STA 141C - Big Data & High Performance Statistical Computing **Spring 2023**

Homework # 1

Lecturer: Bo Y.-C. Ning *Due April 28, 2023*

Please do not distribute!

This homework has three main goals: 1) familiar with catastrophic cancellation; 2) know how to read data and write a function using R or python; 3) know how to exploit the special structure of a matrix to increase the computational speed.

1. Submit your homework using the file name "**LastName_FirstName_hw1**" on **Canvas**
2. Answer all questions with complete sentences. For proofs, please provide the intermediate steps.
3. To answer question 5-3, writing it using the Markdown (or LaTeX) is preferred. But handwriting is also acceptable. You need to combine it with the coding part and submit a single .html or .pdf file.
4. Your code should be readable; writing a piece of code should be compared to writing a page of a book. Adopt the **one-statement-per-line** rule. Consider splitting a lengthy statement into multiple lines to improve readability. (You will lose one point for each line that does not follow the one-statement-per-line rule) Please read this: `https://docs.python-guide.org/writing/style/` and this: `https://irudnyts.github.io/r-coding-style-guide/`.
5. To help understand and maintain code, you should always add comments to explain your code. (homework with no comments will receive 0 points). For a very long comment, break it into multiple lines.
6. Submit your final work with one **.pdf** (or **.html**) file to Canvas. I encourage you to use LaTeX for writing equations and proofs. Handwriting is acceptable, you have to scan it and then combine it with the coding part into one .pdf (or .html) file. Handwriting should be clean and readable.
7. For Jupyter Notebook users, put your answers in new cells after each exercise. You can make as many new cells as you like. Use code cells for code and Markdown cells for text.
8. This assignment will be graded for correctness.

1 Programming exercises

1. The following facts about triangular matrices are useful for understanding the algorithms that will be mentioned in class. Please create some arbitrary matrices in R or python and verify the following facts:

- The product of two upper triangular matrices is upper triangular.
- The inverse of a lower triangular matrix is lower triangular.
- The product of two unit lower triangular matrices is unit lower triangular.
- The inverse of a unit upper triangular matrix is unit upper triangular.

Note that a unit triangular matrix is a triangular matrix with all diagonal entries being 1.

2. Let $a = 0.7, b = 0.2$, and $c = 0.1$

- Test whether $(a+b)+c$ equals 1;
- Test whether $a+(b+c)$ equals 1;
- Test whether $(a+c)+b$ equals 1;
- Explain what you found. (For example, you can find the internal representation of these numbers.)

3. Create the vector $x = (0.988, 0.989, 0.990, \ldots, 1.010, 1.011, 1.012)$.

- Plot the polynomial $y = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1$ at each point in x
- Plot the polynomial $y = (x-1)^7$ at each point in x
- Explain what you found.

4. Read in the matrix in the file 'oringp.dat' on the failure of O-rings leading to the Challenger disaster. The columns are flight number, date, number of O-rings, number failed, and temperature at launch. Compute the correlation between number of failures and temperature at launch, deleting the last, missing observation (the disaster).

5. (Challenging!) Consider the mixed effect model

$$y_i = x_i'\beta + z_i'\gamma + \epsilon_i, \quad i = 1, \ldots, n,$$

where $\epsilon_i \sim N(0, \sigma_0^2)$ are independent normal errors, $\beta \in \mathbb{R}^p$ are fixed effects, $\gamma \in \mathbb{R}^q$ are random effects assumed to be $N(0_q, \sigma_1^2 I_q)$, γ and ϵ_i are independent. For simplicity, assume $\mu_i = x_i'\beta$. Let $y = (y_1, \ldots, y_n)'$, $\mu = (\mu_1, \ldots, \mu_n)'$, $Z = (z_1, \ldots, z_n)'$, then $y \in \mathbb{R}^n$, $\mu \in \mathbb{R}^n$, $Z \in \mathbb{R}^{n \times q}$, and $y \sim N(\mu, \sigma_1^2 ZZ' + \sigma_0^2 I_n)$. Our goal is to evaluate its log-density function given by

$$-\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(\det(\sigma_1^2 ZZ' + \sigma_0^2 I_n)) - \frac{1}{2}(y-\mu)'(\sigma_1^2 ZZ' + \sigma_0^2 I_n)^{-1}(y-\mu). \tag{1.1}$$

(Notice that this is essentially a multivariate normal density with mean μ and covariance $\sigma_1^2 ZZ' + \sigma_0^2 I_n$; see [wikipedia].)

1. Choose n (≥ 100) and $q = 5$, randomly choose a value for σ_0, σ_1, and a vector μ, a matrix Z respectively. Use them to generate y (note that y can be generated from a multivariate normal density $N(\mu, \sigma_1^2 ZZ' + \sigma_0^2 I_n)$; y is a $n \times 1$ vector. You should use set.seed function in R or np.random.seed function in numpy before generating y).
2. Use the default package in R or numpy to evaluate the log-density function (e.g., in R, use dmvnorm(y y, mu = mu, Sigma = Sigma, log = TRUE)). Note that: Sigma = $\sigma_1^2 ZZ' + \sigma_0^2 I_n$.
3. Apply the Woodbury formula

$$(A + UV')^{-1} = A^{-1} - A^{-1}U(I_m + V'A^{-1}U)^{-1}V'A^{-1},$$

where $A \in \mathbb{R}^{n \times n}$, $U, V \in \mathbb{R}^{n \times m}$ to rewrite $(\sigma_1^2 ZZ' + \sigma_0^2 I_n)^{-1}$ and apply the matrix determinant lemma

$$\det(A + UV') = \det(A)\det(I_m + V'A^{-1}U)$$

to re-write $\det(\sigma_1^2 ZZ' + \sigma_0^2 I_n)$. Plugging-in the two terms and rewrite the log-density function in (1.1).

4. We now ready to write a function to evaluate the log-density by yourself using the formula you derived in question 3. Call this function

   dmvnorm_lowrank(y, mu, Z, sigma0, sigma1, log = FALSE).

   The inputs of the function should be y, μ, Z, σ_0, and σ_1 and the output of this function should be the density function if log = FALSE and should be the log-density function if log = TRUE. The default output of the function should be log = FALSE.

5. Test you function by plugging-in the same values of y, μ, Z, σ_0, and σ_1 in (1) into the dmvnorm_lowrank function. The output should be the same as in question 2.

6. Compare the computational speeds between your function and the default package used in question 2. Comment on your findings.

7. (extra credit, 1 point) Increase n (but keep q the same) and try questions 1-6 again. Comment on your findings.

6. In class we learnt about the BLAS and how it has become a de facto standard for basic linear algebra operations. Both R and numpy use the BLAS and LAPACK libraries extensively to accelerate certain types of operations. Apart from *, %*%, eigen, and qr, find out two other common functions use the BLAS and LAPACK routines to speed up calculations.