Gaussian Processes for Computer Experiments

Jeremy Oakley

School of Mathematics and Statistics, University of Sheffield

www.jeremy-oakley.staff.shef.ac.uk

The University Of Sheffield.

Computer models

- Computer model represented by function

$$y = f(x).$$

  f usually not available in closed form.

Computer models

- Computer model represented by function

$$y = f(x).$$

  f usually not available in closed form.
- f constructed from modeller's understanding of the process.
  - A 'substantive' or 'law-driven' model rather than an empirical/statistical model.

Computer models

- Computer model represented by function

$$y = f(x).$$

  f usually not available in closed form.
- f constructed from modeller's understanding of the process.
  - A 'substantive' or 'law-driven' model rather than an empirical/statistical model.
  - There may be no physical '(x, y) data'.

Computer models

- Computer model represented by function

$$y = f(x).$$

  f usually not available in closed form.
- f constructed from modeller's understanding of the process.
  - A 'substantive' or 'law-driven' model rather than an empirical/statistical model.
  - There may be no physical '(x, y) data'.
- f may be deterministic.

Computer models

- Computer model represented by function

$$y = f(x).$$

  f usually not available in closed form.
- f constructed from modeller's understanding of the process.
  - A 'substantive' or 'law-driven' model rather than an empirical/statistical model.
  - There may be no physical '(x, y) data'.
- f may be deterministic.
- Computer experiment: evaluating f at difference choices of x
  - A 'model run': evaluating f at a single choice of x.

Things we want to do with computer models

1. Optimisation: finding input values to produce desired output values

Things we want to do with computer models

1. Optimisation: finding input values to produce desired output values
2. Calibration/inverse problems: finding input values so that output values match observed data

Things we want to do with computer models

1. Optimisation: finding input values to produce desired output values
2. Calibration/inverse problems: finding input values so that output values match observed data
3. Uncertainty propagation. Suppose there is a true, uncertain input X. What is the distribution of $Y = f(X)$?

Things we want to do with computer models

1. Optimisation: finding input values to produce desired output values
2. Calibration/inverse problems: finding input values so that output values match observed data
3. Uncertainty propagation. Suppose there is a true, uncertain input X. What is the distribution of $Y = f(X)$?
4. (Global) sensitivity analysis. Continuing (3), if $X = (X_1, \ldots, X_d)$, how do elements of X contribute to uncertainty in Y?

Things we want to do with computer models

1. Optimisation: finding input values to produce desired output values
2. Calibration/inverse problems: finding input values so that output values match observed data
3. Uncertainty propagation. Suppose there is a true, uncertain input X. What is the distribution of $Y = f(X)$?
4. (Global) sensitivity analysis. Continuing (3), if $X = (X_1, \ldots, X_d)$, how do elements of X contribute to uncertainty in Y?

New(ish) buzzword: "Uncertainty Quantification" (UQ)

Gaussian processes for computer models

- Computer models can be computationally expensive: takes long time to calculate $f(x)$ for a single x.

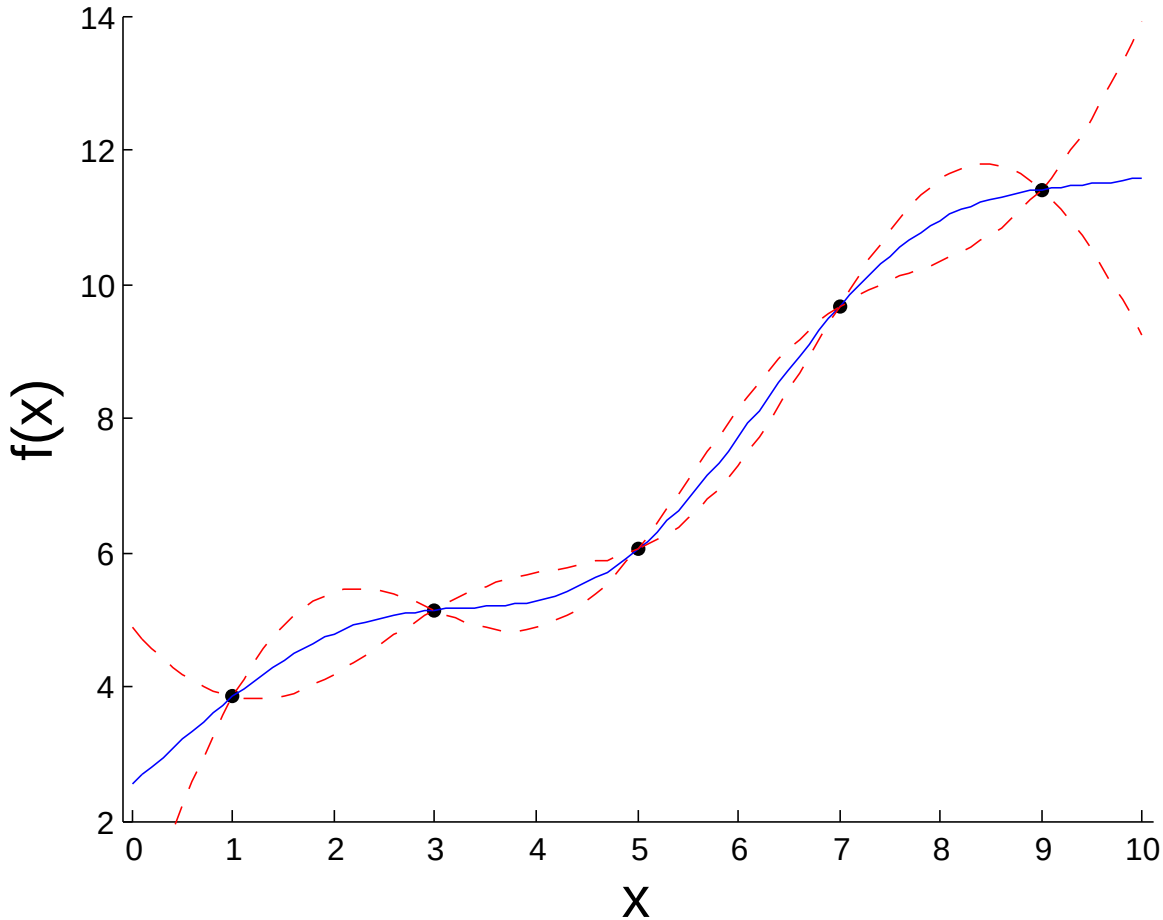
Gaussian processes for computer models

- Computer models can be computationally expensive: takes long time to calculate $f(x)$ for a single x.
- Given limited model runs $y_1 = f(x_1), \ldots, y_n = f(x_n)$, wish to know $y = f(x)$ for many other x values

Gaussian processes for computer models

- Computer models can be computationally expensive: takes long time to calculate $f(x)$ for a single x.
- Given limited model runs $y_1 = f(x_1), \ldots, y_n = f(x_n)$, wish to know $y = f(x)$ for many other x values
- You'll have seen by now that GPs are good for this sort of thing!

Gaussian processes for computer models



Computer experiments: some considerations

Computer experiments: some considerations

- f is often deterministic

Computer experiments: some considerations

- f is often deterministic
- Usually working with/aiming for small training datasets

Computer experiments: some considerations

- f is often deterministic
- Usually working with/aiming for small training datasets
- Can choose the training inputs: experimental design problem

Computer experiments: some considerations

- f is often deterministic
- Usually working with/aiming for small training datasets
- Can choose the training inputs: experimental design problem
- Can sometimes work with 'multi-level' computer models, eg, a fast and coarse model and a slow and accurate model of the same system

Past and current research

- First(?) use of a Gaussian process for a computer model:
  Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. Statistical Science, 4, 409-435.

Past and current research

- First(?) use of a Gaussian process for a computer model:
  Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. Statistical Science, 4, 409-435.
- Bayesian version:
  Currin, C., Mitchell, T., Morris, M. and Ylvisaker, D. (1991). Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments Journal of the American Statistical Association, 86, 953-963.

Past and current research

- First(?) use of a Gaussian process for a computer model:
  Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. Statistical Science, 4, 409-435.
- Bayesian version:
  Currin, C., Mitchell, T., Morris, M. and Ylvisaker, D. (1991). Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments Journal of the American Statistical Association, 86, 953-963.
- GPs for optimisation:
  Jones, D.R., Schonlau, M. and Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions, Journal of Global Optimization, 13, 455-492.

Past and current research

- First(?) use of a Gaussian process for a computer model:
  Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. Statistical Science, 4, 409-435.
- Bayesian version:
  Currin, C., Mitchell, T., Morris, M. and Ylvisaker, D. (1991). Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments Journal of the American Statistical Association, 86, 953-963.
- GPs for optimisation:
  Jones, D.R., Schonlau, M. and Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions, Journal of Global Optimization, 13, 455-492.
- The MUCM community and toolkit
  `www.mucm.ac.uk`
  `www.mucm.ac.uk/toolkit`

Gaussian process emulators

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f
- Need more than an estimate; need to quantify uncertainty, as sample of training data may be small

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f
- Need more than an estimate; need to quantify uncertainty, as sample of training data may be small
- Naturally handled within a Bayesian framework; no 'true' distribution for f. We treat f as an *uncertain function*.

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f
- Need more than an estimate; need to quantify uncertainty, as sample of training data may be small
- Naturally handled within a Bayesian framework; no 'true' distribution for f. We treat f as an *uncertain function*.
- Emulator does not *replace* the model; arguably the term is redundant

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f
- Need more than an estimate; need to quantify uncertainty, as sample of training data may be small
- Naturally handled within a Bayesian framework; no 'true' distribution for f. We treat f as an *uncertain function*.
- Emulator does not *replace* the model; arguably the term is redundant
  - In a computer experiment, may want to know $f(x_1), \ldots, f(x_N)$, but can only observe $f(x_1), \ldots, f(x_n)$, with $n < N$.

Gaussian process emulators

- Various different names for the same thing: surrogate models, response surface models, meta-models, emulators
- Emulator: a probability distribution for the function f
- Need more than an estimate; need to quantify uncertainty, as sample of training data may be small
- Naturally handled within a Bayesian framework; no 'true' distribution for f. We treat f as an *uncertain function*.
- Emulator does not *replace* the model; arguably the term is redundant
  - In a computer experiment, may want to know $f(x_1), \ldots, f(x_N)$, but can only observe $f(x_1), \ldots, f(x_n)$, with $n < N$.
  - As part of the analysis, we work with $p\{f(x_{n+1}), \ldots, f(x_N) | f(x_1), \ldots, f(x_n)\}$, which we get from the emulator.

