Trans-dimensional MCMC tutorial

Hunter Akins

September 21, 2023

Contents

1 Introduction **2**

2 Polynomial problem **2**
2.1 Measurement model . . . 2
2.2 Vector notation . . . 2
2.3 Solving the problem . . . 3

3 Bayesian formulation of the polynomial problem **3**
3.1 Prior distribution . . . 3
3.2 Likelihood function . . . 3
3.3 Bayes' rule . . . 4
3.4 Example: Posterior distribution for prior and noise Gaussian distributions . . . 4
3.5 Summary . . . 4

4 Sampling **5**

5 Markov chains **5**
5.1 Background . . . 5
5.2 Equilibrium distribution . . . 6
5.3 Metropolis-Hastings for detailed balance . . . 7
5.4 Green's formulation . . . 7

6 Metropolis-Hastings sampling for polynomial problem **9**
6.1 Simulation with Gaussian prior and Gaussian noise . . . 9
6.2 Simulation with Gaussian prior and exponential noise . . . 9

7 Practical concerns of MCMC: adaptive proposal covariance matrix and parallel tempering **12**
7.1 Adaptive proposal distribution . . . 12
7.2 Parallel tempering . . . 12

8 Trans-dimensional sampling for the polynomial problem **13**
8.1 Polynomial problem . . . 13
8.2 Sampler moves . . . 14

1 Introduction

This document details the development of a trans-dimensional sampler, motivated by a polynomial regression problem.

2 Polynomial problem

This section introduces notation for the polynomial problem. By polynomial problem, we mean the problem of *inferring* underlying polynomial *coefficients* m by analyzing *measurements* y of the polynomial *values* corrupted with additive noise. Later we will compare the familiar linear inverse solution to a more general Bayesian inference of the posterior distribution.

2.1 Measurement model

Assume that we have made N measurements, $y_1, y_2, \ldots, y_N$ of a system state at times $t_1, t_2, \ldots, t_N$. We have reason to believe that the measurements are corrupted with additive noise obeying some probability distribution. We also have reason to believe that the underlying process $f(t)$ that drives the system state follows a polynomial model, the coefficients of which we would like to know: $f(t, m_0, m_1, \ldots, m_k) = \sum_{l=0}^{k} m_l t^l$. That is, the system state is perfectly determined by knowledge of the polynomial coefficients m_i and the time t. The data model for the measurements is

$$y_i = f(t_i, m_0, m_1, \ldots, m_k) + n_i = \sum_{l=0}^{k} m_l t_i^l + n_i \ , \tag{1}$$

where n_i is the additive noise obeying from some joint distribution on the noise $P_{n_1,n_2,\ldots,n_N}(n_1, n_2, \ldots, n_N)$.

2.2 Vector notation

Let's store the polynomial coefficients in a vector m. Let t be the column vector of the N time values $[t_1, t_2, \ldots, t_N]^T$ at which we have made measurements. We can use the notation t^l to denote this column vector with each of its elements raised to the power l.

Let

$$H = \begin{pmatrix} t^0 & t^1 & \vdots & t^k \end{pmatrix} \tag{2}$$

be the $N \times k+1$ matrix with ith columns given by the time samples raised to the ith power.

Then, the collection of values $f(t_i)$ in a vector $\hat{y} = \begin{pmatrix} f(t_1) & f(t_2) & \ldots, f(t_N) \end{pmatrix}^T$ is linear in m

$$\hat{y} = Hm \ . \tag{3}$$

Last, we need to collect the noise values n_i into a vector n which is distributed according to the joint distribution of the noise values. We now write the measurement model in vector form:

$$y = Hm + n \ . \tag{4}$$

Since the noise is random, the measurement is also random, with statistics determined by H, m, and the statistics of m.

2.3 Solving the problem

At this point, the problem is as follows: "Given measurement values in the vector y, tell me what the polynomial coefficients in m are?" However, the randomness of n makes it impossible to know precisely what m is.

Suppose we knew exactly what the noise realization n was. Then we could solve for m as

$$m = H^{-1}(y - n) \ , \tag{5}$$

assuming that H is invertible. This requires that H be square, and therefore $N = k+1$. If $N < k+1$, the problem is underdetermined and there are infinitely many choices for the polynomial coefficients m that solve the problem. If $N > k + 1$, then presumably the measurements become redundant and the problem is still solvable (assuming that the system truly follows the correct model). Linear inverse theory addresses the problems where $N \neq k + 1$.

The assumption that we know the noise realization n basically corresponds to very high signal-to-noise ratio. In this case we basically know that n is equal to 0 (i.e. that $y - n \approx y$). Then we can solve the problem using linear inverse theory. However, when we only know the distribution of n, we can obtain a statistical description of m that conveys the information that the measurement contains regarding the coefficients. This is what is meant by "solving the problem". That is: "Given measurement values in y and a probability distribution on the noise n, give me a probability distribution on m that captures the uncertainty in the measurements' relationship to the coefficients m. Linear inverse theory is also relevant here as it ....

3 Bayesian formulation of the polynomial problem

This section covers the Bayesian formulation of the polynomial problem. In this formulation of the problem, there is presumed to be prior knowledge of the polynomial coefficients, represented as a (joint) probability distribution on m. The polynomial problem is therefore to use the information in the measurements y to update the distribution on m.

3.1 Prior distribution

We assume that there is a prior distribution on the polynomial coefficients $p_m(m)$. For example, suppose that the system state is set by configuring some knobs on a user interface. The knobs undergo thermal motion, and therefore wander from their initial setting in a random fashion. If the variance of this thermal motion is known, we have a statistical distribution that reflects our knowledge about their current state (given that we know their original configuration).

