Challenges in Computer Experiments

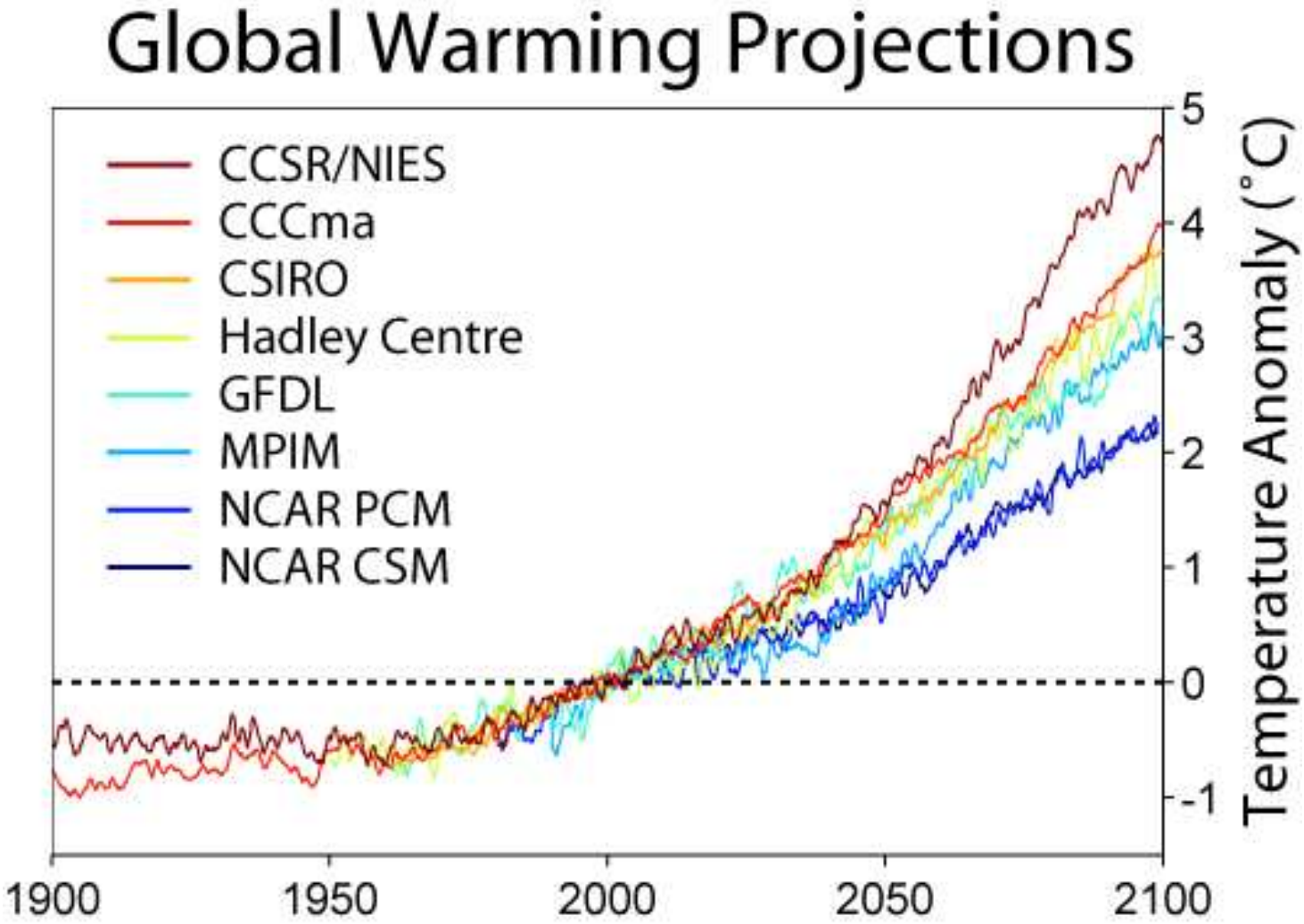
Richard Wilkinson

University of Nottingham

Brasília, 23 July 2010

Challenges in computer experiments

Climate Predictions, IPCC AR4, 2007



Computer experiments

Many statistical challenges are posed by computer experiments, and many of these have only recently begun to be tackled by statisticians.

For example, for deterministic computer experiments, the usual statistical concepts of replication, randomization and blocking are irrelevant.

- Instead we must explicitly account for uncertainty ourselves.

Methods to quantify, analyse and incorporate uncertainty in the application of computer experiments are attracting increasing attention amongst users of simulation. I shall talk about a couple of areas that are actively being worked on:

- Meta-modelling
- Model error

Complex computer experiments

Throughout this talk, think of the simulator as a deterministic function

$$\eta : \mathcal{X} \to \mathcal{Y}$$

Typically, both the input and output space will be vectors (often of high dimension)

We think of two input types

- Control inputs x - e.g., location, time, output index, etc.
- Model parameters θ - e.g., model constants, unknown initial conditions, fudge factors, etc.

and write $y = \eta(x, \theta)$ for the model output (possibly a temporal-spatial field)

Challenges

Some typical problems we might be interested in:

- Calibration
  - How do we estimate unknown model parameters θ given observations $\mathcal{D}$ of the physical system?
- Prediction
  - Given all the unknowns, what is our best prediction and how confident are we in it?
- Uncertainty analysis
  - How does uncertainty about the model inputs θ feed through the model?
- Sensitivity analysis
  - How can we apportion variation in the output to variation in the input parameters? In other words, what inputs are driving the output?

Incorporating and accounting for uncertainty

Perhaps the biggest challenge faced is incorporating uncertainty in computer experiments.

We are used to dealing with uncertainty in physical experiments. But if your computer model is deterministic, there is no natural source of variation and so the experimenter must carefully assess where errors might arise.

Types of uncertainty in computer experiments:

- Observation errors
- Parametric uncertainty
- Model error
- Code uncertainty

Representation of uncertainty

Probability (Bayesian) can be used to represent uncertainty.

- Under minimal rationality assumptions, probability can be shown to be the only rational way to represent uncertainty.

Representation of uncertainty

Probability (Bayesian) can be used to represent uncertainty.

- Under minimal rationality assumptions, probability can be shown to be the only rational way to represent uncertainty.
- Probability is subjective probability - distributions represent degrees of belief of individuals. There is no escaping this interpretation in many applications!

Representation of uncertainty

Probability (Bayesian) can be used to represent uncertainty.

- Under minimal rationality assumptions, probability can be shown to be the only rational way to represent uncertainty.
- Probability is subjective probability - distributions represent degrees of belief of individuals. There is no escaping this interpretation in many applications!
- All uncertainty quantities θ can be given distributions $\pi(\theta)$ that represent our (an experts?) uncertainty about the value - this doesn't mean that they are random quantities, just that we don't know their value.
  - Even unknown function will be described by probability distributions across a class of unknown functions

Representation of uncertainty

Probability (Bayesian) can be used to represent uncertainty.

- Under minimal rationality assumptions, probability can be shown to be the only rational way to represent uncertainty.
- Probability is subjective probability - distributions represent degrees of belief of individuals. There is no escaping this interpretation in many applications!
- All uncertainty quantities θ can be given distributions $\pi(\theta)$ that represent our (an experts?) uncertainty about the value - this doesn't mean that they are random quantities, just that we don't know their value.
  - Even unknown function will be described by probability distributions across a class of unknown functions
- Bayesian approach uses the principle of conditionality, and always (where possible) conditions on our data.

Bayesian Inference

The basics of Bayesian inference are very simple.

- If quantity θ is unknown we describe our uncertainty by prior distribution $\pi(\theta)$.
- Suppose we have data D from model $\pi(D|\theta)$
- Then conditional on observing this data, our posterior distribution is

$$\pi(\theta|D) = \frac{\pi(\theta)\pi(D|\theta)}{\pi(D)}$$

In general we just use the relationship

$$\text{posterior} \propto \text{prior} \times \text{likelihood (model)}$$

Given a model and prior the entire Bayesian statistical approach is fully specified. In practice, computational difficulties add interest!

Meta-modelling

Code uncertainty

For complex simulators, run times might be long, ruling out brute-force approaches such as Monte Carlo methods.

Code uncertainty

For complex simulators, run times might be long, ruling out brute-force approaches such as Monte Carlo methods.

Consequently, we will only know the simulator output at a finite number of points.

- We call this *code uncertainty*.
- All inference must be done using a finite ensemble of model runs

$$\mathcal{D}_{sim} = \{(\theta_i, \eta(\theta_i))\}_{i=1,\ldots,N}$$

- If θ is not in the ensemble, then we are uncertainty about the value of $\eta(\theta)$.

Code uncertainty

For complex simulators, run times might be long, ruling out brute-force approaches such as Monte Carlo methods.

Consequently, we will only know the simulator output at a finite number of points.

- We call this *code uncertainty*.
- All inference must be done using a finite ensemble of model runs

$$\mathcal{D}_{sim} = \{(\theta_i, \eta(\theta_i))\}_{i=1,\ldots,N}$$

- If θ is not in the ensemble, then we are uncertainty about the value of $\eta(\theta)$.

If θ is multidimensional, then even short run times can rule out brute force approaches

- $\dim(\theta) \in \mathbb{R}^{10}$ then 1000 simulator runs is only enough for one point in each corner of the design space.

The design of computational experiments is an active field in statistics.

Meta-modelling

Idea: If the simulator is expensive, build a cheap model of it and use this in any analysis.

'a model of the model'

We call this meta-model an *emulator* of our simulator.

Meta-modelling

Idea: If the simulator is expensive, build a cheap model of it and use this in any analysis.

'a model of the model'

We call this meta-model an *emulator* of our simulator.

There are many types of emulator.

- ideally an emulator should come with an assessment of its accuracy
- rather than just predicting $\eta(\theta)$ it should predict $\pi(\eta(\theta)|\mathcal{D}_{sim})$ - our uncertainty about the simulator value given the ensemble $\mathcal{D}_{sim}$.

Gaussian process emulators are most popular choice for emulator. Built using

- an ensemble of model runs $\mathcal{D}_{sim} = \{(\theta_i, \eta(\theta_i))\}_{i=1,\ldots,N}$
- expert opinion about the simulator output.

Meta-modelling

Gaussian Process Emulators

Gaussian processes provide a flexible nonparametric distributions for our prior beliefs about the functional form of the simulator:

$$\eta(\cdot) \sim GP(m(\cdot), \sigma^2 c(\cdot, \cdot))$$

where $m(\cdot)$ is the prior mean function, and $c(\cdot, \cdot)$ is the prior covariance function (semi-definite).

Meta-modelling

Gaussian Process Emulators

Gaussian processes provide a flexible nonparametric distributions for our prior beliefs about the functional form of the simulator:

$$\eta(\cdot) \sim GP(m(\cdot), \sigma^2 c(\cdot, \cdot))$$

where $m(\cdot)$ is the prior mean function, and $c(\cdot, \cdot)$ is the prior covariance function (semi-definite).

Definition If $f(\cdot) \sim GP(m(\cdot), c(\cdot, \cdot))$ then for any collection of inputs $x_1, \ldots, x_n$ the vector

$$(f(x_1), \ldots, f(x_n))^T \sim MVN(m(\mathbf{x}), \sigma^2 \mathbf{\Sigma})$$

where $\Sigma_{ij} = c(x_i, x_j)$.

Meta-modelling

Gaussian Process Emulators

Gaussian processes are invariant under Bayesian updating.

