ABC and history matching with GPs

Richard Wilkinson, James Hensman

University of Nottingham

Gaussian Process Summer School, Sheffield 2015

Talk plan

(a) Uncertainty quantification (UQ) for computer experiments

(b) Design

(c) Calibration - history matching and ABC

Computer experiments

Rohrlich (1991): Computer simulation is

> *'a key milestone somewhat comparable to the milestone that started the empirical approach (Galileo) and the deterministic mathematical approach to dynamics (Newton and Laplace)'*

Challenges for statistics:
How do we make inferences about the world from a simulation of it?

Computer experiments

Rohrlich (1991): Computer simulation is

> *'a key milestone somewhat comparable to the milestone that started the empirical approach (Galileo) and the deterministic mathematical approach to dynamics (Newton and Laplace)'*

Challenges for statistics:
How do we make inferences about the world from a simulation of it?

- how do we relate simulators to reality?
- how do we estimate tunable parameters?
- how do we deal with computational constraints?
- how do we make uncertainty statements about the world that combine models, data and their corresponding errors?

There is an inherent a lack of quantitative information on the uncertainty surrounding a simulation - unlike in physical experiments.

Uncertainty Quantification (UQ) for computer experiments

- Calibration
  - Estimate unknown parameters θ
  - Usually via the posterior distribution $\pi(\theta|\mathcal{D})$
  - Or history matching
- Uncertainty analysis
  - $f(x)$ a complex simulator. If we are uncertain about x, e.g., $X \sim \pi(x)$, what is $\pi(f(X))$?
- Sensitivity analysis
  - $X = (X_1, \ldots, X_d)^\top$. Can we decompose $\mathbb{V}\text{ar}(f(X))$ into contributions from each $\mathbb{V}\text{ar}(X_i)$?
  - If we can improve our knowledge of any X_i, which should we choose to minimise $\mathbb{V}\text{ar}(f(X))$?
- Simulator discrepancy
  - $f(x)$ is imperfect. How can we quantify or correct simulator discrepancy.

Surrogate/Meta-modelling Emulation

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

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

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

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'

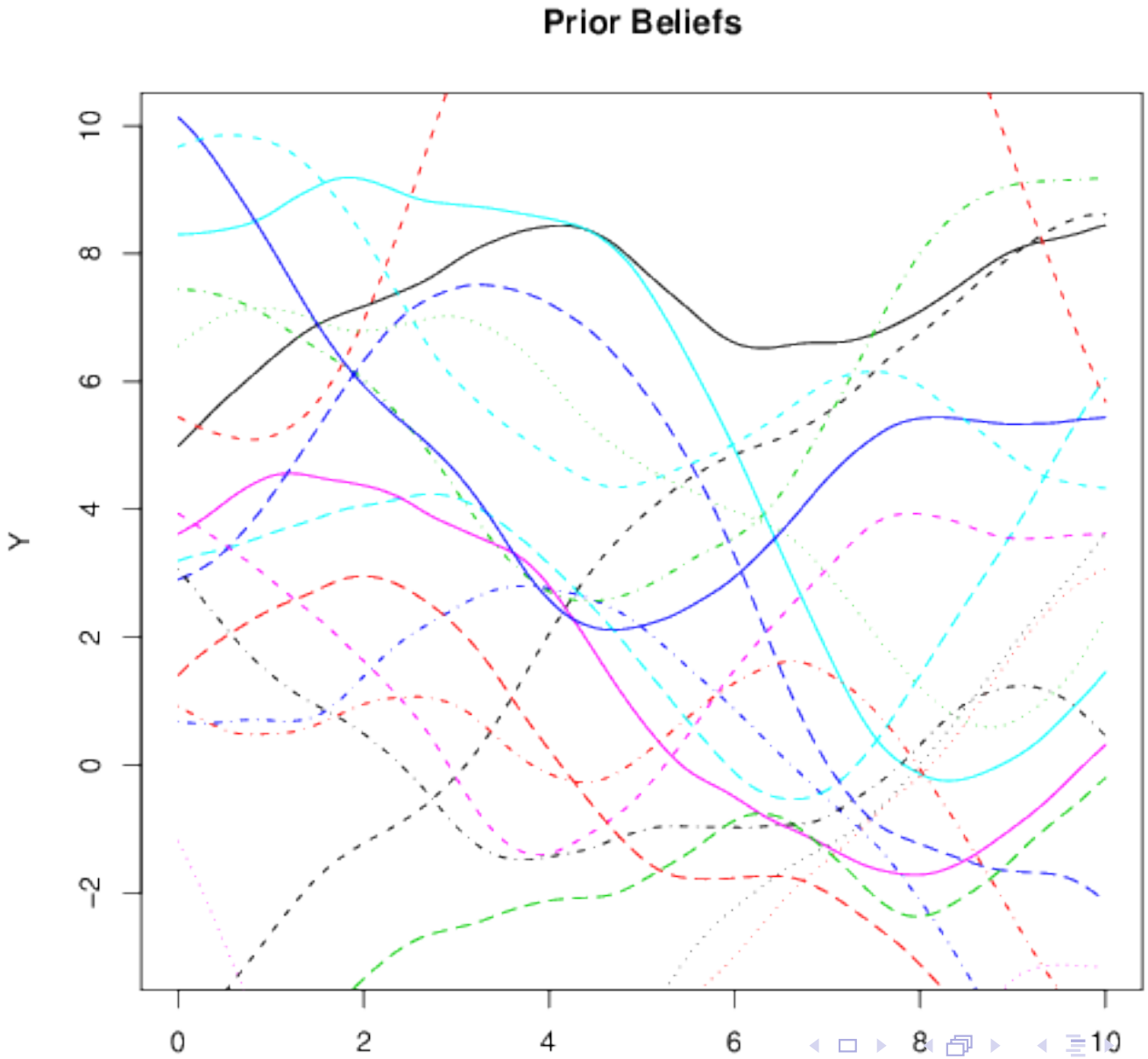
We call this meta-model an *emulator* of our simulator.

Gaussian process emulators are most popular choice for emulator.

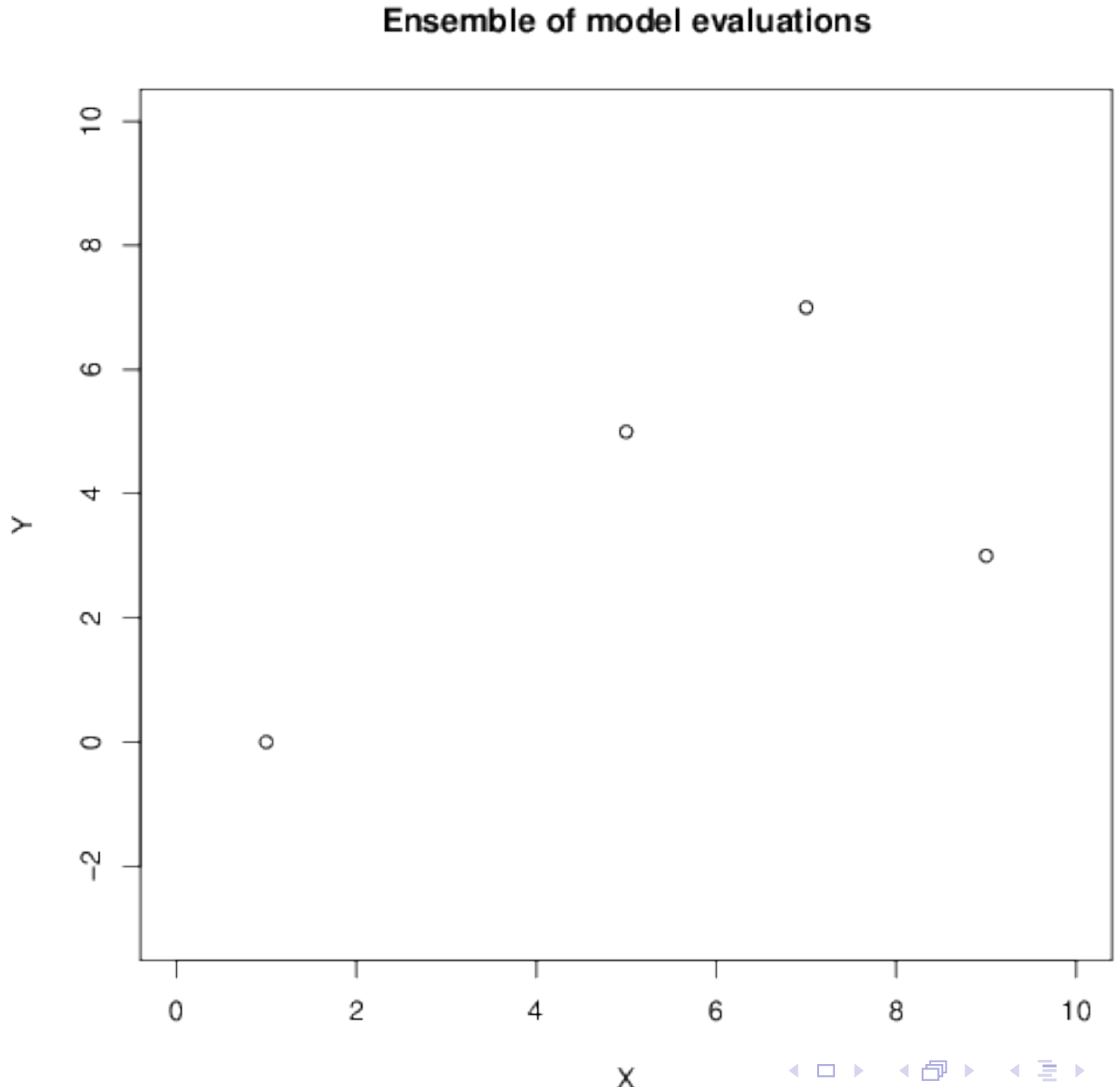
- Built using an ensemble of model runs $\mathcal{D}_{sim} = \{(\theta_i, f(\theta_i))\}_{i=1,\ldots,N}$
- They give an assessment of their prediction accuracy $\pi(f(\theta)|\mathcal{D}_{sim})$

Gaussian Process Illustration

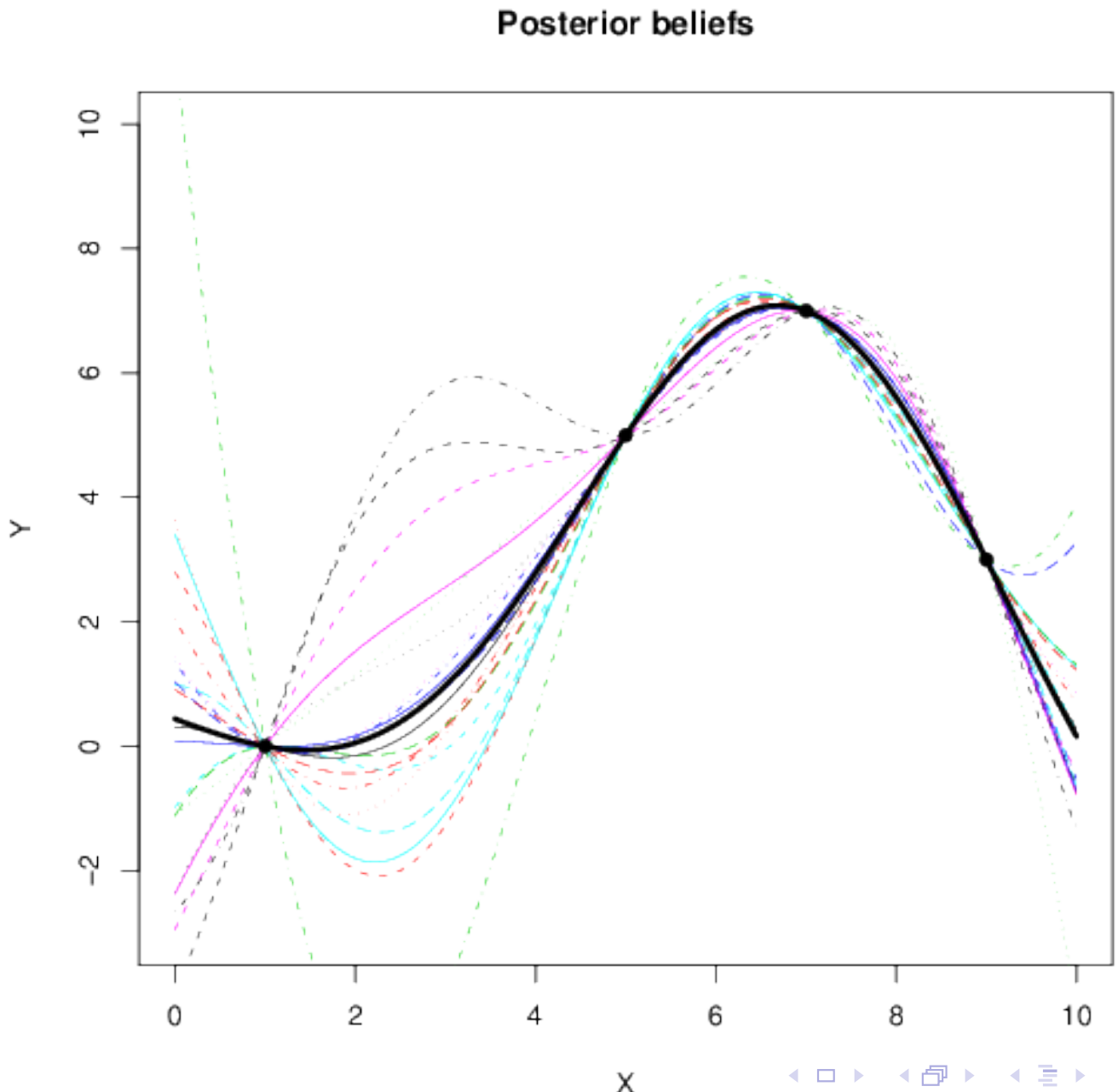
Zero mean



Gaussian Process Illustration



Gaussian Process Illustration

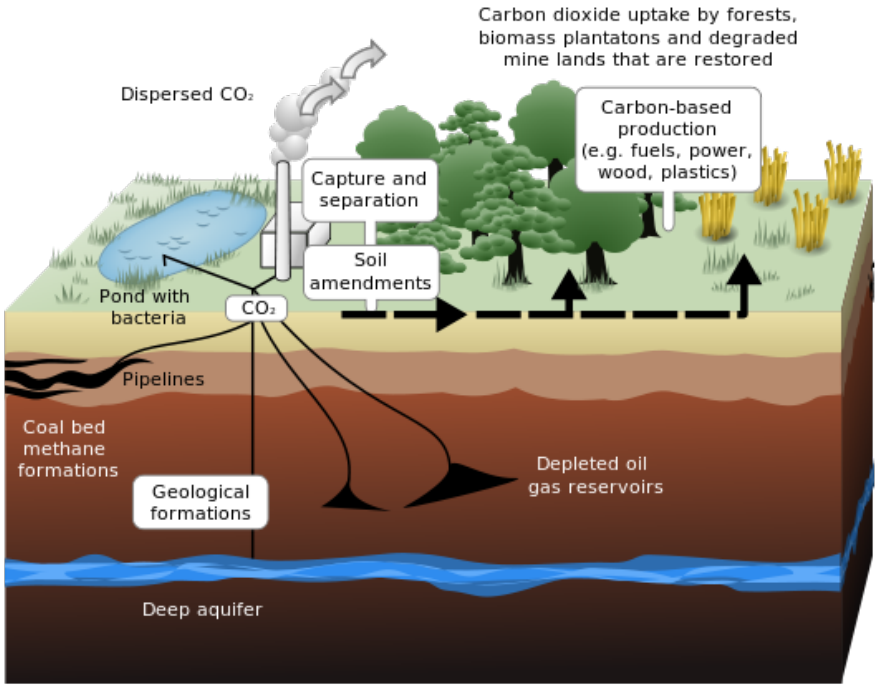


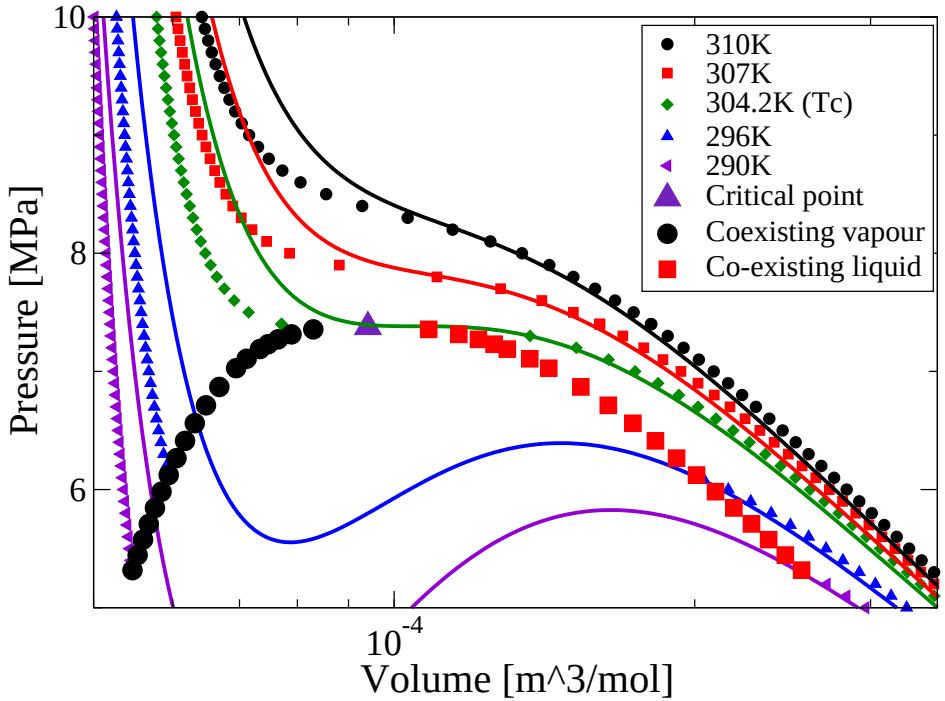
Challenges

- Design: if we can afford n simulator runs, which parameter values should we run it at?
- High dimensional inputs
  - If θ is multidimensional, then even short run times can rule out brute force approaches
- High dimensional outputs
  - Spatio-temporal.
- Incorporating physical knowledge
- Difficult behaviour, e.g., switches, step-functions, non-stationarity...