Continuing this example, suppose that all the knobs are originally set to 0, and the thermal motion is a Wiener process with a knob-dependent variance σ_l^2. Let $\Sigma_M = \mathrm{diag}(\sigma_0^2, \sigma_1^2, \ldots, \sigma_N^2)$. Then

$$p_m(m) = \exp\bigl(-m^T \Sigma_M^{-1} m\bigr) / \sqrt{2\pi \det(\Sigma_M)} \ . \tag{6}$$

Note that since Σ_M is diagonal, the determinant is simply a product of individual variances $\det(\Sigma_M) = \Pi_{l=0}^{k} \sigma_l^2$. After a long time has passed, σ_l becomes very large, which limits the bias that our prior knowledge asserts.

3.2 Likelihood function

The likelihood function $p_{y \mid m}(y \mid m) = \mathcal{L}(m)$ is the probability distribution function for making a measurement y, given that the true polynomial coefficients are fixed at m. Somewhat confusingly,

the notation for $\mathcal{L}$ suppresses the dependence of the function on the measurements y. This is because ultimately these will be "given", at which point the function is really only varying over possible model state m.

For an given value of m, we have that $y = Hm + n$. The probability of making a measurement y is therefore the probability that the noise assumes a value $y - Hm$. If the noise has a probability distribution function (pdf) $p_n(n)$, then the likelihood function is $\mathcal{L}(m) = p_n(y - Hm)$. For example, with a zero-mean white Gaussian noise model $n \sim \mathcal{N}(0, \sigma^2 I)$, the likelihood function is

$$\mathcal{L}(m) = p(y \mid m) = \exp\left(-\frac{||y - Hm||^2}{2\sigma^2}\right)/(2\pi\sigma)^{N/2} \; . \tag{7}$$

3.3 Bayes' rule

Ultimately, we would like to sample the **posterior** probability distribution of the polynomial coefficients m, given **prior** information about m in $p_m(m)$ and the information from the **measurement** in the likelihood function $\mathcal{L} = p_{y \mid m}(y \mid m)$. The posterior distribution of m conditioned on the measurements y is given in terms of the likelihood and the prior distribution by Bayes' rule

$$p_{m \mid y}(m \mid y) = \frac{p_{y \mid m}(y \mid m) p_m(m)}{p_y(y)} \; . \tag{8}$$

The denominator $p_y(y) = \int_M p_{y \mid m}(y \mid m) p_m(m)$ is the "evidence" and captures how likely the mesaurement y is. $p_m(m)$ is the prior distribution of m: it is independent of the measurement value y. We also recognize the likelihood function in the numerator $\mathcal{L}(m) = p_{y \mid m}(y \mid m)$.

In terms of the likelihood function $\mathcal{L}$ the posterior distribution of m is

$$p_{m \mid y}(m \mid y) = \frac{\mathcal{L}(m) p_m(m)}{c} \; , \tag{9}$$

where I have introduced the constant $c = p_y(y)$ to denote the evidence. In MCMC sampling methods, the evidence need not be known to obtain samples from the posterior distribution.

3.4 Example: Posterior distribution for prior and noise Gaussian distributions

In the case that both the noise follows a Gaussian distribution and the prior model uncertainty follows a Gaussian distribution, the posterior distribution will be a product of two Gaussian functions and therefore is a Gaussian as well (this is a well-known result). Using results from Kay (1993), Chapter 10, we have that the mean of the posterior Gaussian is

$$E(m \mid y) = \Sigma_M H^T (H \Sigma_m H^T + \sigma^2 I)^{-1} (y) \tag{10}$$

and the covariance is

$$\Sigma_{m \mid y} = \Sigma_M - \Sigma_M H^T (H \Sigma_M H^T + \sigma^2 I)^{-1} H \Sigma_M \; . \tag{11}$$

This is a useful result for validating our code, since we can check that our sampler provides samples with the appropriate mean and covariance.

3.5 Summary

The polynomial problem can be framed in terms of Bayesian inference as using the measurement y and noise statistics to transform a prior distribution on m to a posterior distribution.

4 Sampling

In the previous section we laid out a polynomial regression problem, framing it in the context of Bayes' theorem as the problem of solving for a posterior distribution. This section asserts that "solving for a posterior distribution" can be framed as "sampling from a distribution".

Assuming that the model is correctly specified and the noise statistics are correctly specified, then the posterior distribution function $p_{m \mid y}(m \mid y)$ gives this statistical description and contains all of the information about the underlying polynomial coefficients that we can obtain by combining our measurement y and our prior knowledge. The posterior distribution can then be used to calculate a "best" set of polynomial coefficients given some desired objective. For example, the expected value of m is given by

$$E(m) = \int_{\mathcal{M}} m p_{m \mid y}(m \mid y) \mathrm{d}m \ , \tag{12}$$

and is the estimate of m that has the minimum variance. We could also calculate the expected value of the system state $f(t)$ at a time t as

$$E(f(t)) = \int_{\mathcal{M}} f(t, m) p_{m \mid y}(m \mid y) \mathrm{d}m \ . \tag{13}$$

Note that these values require integration of a function times the posterior distribution. It is often difficult to obtain a closed form, analytic function for the posterior distribution. Monte Carlo integration uses samples $\{m_i\}$ drawn from the posterior distribution to approximate these integrals in a sum

$$E(a(m)) \approx \frac{1}{N} \sum_i a(m_i) \ . \tag{14}$$

As the number of samples $N \to \infty$, the error in the approximation to the integral goes to zero. In the absence of a perfect, analytic posterior distribution, one can therefore make due with (potentially a lot of) samples drawn from the posterior distribution.

There are many ways to sample a distribution (importance sampling, nested sampling, particle filtering, and more). Markov Chain Monte Carlo (MCMC) is a way to draw samples from a distribution when a user has a means of computing the likelihood function and prior distribution. It is supposed to be the go-to method for sampling high dimensional distributions. It draws the samples by constructing a Markov chain whose equilibrium distribution converges to the desired posterior probability distribution.

