Numerical Linear Algebra Homework Project 2: Least Squares, Orthogonalization, and the SVD

Catalano Giuseppe, Cerrato Nunzia

Problem 1

(1) Suppose we are given m pairs of data points, $(x_1, y_1), \ldots, (x_m, y_m)$. We want to find a linear combination of prescribed functions $\phi_1, \ldots, \phi_n$ whose values at the points $x_i \in [a, b]$, $1 \leq i \leq m$, approximate the values $y_1, \ldots, y_m$ as well as possible. More precisely, the problem is to find a function of the form $f(x) = \alpha_1\phi_1(x) + \cdots + \alpha_n\phi_n(x)$ such that

$$\sum_{i=1}^{m} [y_i - f(x_i)]^2 \leq \sum_{i=1}^{m} [y_i - g(x_i)]^2 \quad \forall g \in \text{Span}(\phi_1, \ldots, \phi_n), \tag{1}$$

where, usually, $m > n$. It is possible to rephrase the problem as:

$$f = \arg \min_{f \in \text{Span}(\phi_1, \ldots, \phi_n)} \sum_{i=1}^{m} [y_i - f(x_i)]^2 . \tag{2}$$

Now we can define a column vector $\mathbf{z} \in \mathbb{R}^n$ such that:

$$[\mathbf{z}]_i = \alpha_i \tag{3}$$

and a matrix A such that:

$$[A\mathbf{z}]_i = f(x_i) = \alpha_1\phi_1(x_i) + \cdots + \alpha_n\phi_n(x_i). \tag{4}$$

In this way, the element of the $i-th$ row and $j-th$ column of the matrix A is:

$$[A]_{ij} = a_{ij} = \phi_j(x_i). \tag{5}$$

Finally, defining a column vector $\mathbf{b} \in \mathbb{R}^n$ such that:

$$[\mathbf{b}]_i = y_i \tag{6}$$

we can rewrite the (2) as follows:

$$\tilde{\mathbf{z}} = \arg \min_{\mathbf{z} \in \mathbb{R}^n} \|\mathbf{b} - A\mathbf{z}\|_2^2 = \arg \min_{\mathbf{z} \in \mathbb{R}^n} \|\mathbf{b} - A\mathbf{z}\|_2, \tag{7}$$

where the function f can be built from $\tilde{\mathbf{z}}$.
(2) Now we suppose to take $\phi_k = x^{k-1}$, $1 \leq k \leq n$. Under this assumption, the matrix A takes the form:

$$A = \begin{bmatrix} x_1^0 & \cdots & x_1^{n-1} \\ \vdots & \ddots & \vdots \\ x_m^0 & \cdots & x_m^{n-1} \end{bmatrix} . \tag{8}$$

We want to prove that, assuming that $x_i \neq x_j$ for $i \neq j$, A has full rank: $\text{rank}(A) = n$.
Proof: Proving that $\text{rank}(A) = n$ is equivalent to prove that $\dim(\ker(A)) = 0$, that means

that $\nexists \mathbf{v} \in \mathbb{R}^n$ s.t. $\mathbf{v} \in \ker(A)$. We want to prove this statement by contraddiction, therefore, we look for a vector $\mathbf{v} \in \mathbb{R}^n$, with $\mathbf{v} \neq \underline{0}$, such that $A\mathbf{v} = \underline{0}$, that means:

$$\begin{cases} v_1 x_1^0 + \cdots + v_n x_1^{n-1} = 0 \\ \vdots \\ v_1 x_m^0 + \cdots + v_n x_m^{n-1} = 0 \end{cases}. \tag{9}$$

Defining the polynomial

$$p_{\mathbf{v}}^{(n-1)}(x) = \sum_{i=1}^{n} v_i x^{i-1} \tag{10}$$

we can observe that, for any choice of $\mathbf{v} \neq \underline{0}$, $p_{\mathbf{v}}^{(n-1)}(x)$ admits at most $n-1$ different roots, therefore $\nexists \mathbf{v} \neq \underline{0}$ such that $A\mathbf{v} = \underline{0}$. This concludes the proof. $\square$

(3) Consider the problem of finding the best fit with a quadratic function $f(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2$ for the following data:

x_i	8	10	12	16	20	30	40	60	100
y_i	0.88	1.22	1.64	2.72	3.96	7.66	11.96	21.56	43.16

