Univariate Polynomial Inference by Monte Carlo Message Length Approximation

Leigh J. Fitzgibbon LEIGHF@CSSE.MONASH.EDU.AU
David L. Dowe DLD@CSSE.MONASH.EDU.AU
Lloyd Allison LLOYD@CSSE.MONASH.EDU.AU
School of Computer Science and Software Engineering, Monash University, Clayton, VIC 3800 Australia

Abstract

We apply the *Message from Monte Carlo* (MMC) algorithm to inference of univariate polynomials. MMC is an algorithm for point estimation from a Bayesian posterior sample. It partitions the posterior sample into sets of regions that contain similar models. Each region has an associated message length (given by Dowe's MMLD approximation) and a point estimate that is representative of models in the region. The regions and point estimates are chosen so that the Kullback-Leibler distance between models in the region and the associated point estimate is small (using Wallace's FSMML Boundary Rule). We compare the MMC algorithm's point estimation performance with Minimum Message Length [12] and Structural Risk Minimisation on a set of ten polynomial and non-polynomial functions with Gaussian noise. The orthonormal polynomial parameters are sampled using reversible jump Markov chain Monte Carlo methods.

1. Introduction

The Minimum Message Length (MML) principle is an invariant Bayesian point estimation technique based on information theory. The basic tenet of MML is that the hypothesis (or model or distribution) which allows for the briefest encoding of the data in a two-part message is the best explanation of the data. This view of inference as a coding process leads to discretisation of the hypothesis space and a *message length*, which gives an objective means to compare hypotheses. The message length is a quantification of the trade-off between model complexity and goodness of fit that was first described by [10]. Since their seminal paper various approximations and derivations have appeared under the veil of Minimum Message Length (MML) [12, 11] or Minimum Description Length (MDL) [6].

MML87 [12] has recently been applied to the problem of model selection in univariate polynomial regression with normally distributed noise by [8]. It was compared against Generalised Cross-Validation (GCV), Finite Prediction Error (FPE), Schwartz's Criterion (SCH), and VC Dimension and Structural Risk Minimisation (SRM) [7] over a set of polynomial and non-polynomial target functions, and for a range of noise levels. The criterion used to judge the performance of the different methods was Squared Prediction Error (SPE). In their results, the MML method had the smallest average SPE of all the methods for all target functions and noise levels. SRM was close behind. The other methods - GCV, FPE and SCH - had squared prediction error in the order of 100 times that of MML and SRM.

The Message from Monte Carlo (MMC) algorithm has been recently described in [1]. It uses the posterior as an importance sampling density to implicitly define uncertainty regions, for which message lengths can be approximated using Monte Carlo methods. Here, we apply MMC to univariate polynomial regression using discrete orthonormal polynomials and compare it with the two best contenders from [8]: MML87 and SRM.

The MMC algorithm, Dowe's MMLD approximation and Wallace's FSMML Boundary Rule are given in Section 2. The polynomial model and Bayesian priors are given in Section 3. A reversible jump Markov chain Monte Carlo algorithm for sampling polynomials is described in Section 4. Experimental evaluation is contained in Section 5 and the conclusion in Section 6.

2. Message from Monte Carlo (MMC)

In this section we describe the Message from Monte Carlo (MMC) algorithm. Many of the mathematical

results for this algorithm are omitted due to space; see [1] for more information. The algorithm approximates the length of an optimal two-part message that would, in principle, allow for the transmission of some observed data over a noiseless coding channel. The first part of the message contains a code-word that represents a model and the second part of the message contains the data encoded using the stated model.

We use a Bayesian setting where the sender and receiver agree on a prior distribution $h(\theta)$ and likelihood function $f(x|\theta)$ over the parameter space, Θ, and data space, X. We use the same notation for a vector of values as we do for a single value (i.e. θ and x can be vector values). When we talk about regions, or uncertainty regions, of the parameter space, we mean arbitrary sets of elements from the parameter space (i.e., $R \subseteq \Theta$). The parameter space can be a union of subspaces of different dimension.

The MMC algorithm uses Dowe's MMLD message length approximation - described in [5] - to calculate message lengths. For some arbitrary set of models from the parameter space, R, MMLD approximates the length of the first part of the message as the negative log integral of the prior over R, and the length of the second part as the expected value (with respect to the prior), over R, of the negative log-likelihood. This gives rise to an MMLD message length of

$$-\log\left(\int_R h(\theta)\,d\theta\right) - \frac{\int_R h(\theta)\cdot\log f(x|\theta)\,d\theta}{\int_R h(\theta)\,d\theta} \qquad (1)$$

The objective of the MMC algorithm is to find the region, R, of the parameter space which minimises the MMLD message length expression. To do so, we need to be able to compute the necessary integrals in Equation 1 accurately and efficiently over a possibly high-dimensional space. The MMC solution is based on posterior sampling and Monte Carlo integration.

