

On-line Inference for Data Streams

Statistics Department

Oxford University

email: clifford@stats.ox.ac.uk

Data streams

Massive data streams are now common in many data processing applications, with rapid accumulation of substantial datasets. For example in

- monitoring and examining Internet traffic,
- recording and interpreting customer transactions (loyalty cards)
- analysing high-frequency financial data in market trading.
- voice and video capture.
- data logging in many areas of scientific enquiry.

Inference

Markov Chain Monte Carlo (MCMC) methods revolutionised statistics in the 1990s by providing practical, computationally feasible access to the flexible and coherent framework of Bayesian inference.

However, massive datasets have produced difficulties for these methods since, with a few simple exceptions, MCMC implementations require a complete scan of what might be several gigabytes of data at each iteration of the MCMC algorithm.

The development of practical Bayesian methodology for the analysis of massive datasets is one of the outstanding challenges of modern statistics.

Bayesian Analysis of a Statistical Model

- Data: x
- Parameters: θ , functions of interest $g(\theta)$
- Likelihood: $L(\theta, x)$
- Prior density: $\pi(\theta)$

The prior density and the likelihood are used to calculate integrals of the form

$$\frac{\int g(\theta)\pi(\theta)L(\theta, x)d\theta}{\int \pi(\theta)L(\theta, x)d\theta} = \int g(\theta)\pi(\theta|x)d\theta$$
$$= E\{g(\theta)|x\} \text{ (posterior expectation)}$$

Numerical Methods: MCMC

Construct a Markov chain, using (say) the Metropolis algorithm, to obtain successive values

$$\theta^1, \theta^2, \dots, \theta^n$$

simulated from the equilibrium density

$$\pi(\theta|x) \propto \pi(\theta)L(\theta, x).$$

Estimate $E\{g(\theta)|x\}$ by

$$\bar{g} = \frac{\sum_{i=1}^n g(\theta^i)}{n}.$$

If $\{\theta^i\}$ were independent then $\text{Var}(\bar{g}) = \sigma_g^2/n$.

Typically $\text{Var}(\bar{g}) = \tau\sigma_g^2/n$ where $\tau > 1$.

τ = correlation time

n/τ = ‘effective sample size’

Suppose now that the observational framework expands, giving additional data and an expanded parameter set.

- Data: x, x^+
- Parameters: θ, θ^+ , functions of interest $g(\theta, \theta^+)$
- Likelihood: $L(\theta, \theta^+; x, x^+)$
- Joint prior density: $\pi(\theta)\pi(\theta^+|\theta)$

Now we need to simulate from $\pi(\theta, \theta^+|x, x^+)$ (*)

Question:

- Can we use the simulations from $\pi(\theta|x)$?
- Or do we have to start completely afresh and sample from (*) using MCMC, etc?

Evolving datasets (signal processing)

Data: noisy observations of an underlying process (the signal)

Data expand with time $(x_1, \dots, x_t) = \mathbf{x}_{1:t}$.

Parameters: history of the underlying process

Parameters expand with time $(\theta_1, \dots, \theta_t)$.

Objectives:

- *Filtering*: maintain knowledge about the current state θ_t , for example to allow estimation of $E\{g(\theta_t)|\mathbf{x}_{1:t}\}$.
- *Smoothing*: the aim is to reconstruct past states of the system by using all the data which have been accumulated so far.
- *Prediction*: make predictions about the state of the system in the future

In many applications, filtering is the key problem, since it forms the basis of prediction and control.

For simplicity *Structural Assumptions* are made about the evolving data set.

- $\pi(\theta_1, \dots, \theta_t) = \pi(\theta_1)\pi(\theta_2|\theta_1)\dots\pi(\theta_t|\theta_{t-1})$
(underlying state is Markov)
- $L(\theta_1, \dots, \theta_t; \mathbf{x}_{1:t}) = \prod_{k=1}^t h(x_k|\theta_k)$
(current observations depend only on the current state).

In general, the underlying state process will depend on unknown parameters that must be incorporated into a full Bayesian model.

The Markov assumption is not severely restrictive and the observational assumptions can be relaxed.

With these assumptions, the current state of knowledge can be updated by

$$\pi(\theta_{t+1}|\mathbf{x}_{1:t}) = \int \pi(\theta_{t+1}|\theta_t)\pi(\theta_t|\mathbf{x}_{1:t})d\mu(\theta_t) \quad (1)$$

$$\pi(\theta_{t+1}|\mathbf{x}_{1:t+1}) = \frac{h(x_{t+1}|\theta_{t+1})\pi(\theta_{t+1}|\mathbf{x}_{1:t})}{p(x_{t+1}|\mathbf{x}_{1:t})} \quad (2)$$

where

$$p(x_{t+1}|\mathbf{x}_{1:t}) = \int h(x_{t+1}|\theta_{t+1})\pi(\theta_{t+1}|\mathbf{x}_{1:t})d\mu(\theta_{t+1}). \quad (3)$$

The first integral is crucial. If θ_t is high-dimensional, evaluation of this integral at each stage will present problems.

Special Case: If there is a linear Gaussian model for the evolution of the underlying state and if the noise is Gaussian, the integrals can be evaluated explicitly. The posterior distributions then turn out to be Gaussian too. This is the basis of the Kalman filter (which basically just updates the means and covariances of the state θ_t .)