GP emulator modelling choices

GP emulator modelling choices

- Convention is to specify the GP model hierarchically, with separate mean and covariance functions.

$$f(x) = h(x)^T \beta + Z(x)$$

GP emulator modelling choices

- Convention is to specify the GP model hierarchically, with separate mean and covariance functions.

$$f(x) = h(x)^T \beta + Z(x)$$

- $h(x)^T \beta$: the mean function, eg, for scalar x, could have

$$h(x)^T \beta = \beta_0 + \beta_1 x.$$

GP emulator modelling choices

- Convention is to specify the GP model hierarchically, with separate mean and covariance functions.

$$f(x) = h(x)^T \beta + Z(x)$$

- $h(x)^T \beta$: the mean function, eg, for scalar x, could have

$$h(x)^T \beta = \beta_0 + \beta_1 x.$$

- $Z(.)$: a zero-mean GP (usually stationary), sometimes called the *residual process*

GP emulator modelling choices

- Convention is to specify the GP model hierarchically, with separate mean and covariance functions.

$$f(x) = h(x)^T \beta + Z(x)$$

- $h(x)^T \beta$: the mean function, eg, for scalar x, could have

$$h(x)^T \beta = \beta_0 + \beta_1 x.$$

- $Z(.)$: a zero-mean GP (usually stationary), sometimes called the *residual process*
- Can think of the role of $Z(.)$ as 'improving' a parametric model

GP emulator modelling choices

- Convention is to specify the GP model hierarchically, with separate mean and covariance functions.

$$f(x) = h(x)^T\beta + Z(x)$$

- $h(x)^T\beta$: the mean function, eg, for scalar x, could have

$$h(x)^T\beta = \beta_0 + \beta_1 x.$$

- $Z(.)$: a zero-mean GP (usually stationary), sometimes called the *residual process*
- Can think of the role of $Z(.)$ as 'improving' a parametric model
- One aim of the mean function is to ensure stationary GP is adequate

- Have

$$E[Z(x)] = 0$$

and

$$Cov(Z(x), Z(x')|\sigma^2, \theta) = \sigma^2 c_\theta(x, x')$$

- Have

$$E[Z(x)] = 0$$

and

$$Cov(Z(x), Z(x')|\sigma^2, \theta) = \sigma^2 c_\theta(x, x')$$

with, eg, for $x = (x_1, \ldots, x_d)$

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Have

$$E[Z(x)] = 0$$

and

$$Cov(Z(x), Z(x')|\sigma^2, \theta) = \sigma^2 c_\theta(x, x')$$

with, eg, for $x = (x_1, \ldots, x_d)$

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Commonly assume

$$p(\beta, \sigma^2) \propto \frac{1}{\sigma^2}$$

- Have

$$E[Z(x)] = 0$$

and

$$Cov(Z(x), Z(x')|\sigma^2, \theta) = \sigma^2 c_\theta(x, x')$$

with, eg, for $x = (x_1, \ldots, x_d)$

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Commonly assume

$$p(\beta, \sigma^2) \propto \frac{1}{\sigma^2}$$

- Given training data (and θ), posterior distribution of f is a t-process (β and σ^2 integrated out of posterior analytically)

Equivalent ways to parameterise a GP

Equivalent ways to parameterise a GP

- Consider, for scalar input x,

$$f(x) = x\beta + Z(x)$$

with $\beta|\sigma^2 \sim N(0, v\sigma^2)$ (with v chosen), and $Z(x)$ a zero mean GP as before.

Equivalent ways to parameterise a GP

- Consider, for scalar input x,

$$f(x) = x\beta + Z(x)$$

with $\beta|\sigma^2 \sim N(0, v\sigma^2)$ (with v chosen), and $Z(x)$ a zero mean GP as before.

- Then

$$E[f(x)] = 0$$

Equivalent ways to parameterise a GP

- Consider, for scalar input x,

$$f(x) = x\beta + Z(x)$$

with $\beta|\sigma^2 \sim N(0, v\sigma^2)$ (with v chosen), and $Z(x)$ a zero mean GP as before.

- Then

$$E[f(x)] = 0$$

and

$$Cov(f(x_1), f(x_2)|\sigma^2, \theta) = v\sigma^2 x_1 x_2 + \sigma^2 c_\theta(x_1, x_2) \quad (1)$$

Equivalent ways to parameterise a GP

- Consider, for scalar input x,

$$f(x) = x\beta + Z(x)$$

with $\beta|\sigma^2 \sim N(0, v\sigma^2)$ (with v chosen), and $Z(x)$ a zero mean GP as before.

- Then

$$E[f(x)] = 0$$

and

$$Cov(f(x_1), f(x_2)|\sigma^2, \theta) = v\sigma^2 x_1 x_2 + \sigma^2 c_\theta(x_1, x_2) \qquad (1)$$

- If we *specify* a zero mean GP with covariance function (1), for fixed v, model is equivalent to the hierarchical GP with the mean function $x\beta$.

Choice of mean function

Choice of mean function

- Can be important if training data set small; gaps between points can be large.

Choice of mean function

- Can be important if training data set small; gaps between points can be large.
- Usually choose simple linear form. If $x = (x_1, \ldots, x_d)$,

$$h(x)^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d.$$

Choice of mean function

- Can be important if training data set small; gaps between points can be large.
- Usually choose simple linear form. If $x = (x_1, \ldots, x_d)$,

$$h(x)^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d.$$

- Some claim $h(x)^T \beta = \beta_0$ constant form works better...

Choice of mean function

- Can be important if training data set small; gaps between points can be large.
- Usually choose simple linear form. If $x = (x_1, \ldots, x_d)$,

$$h(x)^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d.$$

- Some claim $h(x)^T \beta = \beta_0$ constant form works better...
- ...others claim better to include higher order polynomial terms (quadratics, interactions)

Choice of mean function

- Can be important if training data set small; gaps between points can be large.
- Usually choose simple linear form. If $x = (x_1, \ldots, x_d)$,

$$h(x)^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d.$$

- Some claim $h(x)^T \beta = \beta_0$ constant form works better...
- ...others claim better to include higher order polynomial terms (quadratics, interactions)
- For multi level model case, fast model can be used as a prior mean for the slow model

Choice of mean function

- Can be important if training data set small; gaps between points can be large.
- Usually choose simple linear form. If $x = (x_1, \ldots, x_d)$,

$$h(x)^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d.$$

- Some claim $h(x)^T \beta = \beta_0$ constant form works better...
- ...others claim better to include higher order polynomial terms (quadratics, interactions)
- For multi level model case, fast model can be used as a prior mean for the slow model
- Using a second GP (with noise) for the mean can help deal with nonstationarity:
  Ba, S. and Joseph, R. V. (2012). Composite Gaussian process models for emulating expensive functions. The Annals of Applied Statistics 6, 1838-1860

Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{ -\sum_{i=1}^{d} \left(\frac{x_i - x_i'}{\theta_i} \right)^2 \right\}$$

Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{ -\sum_{i=1}^{d} \left(\frac{x_i - x_i'}{\theta_i} \right)^2 \right\}$$

- Maximum likelihood probably most popular

Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{ -\sum_{i=1}^{d} \left(\frac{x_i - x_i'}{\theta_i} \right)^2 \right\}$$

- Maximum likelihood probably most popular
- Some authors do 'full Bayes' using MCMC

Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Maximum likelihood probably most popular
- Some authors do 'full Bayes' using MCMC
- Others fix the correlation parameters, and include more polynomial terms in the mean

Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Maximum likelihood probably most popular
- Some authors do 'full Bayes' using MCMC
- Others fix the correlation parameters, and include more polynomial terms in the mean
- Importance sampling:

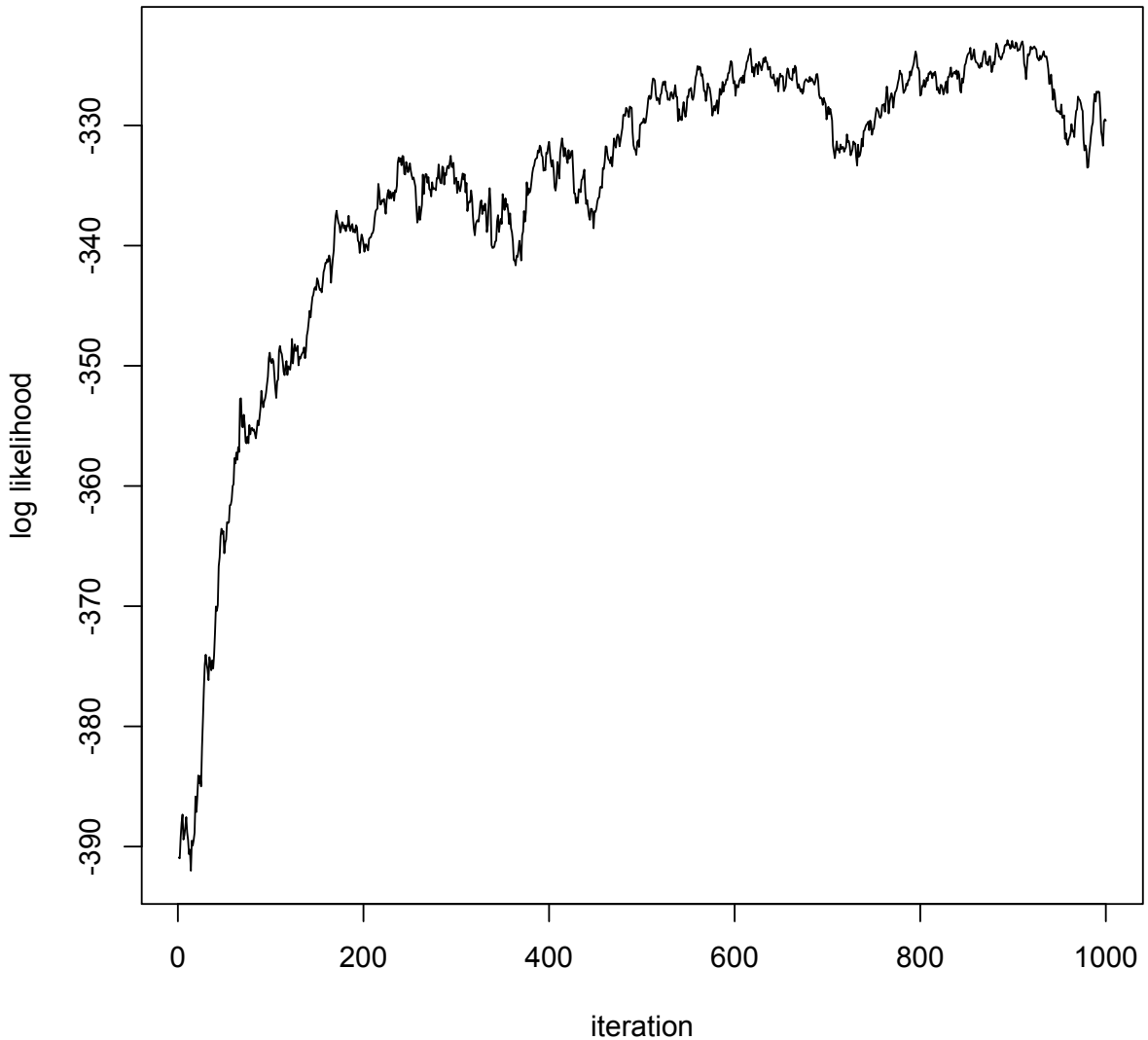
Estimating the correlation function parameters

$x = (x_1, \ldots, x_d)$.

$$c(x, x') = \exp\left\{-\sum_{i=1}^{d}\left(\frac{x_i - x_i'}{\theta_i}\right)^2\right\}$$

- Maximum likelihood probably most popular
- Some authors do 'full Bayes' using MCMC
- Others fix the correlation parameters, and include more polynomial terms in the mean
- Importance sampling:
  Nagy, B., Loeppky, J. L. and Welch, W. J. (2007). Fast Bayesian inference for Gaussian process models.