The optimal region of the parameter space will consist of models with high posterior probability. We therefore draw a sample $S = \{\theta_t : t = 1, ..., N\}$ from the posterior distribution of the parameters θ using the appropriate sampler for some N. The sample can contain models of variable dimension.

The uncertainty region, R, can be implicitly defined by choosing a subset of the sample, $Q \subseteq S$, where we consider Q to be a subset of the true (explicit) region, $R \supseteq Q$. Q is a finite set and we can therefore approximate the terms in the MMLD message length equation (Equation 1) using Monte Carlo integration. These terms require a sample from the prior distribution of the parameters. For example, the expectation in the second part uses the prior, and the integral in the first part can be approximated by the proportion of the sample that falls within the uncertainty region, Q. However, we are sampling from the posterior because we expect the optimal uncertainty region to consist of models with high posterior probability. We use the posterior as an importance sampling density [2] and we weight the sample so that it is distributed as if being generated from the prior.

Converting the MMLD message length expression into its equivalent (unnormalised) probability (by taking $e^{-MessLen}$ of Equation 1) we get

$$\left(\int_R h(\theta)\,d\theta\right) \times \prod_{\theta\in R} f(x|\theta)^{\frac{h(\theta)}{\int_R h(\theta')\,d\theta'}} \qquad (2)$$

It can be seen that the region that minimises the MMLD message length is the region that maximises the product of the integral of the prior probability of models in the region and the weighted geometric mean of the likelihood (weighted by the prior). As the sample size increases, the uncertainty region will typically become smaller and converge to the posterior mode (MAP) estimate. So, while we wish to compute expectations over the uncertainty region weighted by the prior, we know that the uncertainty region will contain models with high posterior probability. For this reason, importance sampling will allow us to evaluate the sums in Equations 3 and 4, even for large amounts of observed data and where the likelihood function is very concentrated. Or, to put it another way, when the importance sampling density is very concentrated, it is concentrated in the area where we wish to do the Monte Carlo integration.

We expect (but do not require) that the prior will integrate to unity. Therefore, the length of the first part of the MMLD message can be approximated by the negative logarithm of the proportion of the weighted sample that falls within the region

$$\text{MMC 1st part length} = -\log\left(\frac{\sum_{\theta\in Q} f(x|\theta)^{-1}}{\sum_{\theta\in S} f(x|\theta)^{-1}}\right) \qquad (3)$$

The length of the second part of the MMLD message is the expected value of $-\log f(x|\theta)$ over the region, R, with respect to the prior $h(\theta)$. This can be approximated using a Monte Carlo integration using the elements of the (weighted) sample that fall within the

region, Q

$$\text{MMC 2nd part length} = \frac{\sum_{\theta \in Q} \frac{-\log f(x|\theta)}{f(x|\theta)}}{\sum_{\theta \in Q} f(x|\theta)^{-1}} \quad (4)$$

The total MMC message length is the sum of Equations 3 and 4.

Attempting to minimise the MMLD equation analytically (Equation 1) yields the following MMLD 'Boundary Rule' (e.g., see [5])

$$-\log f(x|\theta)\Big|_{\theta \in \partial R} = 1 - \frac{\int_R h(\theta) \log f(x|\theta)\, d\theta}{\int_R h(\theta)\, d\theta} \quad (5)$$

where the boundary, ∂R, of R, is an iso-likelihood contour of f. In other words, the values of $f(x|\theta)$ and of $\log f(x|\theta)$ are constant on ∂R. The MMLD Boundary Rule states that for the region which minimises the message length, the negative log-likelihood at the boundary of R is equal to one plus the expected negative log-likelihood over the region weighted by the prior.

An iso-likelihood contour is defined by a single continuous parameter representing the log-likelihood of models on the boundary of the region, $b = -\log f(x|\theta)\Big|_{\theta \in \partial R}$. The Monte Carlo integrations of the message length terms are a discrete function of the boundary. It is therefore sufficient to consider that one or more of the models in the sample, S, lie on the boundary.

