

Emulation

David Cameron

2019-09-11

Plan

- ▶ Very quick introduction to emulators
- ▶ Large topic so could take up a whole week long course in its own right.
- ▶ Focus on conceptual understanding so that you have this before investigating further in literature/ emulator packages (say in R).
- ▶ Very similar to spatial modelling but now in model parameter space rather than physical space.

Why do we need emulators?

For a MCMC chainlength of $O(1e5)$ a model run duration greater than a few seconds is going to be problematic. . .

model run (secs)	MCMC (hrs)	MCMC (days)
0.1	2.78	0.12
0.5	13.89	0.58
1.0	27.78	1.16
3.0	83.33	3.47
5.0	138.89	5.79
10.0	277.78	11.57
60.0	1666.67	69.44
300.0	8333.33	347.22

Why do we need emulators?

For a MCMC chainlength of $O(1e5)$ a model run duration greater than a few seconds is going to be problematic. . .

model run (secs)	MCMC (hrs)	MCMC (days)
0.1	2.78	0.12
0.5	13.89	0.58
1.0	27.78	1.16
3.0	83.33	3.47
5.0	138.89	5.79
10.0	277.78	11.57
60.0	1666.67	69.44
300.0	8333.33	347.22

Solution : Replace model with an approximate faster surrogate.

But : How does uncertainty increase by replacing the model with a fast approximation. . . ?

Why do we need emulators?

How do we create the fast surrogate?

In addition:

If we cannot run the model $f(\cdot)$ for all possible x 's in an MCMC

$\Rightarrow f(x)$ will be uncertain

We need to quantify this uncertainty using probabilities.

A Gaussian process (GP) emulator is an efficient way of doing this.

Emulators - some useful definitions

Define:

Simulator : The original model that we are trying to emulate.

x : Simulator inputs (e.g. parameters that are not known)

$y = f(x)$: simulator is expressed as a function with outputs(data) y

Surrogate : Fast model approximation that we fit to the simulator.

Emulator : Statistical surrogate that also quantifies uncertainty.

What is a Gaussian process emulator? (1)

A probability distribution for a function

If $f(\cdot)$ is a function and it is uncertain

- ▶ For every possible x $f(x)$ is uncertain so it has a distribution
- ▶ For every x and x' $f(x)$ and $f(x')$ have a joint probability distribution

A GP is a probability distribution for the whole function i.e. for all the possible $f(x)$ s

- ▶ A big joint probability distribution for all possible $f(x)$ s in infinite dimensions

What is a Gaussian Process emulator? (2)

Fortunately for us conceptually and mathematically very similar to spatial modelling taught yesterday.

Can think of:

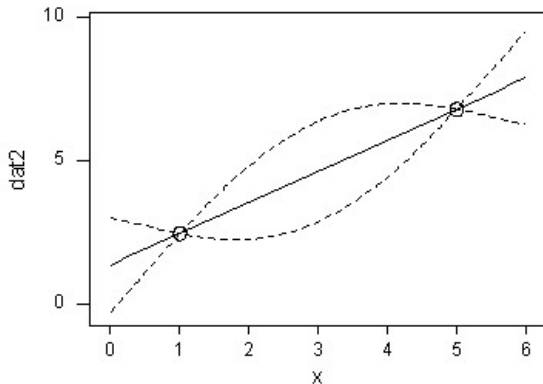
Geostatistics (interpolation in physical space) \equiv Emulation in 2D parameter space.

Concept:

- ▶ in spatial modelling locations close together in physical space should be more similar than those far apart.
- ▶ in emulation input/parameter values close together in parameter space should be more similar than those far apart.
- ▶ in fact in the 2D example (later) we'll use geoR to create the emulator.

