Lucas Abdalah
[TI8419 - Multilinear Algebra] Homeworks
Professors: André Lima e Henrique Goulart

Contents

Homework 0 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Problem 1

For randomly generated $\mathbf{A} \in \mathbb{C}^{N \times N}$ and $\mathbf{B} \in \mathbb{C}^{N \times N}$, evaluate the computational performance (run time) of the following matrix inversion formulas:

(a)

Method 1:

$$(\mathbf{A}_{N \times N} \otimes \mathbf{B}_{N \times N})^{-1}$$

Method 2:

$$(\mathbf{A}_{N\times N})^{-1} \otimes (\mathbf{B}_{N\times N})^{-1}$$

For $n \in \{2, 4, 6, 8, 16, 32, 64\}$.

Results

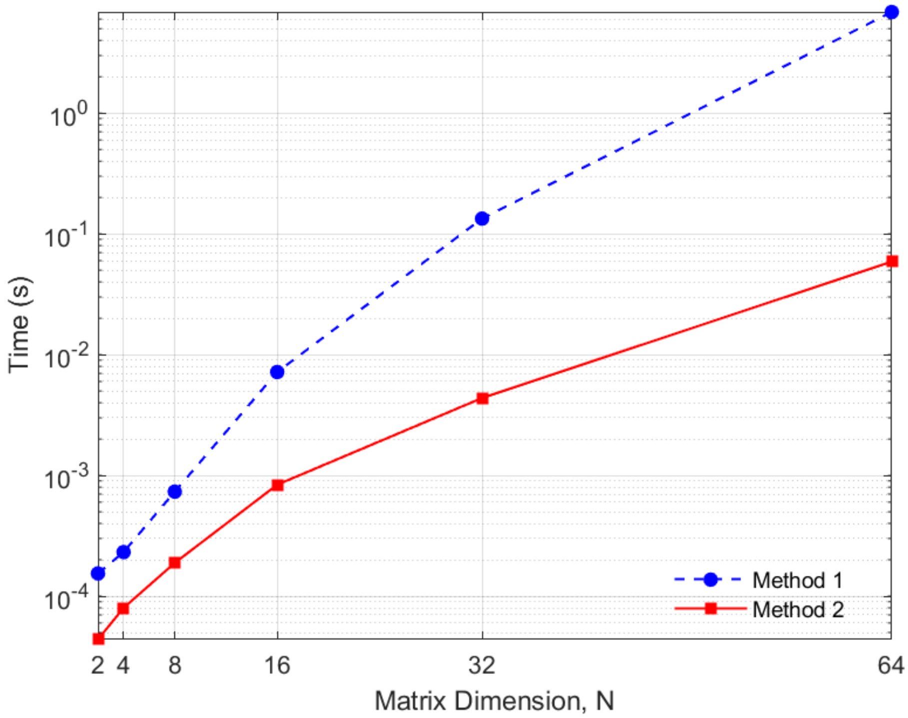
Simulation setup

- 100 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- Compute the mean for each value, for $N = 2, 4, 6, 8, 16, 32, 64$.

Discussion

We can see that for all values of N, the second method outperforms the first. For small values of N, the difference is more subtle, ten times faster. However as the N increases, the performance gap increases up to a hundred times faster.

Problem 1.a script



(b)

Method 1:

$$\left(\mathbf{A}^{(1)}_{N\times N} \otimes \mathbf{A}^{(2)}_{N\times N} \otimes \mathbf{A}^{(3)}_{N\times N} \otimes \cdots \otimes \mathbf{A}^{(K)}_{N\times N}\right)^{-1} = \left(\bigotimes_{k=1}^{K} \mathbf{A}^{(k)}_{N\times N}\right)^{-1}$$

Method 2:

$$\left(\mathbf{A}^{(1)}_{N\times N}\right)^{-1} \otimes \left(\mathbf{A}^{(2)}_{N\times N}\right)^{-1} \otimes \left(\mathbf{A}^{(3)}_{N\times N}\right)^{-1} \otimes \cdots \otimes \left(\mathbf{A}^{(K)}_{N\times N}\right)^{-1} = \bigotimes_{k=1}^{K} \left(\mathbf{A}^{(k)}_{N\times N}\right)^{-1}$$

For $k \in \{2, 4, 6, 8, 10\}$.

Results

Simulation setup

- 200 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- Compute the mean for each value for $N = 2$ and $K = 2, 4, 6, 8, 10$.

Discussion

On the scenario proposed, with $N = 4$, the amount of memory (ram) is up to greater than 64.0 Gb. Since a single complex element requires 16 bytes, the simulation using the homework setup fails from K = 8, since it's required more RAM memory than the available, 19.8 Gb. This value consider 100% of ram use, without taking into count the operational system (OS), backgroud scripts or matlab.

Example

To illustrate, the function kron_dim may be applied for the example with $N = 4$ $k = 7$:

```
Matrix Dimensions: 16384X16384
N of elements: 268435456
Memory use:  4 Gb
```

Since each matrix is 4×4, each Kronnecker product multiplies by 16 the amount of RAM required, hence the matrix product with $K = 8$ leads it to an error.

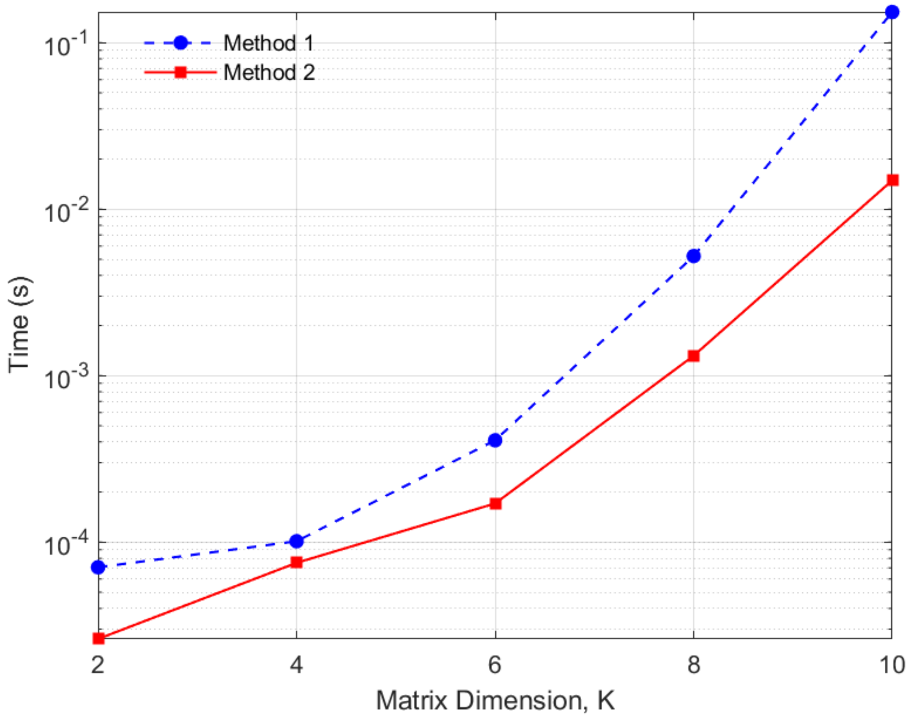
```
Requested 4x16384x4x16384 (64.0GB) array exceeds maximum array size
preference (19.8GB). This might cause MATLAB to become unresponsive.
```

Finally, we set $N = 2$ for maximum usage when $K = 10$, since it leads to a $2^{10} \times 2^{10}$ matrix, with 1048576 elements and only 16 Mb of ram use.

```
Matrix Dimensions: 1024X1024
N of elements: 1048576
Memory use: 16 Mb
```

We can see that for all values of K, the second method outperforms the first. Both results support the hypothesis that the inversion of smaller matrices in Matlab is much more effective.

Problem 1.b script



Problem 2

Let $\text{eig}(\mathbf{X})$ be the function that returns the matrix $\sum_{K\times K}$ of eigenvalues of $\mathbf{X}$. Show algebraically that $\text{eig}(\mathbf{A}\otimes\mathbf{B}) = \text{eig}(\mathbf{A})\otimes\text{eig}(\mathbf{B})$.

Hint: Use the property $(\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\otimes\mathbf{D}) = \mathbf{AC}\otimes\mathbf{BD}$

We write the SVD for each matrix, $\mathbf{A}$ and $\mathbf{B}$, as:

$$\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^H$$
$$\mathbf{B} = \mathbf{U}_B\mathbf{\Sigma}_B\mathbf{V}_B^H,$$

We take advantage of the definitions to the equation $\text{eig}(\mathbf{A}\otimes\mathbf{B})$ and using two times the property suggested by the exercise, we have:

$$\begin{aligned}\text{eig}\left(\mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^H\otimes\mathbf{U}_B\mathbf{\Sigma}_B\mathbf{V}_B^H\right) &= \text{eig}\left[(\mathbf{U}_A\otimes\mathbf{U}_B)(\mathbf{\Sigma}_A\mathbf{V}_A^H\otimes\mathbf{\Sigma}_B\mathbf{V}_B^H)\right]\\ &= \text{eig}\left[(\mathbf{U}_A\otimes\mathbf{U}_B)(\mathbf{\Sigma}_A\otimes\mathbf{\Sigma}_B)(\mathbf{V}_A\otimes\mathbf{V}_B)^H\right]\\ &= \mathbf{\Sigma}_A\otimes\mathbf{\Sigma}_B = \text{eig}(\mathbf{A})\otimes\text{eig}(\mathbf{B}),\end{aligned}$$

by applying the operator $\text{eig}(\cdot)$ that returns the eigenvalue matrix $\mathbf{\Sigma}_A\otimes\mathbf{\Sigma}_B$.

Return to Table of Contents

Table of Contents

Problem 1

For randomly generated $\mathbf{A}$ and $\mathbf{B} \in \mathbb{C}^{N \times N}$, create an algorithm to compute the Hadamard Product $\mathbf{A} \odot \mathbf{B}$. Then, compare the run time of your algorithm with the operator A.*B of the software Octave/Matlab ®. Plot the run time curve as a function of the number of rows/columns $N \in \{2, 4, 8, 16, 32, 64, 128\}$.

Results

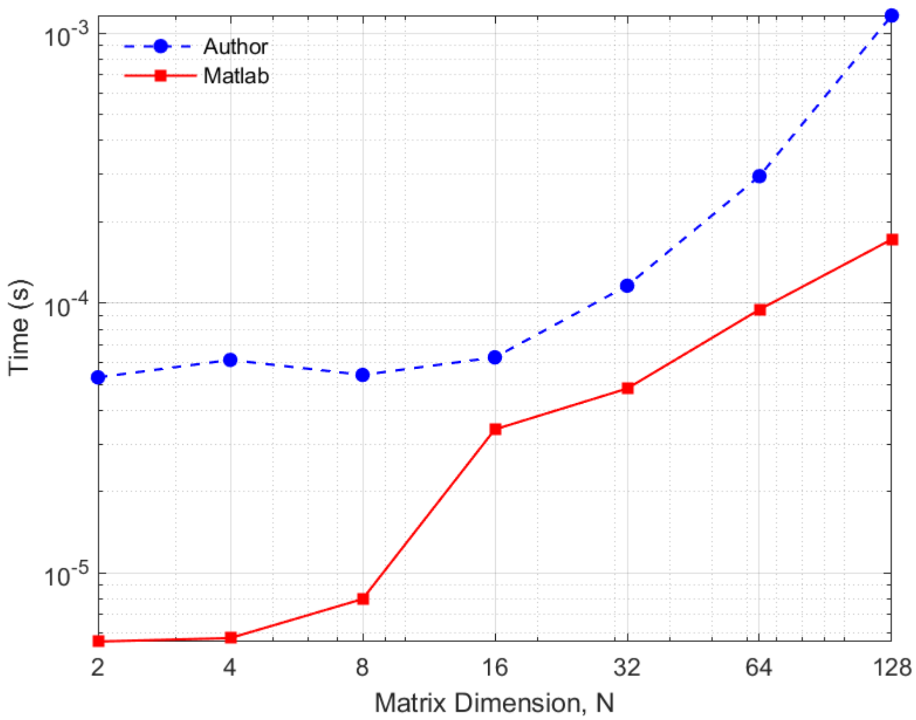
Simulation setup

- 500 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- Compute the mean for each value, for $N = \{2, 4, 6, 8, 16, 32, 64, 128\}$.

Discussion

We can see that for all values of N, Matlab's method outperforms the Author's. For small values of N, the gap between them, 6×10^{-5}s vs 6×10^{-6}s, approximately ten times faster. However as the N increases, that performance gap becomes more subtle.