5 Markov chains

The goal of Markov Chain Monte Carlo is a) to design a Markov Chain whose equilibrium distribution is the desired posterior distribution $\pi_x(x)$, b) generate a random realization of the chain, and c) use the chain samples to calculate expectations via Monte Carlo integration.

5.1 Background

A Markov chain is a random sequence of points $x_0, x_1, \cdots \in \mathcal{X}$, where the probability distribution of the nth point $\pi_{x_n \mid x_0, x_1, \ldots, x_{n-1}}(x_n \mid x_0, x_1, \ldots, x_{n-1}) = \pi_{x_n \mid x_{n-1}}(x_n \mid x_{n-1}) \equiv \mathbb{T}(x_n, x_{n-1})$. That is, the probability distribution of a the nth point in the sequence depends only on the previous point. *Homogeneous* Markov chains are such that the transition probability distribution $\mathbb{T}$ does not change with n ("time"). A Markov chain is completely specified by an initial distribution $x_0 \sim \alpha_{x_0}(x_0)$, and the transition distribution $\mathbb{T}(x_n, x_{n-1})$.

Example 1: A discretely sampled Wiener process (Brownian motion) is described by a Markov chain. $x \in \mathbb{R}$.

$$\mathbb{T}(x_n, x_{n-1}) = \exp\bigl(-(x_n - x_{n-1})/2\sigma^2\bigr)/\sqrt{2\pi\sigma}\ . \tag{15}$$

Typically one considers a known initial point x_*, in which case $\alpha_{x_0}(x_0) = \delta(x_0 - x_*)$.

Example 2: Sequence of independent coin flips as a Markov chain. $x \in \{H, T\}$. Initial measure is $\alpha_{x_0}(x_0) = 1/2$ for $x = H$ and $1/2$ for $x = T$.

$$\mathbb{T}(x_n, x_{n-1}) = \begin{cases} \frac{1}{2} \text{ if } x_{n-1} = H \\ \frac{1}{2} \text{ if } x_{n-1} = T \end{cases} \tag{16}$$

Equipped with a means of sampling from $\mathbb{T}$, one draws a random sample x_0 from an initial distribution, then draw a new random sample x_1 by sampling $\mathbb{T}(x_1, x_0)$, then draw a new random sample x_2 by sampling $\mathbb{T}(x_2, x_1)$, and so on to generate a random sequence $x_0, x_1, \ldots, x_N$. Example 1 will generate a realization of the "drunkards walk" (the motion of a colloid in 1-d static medium). Example 2 will generate a realization of a sequence of coin tosses with an unbiased coin.

5.2 Equilibrium distribution

Under certain conditions, the samples of a Markov chain will be stationary meaning that each sample x_n follows a distribution $\pi_{x_n}(x_n)$. Note that this distribution is different from the conditional probability $\pi_{x_n \mid x_{n-1}}(x_n \mid x_{n-1})$. A sufficient condition for an irreducible Markov chain to have an equilibrium (or "invariant" or "stationary") distribution $\pi_x(x)$ if

$$\pi_x(x) = \int_{\mathcal{X}} \pi_x(x')\mathbb{T}(x, x')\mathrm{d}x'\ . \tag{17}$$

When the chain is "irreducible", the chain converges to this distribution irrespective of the initial distribution α. The equilibrium distribution is the probability distribution of the nth sample in the chain if you do not have information on the previous sample. Another way to understand the relationship between the stationary distribution and the chain is that the normalized histogram of samples from the chain will converge to the probability distribution π. In general, the chain samples are correlated, so subsequent samples do not represent independent samples from the equilibrium distribution.

Detailed balance is a *stronger*, sufficient condition for a chain to converge to a distribution π. It is as follows:

$$\mathbb{T}(x', x)\pi_x(x) = \mathbb{T}(x, x')\pi_x(x')\ . \tag{18}$$

Integrating both sides over x' and using the fact that $\mathbb{T}$ must be normalized

$$\pi(x)\int_{\mathcal{X}} \mathbb{T}(x', x)\mathrm{d}x' = \pi(x) = \int_{\mathcal{X}} \mathbb{T}(x, x')\pi(x')\mathrm{d}x' \tag{19}$$

proves that its a sufficient condition for convergence to π.

An equivalent statement of detailed balance, "integrated detailed balance", is that

$$\int_{(x,x')\in\mathcal{A}\times\mathcal{B}} \pi(x)\mathbb{T}(x', x)\mathrm{d}x\mathrm{d}x' = \int_{(x,x')\in\mathcal{A}\times\mathcal{B}} \pi(x')\mathbb{T}(x, x')\mathrm{d}x\mathrm{d}x' \tag{20}$$

for all Borel sets $A, B \subset \mathcal{X}$. This is relevant for the Metropolis-Hastings-Green algorithm.

5.3 Metropolis-Hastings for detailed balance

To design a Markov chain that satisfies detailed balance for a given target distribution π, the Metropolis-Hastings algorithm constructs a transition probability $\mathbb{T}(x', x)$ in two steps.

The first step draws a candidate next Markov chain sample x' using a convenient (i.e. easy to sample) proposal distribution $q(x', x_{n-1})$. A common choice is a Gaussian with mean equal to x_{n-1} and fixed variance.

The second step is to randomly accept or reject the proposed step x' in such a way as to ensure detailed balance is assured. This will be done with an auxiliary distribution $\alpha(x', x_{n-1})$:

$$\alpha(x', x_{n-1}) = \min\left\{1, \frac{\pi(x')q(x_{n-1}, x')}{\pi(x_{n-1})q(x', x_{n-1})}\right\} . \tag{21}$$

The next step in the chain x_n equals x' if accepted or x_{n-1} if rejected.