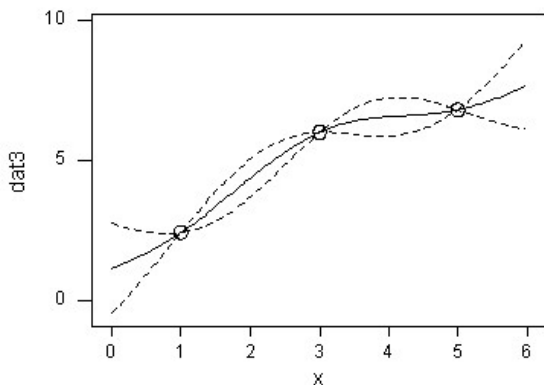
1D example - Two simulator runs

- ▶ Consider a simulator with one input x and one output y
- ▶ Emulator estimate interpolates the data
- ▶ Emulator uncertainty grows between data points



1D example - Three simulator runs

- ▶ Changes estimate and reduces uncertainty.
- ▶ Doesn't lie on the original line but in the region of original uncertainty



1D example - Five simulator runs

- ▶ With more points $f(\cdot)$ is pinned down accurately over the range of simulator runs.

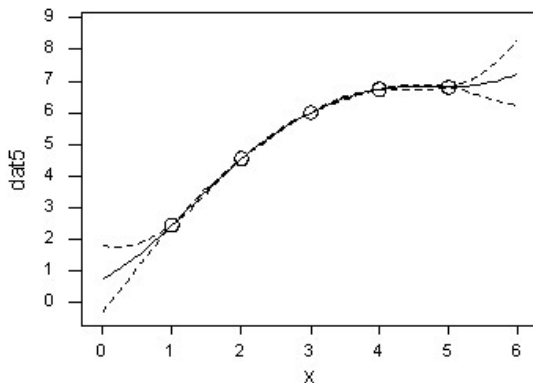


figure from Tony O'Hagan

Main idea

- ▶ Carefully placing enough simulator runs we can build an emulator with low uncertainty (design).

Works for two or more inputs

- ▶ x typically a vector of inputs (eg simulator parameters)
 - ▶ increases dimensionality one input 1D, two 2D, three 3D etc
 - ▶ up to 10 inputs need tens of model runs
 - ▶ becomes difficult for more than 20. 100s of inputs won't work with GP.

Higher dimensions

- ▶ e.g. 2D X is now a plane pinning points down
- ▶ we'll see a 2D example later...

Properties of an emulator

- ▶ Emulator gives us a full probability distribution for every x
 - ▶ Emulator quantifies uncertainty about $f(x)$
- ▶ The mean function is a fast surrogate
 - ▶ to approximate $f(x)$ for any x
- ▶ At every data point x where we have run the simulator it correctly predicts the corresponding y
 - ▶ the emulator mean is an interpolator
- ▶ Not the same as response surfaces and neural nets

Emulator uses

Predicting output $f(x)$ for given inputs x

- ▶ with quantified uncertainty
- ▶ much faster than the simulator itself

Uncertainty analysis

- ▶ propagating input uncertainty through the model
- ▶ predicting $f(x)$ with uncertain x
- ▶ including uncertainty in the surrogate

Sensitivity analysis

- ▶ how inputs influence outputs

Calibration of models

- ▶ crucial to accommodate structural uncertainty (discrepancy return to this later. . .)

Slightly more maths: What is a Gaussian Process GP?

Probability distribution for a function

- ▶ just about any kind of function
- ▶ very flexible
- ▶ mathematically very similar to spatial modelling (geostatistics)

$$f(.) = GP(m(.), c(.,.))$$

Is Gaussian

$f(x) \sim N(m(x), c(x, x))$ with one input is normal distribution

for $2 \dots n$ inputs is a multivariate normal distribution in $2 \dots n$ dimensions

GP parameters

- ▶ $m(\cdot)$ a mean function
 - ▶ can be anything
- ▶ $c(\cdot, \cdot)$ a covariance function

Stationary case where we take the covariance function as a function only of the distance between x and x'

$$c(x, x') = \sigma^2 r(|x - x'|)$$

σ^2 = variance and r = correlation between x and x'

where correlation function $r(\cdot)$ is positive definite iff it is the characteristic function of a probability distribution

$r(0) = 1$ all other values less than 1

Building the emulator

Prior

- ▶ Building begins with a prior GP distribution for the simulator
- ▶ Express knowledge about $f(\cdot)$ held prior to observing any training runs
- ▶ Generally quite uninformative