Problem 1 script



Problem 2

For randomly generated $\mathbf{A}$ and $\mathbf{B} \in \mathbb{C}^{N \times N}$, create an algorithm to compute the Kronecker Product $\mathbf{A} \otimes \mathbf{B}$. Then, compare the run time of your algorithm with the operator kron(A, B) of the software Octave/Matlab ®. Plot the run time curve as a function of the number of rows/columns $N \in \{2, 4, 8, 16, 32, 64, 128\}$.

Results

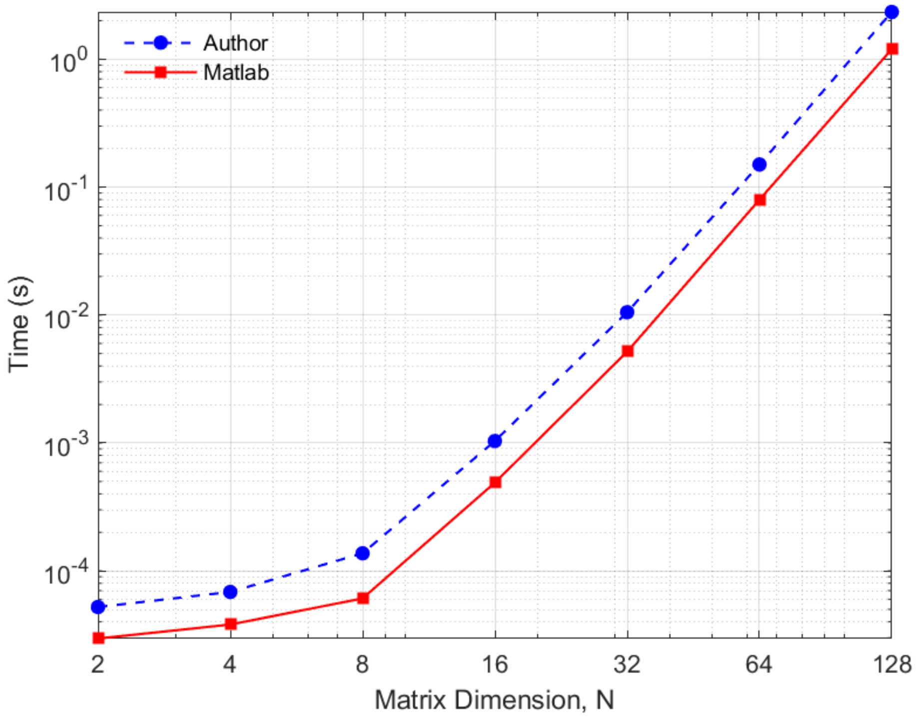
Simulation setup

- 500 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\, 1)$;
- Compute the mean for each value, for $N = \{2, 4, 6, 8, 16, 32, 64, 128\}$.

Discussion

We can see that for all values of N, Matlab's method outperforms the author's. There's a narrow performance gap between them, up to three times faster. The difference varies very little regardless the value of N increase.

Problem 2 script



Problem 3

For randomly generated $\mathbf{A}$ and $\mathbf{B} \in \mathbb{C}^{N \times N}$, create an algorithm to compute the Khatri-Rao Product $\mathbf{A} \diamond \mathbf{B}$ according with the following prototype function:

$$R = kr(A, B).$$

Results

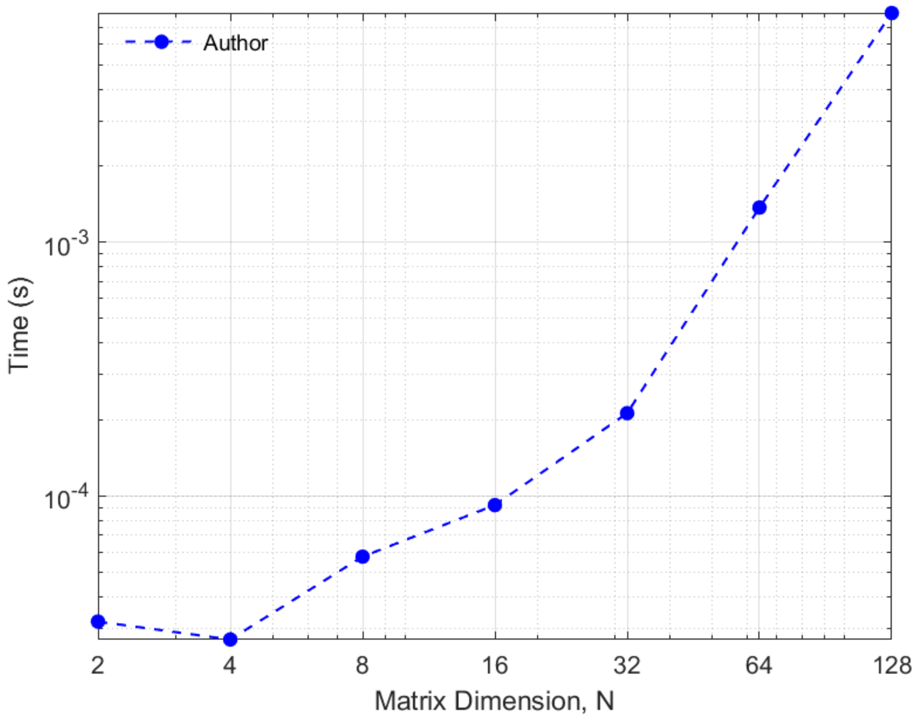
Simulation setup

- 500 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- Compute the mean for each value, for $N = \{2, 4, 6, 8, 16, 32, 64, 128\}$.

Discussion

The method developed by the author present similar behavior to Kronnecker product and a predictable trend for all values of N.

Problem 3 script



Return to Table of Contents

Homework 2 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

- Problem 2

Problem 1

Generate $\mathbf{X} = \mathbf{A} \diamond \mathbf{B} \in \mathbb{C}^{I \times R}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{I \times R}$ and $\mathbf{B} \in \mathbb{C}^{I \times R}$. Compute the left pseudo-inverse of $\mathbf{X}$ and obtain a graph that shows the run time vs. number of rows (I) for the following methods.

Method 1:

Matlab/Octave function: $pinv(\mathbf{X}) = pinv(\mathbf{A} \diamond \mathbf{B})$

Method 2:

$$\mathbf{X}^{\dagger} = (\mathbf{X}^{\top}\mathbf{X})^{-1}\mathbf{X}^{\top} = [(\mathbf{A} \diamond \mathbf{B})^{\top}(\mathbf{A} \diamond \mathbf{B})]^{-1}(\mathbf{A} \diamond \mathbf{B})^{\top}$$

Method 3:

$$\mathbf{X}^{\dagger} = [(\mathbf{A}^{\top}\mathbf{A}) \odot (\mathbf{B}^{\top}\mathbf{B})]^{-1}(\mathbf{A} \diamond \mathbf{B})^{\top}$$

Note: Consider the range of values $I \in \{2, 4, 8, 16, 32, 64, 128, 256\}$ and plot the curves for $R = 2$ and $R = 4$.

Results

Simulation setup

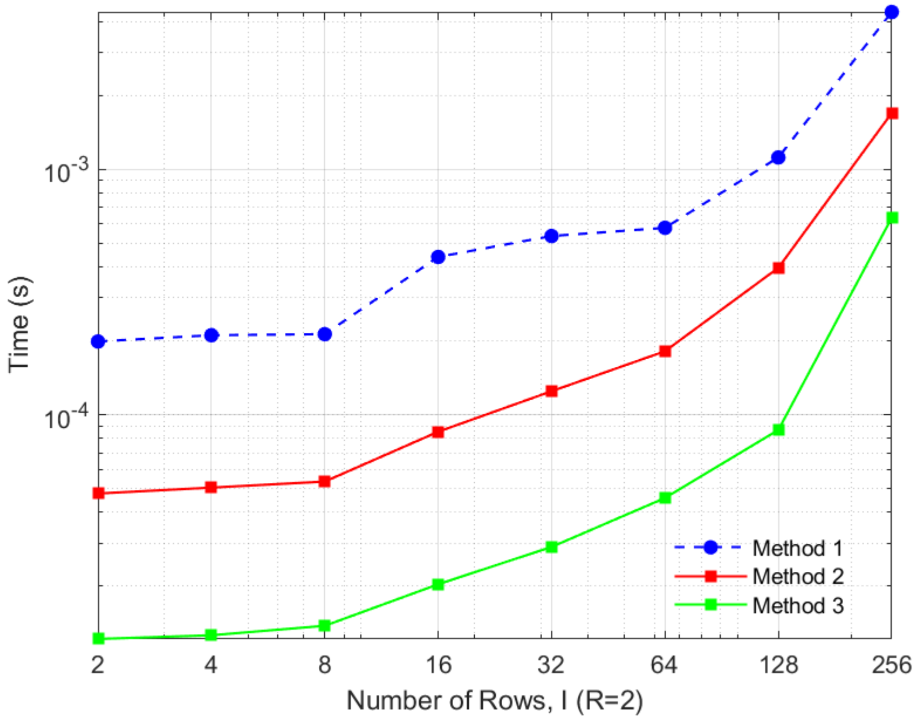
- 500 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- Compute the mean for each value, for $N = \{2, 4, 6, 8, 16, 32, 64, 128, 256\}$.

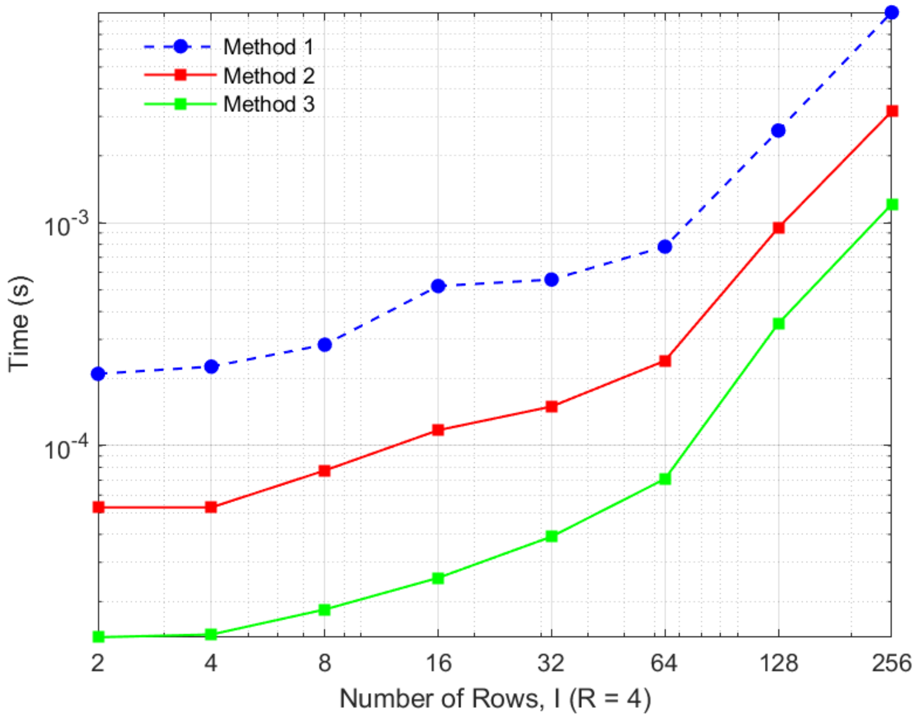
Discussion

We can see that for all values of I, Matlab's method is outperformed by the methods 2 and 3. All methods present a subtle gap between their cost, approximately constant. Method 2 is two times faster then Matlab, while method 3 is ten times faster.

The experiment with $R = 4$ also supports the results presented for $R = 2$, with very similar plots.

Problem 1 script and Figures.





Problem 2

Generate $\overset{N}{\underset{n=1}{\diamond}} \mathbf{A}_{(n)} = \mathbf{A}_{(1)} \diamond \cdots \diamond \mathbf{A}_{(N)}$, where every $\mathbf{A}_{(n)}$ has dimensions 4×2, $n = 1, \ldots, N$. Evaluate the run time associated with the computation of the Khatri-Rao product as a function of the number N of matrices for the above methods.

Note: Consider the range of values $N \in \{2, 4, 6, 8, 10\}$.

The symbols $\odot$ and $\diamond$ denotes the Hadamard and the Khatri-Rao Product, respectively.