Uncertainty quantification for Carbon Capture and Storage

EPSRC: transport





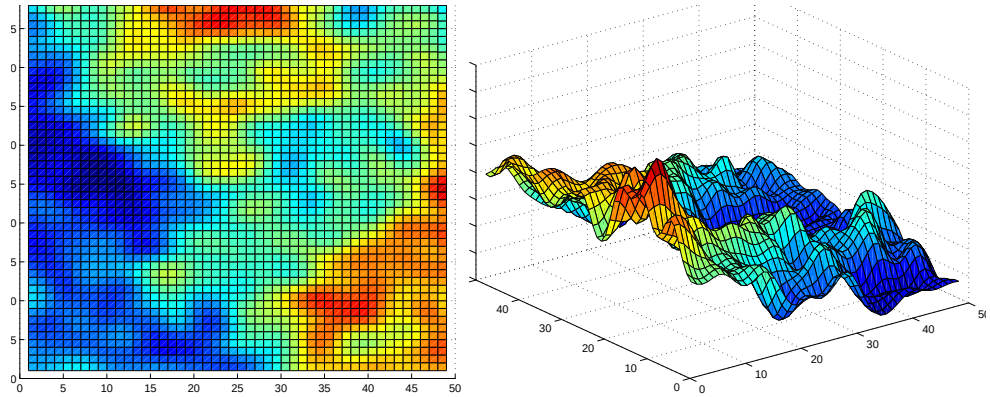
Technical challenges:

- How do we find non-parametric Gaussian process models that i) obey the fugacity constraints ii) have the correct asymptotic behaviour

Knowledge of the physical problem is encoded in a simulator f

Inputs:

Permeability field, K
(2d field)

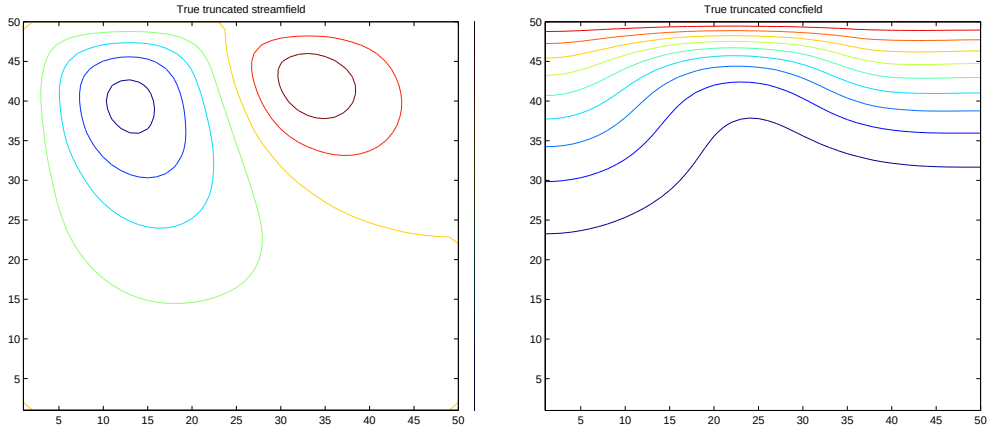


$\downarrow f(K)$

$f(K)$

Outputs:

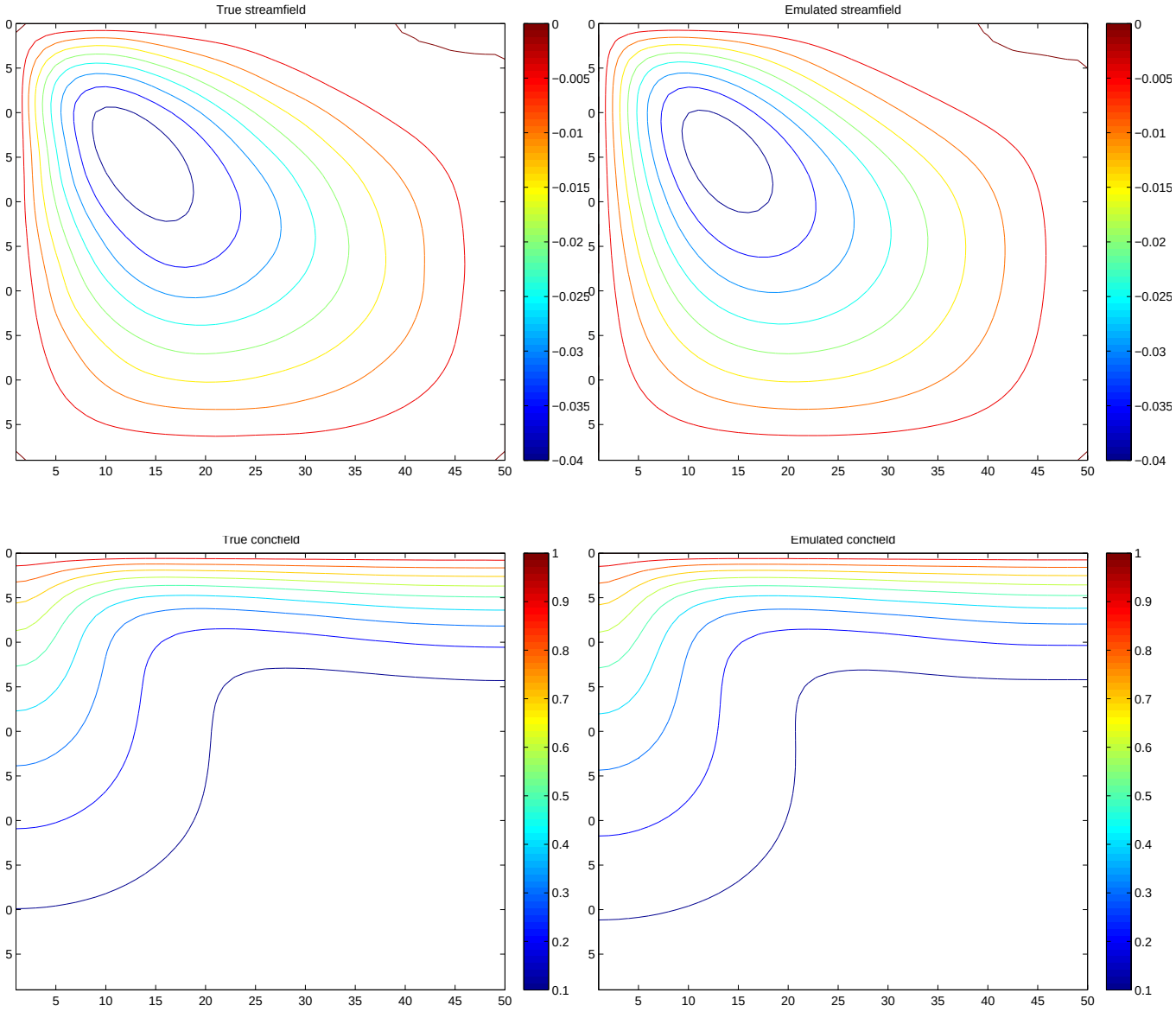
Stream func. (2d field),
concentration (2d field),
surface flux (1d scalar),
$\vdots$



Surface Flux= 6.43, ...

CCS examples

Left=true, right = emulated, 118 training runs, held out test set.



Design

Design

We build GPs using data $\{x_i, y_i\}_{i=1}^n$

- Call the collection $X_n = \{x_i\}_{i=1}^n \subset \mathbb{R}^d$ the **design**

For observational studies we have no control over the design, but we do for computer experiments!

- GP predictions made using a good design will be better than those using a poor design (Cf location of inducing points for sparse GPs)

What are we designing for?

- Global prediction
- Calibration
- Optimization - minimize the Expected Improvement (EI)?

Design for global prediction

e.g. Zhu and Stein 2006

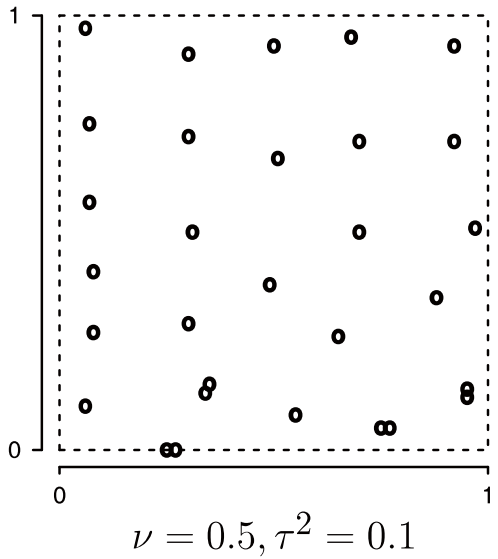
For a GP with known hyper parameters, space filling designs are good as the minimize the average prediction variance

- Latin hypercubes, maximin/minimax, max. entropy

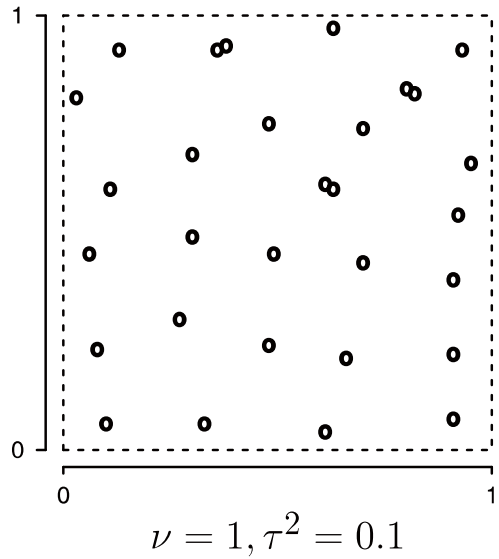
However, if we only want to estimate hyperparameters then maximize

$$\det \mathcal{I}(\theta) = -\det \mathbb{E}\left(\frac{\partial^2}{\partial \theta^2} f(X;\theta)\right)$$

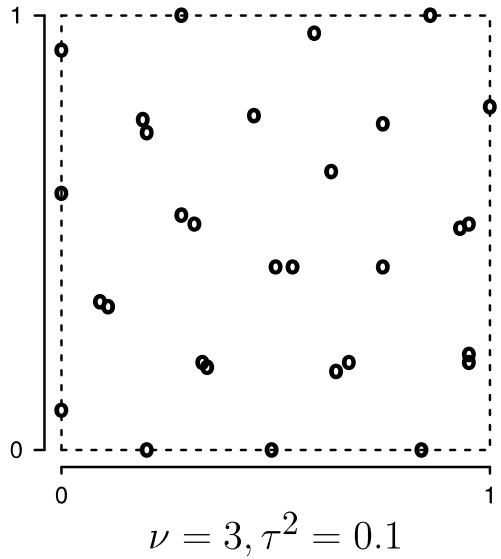
Usually, we want to make good predictions after having estimated parameters, and a trade-off between these two criteria has been proposed.



$\nu = 0.5, \tau^2 = 0.1$



$\nu = 1, \tau^2 = 0.1$



$\nu = 3, \tau^2 = 0.1$

Sequential design

The designs above are all 'one-shot' designs and can be wasteful. Instead we can use adaptive/sequential designs/active learning and add a point at a time:

- Choose location x_{n+1} to maximize some criterion/acquisition rule

$$C(x) \equiv C(x \mid \{x_i, y_i\}_{i=1}^n)$$

- Generate $y_{n+1} = f(x_{n+1})$

For optimization, we've seen that a good criterion for minimizing $f(x)$ is to choose x to maximize the expected improvement criterion

$$C(x) = \mathbb{E}[(\min_{i=1,\ldots,n} y_i - f(x))_+]$$

Sequential design for global prediction

Gramacy and Lee 2009, Beck and Guillas 2015

Many designs work on minimizing some function of the predictive variance/MSE

$$s_n^2(x) = \mathbb{V}\text{ar}(f(x)|D_n)$$

- Active learning MacKay (ALM): choose x at the point with largest predictive variance

$$C_n(x) = s_n^2(x)$$

Sequential design for global prediction

Gramacy and Lee 2009, Beck and Guillas 2015

Many designs work on minimizing some function of the predictive variance/MSE

$$s_n^2(x) = \mathbb{V}\text{ar}(f(x)|D_n)$$

- Active learning MacKay (ALM): choose x at the point with largest predictive variance

  $$C_n(x) = s_n^2(x)$$

  This tends to locate points on the edge of the domain.

Sequential design for global prediction

Gramacy and Lee 2009, Beck and Guillas 2015

Many designs work on minimizing some function of the predictive variance/MSE

$$s_n^2(x) = \mathbb{V}\text{ar}(f(x)|D_n)$$

- Active learning MacKay (ALM): choose x at the point with largest predictive variance

  $$C_n(x) = s_n^2(x)$$

  This tends to locate points on the edge of the domain.
- Active learning Cohn (ALC): choose x to give largest expected reduction in predictive variance

$$C_n(x) = \int s_n^2(x') - s_{n\cup x}^2(x')\mathrm{d}x'$$

Sequential design for global prediction

Gramacy and Lee 2009, Beck and Guillas 2015

Many designs work on minimizing some function of the predictive variance/MSE

$$s_n^2(x) = \mathbb{V}\text{ar}(f(x)|D_n)$$

- Active learning MacKay (ALM): choose x at the point with largest predictive variance

  $$C_n(x) = s_n^2(x)$$

  This tends to locate points on the edge of the domain.
- Active learning Cohn (ALC): choose x to give largest expected reduction in predictive variance

$$C_n(x) = \int s_n^2(x') - s_{n\cup x}^2(x')\mathrm{d}x'$$

ALC tends to give better designs than ALM, but has cost $O(n^3 + N_{ref}\,N_{cand}\,n^2)$ for each new design point

Sequential design for global prediction

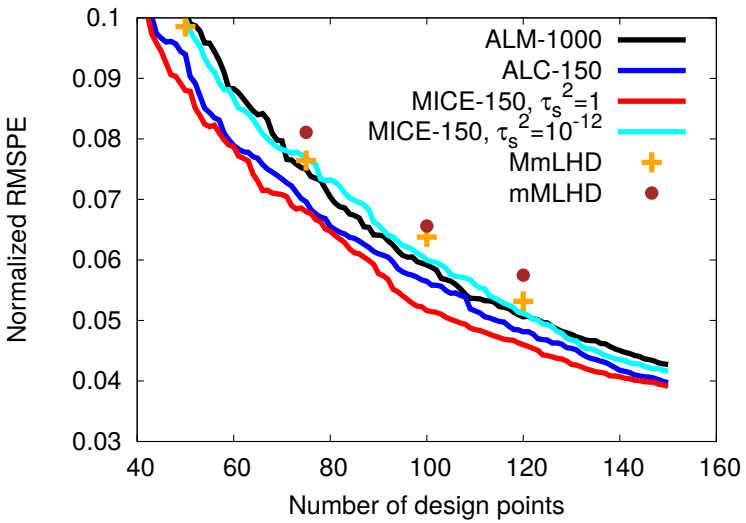
MICE: Beck and Guillas 2015

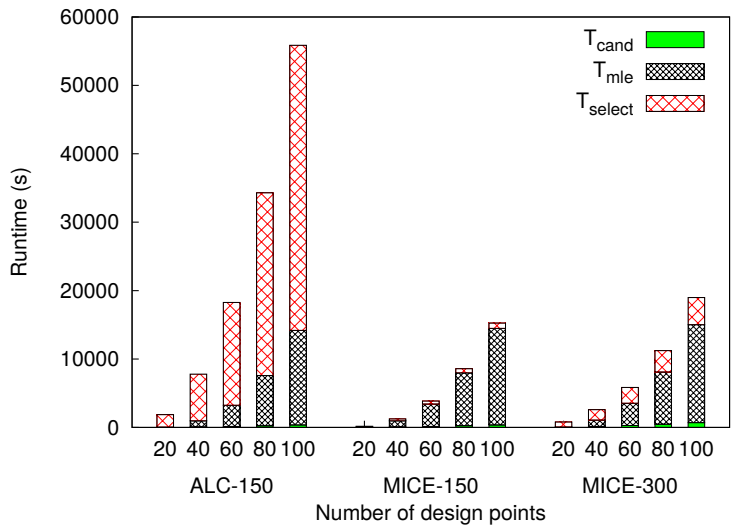
The Mutual Information between Y and Y' is

$$\mathcal{I}(Y; Y') = \mathcal{H}(Y) - \mathcal{H}(Y \mid Y') = KL(p_{y,y'} || p_y p_{y'})$$

Choose design X_n to maximize mutual information between $f(X_n)$ and $f(X_{cand} \setminus X_n)$ where X_{cand} is a set of candidate design points.
A sequential version for GPs reduces to choosing x to maximize

$$C_n(x) = \frac{s_n^2(x)}{s_{cand \setminus (n \cup x)}(x, \tau^2)}$$

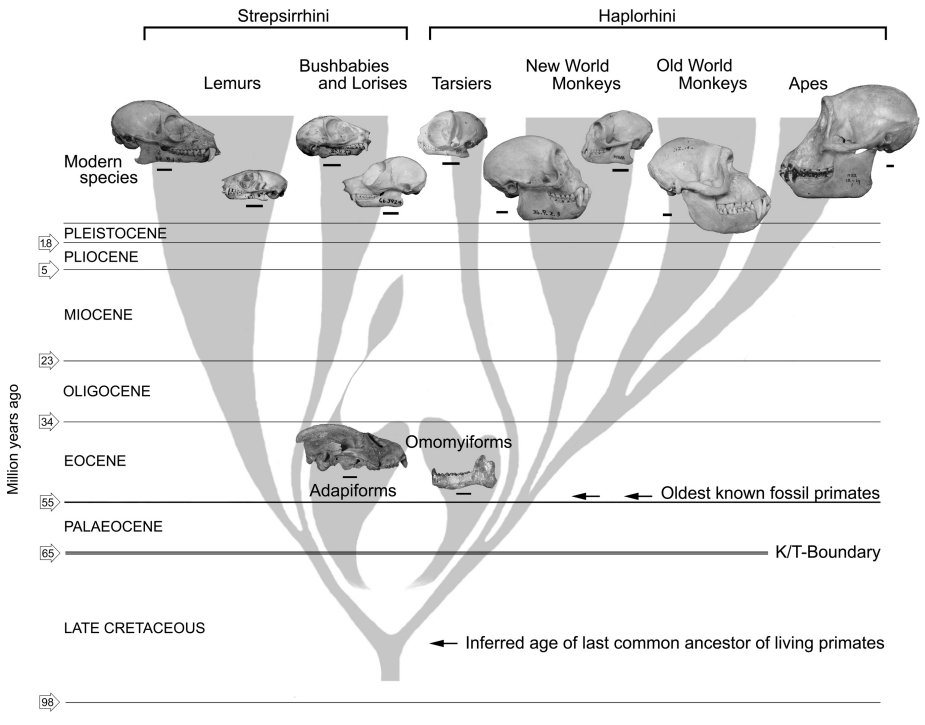


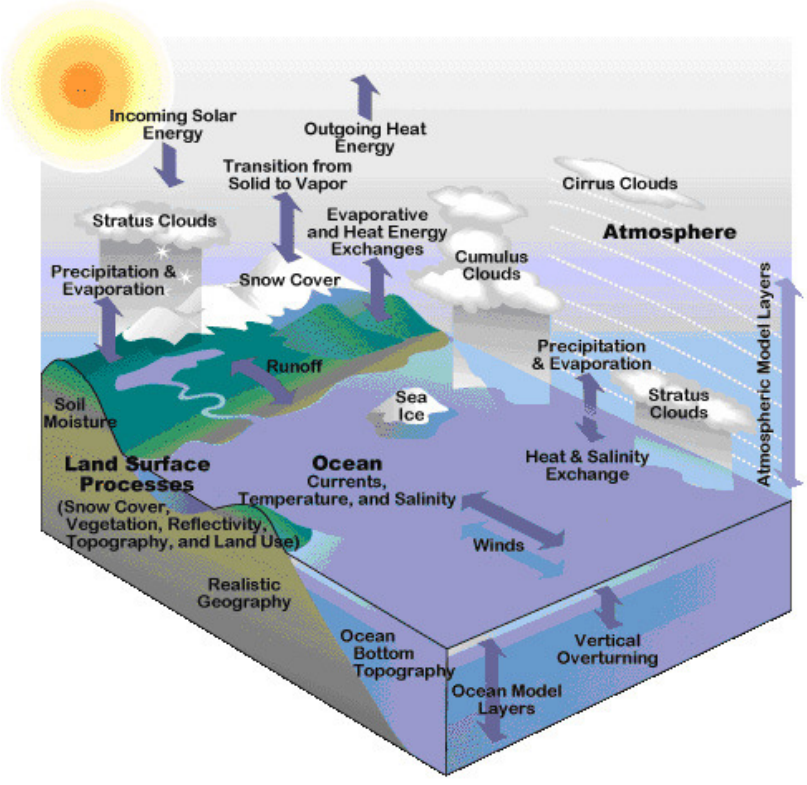


Calibration, history matching and ABC

Inverse problems

- For most simulators we specify parameters θ and i.c.s and the simulator, $f(\theta)$, generates output X.
- The inverse-problem: observe data D, estimate parameter values θ





Two approaches

Probabilistic calibration

Find the posterior distribution

$$\pi(\theta|\mathcal{D}) \propto \pi(\theta)\pi(\mathcal{D}|\theta)$$

for likelihood function
$\pi(\mathcal{D}|\theta) = \int \pi(D|X,\theta)\pi(X|\theta)\mathrm{d}X$
which relates the simulator output, to the data,e.g.,

$$D = X + e + \epsilon$$

where $e \sim N(0, \sigma_e^2)$ represents simulator discrepancy, and $\epsilon \sim N(0, \sigma_\epsilon^2)$ represents measurement error on the data

History matching

Find the plausible parameter set $\mathcal{P}_\theta$:

$$f(\theta) \in \mathcal{P}_D \; \forall \; \theta \in \mathcal{P}_\theta$$

where $\mathcal{P}_D$ is some plausible set of simulation outcomes that are consistent with simulator discrepancy and measurement error, e.g.,

$$\mathcal{P}_D = \{X : |D - X| \leq 3(\sigma_e + \sigma_\epsilon)\}$$

Calibration aims to find a distribution representing plausible parameter values, whereas history matching classifies space as plausible or implausible.

