ABC and the challenges of model calibration

Richard Wilkinson

School of Mathematical Sciences
University of Nottingham

With thanks to Michel Crucifix, Michael Goldstein and others.

Newton Institute - September 2010

Introduction

Consider the following three parts of inference:

1. Modelling
2. Inferential framework
3. Computation

Introduction

Consider the following three parts of inference:

1. Modelling
   - Physical model - generative?
   - Statistical model - observation error, model error - relating the model to the system
   - Prior distributions?
2. Inferential framework
3. Computation

Introduction

Consider the following three parts of inference:

1. Modelling
   - Physical model - generative?
   - Statistical model - observation error, model error - relating the model to the system
   - Prior distributions?
2. Inferential framework
   - Bayesian: find $\pi(\theta|\mathcal{D}) \propto \pi(\theta)\pi(\mathcal{D}|\theta)$ nb. implicit on model choices
3. Computation

Introduction

Consider the following three parts of inference:

1. Modelling
   - Physical model - generative?
   - Statistical model - observation error, model error - relating the model to the system
   - Prior distributions?
2. Inferential framework
   - Bayesian: find $\pi(\theta|\mathcal{D}) \propto \pi(\theta)\pi(\mathcal{D}|\theta)$ nb. implicit on model choices
   - Static model - aim to find $\pi(\theta|\mathcal{D})$
   - Dynamic model - aim to find $\pi(\theta, x_{0:t}|y_{0:t})$
3. Computation

Introduction

Consider the following three parts of inference:

1. Modelling
   - Physical model - generative?
   - Statistical model - observation error, model error - relating the model to the system
   - Prior distributions?
2. Inferential framework
   - Bayesian: find $\pi(\theta|\mathcal{D}) \propto \pi(\theta)\pi(\mathcal{D}|\theta)$ nb. implicit on model choices
   - Static model - aim to find $\pi(\theta|\mathcal{D})$
   - Dynamic model - aim to find $\pi(\theta, x_{0:t}|y_{0:t})$
3. Computation
   - Monte Carlo integration
   - MCMC
   - Emulators
   - Particle filters
   - ABC

Introduction

Consider the following three parts of inference:

1. Modelling
   - Physical model - generative?
   - Statistical model - observation error, model error - relating the model to the system
   - Prior distributions?
2. Inferential framework
   - Bayesian: find $\pi(\theta|\mathcal{D}) \propto \pi(\theta)\pi(\mathcal{D}|\theta)$ nb. implicit on model choices
   - Static model - aim to find $\pi(\theta|\mathcal{D})$
   - Dynamic model - aim to find $\pi(\theta, x_{0:t}|y_{0:t})$
3. Computation - this remains hard even with increased computational resource
   - Monte Carlo integration
   - MCMC
   - Emulators
   - Particle filters
   - ABC

The Van der Pol Oscillator

Seemingly simple systems can pose difficult computational challenges. Consider the deterministic Van der Pol oscillator

- Two dimensional state vector $x_t = (x(t), y(t))$

$$\frac{\mathrm{dx}}{\mathrm{dt}} = -\tau y \qquad \frac{\mathrm{dy}}{\mathrm{dt}} = \alpha\tau(x - y^3/3 + y)$$

Even in the simple case where we have two unknown parameters and two unknown initial conditions $\theta = (\tau, \alpha, x_0, y_0)$, it can be difficult to estimate these parameters using most assimilation schemes.

The Van der Pol Oscillator

To illustrate the difficulties and methodology we generate some data from the model (so there is no model error) and add noise.

$$d_t^x \sim N(x_t, 0.15^2) \qquad d_t^y \sim N(y_t, 0.3^2)$$

Generate 500 observations from 8 cycles of the system, using true parameter value $\theta = c(1, 5, 1, 1)$. Let $\mathcal{D}$ denote the observations.

Can we learn θ? Should be easy!

MCMC

MCMC tends to be the default choice for Bayesian parameter inference. The idea is to iteratively simulate a Markov chain on Θ which has stationary distribution $\pi(\theta|\mathcal{D})$.
Suppose we are currently at θ.

1. Propose a move to θ' from perturbation kernel $q(\theta, \theta')$
2. Simulate the trajectory x' using parameter θ'
3. Accept the move to θ' with probability

$$MH = \min\left(1, \frac{\pi(\theta')q(\theta', \theta)\prod \pi(\mathrm{d}_t \mid x'_t)}{\pi(\theta)q(\theta, \theta')\prod \pi(\mathrm{d}_t \mid x_t)}\right)$$

   otherwise stay at θ.

NB - each step requires that we simulate from the model.