In the following we report the code that solves the normal equations:

$$A^T A\mathbf{v} = A^T \mathbf{b} \tag{11}$$

using the Cholesky factorization algorithm and then compares the result with the one found using the QR factorization of the matrix A.
*** INSERIRE CODICE ***
Using the previous functions we have computed the solution to the minimization problem, obtaining the following results:

$$\begin{array}{cc} \text{Cholesky:} & x = [-1.91914925269909,\ 0.278213536291725,\ 0.001739400875055] \\ \text{QR factorization:} & x = [-1.91914925269904,\ 0.278213536291722,\ 0.001739400875055] \end{array}.$$

From these results we can observe that, for this problem, both the algorithms perform in a similar way. In fact, the results differ at most in the 15th digit. In figure 1 we show the input data and the solutions to the least square problem.

(4) The following code computes the residual $\mathbf{r} = \mathbf{d} - C\hat{\mathbf{x}}$ where $\hat{\mathbf{x}} = [-1.919,\ 0.2782,\ 0.001739]$ is the approximate solution of the least squares problem.
INSERIRE CODICE

The results that we obtain are:

$$\begin{array}{cc} \text{Residual:} & \mathbf{r} = [0.009503999999993,\ 0.716895999999906,\ 62.090847999905236] \\ \text{Norm 2 of the residual:} & \|\mathbf{r}\|_2 = 62.09498720144942 \end{array}.$$

The value of the residual may seem strange, in fact, if we compute the relative error we find:

$$\frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2} = 7.728184292875672\text{e-}05. \tag{12}$$

However, we can observe that, from the relation

$$\frac{1}{k_2(C)} \frac{\|\mathbf{r}\|_2}{\|\mathbf{d}\|_2} \leq \frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2} \leq k_2(C) \frac{\|\mathbf{r}\|_2}{\|\mathbf{d}\|_2} \tag{13}$$

we can obtain the following relation for $\mathbf{r}$:

$$\frac{\|\mathbf{d}\|_2}{k_2(C)} \frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2} \leq \|\mathbf{r}\|_2 \leq k_2(C) \|\mathbf{d}\|_2 \frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2}. \tag{14}$$

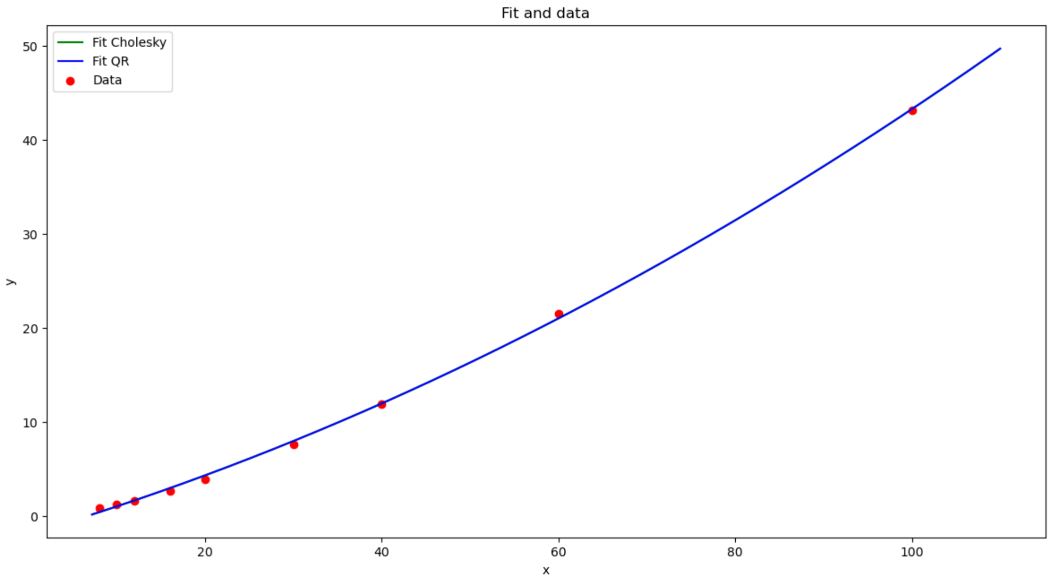


Figure 1

With the following python code we have computed the bounds to the residual. *** INSERIRE CODICE***

From the output of the previous code we know that:

$$5 \times 10^{-7} \lessapprox \|\mathbf{r}\|_2 \lessapprox 3 \times 10^9 \tag{15}$$

this big range is due to the value of $k_2(C) \simeq 8 \times 10^7$ that suggests us that we should not use the residual to measure the accuracy of the solution when the problem is ill-conditioned as in this case.