The transition probability of the associated chain $\mathbb{T}(x_n \mid x_{n-1})$ is therefore the probability of proposing x_n from x_{n-1} and accepting it, or, if $x_n = x_{n-1}$, the probability of rejecting a proposed move (the complement of accepting any move):

$$\mathbb{T}(x_n, x_{n-1}) = q(x_n, x_{n-1})\alpha(x_n, x_{n-1}) + (1 - \int_{x'} \alpha(x', x_{n-1})q(x', x_{n-1})\mathrm{d}x')\delta(x_n - x_{n-1}) . \tag{22}$$

The first term above is the probability of proposing and accepting x_n, and the second term is the probability of a rejection. One can directly verify that detailed balance is satisfied by inserting the transition kernel (22) into the detailed balance equation (18).

The class of Markov chains associated with the Metropolis-Hastings transition probability (regardless of their initial sample distribution) will converge to the equilibrium distribution π. Therefore, all that is required to draw samples from the distribution π is the ability to sample from the proposal q, the ability to evaluate the π, and possibly infinite computation time.

5.4 Green's formulation

Green [1, 2] introduced the reversible jump Metropolis-Hastings algorithm, which is a proposal-acceptance algorithm that allows one to sample across multiple models with potentially differing numbers of dimensions. It is also known as the Metropolis-Hastings-Green algorithm. I found that the derivations in the cited works were a bit challenging for me, as I have limited familiarity with measure theory and the Radon-Nikodyn theorem. I also found the notation to be opaque. I hope what follows is a more painstaking but ultimately more clear description of the description.

The derivation relies on "integrated detailed balance" described above. A step in the chain is a jump from one point in $x \in \mathcal{X}$ to another (possibly equal) point $x' \in \mathcal{X}$.

The transition probability for the reversible jump Markov chain, as in Metropolis-Hastings, is constructed via a proposal followed by an accept-reject stage:

$$\mathbb{T}(x', x) = q(x', x)\alpha(x', x) + (1 - \int_z q(z, x)\alpha(z, x))\delta(x - x') . \tag{23}$$

For this transition kernel, the integrated detailed balance equation (20) reads

$$\int_{(x,x')\in\mathcal{A}\times\mathcal{B}} \pi(x)q(x', x)\alpha(x', x)\mathrm{d}x\mathrm{d}x' = \int_{(x,x')\in\mathcal{A}\times\mathcal{B}} \pi(x')q(x, x')\alpha(x, x')\mathrm{d}x\mathrm{d}x' . \tag{24}$$

Note here that we need note include the rejection, since it contributes the same value to each side (an integral over $A \cap B$ with equal integrand).

Green provides a "constructive representation" of the trans-dimensional sampler that moves the "randomness" part of proposals to an auxiliary random variable, and then considers a deterministic function of the value of the random variable and the current state (this will become more clear shortly). In Metropolis-Hastings, we put the "randomness" in the proposal distribution $q(x', x)$ which is parameterized by the state x. In reversible jump, we place the randomness in a random variable u with distribution $g(u)$, and propose a new state deterministically as $(x', u') = h(x, u) = (h_1(x, u), h_2(x, u)$. With this scheme in place, consider the probability of proposing a jump from x to x': $q(x', x)$ is now specified by the probability distribution $g(u)$ (for the specific u required to move from x to x'). We will introduce the notation $\mathcal{U}$ to refer to the set that is mapped to the set $\mathcal{B}$ under the transformation h_1. Implicit in this is a dependence on x, which is suppressed in the notation. This allows one to move the integral over $x' \in \mathcal{B}$ in the left hand side of equation (24) to an integral over $u \in \mathcal{U}$:

$$\int_{(x,x')\in\mathcal{A}\times\mathcal{B}} \pi(x)q(x', x)\alpha(x', x)\mathrm{d}x\mathrm{d}x' = \int_{(x,u\in\mathcal{A}\times\mathcal{U}} \pi(x)g(u)\alpha(x'(x, u), x)\mathrm{d}x\mathrm{d}u \ . \tag{25}$$

If we assume that the mapping from $x, u \to (x', u')$ is a diffeomorphism, then I can perform the integral on the right hand side of equation (24) over x, u as well! However, we need to use the change-of-variables formula to preserve volume in the integral: changing from $\mathrm{d}x'\mathrm{d}u'$ to $\mathrm{d}x\mathrm{d}u$, we need to include the determinant of the Jacobian transformation

$$|J_h| = \left|\frac{\partial x', u'}{\partial x, u}\right| = \left|\begin{bmatrix} \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial x} \\ \frac{\partial h_2}{\partial x} & \frac{\partial h_2}{\partial u} \end{bmatrix}\right| . \tag{26}$$

Then, the integrated detailed balance equation becomes

$$\int_{(x,u)\in\mathcal{A}\times\mathcal{U}} \pi(x)g(u)\alpha(x'(x, u), x)\mathrm{d}x\mathrm{d}u = \int_{(x,u)\in\mathcal{A}\times\mathcal{U}} \pi(x'(x, u))g'(u'(x, u))\alpha(x'(x, u), x)\left|\frac{\partial x', u'}{\partial x, u}\right|\mathrm{d}x\mathrm{d}u \ . \tag{27}$$

Finally, then, we get that to satisfy detailed balance, it suffices to satisfy

$$\pi(x)g(u)\alpha(x'(x, u), x) = \pi(x'(x, u))g'(u'(x, u))\alpha(x, x'(x, u))\left|\frac{\partial x', u'}{\partial x, u}\right| . \tag{28}$$

If we choose

$$\alpha(x', x) = \min\left\{1, \frac{\pi(x')g'(u')}{\pi(x)g(u)}\left|\frac{\partial x', u'}{\partial x, u}\right|\right\} \ , \tag{29}$$

then we will satisfy detailed balance.

