Bayes linear strategies for emulation and history matching for complex computer models

Michael Goldstein
Durham University*

*Thanks to Wellcome Trust and the EPSRC funded Centre for Energy Systems Integration

Models and physical systems: some examples

Systems biology Models of activity at the cellular level are used to make inferences about the behaviour of the biological organism.

Oil reservoirs An oil reservoir simulator is used to manage assets associated with the reservoir, in order to develop efficient production schedules, etc.

Natural Hazards Floods, volcanoes, tsunamis and so forth, are all studied by large computer simulators.

Disease modelling Agent based models are used to study interventions to control infectious diseases.

Energy planning Simulators of future energy demand and provision are key components of planning for energy investment.

Climate change Large scale climate simulators are constructed to assess likely effects of human intervention upon future climate behaviour.

Galaxy formation The study of the development of the Universe is carried out by using a Galaxy formation simulator.

Models and physical systems: some examples

Systems biology Models of activity at the cellular level are used to make inferences about the behaviour of the biological organism.

Oil reservoirs An oil reservoir simulator is used to manage assets associated with the reservoir, in order to develop efficient production schedules, etc.

Natural Hazards Floods, volcanoes, tsunamis and so forth, are all studied by large computer simulators.

Disease modelling Agent based models are used to study interventions to control infectious diseases.

Energy planning Simulators of future energy demand and provision are key components of planning for energy investment.

Climate change Large scale climate simulators are constructed to assess likely effects of human intervention upon future climate behaviour.

Galaxy formation The study of the development of the Universe is carried out by using a Galaxy formation simulator.

The science in each is completely different. However, the underlying methodology for handling uncertainty is the same.

Sources of Uncertainty

(i) parametric uncertainty (each model requires a, typically high dimensional, parametric specification)
(ii) condition uncertainty (uncertainty as to boundary conditions, initial conditions, and forcing functions),
(iii) functional uncertainty (model evaluations take a long time, so the function is unknown almost everywhere)
(iv) stochastic uncertainty (either the model is stochastic, or it should be),
(v) solution uncertainty (as the system equations can only be solved to some necessary level of approximation).
(vi) structural uncertainty (the model only approximates the physical system),
(vii) measurement uncertainty (as the model is calibrated against system data all of which is measured with error),
(viii) multi-model uncertainty (usually we have not one but many models related to the physical system)
(ix) decision uncertainty (to use the model to influence real world outcomes, we need to relate things in the world that we can influence to inputs to the simulator and through outputs to actual impacts. These links are uncertain.)

General form of problem

We have a collection of observations z on the real world values y of a physical system. This is the system history.

General form of problem

We have a collection of observations z on the real world values y of a physical system. This is the system history.

We have a model for the system.

This is often implemented as a computer simulator $f(x)$.

General form of problem

We have a collection of observations z on the real world values y of a physical system. This is the system history.

We have a model for the system.

This is often implemented as a computer simulator $f(x)$.

The simulator inputs are the parameter collection x

(plus other stuff like decision choices and forcing functions that we suppress to simplify notation)

General form of problem

We have a collection of observations z on the real world values y of a physical system. This is the system history.

We have a model for the system.

This is often implemented as a computer simulator $f(x)$.

The simulator inputs are the parameter collection x

(plus other stuff like decision choices and forcing functions that we suppress to simplify notation)

The simulator output $f(x)$ is the assessment of the system history.

(plus other stuff which may be relevant and useful).

History matching

History matching is the problem of finding the collection $C(z)$ of **all** choices x^* for which $f(x^*)$ is **"near"** observed system history z.

History matching

History matching is the problem of finding the collection $C(z)$ of **all** choices x^* for which $f(x^*)$ is **"near"** observed system history z.

Usually the modellers wish to know whether it is possible to history match at all.

History matching

History matching is the problem of finding the collection $C(z)$ of **all** choices x^* for which $f(x^*)$ is **"near"** observed system history z.

Usually the modellers wish to know whether it is possible to history match at all.

If $C(z)$ is empty, this typically identifies a problem with the model or the data.

In such cases, we aim to identify the conflicts which prevent a full history match.

History matching

History matching is the problem of finding the collection $C(z)$ of **all** choices x^* for which $f(x^*)$ is **"near"** observed system history z.

Usually the modellers wish to know whether it is possible to history match at all.

If $C(z)$ is empty, this typically identifies a problem with the model or the data.

In such cases, we aim to identify the conflicts which prevent a full history match.

More generally, the shape of the set $C(z)$ identifies the constraints on the parameter space that are imposed by the data.

Further uses of history matching

Further uses of history matching

Forecasting

Often, $f(x)$ will also contain outputs for unobservable quantities of interest. These may be future system outcomes or features for which the model is the only assessment.

Further uses of history matching

Durham University

Forecasting

Often, $f(x)$ will also contain outputs for unobservable quantities of interest. These may be future system outcomes or features for which the model is the only assessment.

The collection of evaluations $f(x^*), x^* \in C(z)$ shows the range of simulator forecasts which are consistent with observed history.

Further uses of history matching

Forecasting

Often, $f(x)$ will also contain outputs for unobservable quantities of interest. These may be future system outcomes or features for which the model is the only assessment.

The collection of evaluations $f(x^*), x^* \in C(z)$ shows the range of simulator forecasts which are consistent with observed history.

Decision support

Suppose, for example, that simulator $f(.)$ takes inputs which may help control future outputs.

Further uses of history matching

Forecasting

Often, $f(x)$ will also contain outputs for unobservable quantities of interest. These may be future system outcomes or features for which the model is the only assessment.

The collection of evaluations $f(x^*), x^* \in C(z)$ shows the range of simulator forecasts which are consistent with observed history.

Decision support

Suppose, for example, that simulator $f(.)$ takes inputs which may help control future outputs.

Evaluation of effective control over the range of inputs in $C(z)$ identifies which are the safest control strategies and whether more data is needed before controls are introduced.

Some history of history matching

Some history of history matching

The term “history matching” is widely used in the oil industry.

Some history of history matching

The term “history matching” is widely used in the oil industry.

For their problems, the computer model typically is a simulator of an oil reservoir, and the aim of history matching is to construct a description of the reservoir geology so that the simulator behaves in accord with observed system behaviour, for example at the wells.

Some history of history matching

The term “history matching” is widely used in the oil industry.

For their problems, the computer model typically is a simulator of an oil reservoir, and the aim of history matching is to construct a description of the reservoir geology so that the simulator behaves in accord with observed system behaviour, for example at the wells.

Around 30 years ago, a group at Durham (Peter Craig, Michael Goldstein, Allan Seheult, with postdoc James Smith) saw this as a way to attack a very wide class of problems of uncertainty quantification.

Some history of history matching

The term “history matching” is widely used in the oil industry.

For their problems, the computer model typically is a simulator of an oil reservoir, and the aim of history matching is to construct a description of the reservoir geology so that the simulator behaves in accord with observed system behaviour, for example at the wells.

Around 30 years ago, a group at Durham (Peter Craig, Michael Goldstein, Allan Seheult, with postdoc James Smith) saw this as a way to attack a very wide class of problems of uncertainty quantification.

Here’s a refence to our early work.
Craig, Goldstein, Seheult, Smith (1997) Pressure matching for hydrocarbon reservoirs: a case study in the use of **Bayes linear strategies** for large computer experiments (with discussion)
In Case Studies in Bayesian Statistics, Gastonis et al New York: Springer-Verlag, III,37-93.

History matching as a pre-calibration tool

History matching as a pre-calibration tool

History matching is not a calibration method.

History matching as a pre-calibration tool

History matching is not a calibration method.

This is because model parameters need not have real world physical meanings. Often, parameters only exist inside models, so that different choices may be good for fitting different outputs.

History matching as a pre-calibration tool

History matching is not a calibration method.

This is because model parameters need not have real world physical meanings. Often, parameters only exist inside models, so that different choices may be good for fitting different outputs.

However, when there is a real world meaning for the parameters, or we just wish to see what a formal calibration might look like, then it is often a good idea to do a history match first.

History matching as a pre-calibration tool

History matching is not a calibration method.

This is because model parameters need not have real world physical meanings. Often, parameters only exist inside models, so that different choices may be good for fitting different outputs.

However, when there is a real world meaning for the parameters, or we just wish to see what a formal calibration might look like, then it is often a good idea to do a history match first.

This should greatly improve reliability of Bayesian algorithms for assessing posterior distributions over the parameter space given the observed history.

History matching as a pre-calibration tool

History matching is not a calibration method.

This is because model parameters need not have real world physical meanings. Often, parameters only exist inside models, so that different choices may be good for fitting different outputs.

However, when there is a real world meaning for the parameters, or we just wish to see what a formal calibration might look like, then it is often a good idea to do a history match first.

This should greatly improve reliability of Bayesian algorithms for assessing posterior distributions over the parameter space given the observed history.

This is important because the likelihood surface is complicated and multi-modal, and the Bayes answer often depends on features of the prior distribution which are hard to specify meaningfully.

The Bayesian approach

In the Bayesian approach, all probabilities are the subjective judgements of individuals (at least, in principle).

The Bayesian approach

In the Bayesian approach, all probabilities are the subjective judgements of individuals (at least, in principle).

Michael Goldstein Subjective Bayesian analysis: principles and practice (2006) Bayesian Analysis, 1, 403-420 (and 'Rejoinder to discussion': 465-472)

The Bayesian approach

In the Bayesian approach, all probabilities are the subjective judgements of individuals (at least, in principle).