The number of possible iso-likelihood contours in the sample is equal to the number of distinct values of the likelihood that appear in the sample. If we sort the sample into descending order of likelihood - starting at the maximum - then an iso-likelihood contour is given by a boundary index into the sample vector. For example, let i denote a boundary index into the sample vector. This means that θ_i is on the iso-likelihood boundary, $\theta_i \in \partial R$, and the corresponding uncertainty region is

$$\begin{aligned} Q &= \{\theta \colon \theta \in S, -\log f(x|\theta) \leq -\log f(x|\theta_i)\} \\ &= \{\theta_0, \theta_1, ..., \theta_i\} \end{aligned}$$

since the sample has been ordered (i.e. $-\log f(x|\theta_j) \leq -\log f(x|\theta_{j+1})$).

The MMLD Boundary Rule (Equation 5) states that the negative log-likelihood of models on the boundary of the region is equal to the prior-weighted average negative log-likelihood of models in the region plus one. Therefore, by Monte Carlo integration using the elements of the (importance weighted) sample that fall into Q, the rule becomes

$$\theta \in Q \text{ iff } -\log f(x|\theta) \leq \frac{\sum_{\theta \in Q} \frac{-\log f(x|\theta)}{f(x|\theta)}}{\sum_{\theta \in Q} f(x|\theta)^{-1}} + 1 \quad (6)$$

In order to find the optimal boundary index, we start with the maximum likelihood model and continue to increase the boundary index while the boundary model's negative log-likelihood is less than one plus the average over the region. In other words, we start with $Q = \{\theta_0\}$, and end up with $Q = \{\theta_0, \theta_1, \theta_2, ..., \theta_j\}$ where $-\log f(x|\theta_{j+1})$ is greater than the average negative log-likelihood over $\theta_0, ..., \theta_j$ plus one.

This will give us an implicit region that minimises the MMLD message length. However, use of this method will build regions that do not approximate efficient code-book entries because the region will be (wholly) determined by the likelihood of the observed data (which the receiver does not know) and will (generally) be unsuitable for encoding future data. In order to approximate a decodeable, efficient code-book we must ensure that the region consists of models with similar distributions. To do so, we use Wallace's FSMML Boundary Rule [9][Chapter 4]

$$\theta \in Q \text{ iff } KL(\theta, \hat{\theta}) \leq \frac{\sum_{\theta \in Q} \frac{KL(\theta, \hat{\theta})}{f(x|\theta)}}{\sum_{\theta \in Q} f(x|\theta)^{-1}} + 1 \quad (7)$$

where $\hat{\theta}$ is the point estimate for the region, and $KL(\theta, \hat{\theta})$ is the Kullback-Leibler distance of $\hat{\theta}$ from θ. How can the FSMML Boundary Rule's dependence on the point estimate, θ, be resolved and the rule used efficiently? The approach that we take is to begin with the maximum likelihood estimate, and as we add new members to the region, we allow them to compete for the right to become the point estimate of the region. If the new member is able to encode the data that can arise from the models in the region more efficiently, on average, than any other member of the region, then it wins the right to be the point estimate. The estimate therefore converges as the uncertainty region grows.

We can now describe the MMC algorithm in full. We draw a sample $\{\theta_t : t = 1, ..., N\}$ from the posterior distribution of the parameters θ using the appropriate

sampler for some N. The sample can contain models of variable dimension. First, the sample is ordered in descending order of likelihood. We now allocate space for a boolean array that indicates whether an element of the sample has been allocated to a region. Each element of the allocated array gets initialised to false. We begin the iteration by finding the unallocated model of maximum likelihood. Since the sample has been sorted, this is simply the first unallocated element of the sample. Denote this by θ_i. We begin to grow the first region by starting with θ_i and then initialise the estimate for the region, $\hat{\theta}$, to θ_i. We also initialise variables to calculate the necessary sums in the MMC message length equation (Equations 3 and 4). We continue to increment the pointer, i, until we find the boundary b (i.e. until the MMLD Boundary Rule fails). For each value of i we also check that the model θ_i fulfils the FSMML Boundary Rule. If it does, then we flag it as being allocated to the region. We next consider it as a candidate for the point estimate. If the prior-weighted expected Kullback-Leibler distance between the models in the region and this candidate is less than the equivalent for the currently reigning estimate then the candidate becomes the point estimate.

Each time we increment the pointer, we update the length of the first part of the message, the average over the region (Length of MMC 2nd Part) and the expected Kullback-Leibler distance in constant time. Once the MMLD Boundary Rule is violated, we reconsider all unallocated models to the left (i.e. $< i$) since the estimate may have changed and some models that now deserve to be in the region may have previously been rejected on the first pass. After this second pass, we store all of the results and begin growing a new region using the first unallocated element of the sample (the model of maximum likelihood). This is repeated until all elements of the sample have been assigned to a region. Pseudo-code for the algorithm is given as Algorithm 1.