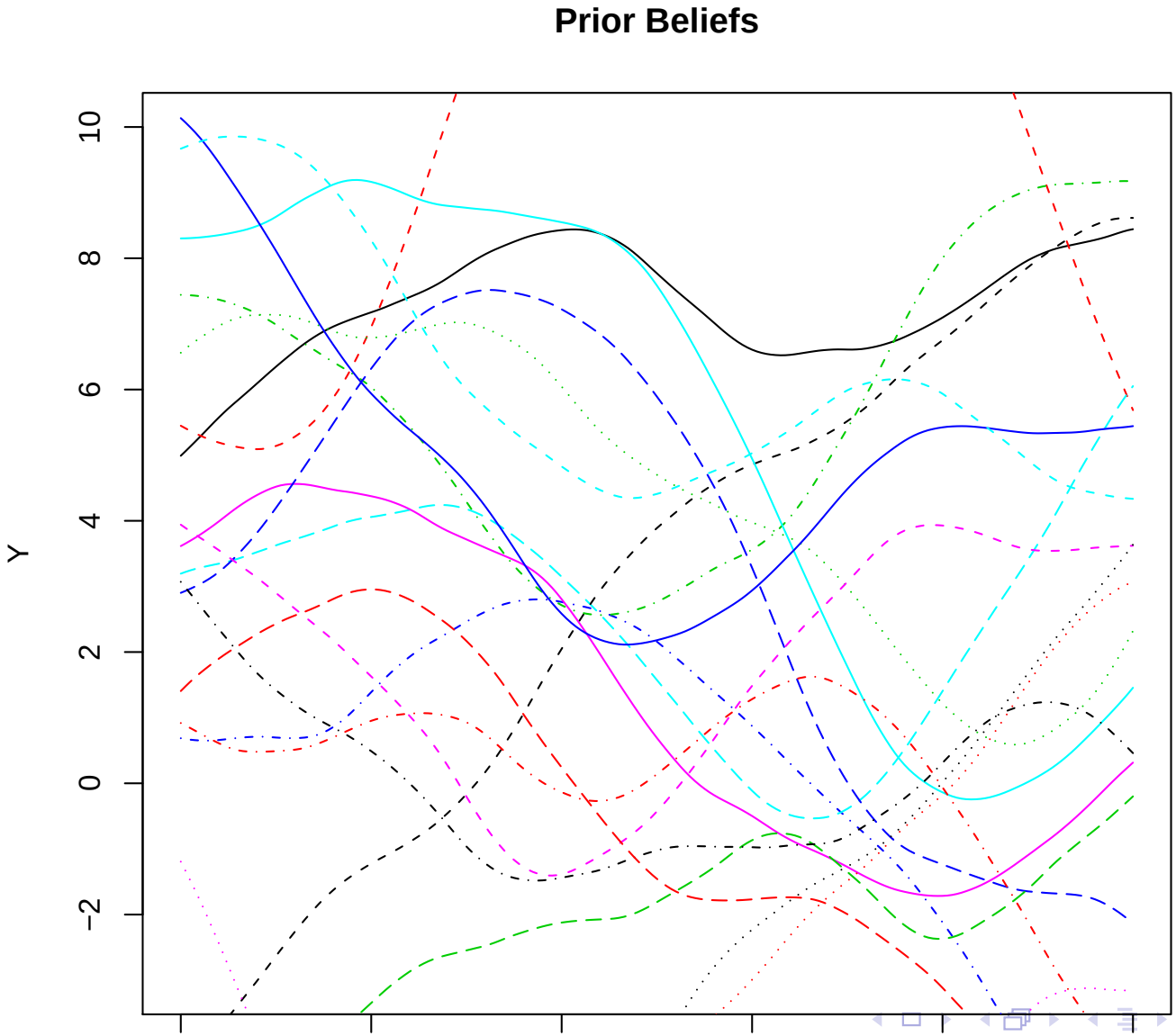
If we observe the ensemble of model runs $\mathcal{D}_{sim}$, then update our prior belief about η in light of the ensemble of model runs:

$$\eta(\cdot)|\mathcal{D}_{sim} \sim GP(m^*(\cdot), \sigma^2 c^*(\cdot,\cdot))$$

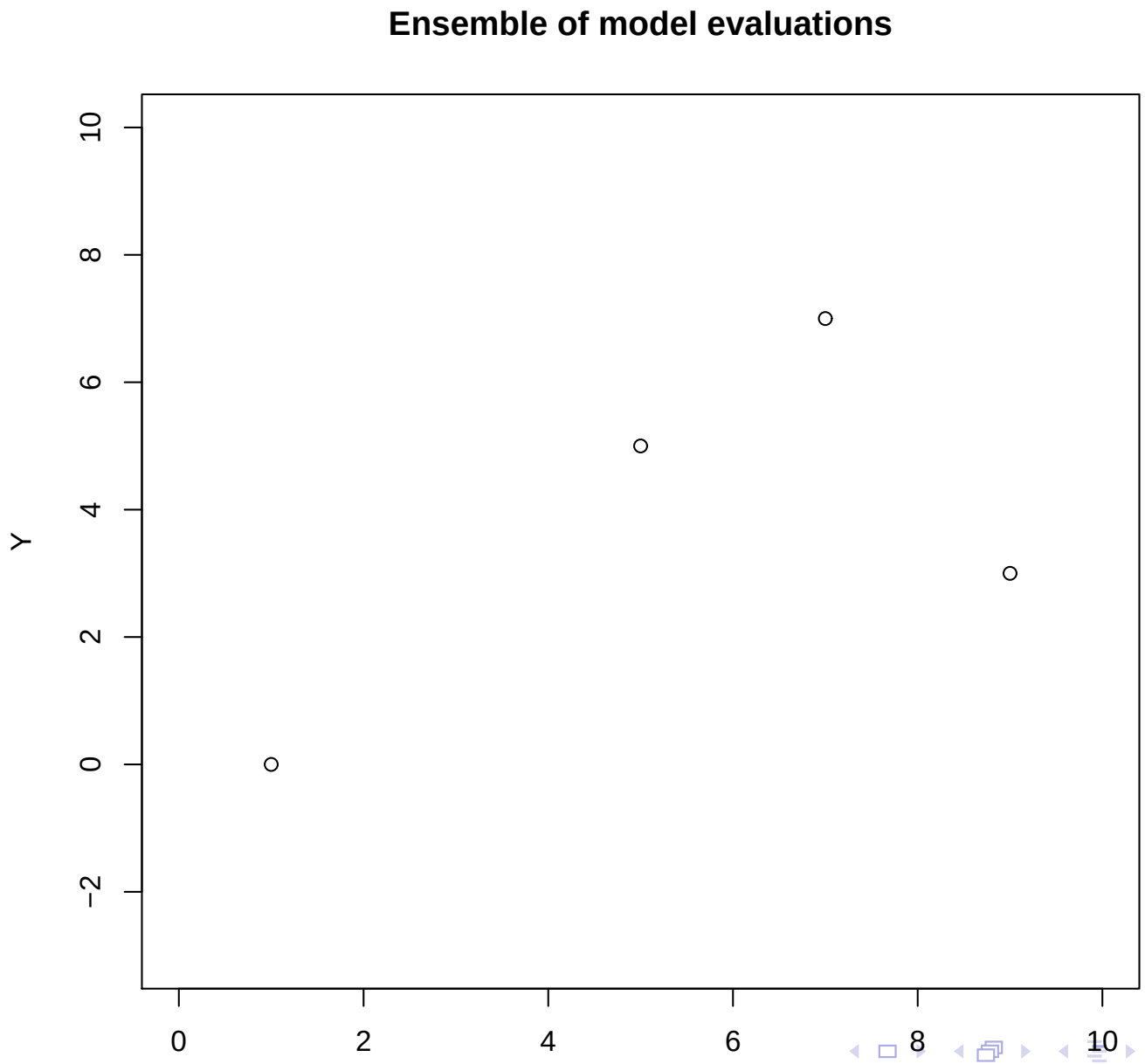
where m^* and c^* are the posterior mean and covariance functions (simple functions of $\mathcal{D}_{sim}$, m and c).

Gaussian Process Illustration

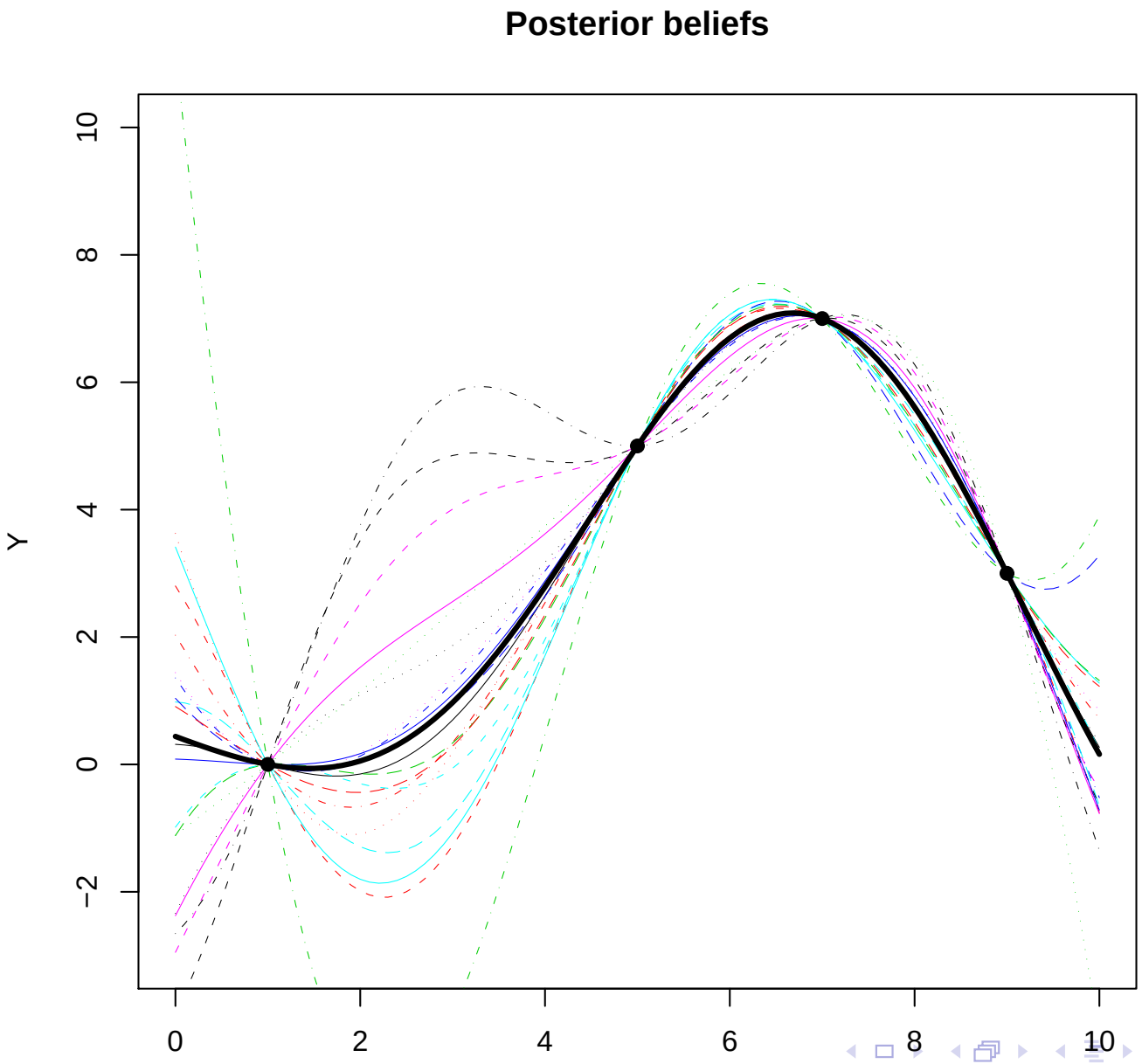
Zero mean



Gaussian Process Illustration



Gaussian Process Illustration



Emulator choices

$$\eta(x) = h(x)\beta + u(x)$$

$$\text{emulator} = \text{mean structure} + \text{residual}$$

Emulator choices

$$\eta(x) = h(x)\beta + u(x)$$
$$\text{emulator} = \text{mean structure} + \text{residual}$$

$u(x)$ can be taken to be a zero-mean Gaussian process

$$u(\cdot) \sim GP(0, c(\cdot, \cdot))$$

Emulator choices

$$\eta(x) = h(x)\beta + u(x)$$
$$\text{emulator} = \text{mean structure} + \text{residual}$$

$u(x)$ can be taken to be a zero-mean Gaussian process

$$u(\cdot) \sim GP(0, c(\cdot, \cdot))$$

Emulator choices:

- mean structure $h(x)$
  - $1, x, x^2, \ldots$, Legendre polynomials?