Results

Simulation setup

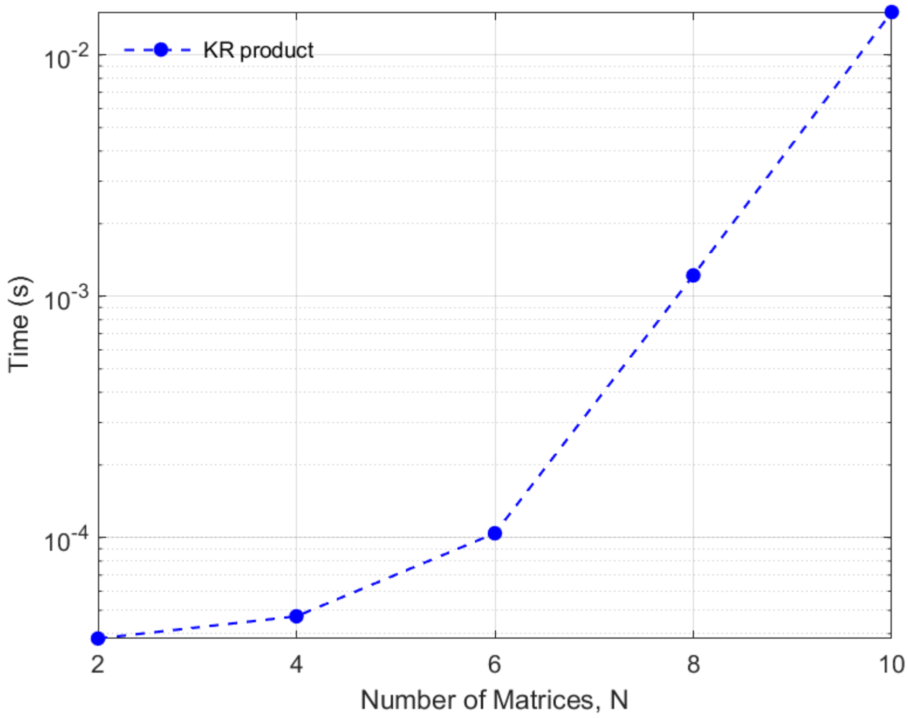
- 500 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- Each matrix has 4×2 dimension;
- Compute the mean for each value, for $N = \{2, 4, 6, 8, 10\}$.

Discussion

The results are consistent with the experiment perfomed in HW1, that for randomly generated $\mathbf{A}$ and $\mathbf{B} \in \mathbb{C}^{N \times N}$, an algorithm to compute the Khatri-Rao Product $\mathbf{A} \diamond \mathbf{B}$ was created according with the following prototype function:

$$R = kr(A, B).$$

Problem 2 script



Return to Table of Contents

Homework 3 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Least-Squares Khatri-Rao Factorization (LSKRF)

Problem 1

Generate $\mathbf{X} = \mathbf{A} \diamond \mathbf{B} \in \mathbb{C}^{20\times 4}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{5\times 4}$ and $\mathbf{B} \in \mathbb{C}^{4\times 4}$. Then, implement the Least-Squares Khatri-Rao Factorization (LSKRF) algorithm that estimate $\mathbf{A}$ and $\mathbf{B}$ by solving the following problem

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \min_{\mathbf{A},\mathbf{B}} ||\mathbf{X} - \mathbf{A} \diamond \mathbf{B}||_F^2$$

Compare the estimated matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ with the original ones. What can you conclude? Explain the results.

Hint: Use the file "krf_matrix.mat" to validate your result.

Results

Simulation setup

- The algorithm that uses the SVD was applied to the initial factor matrices $\mathbf{A}_0$ and $\mathbf{B}_0$ initializated from a Normal distribution $\mathcal{N}(0,\,1)$;

Discussion

To compare the real data with the estimated factors, we may use two main results. The NMSE between the given data and obtained as output to LSKRF. As well the row/column factor scaling, i.e, apply the element-wise division between the given data and algorithm output for $\mathbf{A}$ vs $\hat{\mathbf{A}}$, and $\mathbf{B}$ vs $\hat{\mathbf{B}}$.

NMSE with LSKRF

```
X and X_hat: -629.76 dB
A and A_hat: 10.05 dB
B and B_hat: 11.60 dB
```

Scale factor for A and A_hat with LSKRF

```
A_hat(:,1)./A(:,1): [-1.1; -1.1; -1.1; -1.1; -1.1]
A_hat(:,2)./A(:,2): [0.66; 0.66; 0.66; 0.66; 0.66]
A_hat(:,3)./A(:,3): [-1.1; -1.1; -1.1; -1.1; -1.1]
A_hat(:,4)./A(:,4): [-0.69; -0.69; -0.69; -0.69; -0.69]
```

Scale factor for B and B_hat with LSKRF

```
B_hat(:,1)./B(:,1): [-0.89; -0.89; -0.89; -0.89]
B_hat(:,2)./B(:,2): [1.5; 1.5; 1.5; 1.5]
```

```
B_hat(:,3)./B(:,3): [-0.88; -0.88; -0.88; -0.88]
B_hat(:,4)./B(:,4): [-1.5; -1.5; -1.5; -1.5]
```

We can see that for all columns are composed by the same real value, for both $\mathbf{A}$ and $\mathbf{B}$. Hence, it presents the second evidence to confirm the proper algorithm estimation, since the columns differs only by a scale factor.

Problem 1 script.

Problem 2

Assuming 1000 Monte Carlo experiments, generate $\mathbf{X}_0 = \mathbf{A} \diamond \mathbf{B} \in \mathbb{C}^{IJ \times R}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{I \times R}$ and $\mathbf{B} \in \mathbb{C}^{J \times R}$, with $R = 4$, whose elements are drawn from a normal distribution.

Let $\mathbf{X} = \mathbf{X}_0 + \alpha V$ be a noisy version of $\mathbf{X}_0$, where V is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\mathrm{SNR_{dB}} = 10 \log_{10} \left(\frac{||\mathbf{X}_0||_F^2}{||\alpha V||_F^2} \right) \tag{1}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ obtained with the LSKRF algorithm for the configurations $(I, J) = (10, 10)$ and $(I, J) = (30, 10)$.

Let us define the normalized mean square error (NMSE) measure as follows

$$\mathrm{NMSE}(\mathbf{X}_0) = \frac{1}{1000} \sum_{i=1}^{1000} \frac{||\hat{\mathbf{X}}_0(i) - \mathbf{X}_0(i)||_F^2}{||\mathbf{X}_0(i)||_F^2} \tag{2}$$

where $\mathbf{X}_0(i)$ e $\hat{\mathbf{X}}_0(i)$ represent the original data matrix and the reconstructed one at the $i\mathrm{th}$ experiment, respectively. For each SNR value and configuration, plot the NMSE vs. SNR curve. Discuss the obtained results.

Note: For a given SNR (dB), the parameter α to be used in your experiment is determined from equation (1).

Results

Simulation setup

- 1000 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- SNR range $\{0, 5, 10, 15, 20, 25, 30\}$;
- Compute the LSKRF for each value, for $I = \{10, 30\}$.

Discussion

The results are consistent with the experiment perfomed, that for randomly generated $\mathbf{A}$ and $\mathbf{B}$, confirmed as shown in the previous part, the columns from given to estimated data differs only by a scale factor.

From the figure results, we may assess the SNR gap between the NMSE curves.

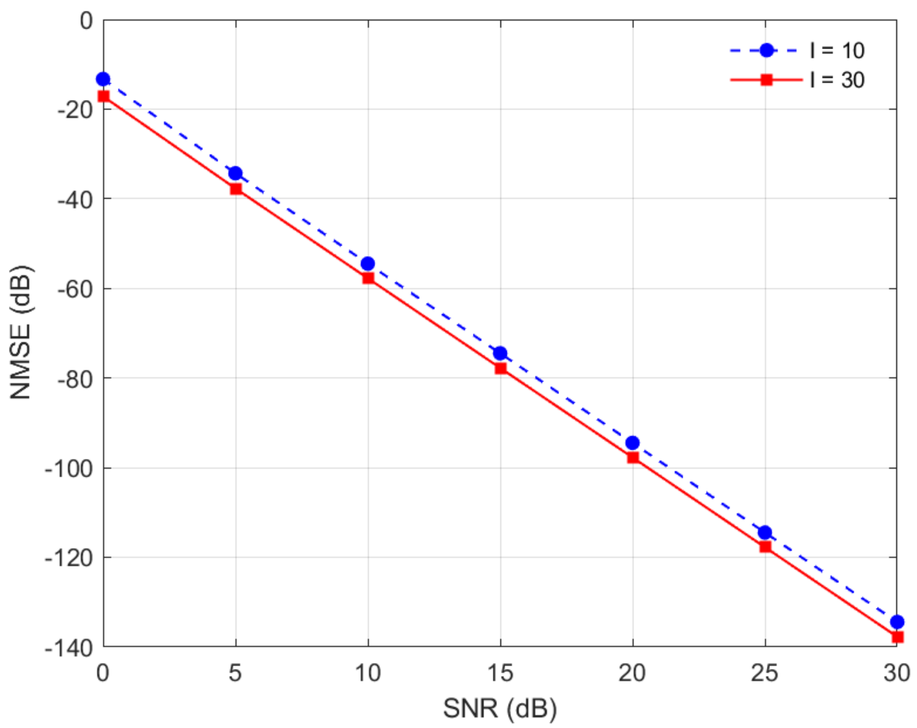
For each value of SNR, respectively:

```
Mean Diff d1 vs d2: 3.86 dB
Mean Diff d1 vs d2: 3.38 dB
Mean Diff d1 vs d2: 3.25 dB
Mean Diff d1 vs d2: 3.27 dB
Mean Diff d1 vs d2: 3.25 dB
Mean Diff d1 vs d2: 3.27 dB
Mean Diff d1 vs d2: 3.31 dB
```

The mean value for the difference (gap) between the curves:

```
Mean Diff: 3.37 dB
```

Problem 2 script and Figures.



Return to Table of Contents

Homework 4 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Least-Squares Kronnecker Product Factorization (LSKronF)

Problem 1

Generate $\mathbf{X} = \mathbf{A} \otimes \mathbf{B} \in \mathbb{C}^{24\times 6}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{4\times 2}$ and $\mathbf{B} \in \mathbb{C}^{6\times 3}$. Then, implement the Least-Squares Kronnecker Product Factorization (LSKronF) algorithm that estimate $\mathbf{A}$ and $\mathbf{B}$ by solving the following problem

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}) = \min_{\mathbf{A},\mathbf{B}} ||\mathbf{X} - \mathbf{A} \otimes \mathbf{B}||_F^2$$

Compare the estimated matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ with the original ones. What can you conclude? Explain the results.

Hint: Use the file "kronf_matrix.mat" to validate your result.

Results

Simulation setup

- The algorithm that uses the SVD was applied to the initial factor matrices $\mathbf{A}_0$ and $\mathbf{B}_0$ initializated from a Normal distribution $\mathcal{N}(0,\,1)$;

Discussion

To compare the real data with the estimated factors, we may use two main results in the Experiment and Validation sections. The NMSE between the given data and obtained as output to LSKronF. As well the row/column factor scaling, i.e, apply the element-wise division between the given data and algorithm output for $\mathbf{A}$ vs $\hat{\mathbf{A}}$, and $\mathbf{B}$ vs $\hat{\mathbf{B}}$.

- Experiment

NMSE with LSKronF

```
X and X_hat: -622.63 dB
A and A_hat: 9.94 dB
B and B_hat: 1.90 dB
```

Scale factor for A and A_hat with LSKronF

```
A_hat(:,1)./A(:,1): [0.19; 0.19; 0.19; 0.19]
A_hat(:,2)./A(:,2): [0.19; 0.19; 0.19; 0.19]
```

Scale factor for B and B_hat with LSKronF

```
B_hat(:,1)./B(:,1): [0.076; 0.076; 0.076; 0.076; 0.076; 0.076]
B_hat(:,2)./B(:,2): [0.076; 0.076; 0.076; 0.076; 0.076; 0.076]
B_hat(:,3)./B(:,3): [0.076; 0.076; 0.076; 0.076; 0.076; 0.076]
```

- Validation Data

NMSE with LSKronF

```
X and X_hat: -608.15 dB
A and A_hat: 10.49 dB
B and B_hat: 13.74 dB
```

Scale factor for A and A_hat with LSKronF

```
A_hat(:,1)./A(:,1): [-0.83; -0.83; -0.83; -0.83]
A_hat(:,2)./A(:,2): [-0.83; -0.83; -0.83; -0.83]
A_hat(:,2)./A(:,2): [-0.83; -0.83; -0.83; -0.83]
```

Scale factor for B and B_hat with LSKronF

```
B_hat(:,1)./B(:,1): [-1.2; -1.2; -1.2; -1.2]
B_hat(:,2)./B(:,2): [-1.2; -1.2; -1.2; -1.2]
```

The NMSE value, with an emphasis to $\text{NMSE}(\mathbf{X_0}, \hat{\mathbf{X}})$ value, with a very low SNR.

We can see that for all columns are composed by the same real value, for both $\mathbf{A}$ and $\mathbf{B}$. Hence, it presents the second evidence to confirm the proper algorithm estimation, since the columns differs only by a scale factor.

Problem 1 script.

Problem 2

Assuming 1000 Monte Carlo experiments, generate $\mathbf{X}_0 = \mathbf{A} \otimes \mathbf{B} \in \mathbb{C}^{IJ \times PQ}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{I \times P}$ and $\mathbf{B} \in \mathbb{C}^{J \times Q}$, whose elements are drawn from a normal distribution.