Initial Gibbs sampler can be helpful for starting an optimiser. Example: 18 input climate model



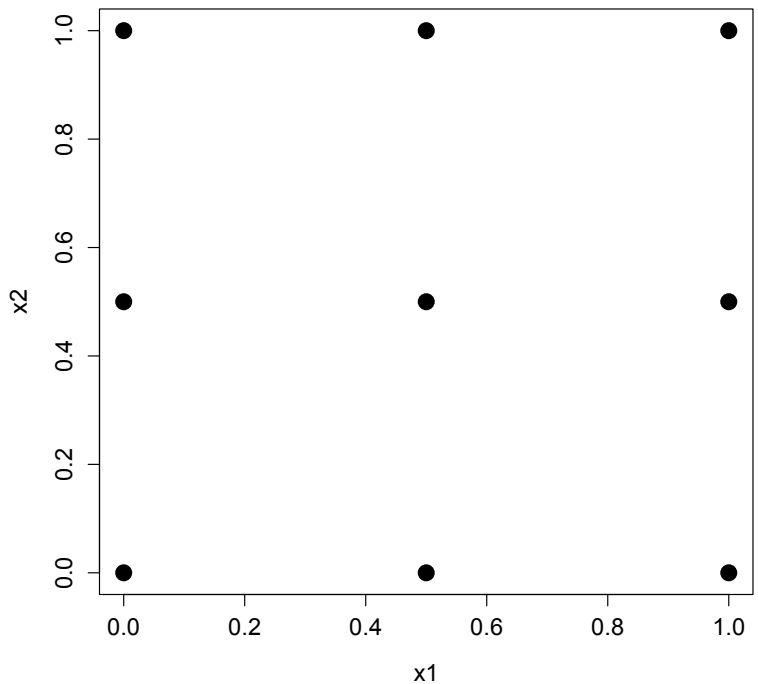
Experimental design

Experimental design

- Often find some inputs uninfluential; product designs can be wasteful

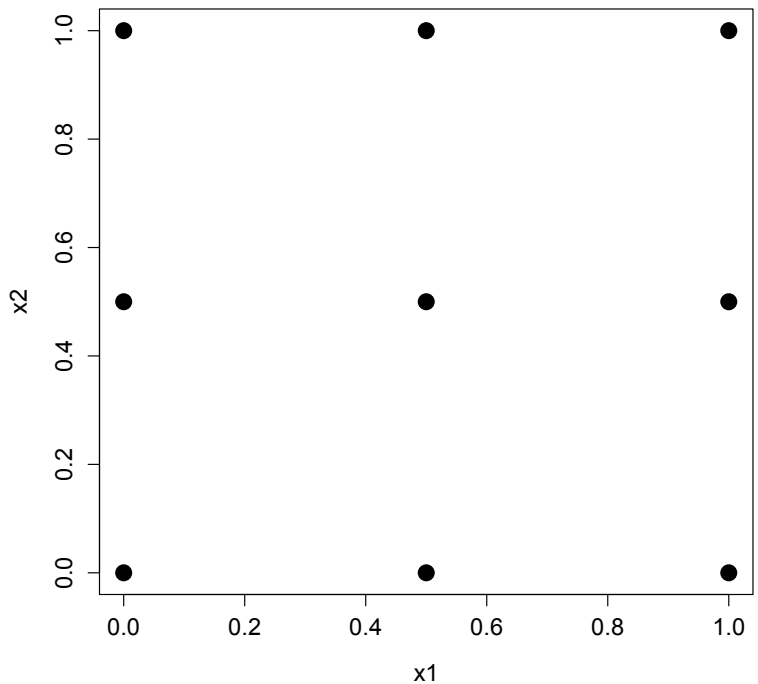
Experimental design

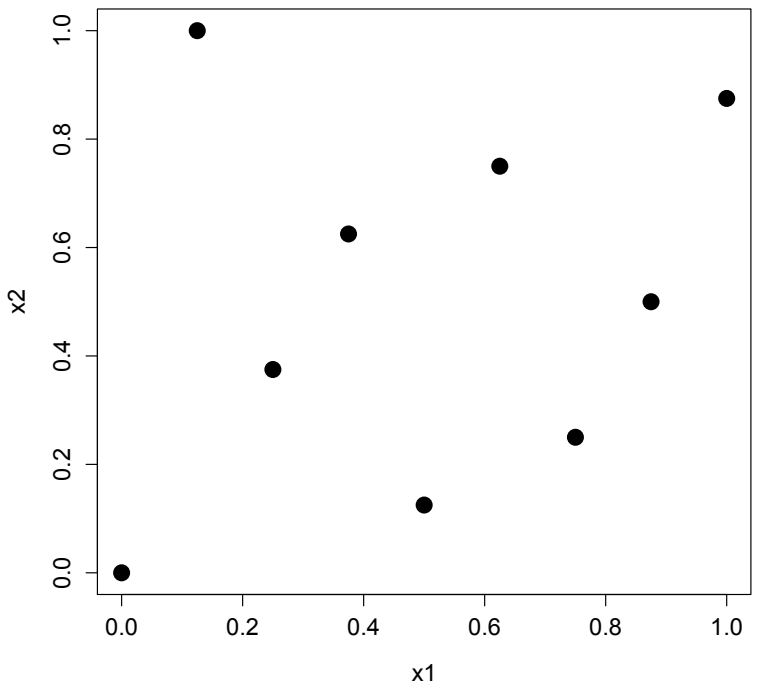
- Often find some inputs uninfluential; product designs can be wasteful



Experimental design

- Often find some inputs uninfluential; product designs can be wasteful

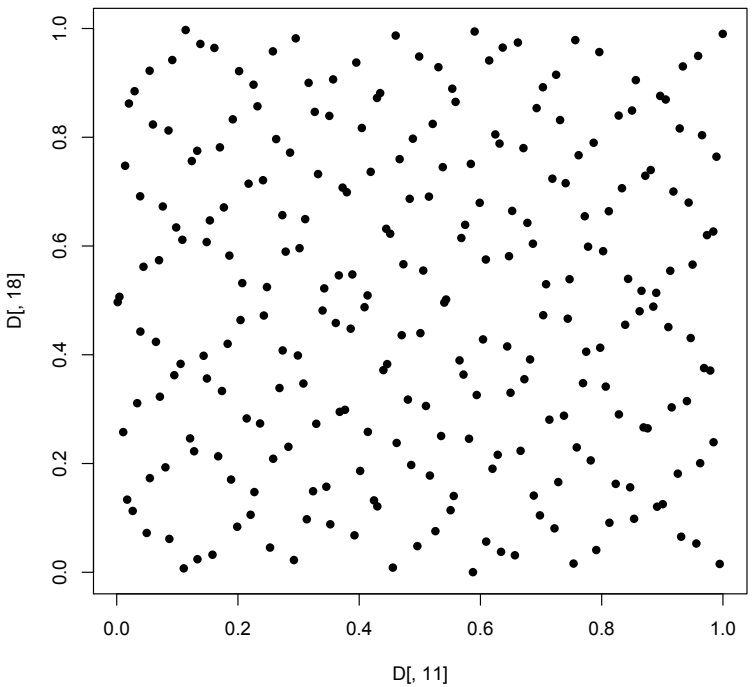




- Space filling designs such as (maximin) Latin Hypercube samples popular

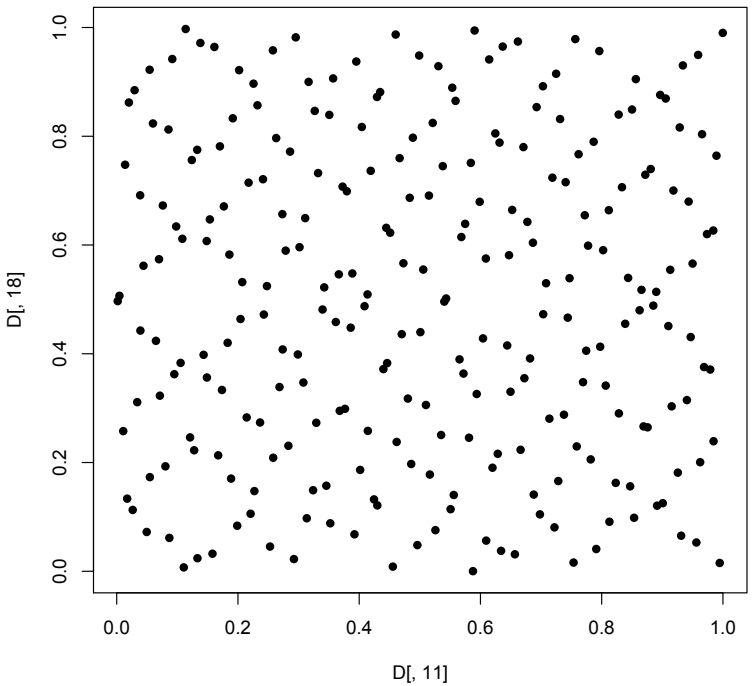
Experimental design

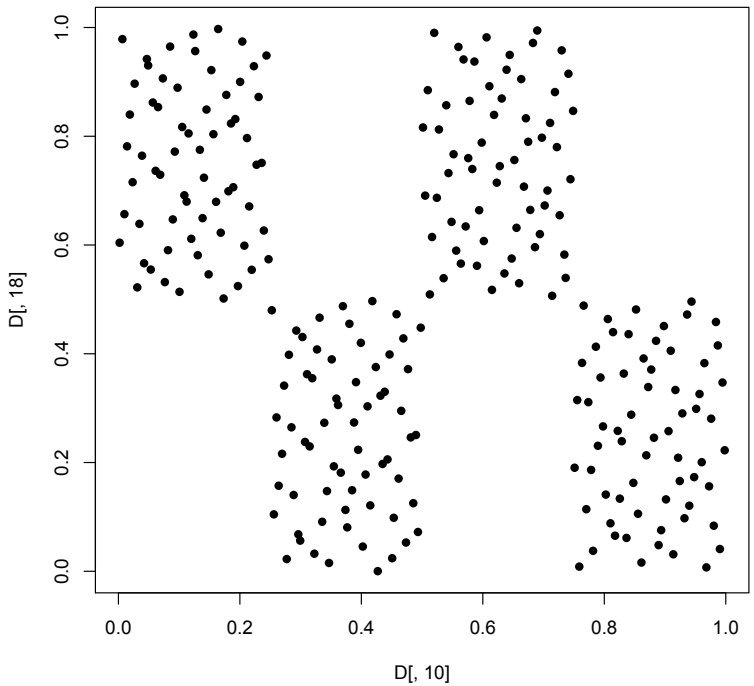
Sobol' sequences also space-filling. Easier for sequential design, but some projections poor.



Experimental design

Sobol' sequences also space-filling. Easier for sequential design, but some projections poor.





Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).
- Example:
  - Suppose $X \sim N(0, 1)$, and either $f(x) = 1 + x$, or $f(x) = x^2$ (we don't know which).

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).
- Example:
  - Suppose $X \sim N(0, 1)$, and either $f(x) = 1 + x$, or $f(x) = x^2$ (we don't know which).
  - If $f(x) = 1 + x$, then $Y = f(X)|f$ has the $N(1, 1)$ distribution

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).
- Example:
  - Suppose $X \sim N(0, 1)$, and either $f(x) = 1 + x$, or $f(x) = x^2$ (we don't know which).
  - If $f(x) = 1 + x$, then $Y = f(X)|f$ has the $N(1, 1)$ distribution
  - If $f(x) = x^2$, then $Y = f(X)|f$ has the χ^2_1 distribution

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).
- Example:
  - Suppose $X \sim N(0, 1)$, and either $f(x) = 1 + x$, or $f(x) = x^2$ (we don't know which).
  - If $f(x) = 1 + x$, then $Y = f(X)|f$ has the $N(1, 1)$ distribution
  - If $f(x) = x^2$, then $Y = f(X)|f$ has the χ^2_1 distribution
  - As f is uncertain, distribution of $Y|f$ is uncertain

Uncertainty propagation

- Suppose that there is a true, uncertain value of the input X
- We specify a probability distribution for X
- What is our uncertainty about $Y = f(X)$?
- We consider inference for the *uncertainty distribution* $p(Y|f)$, itself uncertain. (Distinct from $p(Y)$).
- Example:
  - Suppose $X \sim N(0, 1)$, and either $f(x) = 1 + x$, or $f(x) = x^2$ (we don't know which).
  - If $f(x) = 1 + x$, then $Y = f(X)|f$ has the $N(1, 1)$ distribution
  - If $f(x) = x^2$, then $Y = f(X)|f$ has the χ^2_1 distribution
  - As f is uncertain, distribution of $Y|f$ is uncertain
- By considering uncertainty about $p(Y|f)$, we can consider how inferences might change given more model runs

Consider inference for the uncertain mean

$$E(Y|f) = \int f(x)p_X(x)dx,$$

Consider inference for the uncertain mean

$$E(Y|f) = \int f(x)p_X(x)dx,$$

Normally distributed if f is a GP (O'Hagan, A. (1991) Bayes-Hermite quadrature).

Consider inference for the uncertain mean

$$E(Y|f) = \int f(x)p_X(x)dx,$$

Normally distributed if f is a GP (O'Hagan, A. (1991) Bayes-Hermite quadrature).
Can get closed form expression for mean and variance of $E(Y|f)$ if

- Multivariate normal distribution for X
- Exponentiated quadratic covariance function
- Mean function polynomial in X

Consider inference for the uncertain mean

$$E(Y|f) = \int f(x)p_X(x)dx,$$

Normally distributed if f is a GP (O'Hagan, A. (1991) Bayes-Hermite quadrature).
Can get closed form expression for mean and variance of $E(Y|f)$ if

- Multivariate normal distribution for X
- Exponentiated quadratic covariance function
- Mean function polynomial in X

Haylock, R. G. and O'Hagan, A. (1996). On inference for outputs of computationally expensive algorithms with uncertainty on the inputs, in Bayesian Statistics 5, edited by Bernardo, J. M., Berger, J. O., Dawid, A. P. and Smith, A. F. M., pp. 629-637, Oxford: University Press.