Michael Goldstein Subjective Bayesian analysis: principles and practice (2006) Bayesian Analysis, 1, 403-420 (and 'Rejoinder to discussion': 465-472)

Michael Goldstein, "Why be a Bayesian?", in Advanced Statistical Techniques in Particle Physics. Proceedings, Conference, Durham, UK, March 18-22, 2002, p. 300. 2002.
http://www.ippp.dur.ac.uk/Workshops/02/statistics/proceedings/ /goldstein.pdf

The Bayesian approach

In the Bayesian approach, all probabilities are the subjective judgements of individuals (at least, in principle).

Michael Goldstein Subjective Bayesian analysis: principles and practice (2006) Bayesian Analysis, 1, 403-420 (and 'Rejoinder to discussion': 465-472)

Michael Goldstein, "Why be a Bayesian?", in Advanced Statistical Techniques in Particle Physics. Proceedings, Conference, Durham, UK, March 18-22, 2002, p. 300. 2002.
http://www.ippp.dur.ac.uk/Workshops/02/statistics/proceedings/ /goldstein.pdf

The Bayesian approach can be difficult in large problems because of the extreme level of detail which is required in the specification of beliefs.
(And the technical difficulty of the full Bayes calculations.)

The Bayes linear approach

In the Bayes linear approach, we combine prior judgements of uncertainty with observational data, using **expectation** rather than **probability** as the primitive.

The Bayes linear approach

In the Bayes linear approach, we combine prior judgements of uncertainty with observational data, using **expectation** rather than **probability** as the primitive.

This approach is similar in spirit to a full Bayes analysis, but uses a much simpler approach for prior specification and analysis, and so offers a practical methodology for analysing partially specified beliefs for large problems.

The Bayes linear approach

In the Bayes linear approach, we combine prior judgements of uncertainty with observational data, using **expectation** rather than **probability** as the primitive.

This approach is similar in spirit to a full Bayes analysis, but uses a much simpler approach for prior specification and analysis, and so offers a practical methodology for analysing partially specified beliefs for large problems.

Bayes linear adjustment may be viewed as
(i) an approximation to a full Bayes analysis or
(ii) the appropriate analysis given a partial specification.
(There are rigorous foundations for this viewpoint.)

Bayesian linear adjustment

The Bayes linear adjusted expectation and variance for vector y given vector z are

$$\mathsf{E}_z[y] = \mathrm{E}(y) + \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}(z - \mathrm{E}(z)),$$
$$\mathsf{Var}_z[y] = \mathrm{Var}(y) - \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}\mathrm{Cov}(z,y)$$

Bayesian linear adjustment

The Bayes linear adjusted expectation and variance for vector y given vector z are

$$\mathsf{E}_z[y] = \mathrm{E}(y) + \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}(z - \mathrm{E}(z)),$$
$$\mathsf{Var}_z[y] = \mathrm{Var}(y) - \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}\mathrm{Cov}(z,y)$$

For a detailed treatment, see

Bayes linear Statistics: Theory and Methods, 2007, (Wiley)

Michael Goldstein and David Wooff

Bayesian linear adjustment

The Bayes linear adjusted expectation and variance for vector y given vector z are

$$\mathsf{E}_z[y] = \mathrm{E}(y) + \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}(z - \mathrm{E}(z)),$$
$$\mathsf{Var}_z[y] = \mathrm{Var}(y) - \mathrm{Cov}(y,z)\mathrm{Var}(z)^{-1}\mathrm{Cov}(z,y)$$

For a detailed treatment, see

Bayes linear Statistics: Theory and Methods, 2007, (Wiley)

Michael Goldstein and David Wooff

For a quick overview, see

Bayes linear analysis, 2015, Michael Goldstein, in Wiley StatsRef: Statistics Reference Online (7 pages)

Bayesian linear adjustment

The Bayes linear adjusted expectation and variance for vector y given vector z are

$$\mathsf{E}_z[y] = \mathrm{E}(y) + \mathrm{Cov}(y, z)\mathrm{Var}(z)^{-1}(z - \mathrm{E}(z)),$$
$$\mathsf{Var}_z[y] = \mathrm{Var}(y) - \mathrm{Cov}(y, z)\mathrm{Var}(z)^{-1}\mathrm{Cov}(z, y)$$

For a detailed treatment, see
Bayes linear Statistics: Theory and Methods, 2007, (Wiley)
Michael Goldstein and David Wooff

For a quick overview, see
Bayes linear analysis, 2015, Michael Goldstein, in Wiley StatsRef: Statistics Reference Online (7 pages)

And the uncertainty quantification papers in this talk contain plenty of examples of Bayes linear computations.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Therefore, we must construct a description of uncertainty about the value of $f(x)$ for each x.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Therefore, we must construct a description of uncertainty about the value of $f(x)$ for each x.

Such a representation is often termed an emulator of the simulator.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Therefore, we must construct a description of uncertainty about the value of $f(x)$ for each x.

Such a representation is often termed an emulator of the simulator.

The emulator both contains
(i) an approximation to the simulator and

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Therefore, we must construct a description of uncertainty about the value of $f(x)$ for each x.

Such a representation is often termed an emulator of the simulator.

The emulator both contains
(i) an approximation to the simulator and

(ii) an assessment of the likely magnitude of the error of the approximation.

Function emulation

Uncertainty analysis, for high dimensional problems, is particularly challenging if $f(x)$ is expensive, in time and computational resources, to evaluate for any choice of x.

In such cases, f must be treated as uncertain for all input choices except the small subset for which an actual evaluation has been made.

Therefore, we must construct a description of uncertainty about the value of $f(x)$ for each x.

Such a representation is often termed an emulator of the simulator.

The emulator both contains
(i) an approximation to the simulator and

(ii) an assessment of the likely magnitude of the error of the approximation.

Unlike the original simulator, the emulator is fast to evaluate for any choice of inputs. This allows us to explore model behaviour for all physically meaningful input specifications.

Form of the emulator

We may represent beliefs about component f_i of f, using an emulator:

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x) + u_i(x)$$

Form of the emulator

We may represent beliefs about component f_i of f, using an emulator:

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x) + u_i(x)$$

Global Variation

$\{\beta_{ij}\}$ are unknown scalars,

g_{ij} are known deterministic functions of x, (for example, polynomials)

Form of the emulator

We may represent beliefs about component f_i of f, using an emulator:

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x) + u_i(x)$$

Global Variation

$\{\beta_{ij}\}$ are unknown scalars,

g_{ij} are known deterministic functions of x, (for example, polynomials)

Local Variation

$u_i(x)$ is a second order stationary stochastic process, with (for example) correlation function

$$\mathrm{Corr}(u_i(x), u_i(x')) = \exp(-(\tfrac{\|x-x'\|}{\theta_i})^2)$$

Form of the emulator

We may represent beliefs about component f_i of f, using an emulator:

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x) + u_i(x)$$

Global Variation

$\{\beta_{ij}\}$ are unknown scalars,

g_{ij} are known deterministic functions of x, (for example, polynomials)

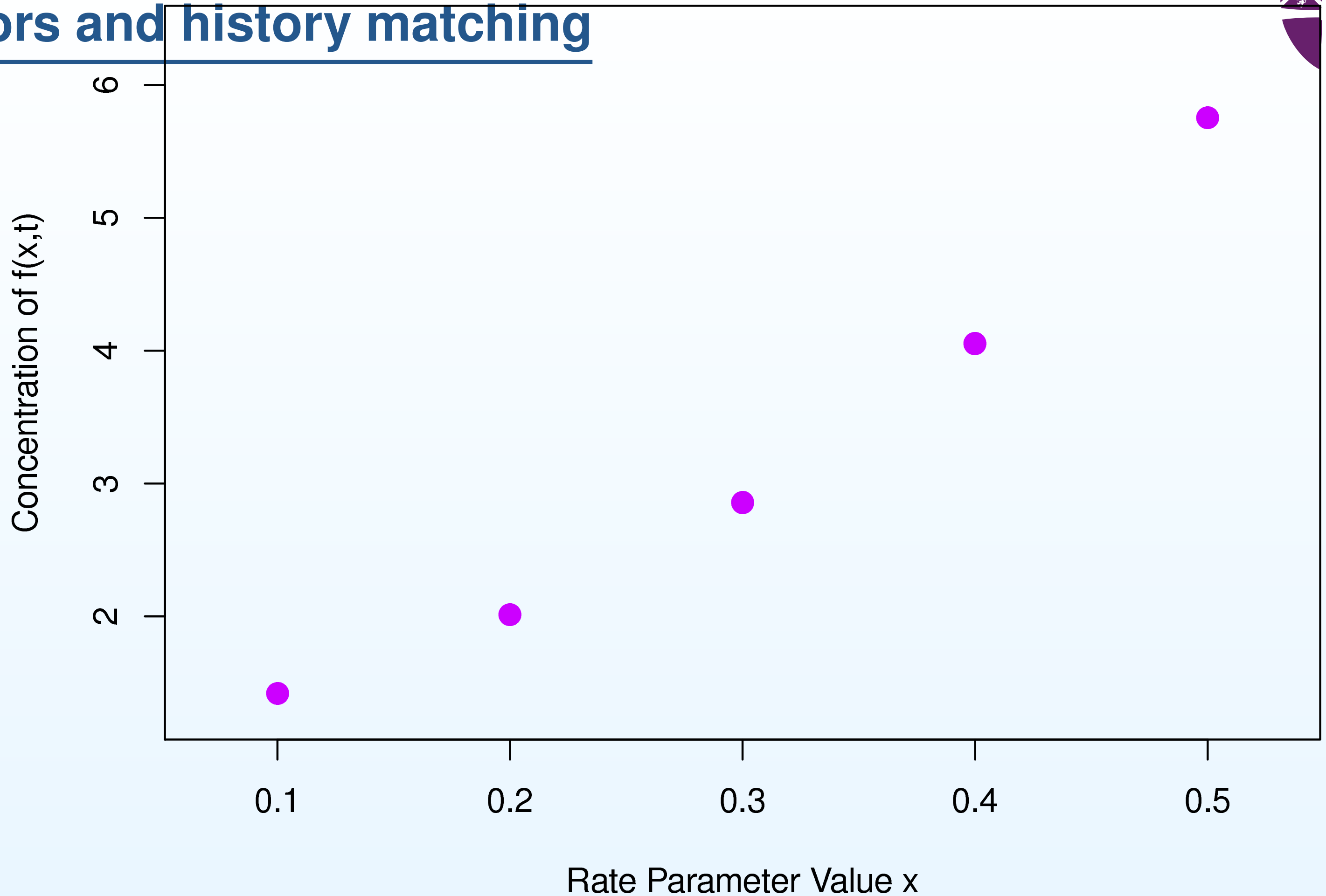
Local Variation

$u_i(x)$ is a second order stationary stochastic process, with (for example) correlation function