Extension: In many practical applications, these assumptions are implausible. In particular, the observation process will often be non-linear. An alternative approach in such cases is the extended Kalman filter (EKF), in which the updated measurements are linearised about the predicted state, permitting the Kalman filter to be applied approximately.

Other approximate methods involve

- approximating distributions by mixtures of Gaussians: the Gaussian sum filter
- approximating the first two moments of the density
- evaluating the required probability density function over a grid in the state space

However, none of these techniques represents a universal algorithm, since they each have to be extensively modified to tackle the problem in hand. For example, methods that evaluate the probability density over a grid in the state space first require the grid to be specified, which is a non-trivial problem in a multi-dimensional space. To avoid misleading results, a large number of grid points will in general be necessary. In addition, a non-trivial computation must be performed at each point.

Sequential Monte Carlo (Particle filters)

Recall that the current state of knowledge is updated by

$$\pi(\theta_{t+1}|\mathbf{x}_{1:t}) = \int \pi(\theta_{t+1}|\theta_t)\pi(\theta_t|\mathbf{x}_{1:t})d\mu(\theta_t) \quad (4)$$

$$\pi(\theta_{t+1}|\mathbf{x}_{1:t+1}) = \frac{h(x_{t+1}|\theta_{t+1})\pi(\theta_{t+1}|\mathbf{x}_{1:t})}{p(x_{t+1}|\mathbf{x}_{1:t})} \quad (5)$$

where

$$p(x_{t+1}|\mathbf{x}_{1:t}) = \int h(x_{t+1}|\theta_{t+1})\pi(\theta_{t+1}|\mathbf{x}_{1:t})d\mu(\theta_{t+1}). \quad (6)$$

We need a way of carrying out these integrals successively for $t = 1, 2, \dots$

Sampling Importance Resampling: Poor Man's Bayes

Rubin devised a simple way of obtaining an approximate sample from a Bayesian posterior distribution.

- Simulate a sample $\tilde{\theta}^1, \tilde{\theta}^2, \dots, \tilde{\theta}^n$ from $\pi(\theta)$.
- Calculate weights $q_i \propto L(\tilde{\theta}^i, x)$; $\sum q_i = 1$
- Sample n times (with replacement) from the discrete θ -distribution with

$$\mathbf{P}(\theta = \tilde{\theta}^i) = q_i.$$

The resulting sample $\theta^1, \theta^2, \dots, \theta^n$ is an “approximate” sample from $\pi(\theta|x)$.

Convergence in probability

The sample obtained is approximate in the sense that

$$\frac{\sum_{i=1}^n g(\theta^i)}{n} \xrightarrow{P} \mathbf{E} \{g(\theta)|x\},$$

as $n \rightarrow \infty$.

The *Sampling Importance Resampling (SIR)* filter (Gordon, Salmond, Smith; Kitagawa; Isard and Blake) is based on Rubin's sampler.

The SIR filter proceeds as follows. Assume that you have a sample $(\theta_t^i)_{i=1,\dots,n}$ from $\pi(\theta_t | \mathbf{x}_{1:t})$:

- (a) **Prediction:** Independently simulate $\tilde{\theta}_{t+1}^i$, using the state transition density $\pi(\theta_{t+1} | \theta_t^i)$, for each $i = 1, \dots, n$,
- (b) **Likelihood:** Upon receipt of observation x_{t+1} , for each value $\tilde{\theta}_{t+1}^i$ calculate the corresponding likelihood $h(x_{t+1} | \theta_{t+1}^i)$. Denote the set of likelihood values, normalised to sum to 1, by $(q_{t+1}^i)_{i=1,\dots,n}$.
- (c) **Update:** Draw a random sample of size n from the discrete distribution taking values $(\tilde{\theta}_{t+1}^i)_{i=1,\dots,n}$ with probabilities $(q_{t+1}^i)_{i=1,\dots,n}$. This is an approximation to a sample from $\pi(\theta_{t+1} | \mathbf{x}_{t+1})$.

The algorithm can be thought of as propagating a swarm of particles in the underlying state space.

At time t the particles are assumed to be an approximate sample from the posterior distribution of θ_t , given the observations so far.

At time $t + 1$ each particle moves to a new location in the state space. The likelihood of this location given x_{t+1} is evaluated and a multinomial sample of particles is then drawn from the discrete distribution with *support points* given by the *particle locations* and *probabilities* proportional to the *likelihoods*.

The process is a form of *Genetic Algorithm* where the ‘fitness’ of a speculative parameter value is proportional to its likelihood.

In its simplest form the SIR filter has various weaknesses.

Outliers

The effect of an outlying observation is to produce a likelihood which is centered in the tail of the prior distribution. Since this tail is represented only sparsely by sample points in the SIR filter, an exceptionally large sample from the prior will be needed to yield a good support for the posterior distribution.

Sample Impoverishment

- lack of diversity: particles may be highly correlated, localised into a restricted region of parameter space, acting as one
- the particle system may collapse to a singleton (extreme lack of diversity)

Track loss

- Particles become trapped in 'impossible' regions of state space (evolutionary dead-ends)

There are various solutions to these problems.

Jittering

Jittering smoothes out the posterior density, using a Gaussian kernel.

Choosing the jitter variance is thus equivalent to choosing the smoothing parameter in density estimation, and there is a corresponding variance/bias tradeoff to be made. Standard rules of thumb can be used to choose the degree of smoothing.