Calibration - Approximate Bayesian Computation (ABC)

ABC algorithms are a collection of Monte Carlo methods used for calibrating simulators

- they do not require explicit knowledge of the likelihood function
- inference is done using simulation from the model (they are 'likelihood-free').

ABC methods are popular in biological disciplines, particularly genetics. They are

- Simple to implement
- Intuitive
- Embarrassingly parallelizable
- Can usually be applied

Rejection ABC

Uniform Rejection Algorithm

- Draw θ from $\pi(\theta)$
- Simulate $X \sim f(\theta)$
- Accept θ if $\rho(D, X) \leq \epsilon$

Rejection ABC

Uniform Rejection Algorithm

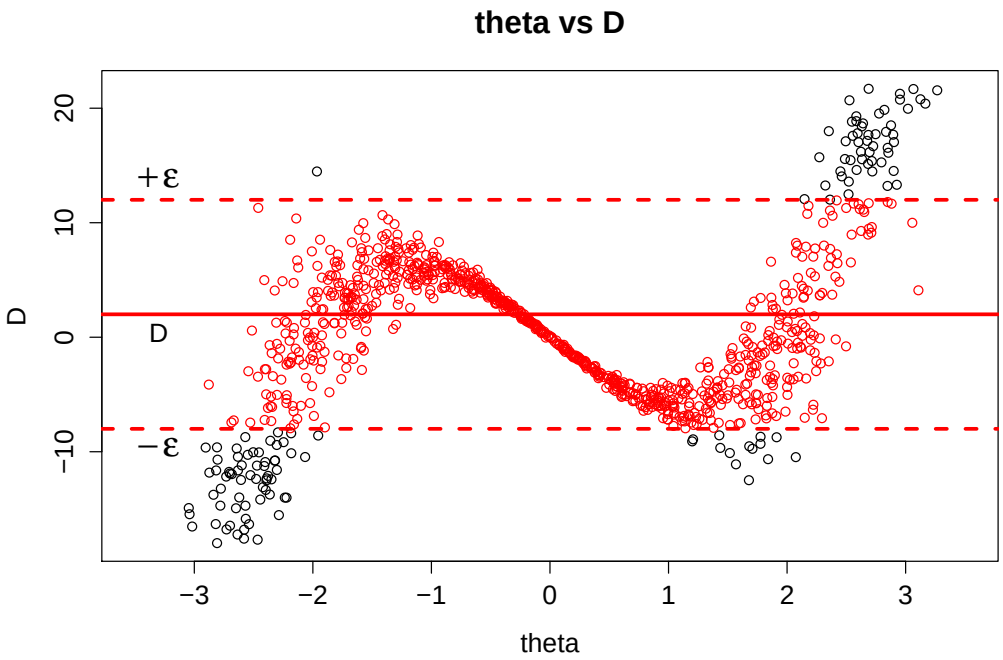
- Draw θ from $\pi(\theta)$
- Simulate $X \sim f(\theta)$
- Accept θ if $\rho(D, X) \leq \epsilon$

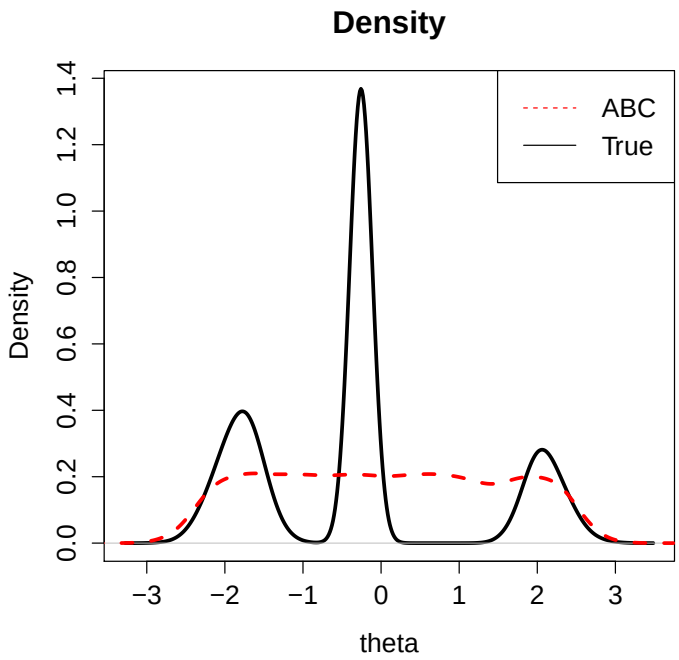
ϵ reflects the tension between computability and accuracy.

- As $\epsilon \to \infty$, we get observations from the prior, $\pi(\theta)$.
- If $\epsilon = 0$, we generate observations from $\pi(\theta \mid D)$.

Rejection sampling is inefficient, but we can adapt other MC samplers such as MCMC and SMC.

$\epsilon = 10$

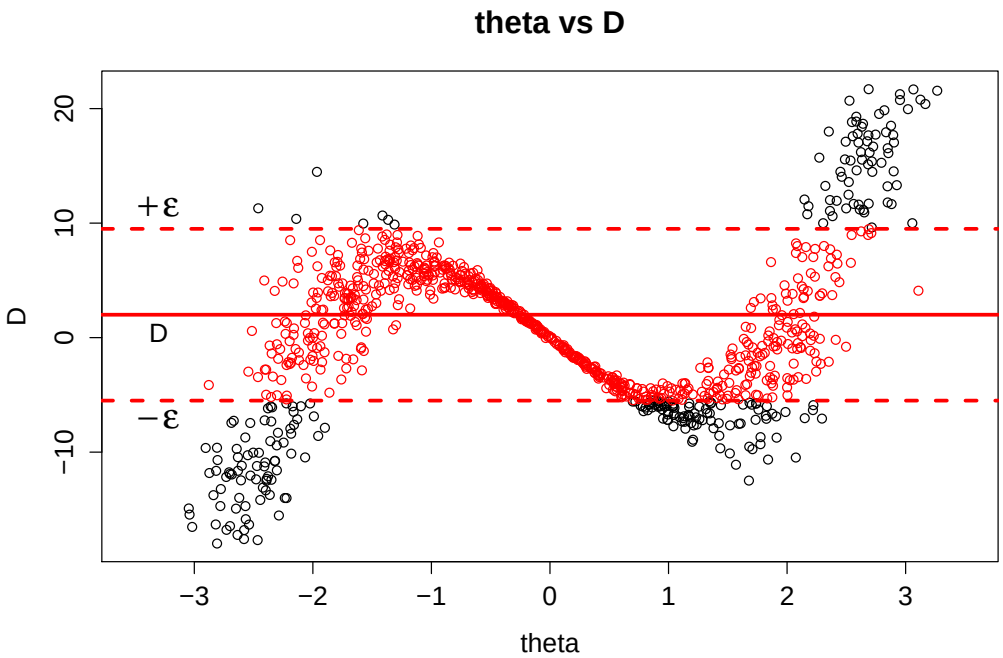


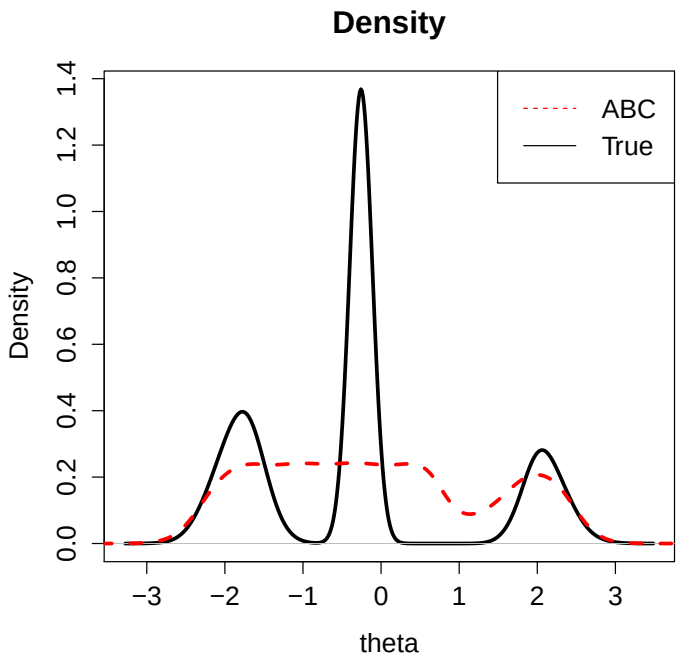


$$\theta \sim U[-10, 10], \qquad X \sim N(2(\theta + 2)\theta(\theta - 2), 0.1 + \theta^2)$$

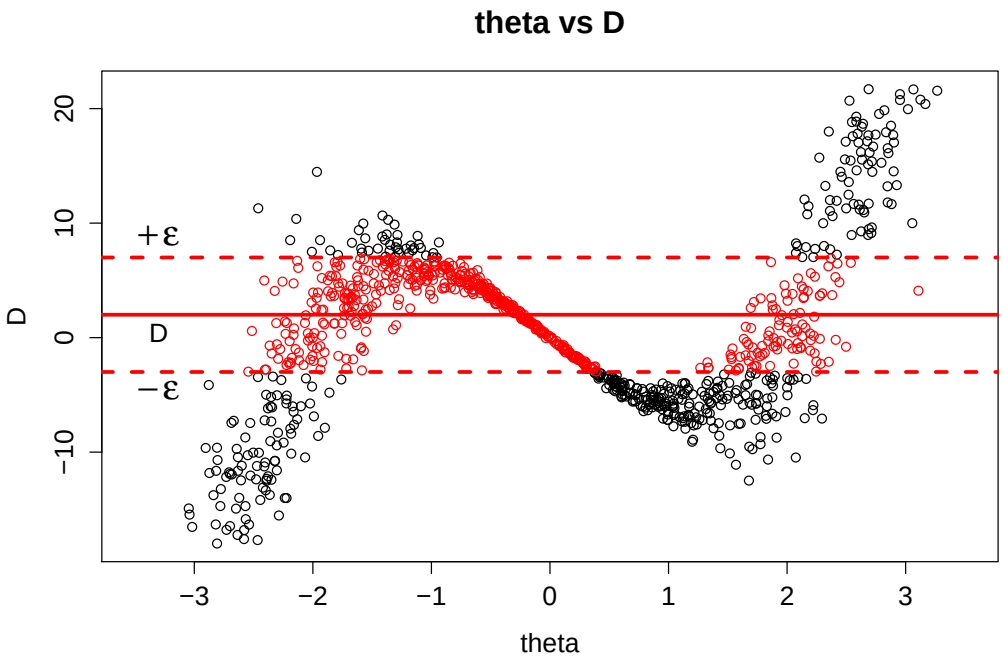
$$\rho(D, X) = |D - X|, \qquad D = 2$$

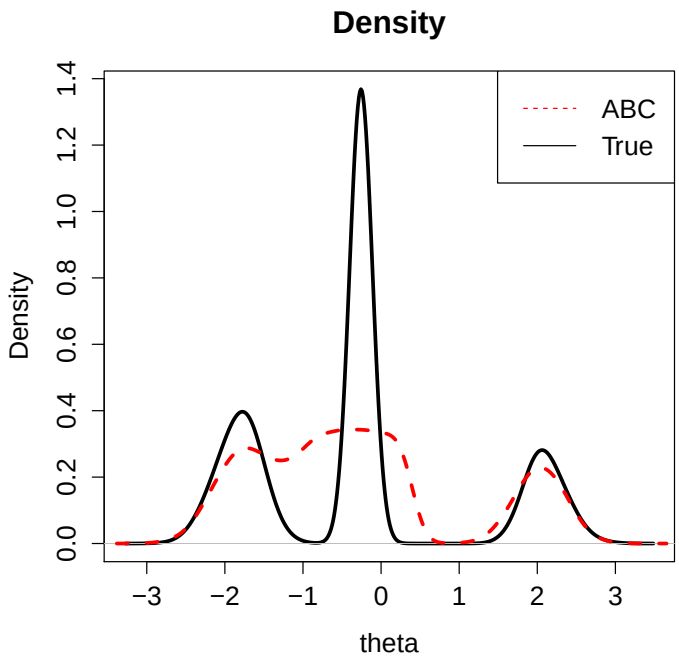
$\epsilon = 7.5$


theta vs D
D
theta
+ε
D
−ε
20
10
0
−10
−3
−2
−1
0
1
2
3

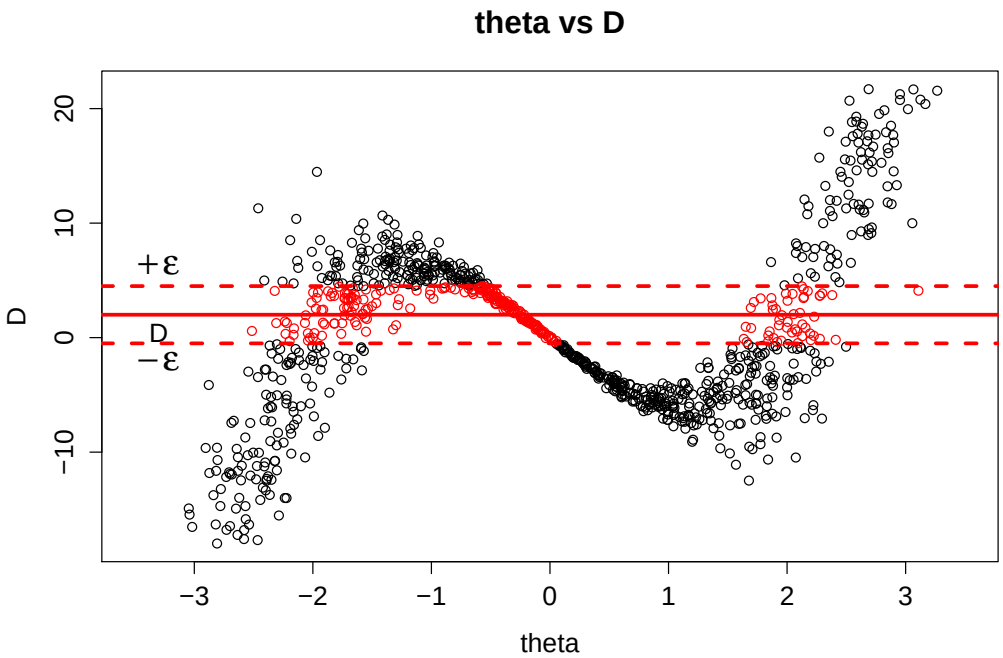

Density
Density
theta
ABC
True
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−3
−2
−1
0
1
2
3

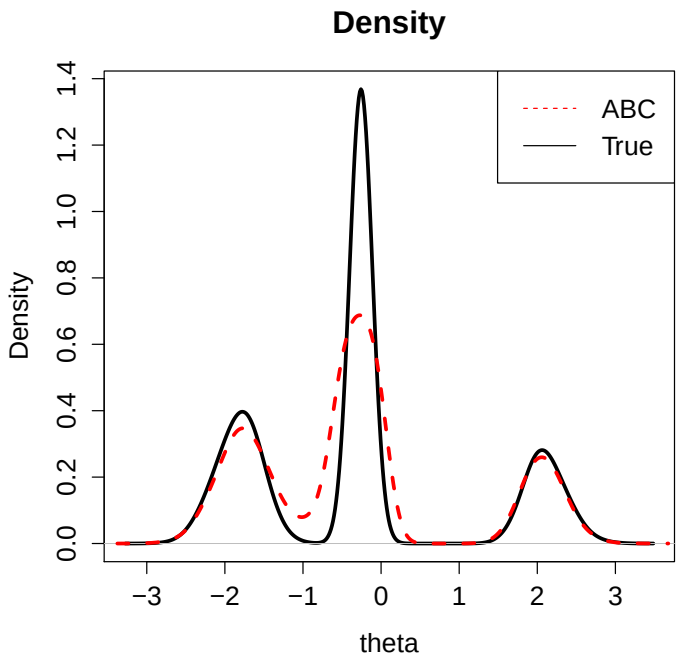
$\epsilon = 5$


theta vs D
D
theta
+ε
D
−ε
20
10
0
−10
−3
−2
−1
0
1
2
3

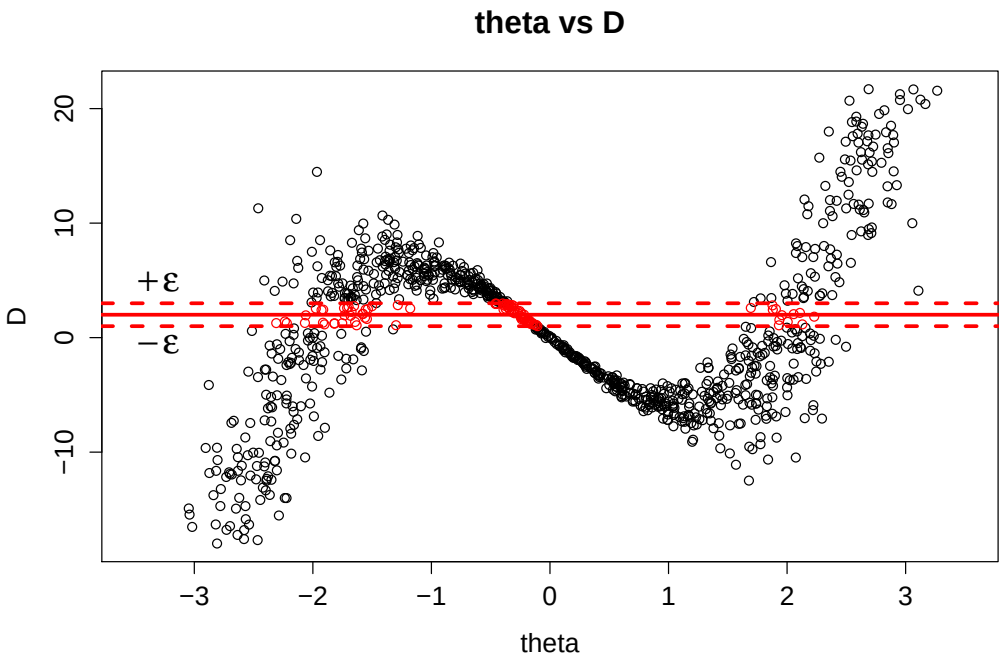

Density
Density
theta
ABC
True
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−3
−2
−1
0
1
2
3

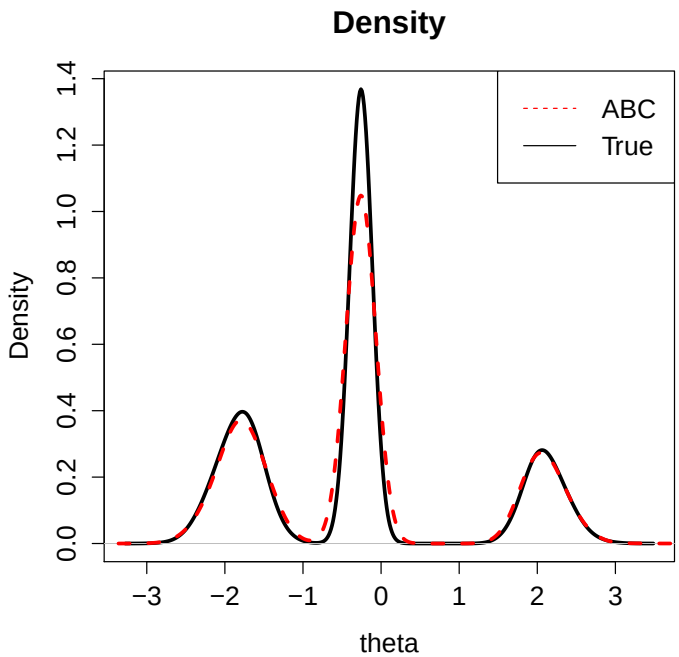
$\epsilon = 2.5$


theta vs D
D
theta
+ε
D
−ε
20
10
0
−10
−3
−2
−1
0
1
2
3


Density
Density
theta
ABC
True
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−3
−2
−1
0
1
2
3

$\epsilon = 1$


theta vs D
D
theta
+ε
−ε
20
10
0
−10
−3
−2
−1
0
1
2
3


Density
Density
theta
ABC
True
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−3
−2
−1
0
1
2
3

Limitations of Monte Carlo methods

Monte Carlo methods are generally guaranteed to succeed if we run them for long enough.