Finally, in what follows, there will be a menu of "moves" that the chain can make. Each move comes with proposal distributions characterized by $g(u)$ and $g'(u')$, as well as a diffeomorphism $x'(x, u), u'(x, u)$. The specific move will be identified with an index m and selected randomly according to some distribution $j_m(x)$ that may depend on the state x. Given that move m is attempted, the acceptance probability becomes

$$\boxed{\alpha_m(x', x) = \min\left\{1, \frac{\pi(x')g'_m(u')j_m(x')}{\pi(x)g_m(u)j_m(x)}\left|\frac{\partial x', u'}{\partial x, u}\right|\right\} \ .} \tag{30}$$

The dependence of the Jacobian on the move type is suppressed for notational convenience.

6 Metropolis-Hastings sampling for polynomial problem

We will first implement Metropolis-Hastings sampling for the polynomial problem when we know the number of polynomial coefficients. The number of coefficients is set to 5. The number of samples $N = 100$. The time grid is uniformly spaced from $t = -1$ to $t = 1$. Data is generated by randomly first selecting a realization of the polynomial coefficients m_{true} and using these to generate the state values $y_{true} = Hm_{true}$. Then, noise is randomly generated according to the distribution under consideration and is added to the measurements to get $y = y_{true} + n$.

6.1 Simulation with Gaussian prior and Gaussian noise

Here we have a Gaussian prior distribution on the coefficients and a Gaussian distribution for the noise. Let's assume an equal variance for the coefficients σ_0^2 and a (different) equal variance for the noise σ_n^2. Let's also assume both distributions are zero-mean. Then, the likelihood distribution is Gaussian

$$\mathcal{L}(m) = p(y \mid m) = \exp\left(-\frac{||y - Hm||^2}{2\sigma_n^2}\right)/(2\pi\sigma_n)^{N/2} \,. \tag{31}$$

The posterior distribution is also Gaussian (see section 3.4), with a mean and covariance that can be calculated from H, y, σ_0^2 and σ_n^2. The problem is therefore to sample this multivariate Gaussian. Since we can actually compute the posterior mean and covariance, this will be a good example to check our sampler.

The Metropolis-Hastings algorithm requires only a proposal distribution to be selected (and will work regardless of the distribution chosen). Here we will use a spherical Gaussian proposal distribution centered at the previous sample with variance σ^2: $q(x', x) \sim \mathcal{N}(x, \sigma^2)$. The variance of the proposal Gaussian is tuned to obtain a reasonable acceptance ratio (see Gelman, Roberts and Gilks 1997 [3]). I ultimately settled on a proposal variance of $\sigma^2 = (.02)^2$. Note that the acceptance ratio doesn't affect the asymptotic behavior of the chain. However, we can't just wait for infinity to come around, so it has a practical effect. The integrated autocorrelation time is the relevant statistic for considering the independence of chain samples (see the well-known Sokal notes from 1996, which can be found online). The initial distribution $\alpha_0(x)$ also doesn't affect the equilibrium distribution. Here we will just draw a sample from the prior distribution to start the chain.

I run the chain for 100000 steps. I discard the first 10000.

The results of this simulation are shown in Figure 1.

6.2 Simulation with Gaussian prior and exponential noise

Now we consider that the noise is independent Laplacian distributed with mean 0 and scale parameter b: $p_{n_i}(n_i) = \frac{1}{2b}\exp(-|n_i|/b)$. The log-likelihood distribution in this case is

$$\log \mathcal{L}(m) = p(y \mid m) = -\sum_{i=0}^{N-1} \frac{|y_i - (Hm)_i|}{b} - N\log(2b) \,. \tag{32}$$

I fix $b = 1$. I use a Gaussian proposal once again, though this time it has a proposal variance of $\sigma^2 = (.05)^2$. I run the chain for 100000 steps. I discard the first 10000. I also use the standard deviation of the noise to get the approximate Gaussian noise distribution. I can use this to compare the results from linear inverse theory using the incorrect assumption of Gaussianity of the noise. The results are shown in Figure 2.