Exploring the uncertainty distribution with emulators: example

Oakley, J. (2004). Estimating percentiles of computer code outputs. Journal of the Royal Statistical Society, Series C, 53, 83-93.

Exploring the uncertainty distribution with emulators: example

Oakley, J. (2004). Estimating percentiles of computer code outputs. Journal of the Royal Statistical Society, Series C, 53, 83-93.

- During storms, excess sewage can spill over into rivers and streams

Exploring the uncertainty distribution with emulators: example

Oakley, J. (2004). Estimating percentiles of computer code outputs. Journal of the Royal Statistical Society, Series C, 53, 83-93.

- During storms, excess sewage can spill over into rivers and streams
- A model $y = f(\mathbf{x})$ (the SIMPOL model) estimates storage volume y required to meet certain environmental standards.

Exploring the uncertainty distribution with emulators: example

Oakley, J. (2004). Estimating percentiles of computer code outputs. Journal of the Royal Statistical Society, Series C, 53, 83-93.

- During storms, excess sewage can spill over into rivers and streams
- A model $y = f(\mathbf{x})$ (the SIMPOL model) estimates storage volume y required to meet certain environmental standards.
- Uncertainty in true input values $\mathbf{X}$ induces uncertainty in the output $Y = f(\mathbf{X})$; WRc wish to know the 95th percentile of $Y|f$

Exploring the uncertainty distribution with emulators: example

Oakley, J. (2004). Estimating percentiles of computer code outputs. Journal of the Royal Statistical Society, Series C, 53, 83-93.

- During storms, excess sewage can spill over into rivers and streams
- A model $y = f(\mathbf{x})$ (the SIMPOL model) estimates storage volume y required to meet certain environmental standards.
- Uncertainty in true input values $\mathbf{X}$ induces uncertainty in the output $Y = f(\mathbf{X})$; WRc wish to know the 95th percentile of $Y|f$
- Can we estimate this percentile efficiently based on a small number of model runs?

- A sequential approach

- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f

- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.

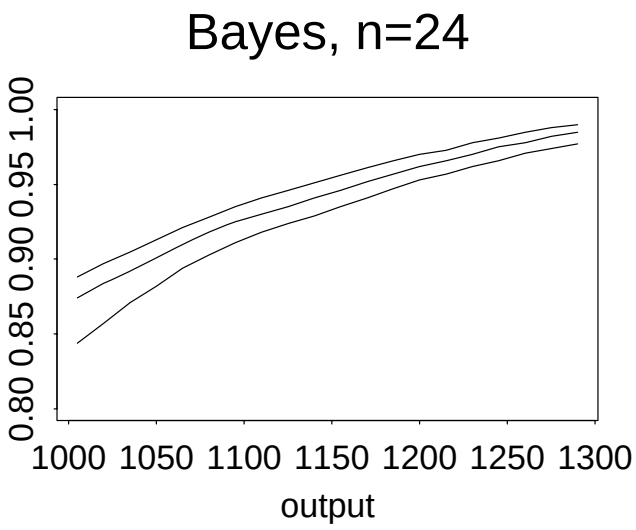
- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.
  3. From (2), identify region of input space R that produces large output values

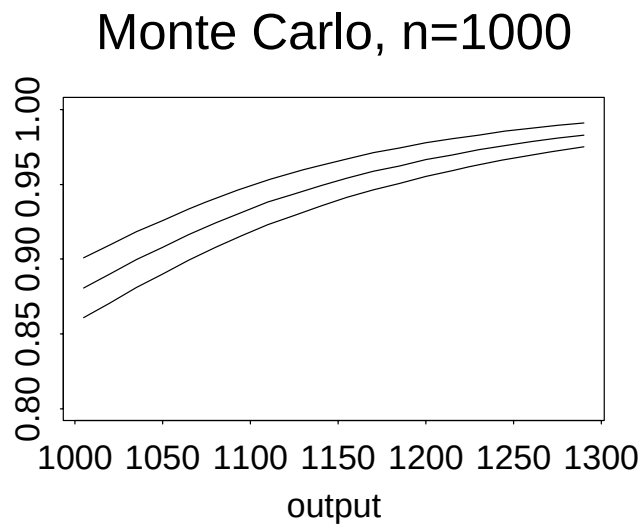
- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.
  3. From (2), identify region of input space R that produces large output values
  4. Run original more times, concentrating input choices in R

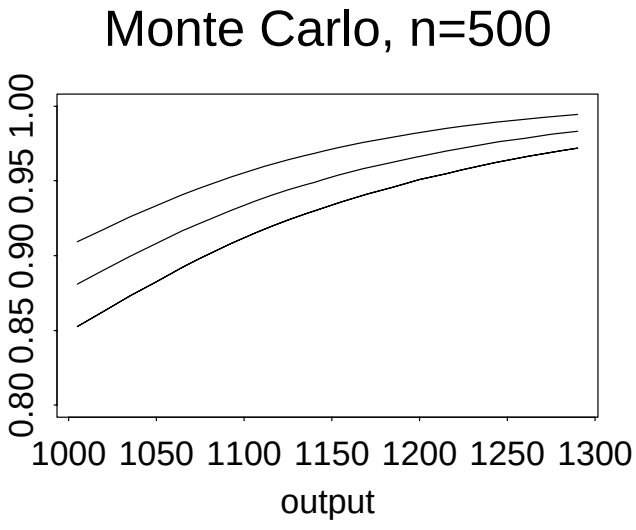
- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.
  3. From (2), identify region of input space R that produces large output values
  4. Run original more times, concentrating input choices in R
  5. Update the emulator to reduce uncertainty about f in input region of interest

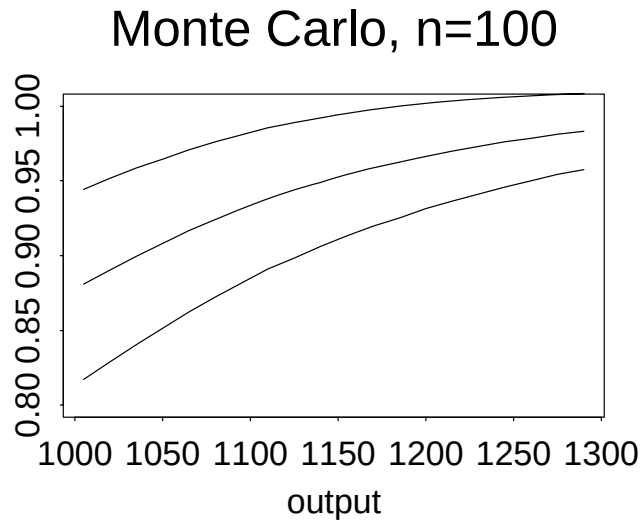
- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.
  3. From (2), identify region of input space R that produces large output values
  4. Run original more times, concentrating input choices in R
  5. Update the emulator to reduce uncertainty about f in input region of interest
- SIMPOL test example, 4 uncertain inputs, 16 runs used for step 1, 8 runs used for step 3.

- A sequential approach
  1. Run the model a small number times, fit emulator to get distribution of f
  2. Simulate approximate realisations of f. For each realisation, use Monte Carlo to estimate 95th percentile, noting the input value.
  3. From (2), identify region of input space R that produces large output values
  4. Run original more times, concentrating input choices in R
  5. Update the emulator to reduce uncertainty about f in input region of interest
- SIMPOL test example, 4 uncertain inputs, 16 runs used for step 1, 8 runs used for step 3.
- Comparison with large Monte Carlo samples.


Bayes, n=24
0.80 0.85 0.90 0.95 1.00
1000 1050 1100 1150 1200 1250 1300
output


Monte Carlo, n=1000
0.80 0.85 0.90 0.95 1.00
1000 1050 1100 1150 1200 1250 1300
output


Monte Carlo, n=500
0.80 0.85 0.90 0.95 1.00
1000 1050 1100 1150 1200 1250 1300
output


Monte Carlo, n=100
0.80 0.85 0.90 0.95 1.00
1000 1050 1100 1150 1200 1250 1300
output

Diagnostics and validation

Diagnostics and validation

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian process emulators. Technometrics 51, 425-438.

Diagnostics and validation

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian process emulators. Technometrics 51, 425-438.

Motivating example:

- We have a model $y = f(x)$, and wish to consider uncertainty in $Y = f(X)$, for uncertain X with distribution G

Diagnostics and validation

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian process emulators. Technometrics 51, 425-438.

Motivating example:

- We have a model $y = f(x)$, and wish to consider uncertainty in $Y = f(X)$, for uncertain X with distribution G
- Monte Carlo approach: sample $x_1, \ldots, x_N$ from G, evaluate $y_1 = f(x_1), \ldots, y_n = f(x_N)$ to obtain sample from Y.

Diagnostics and validation

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian process emulators. Technometrics 51, 425-438.

Motivating example:

- We have a model $y = f(x)$, and wish to consider uncertainty in $Y = f(X)$, for uncertain X with distribution G
- Monte Carlo approach: sample $x_1, \ldots, x_N$ from G, evaluate $y_1 = f(x_1), \ldots, y_n = f(x_N)$ to obtain sample from Y.
- Can use Monte Carlo sample to estimate $E(Y|f)$, $P(Y \leq y|f)$ etc., with 'reliable' associated standard errors for large N.

Diagnostics and validation

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for Gaussian process emulators. Technometrics 51, 425-438.

Motivating example:

- We have a model $y = f(x)$, and wish to consider uncertainty in $Y = f(X)$, for uncertain X with distribution G
- Monte Carlo approach: sample $x_1, \ldots, x_N$ from G, evaluate $y_1 = f(x_1), \ldots, y_n = f(x_N)$ to obtain sample from Y.
- Can use Monte Carlo sample to estimate $E(Y|f)$, $P(Y \leq y|f)$ etc., with 'reliable' associated standard errors for large N.
- Model user limited to small number of runs, so do uncertainty analysis with an emulator.

Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)

Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)
- Intervals show how estimates might change *given further runs of the model*

Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)
- Intervals show how estimates might change *given further runs of the model*
- Can we really be 95% sure $E(Y|f)$ lies in (48,52)?

Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)
- Intervals show how estimates might change *given further runs of the model*
- Can we really be 95% sure $E(Y|f)$ lies in (48,52)?
- Will depend on whether GP emulator is an 'appropriate' statistical model of the computer model f.