Problem 2

(1) Let $A \in \mathbb{R}^{m\times n}$, with $\text{rank}(A) = n$, let $\text{A} = \text{QR}$ be the (full) QR factorization of A, with $Q \in \mathbb{R}^{m\times m}$ orthogonal and $R \in \mathbb{R}^{m\times n}$ upper trapezoidal. Also, let $A = Q_1R_1$ be the reduced QR factorization of A with $Q_1 \in \mathbb{R}^{m\times n}$ having orthonormal columns and $R_1 \in \mathbb{R}^{m\times m}$ upper triangular. Show that R_1 is nonsingular, and that the columns $\mathbf{q}_1..., \mathbf{q}_n$ of Q_1 form an orthonormal basis for $\text{Ran}(A)$, the column space of A. Also, find an orthonormal basis for $\text{Null}(A^T)$, the null space of A^T .
We start showing that R_1 is nonsingular. Since A has full rank, we know that:

$$A\mathbf{x} = \mathbf{0} \Leftrightarrow \mathbf{x} = \mathbf{0} \tag{16}$$

and multiplying both sides for Q_1^T knowing that $Q_1^TQ_1 = \mathcal{I}_n$ we obtain

$$R_1\mathbf{x} = \mathbf{0} \Leftrightarrow \mathbf{x} = \mathbf{0}, \tag{17}$$

that concludes the proof.
Now we want to show that the columns $\mathbf{q}_1..., \mathbf{q}_n$ of Q_1 form an orthonormal basis for $\text{Ran}(A)$. We start observing that

$$\forall \mathbf{y} \in \text{Ran}(A) \exists \mathbf{x} \in \mathbb{R}^n : \mathbf{y} = A\mathbf{x} = Q_1R_1\mathbf{x} \tag{18}$$

and we know this from the definition of range of a matrix. In a similar way, knowing that R_1 is nonsingular and therefore is a bijective map from $\mathbb{R}^n$ to $\mathbb{R}^n$, we can put $\mathbf{x}' = R_i\mathbf{x}$ and say that:

$$\forall \mathbf{y} \in \text{Ran}(A) \exists \mathbf{x} \in \mathbb{R}^n : \mathbf{y} = A\mathbf{x} = Q_1\mathbf{x}', \tag{19}$$

that means that $\text{Ran}(A) = \text{Ran}(Q_1) = \text{Span}\{\mathbf{q}_1, \dots, \mathbf{q}_n\}$.
Now we want to find an orthonormal basis for $\text{Null}(A^T)$. Knowing that $\text{Null}(A^T) = (\text{Ran}(A))^\perp$ we can, immediately, find the solution. In fact:

$$(\text{Ran}(A))^\perp = \text{Span}\{\mathbf{q}_1, \dots, \mathbf{q}_n\}^\perp = \text{Span}\{\mathbf{q}_{n+1} \dots, \mathbf{q}_m\}. \tag{20}$$

This concludes the proof. $\square$

(2) Given tha full rank matrix

$$A = \begin{bmatrix} 1.07 & 1.10 \\ 1.07 & 1.11 \\ 1.07 & 1.15 \end{bmatrix}, \tag{21}$$

compute A^TA in $\beta = 10$, $t = 3$ digit arithmetic and verify if A^TA is positive definite.
We performed the prescribed calculation using the rounding function

$$\text{fl}(x) = \arg\min_{y \in \mathbb{F}} |y - x| \tag{22}$$

where $\mathbb{F} = \{\pm(0.d_1d_2d_3) \times 10^p : d_i \in 0, \dots, 9, \ -2046 \le p \le 2046\}$ and, when there is ambiguity, we approximate always away from 0. Moreover, the rounding function has been applied at each step of the calculation, meaning that, if we consider $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$

$$\mathbf{v}^T\mathbf{u} = \text{fl}\left(\sum_{i=1}^n \text{fl}(v_iu_i)\right). \tag{23}$$

The result of the calculation is the following:

$$A^TA = \begin{bmatrix} 3.42 & 3.6 \\ 3.6 & 3.76 \end{bmatrix}. \tag{24}$$

The last matrix is indefinite, in fact $\text{Det}(A^TA) = 3.42 \times 3.76 - 3.6^2 = 12.8592 - 12.96 < 0$.