Emulator choices

$$\eta(x) = h(x)\beta + u(x)$$
$$\text{emulator} = \text{mean structure} + \text{residual}$$

$u(x)$ can be taken to be a zero-mean Gaussian process

$$u(\cdot) \sim GP(0, c(\cdot, \cdot))$$

Emulator choices:

- mean structure $h(x)$
  - $1, x, x^2, \ldots$, Legendre polynomials?
- covariance function $c(\cdot, \cdot)$
  - Stationary? Smooth?
  - Length-scale?

Calibration

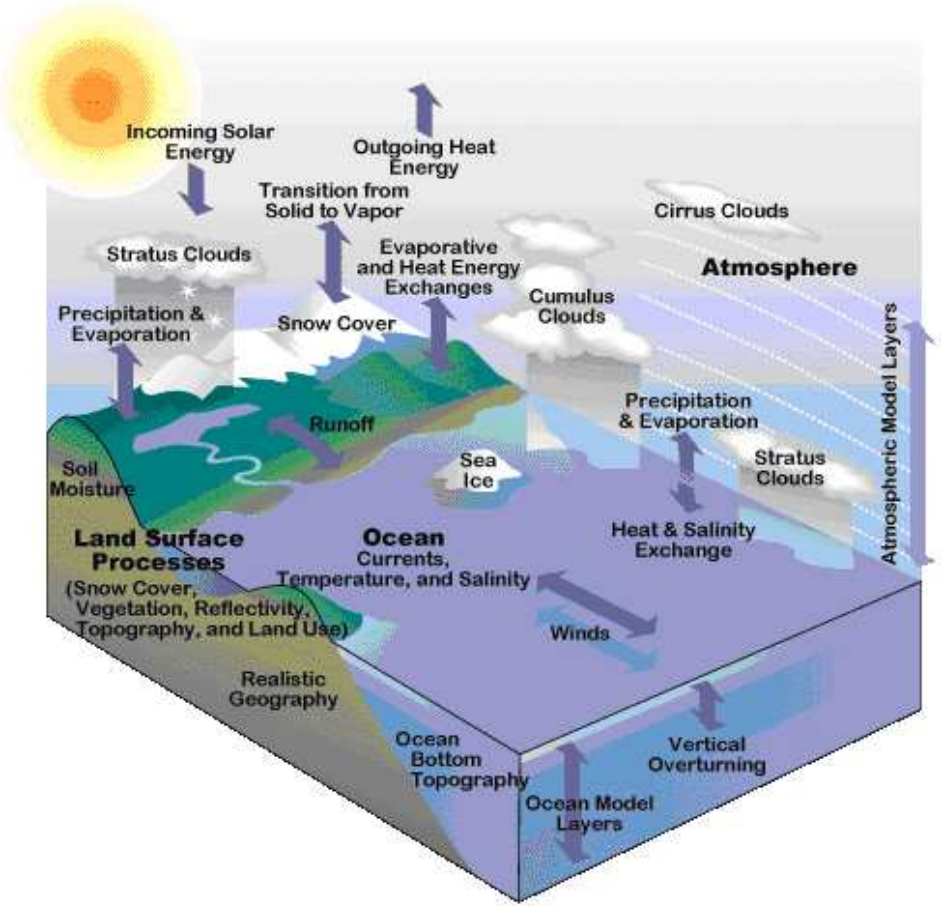
Calibration

Inverse problems

For forwards models we specify parameters θ and i.c.s and the model generates output $\mathcal{D}_{sim}$. Usually, we are interested in the inverse-problem, i.e., observe data $\mathcal{D}_{field}$, want to estimate parameter values.

Different terminology:

- Calibration
- Parameter estimation
- Inverse-problem
- Bayesian inference



Calibration

In a Bayesian approach, we can think of the calibration problem as an inverse probability problem, and find $\pi(\theta|\mathcal{D}, \eta)$.

Calibration

In a Bayesian approach, we can think of the calibration problem as an inverse probability problem, and find $\pi(\theta|\mathcal{D}, \eta)$.
What does this represent? Or rather, what do we believe we are doing?

- Does θ have a physical interpretation, i.e., are we estimating physical parameters?
- Or is θ interpretted statistically? i.e., θ is the value that best explains the data given the model - cf. the coefficients in a linear regression.

e.g., ocean physics models must be run with viscosity several orders of magnitude too large.

Calibration

In a Bayesian approach, we can think of the calibration problem as an inverse probability problem, and find $\pi(\theta|\mathcal{D}, \eta)$.
What does this represent? Or rather, what do we believe we are doing?

- Does θ have a physical interpretation, i.e., are we estimating physical parameters?
- Or is θ interpretted statistically? i.e., θ is the value that best explains the data given the model - cf. the coefficients in a linear regression.

e.g., ocean physics models must be run with viscosity several orders of magnitude too large.
When can we interpret the value found for θ as a physical value?

Calibration

In a Bayesian approach, we can think of the calibration problem as an inverse probability problem, and find $\pi(\theta|\mathcal{D}, \eta)$.
What does this represent? Or rather, what do we believe we are doing?

- Does θ have a physical interpretation, i.e., are we estimating physical parameters?
- Or is θ interpretted statistically? i.e., θ is the value that best explains the data given the model - cf. the coefficients in a linear regression.

e.g., ocean physics models must be run with viscosity several orders of magnitude too large.
When can we interpret the value found for θ as a physical value?

- If the model is a perfect representation of the system

Calibration

In a Bayesian approach, we can think of the calibration problem as an inverse probability problem, and find $\pi(\theta|\mathcal{D}, \eta)$.
What does this represent? Or rather, what do we believe we are doing?

- Does θ have a physical interpretation, i.e., are we estimating physical parameters?
- Or is θ interpretted statistically? i.e., θ is the value that best explains the data given the model - cf. the coefficients in a linear regression.

e.g., ocean physics models must be run with viscosity several orders of magnitude too large.
When can we interpret the value found for θ as a physical value?

- If the model is a perfect representation of the system
- When the model is imperfect, but we have a description (that we believe) of the discrepancy between model and system.

All models are wrong, but some are useful

Kennedy and O'Hagan 2001

- Suppose we have a computer model $\eta(x, \theta)$ that we wish to use to make predictions of a physical system $\zeta(x)$ using observations $D(x)$.
  - θ are model parameters we wish to learn
  - x are control/index parameters, e.g., time, location etc.

All models are wrong, but some are useful

Kennedy and O'Hagan 2001

- Suppose we have a computer model $\eta(x, \theta)$ that we wish to use to make predictions of a physical system $\zeta(x)$ using observations $D(x)$.
  - θ are model parameters we wish to learn
  - x are control/index parameters, e.g., time, location etc.
- Standard approach to calibration is the *best-input* approach, where we assume there is a single 'best' value of θ, which we call $\hat{\theta}$. The model run at $\hat{\theta}$, the hat-run $\eta(\hat{\theta})$, is the best model prediction.

All models are wrong, but some are useful

Kennedy and O'Hagan 2001

- Suppose we have a computer model $\eta(x, \theta)$ that we wish to use to make predictions of a physical system $\zeta(x)$ using observations $D(x)$.
  - θ are model parameters we wish to learn
  - x are control/index parameters, e.g., time, location etc.
- Standard approach to calibration is the *best-input* approach, where we assume there is a single 'best' value of θ, which we call $\hat{\theta}$. The model run at $\hat{\theta}$, the hat-run $\eta(\hat{\theta})$, is the best model prediction.
- Usual statistical assumption is that

$$D(t) = \eta(t, \hat{\theta}) + e_t$$

where $e.$ is a white noise (iid) error process. This is a poor assumption for most models: if the model is imperfect, then residuals $D - \eta(\theta)$ may be correlated, even if the real measurement error process is iid.

Bayesian Calibration Framework II

- Instead, assume that we observe reality plus measurement error.

$$D(t) = \zeta(t) + e(t)$$

Often, $e(\cdot)$ will be a white noise process with known mean and variance.

Bayesian Calibration Framework II

- Instead, assume that we observe reality plus measurement error.

$$D(t) = \zeta(t) + e(t)$$

  Often, $e(\cdot)$ will be a white noise process with known mean and variance.

- Introduce a model error (discrepancy) term. Assume that reality is the best model prediction plus an error

$$\zeta(t) = \eta(t, \hat{\theta}) + \delta(t).$$

  Note δ does not depend on θ.

Bayesian Calibration Framework II

- Instead, assume that we observe reality plus measurement error.

$$D(t) = \zeta(t) + e(t)$$

  Often, $e(\cdot)$ will be a white noise process with known mean and variance.
- Introduce a model error (discrepancy) term. Assume that reality is the best model prediction plus an error

$$\zeta(t) = \eta(t, \hat{\theta}) + \delta(t).$$

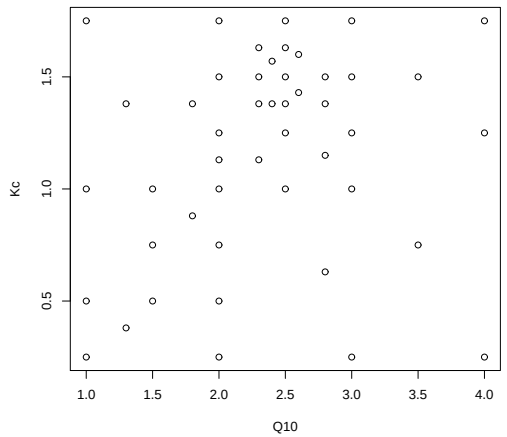
  Note δ does not depend on θ.
- Argue that $\eta(\cdot, \hat{\theta})$ and $\delta(\cdot)$ are independent. Kennedy and O'Hagan 2001 use Gaussian processes to model both the model η and the error δ.