Let $\mathbf{X} = \mathbf{X}_0 + \alpha V$ be a noisy version of $\mathbf{X}_0$, where V is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \left(\frac{||\mathbf{X}_0||_F^2}{||\alpha V||_F^2} \right) \tag{3}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ obtained with the LSKronF algorithm for the configurations $(I, J) = (2, 4)$, $(P, Q) = (3, 5)$ and $(I, J) = (4, 8)$, $(P, Q) = (3, 5)$

Let us define the normalized mean square error (NMSE) measure as follows

$$\text{NMSE}(\mathbf{X}_0) = \frac{1}{1000} \sum_{i=1}^{1000} \frac{||\hat{\mathbf{X}}_0(i) - \mathbf{X}_0(i)||_F^2}{||\mathbf{X}_0(i)||_F^2} \tag{4}$$

where $\mathbf{X}_0(i)$ e $\hat{\mathbf{X}}_0(i)$ represent the original data matrix and the reconstructed one at the $i\text{th}$ experiment, respectively. For each SNR value and configuration, plot the NMSE vs. SNR curve. Discuss the obtained results.

Results

Simulation setup

- 1000 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- SNR range $\{0, 5, 10, 15, 20, 25, 30\}$;
- Compute the LSKronF for each value, for $(I, J) = (2, 4)$, $(P, Q) = (3, 5)$ and $(I, J) = (4, 8)$, $(P, Q) = (3, 5)$.

Discussion

The results are consistent with the experiment perfomed, that for randomly generated $\mathbf{A}$ and $\mathbf{B}$, what confirmed as shown in the previous part, the columns from given to estimated data differs only by a scale factor.

From the figure results, we may assess the SNR gap between the NMSE curves.

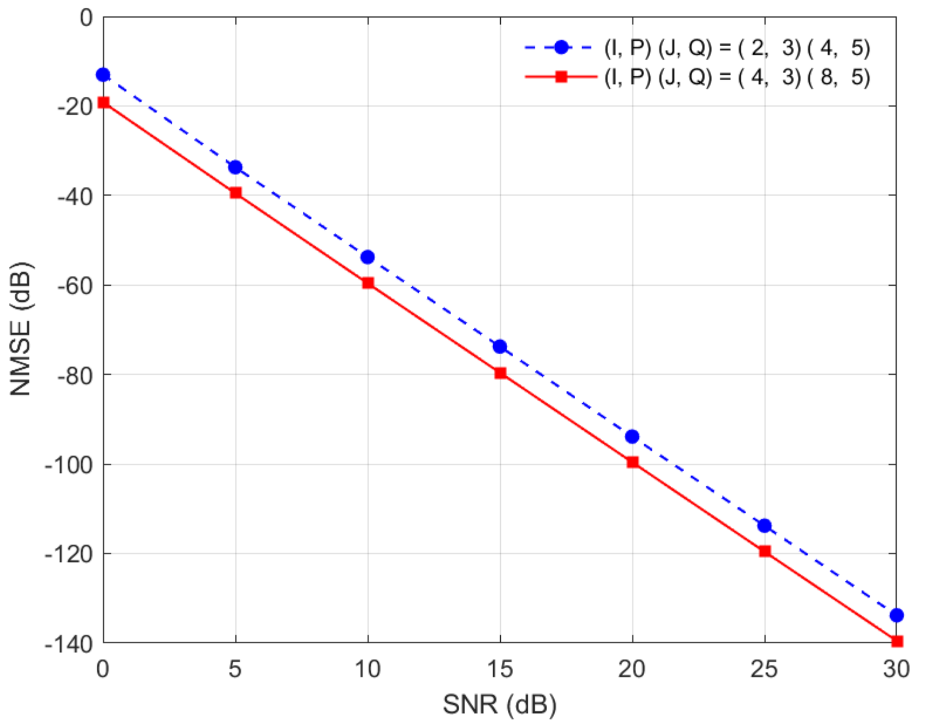
For each value of SNR, respectively:

```
Mean Diff d1 vs d2: 6.15 dB
Mean Diff d1 vs d2: 5.81 dB
Mean Diff d1 vs d2: 5.84 dB
Mean Diff d1 vs d2: 5.84 dB
Mean Diff d1 vs d2: 5.72 dB
Mean Diff d1 vs d2: 5.80 dB
Mean Diff d1 vs d2: 5.71 dB
```

The mean value for the difference (gap) between the curves:

```
Mean Diff: 5.84 dB
```

Problem 2 script and Figures.



Return to Table of Contents

Homework 5 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Kronecker Product Singular Value Decomposition (KPSVD)

Problem 1

Generate a block matrix according to the following structure

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_{1,1} & \dots & \mathbf{X}_{1,N} \\ \vdots & \ddots & \vdots \\ \mathbf{X}_{M,1} & \dots & \mathbf{X}_{M,N} \end{pmatrix}, \mathbf{X}_{i,j} \in \mathbb{C}^{P \times Q}, 1 \leq i \leq M,\, 1 \leq j \leq N\,,$$

Implement the KPSVD for the matrix $\mathbf{X}$ by computing σ_k, $\mathbf{U}_k$, and $\mathbf{V}_k$ such that

$$\mathbf{X} = \sum_{k=1}^{r_{KP}} \sigma_k \mathbf{U}_k \otimes \mathbf{V}_k$$

Results

Simulation setup

- The algorithm that uses the SVD was applied to estimate the original data;
- $M = N = P = Q = 3$;
- Randomly generate $\mathbf{X}_{i,j} = \mathrm{rand}(P, Q), 1 \leq i \leq M,\ 1 \leq j \leq N$
- Initializated from a Normal distribution $\mathcal{N}(0,\ 1)$.

Discussion

We use the experiment with the real rank to validate the algorithm, by observing the NMSE between the given data and obtained as output to KPSVD.

NMSE with KPSVD

```
Original Matrix vs KPSVD estimation (full rank): = -596.10 dB
```

The output present a very low NMSE value what, what may be used as evidence to confirm the proper algorithm estimation.

Problem 1 script.

Problem 2

In the above problem, set $M = N = P = Q = 3$ and randomly generate $\mathbf{X}_{i,j} = \mathrm{rand}(P, Q), 1 \leq i \leq M,\ 1 \leq j \leq N$. Then compute the KPSVD and the Kronecker-rank r_{KP} of $\mathbf{X}$ by using your KPSVD prototype function. Consider $r \leq r_{KP}$. Compute the nearest rank-r for the matrix $\mathbf{X}$.

Results

Simulation setup

- 1000 Monte Carlo Runs;
- The algorithm that uses the SVD was applied to estimate the original data;
- $M = N = P = Q = 3$;
- Randomly generate $\mathbf{X}_{i,j} = \mathrm{rand}(P, Q), 1 \leq i \leq M,\ 1 \leq j \leq N$
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- Compute the KPSVD to assess rank deficiency, for R in range $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, where the matrix presents its full rank for $R = 9$.

Discussion

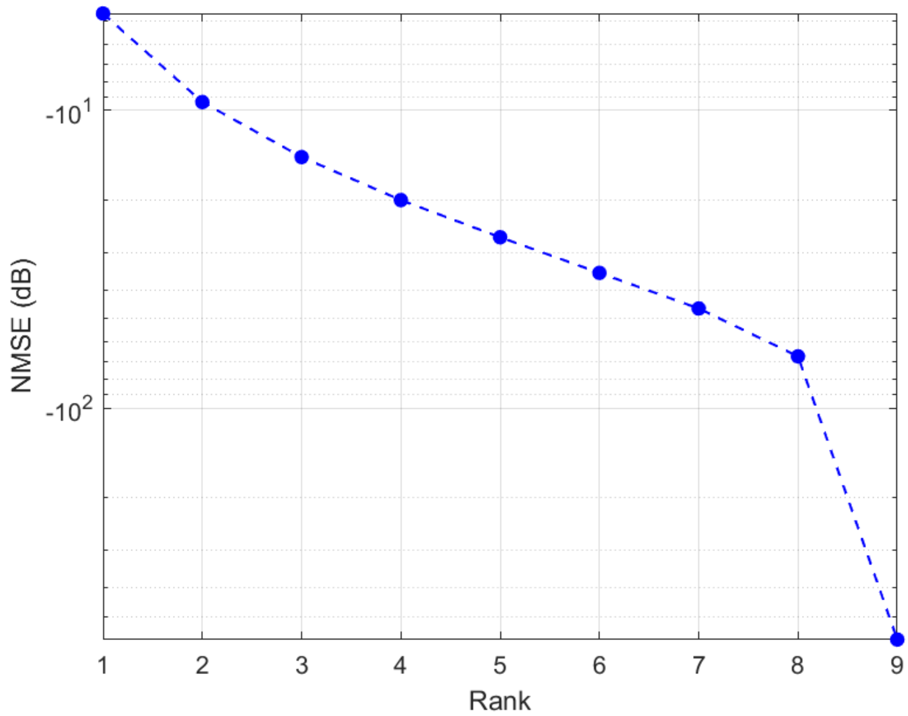
The results are consistent with the proposed scenario, since that for randomly generated $\mathbf{X}$, the algorithm succeds to obtain the lower NMSE (dB) with the known full-rank. Futhermore, we can see that as the rank decreases, the NMSE increases abruptly.

- Original Matrix vs KPSVD estimation

rank	NMSE (dB)
1	-4.72
2	-9.37
3	-14.33
4	-19.99
5	-26.64
6	-35.08
7	-46.19
8	-66.89
9	-597.43

As we can see results, the NMSE reduces as the rank increases, however it reaches the lowest point when the true rank is applied.

Problem 2 script and Figures.



Return to Table of Contents

Table of Contents

Unfolding, folding, and n-mode product

Problem 1

For a third-order tensor $\mathbf{X} \in \mathbb{C}^{I \times J \times K}$, using the concept of n-mode fibers, implement the function unfold according to the following prototype

$$[\mathcal{X}]_{(n)} = \text{unfold}(\mathcal{X}, n)$$

Hint: Use the file "unfolding_folding.mat" to validate your function.

Results

Simulation setup

- The algorithm was applied to reshape the original data into a N-mode tensor;
- N in range $\{1, 2, 3\}$.

Discussion

- Experiment proposed in the example 2.6 of the book Multi-way Analysis With Applications in the Chemical Sciences (Smilde, 2004).

Tensor X

```
X(:, :, 1)
1  2  3;
4  5  6;
7  8  9;
3  2  1;

X(:, :, 2)
5  6  7;
8  9  4;
5  3  2;
4  5  6;
```

Tensor X (mode-1)

```
X(4, 6)
1  2  3  5  6  7;
4  5  6  8  9  4;
```

```
7  8  9  5  3  2;
3  2  1  4  5  6;
```

Tensor X (mode-2)

```
X(3, 8)
1  4  7  3  5  8  5  4;
2  5  8  2  6  9  3  5;
3  6  9  1  7  4  2  6;
```

Tensor X (mode-3)

```
X(2, 12)
1  4  7  3  2  5  8  2  3  6  9  1;
5  8  5  4  6  9  3  5  7  4  2  6;
```

- Validation

Unfold difference

```
sum(X1 - unfold(X, 1)) = 0.00
sum(X2 - unfold(X, 2)) = 0.00
sum(X3 - unfold(X, 3)) = 0.00
```

We assess the difference between the given data and the algorithm output, and we can see that the residuals sum leads to zero.

Problem script.

Fold experiment output log: Unfold Txt File.

EXAMPLE 2.6

Matricizing operation

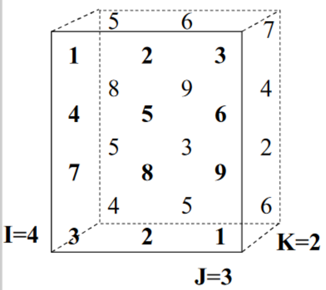


Figure 2.7. A three-way array $\underline{\mathbf{X}}$ ($4 \times 3 \times 2$); boldface characters are in the foremost frontal slice and normal characters are in the back slice.