Conditioning (fitting)

The GP is then conditioned on a number of runs (data y)

$$f(x_1) \dots f(x_n)$$

- ▶ to produce the posterior GP distribution
- ▶ posterior is the emulator

Prior mean function $m(\cdot)$

Mean function can reflect genuine quantitative beliefs regarding $f(\cdot)$

In practise mean function often modelled quite simply as

$$m(x) = \beta^T h$$

h is a vector of regression coefficients

β is a vector of unknown hyperparameters

- ▶ noninformative prior distribution for β
- ▶ a hierarchical model

Prior covariance function $c(.,.)$

- ▶ Beliefs in the form of quantified uncertainty about the simulator
 - ▶ What is not captured by the mean function.
- ▶ In practise we generally use the stationary form

$$c(x, x') = \sigma^2 r(|x - x'|)$$

Popular choices for the correlation function

- ▶ Matern
- ▶ Gaussian correlation function

$$r(d) = \exp(-d^T \Phi^{-1} d) \text{ with } \Phi = \text{diag}(\phi_1^2 \dots \phi_p^2)$$

- ▶ Application specific: whichever gives a good fit to the simulator.

Prior covariance hyperparameters

Again often quite uninformative

Though useful to consider carefully especially if only a few runs of the simulator are possible.

Should reflect how responsive simulator is to changes in the inputs.

How much the simulator can vary over and above what is captured by the mean function.

σ^2 variance

How the magnitude of the function varies with x (input space)

ϕ correlation length

How fast the simulator changes as we move in x (input space)

Fitting

Deriving posterior emulator is

Lower level of hierarchy

Conditioning the GP on the simulator runs $y = f(x_1) \dots f(x_n)$

Upper level of hierarchy

But data also inform hyperparameters β, σ^2, ϕ

Conditioned GP combined with the posterior distribution of the hyperparameters.

$$p(f(\cdot)|\beta, \sigma^2, \phi, y)p(\beta, \sigma^2, \phi|y)$$

2D example

now open emuExample.html (or pdf) (R markdown file is emuExample.Rmd)

Emulation packages

Packages have often been limited to emulating a single scalar output from the simulator but this is now changing (see for eg ExeterUQ).

In R:

- ▶ **BACCO**
- ▶ **GEM-SA** <http://www.tonyohagan.co.uk/academic/GEM/>
- ▶ **ExeterUQ** <https://github.com/vicvolodina93/ExeterUQ>

Useful references

- ▶ 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
- ▶ Conti, S. and O'Hagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference* 140, 640-651
- ▶ Brynjarsdottir, J. and O'Hagan, A. (2014). Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30, 114007 (24pp)

Comprehensive UQ reloaded - another application for GP...

Whether calibrating a simulator (such as the VSEM earlier) or the emulator derived from a simulator we shouldn't forget that our models aren't perfect. Therefore we should always include terms in the calibration that represent model structural errors or discrepancy.

- ▶ We write reality $z(x)$ as the sum of $f(x)$ and discrepancy $\delta(x)$
- ▶ The very simple discrepancy model that we introduced in the previous tutorial, as just a simple multiplicative or additive bias, is probably not adequate for many simulators.
- ▶ Here we have seen how flexible the GP is in emulating complex models.
- ▶ Another application for GP is to model potentially complex spatial discrepancy (open compUQ2.pdf)

Extra slides

Conditioning: a property of the GP

$f(.) \sim GP(m(.), c(.,.))$ then for any x

$f(.)|f(x) \sim GP(m^*(.), c^*(.,.))$ where i.e. it remains a GP

$$m^*(x') = m(x') + c(x', x)\{f(x) - m(x)\}/c(x, x)$$

$$c^*(x', x'') = c(x', x'') - c(x', x)c(x'', x)/c(x, x)$$

At x

$$m^*(x) = f(x) \text{ and } c^*(x, x) = 0 \text{ as we saw before}$$

By induction $f(.)$ remains a GP if we condition on a discrete set of function values $f(x_1) \dots f(x_n)$