Cognitive Modeling: Homework Assignment 2
Stochastic Process Models and Bayesian Estimation

Caleb Carr & Maryellen Marino

March 17, 2024

All answers and solutions to non-programming questions should be submitted to LMS as a **legible** write-up (either fully digital or a scan). All code should be committed to and merged into the main branch of your team's GitHub repository.

Team GitHub: `https://github.com/maryellenmarino/Cognitive_ModelingS24`

Problem 1: True-False Questions (6 points)

Mark all statements which are **FALSE**

1. The solution of the stochastic integral $\int_0^T \mu \, dW_t$ is $\mu(W_T - W_0)$ and is a random variable itself.

   True. Integration with respect to the Wiener process is a random variable, as long as μ is a constant value.

2. The variance of a Wiener process with scale coefficient $\sigma = 1$ at time t is t^2.

   False. The variance of a Wiener process at time t with $\sigma = 1$ is t, not t^2.

3. The standard Drift-Diffusion Model (DDM) assumes that evidence about a dominant alternative accumulates in discrete chunks over time.

   False. The DDM assumes continuous accumulation of evidence, not discrete chunks.

4. The first passage time distribution has a closed-form probability density function, but its density can still be evaluated only numerically.

   False. The first passage time distribution generally does not have a closed-form probability density function and requires numerical methods for evaluation.

5. The Euler-Maruyama method can only be used to simulate linear stochastic systems.

   False. The Euler-Maruyama method can be applied to both linear and nonlinear stochastic differential equations.

6. For any Bayesian analysis, the prior will always have a smaller variance than the posterior.

   False. The variance of the posterior can be smaller or larger than that of the prior, depending on the data and the strength of the prior.

7. In addition to good statistical practices, experimental validation of cognitive models is crucial for ensuring construct validity.

   True. Experimental validation is indeed important for confirming that cognitive models represent the constructs they are intended to measure.

8. The Leaky Competing Accumulator (LCA) model is a special case of the more general Drift-Diffusion Model (DDM).

   False. The LCA and DDM are distinct models with different assumptions and are not in a special case-general case relationship.

9. Markov chain Monte Carlo (MCMC) methods approximate a complex posterior distribution through a simpler, yet analytically tractable, distribution.

   False. MCMC methods sample from the posterior distribution and do not approximate it with a simpler distribution.

10. The Monte Carlo Standard Error (MCSE) in the context of MCMC estimation can be computed by dividing the standard deviation of the chains by the square root of the total number of samples.

    False. This should be out of only the effective samples rather than the total samples.

11. For most Bayesian problems, the more data we collect, the less influence does the prior exert on the resulting inferences.

    True. More data typically diminishes the influence of the prior as the likelihood becomes more informative.

12. The effective sample size (ESS) estimated from MCMC samplers differs from the total number of samples because the samples are not independent (i.e., exhibit non-zero autocorrelation).

    True. ESS accounts for autocorrelation and estimates the number of independent samples equivalent to the correlated samples from MCMC.

Problem 2: Diffusion Model Explorations (8 points)

As extensively discussed in class, the drift-diffusion model (DDM) generates two response time (RT) distributions, one for each boundary (i.e., lower and upper boundaries). This exercise asks you to first explore a somewhat counterintuitive question about the basic DDM: What differences between the means of the two RT distributions does the the model predict?

To approach this question from a simulation-based perspective, you need to repeatedly solve the forward problem with different parameter configurations and collect the two summary statistics, namely, the two empirical means of the resulting RT distributions. First, choose a suitable configuration of the four parameters and vary only the drift rates within a reasonable range (e.g., $v \in [0.5 - 1.5]$) for a total of 25 different drift rates. Make sure that your parameterizations can generate a sufficient number of RTs for both distributions and you don't end up with the process only reaching the upper boundary. Second, for each of your parameter configurations, generate $N = 2000$ synthetic observations and estimate the means of the two distributions. What do you observe regarding the mean difference? Describe and interpret your results. (4 points)

In a similar spirit (keeping all parameters fixed and varying one), explore the effects of each of the parameters on the means and standard deviations of the simulated RT distributions, quantify and describe your results. (4 points)

Solution. The solution is present in `https://github.com/maryellenmarino/Cognitive_ModelingS24/blob/main/hw2/diffusion_model_explorations.ipynb` on the team GitHub.