The MMC algorithm returns a set of regions with associated point estimates and message lengths. If the objective is to select a single model, then the point estimate from the region with the minimum message length is chosen. If the objective is to compute the average of some quantity of interest under the posterior, then Bayesian Model Averaging can be performed at reduced computational cost using the set of point estimates weighted using their message lengths.

The algorithm is computationally feasible because, when attempting to build the region associated with $\hat{\theta}$, the potential candidates for the region must pass the MMLD Boundary Rule. Therefore, the number of candidates presented to the FSMML Boundary Rule is considerably reduced.

When a candidate model is accepted as a member of the region, we compute the expected Kullback-Leibler distance between the existing members of the region and the candidate. This would appear to be an expensive step. However, this sum need only be carried out while it (the partial sum) is less than the expected Kullback-Leibler distance to the current point esti-

Algorithm 1 *Pseudo-code for the Message from Monte Carlo (MMC) algorithm*

sample from the posterior $S = \{\theta_t : t = 1, ..., N\}$
sort the sample such that $f(x|\theta_t) \geq f(x|\theta_{t+1})$
allocate space for the boolean 'allocated' array
initialise all elements of allocated to false
while unallocated elements remain
  $ml \leftarrow 0$
  while ($ml < N$ and allocated[ml]) $ml \leftarrow ml + 1$
  allocated[ml] $\leftarrow$ true
  $\hat{\theta} \leftarrow \theta_{ml}$
  1st length $\leftarrow -\ln \frac{f(x|\theta_{ml})^{-1}}{\sum_{\theta \in S} f(x|\theta)^{-1}}$
  2nd num $\leftarrow -f(x|\theta_{ml})^{-1} \ln f(x|\theta_{ml})$
  2nd denom $\leftarrow f(x|\theta_{ml})^{-1}$
  2nd length $\leftarrow$ 2nd num / 2nd denom
  ekl num $\leftarrow 0$
  ekl denom $\leftarrow f(x|\theta_{ml})^{-1}$
  ekl $\leftarrow$ ekl num / ekl denom
  $Q \leftarrow \{\theta_{ml}\}$
  $i \leftarrow ml + 1$
  while ($i < N$ and allocated[i]) $i \leftarrow i + 1$
  while ($i < N$)
    if ($-\ln f(x|\theta_i) >$ 2nd length + 1) break
    if ($KL(\theta_i, \hat{\theta}) \leq$ ekl + 1) then
      allocated[i] $\leftarrow$ true
      1st length $\leftarrow -\ln \left(e^{-\text{1st length}} + \frac{f(x|\theta_i)^{-1}}{\sum_{\theta \in S} f(x|\theta)^{-1}} \right)$
      2nd num $\leftarrow$ 2nd num $-f(x|\theta_i)^{-1} \ln f(x|\theta_i)$
      2nd denom $\leftarrow$ 2nd denom $+ f(x|\theta_i)^{-1}$
      2nd length $\leftarrow$ 2nd num / 2nd denom
      $Q \leftarrow Q \cup \{\theta_i\}$
      if ($\frac{\sum_{\theta \in Q} KL(\theta, \theta_i) f(x|\theta)^{-1}}{\sum_{\theta \in Q} f(x|\theta)^{-1}} <$ ekl) then
        $\hat{\theta} \leftarrow \theta_i$
        ekl num $\leftarrow \sum_{\theta \in Q} KL(\theta, \theta_i) f(x|\theta)^{-1}$
      else
        ekl num $\leftarrow$ ekl num $+ KL(\theta_i, \hat{\theta}) f(x|\theta_i)^{-1}$
      end
      ekl denom $\leftarrow$ ekl denom $+ f(x|\theta_i)^{-1}$
      ekl $\leftarrow$ ekl num / ekl denom
    end
    $i \leftarrow i + 1$
    while ($i < N$ and allocated[i]) $i \leftarrow i + 1$
  end
  store results for region Q
end

end

mate. The importance weighting gives the Kullback-Leibler distance between models with small likelihoods more weight, so the sum should be performed in ascending order of likelihood so that it (the partial sum) is likely to exceed the current EKL sooner. This strategy was found to yield an efficient algorithm in our experiments.

All that the MMC algorithm requires to operate is a sample from the posterior distribution and a function for computing the Kullback-Leibler distance. It can therefore be viewed as a posterior sampling post-processing algorithm. Markov Chain Monte Carlo (MCMC) [3] methods can be used to do the sampling.