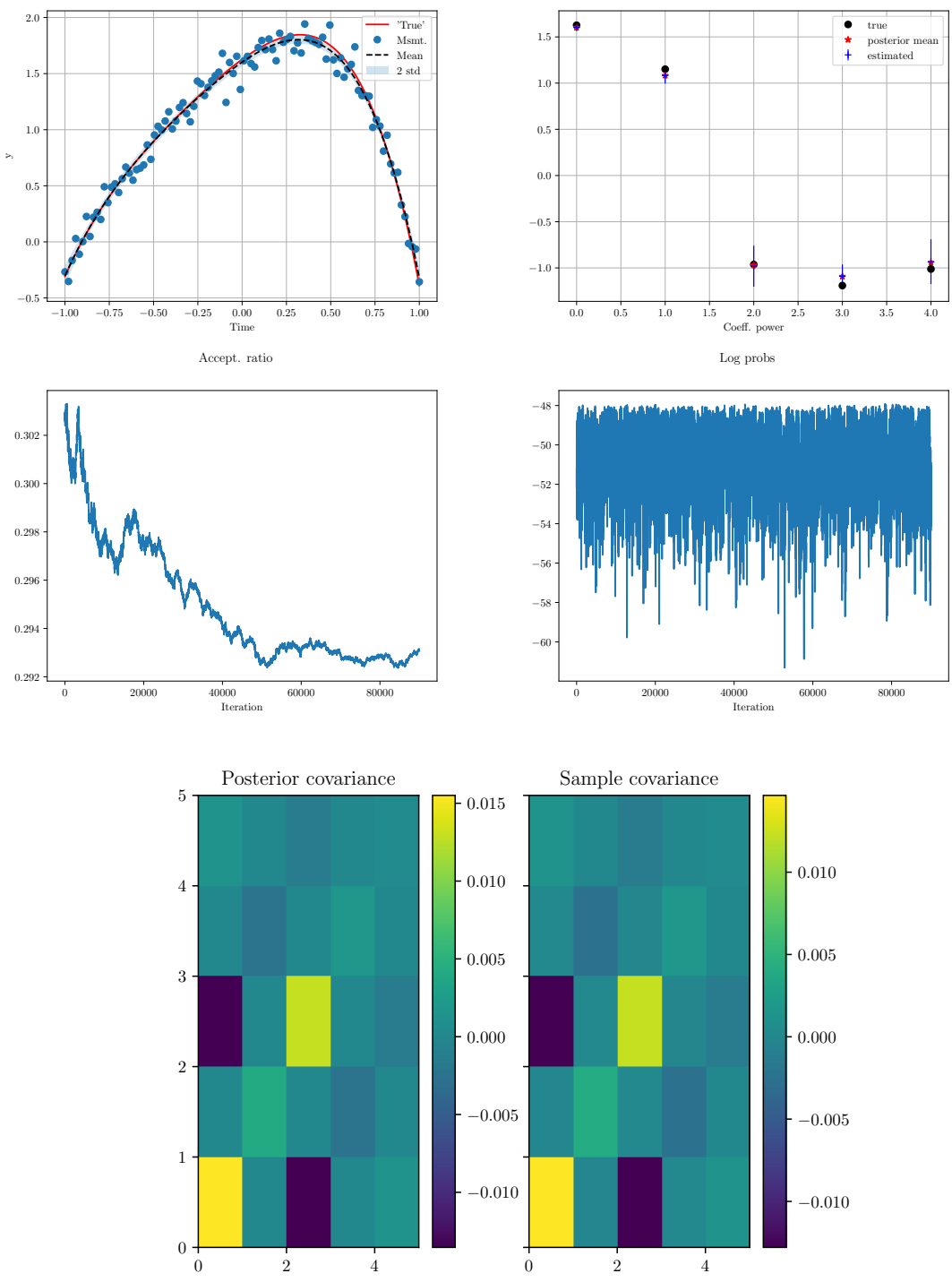

Figure 1: Metropolis-Hastings sampling results. From left to right, top to bottom we have: 1. True polynomial values in black, measured values in blue. 2. True polynomial coefficients, posterior mean using linear inverse theory, and sample mean from MCMC run (with 2 sigma error bars). 3. Acceptance ratio vs. iteration. 4. Log probability of chain as function of sample number. 5. Comparison of the sample covariance matrix and the posterior covariance matrix from linear inverse theory.

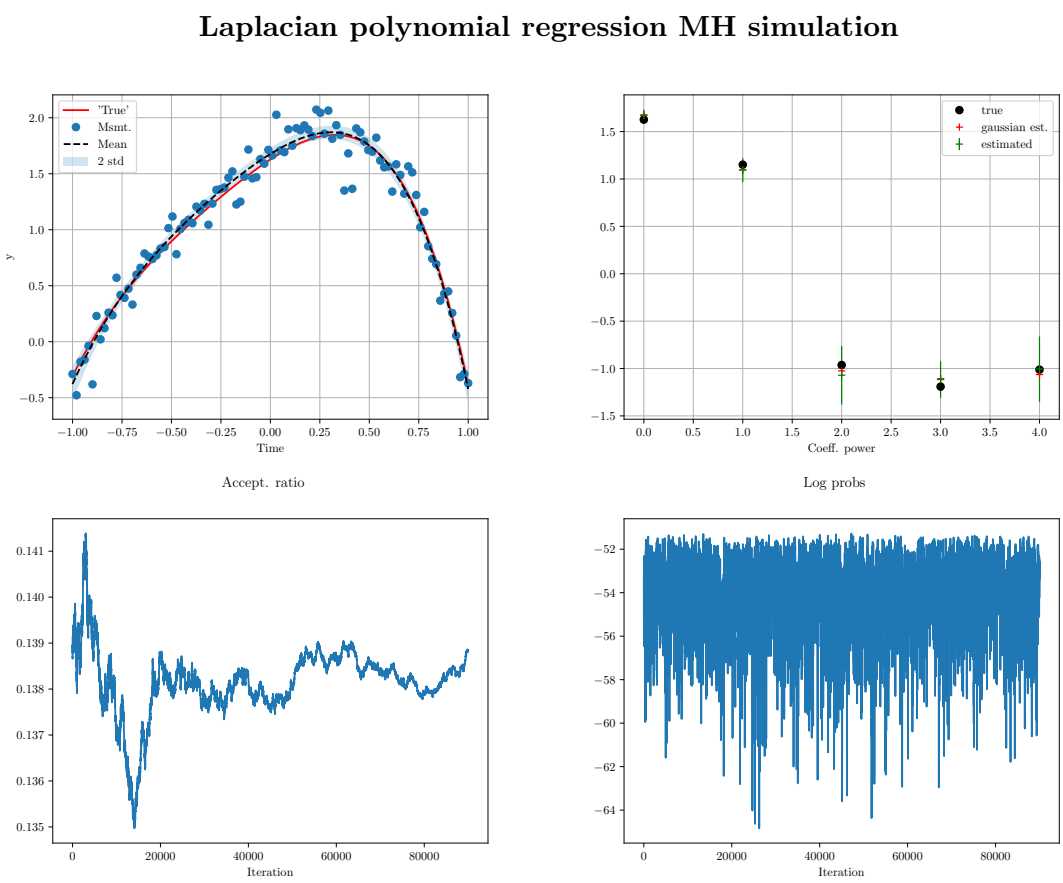

Figure 2: Metropolis-Hastings sampling results. From left to right, top to bottom we have: 1. True polynomial values in black, measured values in blue. 2. True polynomial coefficients, posterior mean using linear inverse theory, and sample mean from MCMC run (with 2 sigma error bars). 3. Acceptance ratio vs. iteration. 4. Log probability of chain as function of sample number. Note that in the top right figure, the red crosses denote the linear inverse best guess under the false assumption of Gaussian noise. The green cross is the minimum variance estimate from the posterior samples from Metropolis-Hastings and is closer to the true values (black dots).