MCMC

The challenge with MCMC is to choose a good proposal distribution q for the chains dynamics. We try the default choice

- A Gaussian random walk proposal scheme
  - randomly choose between the following updates.
    - $\log \tau' \sim N(\log \tau, 0.01)$
    - $\log \alpha' \sim N(\log \alpha, 0.02)$
    - $x_0' \sim N(x_0, 0.1)$
    - $y_0' \sim N(y_0, 0.1)$

I spent a very small amount of time tuning the step-sizes ($\approx$ 3 hours of coding time). The step-size was chosen to try to roughly get 30% of proposals accepted.

MCMC results - observing only x

After 10^5 model evalutations (3 hr 15min), we have a chain of 10^5 values of θ.
Our start point of the chain had a log-likelihood value of $l(x) = -16610$ and the best trajectory we found had $l(x) = -8724$ so we have found a region of higher likelihood -but the fit is still far from satisfactory.

MCMC results

It is clear that we haven't got close to the true parameter values.

The problem is caused by poor mixing in the chains:

Huge improvements in performance could surely be made with better tuning, and by using more sensible proposal distributions.

- Adaptive MCMC schemes (Roberts and Rosenthal 2009) could help with the tuning
- Langevin and Hamiltonian Monte Carlo methods (Girolami and Calderhead 2010) may provide better choices for q.
- However, these methods rely on knowledge of the underlying odes and their derivatives.

Van der Pol likelihood surface

The difficulty of parameter estimation can be explained by looking at the likelihood surface for (τ_1, α) given the **true** initial conditions.

Van der Pol likelihood surface

These problems are only likely to get worst as $\dim(\theta)$ increases. The plot above only showed two of our four parameters varying.

If the model is imperfect the likelihood surface could look even worse, especially if no parameters lead to a good match between the simulator and the data.

Van der Pol likelihood surface

These problems are only likely to get worst as $\dim(\theta)$ increases. The plot above only showed two of our four parameters varying.

If the model is imperfect the likelihood surface could look even worse, especially if no parameters lead to a good match between the simulator and the data.

Note that this type of Bayesian calibration must always result in a posterior distribution for θ. Even if the model is wrong and no match exists.

In this case, we know the model is perfect. However the MCMC sampler gets lost in the mountainous likelihood surface - all the fits it finds are poor, but it still returns a posterior distribution. We must be careful not just to take the output at face value (esp. with emulators).

Approximate Bayesian Computation (ABC)

The challenged faced in the analysis of computer experiments is that we do not typically know the likelihood function, and must instead do inference using only simulation from the model.

Approximate Bayesian computation (ABC) algorithms are a group of Monte Carlo algorithms used for finding posterior distributions

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

ABC methods have recently become popular in the biological sciences.

Although their current incarnation originates from a 1999 paper (Pritchard *et al.*) version of the algorithm exist in most modelling communities.

Likelihood-Free Inference

Rejection Algorithm

- Draw θ from prior $\pi(\cdot)$
- Accept θ with probability $\pi(\mathcal{D} \mid \theta)$

Accepted θ are independent draws from the posterior distribution, $\pi(\theta \mid \mathcal{D})$.

Likelihood-Free Inference

Rejection Algorithm

- Draw θ from prior $\pi(\cdot)$
- Accept θ with probability $\pi(\mathcal{D} \mid \theta)$

Accepted θ are independent draws from the posterior distribution, $\pi(\theta \mid \mathcal{D})$.
If the likelihood, $\pi(\mathcal{D}|\theta)$, is unknown:

'Mechanical' Rejection Algorithm

- Draw θ from $\pi(\cdot)$
- Simulate $X \sim f(\theta)$ from the computer model
- Accept θ if $\mathcal{D} = X$, i.e., if computer output equals observation

The acceptance rate is $\mathbb{P}(\mathcal{D})$: the number of runs to get n observations is negative binomial, with mean $\frac{n}{\mathbb{P}(\mathcal{D})}$: $\Rightarrow$ Bayes Factors!

Approximate Bayesian Computation I

If $\mathbb{P}(\mathcal{D})$ is small, we will rarely accept any θ. Instead, there is an approximate version:

Approximate Rejection Algorithm

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

Approximate Bayesian Computation I

If $\mathbb{P}(\mathcal{D})$ is small, we will rarely accept any θ. Instead, there is an approximate version:

Approximate Rejection Algorithm

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

This generates observations from $\pi(\theta \mid \rho(\mathcal{D}, X) < \delta)$:

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

δ reflects the tension between computability and accuracy.

Approximate Bayesian Computation II

If the data are too high dimensional we never observe simulations that are 'close' to the field data.
Reduce the dimension using summary statistics, $S(\mathcal{D})$.