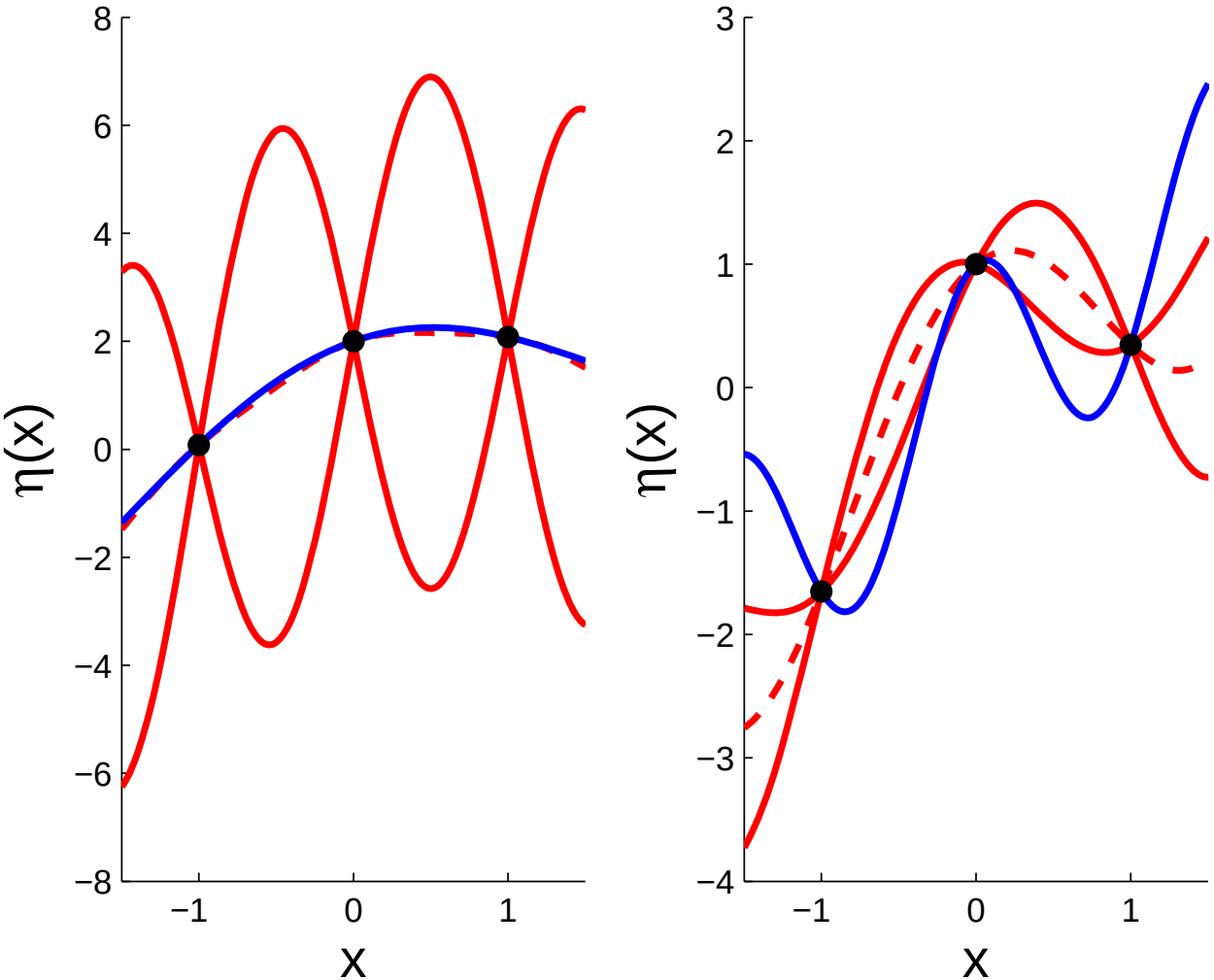
Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)
- Intervals show how estimates might change *given further runs of the model*
- Can we really be 95% sure $E(Y|f)$ lies in (48,52)?
- Will depend on whether GP emulator is an 'appropriate' statistical model of the computer model f.
- Overconfident predictions may lead to poor decisions

Diagnostics and validation: motivating example

- Suppose emulator provides estimate $\hat{E}(Y|f) = 50$, with 95% credible interval (48,52) and $\hat{P}(Y < 70|f) = 0.8$, with 95% credible interval (0.77,0.83)
- Intervals show how estimates might change *given further runs of the model*
- Can we really be 95% sure $E(Y|f)$ lies in (48,52)?
- Will depend on whether GP emulator is an 'appropriate' statistical model of the computer model f.
- Overconfident predictions may lead to poor decisions
- Underconfident predictions may devalue the computer model analysis

Poorly performing emulators



Traditional approach: leave one out CV

- Remove each data point in turn, predict using remaining training data

Traditional approach: leave one out CV

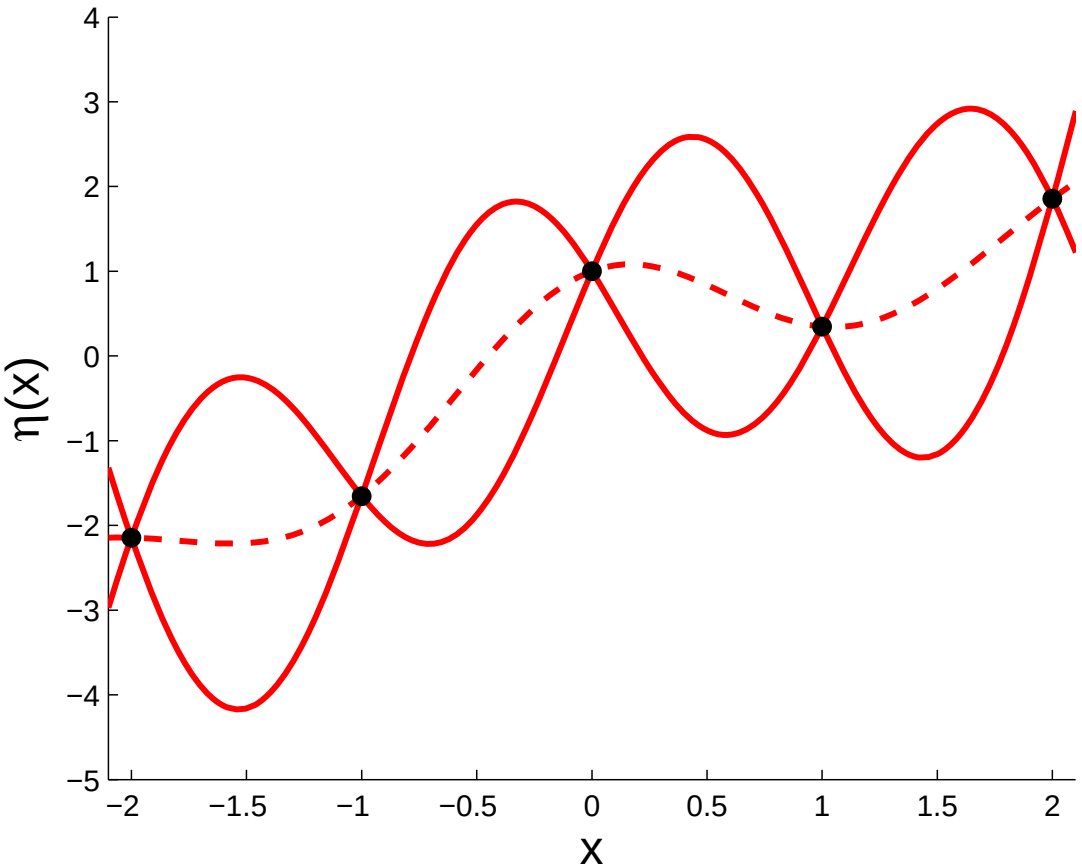
- Remove each data point in turn, predict using remaining training data
- Plot prediction (emulator mean), with error bars, against true value
- Plot standardised prediction errors against predicted values

Traditional approach: leave one out CV

- Remove each data point in turn, predict using remaining training data
- Plot prediction (emulator mean), with error bars, against true value
- Plot standardised prediction errors against predicted values
- Useful, can identify a bad emulator, but
  - ignores correlation in prediction errors
  - hard to judge if uncertainty in emulator is 'appropriate'

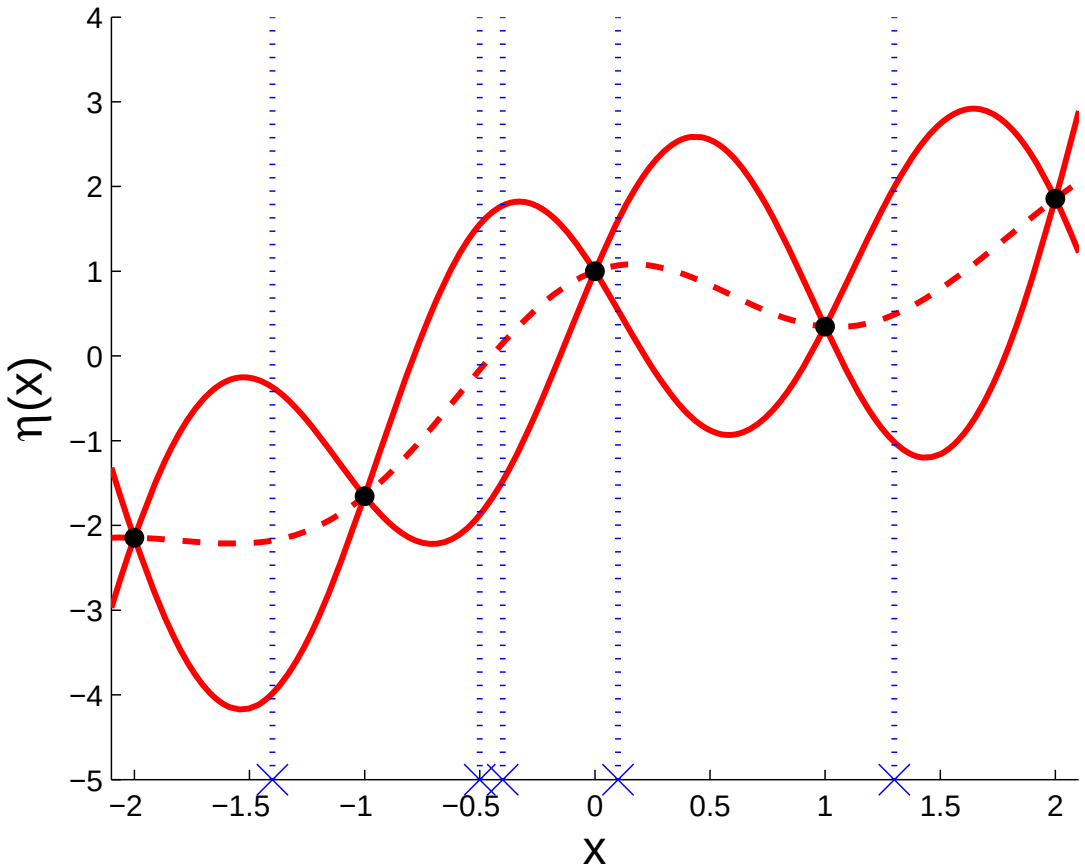
The diagnostic strategy

Run simulator to obtain training data $\mathbf{y}$.
Build emulator given the training data.



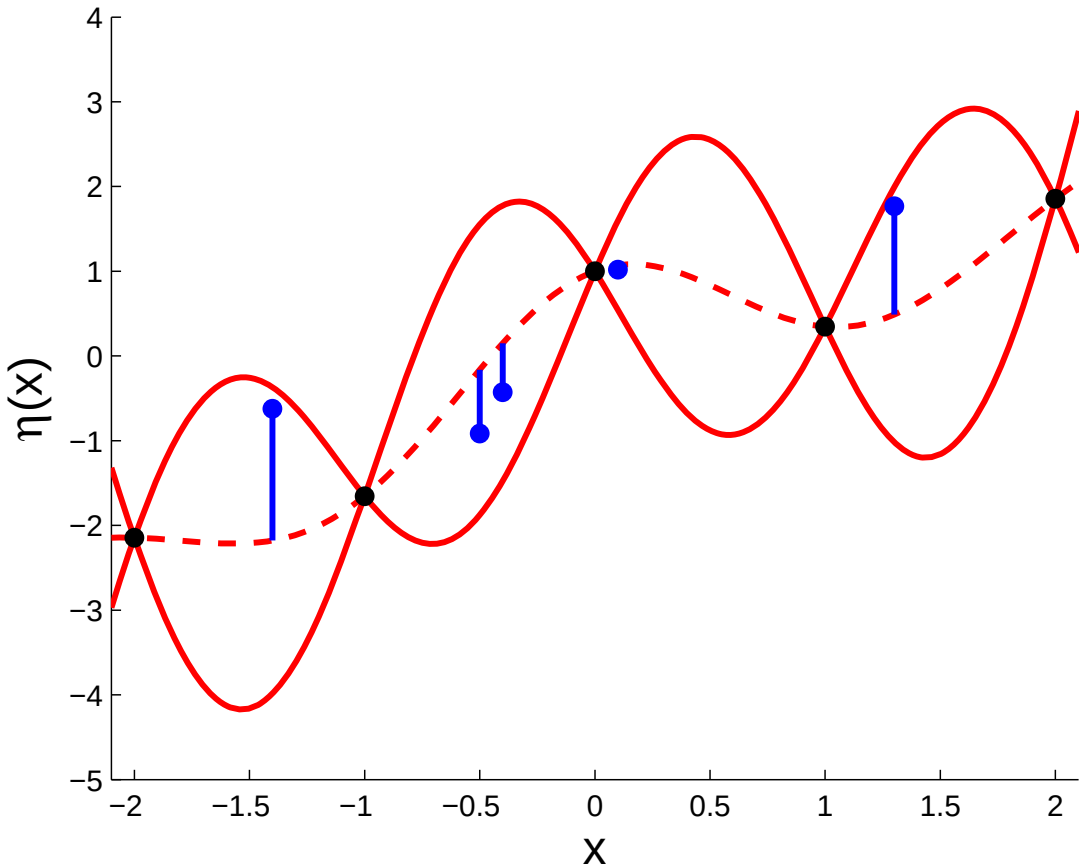
The diagnostic strategy

Select second set validation inputs.
Determine emulator predictions and uncertainty at these inputs.