Problem 3: Feedforward Inhibition Model (10 + 2 points)

The Feedforward Inhibition (FFI) model is a somewhat simplified version of the Leaky Competing Accumulator (LCA) model for multialternative decision making. It features no decay parameters and assumes only forward inhibition. The forward model of the FFI is given by the following stochastic differential equation (SDE)

$$dX_i = (v_i - I_f(i))dt + \sigma dW_t \tag{1}$$

$$I_f(i) = \frac{\alpha}{J-1}\sum_{j \neq i}^{J} v_j \tag{2}$$

$$X_{it} = \max\{X_{it}, 0\} \tag{3}$$

where the symbols denote:

- J – the number of alternatives (equivalently, the number of accumulators).
- v_j – drift rate for the j-th alternative.
- dX_i – infinitesimal change in activation of the i-th accumulator.
- $\alpha \in R_+$ – strength of feedforward inhibition.
- $\sigma \in R_+$ – scale coefficient (typically fixed to 1).

The model also features the necessary threshold parameter a, additive non-decision time τ, and a set of starting points x_{i0} for each accumulator.
Your goals in this exercise are:

1. Implement the model as a stochastic simulator using the Euler-Maruyama method, verify that your implementation can generate reasonably looking response times distributions for two, three, and four alternatives (6 points).
2. Write and document a function that can create pretty visualizations of individual Euler Maruyama runs (i.e., the internal stochastic process responsible for RT generation in a single trial) (2 points).
3. For a fixed parameter configuration, explore systematically the effect of the feedforward inhibition α parameter on the simulated response time (RT) distributions (e.g., mean RT) and describe your results. (2 points)
4. Bonus: Try to make your simulator efficient using numba or explain in detail where and why just-in-time compilation fails. (2 points)

Solution. The solution is present in `https://github.com/maryellenmarino/Cognitive_ModelingS24/blob/main/hw2/Feedforward_Inhibition_Model.ipynb` on the team GitHub.

Problem 4: Prior and Posterior Variance (4 points)

Show that the following identity holds

$$\text{Var}\left[\theta\right] = \mathbb{E}\left[\text{Var}\left[\theta \mid y\right]\right] + \text{Var}\left[\mathbb{E}\left[\theta \mid y\right]\right] \tag{4}$$

Clarification of terms:

1. $\text{Var}\left[\theta\right]$ – Prior variance.
2. $\mathbb{E}\left[\text{Var}\left[\theta \mid y\right]\right]$ – Expected posterior variance.
3. $\text{Var}\left[\mathbb{E}\left[\theta \mid y\right]\right]$ – Variance of posterior mean.

Solution. To show that the identity holds, we will expand the terms on the right side of the identity and simplify.

$$\begin{aligned}
\text{Var}(\theta) &= \mathbb{E}\left[\text{Var}(\theta \mid y)\right] + \text{Var}\left[\mathbb{E}(\theta \mid y)\right] \quad \text{(Starting with the law of total variance)} \\
&= \mathbb{E}\left[\mathbb{E}(\theta^2 \mid y) - (\mathbb{E}(\theta \mid y))^2\right] + \mathbb{E}\left[(\mathbb{E}(\theta \mid y))^2\right] - (\mathbb{E}\left[\mathbb{E}(\theta \mid y)\right])^2 \quad \text{(Expanding the variance terms)} \\
&= \mathbb{E}\left[\mathbb{E}(\theta^2 \mid y)\right] - \mathbb{E}\left[(\mathbb{E}(\theta \mid y))^2\right] + \mathbb{E}\left[(\mathbb{E}(\theta \mid y))^2\right] - (\mathbb{E}(\theta))^2 \quad \text{(Using the linearity of expectation)} \\
&= \mathbb{E}(\theta^2) - (\mathbb{E}(\theta))^2 \quad \text{(Law of Total Expection \& Cancelling and simplifying terms)} \\
&= \text{Var}(\theta) \quad \text{(By the definition of variance)}
\end{aligned}$$

Problem 5: Normal-Normal Model (4 points)

This exercise mimics the approach we took for solving the Beta-Binomial Bayesian butter toast example using a Normal-Normal conjugate model with known variance (i.e., Normal prior, Normal likelihood). You would implement the model in Python (just as we did for the butter toast) and explore the effects of different prior configurations, so you don't necessarily need to analytically derive the results (conjugate models can be easily looked up on the web). However, I recommend that you try it out using pen and paper first!