Example: UVic Earth System Climate Model

With Nathan Urban (Penn State)

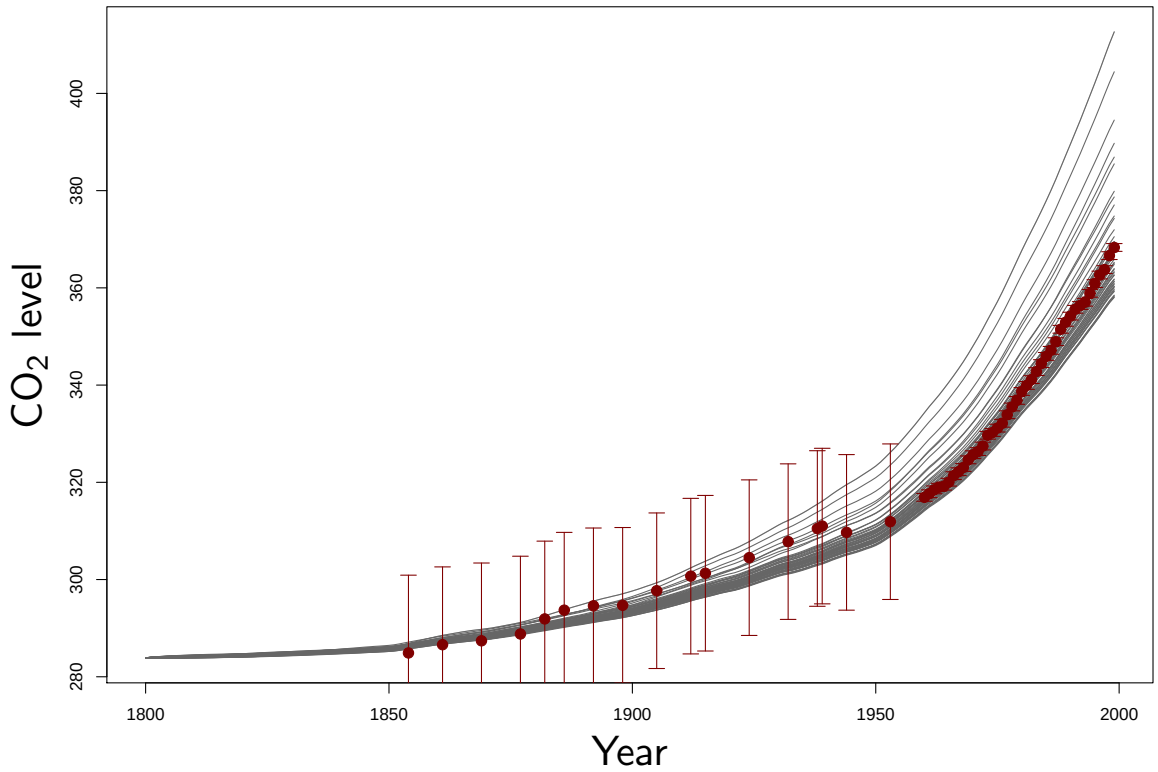
UVic ESCM is an intermediate complexity model with a general circulation ocean and dynamic/thermodynamic sea-ice components coupled to a simple energy/moisture balance atmosphere. It has a dynamic vegetation and terrestrial carbon cycle model (TRIFFID) as well as an inorganic carbon cycle.

- Inputs: Q_{10} = respiration sensitivity to temperature (carbon source) and $K_c = CO_2$ fertilization of photosynthesis (carbon sink).
- Output: time-series of CO_2 values, cumulative carbon flux measurements, ...
- 48 member ensemble, grid design D, output $\mathcal{D}_{sim}$ $(48 \times n)$.



The U.Vic Model

- 60 field measurements, $\mathcal{D}_{field}$:
  - 40 instrumental CO_2 measurements from 1960-1999
  - 17 ice core CO_2 measurements
  - 3 cumulative ocean carbon flux measurements

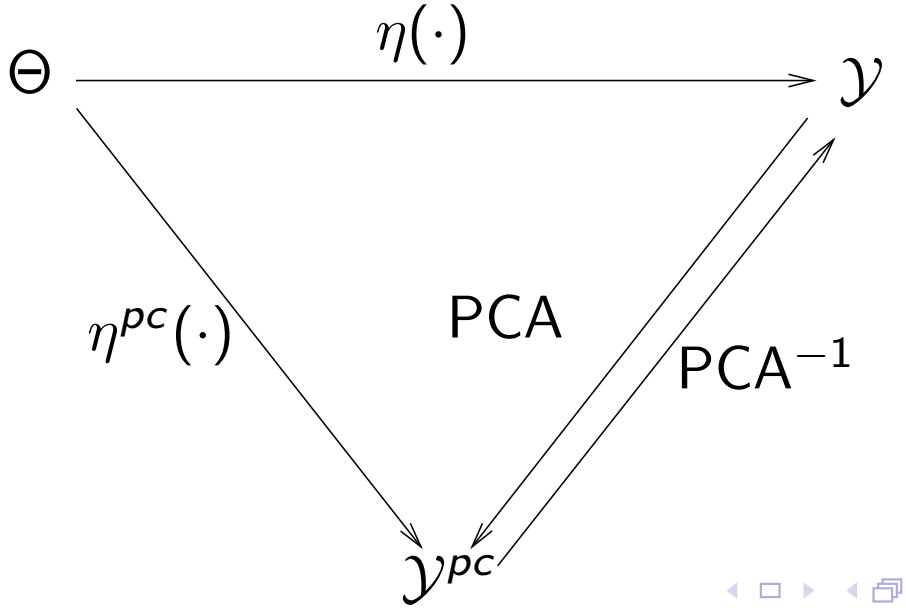


Multivariate Models

Wilkinson 2010

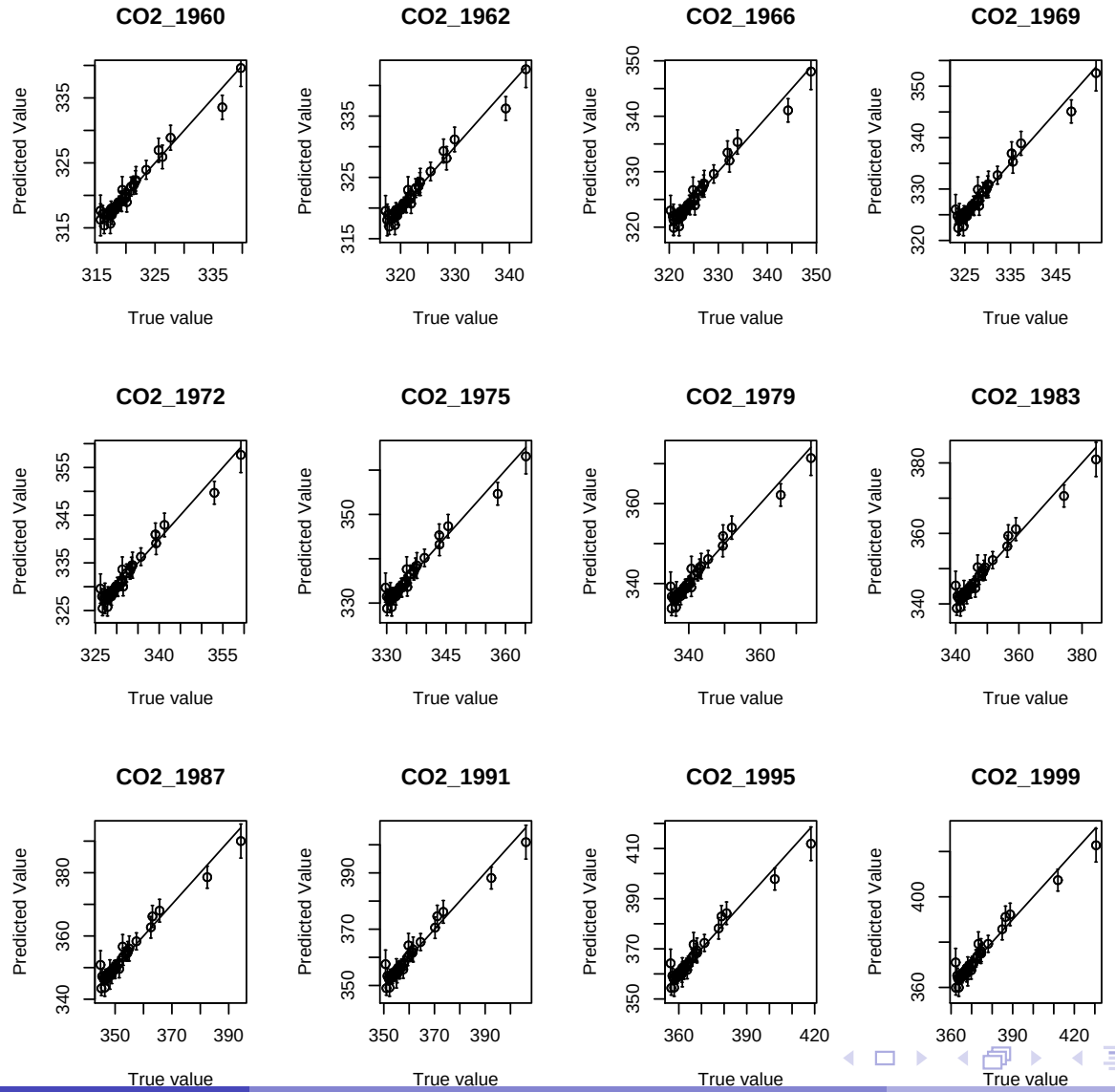
The output from UVic is a temporal spatial field of predicted CO2 values.

To emulate this we need an extension of the univariate GP method produced previously.
If model outputs are highly correlated we can reduce the dimension of the output by projecting the data onto some lower dimensional manifold $\mathcal{Y}^{pc}$.
We can then emulate the function that maps the input space Θ to the reduced dimensional output space $\mathcal{Y}^{pc}$.



Emulator Diagnostics

Cross-validation plots, instrumental data only (1960:1999), using 3PCs (99.2% of variance explained)



