1 Coding (Maximum of a Brownian Motion).

Use Monte Carlo sampling to estimate the expected maximum of a standard Brownian Motion $\mathbb{E}\max_{0\le t\le 1} W(t)$. You will need to make a discrete approximation $W(0), W(\Delta), W(2\Delta), \ldots, W(1)$, take a maximum, and average over N independent trials. How does the accuracy of your solution depend on the mesh size Δ and the number of samples N empirically? *Challenge:* Repeat the exercise for fractional Brownian motion.

2 Coding (Multilevel Monte Carlo).

Apply multilevel Monte Carlo to estimate the expected maximum of a standard Brownian Motion $\mathbb{E}\max_{0\le t\le 1} W(t)$. How many digits of precision can you achieve? *Challenge:* Repeat the exercise for fractional Brownian motion.

3 Coding (Parallel tempering).

By now, you're (hopefully) an expert in parallel tempering. Let's test the limits and see how far parallel tempering can take us. Try sampling from the density

$$\pi(\boldsymbol{x}) \propto \sum\nolimits_{\boldsymbol{c}\in\{-1,+1\}^d} \exp(-10\|\boldsymbol{x}-\boldsymbol{c}\|^2),$$

which gets harder and harder as we increase the dimensionality d. How high can you take d and still do a good job of sampling all 2^d wells? Does the scaling of the temperature parameters $\beta_1 < \beta_2 < \cdots < \beta_N = 1$ (linear versus geometric) make a difference empirically?

4 Coding (Preconditioned Crank-Nicolson)

Use preconditioned Crank-Nicolson to sample a standard Brownian Motion $W(t)$ for $0 \le t \le 1$ after making a noisy observation $W(1) = 1 + Z$, where $Z \sim \mathcal{N}(0, 0.25)$ is an independent random error term. Hint: you need to sample a standard Brownian motion weighted by the likelihood ratio

$$\ell(W) \propto \exp\bigl(-2\bigl|W(1)-1\bigr|^2\bigr).$$

Use preconditioned Crank-Nicolson proposals and accept or reject with the appropriate Metropolis-Hastings probability.

5 Math (Karhunen-Loève).

In Monte Carlo, we have the amazing ability to sample *random functions* in addition to random variables and random vectors. Let's say we want to sample a standard Brownian motion $W(t)$ for $0 \le t \le 1$. There's an inefficient and an efficient way to do this.

1. We could sample at discrete time points $W(0), W(\Delta), W(2\Delta), \ldots, W(1)$ and use linear interpolation to approximate $W(t)$ for $i\Delta < t < (i+1)\Delta$. (*Challenge*: what's the conditional

distribution of $W(t)$ given $W(0), W(\Delta), W(2\Delta), \ldots, W(1)$?) But this leads to a high mean square error

$$\int_0^1 \mathbb{E}|W(t) - \hat{W}(t)|^2 \, dt\,.$$

Calculate the mean square error mathematically or empirically.

2. A smarter option is to use the eigenfunction expansion

$$W(t) = \sum_{k=1}^{\infty} \langle e_k, W \rangle e_k(t),$$

where $e_k(t) = \sqrt{2} \sin((k - 1/2)\pi t)$ are orthonormal basis functions with respect to a standard Gaussian distribution and the coefficients $\langle e_k, W \rangle$ are independent standard Gaussians with variance $1/((k-1/2)\pi)^2$. We can sample the first L coefficients (called the "Karhunen-Loeéve coefficients") and approximate $W(t)$ using the truncated expansion

$$\hat{W}(t) = \sum_{k=1}^{L} \langle e_k, W \rangle e_k(t).$$

Calculate the mean square error mathematically or empirically. *Challenge:* Prove the mean square error is as low as possible, for any L-element basis expansion.

6 Math (Parallel tempering analysis).

Let us consider parallel tempering for a 3-state system with pmf $\pi^T = \big((1-\alpha)/2 \quad \alpha \quad (1-\alpha)/2\big)$. We can target a small value of $\alpha = 10^{-10}$ by sampling a sequence of pmfs with $\alpha_1 > \alpha_2 > \cdots > \alpha_N = 10^{-10}$ with MCMC samplers

$$\mathbb{P} = \begin{pmatrix} 1 - \alpha_i/(1-\alpha_i) & \alpha_i/(1-\alpha_i) & 0 \\ 1/2 & 0 & 1/2 \\ 0 & \alpha_i/(1-\alpha_i) & 1 - \alpha_i/(1-\alpha_i) \end{pmatrix}.$$

Describe the resulting parallel tempering algorithm. What is the best sequence of α_i values? What is the best possible decorrelation rate?

7 Math (Preconditioned Crank-Nicolson).

Calculate the decorrelation rate for the sampler in problem #4.