7 Practical concerns of MCMC: adaptive proposal covariance matrix and parallel tempering

In the previous section I showed a simply Metropolis-Hastings MCMC solution to the polynomial problem. Theoretically, M-H theoretically guarantees that the chain will asymptotically approximate the posterior distribution. Practically, however, I would like the chain to not take forever. A "good" chain therefore is one that produces the most "independent" samples in the shortest amount of time.

I will offer two practical tricks to getting a good chain: adaptive proposal [4] and parallel tempering (see Miasojedow 2013 for a nice discussion [5]).

7.1 Adaptive proposal distribution

In order to draw samples from a Gaussian distribution using MCMC, the optimal proposal distribution is a scaled version of the desired Gaussian's covariance matrix (Need to check this result). That is, if the Gaussian to sample has covariance matrix Σ, then the optimal proposal distribution is $s_D\Sigma$. Here, optimal means that the chain produces the most effectively independent samples per step. Gelman, Gilks, and Roberts (1997) [3] that contains a derivation of the asymptotic (in the dimension of the Gaussian) optimality for a chain sampling a Gaussian. In particular, the chain should have an acceptance ration of 0.23, and $s_d = (2.38)^2/\text{dim}$.

When we sample a posterior distribution, we rarely know the posterior covariance at the outset (or even if the distribution is Gaussian). However, as our chain explores the space, it is converging to the true distribution. Therefore, after a while, we can use our chain samples themselves to estimate the posterior covariance, and then use this is our proposal. Continually updating the proposal covariance using the chain samples is called adaptive Metropolis-Hastings [4]. It breaks homogeneity of the chain. However, asymptotically the sample covariance matrix of the samples converges, so the chain becomes asymptotically homogeneous.

I'm not a statistician, so these considerations are above my paygrade. I use a simple heuristic in designing my chain. First, I take a simple uncorrelated Gaussian prior $\Sigma_0 = \text{diag}(\sigma_1^2, \sigma_2^2, \dots)$ on my parameters. Then, I choose as my proposal distribution $g(x', x_{n-1}) \sim \mathcal{N}(x_{n-1}, \gamma\Sigma_0$. For a given value of γ, I can run the chain for a while and compute the acceptance ratio. I then try and find the value γ_* that minimizes the absolute difference between the acceptance ratio and 0.23. I'm not very precise here. Anywhere between 0.1 and 0.5 is fine by me.

Now, I run my chain say N samples with my proposal as $g(x', x_{n-1}) \sim \mathcal{N}(x_{n-1}, \gamma_*\Sigma_0$. I use these N samples to estimate the sample covariance matrix

$$\hat{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{x}) \ . \tag{33}$$

Finally, I use as my proposal covariance matrix $s_D\hat{\Sigma}$, where $s_d = (2.38)^2/\text{dim}$.

This seems to work well, at least in lower dimensions. It also avoids the question of non-homogeneity.

7.2 Parallel tempering

For a multimodal distribution, chains will often spend a lot of time in a single mode. Again, they will *asymptotically* make their way to all of the modes and sample all of them well. Practically I

would like them to do that today. The go-to trick to deal with this issue is parallel tempering. See Sambridge (2013) [6] for a nice comprehensive presentation.

Basically one considers sampling a "tempered" version of the distribution. Suppose we want to sample multimodal distribution $\pi(x)$. Then, we also run chains that sample distributions $\pi_T(x) = \pi^{1/T}(x)$. Here $T \geq 1$ is the temperature. The effect of the power $1/T$ is nicely demonstrated on a Gaussian. If $\pi(x)$ is Gaussian with variance σ^2, then $\pi_T(x) = \pi^{1/T}(x)$ is Gaussian with variance $T\sigma^2$. Therefore, the effect of raising the temperature is to flatten out the peaks in a distribution.

The parallel part comes in to play by basically running, in parallel, chains at different temperature. Then, swaps are proposed between chains of different temperatures and accepted with a probability that maintains detailed balance (see Sambridge [6] for detailed details).

The cold chain therefore will tend to hang out in a single mode sampling around. The hotter chains spend more time visiting different modes. Occasionally the hot chains will swap with the cold chain, leaving it in a new mode that it then explores.

It is important for trans-dimensional sampling with birth-death moves. When you propose a new birth move, you need to draw a random value for the new dimension variable. Since the chain is typically found in high-likelihood regions, it's unlikely that the random new initial value will be likely enough for acceptance. Therefore you have trouble getting good acceptance ratios for birth and death moves. You run a hot chain that is more likely to accept birth-death moves, and therefore sample across models. The cold chain then explores each model.