3. Polynomial Model

We now describe the polynomial model and priors that we use in the experimental evaluation. The likelihood function for the n observations $(y_i, x_i) : i = 1..n$ is

$$f(y|a_0, ...a_d, \sigma, x) = \frac{1}{(2\pi\sigma^2)^{n/2}} e^{\frac{-1}{2\sigma^2}SE} \tag{8}$$

$$\text{where } SE = \sum_{i=1}^{n} \left(y_i - \sum_{k=0}^{d} a_k \phi_k(x_i) \right)^2 \tag{9}$$

The basis functions, ϕ's, are orthogonal with respect to the data: $\forall_{j \neq k} \sum_{i=1}^{n} \phi_j(x_i)\phi_k(x_i) = 0$ and normalised to unity $\forall_j \sum_{i=1}^{n} \phi_j(x_i)\phi_j(x_i) = 1$ (i.e. generalised Fourier series).

We use two different priors on the generalised Fourier series coefficients $(a_0, ..., a_d)$. Prior 1 is the same prior that [8] used - an (independent) Gaussian prior on each coefficient

$$h_1(a_0, ..., a_d) = \prod_{k=0}^{d} \frac{1}{\sqrt{2\pi}u} e^{-\frac{a_k^2}{2u^2}} \tag{10}$$

$$\text{where } u = \sqrt{\sum_{k=0}^{d} y_k^2/(d+2)} \tag{11}$$

The prior reflects the belief that we expect each of the coefficients and noise to (roughly) contribute to the observed variance of the y_i values equally. We note that [8] use the same prior, and that the difference between their u and our u arises because they normalise the average inner-product of their basis functions.

The second prior that we use on the coefficients is a uniform additive prior

$$h_2(a_0, ..., a_d) = \prod_{i=0}^{d} \frac{1}{4u_i} \tag{12}$$

$$\text{where } u_i = \sqrt{\sum_{k=1}^{n} y_k^2 - \sum_{j<i} a_j^2} \tag{13}$$

This reflects the belief that a coefficient could 'explain' the variance that lower order coefficients have not, and that there is no preference for a specific value over the range $[-2u_i, 2u_i]$.

We use a diffuse inverted gamma prior on σ (for conjugacy): $IGamma(0.0001, 0.0001)$, and a geometric prior on the order of the model: $Geometric(0.9)$.

4. Polynomial Sampler

To sample from the posterior we use Green's reversible jump MCMC methodology [4].

Let k_{max} denote the upper bound on the order of the polynomials under consideration. The sampler consists of three types of moves: birth to a higher dimension, death to a lower dimension and move in the current dimension. We find that the sampler mixes better when allowed to jump more than one order when a birth or death dimension change move is made. So when a birth move is performed on a k-order model, the new order is generated from a $Geometric(0.5)$ distribution which has been normalised over $\{k+1, ..., k_{max}\}$. This is mirrored for the death step over the interval $\{k-1, ..., 0\}$.

For each sweep of the sampler we randomly choose between the $k_{max}+1$ move types: $M = \{0, ..., k_{max}\}$, where $m = k$ denotes the move step (no dimension change), and $m \neq k$ denotes a jump to an m'th order polynomial. We denote the probability of each move conditional on k (the order of the last polynomial in the chain) as $p(m|k)$. The probabilities of birth, death and move are denoted by b, d, and $\nu = 1.0 - b - d$ respectively and are independent of k and chosen by the practitioner[1]. Therefore $p(m|k)$ satisfies[2] the following: $\sum_{m>k} p(m|k) = b$, $\sum_{m<k} p(m|k) = d$, and $p(m|k) = 1.0 - b - d$ for $m = k$. We now detail the samplers for each of the three move types.

[1] In our experiments we use $b = 0.2$ and $d = 0.2$.

[2] Except in the extreme cases where $k = 0$ (d is taken as zero) or $k = k_{max}$ (b is taken as zero).

4.1 Move step

For the move step the new values for the coefficients and σ are generated by Gibbs sampling. In describing the full conditional for coefficients we use the notation a_{-z} to represent the set of all coefficients except a_z (i.e. $a_{-z} = \{a_i : 0 \leq i \leq d \text{ and } i \neq z\}$).

The full conditional distribution for coefficient a_z using Prior 1 (Equation 10) is

$$p(a_z|a_{-z}, \sigma, y, x) \propto e^{-\frac{\sigma^2+u^2}{2\sigma^2u^2}(a_z^2 - \frac{u^2\sum_{i=1}^n y_i\phi_z(x_i)}{\sigma^2+u^2})^2} \quad (14)$$

a_z is normally distributed with mean $\mu' = \frac{u^2\sum_{i=1}^n y_i\phi_z(x_i)}{u^2+\sigma^2}$ and standard deviation $\sigma' = \sqrt{\frac{u^2\sigma^2}{u^2+\sigma^2}}$.