Prior Boosting

Another approach to sample depletion was originally proposed by Rubin, 1987. At the prediction stage of the SIR filter, instead of generating the usual n points, we generate κn points. The likelihood of each of these is calculated, and then n are resampled in the update step in the usual way. Typically $\kappa = 10$.

Random measures

Particle filters work by providing a discrete approximation to the PDF which can be easily updated to incorporate new information as it arrives. More generally our interest will be in approximations which consist of a set of random locations in the state space $(s^i)_{i=1,\dots,n}$, termed the *support*, and a set of associated weights $(m^i)_{i=1,\dots,n}$ summing to 1. The support and the weights together form a *random measure*.

The objective is to choose measures so that

$$\sum_{i=1}^n g(s^i) m^i \approx \int g(\theta) \pi(\theta) d\mu(\theta) \quad (7)$$

for typical functions g of the state space, in the sense that the left-hand side converges (in probability) to the right-hand side as $n \rightarrow \infty$.

Reducing randomness

It is worth emphasising that our fundamental objective is to produce accurate Monte Carlo approximations to the succession of integrals which arise in Bayesian calculations. For accurate Monte Carlo integration, it is essential to eliminate unnecessary randomness and to make careful choices for proposals in importance sampling. The randomness introduced by multinomial sampling can be reduced by systematic sampling

When to resample

This raises the question, when should we resample, and when should we carry forward the weights? The question has been addressed by Liu and Chen (1996, 1998) who propose an *ad hoc* rule based on the variance of the weights $(\tilde{m}_t^i)_{i=1,\dots,n}$.

Tempered weights

An intermediate position is to resample using modified weights: for example, weights proportional to the square root of the original, i.e. $w_t^i \propto \sqrt{m_t^i}$. The resampled points are then carried forward with weights proportional to m_t^i/w_t^i . Similar techniques have been proposed in MCMC sampling to avoid problems in sampling from highly peaked densities.

Better proposals - MCMC

Although it is unrealistic to use MCMC to sample the posterior distribution of the complete state history, under certain circumstances MCMC moves can be introduced in particle filtering. These moves may be successful in preventing sample impoverishment. To accommodate arbitrary transitions it is necessary to store the whole history of the process up to time t .

This can be avoided if the transition kernel only depends on a fixed set of summary statistics, or only upon the last τ time points.

Example: Bearings only tracking

A classic example of non-linear filtering is *bearings only tracking*. An observer (either fixed or moving) observes the bearing of a moving ship. The bearing is the angle of the observation relative to a fixed direction.

The crucial problem is that the observation is a non-linear function of the underlying position of the ship. This type of tracking problem usually causes difficulties for the Extended Kalman Filter.