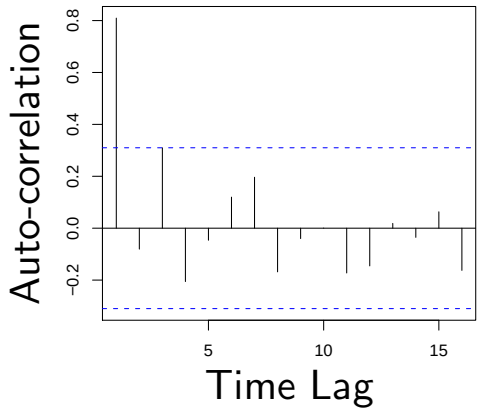
UVic Model Discrepancy

The calibration framework used is:

$$\mathcal{D}_{field}(t) = \eta(\theta, t) + \delta(t) + e(t)$$

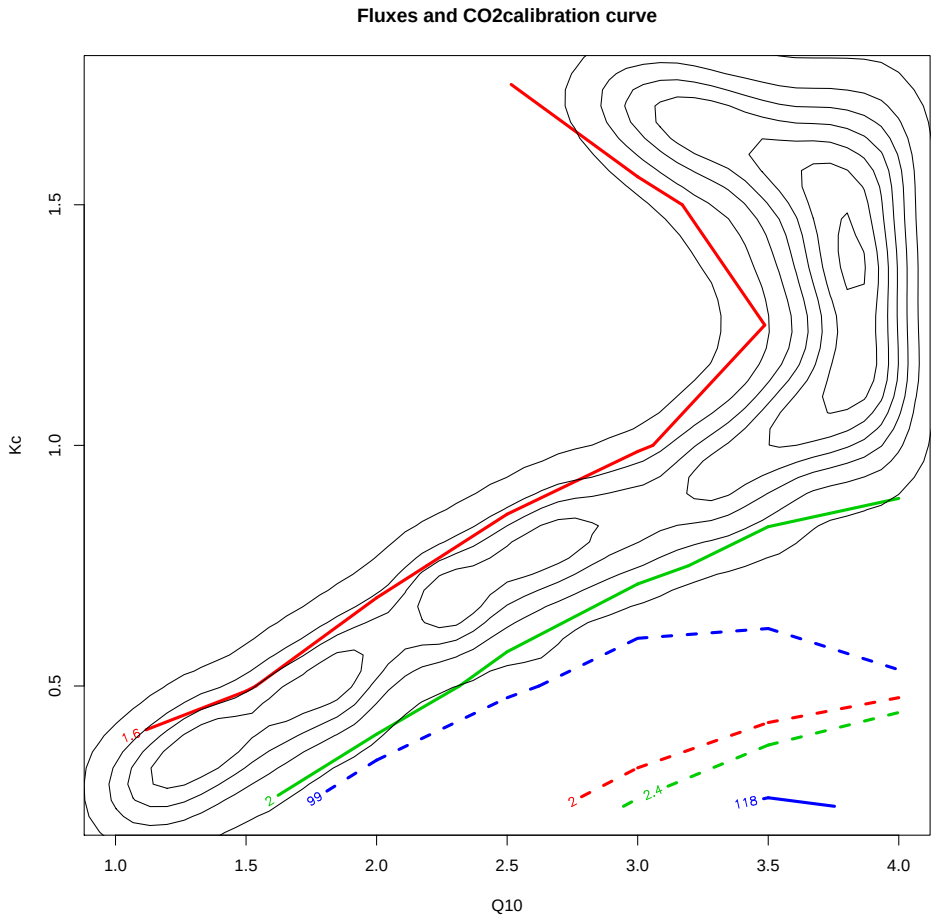
The model predicts the underlying trend, but real climate fluctuates around this. We model

- discrepancy as an AR1 process: $\delta(0) \sim N(0, \sigma_\delta^2)$, and $\delta(t) = \rho\delta(t-1) + N(0, \sigma_\delta^2)$.
- Measurement error as heteroscedastic independent random noise $e(t) \sim N(0, \lambda(t))$.



Results

After several hours of MCMC (Metropolis-within-Gibbs) we find



The data are equally well explained by a range of parameter values.

Learning Model Error

Learning Model Error

In some situations the modellers will be able to describe clearly what form they expect the model error to take. Often however, we will need to learn the form and magnitude of the model error.

Learning Model Error

In some situations the modellers will be able to describe clearly what form they expect the model error to take. Often however, we will need to learn the form and magnitude of the model error.

Consider the dynamic model η where we have state vector x_t which evolves through time

$$x_{t+1} = \eta(x_t, u_t)$$

and suppose we observe

$$y_t = x_t + e_t$$

where $e_t \sim N(0, \sigma^2)$.

Learning Model Error

In some situations the modellers will be able to describe clearly what form they expect the model error to take. Often however, we will need to learn the form and magnitude of the model error.

Consider the dynamic model η where we have state vector x_t which evolves through time

$$x_{t+1} = \eta(x_t, u_t)$$

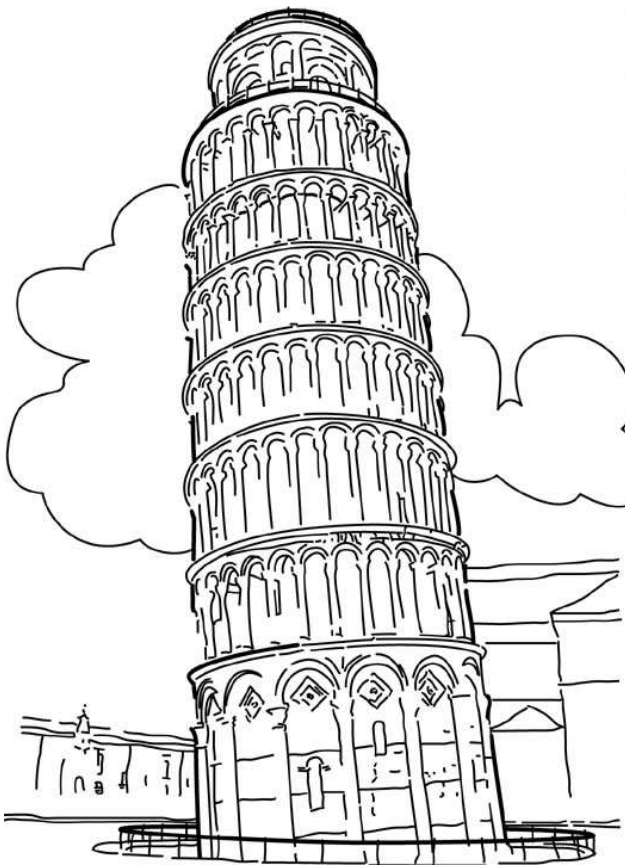
and suppose we observe

$$y_t = x_t + e_t$$

where $e_t \sim N(0, \sigma^2)$.
We ask whether there is a model error term δ that could be learnt:

- State evolution: $x_{t+1} = \eta(x_t, u_t) + \delta(x_t, u_t) + \epsilon_t$

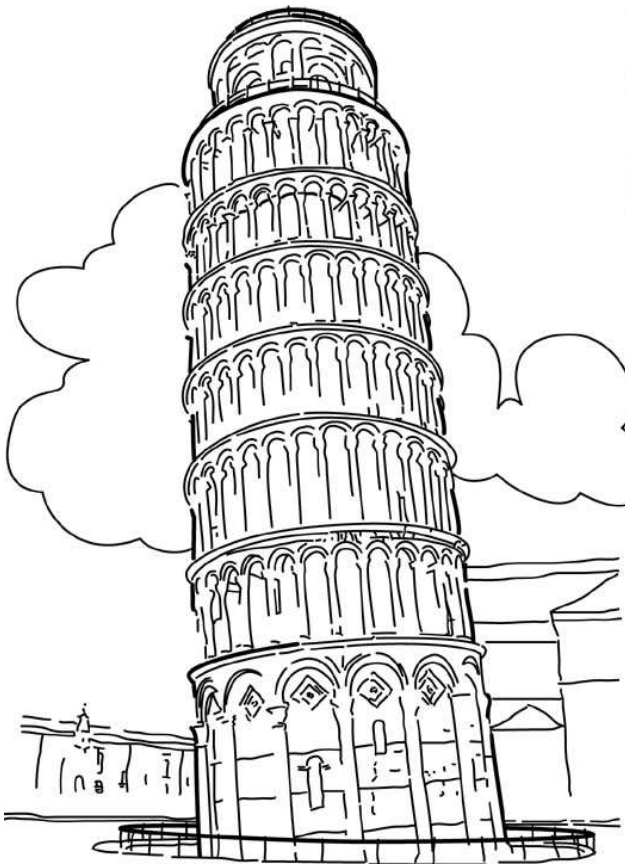
Toy Example: Freefall



Consider an experiment where we drop a weight from a tower and measure its position x_t every Δt seconds.

- Noisy observation: $y_n \sim N(x_n, \sigma^2_{obs})$

Toy Example: Freefall



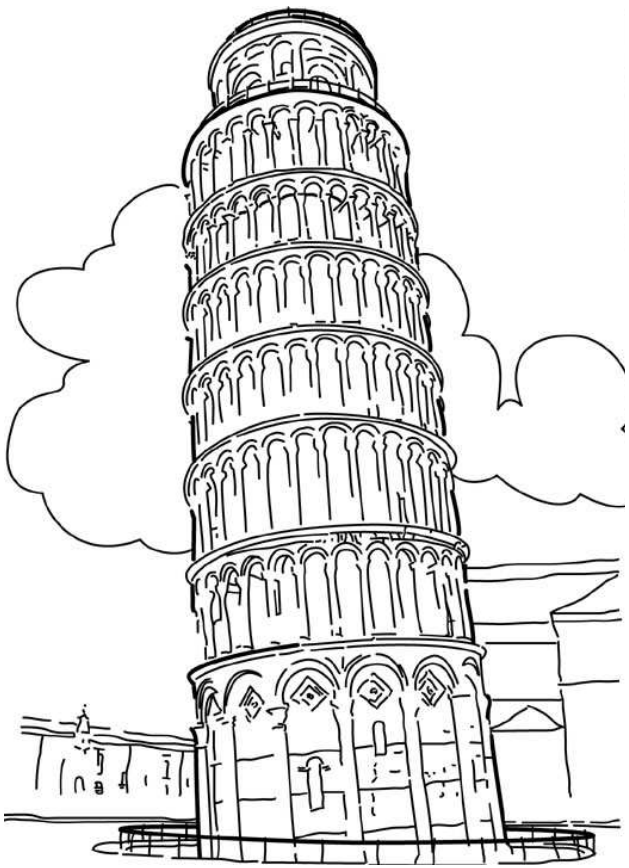
Consider an experiment where we drop a weight from a tower and measure its position x_t every Δt seconds.

- Noisy observation: $y_n \sim N(x_n, \sigma^2_{obs})$

Suppose we are given a computer model based on

$$\frac{\mathrm{dv}}{\mathrm{dt}} = g$$

Toy Example: Freefall



Consider an experiment where we drop a weight from a tower and measure its position x_t every Δt seconds.

- Noisy observation: $y_n \sim N(x_n, \sigma^2_{obs})$

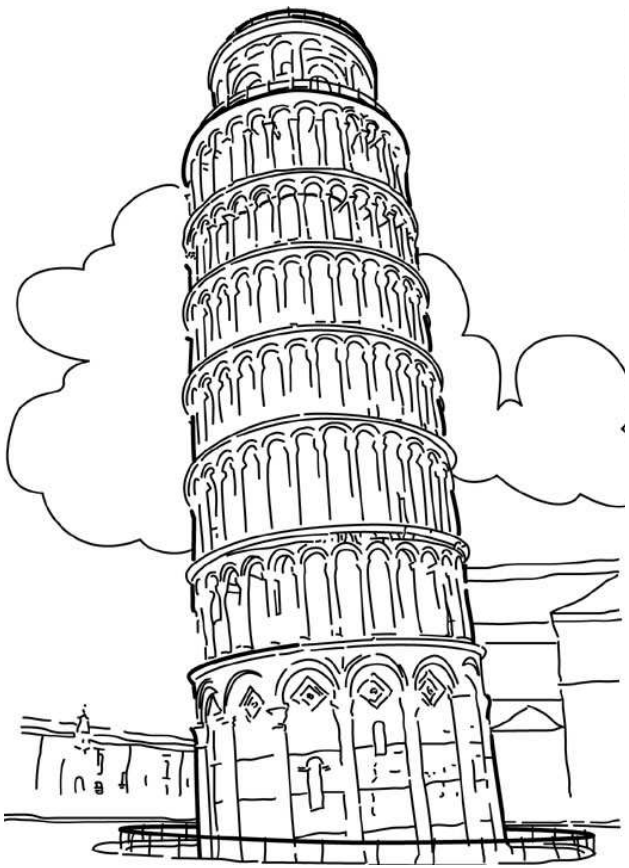
Suppose we are given a computer model based on

$$\frac{\mathrm{dv}}{\mathrm{dt}} = g$$

Which gives predictions at the observations of

- $x_{n+1} = x_n + v_k \Delta t + \frac{1}{2} g (\Delta t)^2$
- $v_{n+1} = v_n + g \Delta t$