An example of matricizing a three-way array $\underline{\mathbf{X}}$ is given in Figure 2.7. Three different matricized three-way arrays are as follows:

$$\mathbf{X}_{(I\times JK)} = \begin{bmatrix} 1 & 2 & 3 & 5 & 6 & 7 \\ 4 & 5 & 6 & 8 & 9 & 4 \\ 7 & 8 & 9 & 5 & 3 & 2 \\ 3 & 2 & 1 & 4 & 5 & 6 \end{bmatrix}$$

$$\mathbf{X}_{(J\times IK)} = \begin{bmatrix} 1 & 4 & 7 & 3 & 5 & 8 & 5 & 4 \\ 2 & 5 & 8 & 2 & 6 & 9 & 3 & 5 \\ 3 & 6 & 9 & 1 & 7 & 4 & 2 & 6 \end{bmatrix}$$

$$\mathbf{X}_{(K\times IJ)} = \begin{bmatrix} 1 & 4 & 7 & 3 & 2 & 5 & 8 & 2 & 3 & 6 & 9 & 1 \\ 5 & 8 & 5 & 4 & 6 & 9 & 3 & 5 & 7 & 4 & 2 & 6 \end{bmatrix}$$

Problem 2

Implement the function fold that converts the unfolding $[\mathcal{X}]_{(n)}$ obtained with $\text{unfold}(\mathcal{X}, n)$ back to the tensor $\mathcal{X} \in \mathbb{C}^{I\times J\times K}$ (i.e., a 3-d array in Matlab/Octave), according to the following prototype:

$$\mathcal{X} = \text{fold}([\mathcal{X}]_{(n)}, [IJK], n)$$

Hint: Use the file "unfolding_folding.mat" to validate your function.

Results

Simulation setup

- The algorithm was applied to build a tensor from a N-mode tensor;
- N in range $\{1, 2, 3\}$.

Discussion

- Experiment proposed in the example 2.6 of the book Multi-way Analysis With Applications in the Chemical Sciences (Smilde, 2004).

Tensor X (mode-1)

```
X(4, 6)
1  2  3  5  6  7;
4  5  6  8  9  4;
7  8  9  5  3  2;
3  2  1  4  5  6;
```

Tensor X (mode-2)

```
X(3, 8)
1  4  7  3  5  8  5  4;
2  5  8  2  6  9  3  5;
3  6  9  1  7  4  2  6;
```

Tensor X (mode-3)

```
X(2, 12)
1  4  7  3  2  5  8  2  3  6  9  1;
```

```
5   8   5   4   6   9   3   5   7   4   2   6;
```

Tensor X from (mode-1)

```
X(:, :, 1)
1   2   3;
4   5   6;
7   8   9;
3   2   1;

X(:, :, 2)
5   6   7;
8   9   4;
5   3   2;
4   5   6;
```

Tensor X from (mode-2)

```
X(:, :, 1)
1   2   3;
4   5   6;
7   8   9;
3   2   1;

X(:, :, 2)
5   6   7;
8   9   4;
5   3   2;
4   5   6;
```

Tensor X from (mode-3)

```
X(:, :, 1)
1   2   3;
4   5   6;
7   8   9;
3   2   1;

X(:, :, 2)
5   6   7;
8   9   4;
5   3   2;
4   5   6;
```

- Validation

Fold difference

```
sum(tenX - fold(X1)) = 0.00
sum(tenX - fold(X2)) = -0.00
sum(tenX - fold(X3)) = -0.00
```

We assess the difference between the given data and the algorithm output, and we can see that the residuals sum leads to zero.

Problem script.

Fold experiment output log: Fold Txt File.

Problem 3

For given matrices $\mathbf{A} \in \mathbb{C}^{P \times I}$, $\mathbf{B} \in \mathbb{C}^{Q \times J}$, $\mathbf{C} \in \mathbb{C}^{R \times K}$ and tensor $\mathcal{X} \in \mathbb{C}^{I \times J \times K}$, calculate the tensor $\mathcal{Y} \in \mathbb{C}^{P \times Q \times R}$ via the following multilinear transformation:

$$\mathcal{Y} = \mathcal{X} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$

Hint: Use the file “multilinear_product.mat” to validate your result.

Results

Simulation setup

- The algorithm was applied to compute the N-mode product between a given tensor and factor matrices.

Discussion

The results are consistent with the proposed scenario, since given data after the algorithm succeds to obtain a very low NMSE (dB) value.

```
NMSE between a given tensor and its version afected by the N-mode product:
-666.47 dB
```

Problem script

Return to Table of Contents

Homework 7 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

High Order Singular Value Decomposition (HOSVD)

Problem 1

For a third-order tensor $\mathcal{X} \in \mathbb{C}^{I \times J \times K}$ implement the truncated high-order singular value decomposition (HOSVD), using the following prototype function:

$$\left[\mathcal{S}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}\right] = \text{hosvd}(\mathcal{X}) \tag{5}$$

Hint: Use the file "hosvd_test.mat" to validate your results.

Results

Simulation setup

- The algorithm that uses the SVD was applied to the given initial tensor $\mathcal{X} \in \mathbb{C}^{I \times J \times K}$;
- $I, J, K = 3, 4, 5$.

Discussion

To compare the given data with the estimated factors, we may use two main experiment results: the orthogonality between the subtensors (slices) of $\mathcal{S}$ and the NMSE between the given data and obtained as output to HOSVD.

The orthogonality assessment consists in compute the function $f_{ort}(\mathcal{S})$, that acumulates the scalar product bewteen the slices, following the equation below.

$$f_{ort}(\mathcal{S}) = \sum_{k_M=1}^{K} \sum_{k_N=1}^{K} \text{vec}(\mathcal{S}_{:,:,k_M})^\top \text{vec}(\mathcal{S}_{:,:,k_N}) \quad for\ k_M \neq k_N$$

We obtain $f_{ort}(\mathcal{S}) = 0$, as expected for successful HOSVD.

```
-------------- The tensor slices are orthogonal if = zero --------------
f_ort = 0.00
```

The values obtained for NMSE present low SNR, with an emphasis to $\text{NMSE}(\mathcal{X}, \hat{\mathcal{X}})$ value, with a very low SNR.

```
--------------- NMSE between a given tensor X and estimation --------------
NMSE: -618.62 dB
--------------- NMSE between a given tensor core S and its estimation -----
NMSE: 6.51 dB
--------------- NMSE between the factor matrices U and their estimation -----
NMSE between U1 and its estimation: 8.52 dB
NMSE between U2 and its estimation: 6.02 dB
NMSE between U3 and its estimation: 4.12 dB
```

We can see that both results, Orthogonality and NMSE, support the proper algorithm estimation hypothesis.

Problem 1 script.

Problem 2

Consider the two third-order tensors $\mathcal{X} \in \mathbb{C}^{8\times4\times10}$ and $\mathcal{Y} \in \mathbb{C}^{5\times5\times5}$ provided in the data file "hosvd_denoising.mat". By using your HOSVD prototype function, find a low multilinear rank approximation for these tensors, defined as

$\tilde{\mathcal{X}} \in \mathbb{C}^{R1\times R2\times R3}$ and $\tilde{\mathcal{Y}} \in \mathbb{C}^{P1\times P2\times P3}$. Then, calculate the normalized mean square error (NMSE) between the original tensor and its approximation, i.e.,:

$$\text{NMSE}(\tilde{\mathcal{X}}) = \frac{||\tilde{\mathcal{X}} - \mathcal{X}||_F^2}{||\mathcal{X}||_F^2} \quad , \quad \text{NMSE}(\tilde{\mathcal{Y}}) = \frac{||\tilde{\mathcal{Y}} - \mathcal{Y}||_F^2}{||\mathcal{Y}||_F^2}$$

Hint: The multilinear ranks of X and Y can be found by analysing the profile of the 1-mode, 2-mode and 3-mode singular values of these tensors.

Results

Simulation setup

- The algorithm that uses the SVD was applied to the given initial tensor $\mathcal{X} \in \mathbb{C}^{R1\times R2\times R3}$ and $\mathcal{Y} \in \mathbb{C}^{P1\times P2\times P3}$;
- $R1, R2, R3 = 8, 4, 10$;
- $P1, P2, P3 = 5, 5, 5$.

Discussion

To compare the both random tensor estimation with given multilinear ranks, we may use NMSE results between the given data and obtained as output to HOSVD. We may assess also by comparing the multilinear rank obtained in the tensor core $\mathcal{S}_{\mathcal{X}}$ and $\mathcal{S}_{\mathcal{Y}}$ estimated with the given ones.

```
--------------- NMSE between a given tensor X and its estimation -----
NMSE: -600.49 dB
--------------- NMSE between a given tensor Y and its estimation -----
NMSE: -610.64 dB
```

The values obtained for NMSE present very low SNR, less than $-600\,\text{dB}$.

As defined in the proposed problem, the given ranks of $\mathcal{X}$ $\mathcal{Y}$ are $R1, R2, R3 = 8, 4, 10$, and $P1, P2, P3 = 5, 5, 5$, respectively.

```
Tensor X multilinear rank: [8 4 10]
Tensor Y multilinear rank: [5 5 5]
```

We can see that that the algorithm provide the expected result, with the given ranks equal to the estimated. In conclusion, both results, NMSE and ranks estimation using the tensor core, support the proper algorithm estimation hypothesis.

Problem 2 script.

Return to Table of Contents

Homework 8 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

High Order Orthogonal Iteration (HOOI)

Problem 1

For a third-order tensor $\mathcal{X} \in \mathbb{C}^{I \times J \times K}$ implement the High Order Orthogonal Iteration (HOOI) method, using the following prototype function:

$$\left[\mathcal{S}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}\right] = \text{hooi}(\mathcal{X}) \tag{6}$$

Compare the results with the HOSVD algorithm.

Hint: Use the file "hooi_test.mat" to validate your results.

Results

Simulation setup

- The algorithm that combines the SVD with a iterative estimation was applied to the given initial tensor $\mathcal{X} \in \mathbb{C}^{I \times J \times K}$;
- $I, J, K = 3, 4, 5$;
- Number of iterations: 100.

Discussion

To compare the given data with the estimated factors, we may use two main experiment results: the orthogonality between the subtensors (slices) of $\mathcal{S}$ and the NMSE between the given data and obtained as output to HOOI.

```
-------------- Number of Iterations --------------
it = 100
```

The orthogonality assessment consists in compute the function $f_{ort}(\mathcal{S})$, that acumulates the scalar product bewteen the slices, following the equation below.

$$f_{ort}(\mathcal{S}) = \sum_{k_M=1}^{K} \sum_{k_N=1}^{K} \text{vec}(\mathcal{S}_{:,:,k_M})^\top \text{vec}(\mathcal{S}_{:,:,k_N}) \quad for\ k_M \neq k_N$$

We obtain $f_{ort}(\mathcal{S}) = 0$, as expected for successful HOOI.

```
-------------- The tensor slices are orthogonal if = zero --------------
f_ort = 0.00
```

The values obtained for NMSE present low SNR, with an emphasis to $\text{NMSE}(\mathcal{X}, \hat{\mathcal{X}})$ value, with a very low SNR.

```
--------------- NMSE between a given tensor X and estimation --------------
NMSE: -629.27 dB
--------------- NMSE between a given tensor core S and its estimation -----
NMSE: 6.80 dB
--------------- NMSE between the factor matrices U and their estimation -----
NMSE between U1 and its estimation: 8.52 dB
NMSE between U2 and its estimation: -17.23 dB
NMSE between U3 and its estimation: 10.02 dB
```

We can see that both results, Orthogonality and NMSE, support the proper algorithm estimation hypothesis.

We assess also the HOOI vs HOSVD perfomance with the NMSE (dB). The HOOI algorithm outperforms HOSVD in four estimations, and has equal perfomance in one.

NMSE (dB)

$(\mathcal{X}, \hat{\mathcal{X}})$	$(\mathcal{S}, \hat{\mathcal{S}})$	$(\mathbf{U}^{(1)}, \hat{\mathbf{U}}^{(1)})$	$(\mathbf{U}^{(2)}, \hat{\mathbf{U}}^{(2)})$	$(\mathbf{U}^{(3)}, \hat{\mathbf{U}}^{(3)})$
+10.65	+0.29	0	+ 23.25	+5.9

Problem 1 script.

Problem 2

Consider the two third-order tensors $\mathcal{X} \in \mathbb{C}^{8\times4\times10}$ and $\mathcal{Y} \in \mathbb{C}^{5\times5\times5}$ provided in the data file "hosvd_denoising.mat". By using your HOOI prototype function, find a low multilinear rank approximation for these tensors, defined as $\tilde{\mathcal{X}} \in \mathbb{C}^{R1\times R2\times R3}$ and $\tilde{\mathcal{Y}} \in \mathbb{C}^{P1\times P2\times P3}$. Then, calculate the normalized mean square error (NMSE) between the original tensor and its approximation, i.e.,:

$$\text{NMSE}(\tilde{\mathcal{X}}) = \frac{||\tilde{\mathcal{X}} - \mathcal{X}||_F^2}{||\mathcal{X}||_F^2} \quad , \quad \text{NMSE}(\tilde{\mathcal{Y}}) = \frac{||\tilde{\mathcal{Y}} - \mathcal{Y}||_F^2}{||\mathcal{Y}||_F^2}$$

Hint: The multilinear ranks of X and Y can be found by analysing the profile of the 1-mode, 2-mode and 3-mode singular values of these tensors.

