Problem Set 2

Due Thursday Mar. 6, 12:30 pm

Comments

- I haven't fully worked these problems myself, so if you run into any strange issues, please post on Ed as there could be mistakes/oversights on my part.
- Please submit as a PDF to Gradescope.
- Please generate the PDF using Quarto. Feel free to work in a Jupyter notebook and then convert to Quarto before rendering to PDF. Other formats that look professional and are designed for working with code and math notation may also be fine - just check with me via a public post on Ed first.
- Remember to note at the start of your document the names of any other students that you worked with on the problem set (or indicating you didn't work with anyone if that was the case) and then indicate in the text or in code comments any specific ideas or code you borrowed from another student or any online reference (including ChatGPT or the like).
- In general, your solution should not just be code - you should have text describing how you approached the problem and what decisions/conclusions you made, though for simple problems, this can be quite short. Your code should have comments indicating what each function or block of code does, and for any lines of code or code constructs that may be hard to understand, a comment indicating what that code does.
- You do not need to (and should not) show exhaustive output, but in general you should show short examples of what your code does to demonstrate its functionality. The output should be produced as a result of the code chunks being run during the rendering process, not by copy-pasting of output from running the code separately (and definitely not as screenshots).
- I do not recommend writing initial answers using a ChatBot, as I think you are likely to fool yourself in terms of how much you are learning about Julia and programming concepts/skills more generally. But it's up to you to decide how heavily to rely on a ChatBot. And refining your initial answers using a ChatBot seems like a good strategy. Using your own knowledge and information online to check the results of a ChatBot and using a ChatBot to check your own coding can both be important/useful.

Problems

- Consider the Gaussian process simulation code in Stat 243, which uses OOP in Python. Convert this to Julia, using structs and functions with type annotation. Make sure to continue to cache information effectively. We'll discuss linear algebra further in Week 6. Include logging and exceptions as you did in PS2.

- positive definiteness - try with Float64 and Float32 on different grids using squared exponential kernel. Show that non p.d. Then try with BigFloat. Compare the timing for doing crossprod/chol for BigFloat vs Float64.
- Have a basic `plot()` method that operates on your GP to plot the simulated time series.
- chol - non-p.d. - have them use slight epsilon
- Allow your code to use GPUs if available on the system (but fall back to the CPU if no GPU is available). Benchmark your code on an SCF GPU (or your own), making sure to report which type of GPU you used for your work. In your time comparisons, also compare to using multiple CPU cores via the threaded BLAS.
- Suppose your users want to generate many time series at once (e.g., for a simulation study). So the output should be a n x m where m is the number of simulations. Use threaded or multi-process parallelization to generate the time series in parallel, using the same underlying cholesky decomposition. [investigate parallel RNG and whether they should just use different seed]