$$\mathrm{Corr}(u_i(x), u_i(x')) = \exp(-(\tfrac{\|x-x'\|}{\theta_i})^2)$$

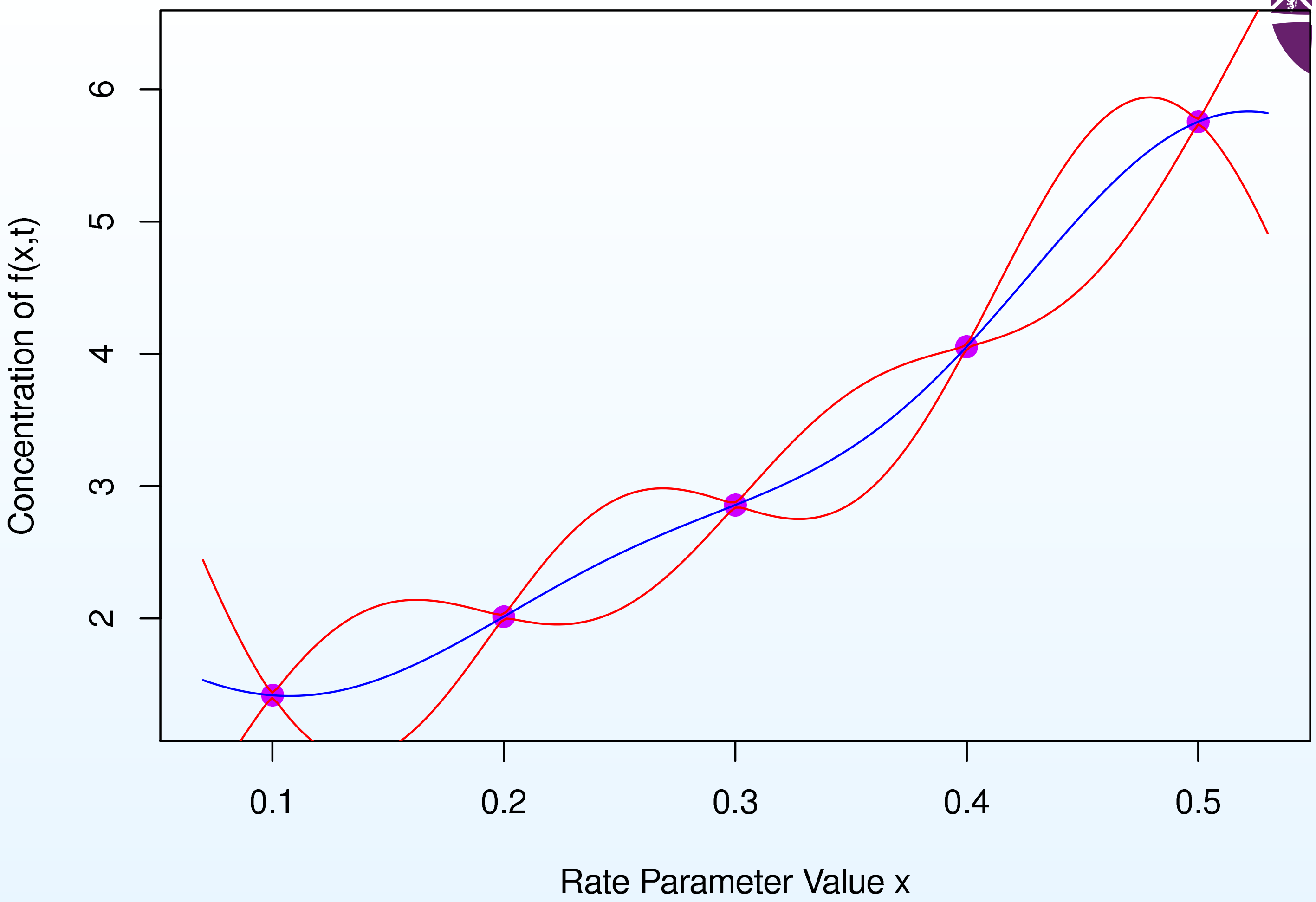
(or a more complex version of this form).

Emulators and history matching

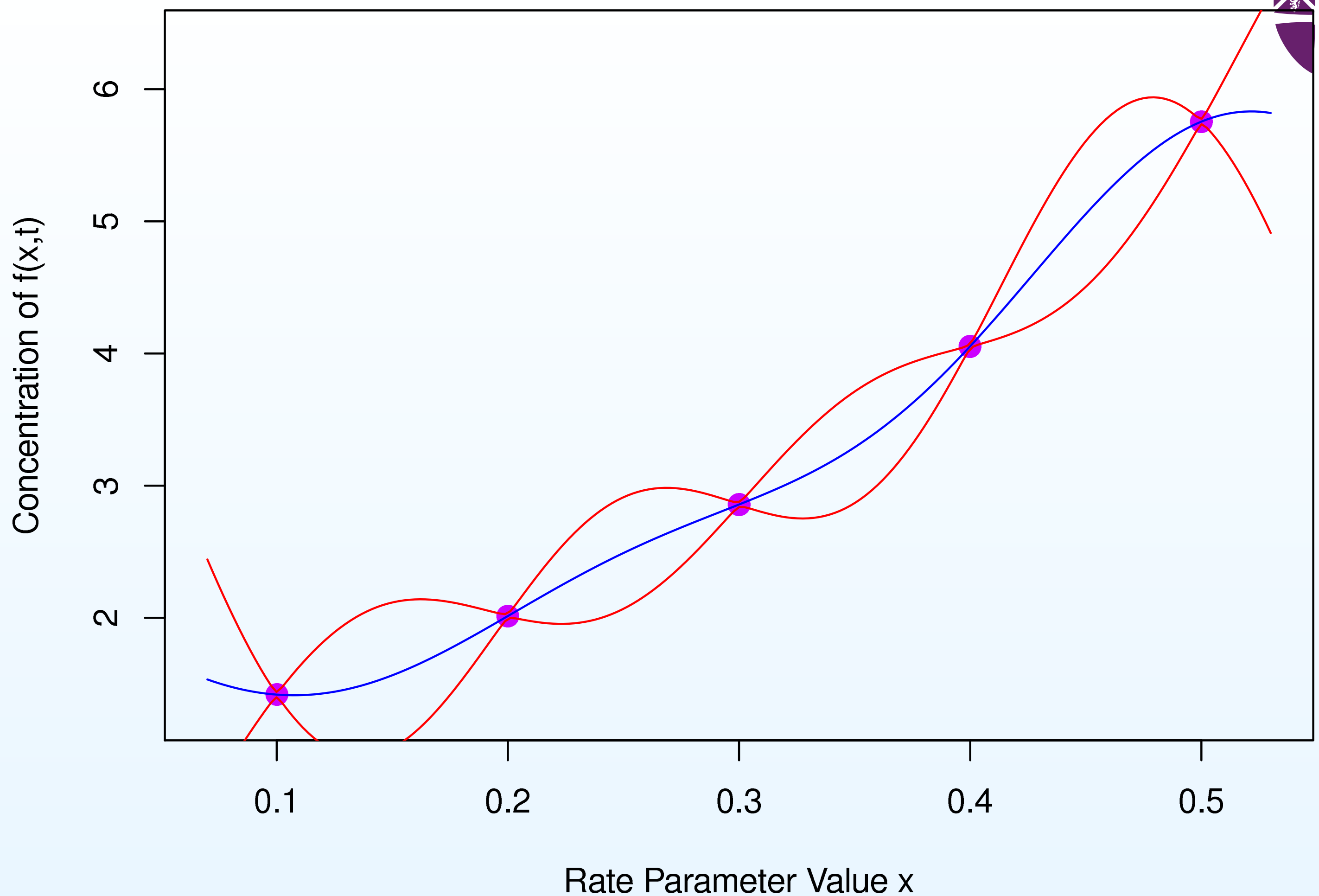


We have made five evaluations of the function $f(x)$.

Suppose that we now build an emulator for $f(x)$ based on these five points.