The full conditional distribution for coefficient a_z over the range $[-2u_z, 2u_z]$ using Prior 2 (Equation 12) is

$$p(a_z|a_{-z}, \sigma, y, x) \propto e^{-\frac{1}{2\sigma^2}\left(\sum_{i=1}^n y_i\phi_z(x_i) - a_z\right)^2} \quad (15)$$

a_z is normally distributed with mean $\mu' = \sum_{i=1}^n y_i\phi_z(x_i)$ and standard deviation $\sigma' = \sigma$.

The full conditional distribution for σ is

$$p(\sigma|a_0, ..., a_d, y, x) \propto \frac{1}{\sigma^{2\alpha+n+1}} e^{\frac{-1}{\sigma^2}\left(\beta + \frac{1}{2}SE\right)} \quad (16)$$

which is an inverted gamma distribution with parameters: $\alpha' = \alpha + \frac{n}{2}$ and $\beta' = \beta + \frac{1}{2}SE$.

4.2 Birth and Death steps

The birth and death steps require dimension matching. If we jump from a k'th order polynomial to an m'th order polynomial then the new coefficients are sampled using Gibbs sampling. To create the necessary bijection between the two spaces we generate $m - k$ variables from an $N(0, 1)$ distribution.

$$(a_0, ..., a_d, \eta_1, ..., \eta_{m-k}, \sigma) <-> (a'_0, ..., a'_m, \sigma') \quad (17)$$

Using Green's notation the acceptance probability of the move is

$$min(1, (\text{likelihood}) \times (\text{prior}) \times (\text{proposal}) \times (\text{Jacobian})) \quad (18)$$

Algorithm 2 *Polynomial Sampler Algorithm*

```
build the table of move probabilities for p(m|k)
k ← 0
while (sample size < N)
  generate a move, m, from p(m|k)
  if (m > k) then
    perform the birth step
  else if (m < k) then
    perform the death step
  else
    perform the move step
  end
  draw σ using a Gibbs step
  k ← m
end
```

end

The proposal ratio is the same for both Prior 1 and 2

$$\text{proposal ratio} = \frac{p(m|k)}{p(k|m)\prod_{j=1}^{m-k}\frac{1}{\sqrt{2\pi}}e^{-\frac{\eta_j^2}{2}}} \quad (19)$$

The Jacobian for Prior 1 is

$$\left|\frac{\partial(a'_0, ..., a'_m, \sigma')}{\partial(a_0, ..., a_d, \eta_1, ..., \eta_{m-k}, \sigma)}\right| = \left(\frac{u^2\sigma^2}{u^2+\sigma^2}\right)^{\frac{m-k}{2}} \quad (20)$$

and the Jacobian for Prior 2 is

$$\left|\frac{\partial(a'_0, ..., a'_m, \sigma')}{\partial(a_0, ..., a_d, \eta_1, ..., \eta_{m-k}, \sigma)}\right| = \sigma^{m-k} \quad (21)$$

The acceptance probability for the death step is the inverse of the acceptance probability for the birth step.

The sampler begins with a zero order least squares model. The algorithm (k is the order of the last polynomial in the chain) is given in Algorithm 2.

The MMC algorithm needs to be able to compute the Kullback-Leibler (KL) distance between two models to build the regions. Since each y_i is independent, the KL distance for a polynomial model is the sum of Gaussian KL distances taken at each known x_i. Given the true model $T = (a_0, ..., a_d, \sigma)$ and inferred model $I = (\hat{a}_0, ..., \hat{a}_{\hat{d}}, \hat{\sigma})$, the (analytical) KL distance simplifies to

$$n\log\frac{\hat{\sigma}}{\sigma} - \frac{n}{2}(1 - \frac{\sigma^2}{\hat{\sigma}^2}) + \frac{1}{2\hat{\sigma}^2}\sum_{j=0}^{max(d,\hat{d})}(a_j - \hat{a}_j)^2 \quad (22)$$

where $\forall_{i>d}\ a_i = 0$ and $\forall_{i>\hat{d}}\ \hat{a}_i = 0$. The KL distance for generalised Fourier series can be computed in $O(max(d, \hat{d}))$ time.

