Solutions for problems in part B

1 May2023: PartB: Higher grades-methods (4 points)

In terms of runtime for large matrices we expect:
Normal equations < QR approach < SVD approach.
Further, given the very high condition number of A, we expect that we get very bad results when using the normal equations. Therefore, we conclude that

- 2(D): Method 2 = (D) Setting up and solving the normal equations.

Further, we expect SVD to produce an accurate result and to be the most expensive one. Therefore, we conclude that

- 4(C): Method 4 = (C) Computing using an SVD.

Distinguishing between the 2 QR-approaches, we expect Householder to be computationally more stable than Gram-Schmidt, resulting in potentially better accuracy. Therefore,

- 1(B): Method 1 = (B) QR using Householder
- 3(A): Method 3 = (A) QR using Gram-Schmidt

In total we get

- 1(B)
- 2(D)
- 3(A)
- 4(C)

Note: In case the runtimes were misread as only runtime(Method 2) < runtime(Method 1) and runtime(Method 3) < runtime(Method 4), as it is much obvious that SVD is faster than QR than to compare the runtimes of the different QR approches and therefore the above mapping makes still most sense.

2 Pareto (6 points)

Between x and u, there is the relationship $F(x) = u$, i.e. (note that for all samples $x > 1$)

$$1 - x^{-\alpha} = u. \tag{1}$$

Computing the inverse function gives

$$x = F^{-1}(u) = \left(\frac{1}{1-u}\right)^{\frac{1}{\alpha}}.$$

Then taking the logarithm on both sides gives

$$\ln(x) = \frac{1}{\alpha} \ln\left(\frac{1}{1-u}\right) = -\frac{1}{\alpha}\ln(1-u).$$

Alternatively, directly operating on (1) gives the same result:

$$1 - u = x^{-\alpha} \quad \Leftrightarrow \quad \ln(x^{-\alpha}) = \ln(1-u) \quad \Leftrightarrow \quad -\alpha\ln(x) = \ln(1-u).$$

We now search α such that

$$\|\ln(1-u) + \alpha\ln(x)\|^2$$

is minimized. This results in the overdetermined problem

$$\underbrace{\begin{pmatrix} \ln 1.1 \\ \ln 3.18 \\ \ln 1.36 \\ \ln 2.11 \\ \ln 1.21 \end{pmatrix}}_{A} (-\alpha) = \underbrace{\begin{pmatrix} \ln(1-0.13) \\ \ln(1-0.82) \\ \ln(1-0.36) \\ \ln(1-0.67) \\ \ln(1-0.25) \end{pmatrix}}_{b}$$

Using python, we solve this to get $\alpha = 1.48$

```
import numpy as np

x = np.array([1.1,3.18,1.36,2.11,1.21])
u = np.array([0.13,0.82,0.36,0.67,0.25])
A = np.log(x)
ATA = np.matmul(np.transpose(A),A)
ATb = np.matmul(np.transpose(A),np.log(1-u))
neg_alpha = ATb/ATA
print(f'-alpha',neg_alpha)
```

3 SVD problem

3.1 Compressing data

Proposed solution:

1. We can use the SVD, i.e., compute U, Σ, V^T such that $A = U\Sigma V^T$. As the rank of the matrix A is only 10, we know that $\sigma_{11} = \ldots \sigma_{250} = 0$. Therefore, by Eckart-Theorem we know that the rank-100-approximation to A is the same as A itself.

2. With $A = U\Sigma V^T$, we compute $A_{10} = \sum_{k=1}^{10} u_k \sigma_k v_k^T$ with u_k and v_k denoting the columns of U and V. Therefore, $u_k \in \mathbb{R}^{10^3}$ and $v_k \in \mathbb{R}^{250}$. In addition we need to store the singular values. Therefore, the total demand is

$$(10^3 + 250 + 1) * 10 = 12,510 \text{ elements.}$$

As each element needs 8 bytes, this gives $8 * 12,510$ bytes = 100KB+80B.

3.2 Finding b

For a positive definite (and in particular symmetric) matrix, the singular values coincide with the eigenvalues. The given matrix is symmetric. Further, as its upper left entry is positive and its determinant is positive for the given span of values for b, it is positive definite:

$$9 - b^2 > 0 \quad \Leftrightarrow \quad -3 < b < 3.$$

We can therefore compute the eigenvalues and find

$$(\lambda - 1)(\lambda - 9) - b^2 = \lambda^2 - 10\lambda + 9 - b^2$$

Setting this to zero gives

$$\lambda_{1,2} = \frac{10 \pm \sqrt{64 + 4b^2}}{2} = 5 \pm \sqrt{16 + b^2}.$$

Therefore, for the larger singular value σ_1 we get

$$\sigma_1 = 5 + \sqrt{16 + b^2} \geq 9$$

and for the smaller singular value σ_2 we get

$$\sigma_2 = 5 - \sqrt{16 + b^2} \overset{!}{\geq} 1 \quad \Rightarrow b = 0.$$