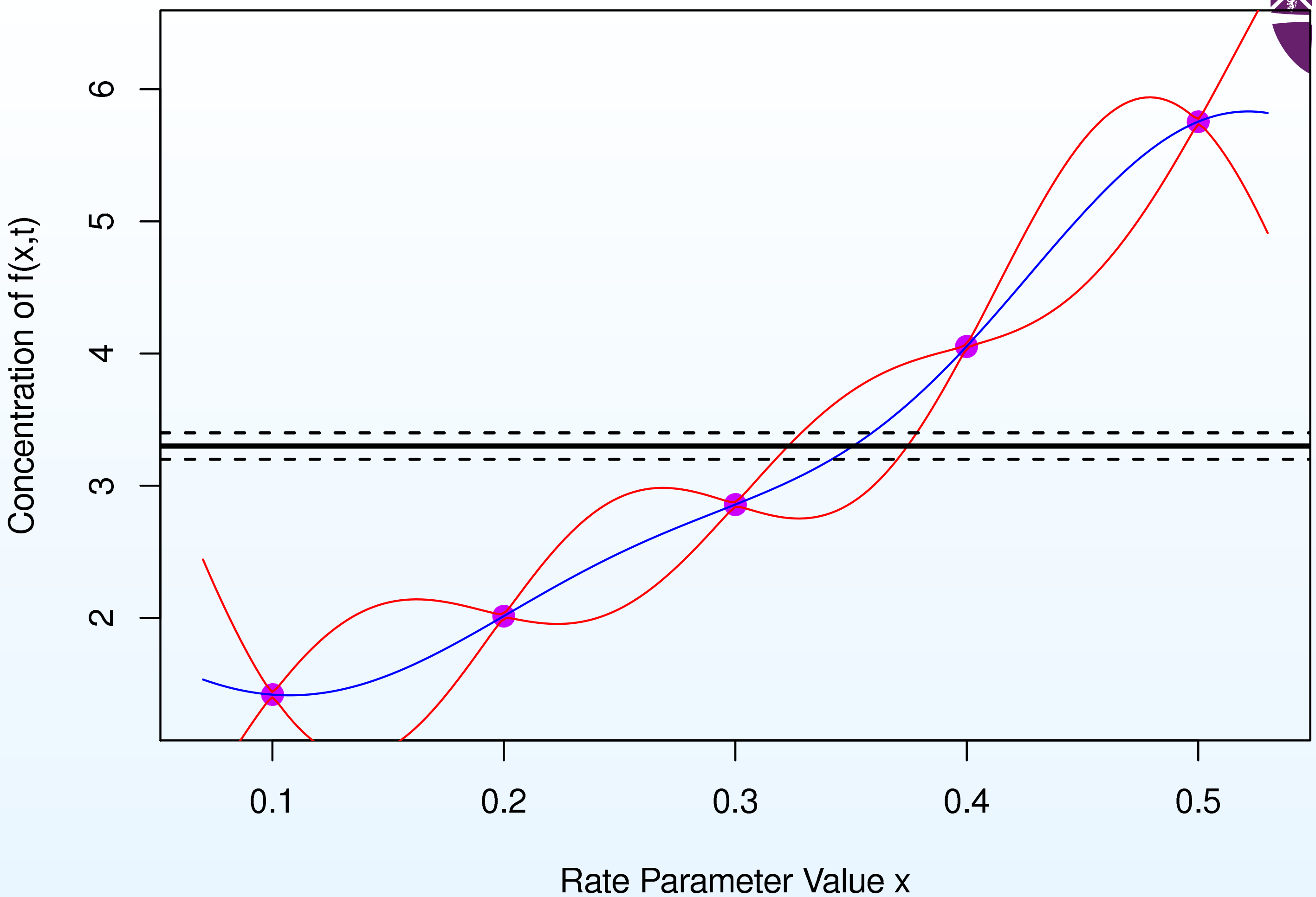


The emulator can be used to represent our beliefs about the behaviour of the model at untested values of x, and is fast to evaluate.



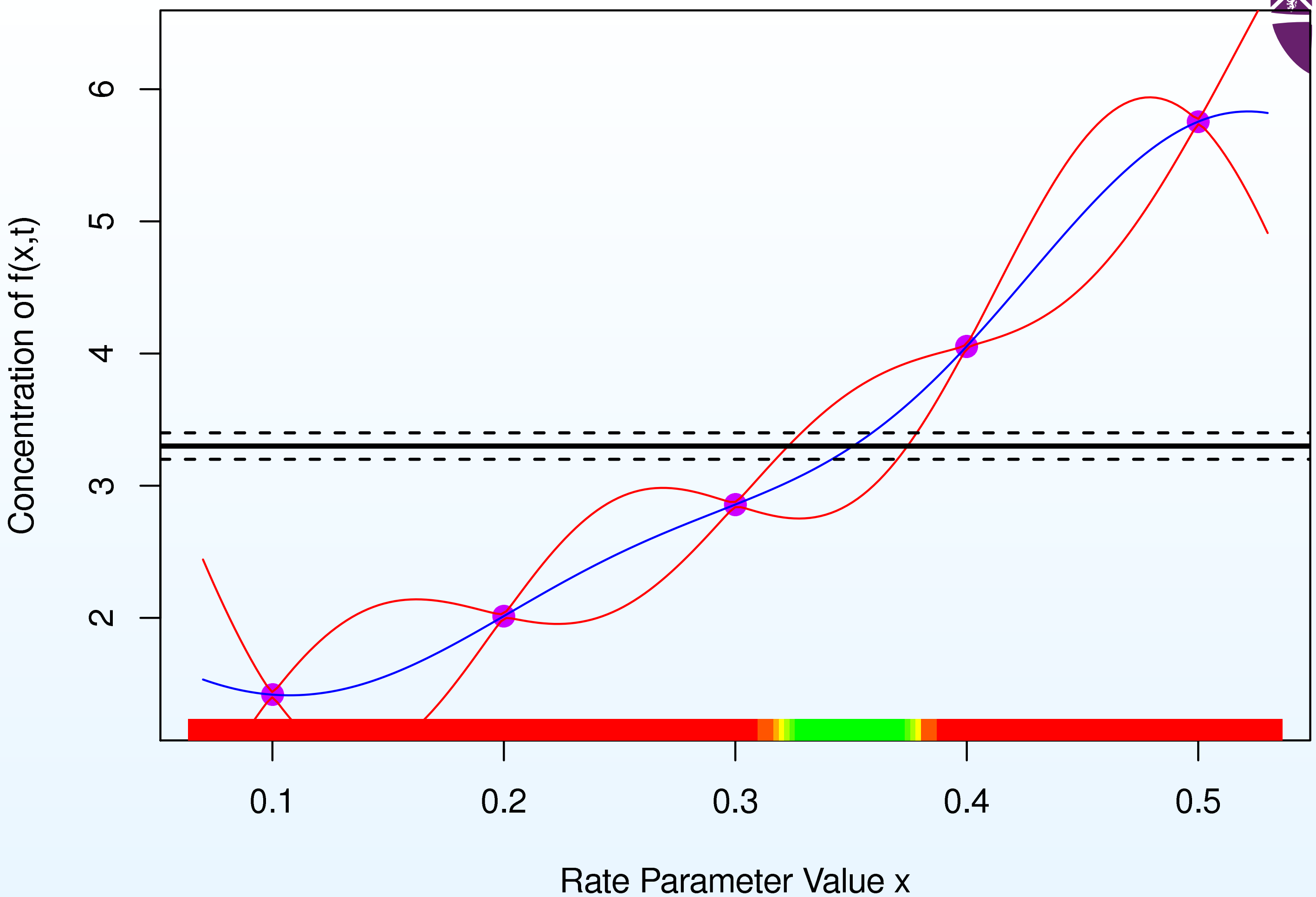
The emulator can be used to represent our beliefs about the behaviour of the model at untested values of x, and is fast to evaluate.

It gives both the expected value of $f(x)$ (the blue line) along with a credible interval for $f(x)$ (the red lines) representing uncertainty about the model's behaviour.