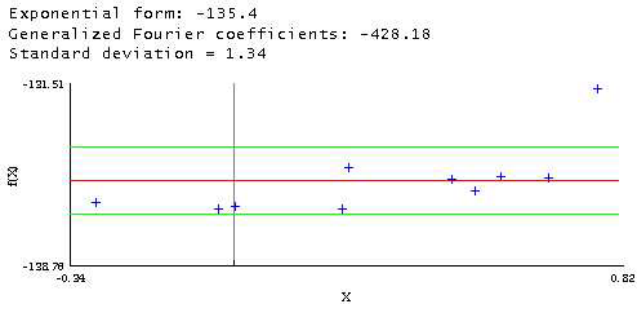

Figure 1. MMC fit for data generated from the low-order polynomial (quintic) with $n = 10$ and $\sigma = 1$

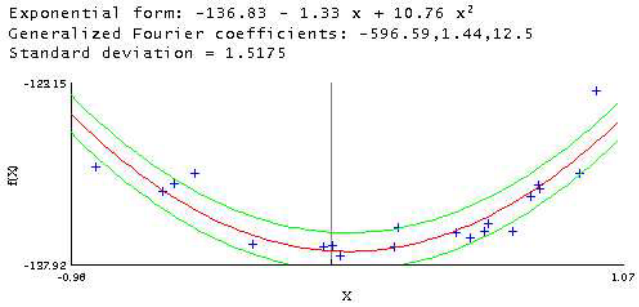

Figure 2. MMC fit for data generated from the low-order polynomial (quintic) with $n = 20$ and $\sigma = 1$

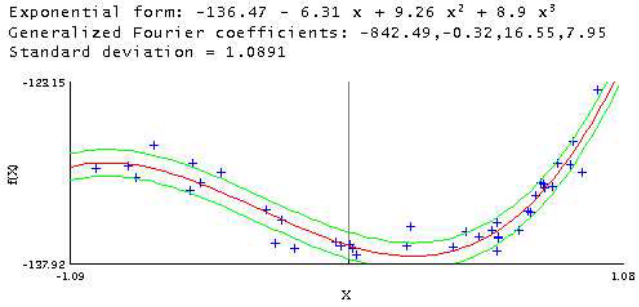

Figure 3. MMC fit for data generated from the low-order polynomial (quintic) with $n = 40$ and $\sigma = 1$

Some example analyses using the MMC algorithm are shown in Figures 1 to 3. The graphs show the generated data[3] as crosses and the fitted function with the estimated noise represented at plus and minus one standard deviation. For this simple example we increased the number of data points, n, from 10 to 20 to 40. We see that MMC is able to choose a model whose complexity is suitable for the given data.

[3] $f = 9.72x^5 + 0.801x^3 + 0.4x^2 - 5.72x - 136.45 + N(0, 1)$

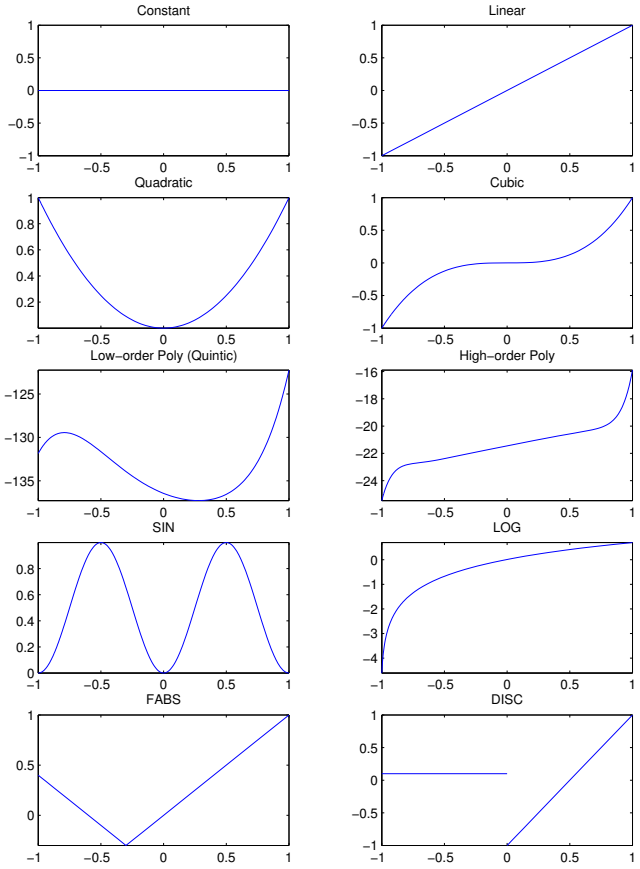

Figure 4. Target functions

5. Experimental Evaluation

We compare the MMC method with two other inference methods on a set of polynomial and non-polynomial target functions. We use the same target functions as [8] and also include some low order polynomials. The target functions are illustrated in Figure 4. Each method is given noisy data generated from the known target function, and is required to infer a single polynomial up to order 20. We use the Squared Prediction Error (SPE) criterion to measure the performance of the methods.