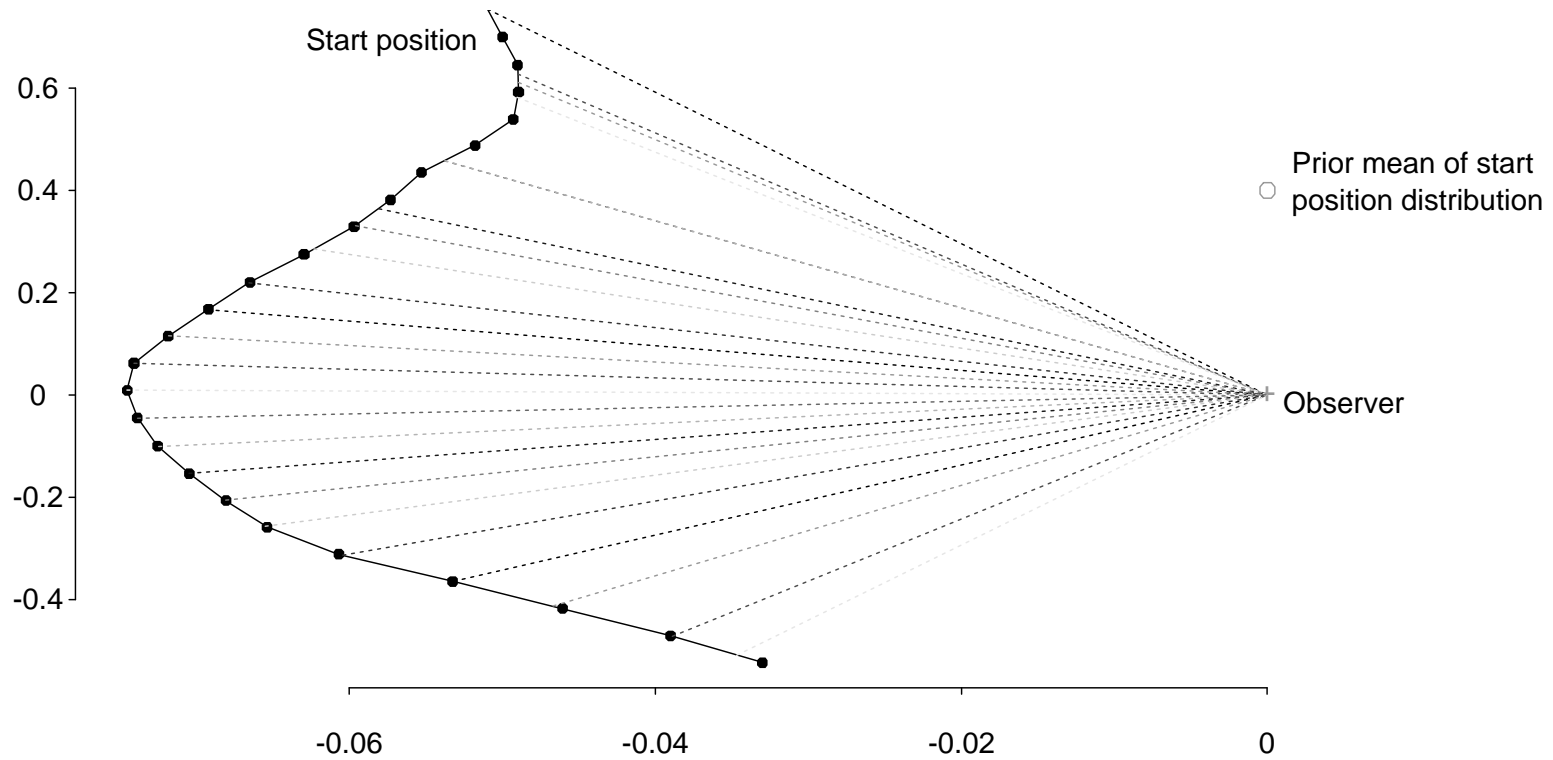


Figure 1: Typical simulated trajectory. Dotted lines show observed bearings

We want to reconstruct the trajectory given the system model, observed bearings and a prior distribution on the initial position and velocity. In particular, suppose we observe t bearings.

It turns out that scaling the track by multiplying each θ_t by a constant λ does not affect the likelihood since none of the angles change. It affects some of the parameters in a simple way. These factors can be incorporated into the filter by extending the *signature* of each particle.

The MCMC scale move, when it is made, is a Gibbs move sampling from a truncated Gamma distribution.

Example: Well-log data

Well-logs are records of the physical and mineralogical characteristics of underground rocks in a region of geological interest. In traditional applications, a probe (called a sonde) is lowered into an existing well-bore by a cable, and acoustical, electrical, nuclear-magnetic or thermal properties of the surrounding rock types are recorded as the sonde descends.

The measurements are of nuclear magnetic response taken at 4500 time points. The underlying signal is piecewise constant; each constant segment relating to a stratum of a single rock type with constant physical properties. The jump discontinuities in the signal occur at times when a new rock stratum is first met.

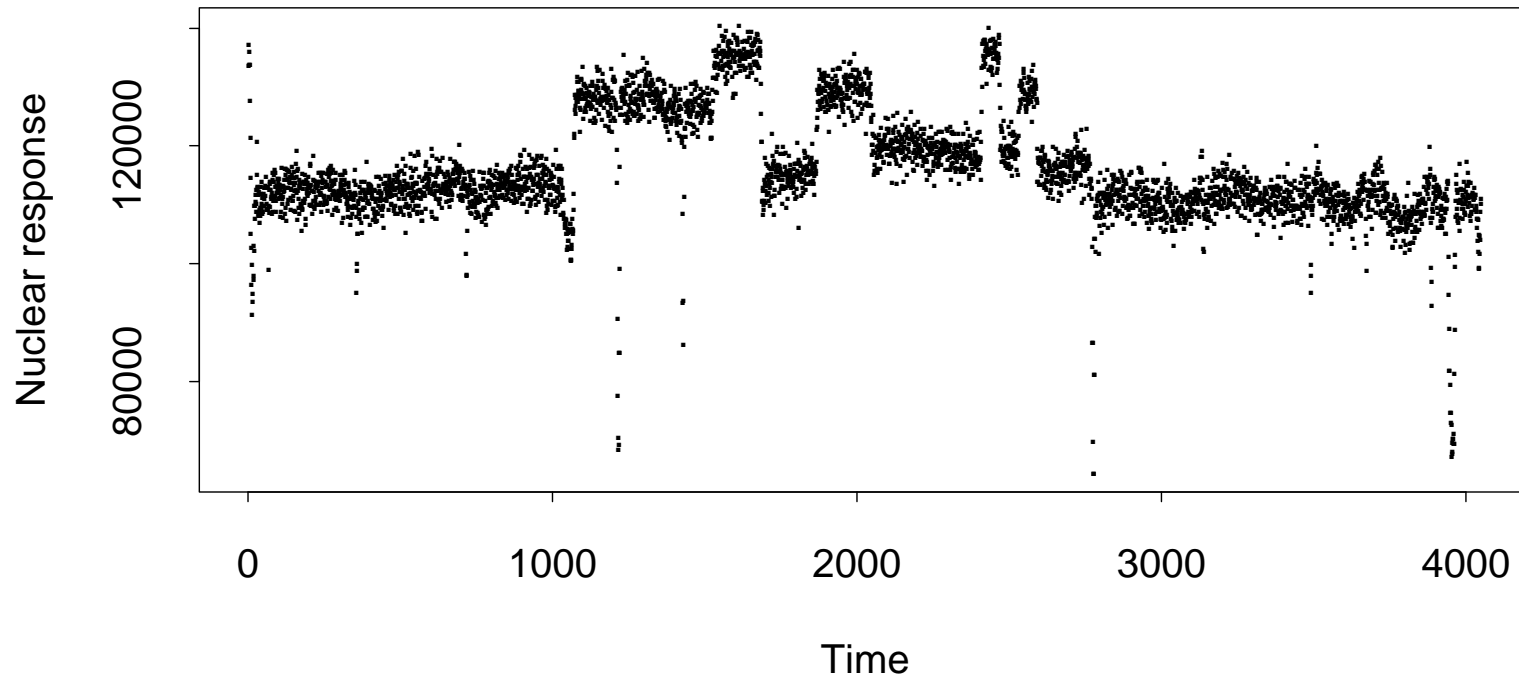


Figure 2: The measurements are of nuclear magnetic response taken at 4500 time points.