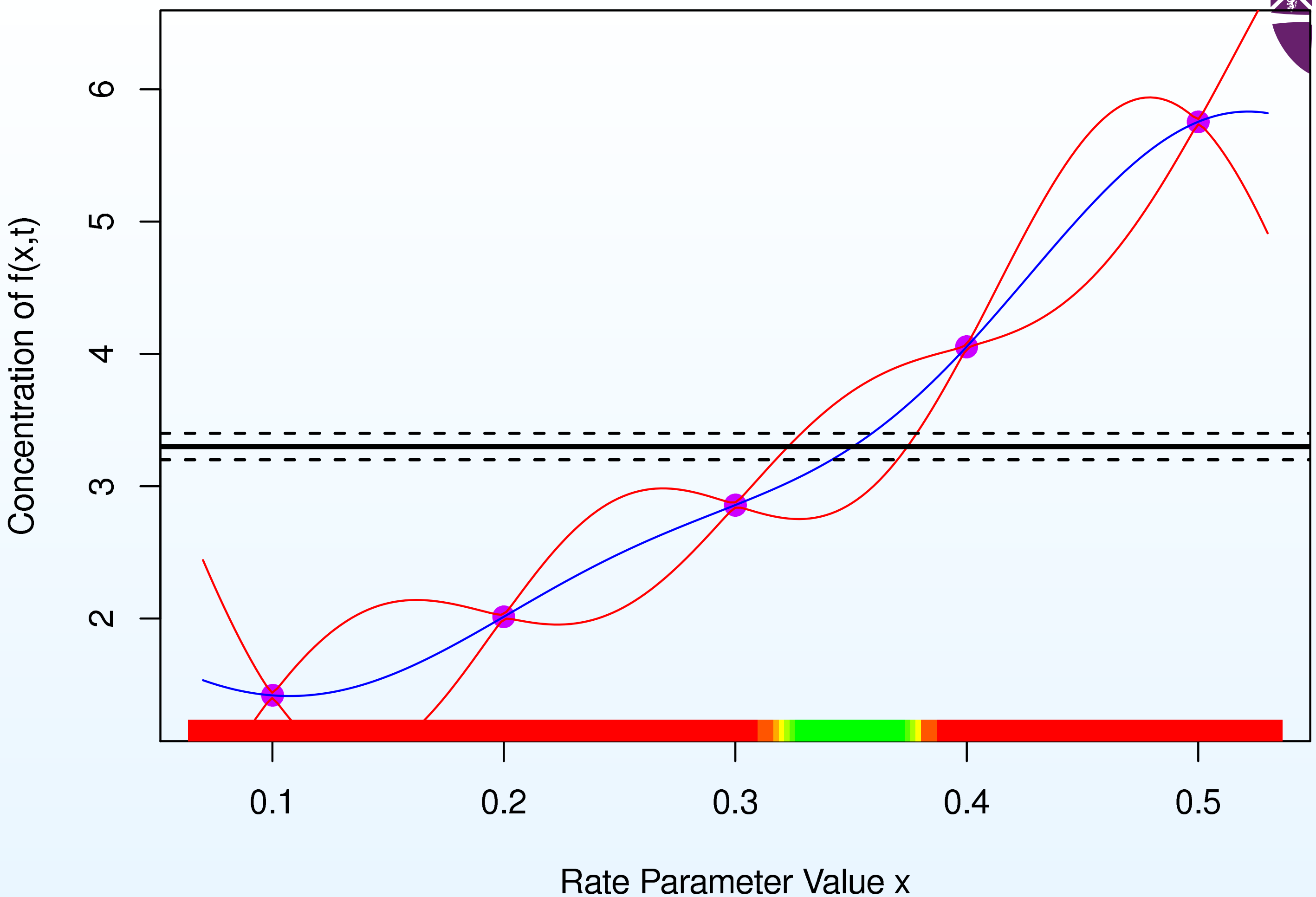


Suppose that we have an observation (represented by the black line with observational errors bounds).

Comparing the emulator to the observed measurement we can identify the set of x values which are “not inconsistent” with this data.

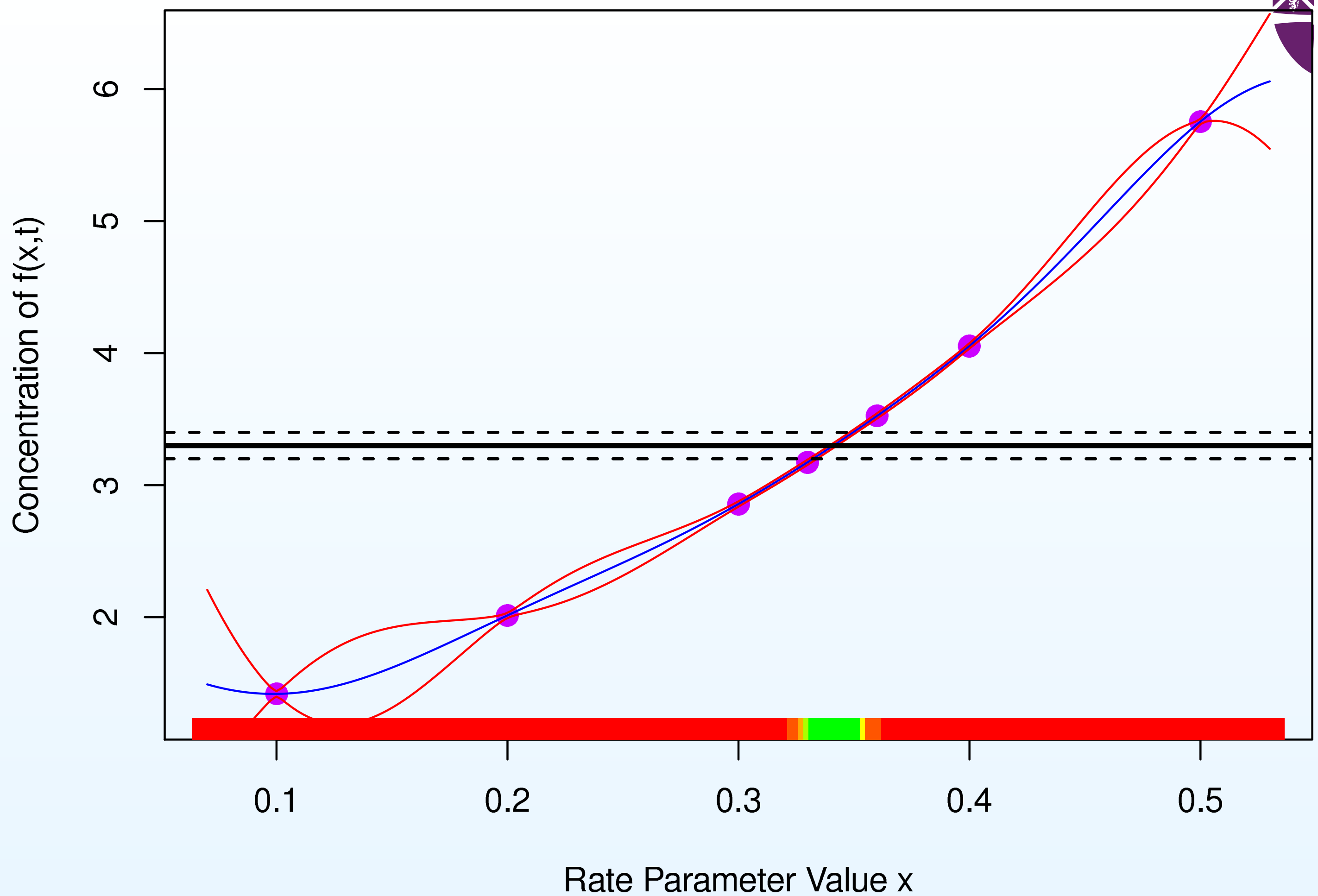


Comparing the emulator to the observed measurement we have identified the set of x values (the green values) which “match” the observed history, when we take into account all of the uncertainties (here, measurement and emulator error).



We now remove all of the implausible x values (the red values) and resample and re-emulate within the green region.

We perform a 2nd iteration or wave of runs to improve emulator accuracy. The runs are located only at non-implausible (green/yellow) points.



Now the emulator is more accurate than the observation, and we can identify the set of all x values of interest.

Emulation for history matching

History matching is an iterative procedure.

At each wave,
[1] we take a sample in the current green space,
[2] refit our emulator(s)
[3] eliminate as much of the green space as we can.

So, at each stage, all that we need is an emulator which is accurate enough to remove some of the green space.

Emulation for history matching

History matching is an iterative procedure.

At each wave,
[1] we take a sample in the current green space,
[2] refit our emulator(s)
[3] eliminate as much of the green space as we can.

So, at each stage, all that we need is an emulator which is accurate enough to remove some of the green space.

This is much easier than building an accurate emulator for the whole space.

We only need accurate emulation of the simulator in the region close to the output match.

Emulation for history matching

History matching is an iterative procedure.

At each wave,
[1] we take a sample in the current green space,
[2] refit our emulator(s)
[3] eliminate as much of the green space as we can.

So, at each stage, all that we need is an emulator which is accurate enough to remove some of the green space.

This is much easier than building an accurate emulator for the whole space.

We only need accurate emulation of the simulator in the region close to the output match.

Further, if we are matching many outputs, then at each wave of history matching we only need to emulate those outputs which are relatively straightforward to emulate at that stage.

(As the green space shrinks, the behaviour of complex functions often simplifies.)

Emulation methods

We fit the emulators, given a collection of carefully chosen model evaluations, using our favourite statistical tools - generalised least squares, maximum likelihood, Bayes (linear) - supported by expert judgement.

Emulation methods

We fit the emulators, given a collection of carefully chosen model evaluations, using our favourite statistical tools - generalised least squares, maximum likelihood, Bayes (linear) - supported by expert judgement.

Here's a possible approach
We use efficient space filling designs to generate the set of simulator evaluations to carry out in order to fit the emulators.
(For example, maximin Latin Hypercubes.)

Emulation methods

We fit the emulators, given a collection of carefully chosen model evaluations, using our favourite statistical tools - generalised least squares, maximum likelihood, Bayes (linear) - supported by expert judgement.

Here's a possible approach
We use efficient space filling designs to generate the set of simulator evaluations to carry out in order to fit the emulators.
(For example, maximin Latin Hypercubes.)

Identify a key collection of outputs to construct emulators for.

Emulation methods

We fit the emulators, given a collection of carefully chosen model evaluations, using our favourite statistical tools - generalised least squares, maximum likelihood, Bayes (linear) - supported by expert judgement.

Here's a possible approach
We use efficient space filling designs to generate the set of simulator evaluations to carry out in order to fit the emulators.
(For example, maximin Latin Hypercubes.)

Identify a key collection of outputs to construct emulators for.

For each of the chosen outputs, $f_i(x)$ say, identify a collection of 'active' inputs, $x_{A(i)}$ say, which are most important in driving variation in that output.
Fit the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

and decompose the local residual $u_i(x)$ as the sum of one term involving $x_{A(i)}$, and one term involving all of the other inputs (possibly just a nugget).

Fitting the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Either fit the whole emulator in one go, or first fit the global form and then fit the local form to the residuals.

Fitting the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Either fit the whole emulator in one go, or first fit the global form and then fit the local form to the residuals.

Assess the scientific plausibility of each emulator
(appropriate choice of active variables and form of global model)

Fitting the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Either fit the whole emulator in one go, or first fit the global form and then fit the local form to the residuals.

Assess the scientific plausibility of each emulator
(appropriate choice of active variables and form of global model)

Use careful diagnostics to test the validity of our emulators, for example, assessing the reliability of the emulator for predicting the simulator at new evaluations.

Fitting the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Either fit the whole emulator in one go, or first fit the global form and then fit the local form to the residuals.