This guarantee is costly and can require more simulation than is possible.

Limitations of Monte Carlo methods

Monte Carlo methods are generally guaranteed to succeed if we run them for long enough.

This guarantee is costly and can require more simulation than is possible.

However,

- Most methods sample naively - they don't learn from previous simulations.
- They don't exploit known properties of the likelihood function, such as continuity
- They sample randomly, rather than using careful design.

We can use methods that don't suffer in this way, but at the cost of losing the guarantee of success.

Likelihood estimation

Wilkinson 2013

It can be shown that ABC replaces the true likelihood $\pi(D|\theta)$ by an ABC likelihood

$$\pi_{ABC}(D|\theta) = \int \pi(D|X)\pi(X|\theta)\mathrm{dX}$$

where $\pi(D|X)$ is the ABC acceptance kernel.

Likelihood estimation

Wilkinson 2013

It can be shown that ABC replaces the true likelihood $\pi(D|\theta)$ by an ABC likelihood

$$\pi_{ABC}(D|\theta) = \int \pi(D|X)\pi(X|\theta)\mathrm{dX}$$

where $\pi(D|X)$ is the ABC acceptance kernel.

We can estimate this using repeated runs from the simulator

$$\hat{\pi}_{ABC}(D|\theta) \approx \frac{1}{N}\sum \pi(D|X_i)$$

where $X_i \sim \pi(X|\theta)$.

Likelihood estimation

Wilkinson 2013

It can be shown that ABC replaces the true likelihood $\pi(D|\theta)$ by an ABC likelihood

$$\pi_{ABC}(D|\theta) = \int \pi(D|X)\pi(X|\theta)\mathrm{dX}$$

where $\pi(D|X)$ is the ABC acceptance kernel.

We can estimate this using repeated runs from the simulator

$$\hat{\pi}_{ABC}(D|\theta) \approx \frac{1}{N}\sum \pi(D|X_i)$$

where $X_i \sim \pi(X|\theta)$.

For many problems, we believe the likelihood is continuous and smooth, so that $\pi_{ABC}(D|\theta)$ is similar to $\pi_{ABC}(D|\theta')$ when $\theta - \theta'$ is small

We can model $L(\theta) = \pi_{ABC}(D|\theta)$ and use the model to find the posterior in place of running the simulator.

History matching waves

Wilkinson 2014

The likelihood is too difficult to model, so we model the log-likelihood instead.

$$l(\theta) = \log L(\theta)$$

History matching waves

Wilkinson 2014

The likelihood is too difficult to model, so we model the log-likelihood instead.

$$l(\theta) = \log L(\theta)$$

However, the log-likelihood for a typical problem ranges across too wide a range of values.

Consequently, most GP models will struggle to model the log-likelihood across the parameter space.

History matching waves

Wilkinson 2014

The likelihood is too difficult to model, so we model the log-likelihood instead.

$$l(\theta) = \log L(\theta)$$

However, the log-likelihood for a typical problem ranges across too wide a range of values.

Consequently, most GP models will struggle to model the log-likelihood across the parameter space.

- Introduce waves of history matching.
- In each wave, build a GP model that can rule out regions of space as implausible.

History matching waves

Wilkinson 2014

The likelihood is too difficult to model, so we model the log-likelihood instead.

$$l(\theta) = \log L(\theta)$$

However, the log-likelihood for a typical problem ranges across too wide a range of values.

Consequently, most GP models will struggle to model the log-likelihood across the parameter space.

- Introduce waves of history matching.
- In each wave, build a GP model that can rule out regions of space as implausible.

When this works, it can give huge savings in the number of simulator runs required.

Example: Ricker Model

The Ricker model is one of the prototypic ecological models.

- used to model the fluctuation of the observed number of animals in some population over time
- It has complex dynamics and likelihood, despite its simple mathematical form.

Ricker Model

- Let N_t denote the number of animals at time t.

$$N_{t+1} = rN_t e^{-N_t + e_r}$$

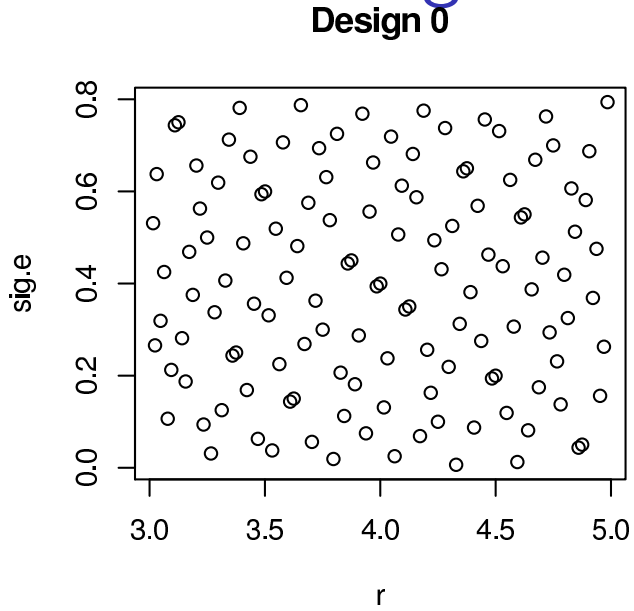
where e_t are independent $N(0, \sigma_e^2)$ process noise

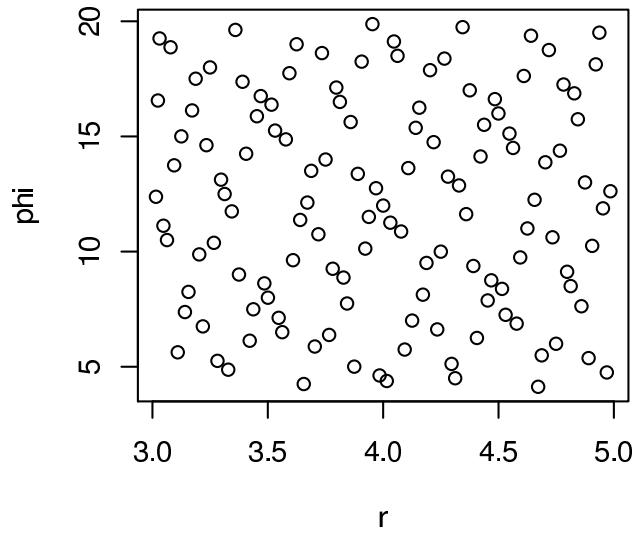
- Assume we observe counts y_t where

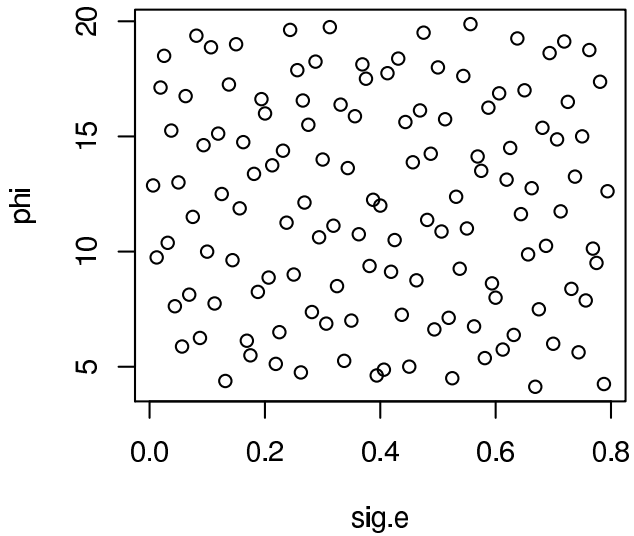
$$y_t \sim Po(\phi N_t)$$

Used in Wood to demonstrate the synthetic likelihood approach.

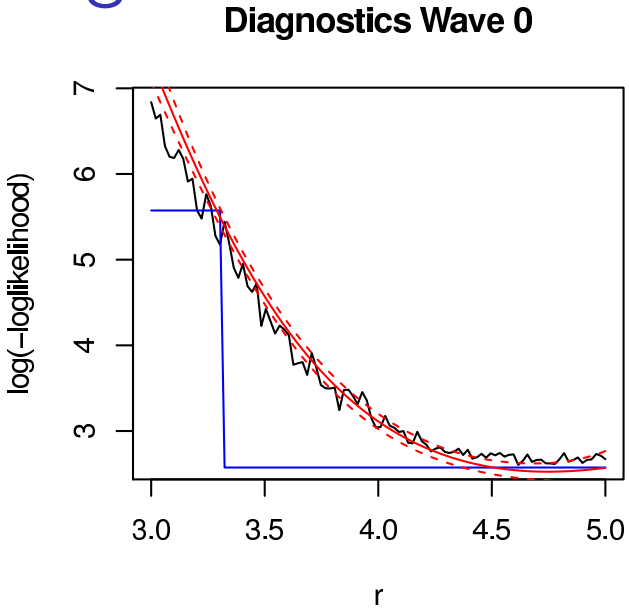
Results - Design 1 - 128 pts

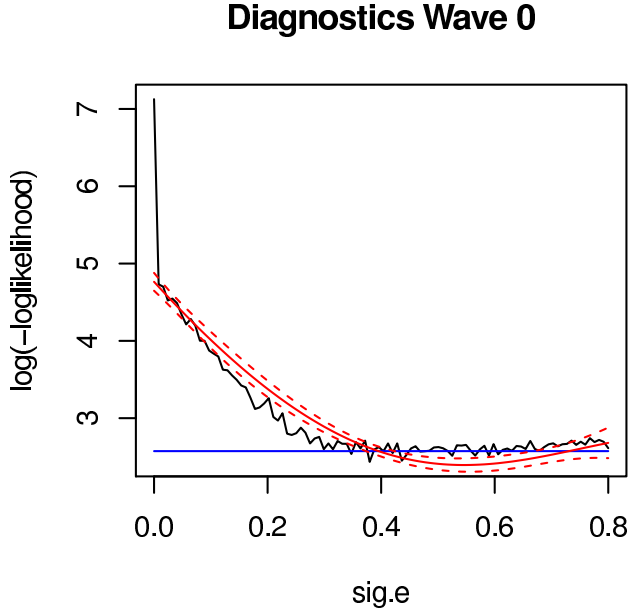


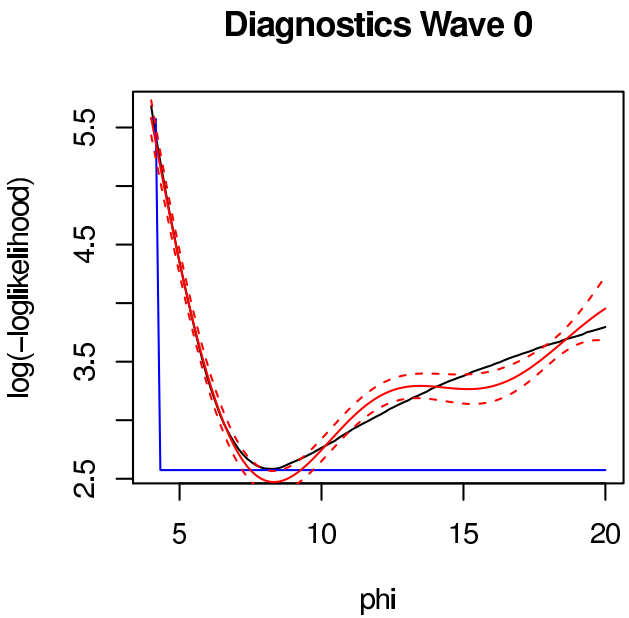




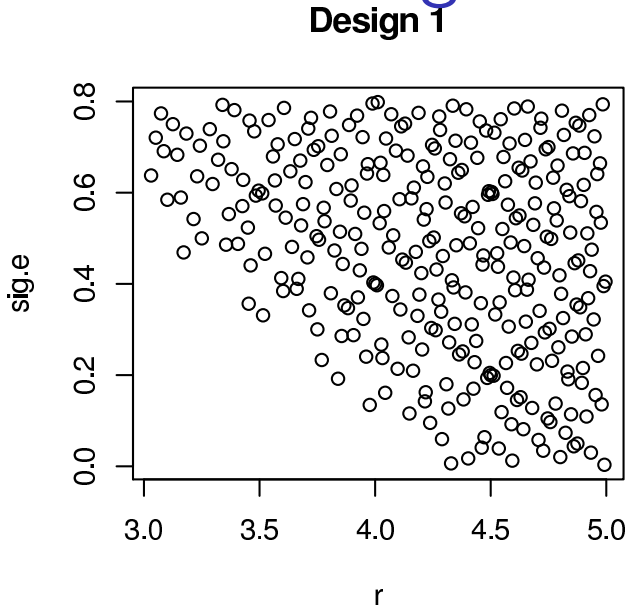
Diagnostics for GP 1 - threshold = 5.6

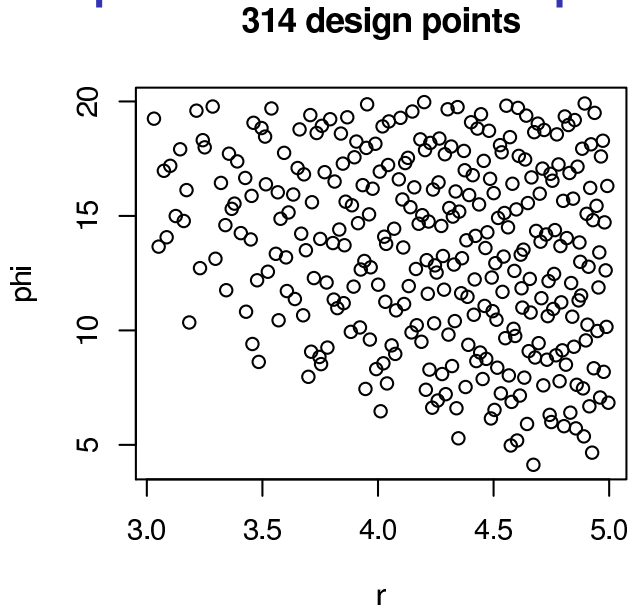


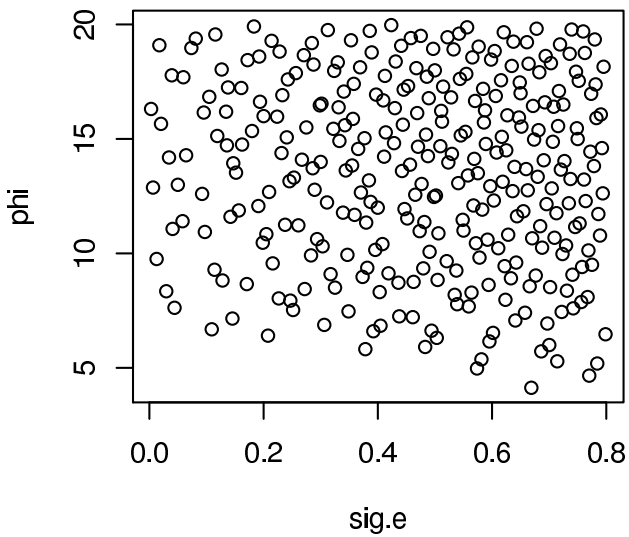




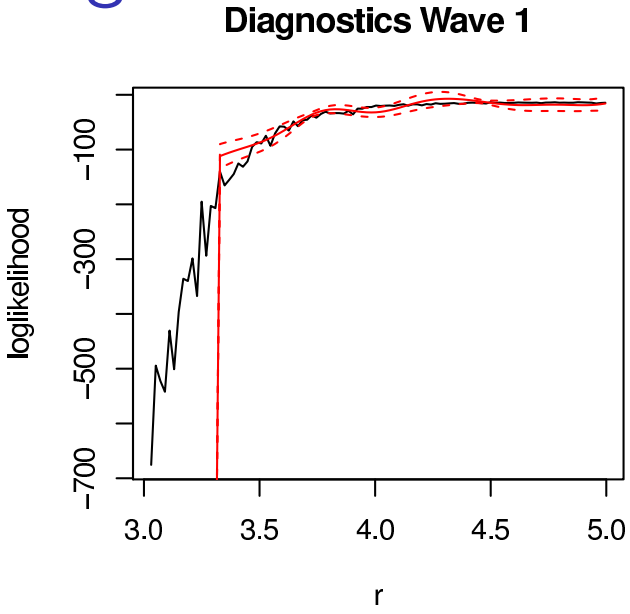
Results - Design 2 - 314 pts - 38% of space implausible

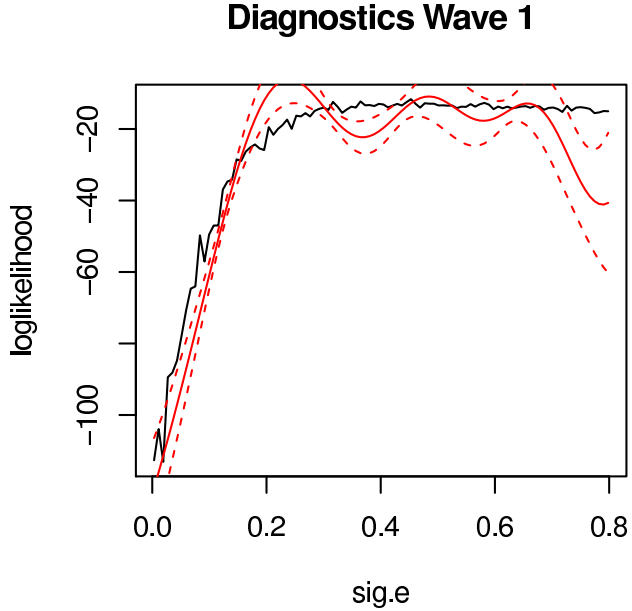


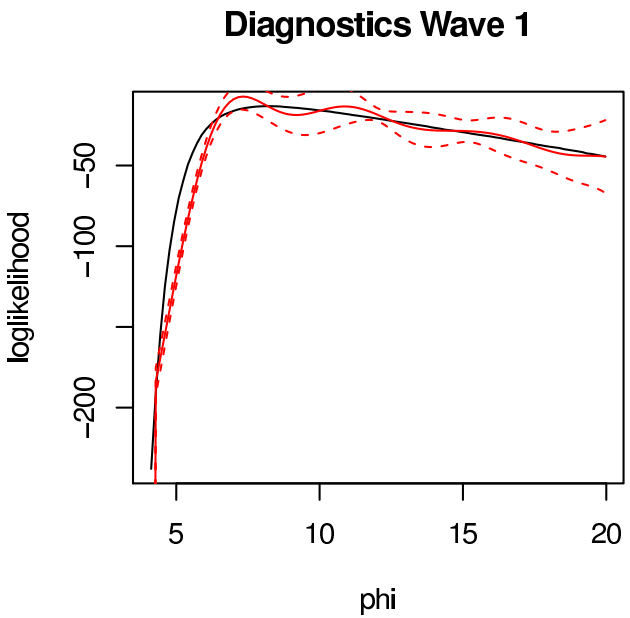




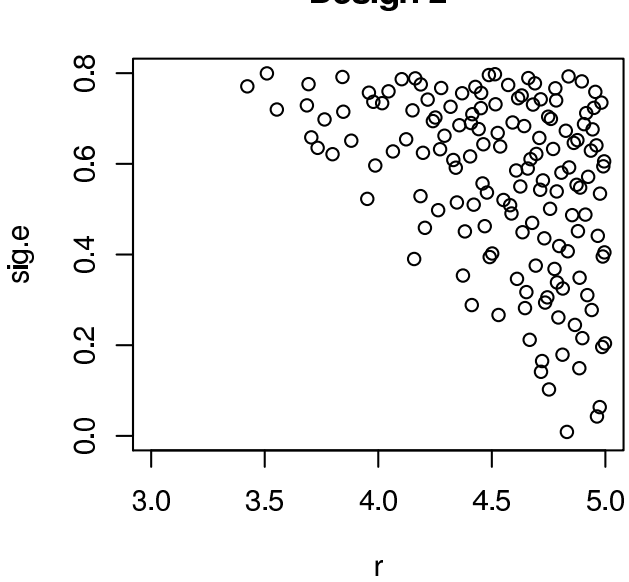
Diagnostics for GP 2 - threshold = -21.8