Approximate Rejection Algorithm With Summaries

- Draw θ from $\pi(\theta)$
- Simulate $X \sim f(\theta)$
- Accept θ if $\rho(S(\mathcal{D}), S(X)) < \delta$

If S is sufficient this is equivalent to the previous algorithm.

There are many extensions: approximate MCMC, approximate SMC, approximate particle-MCMC, etc.

ABC - Van der Pol Oscillator

Rather than use the entire time-series of data $d_{0:T}$ we use a couple of summaries:

- $C = (C_x, C_y)$ number of crossings of zero by the x and y trajectories
- x_0 the initial condition.

For the observed values of these summaries we can count 17 crossings for both x and y, and we take the first observation as the true value of x_0.

ABC - Van der Pol Oscillator

Rather than use the entire time-series of data $d_{0:T}$ we use a couple of summaries:

- $C = (C_x, C_y)$ number of crossings of zero by the x and y trajectories
- x_0 the initial condition.

For the observed values of these summaries we can count 17 crossings for both x and y, and we take the first observation as the true value of x_0. For the metric I tried a simple weighted absolute difference

$$\rho(S, S') = \sum |C_\alpha - C'_\alpha| + 10|x_0 - x'_0|$$

where the primes indicate simulated values. The factor of 10 is there to give both components roughly equal importance.

- With exploration and better understanding, better choices could be made - for example by considering the error on x_0

After 10^5 model evalutations we find we accept 191 runs when $\delta = 1$, and 1151 runs for $\delta = 2$. Plotting the posteriors for $\delta = 2$ we find:

The posterior mode is close to every true parameter (red =true, green=prior).

Note that if we store all the output from the simulations, we are free to go back and experiment with different choices of the summary, metric and tolerance, to see how this affects our results.
For example, just calibrating to the number of crossings gives:

Or if we try to match the vector of crossing times plus the number of crossings.

SMC-ABC

Particle filter methods can also be adapted for use in an ABC setting.

Start with the sequential importance sampler of Del Moral et al. (2006) which aims to sample a selection of N particles successively from a sequence of distributions

$$\pi_1(\theta), \ \ldots, \ \pi_T(\theta)$$

SMC-ABC

Particle filter methods can also be adapted for use in an ABC setting.

Start with the sequential importance sampler of Del Moral et al. (2006) which aims to sample a selection of N particles successively from a sequence of distributions

$$\pi_1(\theta), \ \ldots, \ \pi_T(\theta)$$

At stage t we aim to find a weighted cloud of particles

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

which approximates π_t, i.e.,

$$\pi_t(\theta) \approx \sum_{i=1}^{N} w_i \delta_{\theta_i}(\theta)$$

where $\delta(\cdot)$ is the Dirac delta function.

SMC-ABC

Particle filter methods can also be adapted for use in an ABC setting.

Start with the sequential importance sampler of Del Moral et al. (2006) which aims to sample a selection of N particles successively from a sequence of distributions

$$\pi_1(\theta), \ \ldots, \ \pi_T(\theta)$$

At stage t we aim to find a weighted cloud of particles

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

which approximates π_t, i.e.,

$$\pi_t(\theta) \approx \sum_{i=1}^{N} w_i \delta_{\theta_i}(\theta)$$

where $\delta(\cdot)$ is the Dirac delta function.

π_1 will typically be an easy distribution to sample from (the prior say) and we progress down (think temperature) until we reach the target distribution $\pi_T(\theta) = \pi(\theta|\mathcal{D})$.

Note that in the original conception of the particle filter the sequence of distributions is the sequence of filtering distributions

$$\begin{aligned}
\pi_1 &= \pi(x_1|y_1) \\
&\vdots \\
\pi_T &= \pi(x_1, \ldots, x_T|y_1, \ldots, y_T)
\end{aligned}$$

Note that in the original conception of the particle filter the sequence of distributions is the sequence of filtering distributions

$$\begin{aligned} \pi_1 &= \pi(x_1|y_1) \\ &\vdots \\ \pi_T &= \pi(x_1, \ldots, x_T|y_1, \ldots, y_T) \end{aligned}$$

This is **not** what is being done here. The sequence π_i is an arbitrary sequence of distributions choosen to provide an easy path to the target $\pi(\theta|\mathcal{D})$.

Note that in the original conception of the particle filter the sequence of distributions is the sequence of filtering distributions

$$\begin{aligned}\pi_1 &= \pi(x_1|y_1)\\ &\vdots\\ \pi_T &= \pi(x_1,\ldots,x_T|y_1,\ldots,y_T)\end{aligned}$$

This is **not** what is being done here. The sequence π_i is an arbitrary sequence of distributions choosen to provide an easy path to the target $\pi(\theta|\mathcal{D})$.
To adapt this to an ABC setting, we decide upon a sequence of tolerances

$$\delta_1 > \delta_2 > \ldots > \delta_T$$

and let π_t be the ABC distribution found by the ABC algorithm when we use tolerance δ_t.