The diagnostic strategy

Run simulator to obtain validation data $\mathbf{y}^*$.
Observe emulator prediction errors, relative to emulator uncertainty.



Diagnostics with the validation data (1)

- Can combine prediction errors into a single measure of fit using a Mahalanobis distance

$$D_{MD}(\mathbf{y}^*) = \{\mathbf{y}^* - E(\mathbf{y}^*|\mathbf{y})\}^T Var(\mathbf{y}^*|\mathbf{y})^{-1}\{\mathbf{y}^* - E(\mathbf{y}^*|\mathbf{y})\},$$

- How do we know if prediction errors are too large or too small?

Diagnostics with the validation data (1)

- Can combine prediction errors into a single measure of fit using a Mahalanobis distance

$$D_{MD}(\mathbf{y}^*) = \{\mathbf{y}^* - E(\mathbf{y}^*|\mathbf{y})\}^T Var(\mathbf{y}^*|\mathbf{y})^{-1}\{\mathbf{y}^* - E(\mathbf{y}^*|\mathbf{y})\},$$

- How do we know if prediction errors are too large or too small?

The Mahalanobis distance diagnostic

If GP emulator is 'correct', $D_{MD}(\mathbf{y}^*)$ has an F distribution, so compare observed $D_{MD}(\mathbf{y}^*)$ with this distribution to test hypothesis of a 'good' emulator.

Diagnostics with the validation data (2)

Prediction errors in the validation data are correlated
Transform errors using a 'pivoted Cholesky decomposition' of $Var(\mathbf{y}^*|\mathbf{y})$.

Diagnostics with the validation data (2)

Prediction errors in the validation data are correlated
Transform errors using a 'pivoted Cholesky decomposition' of $Var(\mathbf{y}^*|\mathbf{y})$.
This gives us

- Sequence of independent errors, each approx $N(0, 1)$ for a 'correct' emulator

Diagnostics with the validation data (2)

Prediction errors in the validation data are correlated
Transform errors using a 'pivoted Cholesky decomposition' of $Var(\mathbf{y}^*|\mathbf{y})$.
This gives us

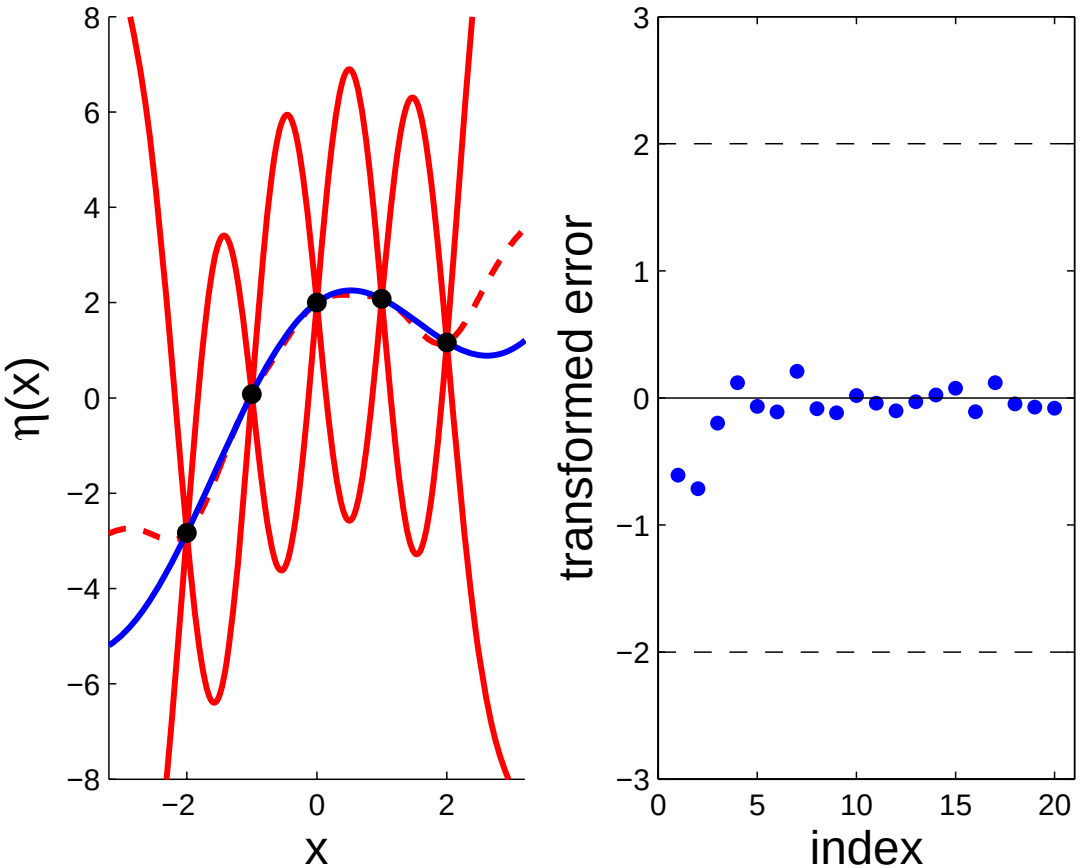
- Sequence of independent errors, each approx $N(0, 1)$ for a 'correct' emulator
- Early points in sequence correspond to most uncertain validation points - how does emulator behave in regions 'distant' to the training data?

Diagnostics with the validation data (2)

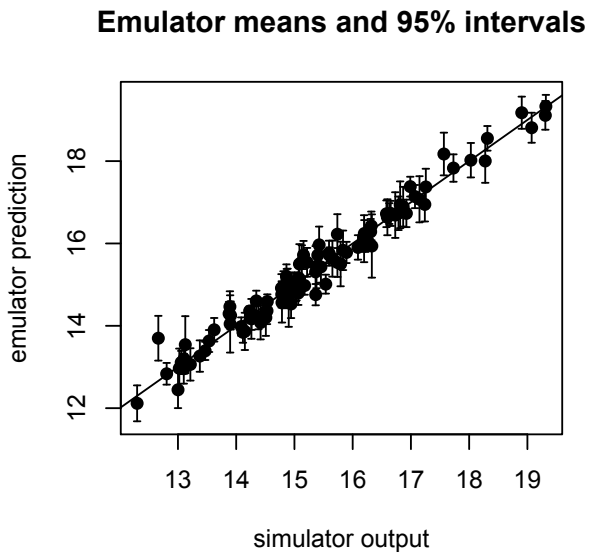
Prediction errors in the validation data are correlated
Transform errors using a 'pivoted Cholesky decomposition' of $Var(\mathbf{y}^*|\mathbf{y})$.
This gives us

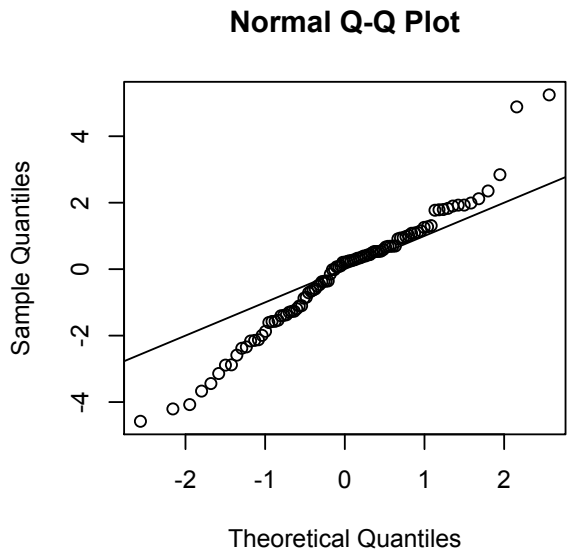
- Sequence of independent errors, each approx $N(0,1)$ for a 'correct' emulator
- Early points in sequence correspond to most uncertain validation points - how does emulator behave in regions 'distant' to the training data?
- Latter points in sequence correspond to validation points close to training data, or other validation points - are we describing the 'smoothness' of the simulator appropriately?

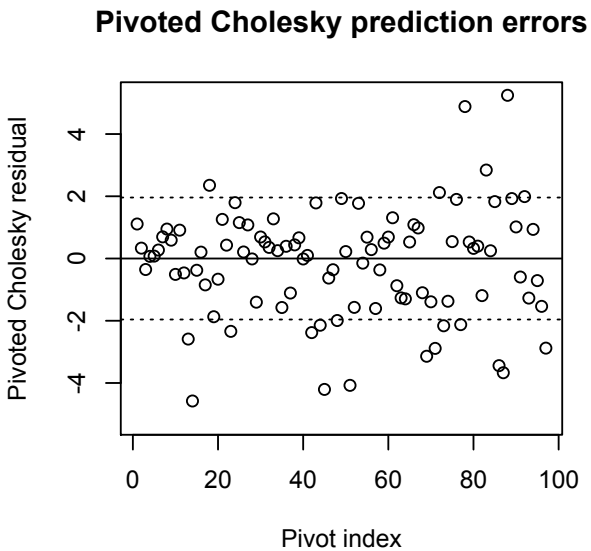
The pivoted Cholesky decomposition diagnostic

