Quiz 1

Instructions:

- In this quiz you are asked to program and test a few functions as described below.
- The quiz is open-book, so feel free to use any code posted on LEARN.
- At the end of class, submit your quiz on LEARN as follows:
  1. Submit *only* two files: **`uwname-functions.R`** and **`uwname-quiz1.R`**, where **`uwname`** is your UW username (so for me it's **`mlysy`**).
  2. To upload these files to Learn, navigate to `Assessments/Dropbox`, then click on `Quiz 1`.
- Do as much of the quiz as possible in class. If you wish to improve your submission after class, you have until Tuesday January 16 at 11:59pm to submit a second version on LEARN (same instructions as above).
- While the first version is individual, the second version may be done in groups. However, you **must include the names of all collaborators** in the `Comments` field during the LEARN file upload process.
- Organized, well-commented, and efficient code is required for full marks.

Q1. Let $X \sim \mathcal{N}(\mu, \Sigma)$ be and n-dimensional multivariate normal random variable. Let $X = (X_1, X_2)$ with $X_1 \in \mathbb{R}^p$, $X_2 \in \mathbb{R}^q$, such that $n = p + q$. Then in block notation the distribution of X is expressed as

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim \mathcal{N}\left\{\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}\right\}.$$

Moreover, we have the following marginal and conditional distributions:

$$\begin{aligned} X_1 &\sim \mathcal{N}(\mu_1, \Sigma_{11}) \\ X_2 \mid X_1 &\sim \mathcal{N}(\mu_2^\star, \Sigma_2^\star), \qquad \begin{aligned} \mu_2^\star &= \mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(X_1 - \mu_1) \\ \Sigma_2^\star &= \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}. \end{aligned} \end{aligned} \tag{1}$$

In this question, you are asked to verify this result numerically.

(a) In the file **`uwname-functions.R`**, write a function which computes the conditional mean and variance $\mu_2^\star$ and $\Sigma_2^\star$. The function should have *exactly* the following name and argument signature,

```
cmvn <- function(mu, Sigma, x1, ind1)
```

where `mu` = μ, `Sigma` = Σ, `x1` = X_1, and `ind1` is a vector of `TRUE`/`FALSE` values of length n with exactly p `TRUE` values, indicating which of the elements of (μ, Σ) correspond to X_1[1]. The output of the function *must* be a list with elements named `cmu2` and `cSigma2` (the "`c`" stands for "conditional"), corresponding to $(\mu_2^\star, \Sigma_2^\star)$. For full marks, inversion of variance matrices must be performed efficiently.

Hint: If `M` is a matrix, the command `M[ind1,ind2,drop = FALSE]` prevents the result from being downcast to a vector/scalar if `ind1` and/or `ind2` are of length 1.

(b) In the file **`uwname-quiz1.R`**, write a script to verify that

$$\log \varphi(X \mid \mu, \Sigma) = \log \varphi(X_1 \mid \mu_1, \Sigma_{11}) + \log \varphi(X_2 \mid \mu_2^\star, \Sigma_2^\star),$$

where $\varphi(\cdot \mid \mu, \Sigma)$ is the PDF of $X \sim \mathcal{N}(\mu, \Sigma)$. For the multivariate normal PDF, you can use either the function `dmvn` provided in the course notes (in **`mvn-functions.R`**), or the function `dmvnorm` in the **R** package **`mvtnorm`**. To minimize the chance of "double-cancellation" errors, your script should generate random values of n, X, μ, Σ, and `ind1`.

[1] While it's easier to code the function such that `x1` are simply the first elements of `mu`, the extra effort of using `ind1` here makes the function far easier to use, as you will see in the next question.

Q2. Recall that Brownian motion $B_t, t \geq 0$ is a continuous Gaussian process with $B_0 = 0$, $E[B_t] = 0$, and $\mathrm{cov}(B_s, B_t) = \min(s, t)$. This means that for any finite sequence of time points

$$\boldsymbol{t} = (t_1, \ldots, t_N), \qquad 0 < t_1 < \cdots < t_N,$$

the *path skeleton* at these time points $\boldsymbol{B_t} = (B_{t_1}, \ldots, B_{t_N})$ is multivariate normal:

$$\boldsymbol{B_t} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}), \qquad \boldsymbol{\Sigma}_{ij} = \min(t_i, t_j). \tag{2}$$

In class, we saw that the path skeleton above can be made more precise in the following sense. Consider the intermediate time sequence

$$\boldsymbol{s} = (s_1, \ldots, s_N), \qquad 0 < s_1 < t_1 < s_2 < t_2 < \cdots < s_N < t_N,$$

and the corresponding Brownian motion values $\boldsymbol{B_s} = (B_{s_1}, \ldots, B_{s_N})$. Then we have the conditional distribution

$$\boldsymbol{B_s} \,|\, \boldsymbol{B_t} \sim \mathcal{N}\Big(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2)\Big),$$

where

$$\mu_i = \sigma_i^2 \left(\frac{B_{t_{i-1}}}{s_i - t_{i-1}} + \frac{B_{t_i}}{t_i - s_i} \right), \qquad \sigma_i^2 = \left(\frac{1}{s_i - t_{i-1}} + \frac{1}{t_i - s_i} \right)^{-1}, \tag{3}$$

and we have defined $t_0 = 0$. In this question, you are asked to verify this conditional distribution numerically.

(a) In the file **`uwname-functions.R`**, write a function which computes the variance matrix of $\boldsymbol{B_t}$. This function should have *exactly* the following name and argument signature,

```
bmV <- function(tseq)
```

where `tseq` = $\boldsymbol{t}$, and the output is the $N \times N$ variance matrix $\boldsymbol{\Sigma}$ defined by (2). For full marks, do not compute this variance matrix with a for-loop. Instead, use the **R** functions `outer` and `pmin` (of which the latter computes the elementwise minimum between two arrays).

(b) In the file **`uwname-functions.R`**, write a function to compute the conditional mean and standard deviations $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ defined by (3). The function should have *exactly* the following name and argument signature,

```
cbm <- function(sseq, tseq, Bt)
```

where `sseq` = $\boldsymbol{s}$, `tseq` = $\boldsymbol{t}$, and `Bt` = $\boldsymbol{B_t}$. The output of the function *must* be a list with elements `cmu` and `csigma`, corresponding to (μ, σ).

(c) In the file **`uwname-quiz1.R`**, write a script to verify that formula (3) is correct. That is, for arbitrary (but consistent) time points t and s:

- Use the function `bmV` to generate the variance matrix of (B_t, B_s).
- Use the function `cvmn` from **Q1** to calculate the conditional mean and variance of $B_s \mid B_t$ using the general multivariate normal result (1).
- Check that this mean and variance are exactly the same as those returned by `cbm`.

Your script should do this with random values of N, s, t, and B_t.

Hint: You can generate consistent timepoints s and t as follows:

```
N <- sample(2:10, 1) # number of timepoints
# generate consistent sseq and tseq
stseq <- cumsum(rexp(2*N)) # generate both simultaneously
sseq <- stseq[seq(from = 1, to = 2*N, by = 2)] # odd elements
tseq <- stseq[seq(from = 2, to = 2*N, by = 2)] # even elements
```