- Think of the sequence π_i as a sequence of heated posteriors.

Toni *et al.* (2008)

Assume we have a cloud of weighted particles $\{(\theta_i, w_i)\}_{i=1}^{N}$ that were accepted at step $t-1$.

1. Sample θ from the previous population according to the weights.
2. Perturb the particles according to perturbation kernel q_t. I.e.,
$$\tilde{\theta} \sim q_t(\theta, \cdot)$$
3. Reject particle immediately if $\tilde{\theta}$ has zero prior density, i.e., if
$$\pi(\tilde{\theta}) = 0$$
4. Otherwise simulate $X \sim f(\tilde{\theta})$ from the simulator. If $\rho(S(X), S(\mathcal{D})) \leq \delta_t$ accept the particle, otherwise reject.
5. Give the accepted particle weight
$$w_i = \frac{\pi(\tilde{\theta})}{\sum_{\theta_i} q_t(\theta_i, \tilde{\theta})}$$
6. Repeat steps 1-5 until we have N accepted particles at step t.

SMC-ABC: Van der Pol results

The hope is that by moving a cloud of particles about in space, we can get away with many fewer model evaluations.

After each stage the resampling and the perturbation allow us to refresh our sample to look only at particles that are doing well.

SMC-ABC: Van der Pol results

The hope is that by moving a cloud of particles about in space, we can get away with many fewer model evaluations.

After each stage the resampling and the perturbation allow us to refresh our sample to look only at particles that are doing well.

For the Van der Pol problem, we tried a sequence of tolerances

$$\delta_1 = 10, \;\; 5, \;\; 2.5, \;\; 1.75, \;\; 1, \;\; 0.5 = \delta_T$$

and used a Gaussian random walk perturbation kernel

$$q_t(\theta, \cdot) \sim N(\theta, \Sigma_t)$$

SMC-ABC: Van der Pol results

The hope is that by moving a cloud of particles about in space, we can get away with many fewer model evaluations.

After each stage the resampling and the perturbation allow us to refresh our sample to look only at particles that are doing well.

For the Van der Pol problem, we tried a sequence of tolerances

$$\delta_1 = 10, \;\; 5, \;\; 2.5, \;\; 1.75, \;\; 1, \;\; 0.5 = \delta_T$$

and used a Gaussian random walk perturbation kernel

$$q_t(\theta, \cdot) \sim N(\theta, \Sigma_t)$$

We estimated the step-size using the empirical variance of the previous population

$$\Sigma_t = \text{diag}(\mathbb{V}\text{ar}(\theta_{\cdot}^{t-1}))/8$$

-nb only sensible if $\pi(\theta|\mathcal{D})$ is uni-modal.

Using the number of crossings and the initial conditions as the summary, with the same metric as before, we found we only needed 71179 model evaluations to find 1000 particles within a distance of 0.5 away from the observation.

With plain rejection we had only found 57 particles of this closeness after 10^5 model runs.

Even with no tuning of the delta-schedule or the update kernel, SMC-ABC is 25 times more efficient than rejection ABC.

Using the number of crossings and the initial conditions as the summary, with the same metric as before, we found we only needed 71179 model evaluations to find 1000 particles within a distance of 0.5 away from the observation.

With plain rejection we had only found 57 particles of this closeness after 10^5 model runs.

Even with no tuning of the delta-schedule or the update kernel, SMC-ABC is 25 times more efficient than rejection ABC.

A draw-back of SMC-ABC as opposed to rejection based ABC is that we can't use the trick of choosing the metric after the event, and so must decide on a good summary and metric before we start.

Likelihood function as a summary

Instead of an ad-hoc summary statistic (such as the number of crossings), we can use the likelihood function as the summary of the trajectory.

$$S(X) = \log \prod \pi(D_t|X_t)$$

where

$$\pi(D_t|X_t) \propto \exp(-(D_t - X_t)/\sigma^2)$$

is a Gaussian pdf representing the measurement error process (I have assumed we only observe x).

Likelihood function as a summary

Instead of an ad-hoc summary statistic (such as the number of crossings), we can use the likelihood function as the summary of the trajectory.

$$S(X) = \log \prod \pi(D_t|X_t)$$

where

$$\pi(D_t|X_t) \propto \exp(-(D_t - X_t)/\sigma^2)$$

is a Gaussian pdf representing the measurement error process (I have assumed we only observe x).