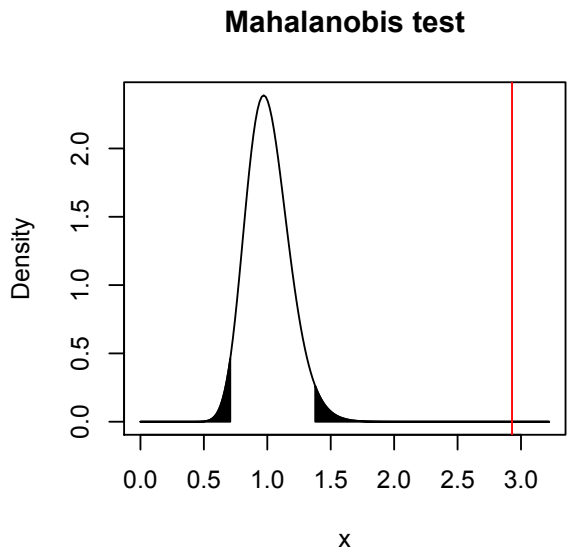


Example: 18 input climate model, constant mean function

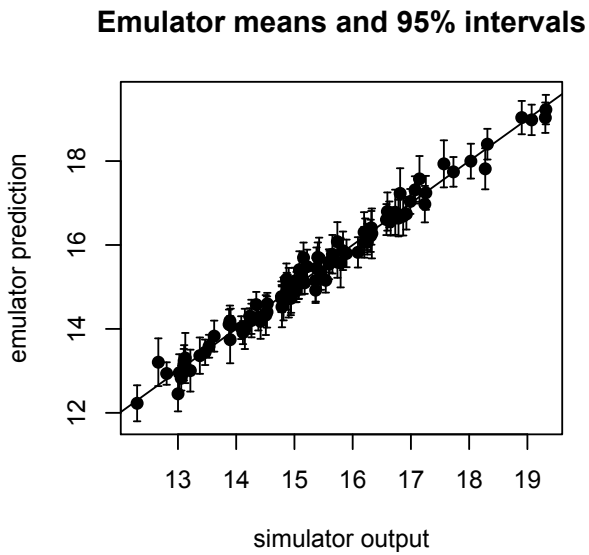


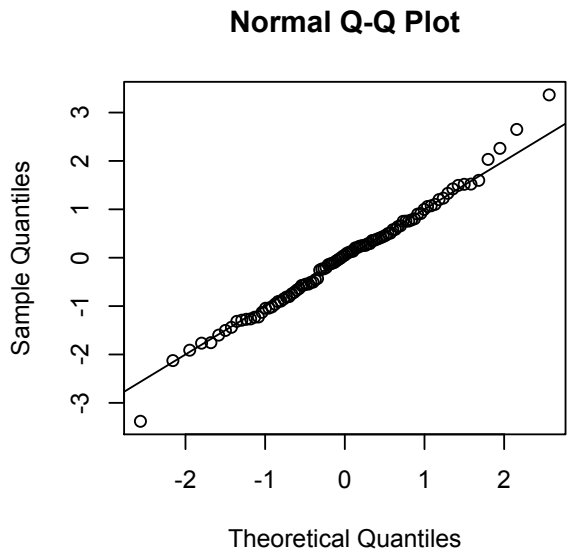


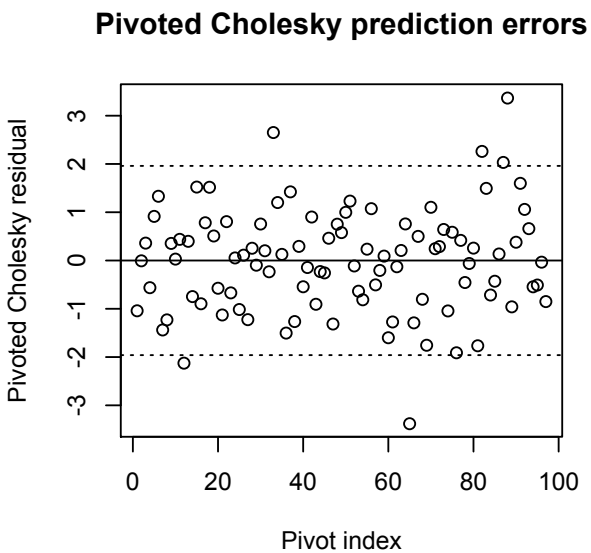


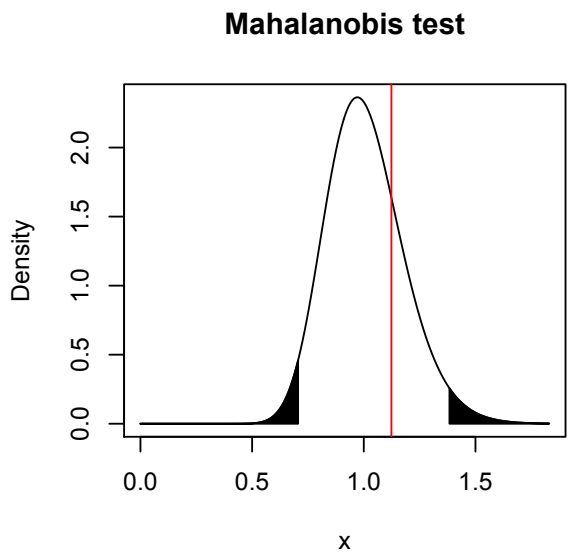


Example: 18 input climate model, linear mean function









Sensitivity analysis of model outputs

Sensitivity analysis of model outputs

- Local sensitivity analysis: calculating partial derivatives of f

Sensitivity analysis of model outputs

- Local sensitivity analysis: calculating partial derivatives of f
- Global or probabilistic sensitivity analysis: which elements in $X = \{X_1, \ldots, X_d\}$ are most responsible for the uncertainty in $Y = f(X)$?

Sensitivity analysis of model outputs

- Local sensitivity analysis: calculating partial derivatives of f
- Global or probabilistic sensitivity analysis: which elements in $X = \{X_1, \ldots, X_d\}$ are most responsible for the uncertainty in $Y = f(X)$?
- Variance based approach: investigate how inputs contribute to $Var(Y)$ (given f).

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., J.Cariboni, Gatelli, D., Saisana, M. and Tarantola, S. (2008). Global Sensitivity Analysis: The Primer, New York: Wiley.

Variance-based global sensitivity analysis

- Main effect variance:

$$Var_{X_i}\{E(Y|X_i)\}$$

Variance-based global sensitivity analysis

- Main effect variance:

$$Var_{X_i}\{E(Y|X_i)\}$$

- The expected reduction in variance if value of X_i is learnt, because

$$Var_{X_i}\{E(Y|X_i)\} = Var(Y) - E_{X_i}\{Var(Y|X_i)\}$$

Variance-based global sensitivity analysis

- Main effect variance:

$$Var_{X_i}\{E(Y|X_i)\}$$

- The expected reduction in variance if value of X_i is learnt, because

$$Var_{X_i}\{E(Y|X_i)\} = Var(Y) - E_{X_i}\{Var(Y|X_i)\}$$

- If inputs independent, can decompose $Var(Y)$ into 'main effect' variances and 'interaction' variances, similar to ANOVA.

Variance-based global sensitivity analysis

- Main effect variance:

$$Var_{X_i}\{E(Y|X_i)\}$$

- The expected reduction in variance if value of X_i is learnt, because

$$Var_{X_i}\{E(Y|X_i)\} = Var(Y) - E_{X_i}\{Var(Y|X_i)\}$$

- If inputs independent, can decompose $Var(Y)$ into 'main effect' variances and 'interaction' variances, similar to ANOVA.
- Depends on specification of $p(X)$.

Variance-based global sensitivity analysis

- Main effect variance:

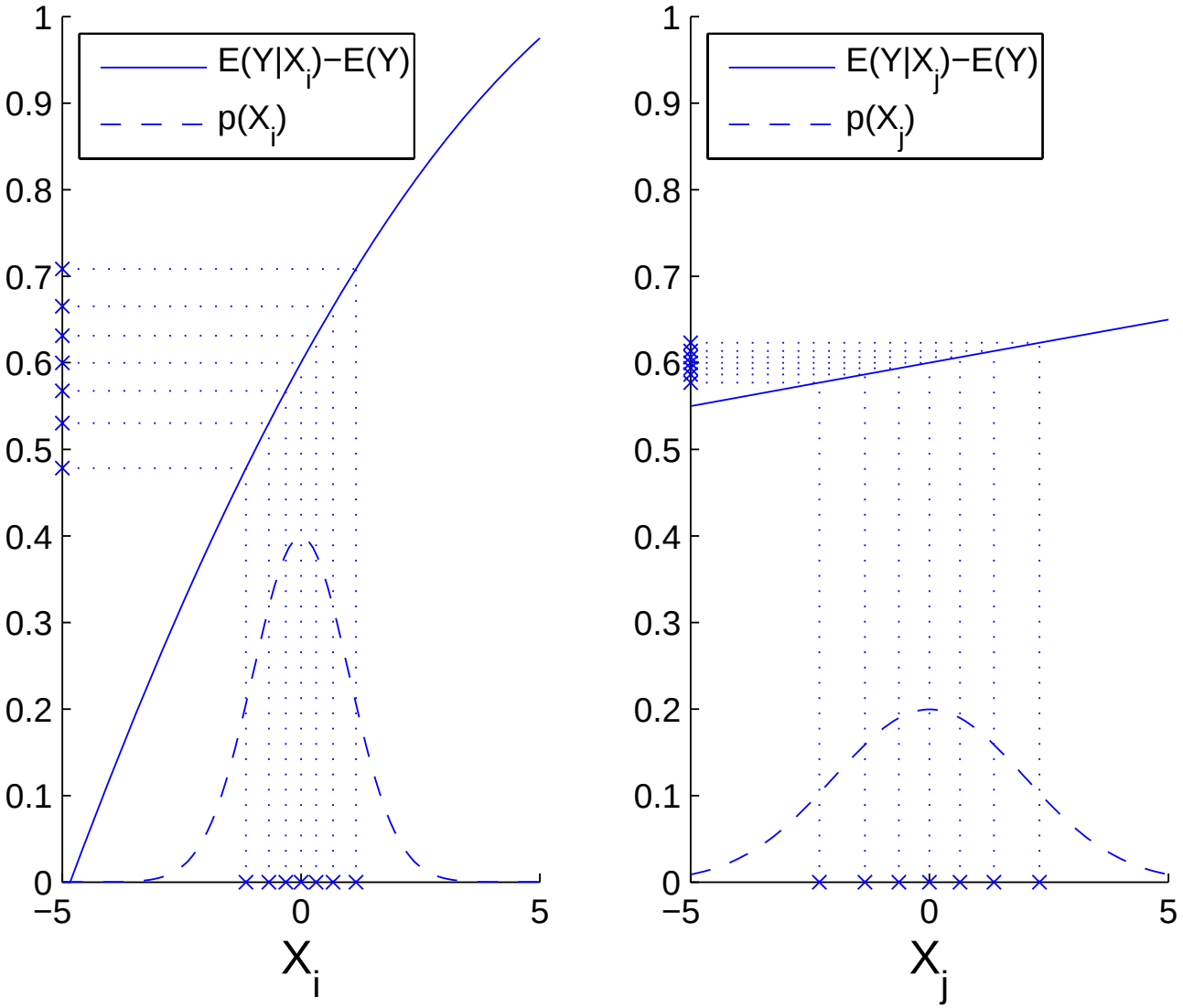
$$Var_{X_i}\{E(Y|X_i)\}$$

- The expected reduction in variance if value of X_i is learnt, because

$$Var_{X_i}\{E(Y|X_i)\} = Var(Y) - E_{X_i}\{Var(Y|X_i)\}$$

- If inputs independent, can decompose $Var(Y)$ into 'main effect' variances and 'interaction' variances, similar to ANOVA.
- Depends on specification of $p(X)$.
- Can measure input importance in different ways, eg decision-theoretic approaches.

Main effect plots and variances



Computation with emulators

Bayesian quadrature again. Now want

$$E(Y|X_i, f) = \int f(x_{-i}, X_i) p_{X_{-i}}(x_{-i}|X_i) dx_{-i},$$

Computation with emulators

Bayesian quadrature again. Now want

$$E(Y|X_i, f) = \int f(x_{-i}, X_i) p_{X_{-i}}(x_{-i}|X_i) dx_{-i},$$

Again, normally distributed with GP model for f. Can obtain mean and variance analytically for certain GP modelling choices and normally distributed inputs.