Simulation setup

- The algorithm that uses the SVD was applied to the given initial tensor $\mathcal{X} \in \mathbb{C}^{R1 \times R2 \times R3}$ and $\mathcal{Y} \in \mathbb{C}^{P1 \times P2 \times P3}$;
- $R1, R2, R3 = 8, 4, 10$;
- $P1, P2, P3 = 5, 5, 5$.

Discussion

To compare the both random tensor estimation with given multilinear ranks, we may use NMSE results between the given data and obtained as output to HOOI. We may assess also by comparing the multilinear rank obtained in the tensor core $\mathcal{S}_{\mathcal{X}}$ and $\mathcal{S}_{\mathcal{Y}}$ estimated with the given ones.

```
--------------- NMSE between a given tensor X and its estimation -----
NMSE: -603.32 dB
--------------- NMSE between a given tensor Y and its estimation -----
NMSE: -619.16 dB
```

The values obtained for NMSE present very low SNR, less than $-600\,\mathrm{dB}$.

As defined in the proposed problem, the given ranks of $\mathcal{X}$ $\mathcal{Y}$ are $R1, R2, R3 = 8, 4, 10$, and $P1, P2, P3 = 5, 5, 5$, respectively.

```
Tensor X multilinear rank: [8 4 10]
Tensor Y multilinear rank: [5 5 5]
```

We can see that that the algorithm provide the expected result, with the given ranks equal to the estimated.

We assess also the HOOI vs HOSVD perfomance with the NMSE (dB). The HOOI algorithm outperforms HOSVD in four estimations, and has equal perfomance in one.

NMSE (dB)

$(\mathcal{X}, \hat{\mathcal{X}})$	$(\mathcal{Y}, \hat{\mathcal{Y}})$
+2.83	+8.52

In conclusion, both results, NMSE and ranks estimation using the tensor core, support the proper algorithm estimation hypothesis.

Problem 2 script.

Return to Table of Contents

Table of Contents

Multidimensional Least-Squares Khatri-Rao Factorization (MLS-KRF)

Problem 1

Let $\mathbf{X} = \mathbf{A}^{(1)} \diamond \mathbf{A}^{(2)} \diamond \cdots \diamond \mathbf{A}^{(N)} \in \mathbb{C}^{I_1 I_2 \cdots I_N \times R}$ be a matrix generated from the Khatri-Rao product of N matrices $\mathbf{A}^{(n)} \in \mathbb{C}^{I_n \times R}$, with $n = 1, 2, \ldots, N$. Considering $N = 3$ and choosing your own values for R and $I_n, n = 1, 2, 3$, implement the MLS-KRF algorithm to find the estimates of $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$ by solving the following problem:

$$(\hat{\mathbf{A}}^{(1)}, \hat{\mathbf{A}}^{(2)}, \hat{\mathbf{A}}^{(3)}) = \min_{\hat{\mathbf{A}}^{(1)}, \hat{\mathbf{A}}^{(2)}, \hat{\mathbf{A}}^{(3)}} ||\mathbf{X} - \mathbf{A}^{(1)} \diamond \mathbf{A}^{(2)} \diamond \mathbf{A}^{(3)}||_F^2$$

Compare the estimated matrices $\hat{\mathbf{A}}^{(1)}$, $\hat{\mathbf{A}}^{(2)}$ and $\hat{\mathbf{A}}^{(3)}$ with the original ones. What can you conclude? Explain the results.

Hint: Use the file "krf_matrix_3D.mat" to validate your result.

Results

Simulation setup

- The algorithm that uses the Khatri-Rao Factorization was applied to the initial factor matrices, initializated from a Normal distribution $\mathcal{N}(0,\, 1)$;

Discussion

To compare the real data with the estimated factors, we may use two main results in the Experiment and Validation sections. The NMSE between the given data and obtained as output to MLSKRF. As well the row/column factor scaling, i.e, apply the element-wise division between the given data and algorithm output for $\mathcal{X}$ vs $\hat{\mathcal{X}}$, $\mathbf{A}$ vs $\hat{\mathbf{A}}$, $\mathbf{B}$ vs $\hat{\mathbf{B}}$ and $\mathbf{C}$ vs $\hat{\mathbf{C}}$.

NMSE with MLSKRF

```
X and X_hat: -3.34 dB
A and A_hat: -0.92 dB
B and B_hat: 1.39 dB
C and C_hat: 2.22 dB
```

Scale factor for X and X_hat with MLSKRF

```
X_hat./X(1:160, 1): 0.23
X_hat./X(1:160, 2): 0.36
X_hat./X(1:160, 3): 0.2
X_hat./X(1:160, 4): 0.018
```

Scale factor for A and A_hat with MLSKRF

```
A_hat./A(1:5, 1): 0.28
A_hat./A(1:5, 2): 0.3
A_hat./A(1:5, 3): 0.27
A_hat./A(1:5, 4): -0.13
```

Scale factor for B and B_hat with MLSKRF

```
B_hat./B(1:4, 1): 0.39
B_hat./B(1:4, 2): -0.47
B_hat./B(1:4, 3): 0.35
B_hat./B(1:4, 4): 0.15
```

Scale factor for B and B_hat with MLSKRF

```
C_hat./C(1:8, 1): -0.27
C_hat./C(1:8, 2): 0.31
C_hat./C(1:8, 3): -0.26
C_hat./C(1:8, 4): 0.11
```

The NMSE value, with an emphasis to $\mathrm{NMSE}(\mathcal{X}, \hat{\mathcal{X}})$ value.

We can see that for all columns are composed by the same real value, for all matrices factors. Hence, it presents the second evidence to confirm the proper algorithm estimation, since the columns differs only by a scale factor.

Problem 1 script.

Problem 2

Assuming 1000 Monte Carlo experiments, generate $\mathbf{X}_0 = \mathbf{A} \diamond \mathbf{B} \diamond \mathbf{C} \in \mathbb{C}^{I_1 I_2 I_3 \times R}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{I_1 \times R}$, $\mathbf{B} \in \mathbb{C}^{I_2 \times R}$ and $\mathbf{C} \in \mathbb{C}^{I_3 \times R}$, with $R = 4$, whose elements are drawn from a normal distribution. Let $\mathbf{X} = \mathbf{X}_0 + \alpha V$ be a noisy version of $\mathbf{X}_0$, where V is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\mathrm{SNR_{dB}} = 10 \log_{10} \left(\frac{||\mathbf{X}_0||_F^2}{||\alpha V||_F^2} \right) \tag{7}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$ and $\hat{\mathbf{C}}$ via the MLS-KRF algorithm, assuming $I_1 = 2$, $I_2 = 3$ and $I_3 = 4$.

Let us define the normalized mean square error (NMSE) measure as follows

$$\mathrm{NMSE}(\mathbf{X}_0) = \frac{1}{1000}\sum_{i=1}^{1000}\frac{||\hat{\mathbf{X}}_0(i) - \mathbf{X}_0(i)||_F^2}{||\mathbf{X}_0(i)||_F^2} \tag{8}$$

where $\mathbf{X}_0(i)$ e $\hat{\mathbf{X}}_0(i)$ represent the original data matrix and the reconstructed one at the $i\mathrm{th}$ experiment, respectively. For each SNR value and configuration, plot the NMSE vs. SNR curve. Discuss the obtained results.

Note: For a given SNR (dB), the parameter α to be used in your experiment is determined from equation (1).

Results

Simulation setup

- 1000 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- SNR range $\{0, 5, 10, 15, 20, 25, 30\}$;
- For $I_1 = 2$, $I_2 = 3$ and $I_3 = 4$;
- For $R = 4$.

Discussion

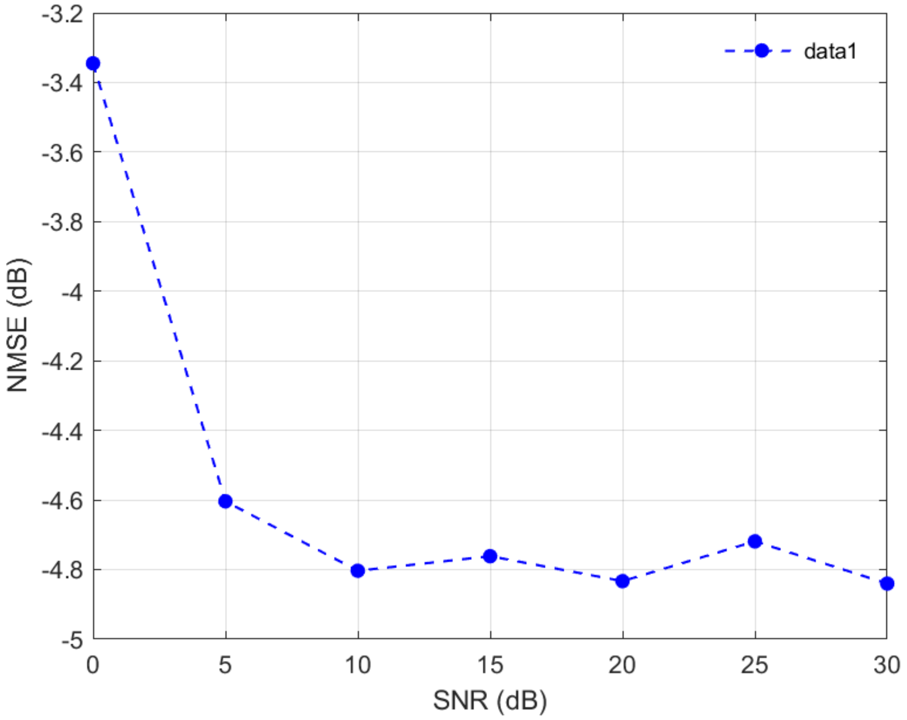
The results are consistent with the experiment perfomed, that for randomly generated $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, what confirmed as shown in the previous part, the columns from given to estimated data differs only by a scale factor.

From the figure results, we may assess the SNR gap between the NMSE curves.

For each value of SNR, respectively:

SNR	NMSE
0	-3.34
5	-4.60
10	-4.80
15	-4.76
20	-4.83
25	-4.72
30	-4.84

Problem 2 script and Figures.



Return to Table of Contents

Homework 10 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Multidimensional Least-Squares Kronecker Factorization (MLS-KronF)

Problem 1

Let $\mathbf{X} \in \mathbb{C}^{I_1 I_2 \dots I_N \times J_1 J_2 \dots J_N}$ be a matrix that we wish to approximate as $\mathbf{X} \approx \mathbf{A}^{(1)} \otimes \mathbf{A}^{(2)} \otimes \cdots \otimes \mathbf{A}^{(N)}$, that is, as Kronecker product of N matrices $\mathbf{A}^{(n)} \in \mathbb{C}^{I_n \times J_n}$ with $n = 1, 2, \dots, N$. For $N = 3$ and arbitrary I_n and J_n, implement the MLSKronF algorithm that estimates $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, $\mathbf{A}^{(3)}$ by solving the following problem:

$$(\hat{\mathbf{A}}^{(1)}, \hat{\mathbf{A}}^{(2)}, \hat{\mathbf{A}}^{(3)}) = \min_{\hat{\mathbf{A}}^{(1)}, \hat{\mathbf{A}}^{(2)}, \hat{\mathbf{A}}^{(3)}} ||\mathbf{X} - \mathbf{A}^{(1)} \otimes \mathbf{A}^{(2)} \otimes \mathbf{A}^{(3)}||_F^2$$

using either the truncated HOSVD or the HOOI initialized with the HOSVD (you should implement both versions).

Test the algorithms on a matrix that exactly follows the model. Compare the estimated matrices $\hat{\mathbf{A}}^{(1)}$, $\hat{\mathbf{A}}^{(2)}$ and $\hat{\mathbf{A}}^{(3)}$ with the original ones. What can you conclude? Explain the results.

Hint: Use the file "Practice_10_kronf_matrix_3D.mat" to validate your result.

Results

Simulation setup

- The algorithm that uses the Kronecker Factorization was applied to the initial factor matrices, initializated from a Normal distribution $\mathcal{N}(0,\ 1)$;
- HOSVD initization
- HOOI initization

Discussion

To compare the real data with the estimated factors, we may use the experimental results for NMSE between the given data and obtained as output to MLSKRF with HOSVD and HOOI.

NMSE with MLSKronF (HOSVD)

```
X and X_hat: -0.62 dB
A and A_hat: 4.07 dB
B and B_hat: 3.92 dB
C and C_hat: 1.91 dB
```

NMSE with MLSKronF (HOOI)

```
X and X_hat: -0.62 dB
A and A_hat: -9.49 dB
B and B_hat: -9.04 dB
C and C_hat: -3.97 dB
```

The NMSE value, with an emphasis to $\text{NMSE}(\mathcal{X}, \hat{\mathcal{X}})$ value, suport the hypothesis of a propor implementation of the algorithm.

We can see that MLSKRF initialized with HOOI outperforms HOSVD for all values, presenting smaller NMSE values also for the factor matrices.

Problem 1 script.