Note that because of the assumption of Gaussian observation errors this is equivalent to minimizing the sum of squares differences between the data and the trajectory.

Likelihood function as a summary

Instead of an ad-hoc summary statistic (such as the number of crossings), we can use the likelihood function as the summary of the trajectory.

$$S(X) = \log \prod \pi(D_t|X_t)$$

where

$$\pi(D_t|X_t) \propto \exp(-(D_t - X_t)/\sigma^2)$$

is a Gaussian pdf representing the measurement error process (I have assumed we only observe x).

Note that because of the assumption of Gaussian observation errors this is equivalent to minimizing the sum of squares differences between the data and the trajectory.

As before I use

$$\rho(X, D) = |S(X) - S(D)|$$

$\sim$ a warmed likelihood - similar to annealing/tempering?

SMC-ABC tau1 posterior - warning!

SMC-ABC alpha posterior

SMC-ABC x_0 posterior

SMC-ABC y_0 posterior

Best fit $\theta = (1.005,\ 4.935,\ 1.026,\ 0.765)$

A quick aside on implicit models

The methods examined so far have the advantage that

- they do not require knowledge of the model equations (including the MCMC scheme)
- they will work if the model is replaced by a stochastic model

If we are in the nice situation where

- we know the model equations

$$\dot{x} = f(x, u, t|\theta)$$

- the model is deterministic

then other methods may give better performance.

- Langevin and Hamiltonian MCMC.
- Ramsay *et al.* 2007 - ad-hoc parameter scheme (fits spines to the trajectories, calculates derivatives and uses these to estimate parameters) seems remarkably efficient on small problems.

A quick aside on implicit models

Rao-Blackwell type conjecture: If part of the likelihood function is available, it is more efficient to use this than to simulate from it in most inference schemes.

For example, suppose we have the calibration model

$$y = f(\theta)$$
$$\mathcal{D} = y + e$$

where e has a known distribution, but $f(\theta)$ can only be simulated from. Then I suspect it is always better to use a partial likelihood method, rather than adding a simulation of e to the simulation $f(\theta)$.

A quick aside on implicit models

Rao-Blackwell type conjecture: If part of the likelihood function is available, it is more efficient to use this than to simulate from it in most inference schemes.

For example, suppose we have the calibration model

$$y = f(\theta)$$
$$\mathcal{D} = y + e$$

where e has a known distribution, but $f(\theta)$ can only be simulated from. Then I suspect it is always better to use a partial likelihood method, rather than adding a simulation of e to the simulation $f(\theta)$.

Consequence: when using emulators (in an MCMC scheme say) it is better to use the likelihood description they provide rather than drawing samples from them.

Stochastic Dynamics

Statistically, we are in a static model setting as given θ, the trajectory is fully specified and we aren't trying to do state estimation.

For this reason, the Del Moral SIS filter does not suffer from the problems of particle coalescence that make simultaneous state and parameter estimation so difficult.

Stochastic Dynamics

Statistically, we are in a static model setting as given θ, the trajectory is fully specified and we aren't trying to do state estimation.

For this reason, the Del Moral SIS filter does not suffer from the problems of particle coalescence that make simultaneous state and parameter estimation so difficult.

In models with stochastic dynamics where we want to also perform state estimation, ie, find

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

we face a harder challenge.
We can't use the Del Moral filter with $\pi_t = \pi(\theta, x_{0:t}|y_{0:t})$ because we can't perturb θ between steps.

One possibility is to embed a data-assimilation scheme within the SMC algorithm where we carefully watch the data-assimilation and stop if it is clear that the particle will be rejected.

So what are we doing when we use ABC?

Wilkinson 2008

Approximate Rejection Algorithm

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

- What is the approximation?
  - We wish to solve $\mathcal{D} = f(\theta)$.
  - Accepted θ are not from $\pi(\theta|\mathcal{D}, f)$, but from some approximation to it.
- How do we choose
  - distance measure $\rho(\cdot, \cdot)$
  - tolerance δ
  - summary statistic $S(\cdot)$, etc?

The error in ABC

Approximate Rejection Algorithm

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

It is possible to show that output from this algorithm is an exact draw from the posterior when we assume that the measurement is made in the presence of a uniform additive error term.

$$D = f(\theta) + \epsilon$$

or more generally it introduces a distribution $\pi(D|X)$ relating the simulator output to the observation.

