Multilevel Emulation
25
20
15
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0

Athena Model

Athena Model

- Athena Model is a stochastic simulation of (large) offshore windfarm performance

Athena Model

- Athena Model is a stochastic simulation of (large) offshore windfarm performance
- Key output: Availability

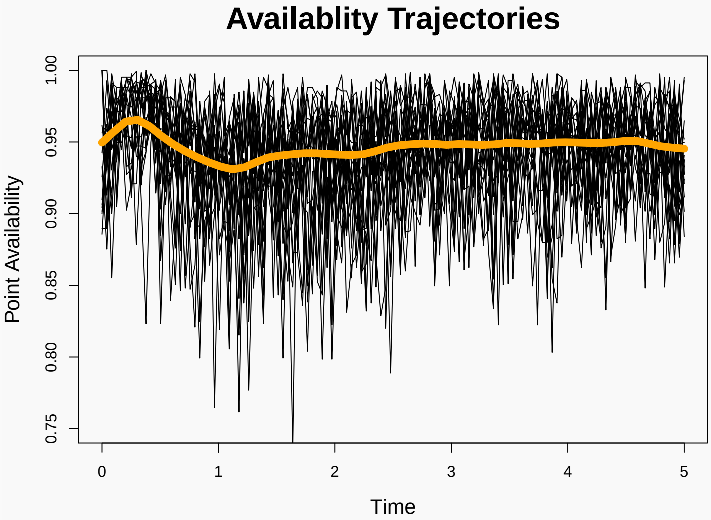
Athena Model

- Athena Model is a stochastic simulation of (large) offshore windfarm performance
- Key output: Availability
- Emulation part of a larger problem: decision analysis under uncertainty

Athena Model

- Athena Model is a stochastic simulation of (large) offshore windfarm performance
- Key output: Availability
- Emulation part of a larger problem: decision analysis under uncertainty
- A reliable emulator is needed to perform a efficient and accurate decision making process

Athena Model



Athena Model

- Accurate simulations take approx 30 mins for one run
- Cheap simulations take under a minute

Athena Model

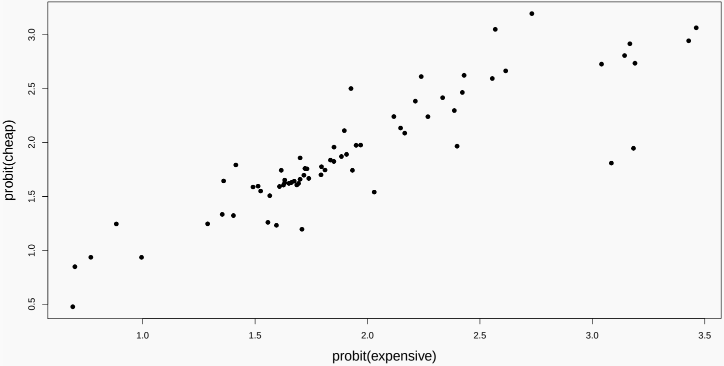
- Accurate simulations take approx 30 mins for one run
- Cheap simulations take under a minute
- Cheap simulation involves using coarse time steps and simplified model topology

Athena Model

- Accurate simulations take approx 30 mins for one run
- Cheap simulations take under a minute
- Cheap simulation involves using coarse time steps and simplified model topology
- Will vary three inputs;
    - Learning rate
    - Cable failure rate
    - Cable repair rate

SML with Athena

- $R^2 = 0.76$ (noisy observations)
- I think it would be a shame to ignore this cheap source of information



SML with Athena

- Okay cool but does it work?

SML with Athena

- Okay cool but does it work?
- Summaries based on 500 unseen validation points

	MSE (probit)	MSE/10^{-4} (original)	Score
HetGP	0.0296	3.72	1835
SML	0.0159	0.694	2025

SML with Athena

- Okay cool but does it work?
- Summaries based on 500 unseen validation points

	MSE (probit)	MSE/10^{-4} (original)	Score
HetGP	0.0296	3.72	1835
SML	0.0159	0.694	2025

- These comparisons are based on the same training budget in CPU time

Conclusions

- Shown SML favourable over HetGP in synthetic case and for Athena model
- Would want to know possible failure cases of SML
- Potential bottleneck: many cheap points means inverting larger matrices, want to find a way to minimise this?
- A design rule? E.g. how much time on expensive and how much time on cheap data

Future Work

- I need to adapt Athena to facilitate floating offshore wind turbines
- Ultimate goal: decision making framework for floating offshore windfarms
- Not a clue how I'm going to do that bit!

Cheers!

`j.c.kennedy1@ncl.ac.uk`