8 Trans-dimensional sampling for the polynomial problem

Suppose that the order of the polynomial k is in fact unknown. Rather we have some prior knowledge on the order, given as a probability distribution $p(k)$. This gives us a set of possible models for the data, which we can index by the polynomial order k. We will denote a model state vector for the kth model as m_k. We would ultimately like to obtain a posterior distribution on the models, as well as the model state, which is connected to the measurement by Bayes' rule

$$p_{m_k,k \mid y}(m_k, k \mid y) = \frac{p_{y \mid m_k,k}(y \mid m_k, k) p_{m_k,k}(m_k, k)}{p_y(y)} \ . \tag{34}$$

The normalization in the denominator has to sum over model indices k and integrate over states

$$p_y(y) = \sum_{k'} \int_{M_{k'}} p(y \mid m_{k'}, k') \ . \tag{35}$$

We can identify the probability distribution $p_{y \mid m_k,k}(y \mid m_k, k)$ as being proportional to the likelihood function $\mathcal{L}(m_k, k)$ which now also depends on the model order k. $p_{m_k,k}(m_k, k)$ is the prior on m_k, k.

8.1 Polynomial problem

For the polynomial problem, the coefficients for a degree k polynomial live in R^{k+1}. Suppose the considered polynomial degrees are specified in the countable set $\mathcal{K}$. The possible states live in $\mathcal{X} = \cup_{k \in \mathcal{K}}(\{k\} \times R^{k+1})$. By appending the degree to the state we can always identify the degree. Our notation above used

$$m = m_0, m_1, \ldots, m_l \ . \tag{36}$$

To identify order, we will use a subscript:

$$m_k = k, m_0, m_1, \ldots, m_k \ . \tag{37}$$

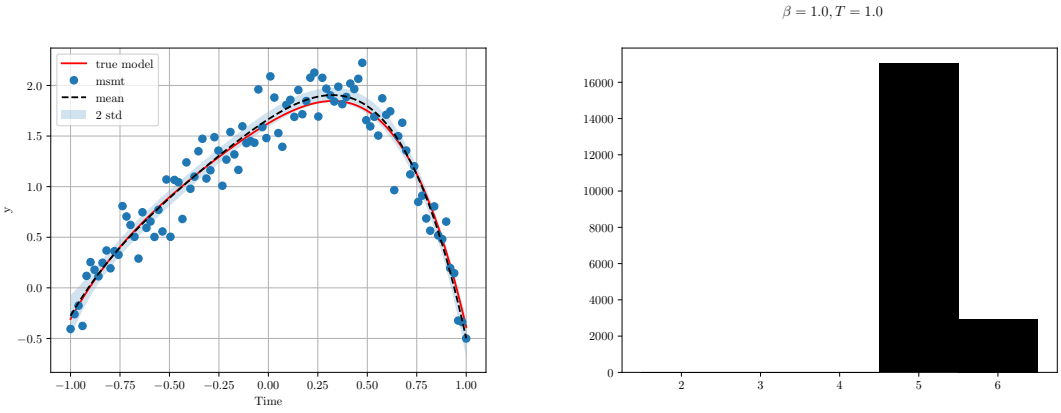

Figure 3: Left: Posterior mean curve from trans-dimensional sampling. Right: Histogram of sample dimension (polynomial degree plus one). Note that most of the samples are at the correct model dimension of five (polynomial degree four).

8.2 Sampler moves

We need to define the moves (proposal steps) that our chain will perform.

The first is a **perturb** move, which maintains the order fixed and simply perturbs the state with proposal g. The diffeomorphism is simply $x', u' = x, u$ so its Jacobian determinant is one. Here the acceptance distribution is

$$\alpha_{birth}(x', x) = \min\{1, \frac{\pi(x')g(u')j_{pert}(x')}{\pi(x)g(u)j_{pert}(x)}\} . \tag{38}$$

Assuming that the probability of performing a perturb move is independent of position, then the j terms will cancel.

The second is a **birth-death** move, which in one direction moves to one order higher than the current state order and in the reverse direction moves to one order lower. Suppose the current state is k. The k state parameters are still meaningful when we jump up to the next order, so it makes sense for our birth diffeomorphism h to preserve those values. We can simply generate a single number u according to a distribution $g(u)$ and use that for the value of the new parameter m_{k+1}. The reverse move would just assign the value of m_{k+1} to the value of u with probability 1, so $g'(u') = 1$. The reverse move is referred to as a **death move**. Due to the simple mapping used, the Jacobian of h is just the identity matrix, so it has determinant 1. The acceptance distribution for the birth move is therefore

$$\alpha_{birth}(x', x) = \min\{1, \frac{\pi(x')(1)j_{death}(x')}{\pi(x)g_{birth}(u)j_{birth}(x)}\} . \tag{39}$$

We see here that strictly trans-dimensional moves come in pairs, since the reverse move is necessarily trans-dimensional. The acceptance distribution for the reverse move is

$$\alpha_{death}(x', x) = \min\{1, \frac{\pi(x')g_{birth}(u)j_{birth}(x')}{\pi(x)(1)j_{death}(x)}\} . \tag{40}$$

References

[1] P. J. Green, "Reversible jump markov chain monte carlo computation and bayesian model determination," *Biometrika*, vol. 82, no. 4, pp. 711–732, 1995.

[2] P. J. Green and D. I. Hastie, "Reversible jump mcmc," *Genetics*, vol. 155, no. 3, pp. 1391–1403, 2009.

[3] A. Gelman, W. R. Gilks, and G. O. Roberts, "Weak convergence and optimal scaling of random walk metropolis algorithms," *The annals of applied probability*, vol. 7, no. 1, pp. 110–120, 1997.

[4] H. Haario, E. Saksman, and J. Tamminen, "An adaptive metropolis algorithm," *Bernoulli*, pp. 223–242, 2001.

[5] B. Miasojedow, E. Moulines, and M. Vihola, "An adaptive parallel tempering algorithm," *Journal of Computational and Graphical Statistics*, vol. 22, no. 3, pp. 649–664, 2013.

[6] M. Sambridge, "A parallel tempering algorithm for probabilistic sampling and multimodal optimization," *Geophysical Journal International*, vol. 196, no. 1, pp. 357–374, 2014.