Computation with emulators

Bayesian quadrature again. Now want

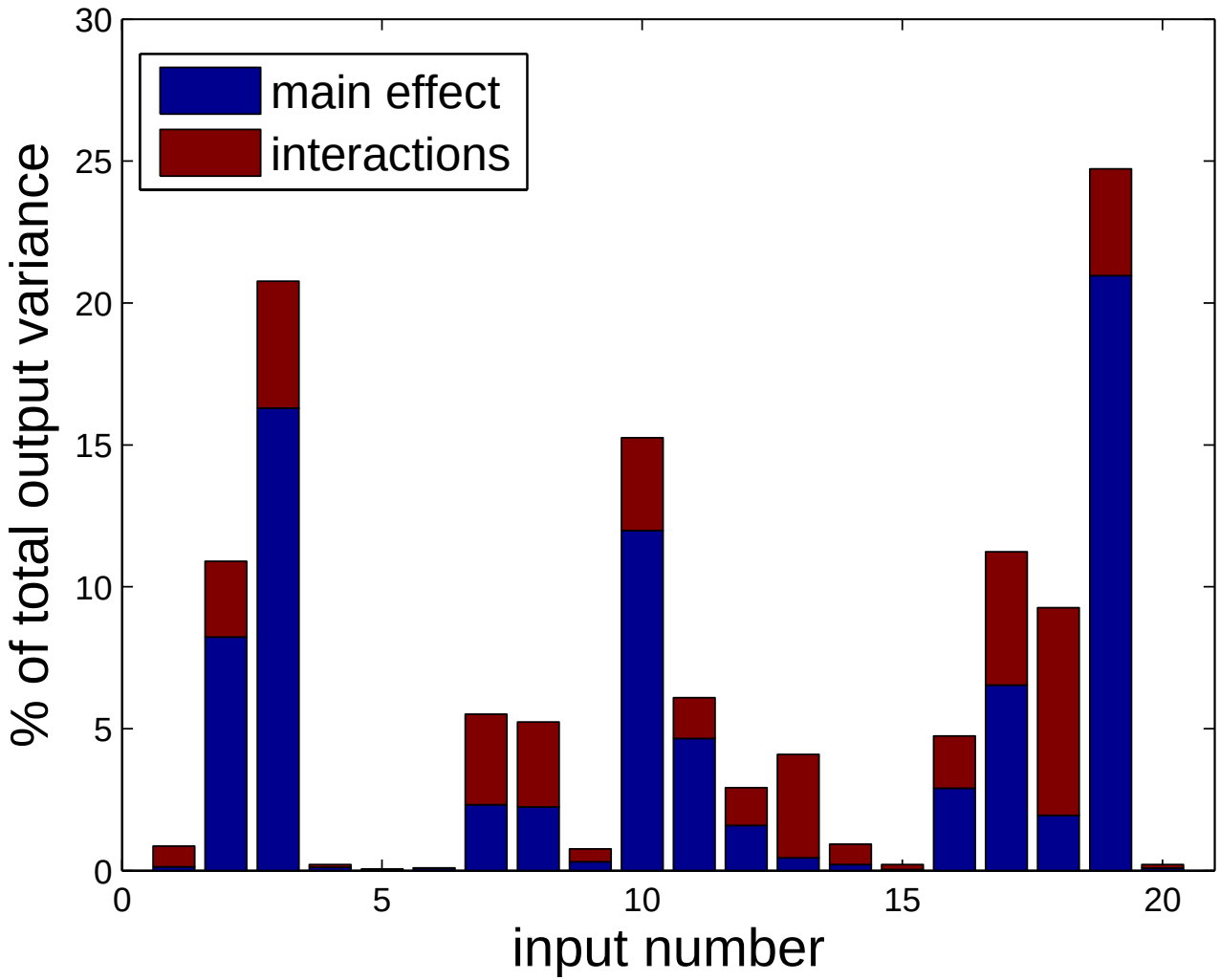
$$E(Y|X_i, f) = \int f(x_{-i}, X_i) p_{X_{-i}}(x_{-i}|X_i) dx_{-i},$$

Again, normally distributed with GP model for f. Can obtain mean and variance analytically for certain GP modelling choices and normally distributed inputs.

Oakley, J. E. and O'Hagan, A. (2004). Probabilistic sensitivity of complex models: a Bayesian approach, J. Roy. Statist. Soc. Ser. B, 66: 751-769.

Variance-based SA for an epidemic model

Acknowledgement: JP Gosling and others. See `www.mucm.ac.uk/toolkit` Case Study 1



Calibration and inverse problems

Calibration and inverse problems

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society, Series B. 63, 425-464.

Calibration and inverse problems

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society, Series B. 63, 425-464.

Example.

- A Gaussian plume deposition model $f(\mathbf{x}, \theta)$ predicts deposition of radionuclides at a location $\mathbf{x}$ following release of concentration θ from point source. θ is unknown.

Calibration and inverse problems

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society, Series B. 63, 425-464.

Example.

- A Gaussian plume deposition model $f(\mathbf{x}, \theta)$ predicts deposition of radionuclides at a location $\mathbf{x}$ following release of concentration θ from point source. θ is unknown.
- Measurements of the true deposition $z(\mathbf{x})$ at a limited number of locations $\mathbf{x}$ available.

Calibration and inverse problems

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society, Series B. 63, 425-464.

Example.

- A Gaussian plume deposition model $f(\mathbf{x}, \theta)$ predicts deposition of radionuclides at a location $\mathbf{x}$ following release of concentration θ from point source. θ is unknown.
- Measurements of the true deposition $z(\mathbf{x})$ at a limited number of locations $\mathbf{x}$ available.
- Aim: to predict deposition at other locations using both data and model.

Calibration and inverse problems

Kennedy, M. and O'Hagan, A. (2001). Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society, Series B. 63, 425-464.

Example.

- A Gaussian plume deposition model $f(\mathbf{x}, \theta)$ predicts deposition of radionuclides at a location $\mathbf{x}$ following release of concentration θ from point source. θ is unknown.
- Measurements of the true deposition $z(\mathbf{x})$ at a limited number of locations $\mathbf{x}$ available.
- Aim: to predict deposition at other locations using both data and model.
- What value of θ do we use?

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

- Need discrepancy
  1. to quantify uncertainty properly;

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

- Need discrepancy
  1. to quantify uncertainty properly;
  2. to learn about a 'true' θ

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

- Need discrepancy
  1. to quantify uncertainty properly;
  2. to learn about a 'true' θ
- But hard to specify; prior is important

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

- Need discrepancy
  1. to quantify uncertainty properly;
  2. to learn about a 'true' θ
- But hard to specify; prior is important
- Doesn't always go down well with modellers!

The calibration model

$$z(\mathbf{x}_i) = \rho f(\mathbf{x}_i, \theta) + \delta(\mathbf{x}_i) + \epsilon_i$$

$\delta(\mathbf{x}_i)$ is the **discrepancy** (bias) between model output and reality. Modelled with another GP.

- Need discrepancy
  1. to quantify uncertainty properly;
  2. to learn about a 'true' θ
- But hard to specify; prior is important
- Doesn't always go down well with modellers!

  "I'm horrified! You should be improving your models with better physics!"

History matching

Emphasis on ruling out ‘implausible’ regions of input space, prior to calibration

History matching

Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data

History matching

Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data
3. (Conservatively), discard implausible regions: regions where the best observed fit is 'poor'
   - May still be fairly uncertain about f at this stage, given D_1 only

History matching

Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data
3. (Conservatively), discard implausible regions: regions where the best observed fit is 'poor'
   - May still be fairly uncertain about f at this stage, given D_1 only
4. Get second training set $D_2 = \{y_{n+1} = f(x_{n+1}), \ldots, y_n = f(x_{n+m})\}$ over the 'plausible' design region

History matching

Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data
3. (Conservatively), discard implausible regions: regions where the best observed fit is 'poor'
   - May still be fairly uncertain about f at this stage, given D_1 only
4. Get second training set $D_2 = \{y_{n+1} = f(x_{n+1}), \ldots, y_n = f(x_{n+m})\}$ over the 'plausible' design region
5. Now discard further implausible regions, with improved information about f where it is needed

History matching

Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data
3. (Conservatively), discard implausible regions: regions where the best observed fit is 'poor'
   - May still be fairly uncertain about f at this stage, given D_1 only
4. Get second training set $D_2 = \{y_{n+1} = f(x_{n+1}), \ldots, y_n = f(x_{n+m})\}$ over the 'plausible' design region
5. Now discard further implausible regions, with improved information about f where it is needed
6. Iterate as necessary

History matching

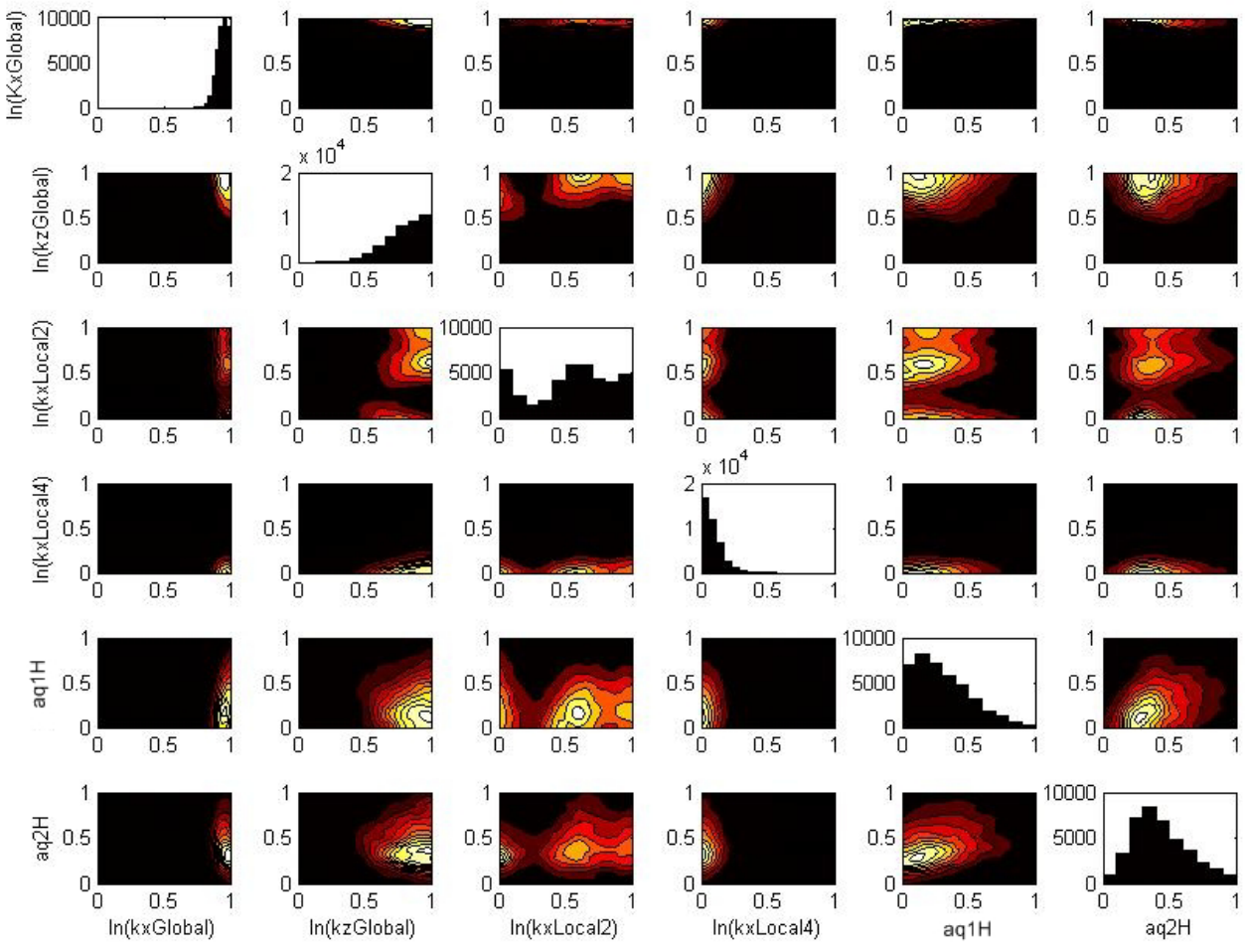
Emphasis on ruling out 'implausible' regions of input space, prior to calibration

1. Start with training runs $D_1 = \{y_1 = f(x_1), \ldots, y_n = f(x_n)\}$ over some design region
2. Fit emulator to the training data
3. (Conservatively), discard implausible regions: regions where the best observed fit is 'poor'
   - May still be fairly uncertain about f at this stage, given D_1 only
4. Get second training set $D_2 = \{y_{n+1} = f(x_{n+1}), \ldots, y_n = f(x_{n+m})\}$ over the 'plausible' design region
5. Now discard further implausible regions, with improved information about f where it is needed
6. Iterate as necessary

I Vernon, M Goldstein and R G Bower (2010), "Galaxy Formation: A Bayesian Uncertainty Analysis", Bayesian Analysis 05(04): 619–670 (with discussion)

History matching: oil reservoir simulator

Acknowledgement: JP Gosling and others. See `www.mucm.ac.uk/toolkit` Case Study 3



And finally...

If you like Sheffield, and you like GPs:

Uncertainty in Computer Models 2014
Sheffield
July 28-30
`www.mucm.ac.uk/UCM2014.html`

Abstract submissions for talks and posters welcome