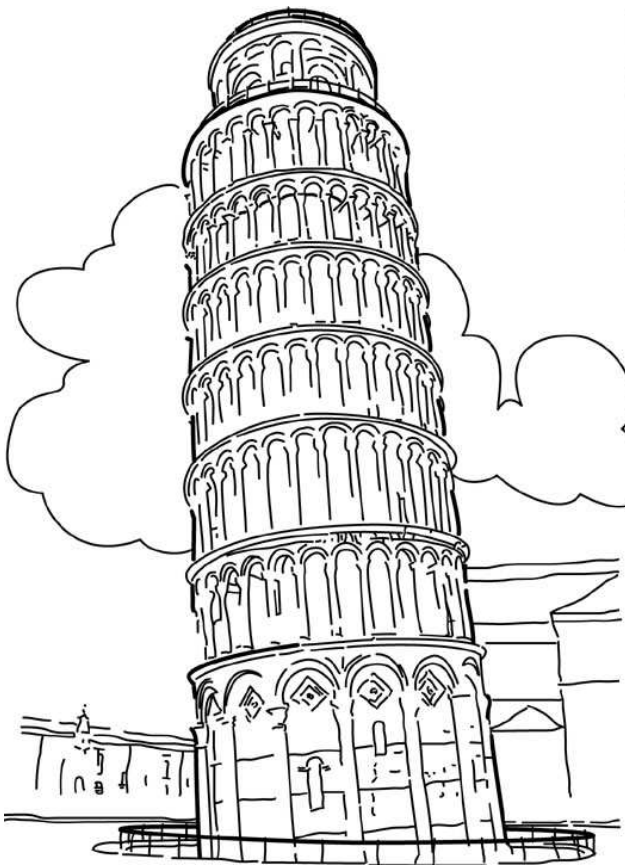
Toy Example: Freefall



Assume that the 'true' dynamics include a Stokes' drag term

$$\frac{\mathrm{dv}}{\mathrm{dt}} = g - kv$$

Toy Example: Freefall



Assume that the 'true' dynamics include a Stokes' drag term

$$\frac{\mathrm{dv}}{\mathrm{dt}} = g - kv$$

Which gives single time step updates

$$x_{n+1} = x_n + \frac{1}{k}(\frac{g}{k} - v_t)(\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} - 1) + \frac{\mathrm{g}\Delta\mathrm{t}}{\mathrm{k}}$$

$$v_{n+1} = (v_n - \frac{g}{k})\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} + \frac{\mathrm{g}}{\mathrm{k}}$$

Model Error Term

In this toy problem, the true discrepancy function can be calculated.

- It is a two dimensional function

$$\delta = \begin{pmatrix} \delta_x \\ \delta_v \end{pmatrix} = \zeta - f$$

giving the difference between the one time-step ahead dynamics of reality and the prediction from our model.

If we expand $\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}}$ to second order we find

$$\delta(x, v, t) = \begin{pmatrix} \delta_x \\ \delta_v \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{-gk(\Delta t)^2}{2} \end{pmatrix} - v_t \begin{pmatrix} \frac{k(\Delta t)^2}{2} \\ k\Delta t(1 - \frac{k\Delta t}{2}) \end{pmatrix}$$

Model Error Term

In this toy problem, the true discrepancy function can be calculated.

- It is a two dimensional function

$$\delta = \begin{pmatrix} \delta_x \\ \delta_v \end{pmatrix} = \zeta - f$$

  giving the difference between the one time-step ahead dynamics of reality and the prediction from our model.

If we expand $\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}}$ to second order we find

$$\delta(x, v, t) = \begin{pmatrix} \delta_x \\ \delta_v \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{-gk(\Delta t)^2}{2} \end{pmatrix} - v_t \begin{pmatrix} \frac{k(\Delta t)^2}{2} \\ k\Delta t(1 - \frac{k\Delta t}{2}) \end{pmatrix}$$

This is solely a function of v.

- Note, to learn δ we only have the observations $y_1, \ldots, y_n$ of $x_1, \ldots, x_n$ - we do not observe v.

Expected form of the discrepancy

Forget the previous slide.

Expected form of the discrepancy

Forget the previous slide.

There are three variables in this problem, displacement, velocity and time (x, v, t) so we might think to model δ as a function of these three terms.

Expected form of the discrepancy

Forget the previous slide.

There are three variables in this problem, displacement, velocity and time (x, v, t) so we might think to model δ as a function of these three terms.

However, the principal of universality says that nature is consistent throughout all space and time (background independence), so with a little thought we might reason that δ should be independent of x and t.

Expected form of the discrepancy

Forget the previous slide.

There are three variables in this problem, displacement, velocity and time (x, v, t) so we might think to model δ as a function of these three terms.

However, the principal of universality says that nature is consistent throughout all space and time (background independence), so with a little thought we might reason that δ should be independent of x and t.

With input from an experienced user of our model, it is feasible we might be able to get other information such as that δ approximately scales with v, or at least that the error is small at low speeds and large at high speeds.

Parametric approach

Start with a parametric model for δ, e.g.,

$$\delta_x(x) = \sum_{i=0}^{p} \alpha_i x^i + \sum_{i=0}^{q} \beta_i v^i + \epsilon$$

where $\epsilon \sim N(0, \tau)$, with $\theta_x = (\tau, \alpha_0, \ldots, \alpha_p, \beta_0, \ldots, \beta_q)$ unknown (and similarly for δ_v).

Parametric approach

Start with a parametric model for δ, e.g.,

$$\delta_x(x) = \sum_{i=0}^{p} \alpha_i x^i + \sum_{i=0}^{q} \beta_i v^i + \epsilon$$

where $\epsilon \sim N(0, \tau)$, with $\theta_x = (\tau, \alpha_0, \ldots, \alpha_p, \beta_0, \ldots, \beta_q)$ unknown (and similarly for δ_v).

- The problem now looks like a missing data problem:

$$\pi(x_{0:t}, y_{0:t}|\theta) = \pi(y_{0:t}|x_{0:t})\pi(x_{0:t}|\theta)$$

  is easy to work with when $x_{0:t}$ and $y_{0:t}$ are known. However $x_{0:t}$ is missing and $\pi(y_{0:t}|\theta)$ is unknown.

Parametric approach

Start with a parametric model for δ, e.g.,

$$\delta_x(x) = \sum_{i=0}^{p} \alpha_i x^i + \sum_{i=0}^{q} \beta_i v^i + \epsilon$$

where $\epsilon \sim N(0, \tau)$, with $\theta_x = (\tau, \alpha_0, \ldots, \alpha_p, \beta_0, \ldots, \beta_q)$ unknown (and similarly for δ_v).

- The problem now looks like a missing data problem:

$$\pi(x_{0:t}, y_{0:t}|\theta) = \pi(y_{0:t}|x_{0:t})\pi(x_{0:t}|\theta)$$

  is easy to work with when $x_{0:t}$ and $y_{0:t}$ are known. However $x_{0:t}$ is missing and $\pi(y_{0:t}|\theta)$ is unknown.
- The EM algorithm can be used to estimate the best fitting model for δ from the specified class of models.

An EM algorithm for estimating δ

We iterate between the E and M steps:

- E-step: Calculate

$$Q(\theta, \theta^{(m)}) = \mathbb{E}_{X_{0:T}}\left[\log \pi(X_{0:T}, y_{0:T}|\theta) \mid y_{0:T}, \theta^{(m)}\right]$$

- M-step: Maximize Q and set

$$\theta^{(m+1)} = \arg\max_{\theta} Q(\theta, \theta^{(m)})$$

An EM algorithm for estimating δ

We iterate between the E and M steps:

- E-step: Calculate

$$Q(\theta, \theta^{(m)}) = \mathbb{E}_{X_{0:T}}\left[\log \pi(X_{0:T}, y_{0:T}|\theta) \mid y_{0:T}, \theta^{(m)}\right]$$

  - This expectation is taken with respect to the distribution $\pi(x_{0:T} \mid y_{0:T}, \theta^{(m)})$

- M-step: Maximize Q and set

$$\theta^{(m+1)} = \arg\max_{\theta} Q(\theta, \theta^{(m)})$$

An EM algorithm for estimating δ

We iterate between the E and M steps:

- E-step: Calculate

$$Q(\theta, \theta^{(m)}) = \mathbb{E}_{X_{0:T}}\left[\log \pi(X_{0:T}, y_{0:T}|\theta) \mid y_{0:T}, \theta^{(m)}\right]$$

  - This expectation is taken with respect to the distribution $\pi(x_{0:T} \mid y_{0:T}, \theta^{(m)})$
  - This is the smoothing distribution from the fully specified model, and is not known analytically. However, it can be sampled from and the Monte Carlo expectation used for Q (stochastic EM algorithm, Wei and Tanner 1990).

- M-step: Maximize Q and set

$$\theta^{(m+1)} = \arg\max_{\theta} Q(\theta, \theta^{(m)})$$

An EM algorithm for estimating δ

We iterate between the E and M steps:

- E-step: Calculate

$$Q(\theta, \theta^{(m)}) = \mathbb{E}_{X_{0:T}}\left[\log \pi(X_{0:T}, y_{0:T}|\theta) \mid y_{0:T}, \theta^{(m)}\right]$$

  - This expectation is taken with respect to the distribution $\pi(x_{0:T} \mid y_{0:T}, \theta^{(m)})$
  - This is the smoothing distribution from the fully specified model, and is not known analytically. However, it can be sampled from and the Monte Carlo expectation used for Q (stochastic EM algorithm, Wei and Tanner 1990).

- M-step: Maximize Q and set

$$\theta^{(m+1)} = \arg\max_{\theta} Q(\theta, \theta^{(m)})$$

  - For the linear parametric model assumed here, it can be shown that this step reduces to fitting a linear regression model.

Comments

This gives a sequence $\theta^{(0)}, \theta^{(1)}, \ldots$ that tends to the maximum likelihood estimates $\text{argmax}_\theta \pi(y_{0:t}|\theta)$.

We can think of this as two steps which we loop around

1. Given an estimate for θ (and hence δ), estimate the true trajectory $x_{0:T}$ from $\pi(x_{0:T} \mid y_{0:T}, \theta)$.
2. Given samples from $\pi(x_{0:T} \mid y_{0:T}, \theta)$, estimate a value for θ.

The EM algorithm suggests that this converges to the mle (subject to problems with the expectation being approximated by a Monte Carlo sum).

Comments

This gives a sequence $\theta^{(0)}, \theta^{(1)}, \ldots$ that tends to the maximum likelihood estimates $\text{argmax}_\theta \pi(y_{0:t}|\theta)$.

We can think of this as two steps which we loop around

1. Given an estimate for θ (and hence δ), estimate the true trajectory $x_{0:T}$ from $\pi(x_{0:T} \mid y_{0:T}, \theta)$.
2. Given samples from $\pi(x_{0:T} \mid y_{0:T}, \theta)$, estimate a value for θ.

The EM algorithm suggests that this converges to the mle (subject to problems with the expectation being approximated by a Monte Carlo sum).

We require samples from the smoothing distribution $\pi(x_{0:T}|y_{0:T}, \theta)$

- We can generate approximate samples using the KF and its extensions, but this can be difficult to achieve good results
- Sequential Monte Carlo methods can be used to generate a more accurate approximation.