The prior distribution for the mean is given by (μ_0 and σ_0 are known hyperparameters):

$$p(\mu) = \mathcal{N}(\mu|\mu_0, \sigma_0) \tag{5}$$

The likelihood distribution is given by (the variance σ^2 is assumed to be known):

$$p(y|\mu, \sigma^2) = \mathcal{N}(y|\mu, \sigma^2) \tag{6}$$

Come up with an entertaining use case for the model, conjure up (i.e., simulate) a modest number of "data points", and compute the posterior given a self-chosen configuration for the prior. Visualize the difference between prior and posterior using histograms, as we did in class (4 points).

The posterior, according to https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf is:

$$\mu|x \propto \mathcal{N}\left(\frac{\sigma_0^2}{\sigma^2+\sigma_0^2}x + \frac{\sigma^2}{\sigma^2+\sigma_0^2}\mu_0, \left(\frac{1}{\sigma_0^2}+\frac{1}{\sigma^2}\right)^{-1}\right) \tag{7}$$

Solution. The solution is present in https://github.com/maryellenmarino/Cognitive_ModelingS24/blob/main/hw2/Normal_Normal_Model.ipynb on the team GitHub.

Problem 6: Simple Bayesian Regression With Stan (6 points)

For this exercise, you will write a probabilistic program in `Stan` that can sample from the posterior $p(\alpha, \beta \mid \mathcal{D})$ of a simple (linear) Bayesian regression given a data set $\mathcal{D} = \{x_n, y_n\}_{n=1}^N$. The program should represent the following generative specification:

$$\begin{aligned}
\sigma^2 &\sim \text{Inv-Gamma}(1, 1) \\
\alpha &\sim \mathcal{N}(0, 10) \\
\beta &\sim \mathcal{N}(0, 10) \\
y_n &\sim \mathcal{N}(\alpha + \beta\, x_n, \sigma^2) \quad \text{for } n = 1, \ldots, N,
\end{aligned}$$

which essentially encodes the assumption that an outcome y is predicted from a covariate x via:

$$y_n = \alpha + \beta x_n + \epsilon_n \quad \text{with} \quad \epsilon_n \sim \mathcal{N}(0, \sigma^2) \tag{8}$$

First, simulate a data set with fixed intercept (α), slope (β), and noise (σ) parameters of your choosing, as well as a pre-set number of observations N. For instance, your simulation could look something like this:

```
N = 100
alpha = 2.3
sigma = 2.
slope = 4.
x = np.random.normal(size=N)
y = alpha + slope * x + sigma * np.random.normal(size=N)
```

Then, pass the data to your program, inspect convergence and efficiency diagnostics, and summarize your inferential results both numerically and graphically. How accurate are the posterior means and how much uncertainty is left? (4 points)
Repeat the analysis with ten times as many observations and report what happens to the precision and uncertainty (2 points).

Solution. The solution is present in https://github.com/maryellenmarino/Cognitive_ModelingS24/blob/main/hw2/simple_bayesian_reg.ipynb on the team GitHub.

Problem 7: Estimating the Drift-Diffusion Model (8 points)

In this exercise, we will perform Bayesian estimation of the Drift-Diffusion Model (DDM) given an actual data set. Load the data sample response times.csv, which contains three columns: rt (the response times), choice (the categorical choices), and condition (a condition indicator). There are N = 300 rows in total, corresponding to 300 trials in an experiment.

There were two conditions in this experiment - an easy one (participants had to classify a familiar face in a crowd of 10 people) and a difficult one (participants had to classify a face in a crowd of 100 people). Unfortunately, a colleague of yours has lost the codebook and no longer knows which indicator in the column condition corresponds to which condition.

This is where you step in. Given your knowledge of the interpretation of four key diffusion model parameters, you offer to perform a model-based analysis and disentangle the conditions by estimating the parameters of the diffusion model. Which parameter best reflects task difficulty (i.e., amount of information to be processed)? Indeed, once you answer this question, you automatically know which parameter you need to vary between the conditions to answer the question. Fortunately for you, a template script is already provided.

Your task is to complete the Stan program provided in diffusion model.stan, solve the inverse problem from the data, and determine which indicator corresponds to the unknown difficult condition based on your estimates (6 points). Along the way, you also need to ensure computational faithfulness by reporting and interpreting convergence and efficiency diagnostics (2 points)

The parameter that best represents difficulty of the task is v (drift rate). As evidenced by the model, the Drift Rate for condition 0 (2.660) is clearly much higher than the drift rate for condition 1 (0.519), so it can be said that decision time for condition 0 is faster. Given this, we can say that condition 0 is the easier task.

Solution. The solution is present in `https://github.com/maryellenmarino/Cognitive_ModelingS24/blob/main/hw2/Estimating_Drift_Diffusion.ipynb` on the team GitHub.