The error in ABC

Approximate Rejection Algorithm

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

It is possible to show that output from this algorithm is an exact draw from the posterior when we assume that the measurement is made in the presence of a uniform additive error term.

$$D = f(\theta) + \epsilon$$

or more generally it introduces a distribution $\pi(D|X)$ relating the simulator output to the observation.
If $\rho(x, y) = |x - y|$, then this is equivalent to assuming uniform error on $[-\delta, \delta]$. Accepted θ are from the posterior

$$\pi(\theta|D, f, \epsilon \sim U[-\delta, \delta])$$

The error in ABC

Approximate Rejection Algorithm

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

It is possible to show that output from this algorithm is an exact draw from the posterior when we assume that the measurement is made in the presence of a uniform additive error term.

$$D = f(\theta) + \epsilon$$

or more generally it introduces a distribution $\pi(D|X)$ relating the simulator output to the observation.
If $\rho(x, y) = |x - y|$, then this is equivalent to assuming uniform error on $[-\delta, \delta]$. Accepted θ are from the posterior

$$\pi(\theta|D, f, \epsilon \sim U[-\delta, \delta])$$

ABC gives 'exact' inference under a different model!

A general error structure

Suppose ϵ is distributed with density $\pi_\epsilon(\cdot)$. We can modify the ABC rejection algorithm to give perform inference from the model $D = f(\theta) + \epsilon$ where we now control the distribution of the error ϵ.

A general error structure

Suppose ϵ is distributed with density $\pi_\epsilon(\cdot)$. We can modify the ABC rejection algorithm to give perform inference from the model $D = f(\theta) + \epsilon$ where we now control the distribution of the error ϵ.

Generalized ABC

- Draw $\theta \sim \pi(\theta)$
- Simulate X from model $X \sim f(\theta)$
- Accept θ with probability $r = \frac{\pi_\epsilon(D-X)}{c}$

Here, c is a constant chosen to maximise the acceptance probability, and guarantee $r \leq 1$. Typically, $c = \pi_e(0)$ is the best we can do.

A general error structure

Suppose ϵ is distributed with density $\pi_\epsilon(\cdot)$. We can modify the ABC rejection algorithm to give perform inference from the model $D = f(\theta) + \epsilon$ where we now control the distribution of the error ϵ.

Generalized ABC

- Draw $\theta \sim \pi(\theta)$
- Simulate X from model $X \sim f(\theta)$
- Accept θ with probability $r = \frac{\pi_\epsilon(D-X)}{c}$

Here, c is a constant chosen to maximise the acceptance probability, and guarantee $r \leq 1$. Typically, $c = \pi_e(0)$ is the best we can do.

Proposition

Accepted θ are samples from the posterior distribution $\pi(\theta|D, \epsilon \sim \pi_\epsilon)$ where $D = f(\theta) + \epsilon$.

A general error structure

Suppose ϵ is distributed with density $\pi_\epsilon(\cdot)$. We can modify the ABC rejection algorithm to give perform inference from the model $D = f(\theta) + \epsilon$ where we now control the distribution of the error ϵ.

Generalized ABC

- Draw $\theta \sim \pi(\theta)$
- Simulate X from model $X \sim f(\theta)$
- Accept θ with probability $r = \frac{\pi_\epsilon(D-X)}{c}$

Here, c is a constant chosen to maximise the acceptance probability, and guarantee $r \leq 1$. Typically, $c = \pi_e(0)$ is the best we can do.

Proposition

Accepted θ are samples from the posterior distribution $\pi(\theta|D, \epsilon \sim \pi_\epsilon)$ where $D = f(\theta) + \epsilon$.

This imples that using the 0-1 cutoff we have been using corresponds to assuming a uniformly distributed error term.

Choosing discrepancies

Using ABC is equivalent to adding additional variability into the model.

- $\exists$ many interesting papers saying how to make this variability small
- Instead ask, given that we are stuck with this additional variability, can we use it in a useful manner, or if not, how can we make sure it does little harm?

Choosing discrepancies

Using ABC is equivalent to adding additional variability into the model.

- $\exists$ many interesting papers saying how to make this variability small
- Instead ask, given that we are stuck with this additional variability, can we use it in a useful manner, or if not, how can we make sure it does little harm?

How can we choose a distribution for ϵ?

- Let ϵ be measurement/sampling error on D
  - Measurement error may be built into models. Could we remove it from the simulations and use the ABC to do this step?

Choosing discrepancies

Using ABC is equivalent to adding additional variability into the model.

- $\exists$ many interesting papers saying how to make this variability small
- Instead ask, given that we are stuck with this additional variability, can we use it in a useful manner, or if not, how can we make sure it does little harm?

How can we choose a distribution for ϵ?