Problem 2

Assuming 1000 Monte Carlo experiments, generate $\mathbf{X}_0 = \mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C} \in \mathbb{C}^{I_1 I_2 I_3 \times J_1 J_2 J_3}$, for randomly chosen $\mathbf{A} \in \mathbb{C}^{I_1 \times J_1}$, $\mathbf{B} \in \mathbb{C}^{I_2 \times J_2}$ and $\mathbf{C} \in \mathbb{C}^{I_3 \times J_3}$, whose elements are drawn from a normal distribution. Let $\mathbf{X} = \mathbf{X}_0 + \alpha V$ be a noisy version of $\mathbf{X}_0$, where V is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\mathrm{SNR_{dB}} = 10\log_{10}\left(\frac{||\mathbf{X}_0||_F^2}{||\alpha V||_F^2}\right) \tag{9}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$ and $\hat{\mathbf{C}}$ via the MLS-KronF algorithm, assuming:

1. $(I_1, I_2, I_3) = (J_1, J_2, J_3) = (2, 2, 2)$;
2. $(I_1, I_2, I_3) = (J_1, J_2, J_3) = (5, 5, 5)$;
3. $(I_1, I_2, I_3) = (J_1, J_2, J_3) = (2, 3, 4)$;
4. $(I_1, I_2, I_3) = (2, 3, 4)$ and $(J_1, J_2, J_3) = (5, 6, 7)$.

Let us define the normalized mean square error (NMSE) measure as follows

$$\mathrm{NMSE}(\mathbf{X}_0) = \frac{1}{1000}\sum_{i=1}^{1000}\frac{||\hat{\mathbf{X}}_0(i) - \mathbf{X}_0(i)||_F^2}{||\mathbf{X}_0(i)||_F^2} \tag{10}$$

where $\mathbf{X}_0(i)$ e $\hat{\mathbf{X}}_0(i)$ represent the original data matrix and the reconstructed one at the $i\mathrm{th}$ experiment, respectively. For each SNR value and configuration, plot the NMSE vs. SNR curve. Discuss the obtained results.

Note: For a given SNR (dB), the parameter α to be used in your experiment is determined from equation (1).

Results

Simulation setup

- 1000 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\ 1)$;
- SNR range $\{0, 5, 10, 15, 20, 25, 30\}$;
- Assuming four scenarios as proposed.

Discussion

The results are consistent with the experiment perfomed, that provide small SNR values for the randomly generated factors $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$.

From the figure results, we may assess the SNR gap between the NMSE curves.

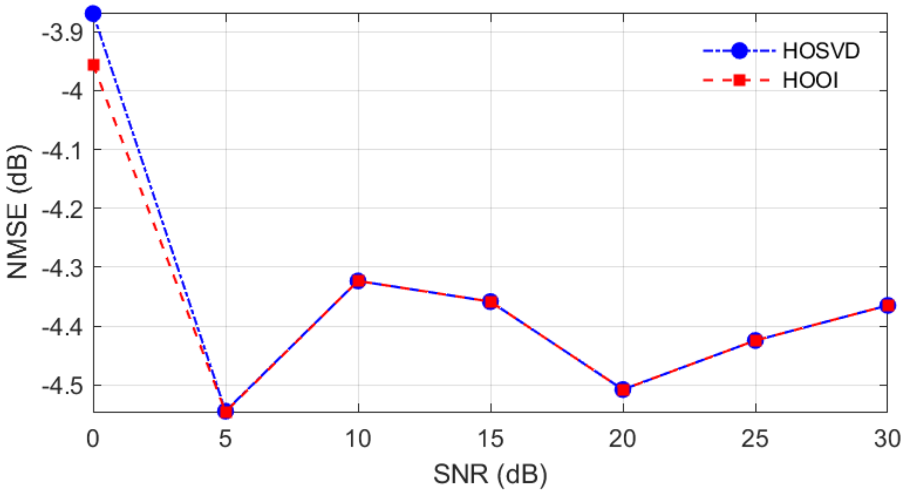
For each value of SNR, respectively:

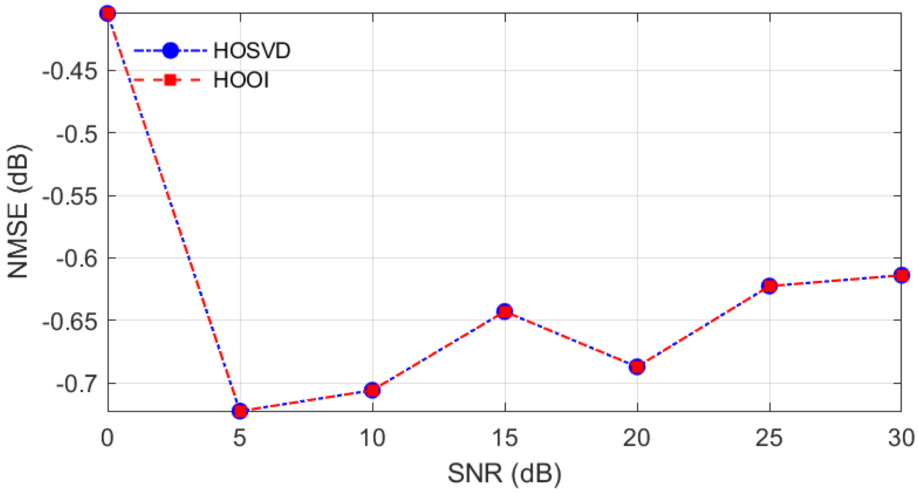
SNR	Scenario 1 NMSE	Scenario 2 NMSE	Scenario 3 NMSE	Scenario 4 NMSE
0	8.64e-02	0.00e+00	5.22e-04	1.11e-16
5	1.23e-03	-3.33e-16	4.07e-04	-1.11e-16
10	0.00e+00	-1.11e-16	-1.23e-04	0.00e+00
15	1.78e-15	0.00e+00	-2.22e-16	1.11e-16
20	-1.37e-05	0.00e+00	0.00e+00	1.11e-16
25	8.88e-16	6.66e-16	0.00e+00	0.00e+00
30	8.88e-16	0.00e+00	0.00e+00	-4.44e-16

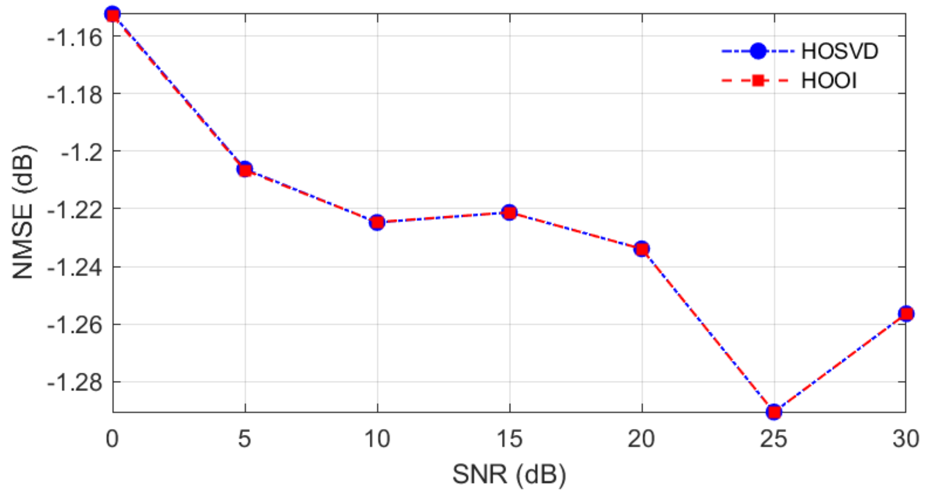
SNR	Scenario 1 NMSE	Scenario 2 NMSE	Scenario 3 NMSE	Scenario 4 NMSE
Mean	1.25e-02	3.17e-17	1.15e-04	-3.17e-17

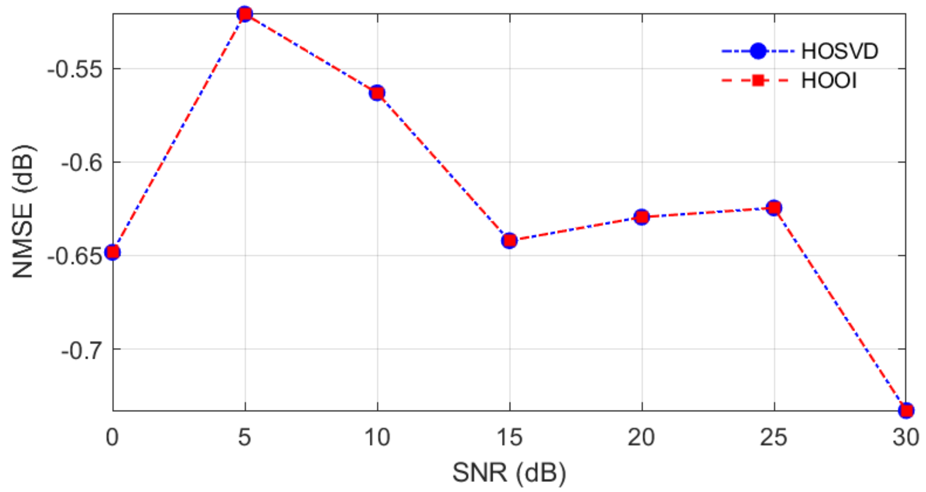
We can see that the HOOI initialization outpefomrs HOSVD with a small advantage Each experiment is implemented in: Problem 2 - Scenario 1, Problem 2 - Scenario 2, Problem 2 - Scenario 3, Problem 2 - Scenario 4.

Code that generates the figures: Problem 2 script,









Return to Table of Contents

Homework 11 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Alternating Least Squares (ALS) Algorithm

Problem 1

For the third-order tensor $\mathcal{X} \in \mathbb{C}^{I\times J\times K}$ provided in the file "cpd_tensor.mat", implement the plain-vanilla Alternating Least Squares (ALS) algorithm that estimates the factor matrices $\mathbf{A} \in \mathbb{C}^{I\times R}$, $\mathbf{B} \in \mathbb{C}^{J\times R}$ and $\mathbf{C} \in \mathbb{C}^{K\times R}$ by solving the following problem:

$$(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}) = \min_{\mathbf{A},\mathbf{B},\mathbf{C}} ||\mathcal{X} - \sum_{r=1}^{R} a_r \circ b_r \circ c_r||_F^2$$

where $\mathbf{A} = [a_1, \dots, a_R]$, $\mathbf{B} = [b_1, \dots, b_R]$, $\mathbf{C} = [c_1, \dots, c_R]$.

Considering a successful run, compare the estimated matrices $\mathbf{hat A}, \hat{\mathbf{B}}, \hat{\mathbf{C}}$ with the original ones (also provided in the same Matlab file). Explain the results.

Hint: An error measure at the ith iteration can be calculated from the following formula:

$$e_{(i)} = \min_{\mathbf{A},\mathbf{B},\mathbf{C}} ||[\mathcal{X}]_{(1)} - \hat{\mathbf{A}}_{(i)} \left((\hat{\mathbf{C}}) \diamond \hat{\mathbf{B}}_{(i)} \right)^\top ||_F$$

Results

Simulation setup

- The ALS algorithm that estimates the factor matrices of $\hat{\mathcal{X}}$ from a given tensor $\mathcal{X}$, minimizing the distance between them.
- $I, J, K = 8, 4, 5$;
- $R = 3$;
- Initialized from a Normal distribution $\mathcal{N}(0,\, 1)$.

Discussion

To compare the real data with the estimated factors, we may use the experimental results for NMSE between the given data and obtained as output to ALS.

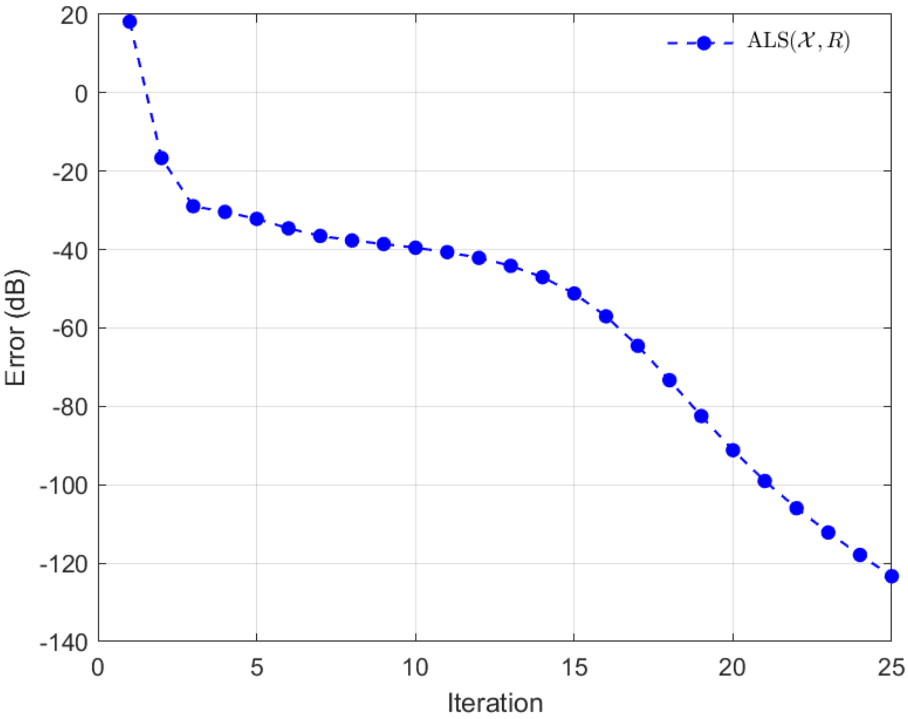
NMSE for ALS Validation

```
X and X_hat: -123.33 dB
A and A_hat: 27.96 dB
B and B_hat: 0.64 dB
C and C_hat: 25.72 dB
```

The results are consistent with the proposed scenario, since that for randomly generated $\mathbf{X}$, the algorithm succeds to obtain factors with small NMSE (dB) values.