Assess the scientific plausibility of each emulator
(appropriate choice of active variables and form of global model)

Use careful diagnostics to test the validity of our emulators, for example, assessing the reliability of the emulator for predicting the simulator at new evaluations.

Often, some outputs turn out to be straightforward to emulate, while others are more difficult.

Fitting the emulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Either fit the whole emulator in one go, or first fit the global form and then fit the local form to the residuals.

Assess the scientific plausibility of each emulator
(appropriate choice of active variables and form of global model)

Use careful diagnostics to test the validity of our emulators, for example, assessing the reliability of the emulator for predicting the simulator at new evaluations.

Often, some outputs turn out to be straightforward to emulate, while others are more difficult.

If the model is stochastic, then we also emulate the variance, for example using Bayes linear estimation for the variance of each $f(x)$ given evaluation of some repetitions at chosen design runs.

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

We don't observe $f(x)$ for most values of x, so we use the emulator expectation $\mathrm{E}(f(x))$ with variance σ_f^2.

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

We don't observe $f(x)$ for most values of x, so we use the emulator expectation $\mathrm{E}(f(x))$ with variance σ_f^2.

We don't judge the simulator to be a perfect representation of reality, so we introduce a structural discrepancy for example viewing the relation between y and $f(x)$ at an acceptable choice of x as $y = f(x) + \epsilon$ with variance σ_ϵ^2.

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

We don't observe $f(x)$ for most values of x, so we use the emulator expectation $\mathrm{E}(f(x))$ with variance σ_f^2.

We don't judge the simulator to be a perfect representation of reality, so we introduce a structural discrepancy for example viewing the relation between y and $f(x)$ at an acceptable choice of x as $y = f(x) + \epsilon$ with variance σ_ϵ^2.

We use an 'implausibility measure' $I(x)$ based on a probabilistic metric such as

$$I(x) = \frac{(z - \mathrm{E}(f(x)))^2}{\mathrm{Var}(z - \mathrm{E}(f(x)))} = \frac{(z - \mathrm{E}(f(x)))^2}{\sigma_e^2 + \sigma_f^2 + \sigma_\epsilon^2}$$

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

We don't observe $f(x)$ for most values of x, so we use the emulator expectation $\mathrm{E}(f(x))$ with variance σ_f^2.

We don't judge the simulator to be a perfect representation of reality, so we introduce a structural discrepancy for example viewing the relation between y and $f(x)$ at an acceptable choice of x as $y = f(x) + \epsilon$ with variance σ_ϵ^2.

We use an 'implausibility measure' $I(x)$ based on a probabilistic metric such as

$$I(x) = \frac{(z - \mathrm{E}(f(x)))^2}{\mathrm{Var}(z - \mathrm{E}(f(x)))} = \frac{(z - \mathrm{E}(f(x)))^2}{\sigma_e^2 + \sigma_f^2 + \sigma_\epsilon^2}$$

Large values of $I(x)$ suggest it is implausible that $f(x)$ is a good match to y

Implausibility

We want to judge whether input x produces output $f(x)$ which is near system value y. We don't observe y but see z.

We might judge that $z = y + e$ where e has zero mean and variance σ_e^2.

We don't observe $f(x)$ for most values of x, so we use the emulator expectation $\mathrm{E}(f(x))$ with variance σ_f^2.

We don't judge the simulator to be a perfect representation of reality, so we introduce a structural discrepancy for example viewing the relation between y and $f(x)$ at an acceptable choice of x as $y = f(x) + \epsilon$ with variance σ_ϵ^2.

We use an 'implausibility measure' $I(x)$ based on a probabilistic metric such as

$$I(x) = \frac{(z - \mathrm{E}(f(x)))^2}{\mathrm{Var}(z - \mathrm{E}(f(x)))} = \frac{(z - \mathrm{E}(f(x)))^2}{\sigma_e^2 + \sigma_f^2 + \sigma_\epsilon^2}$$

Large values of $I(x)$ suggest it is implausible that $f(x)$ is a good match to y

(for example, using Pukelsheim's 3 sigma rule).

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma_{e_i}^2 + \sigma_{f_i}^2 + \sigma_{\epsilon_i}^2}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.

The implausibility calculation can be performed over collections of outputs.

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.

The implausibility calculation can be performed over collections of outputs.

We can reject parts of the input space based on maximum implausibility or a vector version of implausibility based on Mahalobis distance.

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.
The implausibility calculation can be performed over collections of outputs.
We can reject parts of the input space based on maximum implausibility or a vector version of implausibility based on Mahalobis distance.

Having identified a non-implausible region of the input space, we resample the reduced region, refit the emulators and repeat the analysis, continuing until we identify the region of acceptable matches.

History matching by implausibility

Durham University

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.
The implausibility calculation can be performed over collections of outputs.
We can reject parts of the input space based on maximum implausibility or a vector version of implausibility based on Mahalobis distance.

Having identified a non-implausible region of the input space, we resample the reduced region, refit the emulators and repeat the analysis, continuing until we identify the region of acceptable matches.

I. Vernon, M. Goldstein, R. Bower (2010), Galaxy Formation: a Bayesian Uncertainty Analysis (with discussion) , Bayesian Analysis, 5(4): 619–670.

History matching by implausibility

$$I_i(x) = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\mathrm{Var}(z_i - \mathrm{E}(f_i(x)))} = \frac{(z_i - \mathrm{E}(f_i(x)))^2}{\sigma^2_{e_i} + \sigma^2_{f_i} + \sigma^2_{\epsilon_i}}$$

Inputs x with large $I(x)$ are unlikely to be appropriate choices.
The implausibility calculation can be performed over collections of outputs.
We can reject parts of the input space based on maximum implausibility or a vector version of implausibility based on Mahalobis distance.

Having identified a non-implausible region of the input space, we resample the reduced region, refit the emulators and repeat the analysis, continuing until we identify the region of acceptable matches.

I. Vernon, M. Goldstein, R. Bower (2010), Galaxy Formation: a Bayesian Uncertainty Analysis (with discussion) , Bayesian Analysis, 5(4): 619–670.

Andrianakis, Vernon, McCreesh, McKinley, Oakley, Nsubuga, Goldstein, White (2017) History matching of a complex epidemiological model of human immunodeficiency virus transmission by using variance emulation, J R Stat Soc Ser C,717-740

The HMER package

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

The HMER package

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

Andrew Iskauskas, Michael Goldstein, Ian Vernon (Durham)

The HMER package

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

Andrew Iskauskas, Michael Goldstein, Ian Vernon (Durham)

Nicky McCreesh, Danny Scarponi, Richard White (LSHTM)

The HMER package

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

Andrew Iskauskas, Michael Goldstein, Ian Vernon (Durham)

Nicky McCreesh, Danny Scarponi, Richard White (LSHTM)

TJ McKinley (Exeter)

The HMER package

Durham University

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

Andrew Iskauskas, Michael Goldstein, Ian Vernon (Durham)

Nicky McCreesh, Danny Scarponi, Richard White (LSHTM)

TJ McKinley (Exeter)

building on a previous collaboration funded by MRC.

The HMER package

HMER (history matching and emulation in R) is a system developed under a collaboration, with Wellcome Trust funding, between

Andrew Iskauskas, Michael Goldstein, Ian Vernon (Durham)

Nicky McCreesh, Danny Scarponi, Richard White (LSHTM)

TJ McKinley (Exeter)

building on a previous collaboration funded by MRC.

The package is customised for epidemic models, but the underlying methodology is fully general.

The HMER package

The package is available from CRAN.

You can find detailed documentation at

https://github.com/andy-iskauskas/hmer

The HMER package

The package is available from CRAN.

You can find detailed documentation at

https://github.com/andy-iskauskas/hmer

The project web-page, which has lots of support material is at

https://hmer-package.github.io/website/

The HMER package

The package is available from CRAN.

You can find detailed documentation at

https://github.com/andy-iskauskas/hmer

The project web-page, which has lots of support material is at

https://hmer-package.github.io/website/

The programme has been extensively tested.

The HMER package

The package is available from CRAN.

You can find detailed documentation at

https://github.com/andy-iskauskas/hmer

The project web-page, which has lots of support material is at

https://hmer-package.github.io/website/

The programme has been extensively tested.

For example history matching a complex deterministic model for the country-level implementation of tuberculosis vaccines to 114 countries, fitting to 9–13 target measures, by varying 19–22 input parameters.

The HMER package

The package is available from CRAN.

You can find detailed documentation at

https://github.com/andy-iskauskas/hmer

The project web-page, which has lots of support material is at

https://hmer-package.github.io/website/

The programme has been extensively tested.

For example history matching a complex deterministic model for the country-level implementation of tuberculosis vaccines to 114 countries, fitting to 9–13 target measures, by varying 19–22 input parameters.

105 countries were successfully matched (i.e. producing many parameter choices which match history)

The remaining 9 countries revealed evidence of model or data misspecification.

Multi-level emulation

If the simulator is slow to evaluate, then we may be able to create a fast approximation f^* to the simulator f,

Multi-level emulation

If the simulator is slow to evaluate, then we may be able to create a fast approximation f^* to the simulator f,

We use many runs of the fast simulator to create the emulators

$$f_i^*(x) = \sum_j \beta_{ij}^* g_{ij}(x_{A(i)}) + u_i^*(x)$$

Multi-level emulation

If the simulator is slow to evaluate, then we may be able to create a fast approximation f^* to the simulator f,

We use many runs of the fast simulator to create the emulators

$$f_i^*(x) = \sum_j \beta_{ij}^* g_{ij}(x_{A(i)}) + u_i^*(x)$$

Use the collections β_i^*, u_i^* as priors for our judgements for the elements of the emulator for the slow simulator

$$f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$$

Multi-level emulation

Given this prior, we may create a small but informative design to run for the slow simulator and use this to update the prior and construct the full emulator.