Measurement while drilling

There are severe technical difficulties both in obtaining useful measurements and in the transmission of these data to the surface.

Progress is currently being made with these problems in the gas and oil drilling industry.

Other areas in which the use of traditional sondes is inadequate, include the exploration of leakage below buried waste. These investigations are carried out by horizontal drilling making it impossible to lower a sonde into the borehole. Attachment of recording devices to the drill head may be the only way in which data can be collected – thus enabling drilling to be steered towards areas of high contamination.

Batch processing

When batch processing the complete well-log data of Figure 2 (for example, by eye) the detection of change-points appears straightforward.

The on-line problem is made difficult by clusters of outliers in the data.

Differentiating between these outliers and the true change-points which is difficult, especially as the detection of each change-point needs to be made very soon after it occurs.

Markov chain Monte Carlo (MCMC) methods are inappropriate for such on-line problems, because of the need for real-time inference.

Sequential Monte Carlo

We use a hidden Markov model to model regime switching in the well-log data. The (measurable) state is the expected nuclear magnetic response for the current rock strata. The hidden Markov chain allows for both changes in the rock strata, and the possibility that the current measurements are outliers.

The conjugacy in the model we use means that conditional on knowing the history of the hidden Markov chain, the posterior distribution of the history of the measurable state can be calculated analytically using the Kalman filter.

The posterior distribution can be written as a mixture distribution, with each term in the mixture referring to a single possible value of the history of the hidden Markov chain. Liu and Chen (2000) show that for such problems, the efficiency of the particle filter can be greatly improved if, instead of each particle representing a possible value of the history of the state, each particle represents a possible history of the hidden Markov chain (or a suitable summary of that history). This technique is called *marginalisation* or *collapsing*.

With such an approach, the posterior can be calculated exactly using a finite number of particles.

Unfortunately, the number of particles needs to increase exponentially with the number of measurements, and becomes infeasibly large for even small data sets (let alone the data set shown in Figure 2, where there are 4050 measurements).

To restrict the number of particles used by the particle filter, resampling must be used. (At each time stage a smaller, but hopefully representative, sample of particles are chosen from the large number of current particles.)

Previous analyses

The well-log data of Figure 2 has been analysed previously by Fitzgerald et al (1996). They batch-processed the data using a Gibbs sampler.

Outliers were removed by hand, before the data was analysed, and the number of change-points was fixed prior to the analysis.

In contrast, in the sequential analysis using particle filter, both dealing with outliers, and allowing for an unknown number of change-points, is incorporated automatically.

A statistical model for the well-log data

We assume a two-dimensional Hidden Markov Model, with states $I_t = (S_t, O_t)$, where S_t and O_t both taking values in $\{1, 2\}$.

Conditional on I_t , the measurable state (the expected nuclear magnetic response) satisfies

$$X_t = \begin{cases} X_{t-1} & \text{if } S_t = 1, \\ \mu + \sigma Z_t & \text{if } S_t = 2. \end{cases}, \quad (8)$$

and the measurements satisfy

$$Y_t = \begin{cases} X_t + \tau_1 Z_t^* & \text{if } O_t = 1, \\ \nu + \tau_2 Z_t^* & \text{if } O_t = 2. \end{cases} \quad (9)$$