Remark: the $\text{NMSE}(\mathbf{B}, \hat{\mathbf{B}})$ presents the smaller error since we choose to fix $\hat{\mathbf{A}}$ and $\hat{\mathbf{C}}$ to estimate $\hat{\mathbf{B}}$ at the first iteration. This choice was arbitrary and its behavior is presented for any factor matrix that is estimated first.

Problem 1 script.



Problem 2

In a Monte Carlo experiment with $M = 1000$ realizations, generate a tensor $\mathcal{X}_{(0)} = \text{CPD}(\mathbf{A}, \mathbf{B}, \mathbf{C})$, where $\mathbf{A} \in \mathbb{C}^{I \times R}$, $\mathbf{B} \in \mathbb{C}^{J \times R}$ and $\mathbf{C} \in \mathbb{C}^{K \times R}$ have unit norm columns with elements randomly drawn from a normal distribution, with $R = 3$.

Let $\mathcal{X} = \mathcal{X}_0 + \alpha \mathcal{V}$ be a noisy version of $\mathcal{X}_0$, where $\mathcal{V}$ is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \left(\frac{||\mathcal{X}_0||_F^2}{||\alpha \mathcal{V}||_F^2} \right) \tag{11}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$ and $\hat{\mathbf{C}}$ obtained with the ALS algorithm for $(I, J, K) = (10, 4, 2)$.

Let us define the normalized mean square error (NMSE) measure

$$\text{NMSE}(\mathbf{Q}) = \frac{1}{M} \sum_{m=1}^{M} \frac{||\hat{\mathbf{Q}}(m) - \mathbf{Q}(m)||_F^2}{||\mathbf{Q}(m)||_F^2} \tag{12}$$

where $\mathbf{Q}(m)$ and $\hat{\mathbf{Q}}$ denote the original data matrix and the reconstructed one at the m-th Monte Carlo experiment, respectively. For each SNR value, plot NMSE(A), NMSE(B) and NMSE(C) as a function of the SNR. Discuss the obtained results.

HintHint: Don't forget to take into account the inherent ambiguities of the CP decomposition.

Simulation setup

- 1000 Monte Carlo Runs;
- Each Monte Carlo iteration uses a new matrix initialization from a Normal distribution $\mathcal{N}(0,\,1)$;
- The ALS algorithm that estimates the factor matrices of $\hat{\mathcal{X}}$ from a given tensor $\mathcal{X}$, minimizing the distance between them.
- SNR range $\{0, 5, 10, 15, 20, 25, 30\}$;
- $I, J, K = 10, 4, 2$;
- $R = 3$.

Discussion

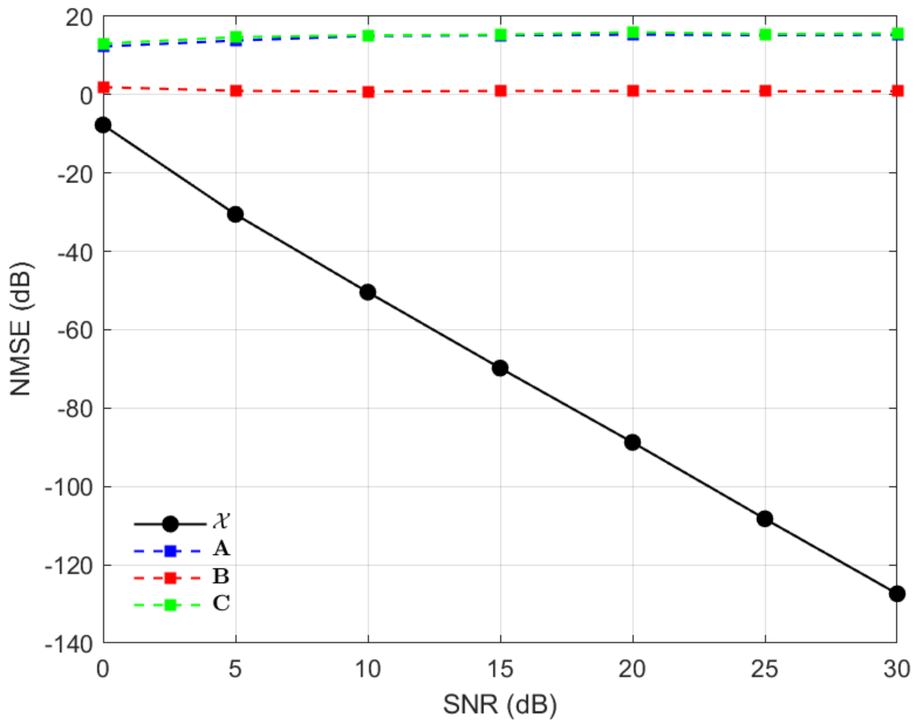
The Monte Carlo algorithm provides a repeated random sampling to obtain numerical results with the real data, a powerful tool to assess the trend using randomness.The experimental results for NMSE compare the given random data and obtained as output to ALS.

SNR [dB]	NMSE(X) [dB]	NMSE(A) [dB]	NMSE(B) [dB]	NMSE(C) [dB]
0	-7.71	12.32	1.94	13.06
5	-30.53	13.83	1.00	14.69
10	-50.40	15.03	0.82	15.18
15	-69.78	15.17	0.98	15.33
20	-88.74	15.34	0.95	15.97
25	-108.26	15.23	0.89	15.48
30	-127.41	15.31	0.87	15.63

The results are consistent with the second problem scenario, since that for randomly generated data, the algorithm succeds to obtain factors with small NMSE (dB) values. As well the SNR and NMSE for $\mathbf{X}$ are inversely proportional variables, i.e, as the SNR increases, the NMSE decreases.

The factor matrices remains practically stagnant, with very few variation as the SNR varies. The main notable difference is the remark indicated in the first problem, which indicates that the $\mathrm{NMSE}(\mathbf{B}, \hat{\mathbf{B}})$ presents the smaller error since we choose to fix $\hat{\mathbf{A}}$ and $\hat{\mathbf{C}}$ to estimate $\hat{\mathbf{B}}$ at the first iteration.

Problem 2 script and Figures.



Return to Table of Contents

Homework 12 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Tensor Kronecker Product Singular Value Decomposition (TKPSVD)

Problem 1

On a previous homework we have implemented the KPSVD (Kronecker Product Singular Value Decomposition) algorithm. Now, we will implement the generalization of that to tensors, namely, the TKPSVD (Tensor Kronecker Product Singular Value Decomposition) algorithm. Consider the N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Then, a TKPSVD of $\mathcal{X}$ can be written as

$$\mathcal{X} = \sum_{j=1}^{R} \sigma_j \mathcal{A}^{(d)} \otimes \mathcal{A}^{(d-1)} \otimes \cdots \otimes \mathcal{A}^{(1)}$$

where the tensors $\mathcal{A}_j^{(i)} \in \mathbb{R}^{I_1^{(i)} \times I_2^{(i)} \times \cdots \times I_N^{(i)}}$ satisfy

$$||\mathcal{A}_j^{(i)}||_F = 1\,, \prod_{d}^{i=1} I_k^{(i)} = I_K\,, 1 \leq k \leq N$$

Set $R = 1$ and generate $\mathcal{X} = \sigma \mathcal{A}^{(3)} \otimes \mathcal{A}^{(2)} \otimes \mathcal{A}^{(1)}$, for randomly chosen σ, $\mathcal{A}^{(1)} \in \mathbb{R}^{10\times4\times2}$, $\mathcal{A}^{(2)} \in \mathbb{R}^{5\times2\times2}$, and $\mathcal{A}^{(3)} \in \mathbb{R}^{2\times2\times2}$.

To avoid scaling ambiguity issues, normalize the model so that $||\mathcal{A}^{(i)}||_F = 1$ for $i = 1, 2, 3$. (In other words, the norm will be absorbed by σ). Then, implement the TKPSVD algorithm that estimate $\mathcal{A}^{(1)}$, $\mathcal{A}^{(2)}$, $\mathcal{A}^{(3)}$ by solving the following problem

$$(\hat{\mathbf{A}}^{(1)}, \hat{\mathbf{A}}^{(2)}, \hat{\mathbf{A}}^{(3)}) = \min_{\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \mathcal{A}^{(3)}} ||\mathcal{X} - \sigma \mathcal{A}^{(3)} \otimes \mathcal{A}^{(2)} \otimes \mathcal{A}^{(1)}||_F^2$$

Compare the estimated tensors $\hat{\mathbf{A}}^{(1)}$, $\hat{\mathbf{A}}^{(2)}$, $\hat{\mathbf{A}}^{(3)}$ with the original (normalized) ones. What can you conclude? Explain the results.

Results

Problem 2

In a Monte Carlo experiment with $M = 1000$ realizations, generate

$$\mathcal{X}_{(0)} = \sum_{r=1}^{R} \sigma_r \mathcal{A}^{(3)} \otimes \mathcal{A}^{(2)} \otimes \mathcal{A}^{(1)}$$

for $R = 2$ and randomly chosen $\mathcal{A}^{(1)} \in \mathbb{R}^{10\times4\times2}$, $\mathcal{A}^{(2)} \in \mathbb{R}^{5\times2\times2}$, and $\mathcal{A}^{(3)} \in \mathbb{R}^{2\times2\times2}$ whose elements are drawn from a standard normal distribution. As in the previous case, the scalars σ_r are meant to absorb the scaling of each term, while the random tensors will have unit Frobenius norm (you should first draw these tensors and then normalize them).

Let $\mathcal{X} = \mathcal{X}_0 + \alpha\mathcal{V}$ be a noisy version of $\mathcal{X}_0$, where $\mathcal{V}$ is the additive noise term, whose elements are drawn from a normal distribution. The parameter α controls the power (variance) of the noise term, and is defined as a function of the signal to noise ratio (SNR), in dB, as follows

$$\mathrm{SNR}_{\mathrm{dB}} = 10\log_{10}\left(\frac{||\mathcal{X}_0||_F^2}{||\alpha\mathcal{V}||_F^2}\right) \tag{13}$$

Assuming the SNR range $\{0, 5, 10, 15, 20, 25, 30\}$ dB, find the estimates $\hat{\mathbf{A}}^{(1)}$, $\hat{\mathbf{A}}^{(2)}$ and $\hat{\mathbf{A}}^{(3)}$ obtained with the ALS algorithm for $(I, J, K) = (10, 4, 2)$.

Let us define the normalized mean square error (NMSE) measure

$$\mathrm{NMSE}(\mathbf{Q}) = \frac{1}{M}\sum_{m=1}^{M}\frac{||\hat{\mathbf{Q}}(m) - \mathbf{Q}(m)||_F^2}{||\mathbf{Q}(m)||_F^2} \tag{14}$$

where $\mathbf{Q}(m)$ and $\hat{\mathbf{Q}}$ denote the original data matrix and the reconstructed one at the m-th Monte Carlo experiment, respectively. For each SNR value, plot NMSE $(\mathbf{A}^{(j)})$ as a function of the SNR for $j = 1, 2, 3$, where the matrix $\mathbf{A}^{(j)}$ is defined as

$$\mathbf{A}^{(j)} = \left[\mathrm{vec}(\mathcal{A}_1^{(j)})\mathrm{vec}(\mathcal{A}_2^{(j)})\right] \tag{15}$$

Discuss the obtained results.

HintHint: Don't forget to take into account the inherent ambiguities of the decomposition when computing the NMSE.

Results

Return to Table of Contents

Homework 13 [TI8419 - Multilinear Algebra]

Lucas Abdalah

Professors: André Lima e Henrique Goulart

Table of Contents

Tensor Train Singular Value Decomposition (TT-SVD)

Problem 1

Results

Problem 2

Results

Return to Table of Contents