Multi-level emulation

Given this prior, we may create a small but informative design to run for the slow simulator and use this to update the prior and construct the full emulator.

(so this involves multi-level, many-output design, based on the choices of active variables for each output)

Multi-level emulation

Given this prior, we may create a small but informative design to run for the slow simulator and use this to update the prior and construct the full emulator.

(so this involves multi-level, many-output design, based on the choices of active variables for each output)

Comment Simulators should be designed to support this process.

Multi-level emulation

Given this prior, we may create a small but informative design to run for the slow simulator and use this to update the prior and construct the full emulator.

(so this involves multi-level, many-output design, based on the choices of active variables for each output)

Comment Simulators should be designed to support this process.

J. Cumming, M. Goldstein Bayes Linear Uncertainty Analysis for Oil Reservoirs Based on Multiscale Computer Experiments (2009), in the Handbook of Applied Bayesian Analysis,eds A. O'Hagan, M. West, OUP

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish:

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish:

interpolation (roughly, predicting function values within the convex hull of the evaluations used to build the emulator)

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish:

interpolation (roughly, predicting function values within the convex hull of the evaluations used to build the emulator)

and **extrapolation** (Predicting function values outside the convex hull).

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish: **interpolation** (roughly, predicting function values within the convex hull of the evaluations used to build the emulator)

and **extrapolation** (Predicting function values outside the convex hull).

Small samples in high dimensions mean that almost all emulator evaluations are extrapolations.

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish: **interpolation** (roughly, predicting function values within the convex hull of the evaluations used to build the emulator)

and **extrapolation** (Predicting function values outside the convex hull).

Small samples in high dimensions mean that almost all emulator evaluations are extrapolations.

In such cases, when using an emulator of form $f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$, it is the global part of the emulator $\sum_j \beta_{ij} g_{ij}(x_{A(i)})$ which dominates the emulator prediction.

Interpolation versus extrapolation

One of the most important issues in building reliable emulators is to distinguish: **interpolation** (roughly, predicting function values within the convex hull of the evaluations used to build the emulator)

and **extrapolation** (Predicting function values outside the convex hull).

Small samples in high dimensions mean that almost all emulator evaluations are extrapolations.

In such cases, when using an emulator of form $f_i(x) = \sum_j \beta_{ij} g_{ij}(x_{A(i)}) + u_i(x)$, it is the global part of the emulator $\sum_j \beta_{ij} g_{ij}(x_{A(i)})$ which dominates the emulator prediction.

So this form needs to be chosen carefully for scientific plausibility and needs to be subjected to careful diagnostic testing.

Varying coefficient emulators

When we rely on the global component of the emulator for our predictions, we cannot use the local form to soak up the residual variation. Therefore, it is a good idea to be more careful in specifying this form.

Varying coefficient emulators

When we rely on the global component of the emulator for our predictions, we cannot use the local form to soak up the residual variation. Therefore, it is a good idea to be more careful in specifying this form.

One approach is to use varying coefficient emulators. These replace

$$f(x) = \sum_j \beta_j g_j(x) + u(x)$$

Varying coefficient emulators

When we rely on the global component of the emulator for our predictions, we cannot use the local form to soak up the residual variation. Therefore, it is a good idea to be more careful in specifying this form.

One approach is to use varying coefficient emulators. These replace

$$f(x) = \sum_j \beta_j g_j(x) + u(x)$$

with the varying coefficient form

$$f(x) = \sum_j \beta_j(x) g_j(x) + u(x)$$

Varying coefficient emulators

When we rely on the global component of the emulator for our predictions, we cannot use the local form to soak up the residual variation. Therefore, it is a good idea to be more careful in specifying this form.

One approach is to use varying coefficient emulators. These replace

$$f(x) = \sum_j \beta_j g_j(x) + u(x)$$

with the varying coefficient form

$$f(x) = \sum_j \beta_j(x) g_j(x) + u(x)$$

where the $\beta_j(x)$ are themselves second order stationary processes.

Varying coefficient emulators

When we rely on the global component of the emulator for our predictions, we cannot use the local form to soak up the residual variation. Therefore, it is a good idea to be more careful in specifying this form.

One approach is to use varying coefficient emulators. These replace

$$f(x) = \sum_j \beta_j g_j(x) + u(x)$$

with the varying coefficient form

$$f(x) = \sum_j \beta_j(x) g_j(x) + u(x)$$

where the $\beta_j(x)$ are themselves second order stationary processes.

For how these are fitted and used for computer design problems, see **Wilson, Amy L., Goldstein, Michael & Dent, Chris J.** (2022). Varying Coefficient Models and Design Choice for Bayes Linear Emulation of Complex Computer Models with Limited Model Evaluations. SIAM/ASA Journal on Uncertainty Quantification 10(1): 350-378.

Limitations of physical models

A physical model is a description of the way in which system properties (the inputs to the model) affect system behaviour (the output of the model).

Limitations of physical models

A physical model is a description of the way in which system properties (the inputs to the model) affect system behaviour (the output of the model).

This description involves two basic types of simplification.

Limitations of physical models

A physical model is a description of the way in which

system properties (the inputs to the model)

affect system behaviour (the output of the model).

This description involves two basic types of simplification.

(i) we approximate the properties of the system (as these properties are too complicated to describe fully and anyway we don't know them)

Limitations of physical models

A physical model is a description of the way in which
system properties (the inputs to the model)
affect system behaviour (the output of the model).

This description involves two basic types of simplification.

(i) we approximate the properties of the system (as these properties are too complicated to describe fully and anyway we don't know them)

(ii) we approximate the rules for finding system behaviour given system properties (because of necessary mathematical and numerical simplifications, and because we do not fully understand the relationships which govern the process).

Limitations of physical models

A physical model is a description of the way in which
system properties (the inputs to the model)
affect system behaviour (the output of the model).

This description involves two basic types of simplification.

(i) we approximate the properties of the system (as these properties are too complicated to describe fully and anyway we don't know them)

(ii) we approximate the rules for finding system behaviour given system properties (because of necessary mathematical and numerical simplifications, and because we do not fully understand the relationships which govern the process).

Neither of these approximations invalidates the modelling process.

Limitations of physical models

A physical model is a description of the way in which system properties (the inputs to the model) affect system behaviour (the output of the model).

This description involves two basic types of simplification.

(i) we approximate the properties of the system (as these properties are too complicated to describe fully and anyway we don't know them)

(ii) we approximate the rules for finding system behaviour given system properties (because of necessary mathematical and numerical simplifications, and because we do not fully understand the relationships which govern the process).

Neither of these approximations invalidates the modelling process.

Problems only arise when we forget these simplifications and confuse the analysis of the model with the corresponding analysis for the physical system itself.

Structural discrepancy

Structural uncertainty assessessment should form a central part of the problem analysis. We may distinguish two types of model discrepancy.

Structural discrepancy

Structural uncertainty assessessment should form a central part of the problem analysis. We may distinguish two types of model discrepancy.

(i) **Internal discrepancy**

Any aspect of discrepancy we can assess by direct experiments on the computer simulator.

Structural discrepancy

Structural uncertainty assessessment should form a central part of the problem analysis. We may distinguish two types of model discrepancy.

(i) **Internal discrepancy**

Any aspect of discrepancy we can assess by direct experiments on the computer simulator.

(ii) **External discrepancy**

This arises from the inherent limitations of the modelling process embodied in the simulator.

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

we may allow parameters to vary over time/space.

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

we may allow parameters to vary over time/space.

We assess internal discrepancy by
(i) carrying out detailed experiments to determine discrepancy variance for certain input choices,

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

we may allow parameters to vary over time/space.

We assess internal discrepancy by
(i) carrying out detailed experiments to determine discrepancy variance for certain input choices,

(ii) using emulation to extend the variance assessment over the input space.

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

we may allow parameters to vary over time/space.

We assess internal discrepancy by
(i) carrying out detailed experiments to determine discrepancy variance for certain input choices,

(ii) using emulation to extend the variance assessment over the input space.

(Simulators should be designed and implemented to support this methodology.)

Internal discrepancy

We may assess aspects of internal discrepancy by, for example

varying parameters/forcing functions held fixed in the standard analysis,

we may add random noise to the state vector which the model propagates,

we may allow parameters to vary over time/space.

We assess internal discrepancy by
(i) carrying out detailed experiments to determine discrepancy variance for certain input choices,

(ii) using emulation to extend the variance assessment over the input space.

(Simulators should be designed and implemented to support this methodology.)

M. Goldstein and N. Huntley (2017) Bayes linear emulation, history matching and forecasting for complex computer simulators, in The Handbook of Uncertainty Quantification, Ghanem, Higdon, Owhad (eds), Springer

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)

It is determined by a combination of expert judgements and statistical estimation.

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)

It is determined by a combination of expert judgements and statistical estimation.

The simplest way to incorporate external discrepancy is to add an extra component of uncertainty to the simulator outputs.

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)
It is determined by a combination of expert judgements and statistical estimation.
The simplest way to incorporate external discrepancy is to add an extra component of uncertainty to the simulator outputs.

For example we may introduce, say, 10% additional error to account for structural discrepancy.

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)
It is determined by a combination of expert judgements and statistical estimation.
The simplest way to incorporate external discrepancy is to add an extra component of uncertainty to the simulator outputs.

For example we may introduce, say, 10% additional error to account for structural discrepancy.

Better is to consider what we know about the limitations of the model, and build a probabilistic representation of additional features of the relationship between system properties and behaviour.