Filtering - $\pi(x_t|y_{0:t})$

The bootstrap filter

1. **Initialize** t=1
   For $i = 1, \ldots, N$ sample $x_1^{(i)} \sim \pi(x_1)$, set $t = 2$
2. **Importance step**
   - For $i = 1, \ldots, N$, sample

$$\tilde{x}_t^{(i)} \sim \pi(x_t|x_{t-1}^{(i)}) \qquad \sim f(x_{t-1}) + \delta(x_{t-1})$$

   - Calculate the importance weights

$$\tilde{w}^{(i)} \propto \pi(y_t|\tilde{x}_t^{(i)}) \qquad = \phi(y_t; x_t, \sigma^2_{obs})$$

3. **Selection step**
   - Sample with replacement N particles $(x_t^{(i)}, i = 1, \ldots, N)$ from $(\tilde{x}_t^{(i)}, i = 1, \ldots, N)$ according to the importance weights.
   - Set $t = t + 1$ and go to step 2. Reset all weights to be proportional to 1.

Smoothing $\pi(x_{0:T} \mid y_{0:T})$

Godsill, Doucet and West 2004

Assume we have filtered particles $\{x_t^{(i)}\}_{i=1,\ldots,N, t=1,\ldots,T}$ with $x_t^{(i)} \sim \pi(x_t|y_{0:t})$ (assume all weights are $\propto 1$ because of gratuitous resampling in the filter).

Smoothing

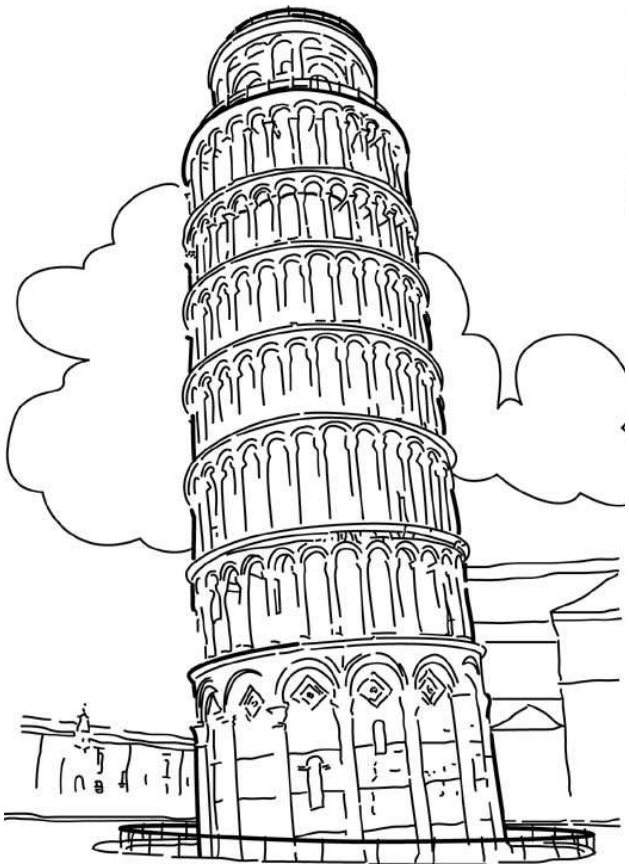
- Choose $\tilde{x}_T = x_T^{(i)}$ at random from filtered particles at time T.
- For $t = T-1$ to 1:
  - Calculate $w_{t|t+1}^{(i)} \propto \pi(\tilde{x}_{t+1}|x_t^{(i)})$ for each i
  - Choose $\tilde{x}_t = x_t^{(i)}$ with probability $w_{t|t+1}^{(i)}$

Then $\tilde{x}_{1:T}$ is an approximate realization from $\pi(x_{1:T}|y_{1:T})$.

NB The marginal smoother of Fearnhead, Wyncoll and Tawn (2008) gives all we require (i.e., pairs (x_t, x_{t+1})) and may be more efficient.

Results from freefall example

k=0.1



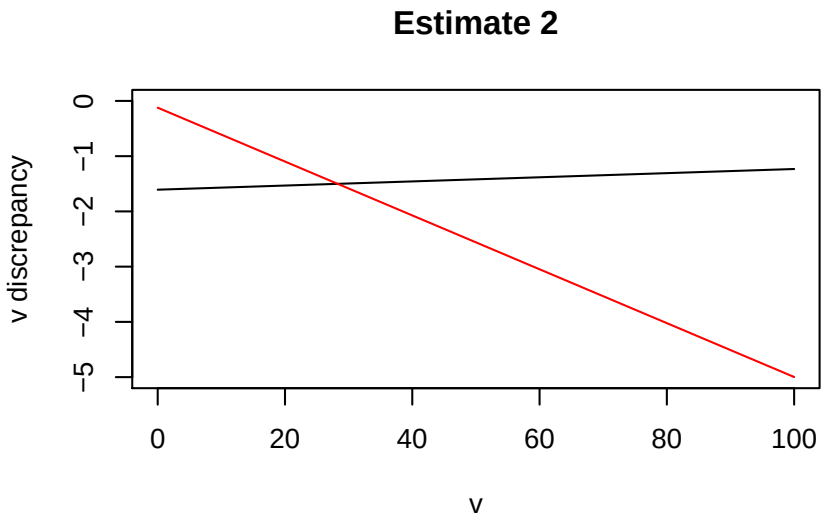
We take a sequence of 100 measurements of x, taken every 0.25 seconds.

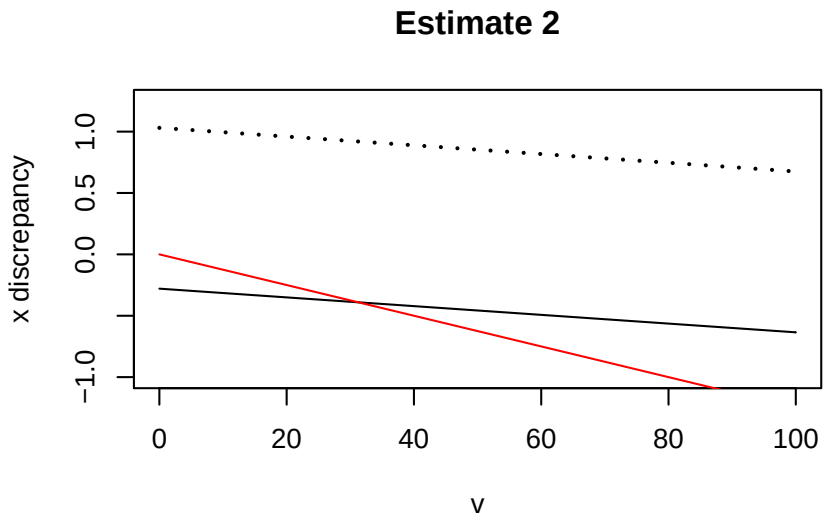
We assume the discrepancy is linear in v and x.

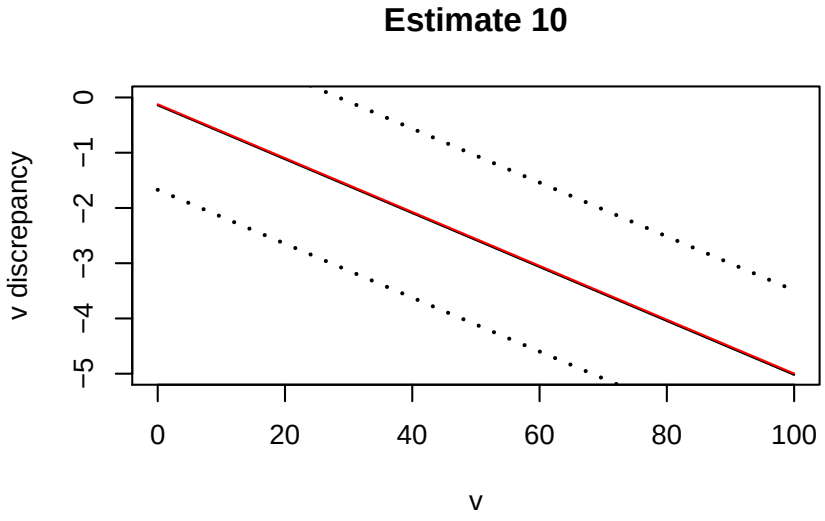
We use 1000 filtering particles and 3 smoothed trajectories giving 3×100 observations of δ.

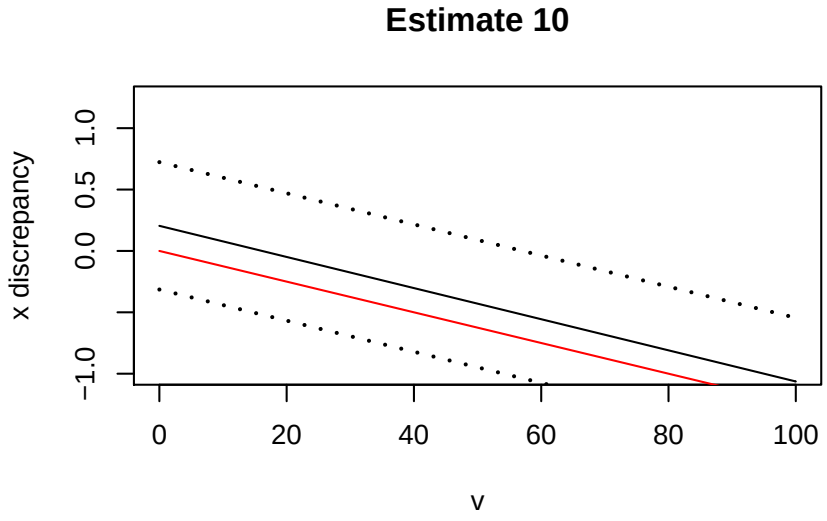
We then iterate through the EM algorithm.