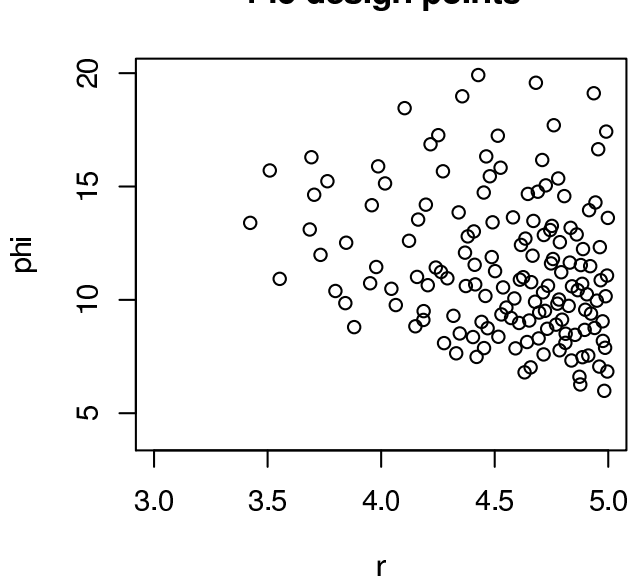


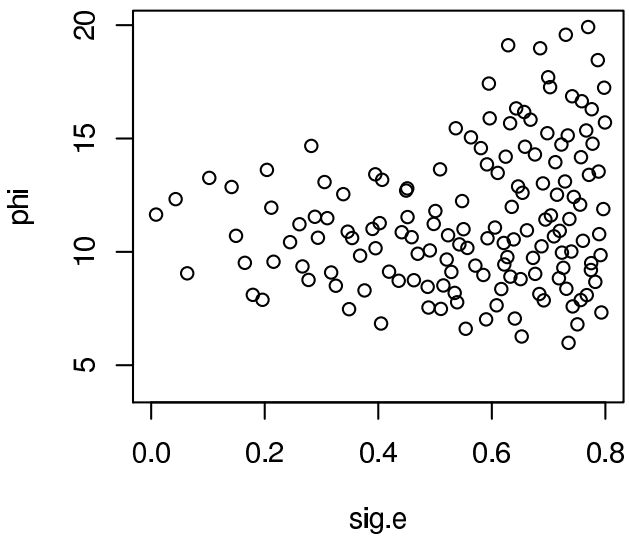




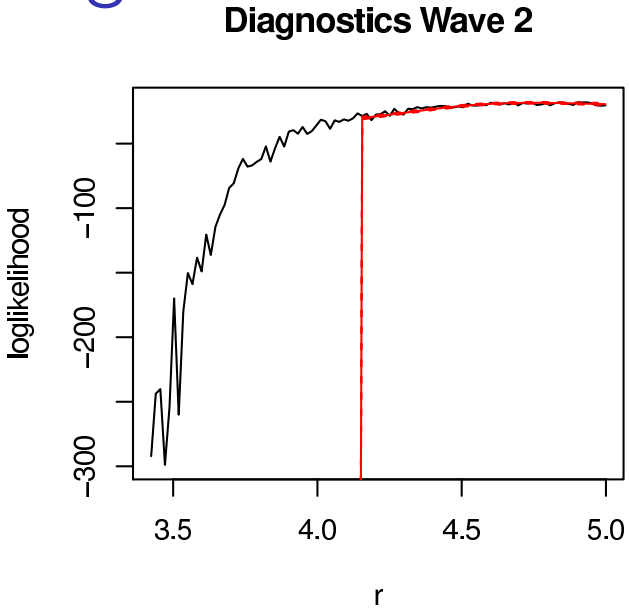
Design 3 - 149 pts - 62% of space implausible

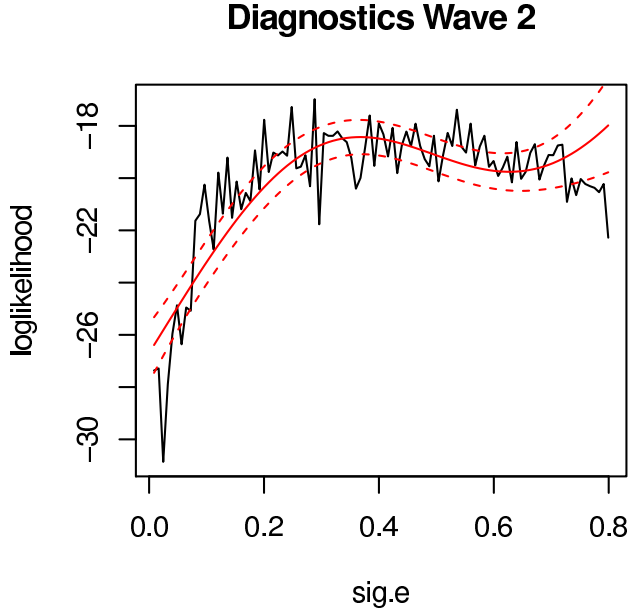


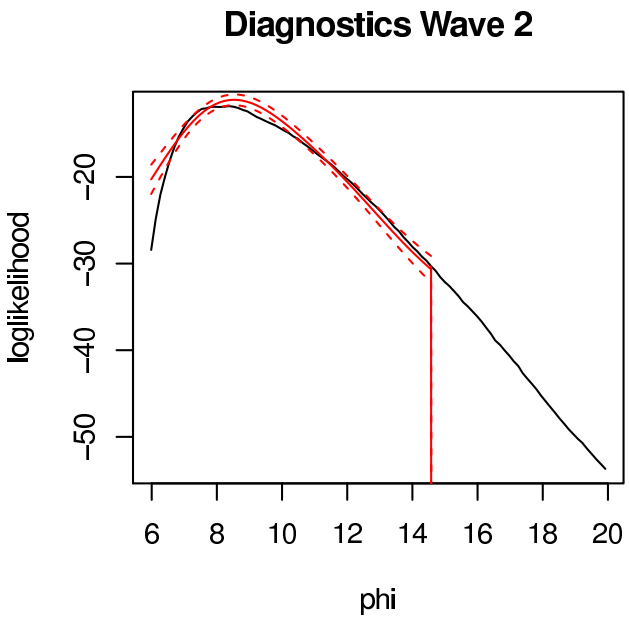




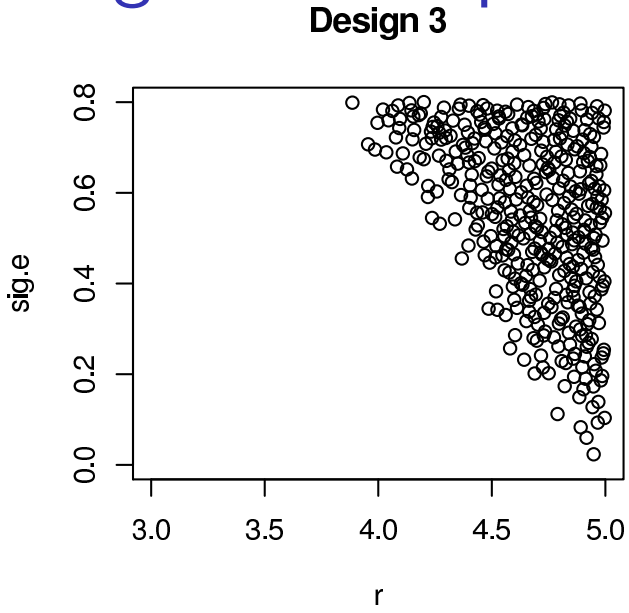
Diagnostics for GP 3 - threshold = -20.7

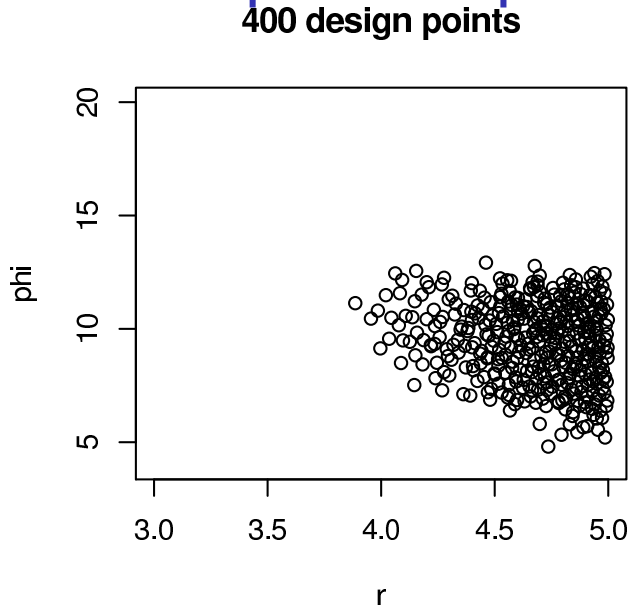


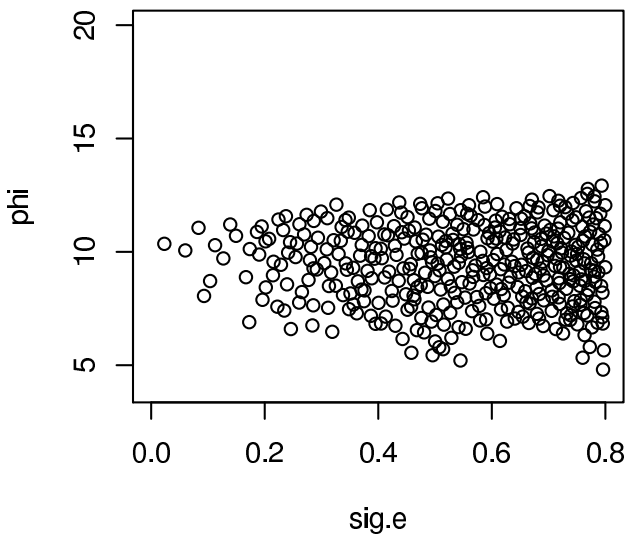




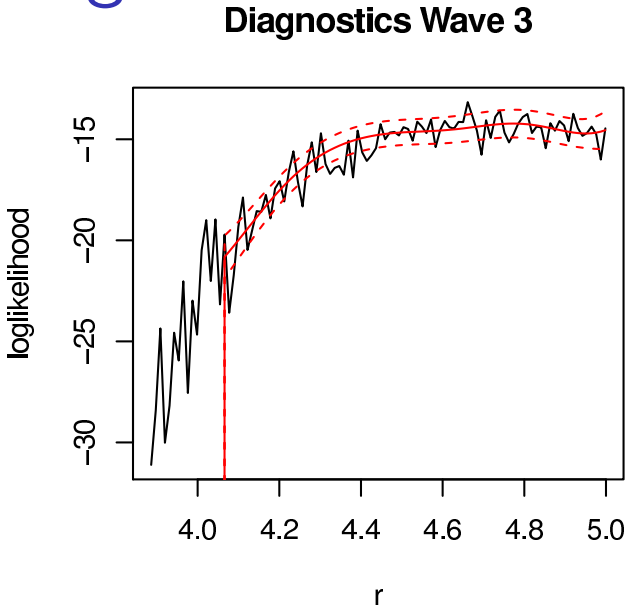
Design 4 - 400 pts - 95% of space implausible

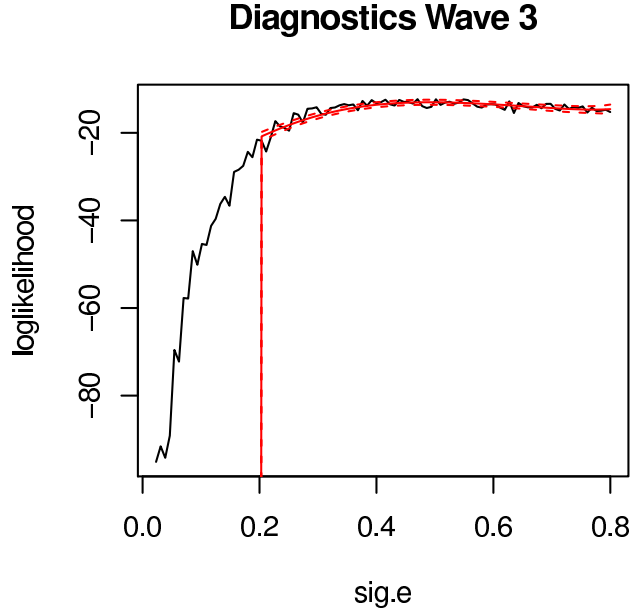


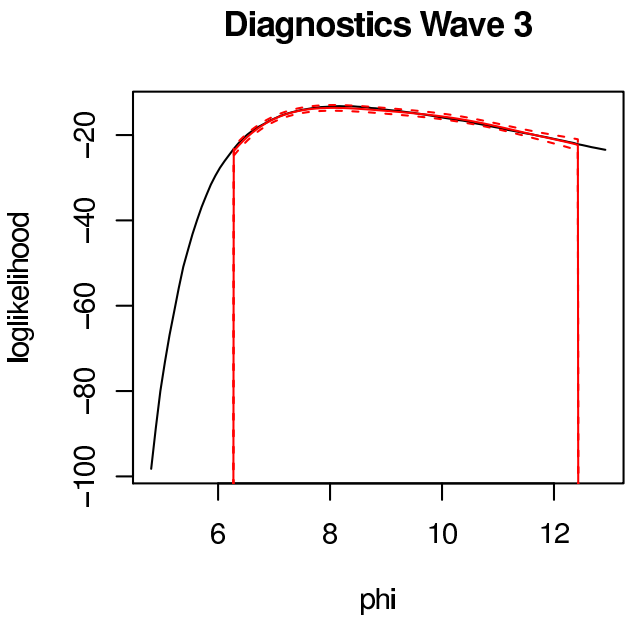




Diagnostics for GP 4 - threshold = -16.4

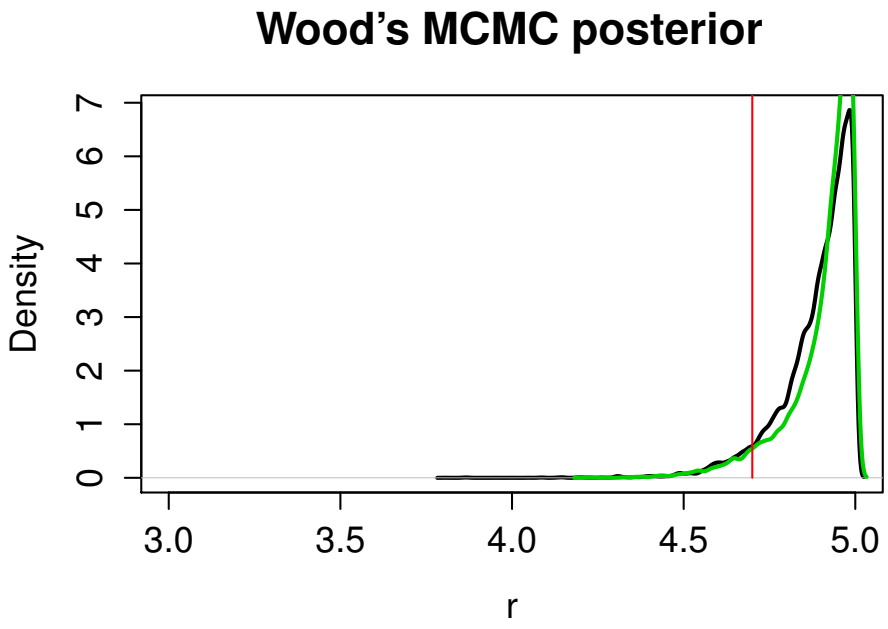


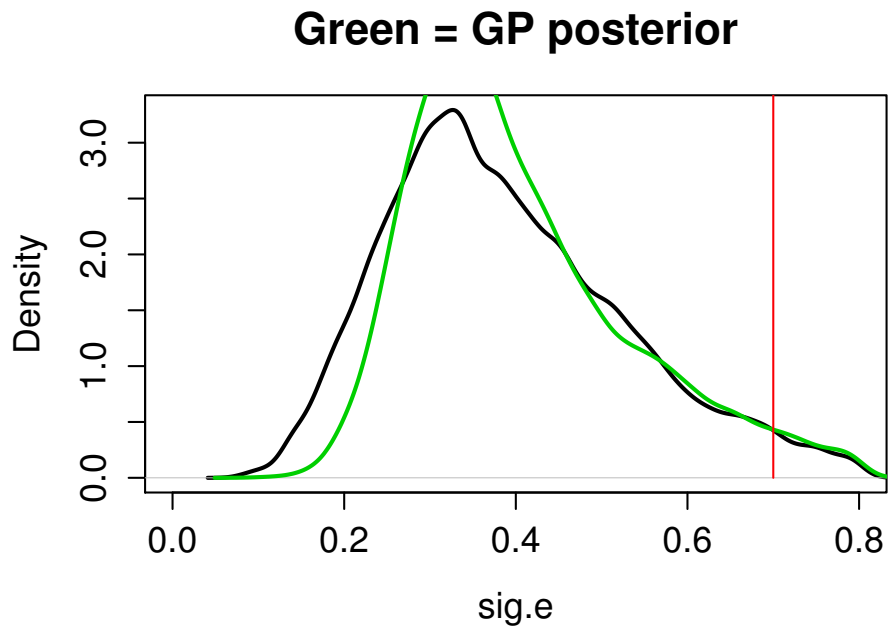


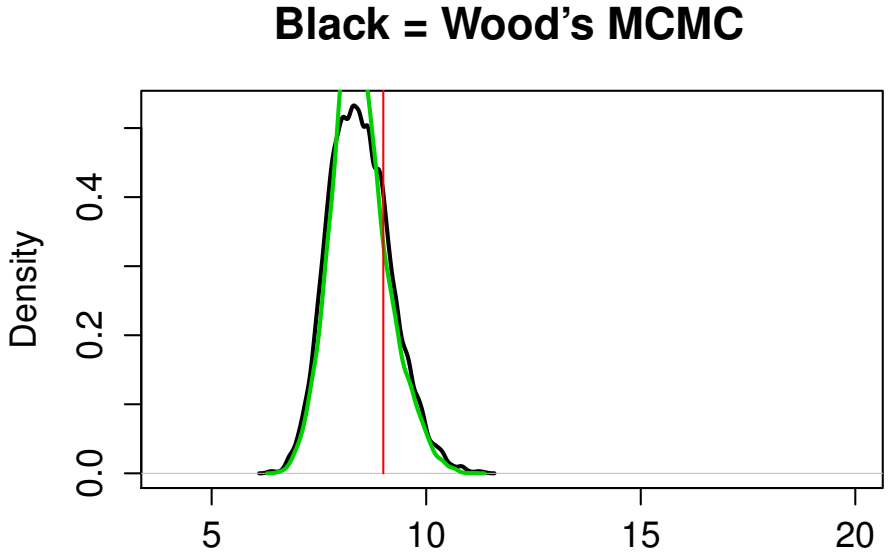


MCMC Results

Comparison with Wood 2010, synthetic likelihood approach







Computational details

- The Wood MCMC method used $10^5 \times 500$ simulator runs
- The GP code used $(128 + 314 + 149 + 400) = 991 \times 500$ simulator runs
  - 1/100th of the number used by Wood's method.

By the final iteration, the Gaussian processes had ruled out over 98% of the original input space as implausible,

- the MCMC sampler did not need to waste time exploring those regions.