- Let ϵ be measurement/sampling error on D
  - Measurement error may be built into models. Could we remove it from the simulations and use the ABC to do this step?
- Let ϵ be the discrepancy between the model and reality
  - In a deterministic model setting, Goldstein and Rougier 2008, and Kennedy and O'Hagan (2001) (amongst others), have offered advice for thinking about model discrepancies.
  - In a stochastic model setting, what the model error represents is much less clear.

Comments

- Prior specification of the distribution of the model error is hard!

Comments

- Prior specification of the distribution of the model error is hard!
- For many models, the variance of the model and measurement error may not be large enough to make the ABC acceptance rate high enough to generate any successes. This is telling us something has been mis-specified and we need to rethink the choices we have made.

Comments

- Prior specification of the distribution of the model error is hard!
- For many models, the variance of the model and measurement error may not be large enough to make the ABC acceptance rate high enough to generate any successes. This is telling us something has been mis-specified and we need to rethink the choices we have made.
- It can help us interpret our results, and highlight where we may being doing something undesirable with the modeling or inference.

Comments

- Prior specification of the distribution of the model error is hard!
- For many models, the variance of the model and measurement error may not be large enough to make the ABC acceptance rate high enough to generate any successes. This is telling us something has been mis-specified and we need to rethink the choices we have made.
- It can help us interpret our results, and highlight where we may being doing something undesirable with the modeling or inference.
- It can help guide our choice of metric. We don't expect any model to perfectly fit the data, and ABC can be viewed as adding in enough variability to allow a fit to be found. Given that we are adding in variability, we can control where it is placed.

Comments

- Prior specification of the distribution of the model error is hard!
- For many models, the variance of the model and measurement error may not be large enough to make the ABC acceptance rate high enough to generate any successes. This is telling us something has been mis-specified and we need to rethink the choices we have made.
- It can help us interpret our results, and highlight where we may being doing something undesirable with the modeling or inference.
- It can help guide our choice of metric. We don't expect any model to perfectly fit the data, and ABC can be viewed as adding in enough variability to allow a fit to be found. Given that we are adding in variability, we can control where it is placed.
- ABC can also be viewed as smoothing the model pdf through a kernel defined by the metric:

$$\pi(\mathcal{D}|\theta) = \int \pi(D|X)\pi(X|\theta)\mathrm{d}\theta$$

Comments

- Prior specification of the distribution of the model error is hard!
- For many models, the variance of the model and measurement error may not be large enough to make the ABC acceptance rate high enough to generate any successes. This is telling us something has been mis-specified and we need to rethink the choices we have made.
- It can help us interpret our results, and highlight where we may being doing something undesirable with the modeling or inference.
- It can help guide our choice of metric. We don't expect any model to perfectly fit the data, and ABC can be viewed as adding in enough variability to allow a fit to be found. Given that we are adding in variability, we can control where it is placed.
- ABC can also be viewed as smoothing the model pdf through a kernel defined by the metric:

$$\pi(\mathcal{D}|\theta) = \int \pi(D|X)\pi(X|\theta)\mathrm{d}\theta$$

- There exist generalised versions of ABC-MCMC and ABC-SMC (Wilkinson, forthcoming).

History-matching

With thanks to Michael Goldstein

1. Design an ensemble of model runs $\{\theta\}_{i=1}^{N}$ and build a second-order emulator of the model $f(\theta)$

$$f(\theta) = \mathbb{E}f(\theta) + \delta(\theta)$$

where $\mathbb{E}f(\theta)$ is the posterior expectation of our belief about the simulators value when run at θ. δ is the residual with $\mathbb{E}(\delta(\theta)) = 0$, and is $\mathbb{V}\text{ar}(\delta(\theta))$ determined by our prior and data.

History-matching

With thanks to Michael Goldstein

1. Design an ensemble of model runs $\{\theta\}_{i=1}^{N}$ and build a second-order emulator of the model $f(\theta)$

$$f(\theta) = \mathbb{E}f(\theta) + \delta(\theta)$$

where $\mathbb{E}f(\theta)$ is the posterior expectation of our belief about the simulators value when run at θ. δ is the residual with $\mathbb{E}(\delta(\theta)) = 0$, and is $\mathbb{V}\text{ar}(\delta(\theta))$ determined by our prior and data.