External discrepancy

External discrepancy arises from the inherent limitations of the modelling process embodied in the simulator (plus call aspects of potential internal discrepancy investigations that could not be carried out)
It is determined by a combination of expert judgements and statistical estimation.
The simplest way to incorporate external discrepancy is to add an extra component of uncertainty to the simulator outputs.

For example we may introduce, say, 10% additional error to account for structural discrepancy.

Better is to consider what we know about the limitations of the model, and build a probabilistic representation of additional features of the relationship between system properties and behaviour.

Sometimes, this is called **reification**,
(from reify - to treat an abtract concept as if it was real).

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

For example, if the emulator for the simulator is
$f(x) = \sum_j \beta_j g_{ij}(x) + u(x)$

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

For example, if the emulator for the simulator is
$f(x) = \sum_j \beta_j g_{ij}(x) + u(x)$

then our emulator for the reified form might be
$f^r(x) = \sum_j \beta_j^r g_{ij}(x) + u^r(x)$

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

For example, if the emulator for the simulator is
$f(x) = \sum_j \beta_j g_{ij}(x) + u(x)$

then our emulator for the reified form might be
$f^r(x) = \sum_j \beta_j^r g_{ij}(x) + u^r(x)$

where the elements of the simulator form act as priors for the reified form.

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

For example, if the emulator for the simulator is
$f(x) = \sum_j \beta_j g_{ij}(x) + u(x)$

then our emulator for the reified form might be
$f^r(x) = \sum_j \beta_j^r g_{ij}(x) + u^r(x)$

where the elements of the simulator form act as priors for the reified form.

If we add this component of variation to our implausibility measures, then this is equivalent to carrying out a history match for the reified model.

Reified discrepancy

We cannot evaluate the reified simulator, but we can emulate it.

For example, if the emulator for the simulator is
$f(x) = \sum_j \beta_j g_{ij}(x) + u(x)$

then our emulator for the reified form might be
$f^r(x) = \sum_j \beta_j^r g_{ij}(x) + u^r(x)$

where the elements of the simulator form act as priors for the reified form.

If we add this component of variation to our implausibility measures, then this is equivalent to carrying out a history match for the reified model.

M. Goldstein and J.C.Rougier (2009). Reified Bayesian modelling and inference for physical systems (with discussion), JSPI, 139, 1221-1239

Forecasting

Suppose that we want to predict some future outcome y_p, given observed historical data on earlier values z_h. The Bayes linear update of y_p given z_h requires the joint variance structure of y_p, z_h.

Forecasting

Suppose that we want to predict some future outcome y_p, given observed historical data on earlier values z_h. The Bayes linear update of y_p given z_h requires the joint variance structure of y_p, z_h.

This can be derived directly from the decomposition

$$y = f(x^*) + \epsilon^*, \; z_h = y_h + e$$

Forecasting

Suppose that we want to predict some future outcome y_p, given observed historical data on earlier values z_h. The Bayes linear update of y_p given z_h requires the joint variance structure of y_p, z_h.

This can be derived directly from the decomposition

$$y = f(x^*) + \epsilon^*, \; z_h = y_h + e$$

This analysis is tractable even for large systems. Careful discrepancy assessment will

Forecasting

Suppose that we want to predict some future outcome y_p, given observed historical data on earlier values z_h. The Bayes linear update of y_p given z_h requires the joint variance structure of y_p, z_h.

This can be derived directly from the decomposition

$$y = f(x^*) + \epsilon^*, \; z_h = y_h + e$$

This analysis is tractable even for large systems. Careful discrepancy assessment will

(i) correct our overconfidence in our projections
(by adding appropriate levels of additional uncertainty)

(ii) increase our forecast accuracy
(by correcting for systematic biases in our simulator).

Forecasting

Suppose that we want to predict some future outcome y_p, given observed historical data on earlier values z_h. The Bayes linear update of y_p given z_h requires the joint variance structure of y_p, z_h.

This can be derived directly from the decomposition

$$y = f(x^*) + \epsilon^*, \; z_h = y_h + e$$

This analysis is tractable even for large systems. Careful discrepancy assessment will

(i) correct our overconfidence in our projections
(by adding appropriate levels of additional uncertainty)

(ii) increase our forecast accuracy
(by correcting for systematic biases in our simulator).

Goldstein, M. and Rougier, J. C. (2006) 'Bayes linear calibrated prediction for complex systems.', Journal of the American Statistical Association.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$.
Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

For each input x, produce upper and lower bounds $U(x)$ and $L(x)$ (for example taking the mean plus or minus 3 SD).

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

For each input x, produce upper and lower bounds $U(x)$ and $L(x)$ (for example taking the mean plus or minus 3 SD).

Search x space and find the minimum value of $U(x)$. Call this U.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

For each input x, produce upper and lower bounds $U(x)$ and $L(x)$ (for example taking the mean plus or minus 3 SD).

Search x space and find the minimum value of $U(x)$. Call this U.

Any x with $L(x) > U$ can be removed from the space as ”implausible” to be the minimum.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

For each input x, produce upper and lower bounds $U(x)$ and $L(x)$ (for example taking the mean plus or minus 3 SD).

Search x space and find the minimum value of $U(x)$. Call this U.

Any x with $L(x) > U$ can be removed from the space as ”implausible” to be the minimum.

[Wave 2, etc] Having removed the non-implausible values, we run a second wave, and repeat. Continue iterating until we reach “good” solutions.

Optimisation via history matching

Suppose that we want to optimise (say minimise) a complex function $f(x)$. Here's how we may use history matching methods to find the class of solutions which are close to the optimum.

[Wave 1] Take a sample of x values, and build an emulator for $f(x)$.

For each input x, produce upper and lower bounds $U(x)$ and $L(x)$ (for example taking the mean plus or minus 3 SD).

Search x space and find the minimum value of $U(x)$. Call this U.

Any x with $L(x) > U$ can be removed from the space as ”implausible” to be the minimum.

[Wave 2, etc] Having removed the non-implausible values, we run a second wave, and repeat. Continue iterating until we reach “good” solutions.

Du, Hailiang, Sun, Wei, Goldstein, Michael & Harrison, Gareth (2021). Optimization via Statistical Emulation and Uncertainty Quantification: Hosting Capacity Analysis of Distribution Networks. IEEE Access 9: 118472-118483.

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

[2] Assess structural discrepancy δ_1 and δ_2 for the two models.

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

[2] Assess structural discrepancy δ_1 and δ_2 for the two models.

[3] Assess uncertainty for the combined system by
(i) making random draws $\hat{y}$ from $\hat{f}_1(x) + \delta_1$

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

[2] Assess structural discrepancy δ_1 and δ_2 for the two models.

[3] Assess uncertainty for the combined system by
(i) making random draws $\hat{y}$ from $\hat{f}_1(x) + \delta_1$
(ii) making random draws $\hat{u}$ from $\hat{f}_2(\hat{y}, w) + \delta_2$

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

[2] Assess structural discrepancy δ_1 and δ_2 for the two models.

[3] Assess uncertainty for the combined system by
(i) making random draws $\hat{y}$ from $\hat{f}_1(x) + \delta_1$
(ii) making random draws $\hat{u}$ from $\hat{f}_2(\hat{y}, w) + \delta_2$

This approach is modular. We can emulate and assess structural discrepancy over each model separately, then combine all of the specifications to carry out the composite uncertainty analysis over any subsystems of interest.

Systems integration

Much systems work involves individual systems talking to each other. So, for example, we might have system one, $f_1(x)$, whose output y forms an input to system two , $f_2(y, w)$. The way to assess uncertainty for the combined system, over the whole range of input choices, is

[1] Build emulators, $\hat{f}_1$ and $\hat{f}_2$ for f_1 and f_2

[2] Assess structural discrepancy δ_1 and δ_2 for the two models.

[3] Assess uncertainty for the combined system by
(i) making random draws $\hat{y}$ from $\hat{f}_1(x) + \delta_1$
(ii) making random draws $\hat{u}$ from $\hat{f}_2(\hat{y}, w) + \delta_2$

This approach is modular. We can emulate and assess structural discrepancy over each model separately, then combine all of the specifications to carry out the composite uncertainty analysis over any subsystems of interest.

Oughton, R, Goldstein, M. & Hemmings, J. (2022). Intermediate Variable Emulation: using internal processes in simulators to build more informative emulators. SIAM/ASA Journal on Uncertainty Quantification: 268-293.

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Key features of this methodology are

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Key features of this methodology are

(i) **simulator emulation**, to allow us to explore the full range of outputs of the simulator

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Key features of this methodology are

(i) **simulator emulation**, to allow us to explore the full range of outputs of the simulator

(ii) **structural discrepancy modelling**, to make reliable uncertainty comparisons with the real world

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Key features of this methodology are

(i) **simulator emulation**, to allow us to explore the full range of outputs of the simulator

(ii) **structural discrepancy modelling**, to make reliable uncertainty comparisons with the real world

(iii) **system forecasting and optimisation** to identify real world properties and decisions which are appropriate and robust across all conditions consistent with historical outcomes

Concluding comments

History matching is a flexible, tractable and robust Bayes, and Bayes linear, methodology for analysing complex real world phenomena which are modelled by computer simulators.

Key features of this methodology are

(i) **simulator emulation**, to allow us to explore the full range of outputs of the simulator

(ii) **structural discrepancy modelling**, to make reliable uncertainty comparisons with the real world

(iii) **system forecasting and optimisation** to identify real world properties and decisions which are appropriate and robust across all conditions consistent with historical outcomes

Reminder: here's the HMER project web-page

https://hmer-package.github.io/website/