The following methods are included in the evaluation: MMC using Prior 1 (MMC-1); MMC using Prior 2 (MMC-2); Minimum Message Length (MML87) [8]; and Structural Risk Minimization (SRM) [7]. For the MMC methods we sample 3000 polynomials from the posterior, the first 500 being discarded as burn-in.

The experiment consisted of 1000 trials for $n = 10$ and $n = 100$, and for high and low noise values. High noise is defined as a Signal to Noise Ratio (SNR) of 0.78125 where the SNR is defined as the second moment of the

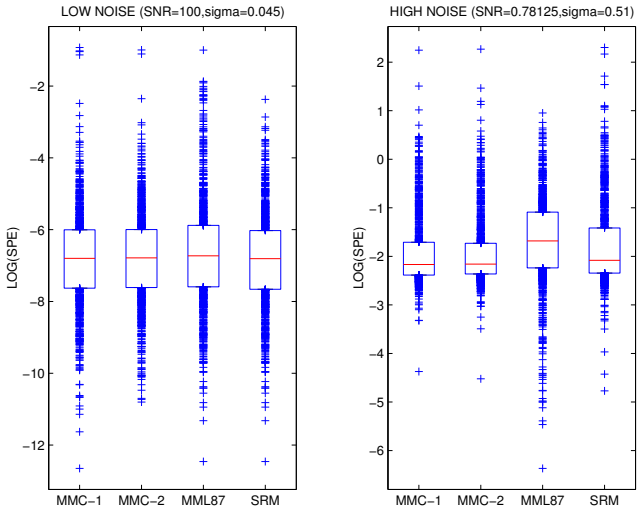

Figure 5. Comparison of Methods on the Quadratic Function (n=10): $f(x) = x^2 + N(0, \sigma^2)$

function about zero divided by the noise. Low noise is defined as an SNR of 100. Due to the number of results, they could not all be included (see [1] for more). A subset of the results are displayed in a boxplot (Figure 5). The boxplot is of the natural logarithm of the SPE, because the log-SPE data looked to be normally distributed upon inspection. Each box shows the lower quartile, median and upper quartile.

The subset of results presented was chosen because it is representative of the majority of the omitted results. The MMC method was found to have lower SPE medians, and better inter-quartile ranges, on average, than MML87 and SRM. The MMC results where almost identical for the two priors used. We note that our results may be interpreted differently to those presented in [8] because they plot the average SPE, for which MML87 outperforms SRM.

6. Conclusion

We have applied the Message from Monte Carlo (MMC) inference method (using the MMLD approximation and FSMML Boundary Rules) to univariate polynomial model selection. The MMC method was compared with Structural Risk Minimisation (SRM) and Minimum Message Length (MML87) and was found to have the lowest median Squared Prediction Error (SPE) (on average) for data generated from a range of polynomial and non-polynomial target functions. The MMC algorithm is flexible, only requiring a sample from the posterior distribution and a means of calculating the Kullback-Leibler distance. This generality allows it to be easily applied to new and difficult problems.

References

[1] L. J. Fitzgibbon, D. L. Dowe, and L. Allison. Message from Monte Carlo. Technical Report 107, School of Computer Science and Software Engineering, Monash University, Clayton, Victoria 3800, Australia, 2002.

[2] J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339, Nov. 1989.

[3] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov chain Monte Carlo in practice*. Chapman-Hall, London, 1996.

[4] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[5] E. Lam. Improved approximations in MML. Honours thesis, Monash University, School of Computer Science and Software Engineering, Monash University, Clayton, Australia, 2000.

[6] J. J. Rissanen. Hypothesis selection and testing by the MDL principle. *Computer Journal*, 42(4):260–269, 1999.

[7] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[8] M. Viswanathan and C. S. Wallace. A note on the comparison of polynomial selection methods. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*, pages 169–177. Morgan Kauffman, January 1999.

[9] C. S. Wallace. *PAKDD-98 Tutorial: Data Mining*. Monash University, Australia (Book in preparation), 1998.

[10] C. S. Wallace and D. M. Boulton. An information measure for classification. *Computer Journal*, 11(2):185–194, August 1968.

[11] C. S. Wallace and D. L. Dowe. Minimum message length and Kolmogorov complexity. *Computer Journal*, 42(4):270–283, 1999.

[12] C. S. Wallace and P. R. Freeman. Estimation and inference by compact encoding (with discussion). *Journal of the Royal Statistical Society. Series B (Methodological)*, 49:240–265, 1987.