2. Relate the model to the system

$$y = f(\theta) + \epsilon$$

where ϵ is our model discrepancy (beliefs specified by a mean and covariance function)

History-matching

With thanks to Michael Goldstein

1. Design an ensemble of model runs $\{\theta\}_{i=1}^{N}$ and build a second-order emulator of the model $f(\theta)$

$$f(\theta) = \mathbb{E}f(\theta) + \delta(\theta)$$

   where $\mathbb{E}f(\theta)$ is the posterior expectation of our belief about the simulators value when run at θ. δ is the residual with $\mathbb{E}(\delta(\theta)) = 0$, and is $\mathbb{V}\text{ar}(\delta(\theta))$ determined by our prior and data.
2. Relate the model to the system

$$y = f(\theta) + \epsilon$$

   where ϵ is our model discrepancy (beliefs specified by a mean and covariance function)
3. Relate the system to the model (e represents measurement error)

$$\mathcal{D} = y + e$$

History-matching

With thanks to Michael Goldstein

1. Design an ensemble of model runs $\{\theta\}_{i=1}^{N}$ and build a second-order emulator of the model $f(\theta)$

$$f(\theta) = \mathbb{E}f(\theta) + \delta(\theta)$$

   where $\mathbb{E}f(\theta)$ is the posterior expectation of our belief about the simulators value when run at θ. δ is the residual with $\mathbb{E}(\delta(\theta)) = 0$, and is $\mathbb{V}\text{ar}(\delta(\theta))$ determined by our prior and data.
2. Relate the model to the system

$$y = f(\theta) + \epsilon$$

   where ϵ is our model discrepancy (beliefs specified by a mean and covariance function)
3. Relate the system to the model (e represents measurement error)

$$\mathcal{D} = y + e$$

This gives us a second-order specification of the relationship between the emulator mean $\mathbb{E}f(\theta)$ and the data $\mathcal{D}$.

History-matching 2

Craig *et al.* 1996

4 We then say θ is implausible if, say

$$\| \mathcal{D} - \mathbb{E}f(\theta) \| > 3\sigma$$

where σ^2 is the combined variance implied by the emulator, discrepancy and measurement error.

History-matching 2

Craig *et al.* 1996

4 We then say θ is implausible if, say

$$\| \mathcal{D} - \mathbb{E}f(\theta) \| > 3\sigma$$

where σ^2 is the combined variance implied by the emulator, discrepancy and measurement error.

If θ is not implausible we don't discard it. The result is a region of space that we can't rule out at this stage of the history-match.

Note, we might go through several stages of history matching, where we focus on an ever smaller region of space and build a more careful emulator on this region.

Note also that the result of a history-match (unlike the result of a likelihood-based MCMC calculation) may be that there is no not-implausible region of parameter space.

Relationship between ABC and history-matching

There is clearly a relationship between generalised ABC and history-matching. The main difference seems to be that history-matching replaces the Monte Carlo simulation in the ABC with an emulation step.

Relationship between ABC and history-matching

There is clearly a relationship between generalised ABC and history-matching. The main difference seems to be that history-matching replaces the Monte Carlo simulation in the ABC with an emulation step.

- If it is possible to build a good emulator, history-matching is likely to be much more computationally efficient (also looking only at second-order moments simplifies the calculations).
- If we can't build an emulator then it means the problem is inherently hard and we're forced back into repeatedly simulating.
- Stochastic models may prove particularly hard to emulate.

Relationship between ABC and history-matching

There is clearly a relationship between generalised ABC and history-matching. The main difference seems to be that history-matching replaces the Monte Carlo simulation in the ABC with an emulation step.

- If it is possible to build a good emulator, history-matching is likely to be much more computationally efficient (also looking only at second-order moments simplifies the calculations).
- If we can't build an emulator then it means the problem is inherently hard and we're forced back into repeatedly simulating.
- Stochastic models may prove particularly hard to emulate.

History-matching is a second-order method $\Rightarrow$ less modelling assumptions need to be made. Although we could of course write down a second-order version of ABC.

Similarly to ABC, there is also a less principled version of history-matching where instead of a careful specification of model and measurement error we just seek to meet some tolerance requirement for the difference between the simulator prediction and the data (note the different roots of the two methods).

Conclusions

Approximate Bayesian Computation gives exact inference for the wrong model.

- To move beyond inference conditioned on a perfect model hypothesis, we should account for model error.
- ABC algorithms can be considered as adding additional variability on to the model outputs.
- If done wisely, ABC can be viewed not as an approximate form of Bayesian inference, but instead as coming closer to the inference we want to do.
- For complex models and complex data, tuning to a low dimensional summary of the data (one we believe the model should be able to predict) may be the best we can hope for.

Conclusions

Approximate Bayesian Computation gives exact inference for the wrong model.

- To move beyond inference conditioned on a perfect model hypothesis, we should account for model error.
- ABC algorithms can be considered as adding additional variability on to the model outputs.
- If done wisely, ABC can be viewed not as an approximate form of Bayesian inference, but instead as coming closer to the inference we want to do.
- For complex models and complex data, tuning to a low dimensional summary of the data (one we believe the model should be able to predict) may be the best we can hope for.

Thank you for listening!