Design for calibration

with James Hensman

Implausibility

When using emulators for history-matching and ABC, the aim is to accurately classify space as plausible or implausible by estimating the probability

$$p(\theta) = \mathbb{P}(\theta \in \mathcal{P}_\theta)$$

based upon a GP model of the simulator or likelihood

$$f(\theta) \sim GP(m(\cdot), c(\cdot, \cdot))$$

Implausibility

When using emulators for history-matching and ABC, the aim is to accurately classify space as plausible or implausible by estimating the probability

$$p(\theta) = \mathbb{P}(\theta \in \mathcal{P}_\theta)$$

based upon a GP model of the simulator or likelihood

$$f(\theta) \sim GP(m(\cdot), c(\cdot, \cdot))$$

The key determinant of emulator accuracy is the design used

$$D_n = \{\theta_i, f(\theta_i)\}_{i=1}^{N}$$

Entropic designs

Calibration doesn't need a global approximation to the simulator - this is wasteful

- Instead build a sequential design $\theta_1, \theta_2, \ldots$ using the current classification

$$p(\theta) = \mathbb{P}(\theta \in \mathcal{P}_\theta | D_n)$$

to guide the choice of design points

Entropic designs

Calibration doesn't need a global approximation to the simulator - this is wasteful

- Instead build a sequential design $\theta_1, \theta_2, \ldots$ using the current classification

$$p(\theta) = \mathbb{P}(\theta \in \mathcal{P}_\theta | D_n)$$

to guide the choice of design points

First idea: add design points where we are most uncertain

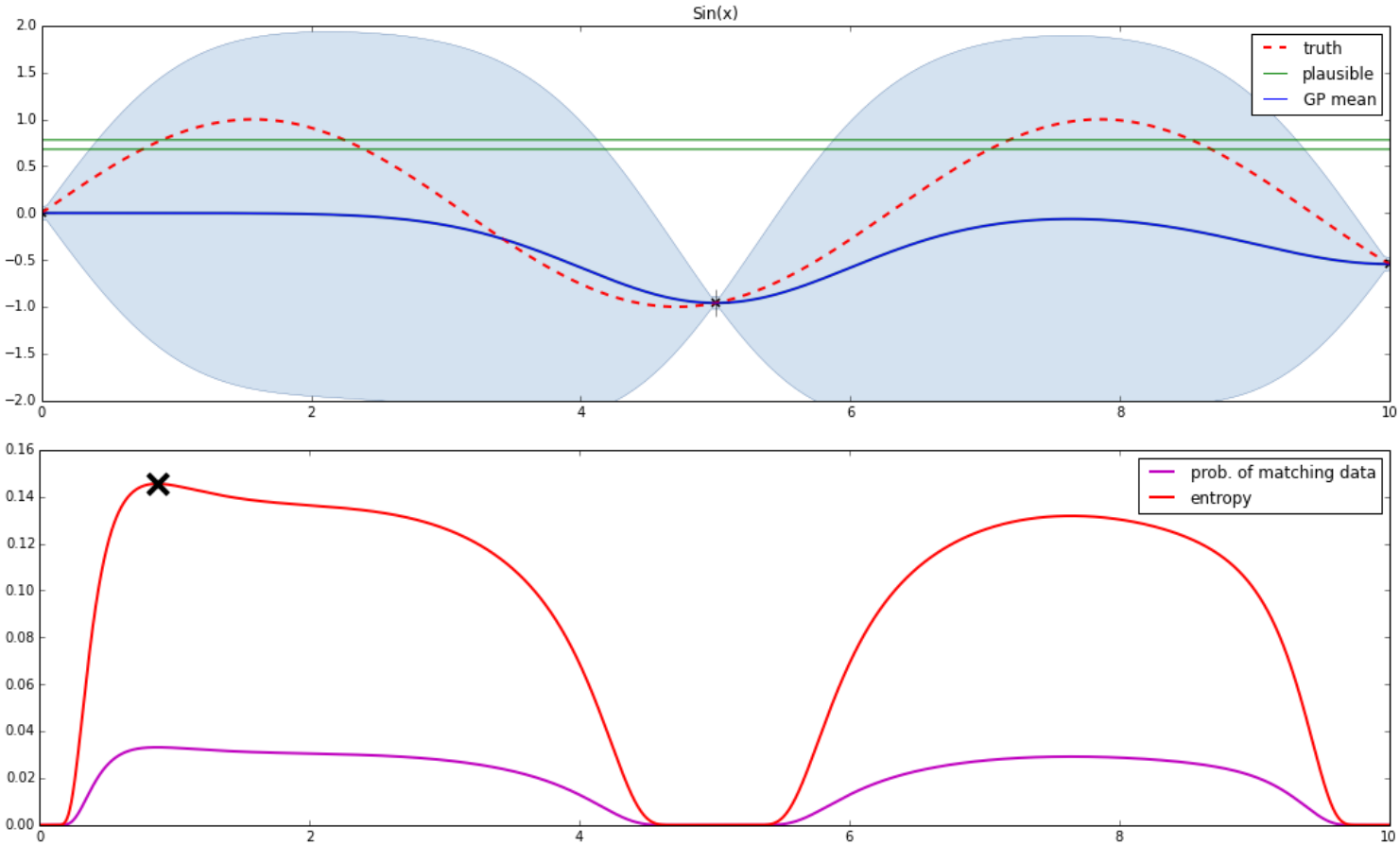
- The entropy of the classification surface is

$$E(\theta) = -p(\theta) \log p(\theta) - (1 - p(\theta)) \log(1 - p(\theta))$$

- Choose the next design point where we are most uncertain.

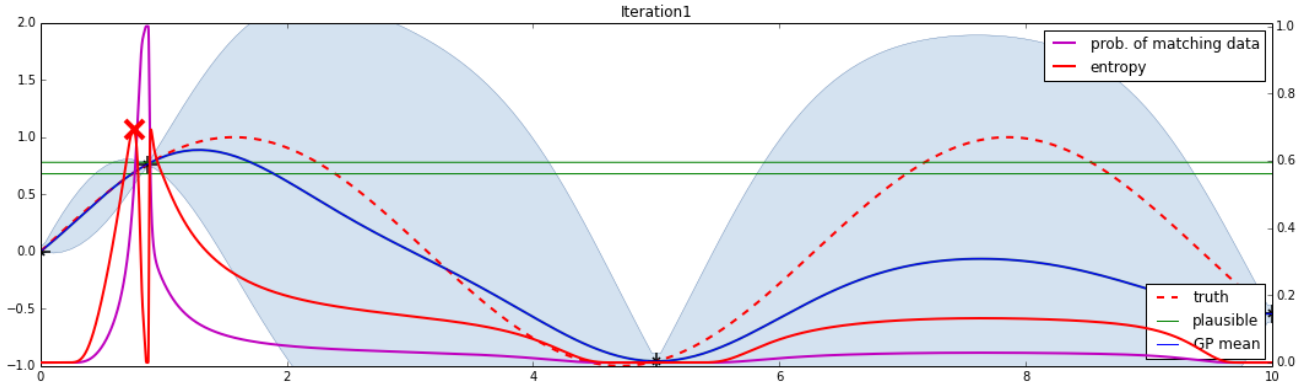
$$\theta_{n+1} = \arg\max E(\theta)$$

Toy 1d example $f(\theta) = \sin\theta$

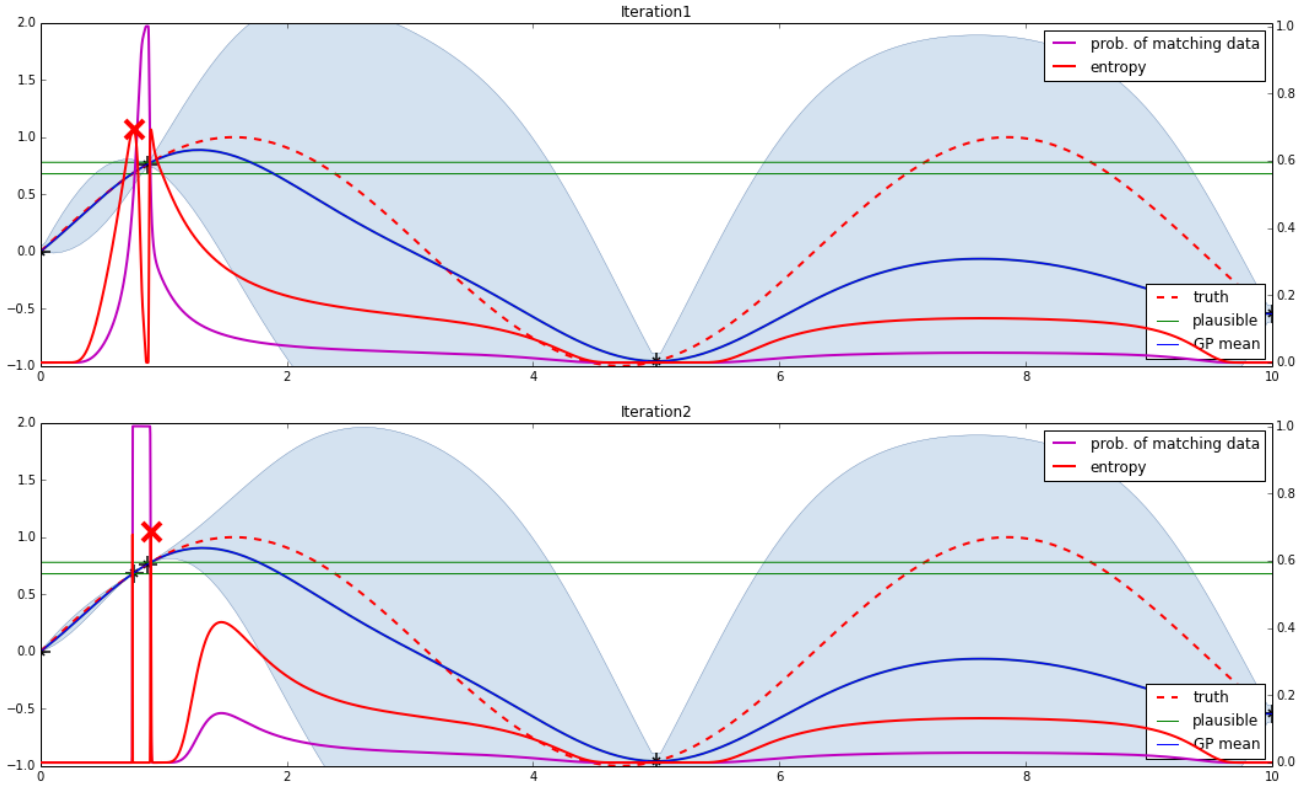


Add a new design point (simulator evaluation) at the point of greatest entropy

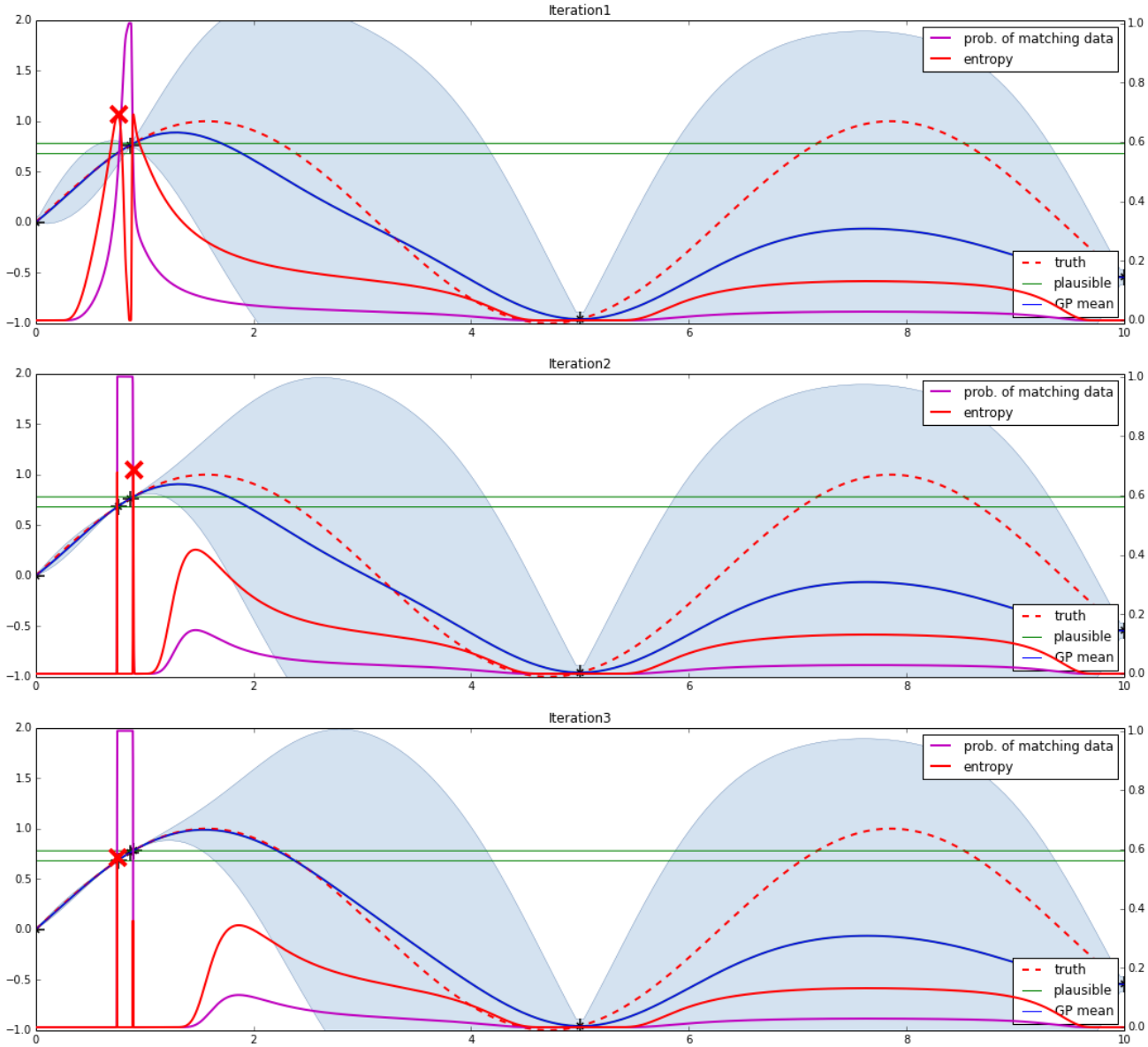
Toy 1d example $f(\theta) = \sin\theta$



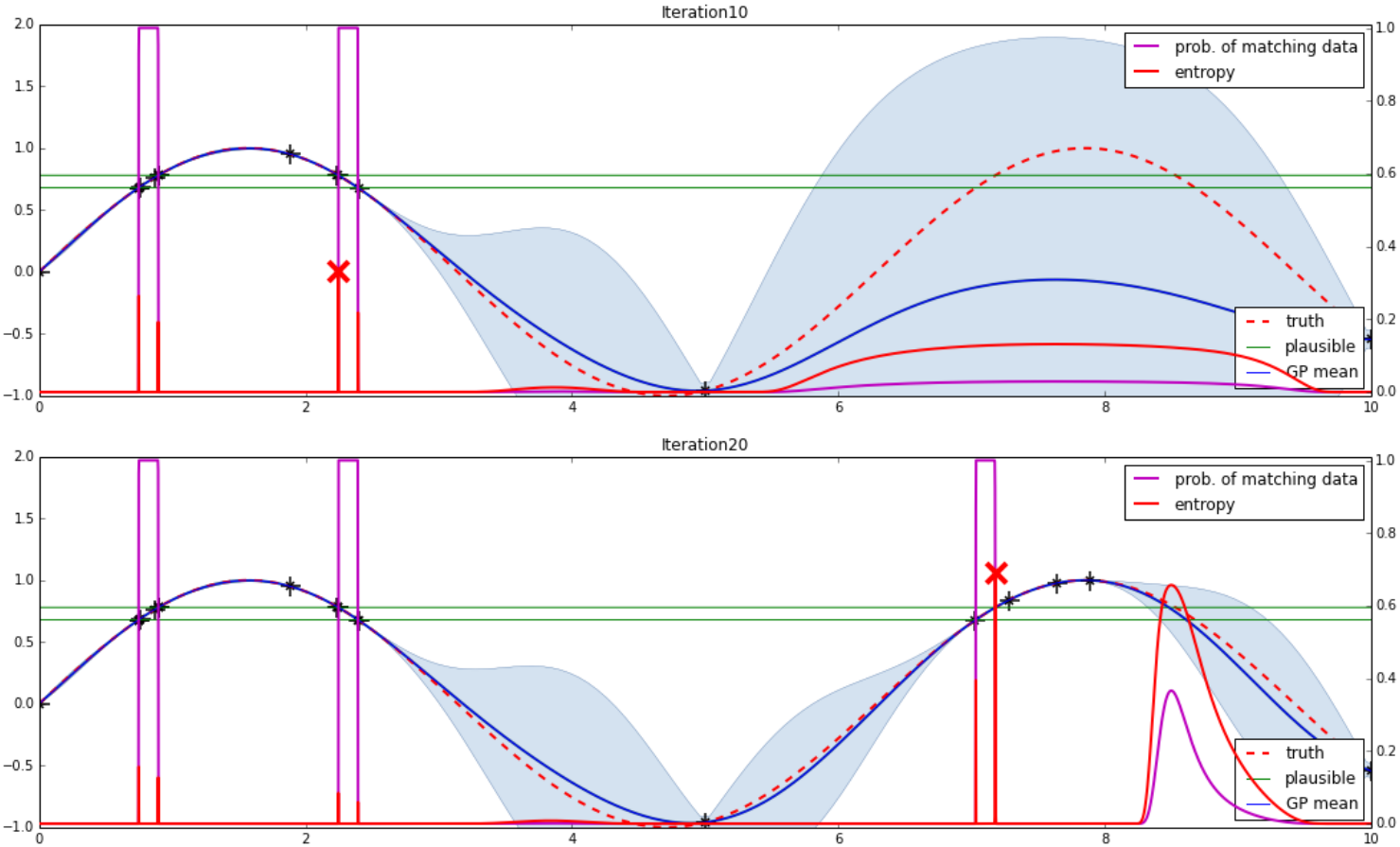
Toy 1d example $f(\theta) = \sin\theta$



Toy 1d example $f(\theta) = \sin\theta$



Toy 1d example $f(\theta) = \sin\theta$ - After 10 and 20 iterations



This criterion spends too long resolving points at the edge of the classification region.

Expected average entropy

Chevalier *et al.* 2014

Instead, we can find the average entropy of the classification surface

$$E_n = \int E(\theta) \mathrm{d}\theta$$

where n denotes it is based on the current design of size n.