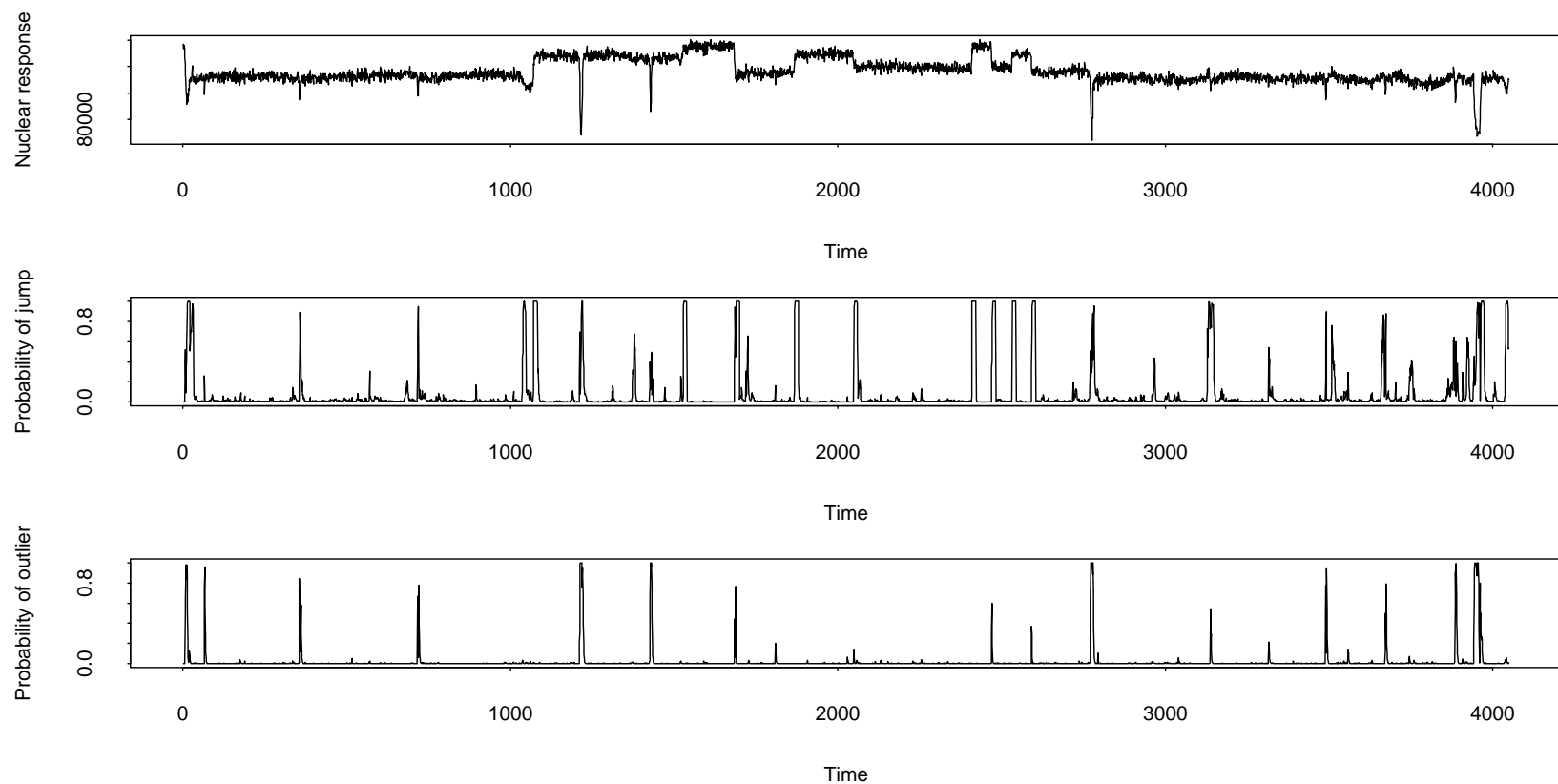


Figure 3: Results of on-line analysis of the well-log data (top) by the new particle filter. The particle filter used 100 particles. The estimates of the probabilities of a recent change-point (middle), and the probability of the measurement being an outlier (bottom) are both shown.

An easy evaluation of the performance of the filter can be gained from looking at an estimate of the underlying signal.

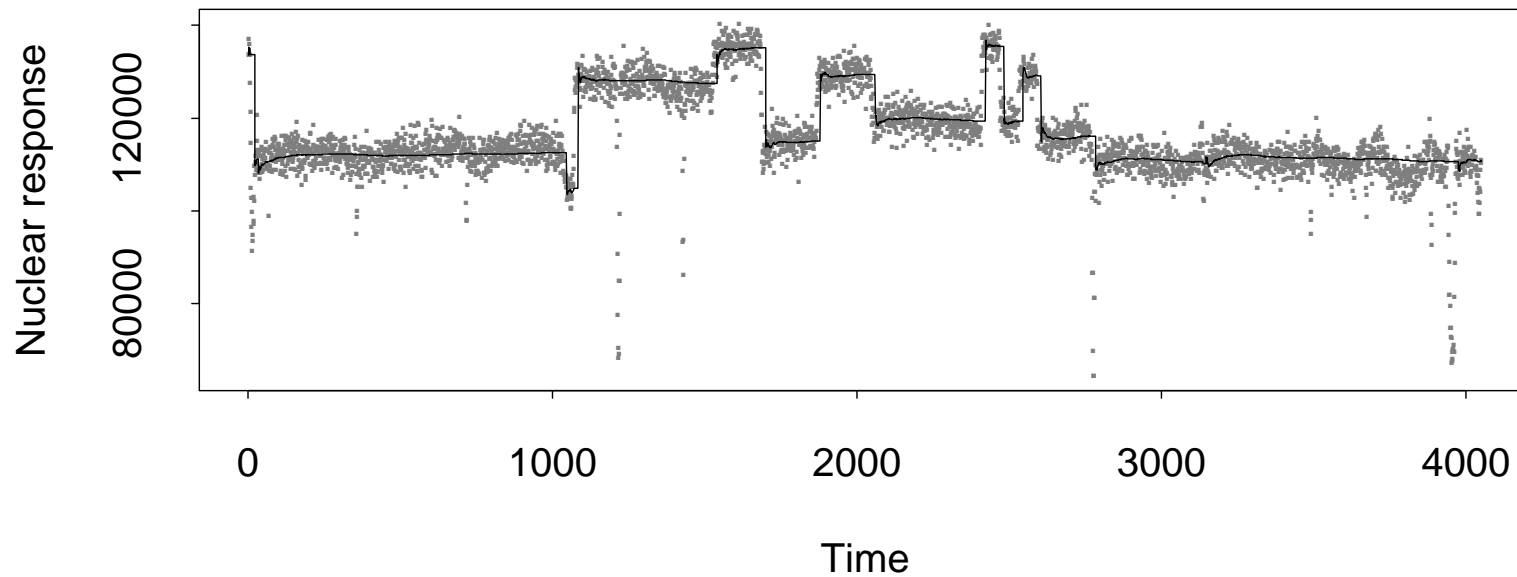


Figure 4: An estimate of the underlying signal for the well-log data.

