Estimating model error in dynamic models

Richard Wilkinson[1] and Jeremy Oakley[2]

[1]School of Mathematical Sciences, University of Nottingham
[2]Department of Probability and Statistics, University of Sheffield
r.d.wilkinson@nottingham.ac.uk

UCM conference July 2010

All models are wrong, but ...

Lets acknowledge that most models are imperfect.

All models are wrong, but ...

Lets acknowledge that most models are imperfect. Consequently,

- predictions will be wrong, or will be made with misleading degree of confidence
- solving the inverse problem $y = f(\theta) + e$ may not give sensible results.
    - e is measurement error
    - $f(\theta)$ is our computer model
    - y is our data

All models are wrong, but ...

Lets acknowledge that most models are imperfect. Consequently,

- predictions will be wrong, or will be made with misleading degree of confidence
- solving the inverse problem $y = f(\theta) + e$ may not give sensible results.
    - e is measurement error
    - $f(\theta)$ is our computer model
    - y is our data

Can we

- account for the error?
- correct the error?

All models are wrong, but ...

Lets acknowledge that most models are imperfect. Consequently,

- predictions will be wrong, or will be made with misleading degree of confidence
- solving the inverse problem $y = f(\theta) + e$ may not give sensible results.
  - e is measurement error
  - $f(\theta)$ is our computer model
  - y is our data

Can we

- account for the error?
- correct the error?

Kennedy and O'Hagan (2001) suggested we introduce reality ζ into our statistical inference

- Reality $\zeta = f(\hat{\theta}) + \delta$, the best model prediction plus model error $\delta(x)$.
- Data $y = \zeta + e$ where e represents measurement error

Dynamic models

- For dynamical systems the model sequentially makes predictions before then observing the outcome.
- Embedded in this process is information about how well the model performs for a single time-step.
- We can specify a class of models for the error, and then try to learn about the error from our predictions and the realised data.

Mathematical Framework

Suppose we have

- State vector x_t which evolves through time. Let $x_{0:T}$ denote $(x_0, x_1, \dots, x_T)$.

Mathematical Framework

Suppose we have

- State vector x_t which evolves through time. Let $x_{0:T}$ denote $(x_0, x_1, \dots, x_T)$.
- Computer model f which encapsulates our beliefs about the dynamics of the state vector

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

which depends on forcings u_t. We treat f as a black-box.

Mathematical Framework

Suppose we have

- State vector x_t which evolves through time. Let $x_{0:T}$ denote $(x_0, x_1, \dots, x_T)$.
- Computer model f which encapsulates our beliefs about the dynamics of the state vector

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

  which depends on forcings u_t. We treat f as a black-box.
- Observations

$$y_t = h(x_t)$$

  where $h(\cdot)$ usually contains some stochastic element

Moving from white to coloured noise

A common approach is to treat the model error as white noise

- State evolution: $x_{t+1} = f(x_t, u_t) + \epsilon_t$ where ϵ_t are iid rvs.

Moving from white to coloured noise

A common approach is to treat the model error as white noise

- State evolution: $x_{t+1} = f(x_t, u_t) + \epsilon_t$ where ϵ_t are iid rvs.

Instead of the white noise model error, we ask whether there is a stronger signal that could be learnt:

- State evolution: $x_{t+1} = f(x_t, u_t) + \delta(x_t, u_t) + \epsilon_t$
- Observations: $y_t = h(x_t)$.

Moving from white to coloured noise

A common approach is to treat the model error as white noise

- State evolution: $x_{t+1} = f(x_t, u_t) + \epsilon_t$ where ϵ_t are iid rvs.

Instead of the white noise model error, we ask whether there is a stronger signal that could be learnt:

- State evolution: $x_{t+1} = f(x_t, u_t) + \delta(x_t, u_t) + \epsilon_t$
- Observations: $y_t = h(x_t)$.

Our aim is to learn a functional form plus stochastic error description of δ

Why this is difficult?