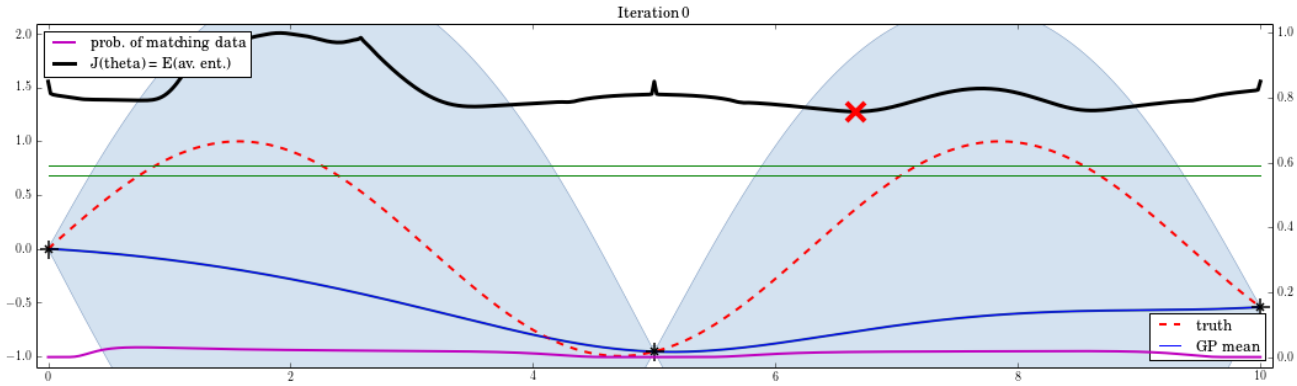
- Choose the next design point, θ_{n+1}, to minimise the expected average entropy

  $$\theta_{n+1} = \arg\min J_n(\theta)$$

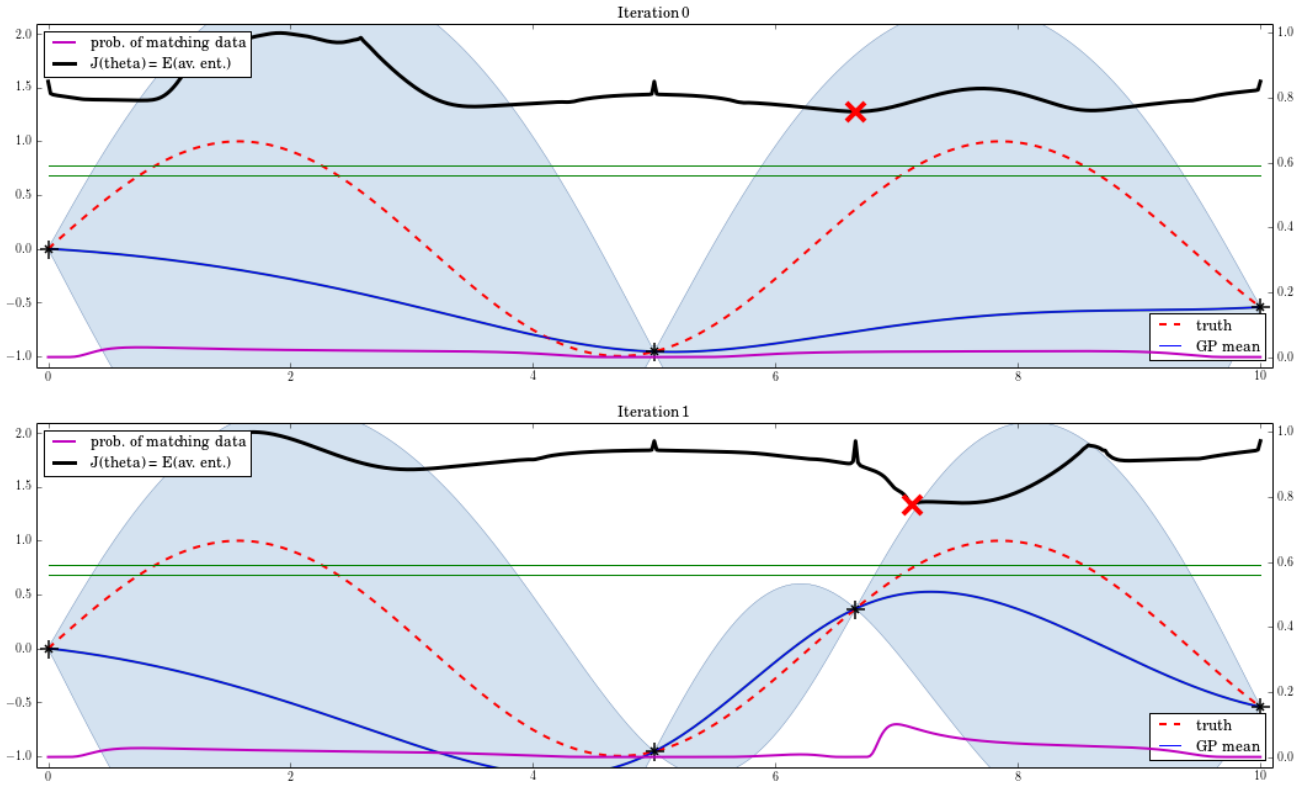
  where

  $$J_n(\theta) = \mathbb{E}(E_{n+1} | \theta_{n+1} = \theta)$$

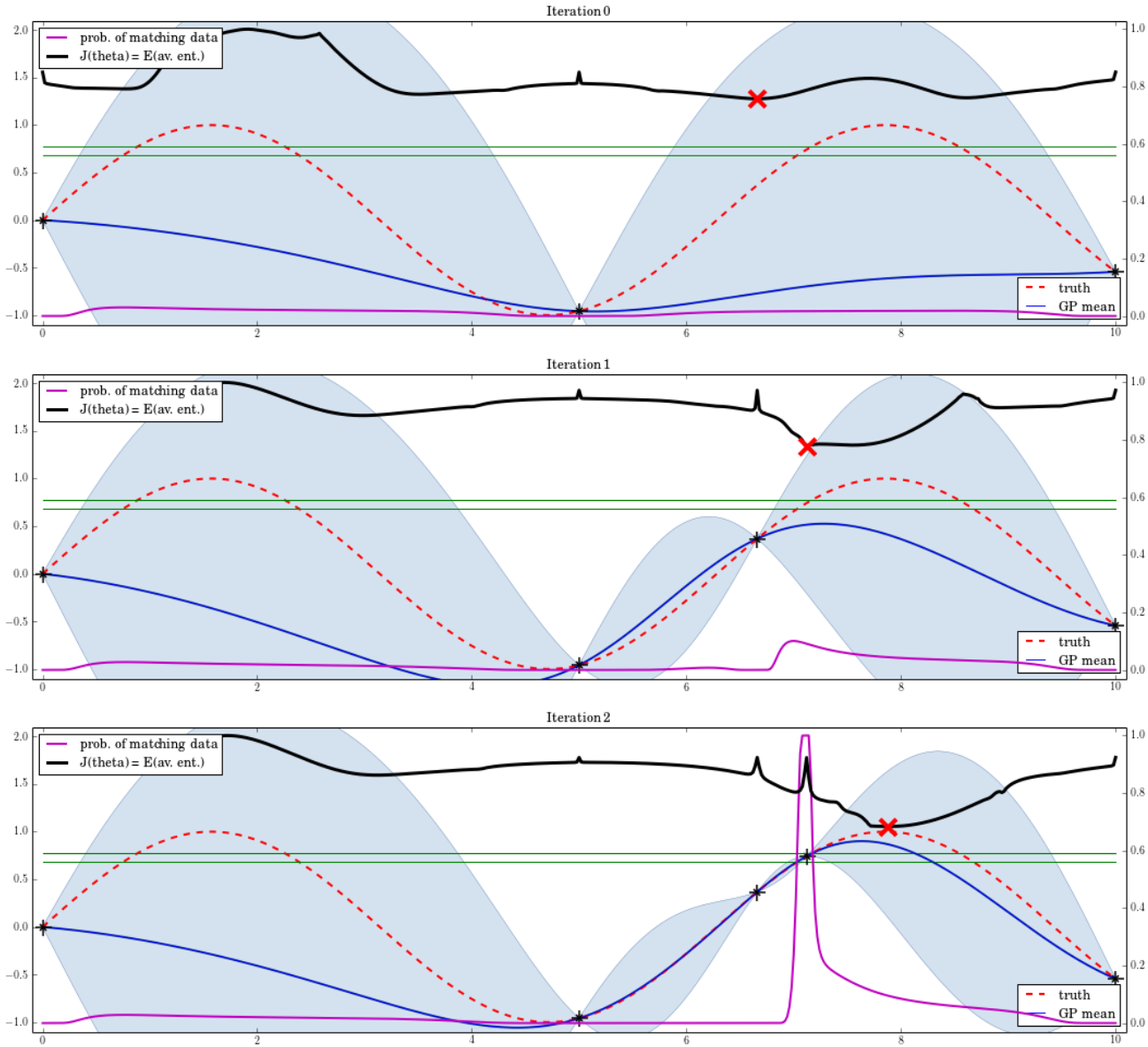
Toy 1d example $f(\theta) = \sin\theta$ - Expected entropy



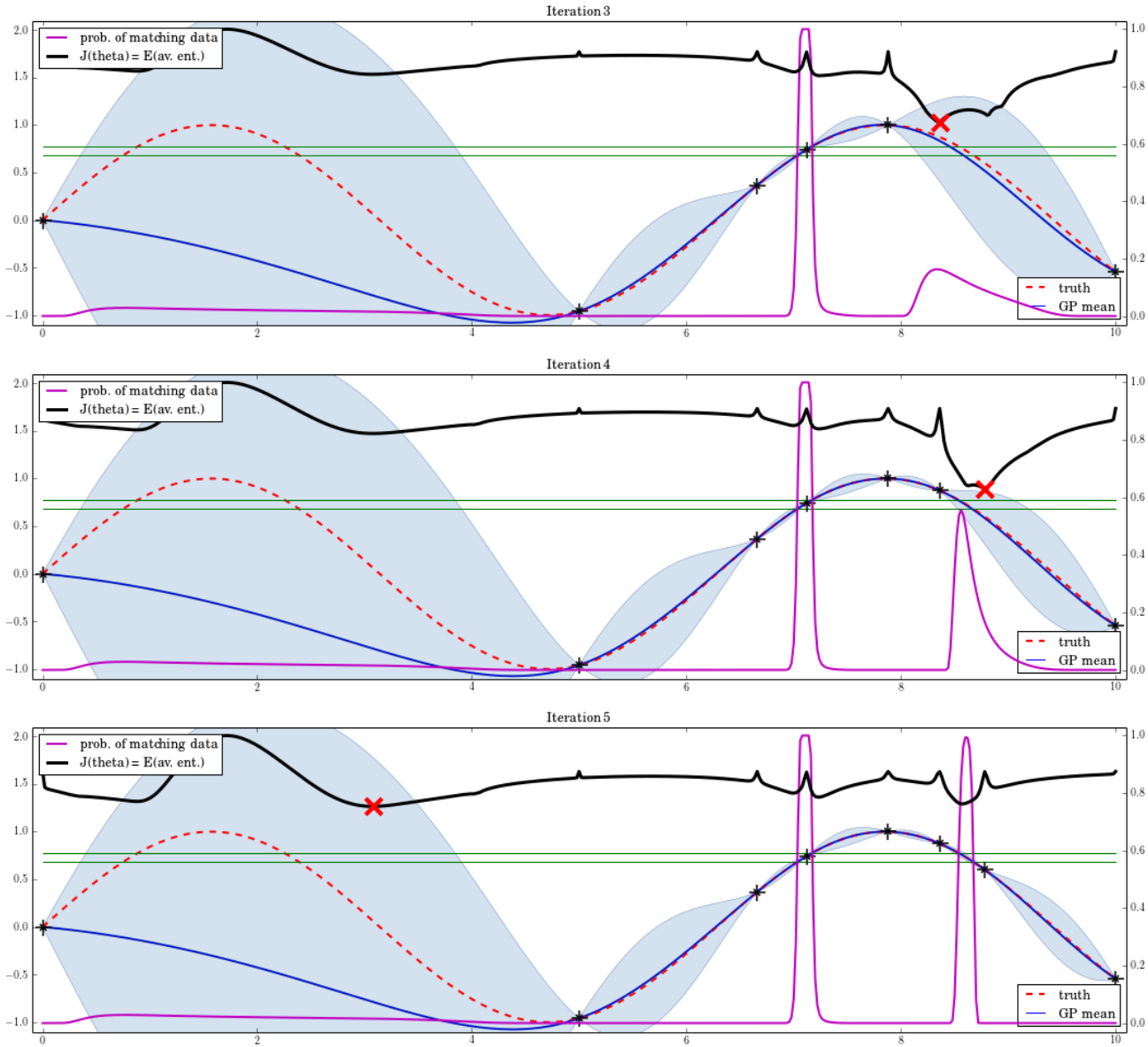
Toy 1d example $f(\theta) = \sin\theta$ - Expected entropy



Toy 1d example $f(\theta) = \sin\theta$ - Expected entropy

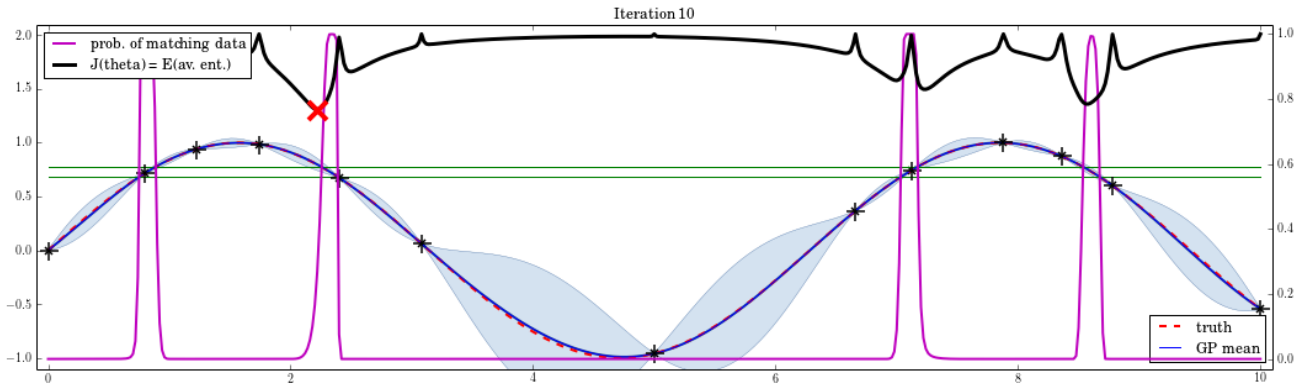


Toy 1d example $f(\theta) = \sin\theta$ - Expected entropy



Toy 1d: min expected entropy vs max entropy

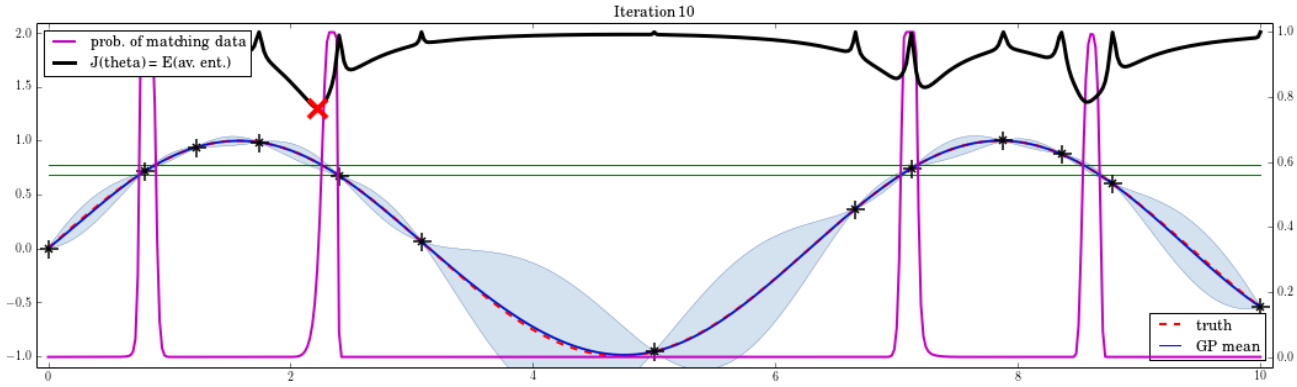
After 10 iterations, choosing the point of maximum entropy



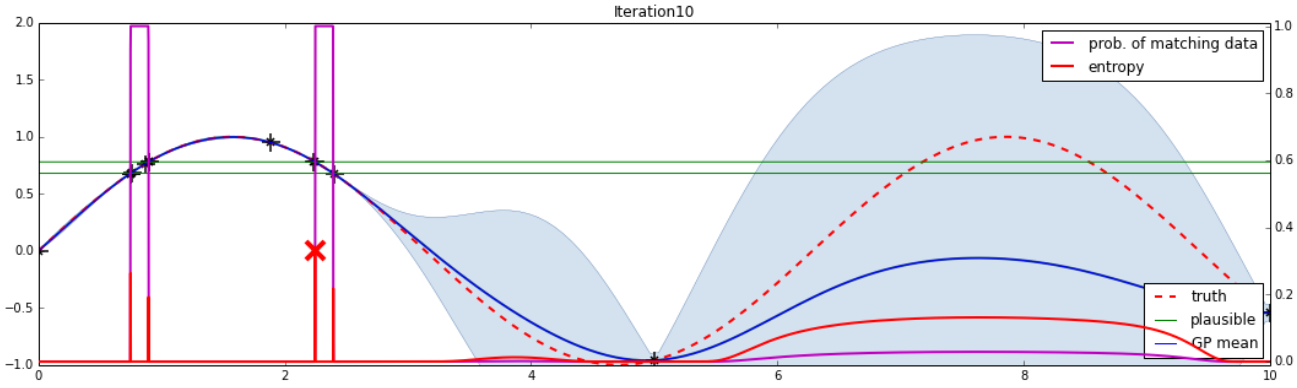
we have found the plausible region to reasonable accuracy.

Toy 1d: min expected entropy vs max entropy

After 10 iterations, choosing the point of maximum entropy



we have found the plausible region to reasonable accuracy.
Whereas maximizing the entropy has not



In 1d, a simpler space filling criterion would work just as well.

Solving the optimisation problem

Finding θ which minimises $J_n(\theta) = \mathbb{E}(E_{n+1}|\theta_{n+1} = \theta)$ is expensive.

- Even for 3d problems, grid search is prohibitively expensive
- Dynamic grids help

Solving the optimisation problem

Finding θ which minimises $J_n(\theta) = \mathbb{E}(E_{n+1}|\theta_{n+1} = \theta)$ is expensive.

- Even for 3d problems, grid search is prohibitively expensive
- Dynamic grids help

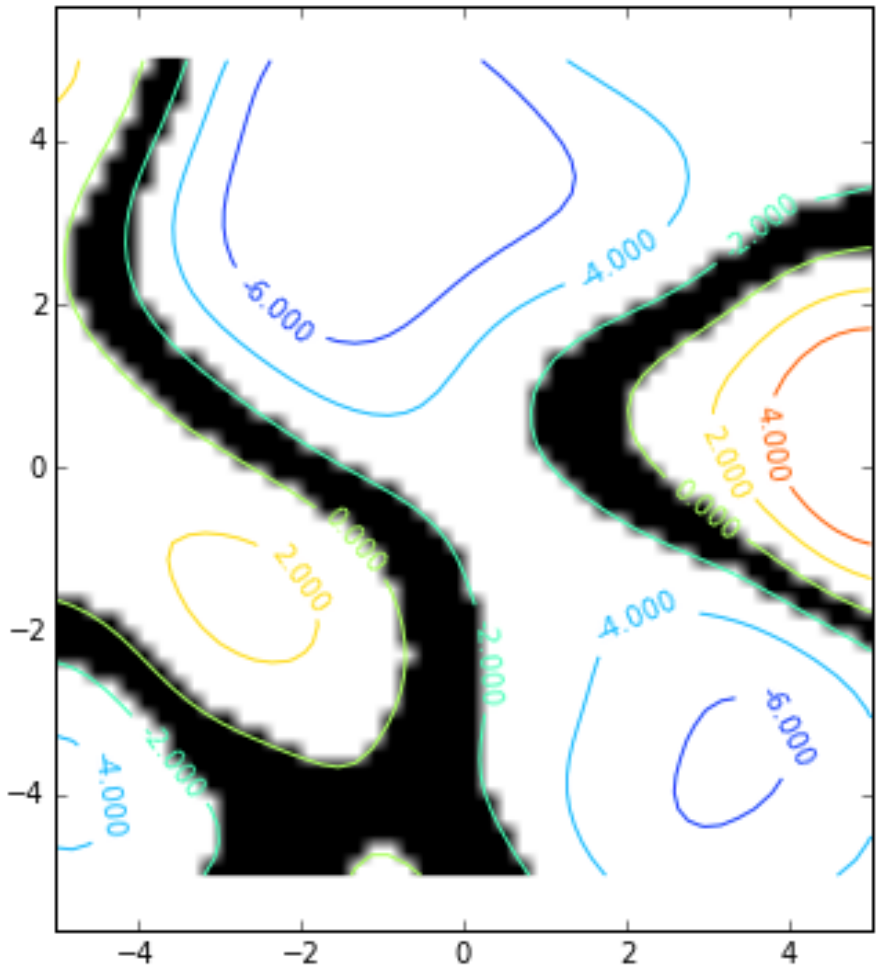
We can use Bayesian optimization to find the optima:

1. Evaluate $J_n(\theta)$ at a small number of locations
2. Build a GP model of $J_n(\cdot)$
3. Choose the next θ at which to evaluate J_n so as to minimise the expected-improvement (EI) criterion
4. Return to step 2.