How the estimate was obtained

The estimate was obtained from the output of the particle filter. The change-points were fixed to be at times where the posterior probability of a recent jump was greater than 0.9, of which there were 16.

The value of the state between each pair of adjacent change-points was estimated by the mean of all measurements in that time period which had negligible probability of being an outlier.

Example: Tracking in clutter

An important class of problems concerns the detection of targets in the presence of clutter. The data consist of false responses and target responses, without distinction. The objective is to identify and track the target.

For illustration Figure 5 shows a 1000 simulated observations with a single target and 7 false responses. The position of target X_t is given by

$$X_t = \begin{cases} X_{t-1} + W_t & \text{where } W_t \sim N(0, \sigma_W^2) & \text{small change regime} \\ V_t & \text{where } V_t \sim N(0, \sigma_V^2) & \text{large change regime} \end{cases}, \quad (10)$$

where regime switches are determined by a hidden Markov model with switching parameter p_J ,

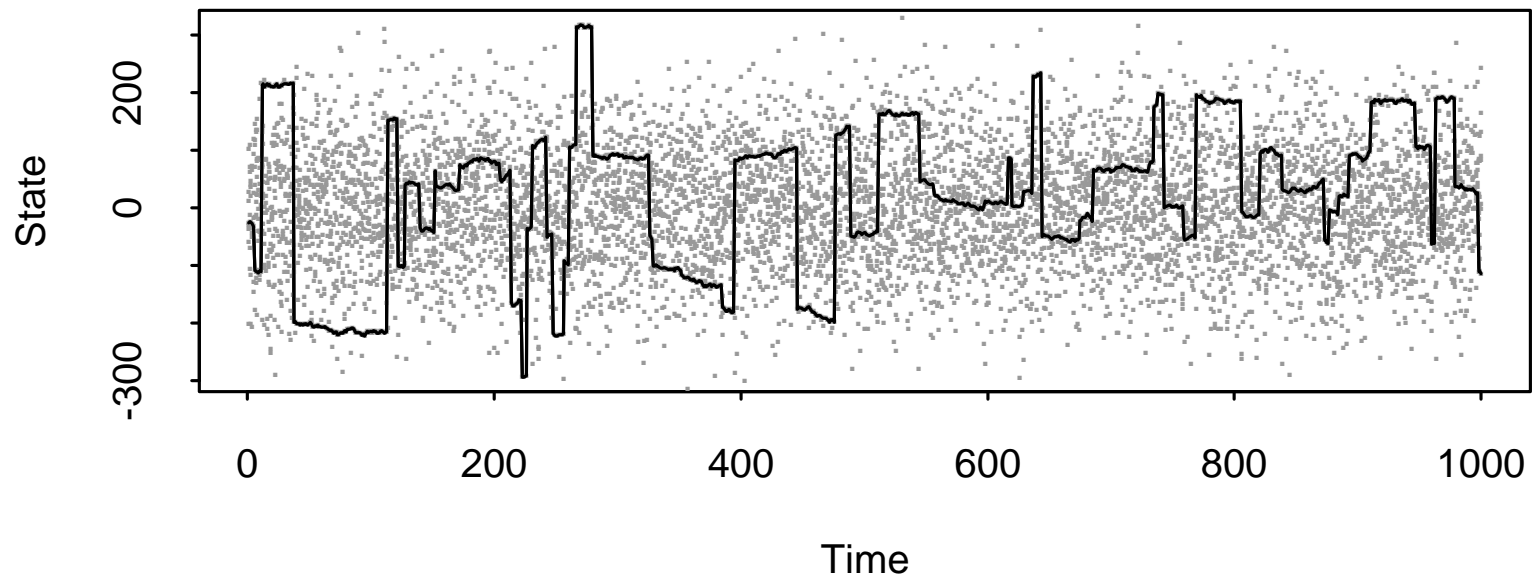


Figure 5: Target tracking in clutter: solid line is the true target position, grey dots represent observations and clutter.

Evolving datasets – Particle filters

In Bayesian statistical analysis, our aim is to find the posterior density $\pi(\theta|y_{1:N})$ of the parameter given the data $y_{1:N} = \{y_1, \dots, y_N\}$. The parameter θ may be of high dimension $\theta = (\theta_1, \dots, \theta_p)$.

In a standard non-dynamic (static) problem all the data $y_{1:N}$ are available at once, and we know that

$$\pi(\theta|y_{1:N}) \propto \pi(\theta) f(y_{1:N}; \theta)$$

where $f(y_{1:N}; \theta)$ is the likelihood.

Markov chain Monte Carlo is one method of analysis. To use it we need to construct a Markov chain on the space of possible θ values with $\pi(\theta|y_{1:N})$ as its equilibrium. By running the chain for a long period of time, values from the equilibrium can be harvested and used to summarise the target distribution.

Problems

Suppose for example that the Metropolis sampler is used.

At each step t the current value of $\theta^{(t)}$ is modified by proposing a new value, θ , sampled from a proposal density $g(\theta|\theta^{(t)})$.