Measurement error $\sigma_{obs} = 0.25m$

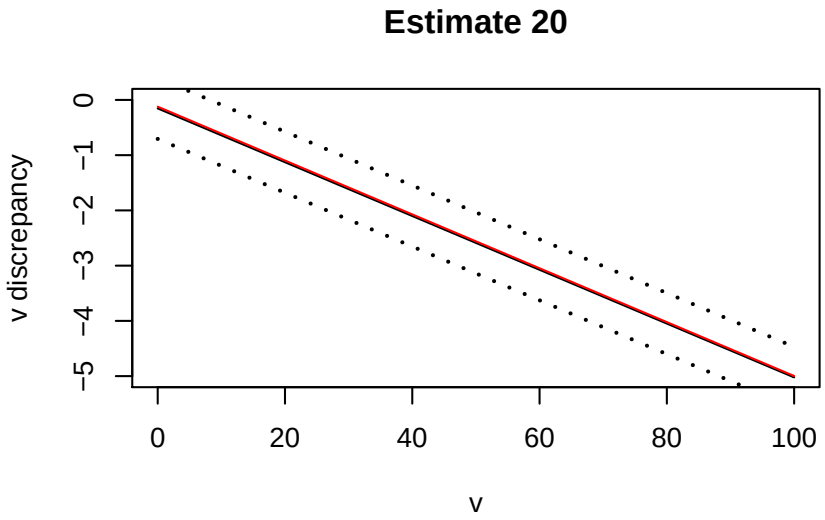


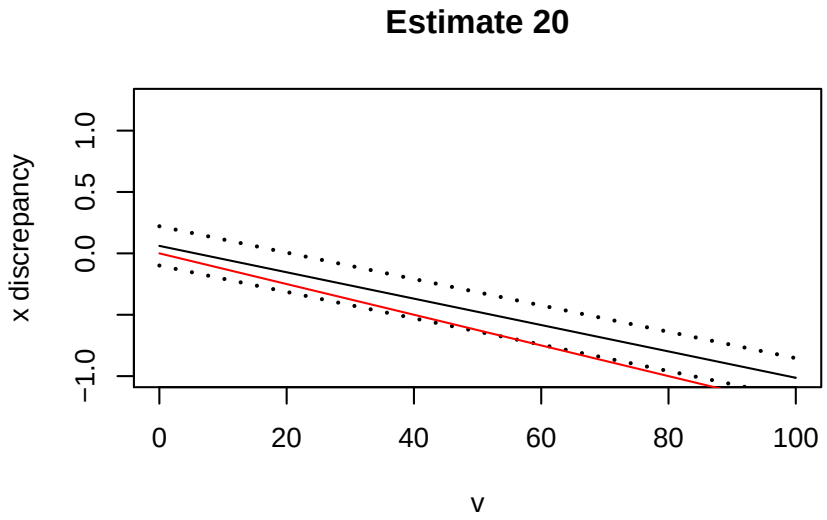


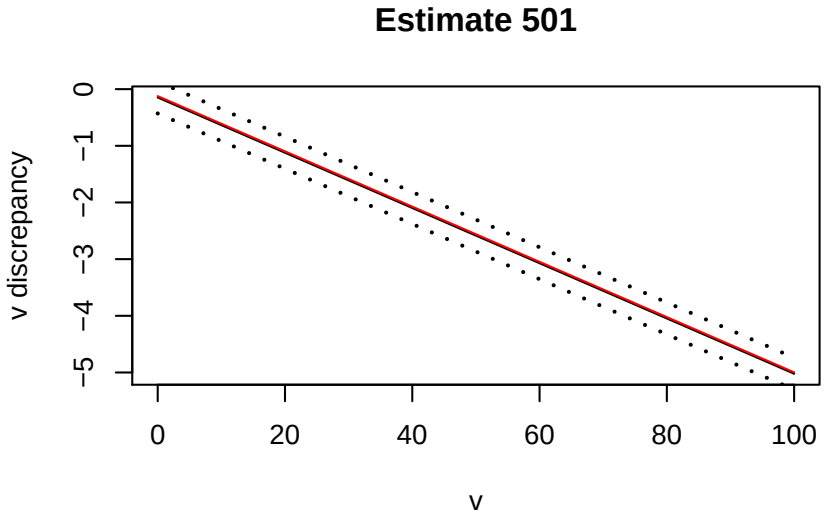


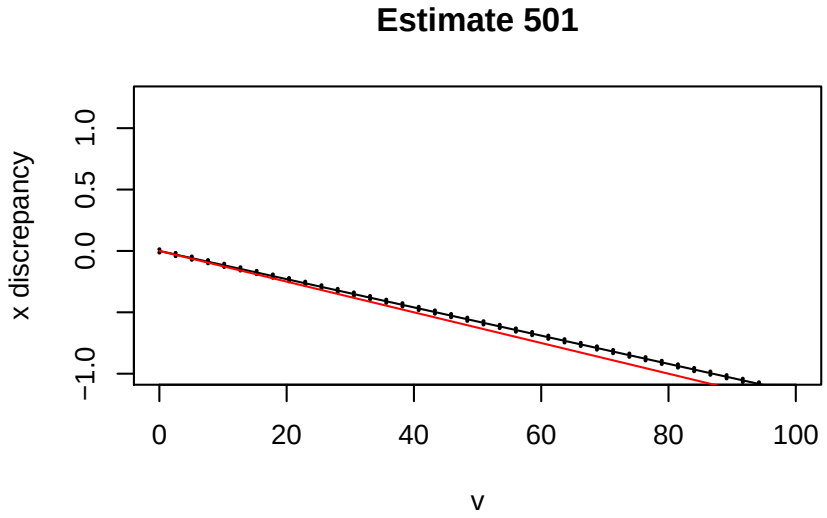


Measurement error $\sigma_{obs} = 0.25m$

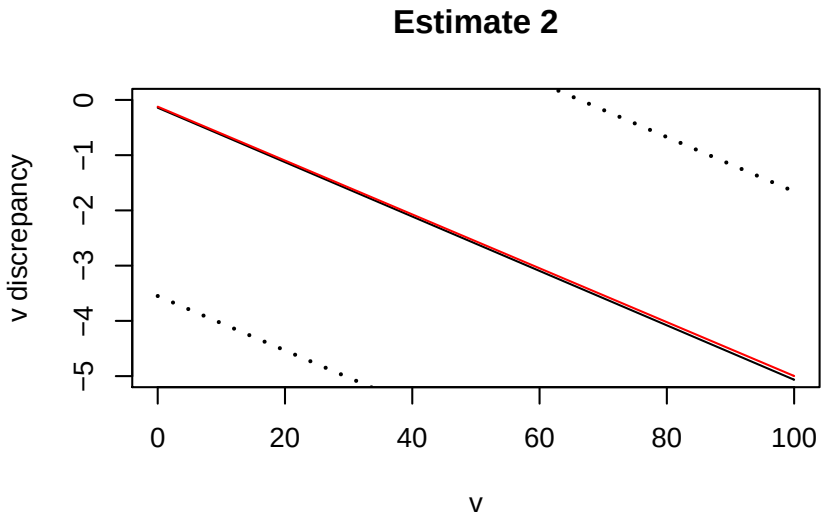


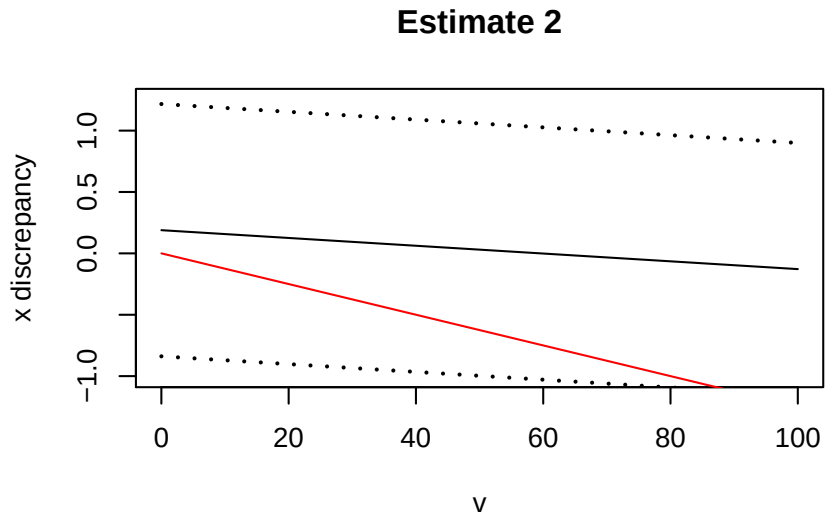


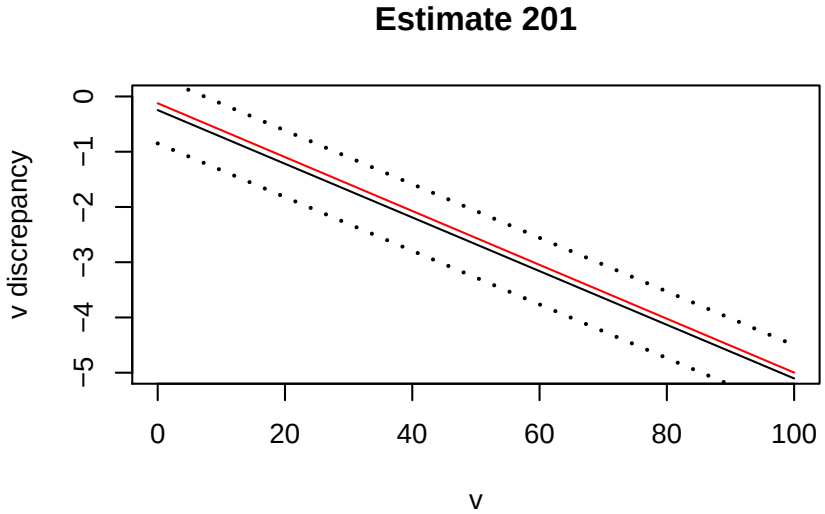


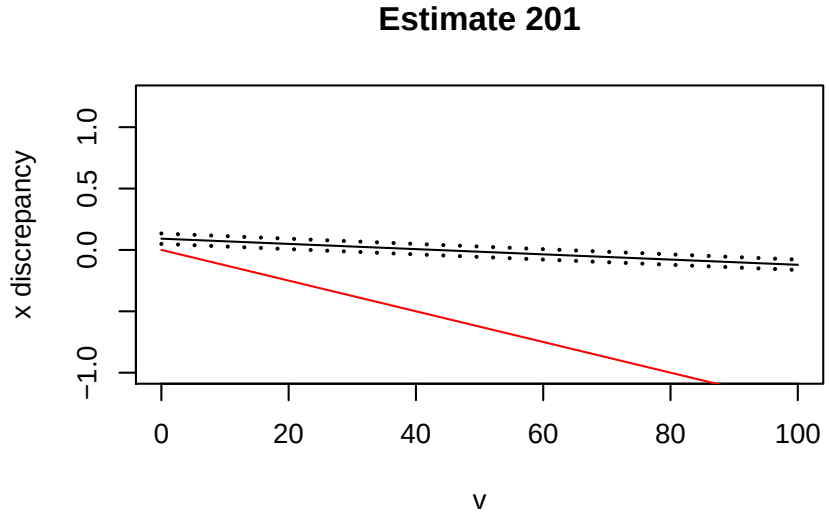


Measurement error $\sigma_{obs} = 1m$









Summary

Computer experiments are increasingly being used to learn about the world and to make decisions.

The statistical analysis of computer experiments is vitally important in many cases, yet the theory is still in its infancy.

- The incorporation of uncertainty is key if models are to be used in a decision process
- This involves the quantification of model error
- Meta-modelling is an important tool which is increasingly being used to overcome computational difficulties

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.
- D.M. Ricciuto, R. Tonkonojenkov, N. Urban, Wilkinson, D. Matthews, K.J. Davis, and K. Keller, *Assimilation of global carbon cycle observations into an Earth system model to estimate uncertain terrestrial carbon cycle parameters*, in submission.
- Sacks J, Welch WJ, Mitchell TJ and Wynn HP 1989 Design and analysis of computer experiments. *Statistical Science* **4**, 409–423.
- Santner TJ, Williams BJ and Notz W 2003 *The Design and Analysis of Computer Experiments*, Springer.
- Wilkinson, J.E. Oakley, and A. O'Hagan, *Estimating model error in data assimilation*, Technical report 09/08, Department of Probability and Statistics, University of Sheffield, 2009.
- Wilkinson, to appear 2010, Bayesian calibration of expensive multivariate computer experiments, *Large-scale inverse problems and quantification of uncertainty*, Wiley.
- Bastos L and Wilkinson 2010 *Análise Estatística de Simuladores*, available at www.maths.nottingham.ac.uk/user/pmzrdw.
- MUCM toolkit, available at `www.mucm.sheffield.ac.uk`