History match

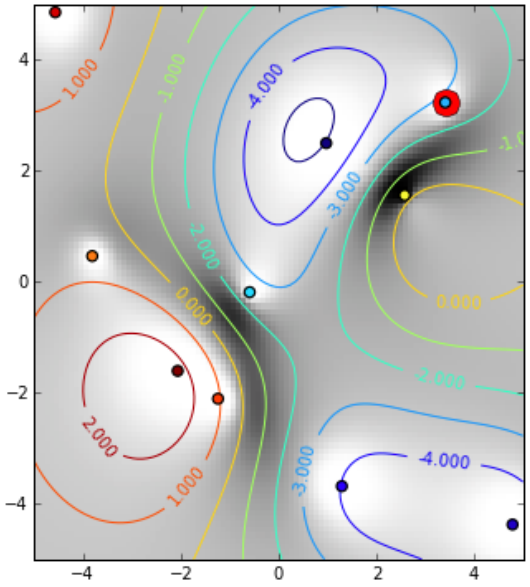
Can we learn the following plausible set?

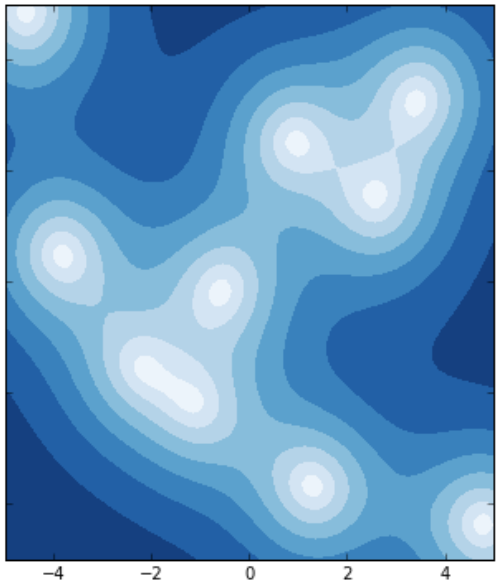
- A sample from a GP on $\mathbb{R}^2$.
- Find x s.t. $-2 < f(x) < 0$

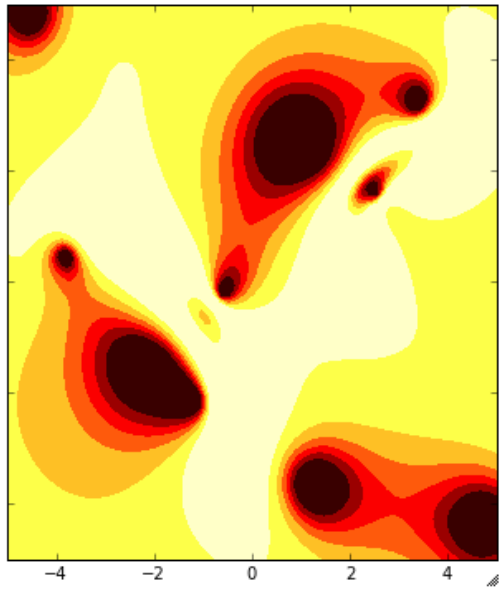


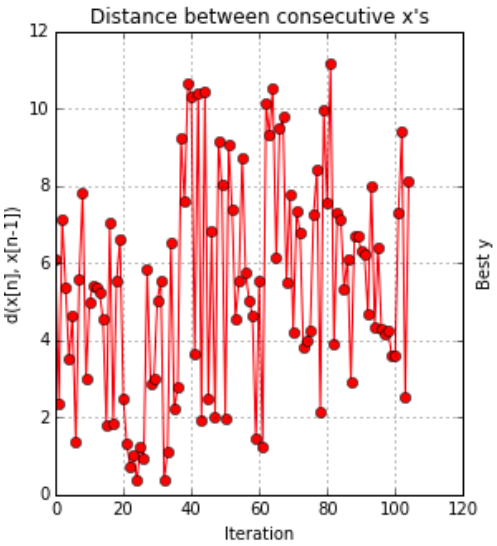
Iteration 10

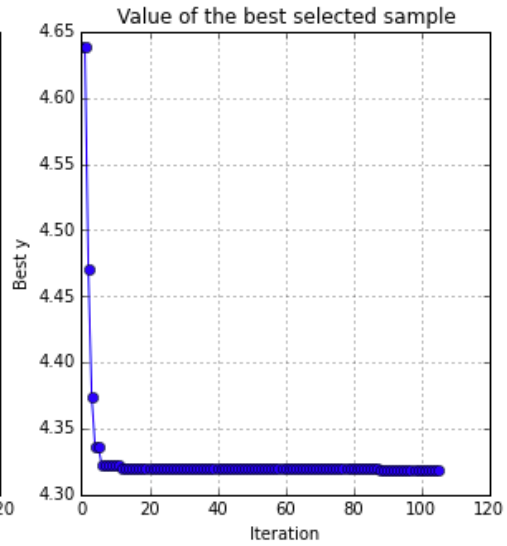
Left=$p(\theta)$, middle= $E(\theta)$, right = $\tilde{J}(\theta)$

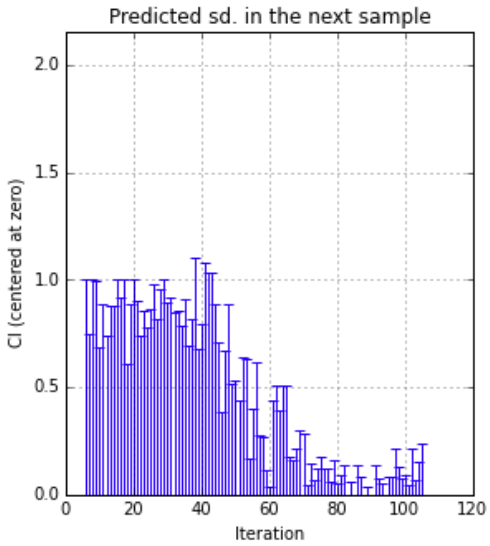






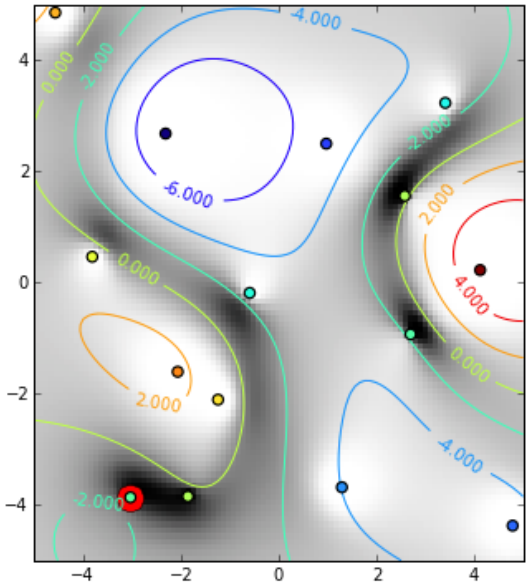


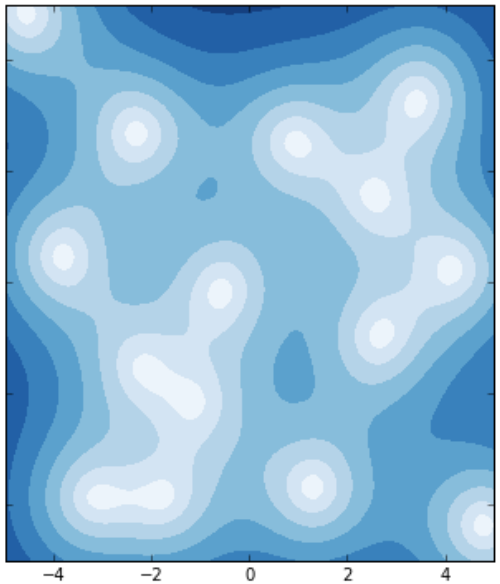


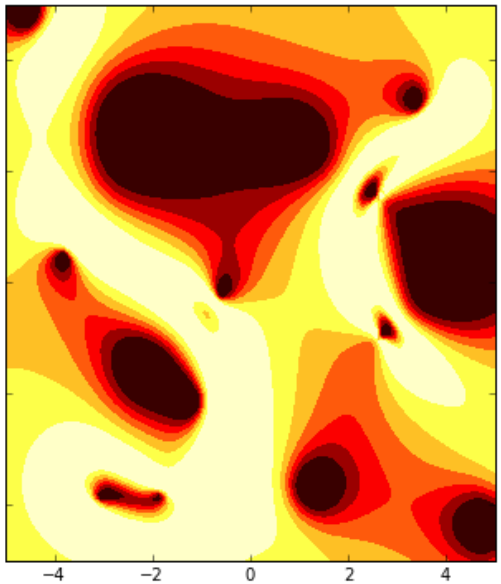


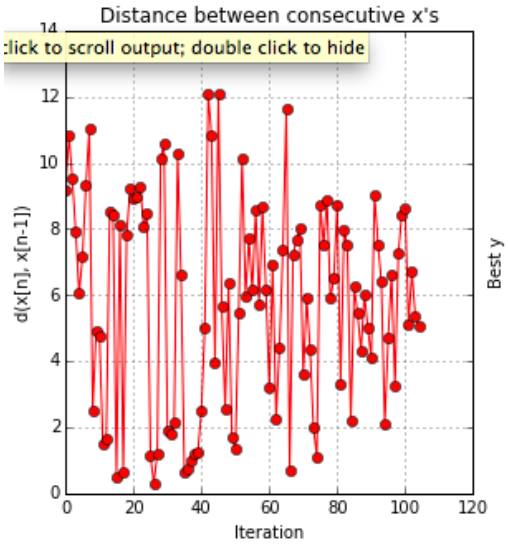
Iterations 15

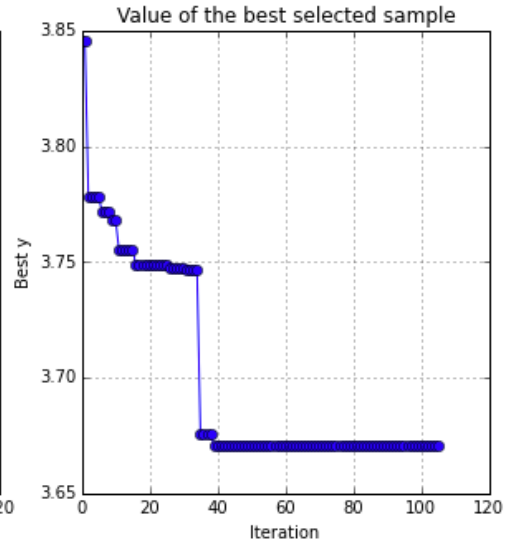
Left=$p(\theta)$, middle= $E(\theta)$, right = $\tilde{J}(\theta)$

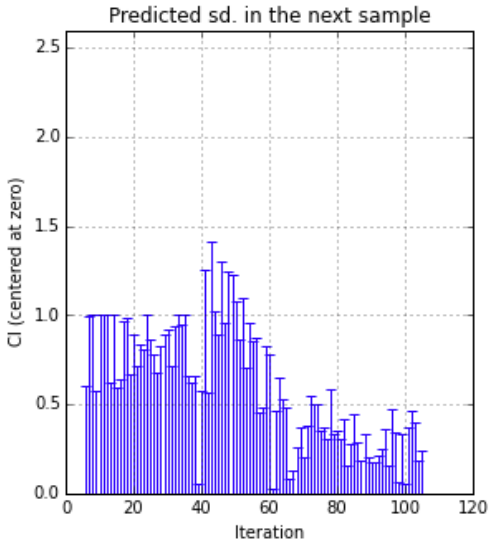




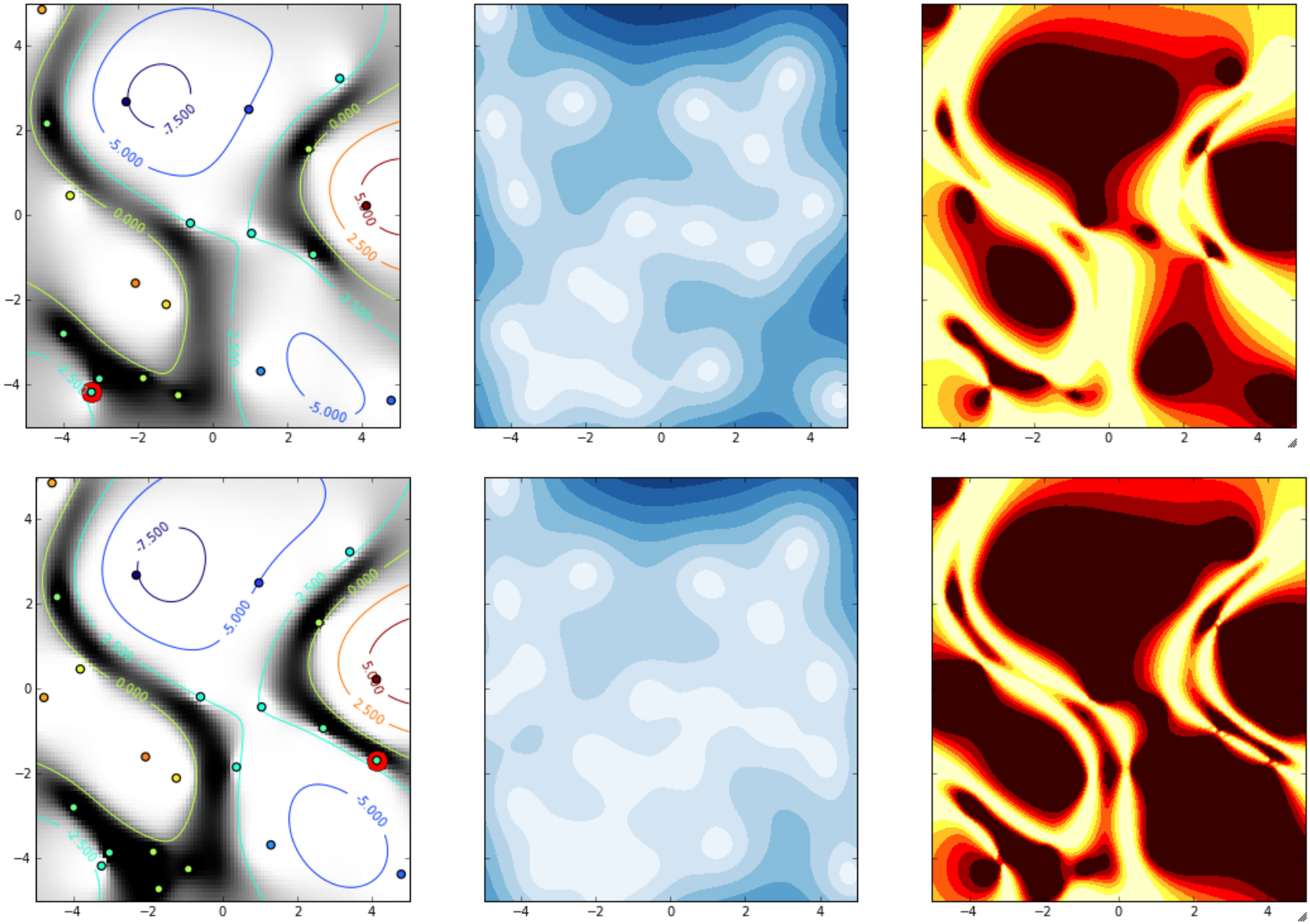








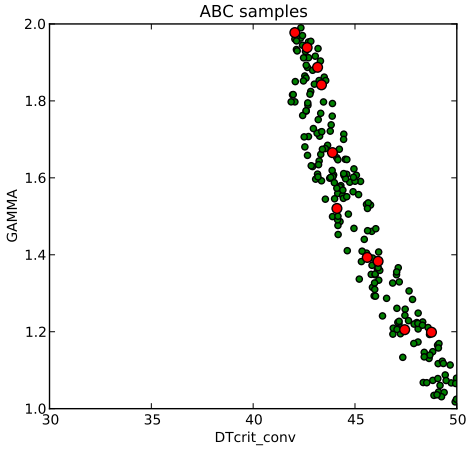
Iterations 20 and 24

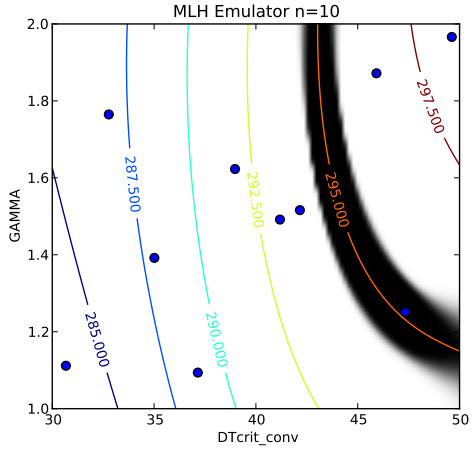


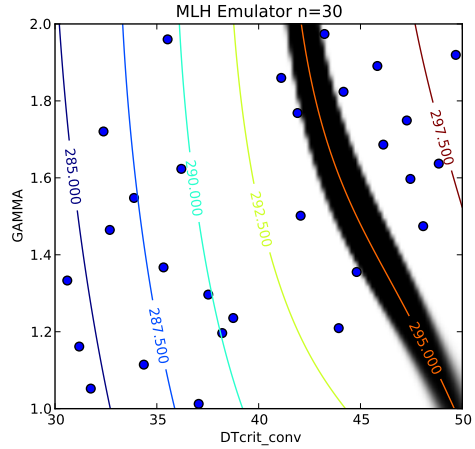
Video

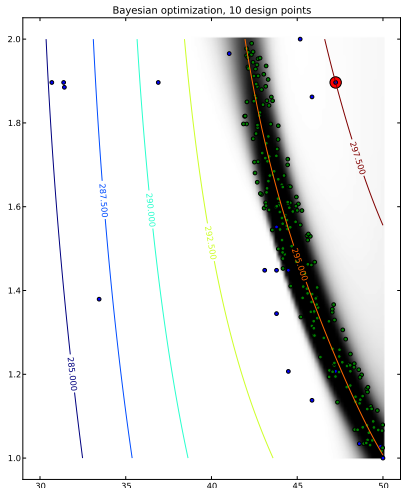
EPm: climate model

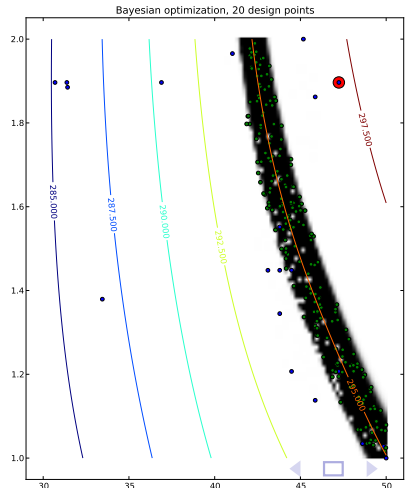
- 3d problem
- DTcrit_conv - critical temperature gradient that triggers convection
- GAMMA - emissivity parameter for water vapour
- Calibrate to global average surface temperature











Conclusions

Gaussian processes are widely used in the analysis of complex computer models

- Good design can lead to substantial improvements in accuracy
- Design needs to be specific to the task required
- Space-filling designs are inefficient for calibration and history matching problems.
- Entropy based designs give good trade-off between exploration and defining the plausible region
- Bayesian optimisation techniques allow us to solve the hard computation needed to use optimal entropic designs
- Using emulators of the likelihood function, although adding another layer of approximation, does enable progress in hard problems.

Conclusions

Gaussian processes are widely used in the analysis of complex computer models

- Good design can lead to substantial improvements in accuracy
- Design needs to be specific to the task required
- Space-filling designs are inefficient for calibration and history matching problems.
- Entropy based designs give good trade-off between exploration and defining the plausible region
- Bayesian optimisation techniques allow us to solve the hard computation needed to use optimal entropic designs
- Using emulators of the likelihood function, although adding another layer of approximation, does enable progress in hard problems.

Thank you for listening!

r.d.wilkinson@sheffield.ac.uk