The new value θ is accepted and becomes $\theta^{(t+1)}$ with acceptance probability

$$A(\theta^{(t)}, \theta; y_{1:N}) = \min \left\{ 1, \frac{g(\theta^{(t)}|\theta)\pi(\theta|y_{1:N})}{g(\theta|\theta^{(t)})\pi(\theta^{(t)}|y_{1:N})} \right\}.$$

The problem is that that $A(\theta^{(t)}, \theta; y_{1:N})$ depends on the whole of $y_{1:N}$.

When the data set is massive, computing the acceptance probability is a non-trivial calculation, since it involves scanning through each of the data values. When N is of the order of millions this can be a very time consuming task, and furthermore this task has to be repeated until the MCMC algorithm has converged, which may take several thousand steps.

Applications of this size are common in data-mining. For example: records of all transactions (items purchased by individuals etc) are recorded by supermarkets; records of web-site access are maintained by internet service providers.

Bayesian analysis of massive data sets

1. A sequential particle filter method for static problems: N. Chopin
2. Bayesian analysis of massive datasets via particle filters: G. Ridgeway and D. Madigan
3. Particle filters for mixture models with an unknown number of components: P. Fearnhead

The first two papers tackled the general problem of dealing with large datasets using a sub-sampling approach.

The last paper is a special application dealing with the “model-based clustering”.

Simple use of sub-sampling

The basic idea is to use a sub-sample of the data $y_{1:n}$ where $n \ll N$.

If n is small enough then MCMC can be run on the subsample, to yield $\{\theta_i\}, i = 1, \dots, M$, a sample of values from $\pi(\theta|y_{1:n})$.

Each of these values then receives an importance weight w_i from the rest of the sample, given by

$$w_i = \frac{\pi(\theta_i|y_{1:N})}{\pi(\theta_i|y_{1:n})}.$$

Independent observations

A simplification occurs when observations are independent, since

$$\frac{\pi(\theta|y_{1:N})}{\pi(\theta|y_{1:n})} = \frac{\pi(\theta)f(y_{1:N};\theta)}{\pi(\theta)f(y_{1:n};\theta)} = f(y_{n+1:N};\theta).$$

Similar simplifications occur when the observations have Markov dependence.

The practical impact is that the dataset only needs to be scanned MN times.

Sub-sampling: improvements

In the first two papers, various improvements are proposed. The basic idea is to consider a succession of values of n , say n_1, n_2, \dots, N and to apply the particle filters to the successively augmented datasets, pretending that these form a time series.

A general particle filter has two components, sampling/resampling and moving. The move steps are MCMC transitions designed to refresh the particle support set.

The decision on when to move can be based on the distribution of particle weights. If the distribution is highly skewed then moves should be made.

These MCMC steps are computationally expensive, but as the data are successively augmented, the distribution of particle weights is expected to become less skewed. So moves are made less often.

Evolving datasets – Data sketching

1. Stable distributions [...] embedding and data stream computation. P. Indyk
2. An improved data stream summary [...]: G. Cormode and Muthukrishnan
3. Probabilistic counting algorithms [...]: P. Flajolet and G.N. Martin

Currently, there is considerable excitement at the prospect of developing methods for reducing large datasets to small but informative “sketches”.

The first two papers exploit an ingenious combination of random projections (using stable law distributions) and universal hashing to produce sketches of large datasets that enable l_p norm queries to be answered rapidly. The third paper is a careful analysis of an ingenious way of counting large numbers with a tiny amount of memory. This is related to the Additive-increase multiplicative-decrease processes studied by Bertoin-Biane-Yor.

Data sketching

The basic ingredients:

- A data stream of items with types in some set A .
- A pseudo-random mapping $h : A \rightarrow R$ such that

$$P(h(a) < x) = F_p(x),$$

where F_p is the distribution function of a symmetric stable distribution with parameter p , and where

$$P(|h(a) - h(b)| < \epsilon) = O(\epsilon), a \neq b.$$

(Universal hash function)

The data comes in a stream, $(a_1, w_1), (a_2, w_2), \dots$ where a_i is the type of the i th item in the stream and w_i is the multiplicity.

How do you find out how many different types there are? How many different words are there in Shakespeare?

Making the Sketch

Calculate

$$S = \sum_{i=1}^n h(a_i)w_i = \sum_{j=1}^m x_j \sum_{i:h(a_i)=x_j} w_i = \sum_{j=1}^m x_j c_j.$$

Now using the property of stable distributions

$$S \sim X \sum_{j=1}^m |c_j|^p, \text{ where } X \text{ has a stable distribution with parameter } p.$$

By replicating the process, we can use the median of a sample of S values to estimate the scaling term $\sum_{j=1}^m |c_j|^p$. When p is small this gives an estimate of the number of distinct items.

Extras:

- You can compare two streams (just subtract the sketches).
- You can update sketches as new data arrive (“stock control”)

Concluding remarks

The Bayesian analysis of massive datasets remains a challenging problem.

MCMC methods are not feasible for these datasets.

Particle filters are promising. They are particularly effective when

- distributional conjugacy can be exploited
- sufficient statistics are available, permitting occasional MCMC moves to be made at low computational cost

For complex problems, particles need to be tagged with extensive information. The design of efficient database management systems for these data is an open problem.

Data sketches have the potential for summarising both the data and the particle systems that represent the posterior distribution.