- $x_{0:T}$ is usually unobserved, but given observations $y_{0:T}$ and a fully specified model we can infer $x_{0:T}$.
  - the filtering/smoothing problem
- When we want to learn the discrepancy $\delta(x)$ we are in the situation where we estimate δ from $x_{0:T}, \ldots$
- but we must estimate $x_{0:T}$ from a description of δ.

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

$$\theta^{(m+1)} = \text{arg max}_\theta Q(\theta, \theta^{(m)})$$

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

$$\theta^{(m+1)} = \text{arg max}_\theta Q(\theta,\theta^{(m)})$$

  - 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 $\theta^{(0)}, \theta^{(1)}, \dots$ 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 $\mathrm{argmax}_\theta \pi(y_{0:t}|\theta)$.

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
     $$\tilde{w}^{(i)} \propto \pi(y_t|\tilde{x}_t^{(i)}) \qquad = \phi(y_t; x_t, \sigma_{obs}^2)$$
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 $\{x_t^{(i)}\}_{i=1,\dots,N,t=1,\dots,T}$ 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 $\sigma_{obs} = 1m$

Comments on results

- We have learnt the discrepancy (a function of v) using only observations on x.
- Fitting higher order regression terms we find similar results over the range of interest (although parameters are not necessarily well identified).
- Larger measurement errors give much less reliable results - sometimes leading to misleading statements of accuracy.
- 500 iterations of EM is overkill! Many fewer would suffice.
- Using an adaptive scheme for the number of filtering and smoothing particles could improve accuracy and efficiency.
- Tend to see estimates of slope converging rapidly, but estimates of error variance taking a long time to decrease.

Gaussian Processes

We can use the same ideas, but replace the parametric model by a non-parametric GP model.

Algorithm Summary

A heuristic algorithm for learning $\delta(\cdot)$

1. Using the white noise discrepancy model, draw sample trajectories $x_{0:T}^{(j)}$ from $\pi(x_{0:T}|y_{0:T})$.
2. Using these realizations, estimate values of $\delta_1(\cdot)$ and fit a Gaussian process model for δ_1.
3. At stage m, use discrepancy δ_m to sample from $\pi(x_{0:T}|y_{0:T}, \delta_m)$.
4. Use realizations $x_{0:T}^{(j)}$ from step 3 to estimate δ_{m+1}:

$$\delta_{m+1}(x_t^{(j)}) = x_{t+1}^{(j)} - f_\phi(x_t^{(j)})$$

5. Fit a GP model to these data. Return to step 3.

Sequence of GP discrepancy estimates

Toy 1D model

Identifying potential for learning δ

It can be hard to discover whether it is worth departing from a white noise model for δ. How can we assess whether there is a functional form δ for the error that improves upon white noise?

If we plot $y_{t+1} - f(y_t)$ against y_t, it can look like white noise is a good model for the error.

If we know x_0 without error, we might try plotting $y_{t+1} - f(x_t)$ vs x_t where x_t is a trajectory simulated from the model. But this can be shown to look like white noise even for very simple models.

Looking at $\tilde{x}_{t+1} - f(\tilde{x}_t)$ where $\tilde{x}_t$ is an estimate of the true trajectory (a realization from $\pi(x_{0:T}|y_{0:T})$) can help

- but this requires a model for the error (white noise?)
- and an estimation algorithm

and still doesn't usually show any pattern.

Concluding remarks

- Using a functional model discrepancy can improve forecasts and state estimates. The discrepancy can be learnt from observations.
- Approach is computationally intensive and can be unstable. Even for the toy gravity model, 100 iterations of the algorithm can take several minutes.
- Sequential approaches are extremely costly, which is why we've used a batch approach here.
- If the modellers have beliefs about the shape of the model error, it is possible to incorporate this into our *a priori* description of the GP model.
- The stochastic EM algorithm can be made more efficient by increasing the number of Monte Carlo samples (thus reducing the MC error) as we iterate through the EM algorithm.
- Simultaneous discrepancy estimation and (computer) model parameter estimation is a hard problem.
  - Intuition suggests a carefully restricted model for δ would be necessary.

Thank you for listening!