(3) Compute the QR factorization of the matrix

$$A = \begin{bmatrix} 1.07 & 1.10 \\ 1.07 & 1.11 \\ 1.07 & 1.15 \end{bmatrix} = [\mathbf{a}_1 \ \mathbf{a}_2] \tag{25}$$

in 3 digit arithmetic, i.e. with $\beta = 10$, $t = 3$. The $\text{fl}(x)$ function has been applied at each step, as in the previous exercise.
We want to compute H_1A:

$$H_1A = \begin{bmatrix} r_{11} & \mathbf{r}_1^T \\ \mathbf{0} & A_1 \end{bmatrix}. \tag{26}$$

$$r_{11} = -\text{sgn}(a_{11})\|\mathbf{a}_1\|_2 = \|\mathbf{a}_1\|_2 = \sqrt{1.14 \times 3} = \sqrt{3.42} = -1.85 \tag{27}$$

$$\hat{\mathbf{u}}_1 = \mathbf{a}_1 + r_{11}\mathbf{e}_1 = \begin{pmatrix} 1.07 \\ 1.07 \\ 1.07 \end{pmatrix} + 1.85 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2.92 \\ 1.07 \\ 1.07 \end{pmatrix} \tag{28}$$

$$\|\hat{\mathbf{u}}\|_2 = \sqrt{8.53 + 1.14 + 1.14} = \sqrt{10.81} = 3.29 \tag{29}$$

$$\mathbf{u}_1 = \hat{\mathbf{u}}_1/\|\hat{\mathbf{u}}\|_2 = \begin{pmatrix} 0.888 \\ 0.325 \\ 0.325 \end{pmatrix} \tag{30}$$

$$\begin{aligned} H_1\mathbf{a}_2 &= \mathbf{a}_2 - 2(\mathbf{u}_1^T\mathbf{a}_2)\mathbf{u}_1 \\ &= \begin{pmatrix} 1.10 \\ 1.11 \\ 1.15 \end{pmatrix} - 2(0.977 + 0374 + 0.374) \begin{pmatrix} 0.888 \\ 0.325 \\ 0.325 \end{pmatrix}. \\ &= \begin{pmatrix} 1.10 \\ 1.11 \\ 1.15 \end{pmatrix} - 3.42 \begin{pmatrix} 0.888 \\ 0.325 \\ 0.325 \end{pmatrix} = \begin{pmatrix} -1.94 \\ 0 \\ 0.04 \end{pmatrix} \end{aligned} \tag{31}$$

Therefore we have:

$$H_1 A = \begin{pmatrix} -1.85 & -1.94 \\ 0 & 0 \\ 0 & 0.04 \end{pmatrix}. \tag{32}$$

Now we iterate the same procedure on A_1, with

$$A_1 = [\mathbf{a}_2^{(1)}] = \begin{pmatrix} 0 \\ 0.04 \end{pmatrix}. \tag{33}$$

$$\hat{\mathbf{u}}_2^{(1)} = \mathbf{a}_2^{(1)} + \mathrm{sgn}(a_{12}^{(1)})\|\mathbf{a}_2^{(1)}\|_2\mathbf{e}_2^{(1)} = \begin{pmatrix} 0 \\ 0.04 \end{pmatrix} + \mathrm{sgn}(0)0.04 \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.04 \\ 0.04 \end{pmatrix} \tag{34}$$

$$\mathbf{u}_2^{(1)} = \hat{\mathbf{u}}_2^{(1)}/\|\hat{\mathbf{u}}_2^{(1)}\|_2 = \begin{pmatrix} 0.707 \\ 0.707 \end{pmatrix}. \tag{35}$$

The complete vector $\mathbf{u}_2$ will be:

$$\mathbf{u}_2 = \begin{pmatrix} 0 \\ 0.707 \\ 0.707 \end{pmatrix} \tag{36}$$

and the element r_{22} willl be

$$r_{22} = -\mathrm{sgn}(a_{12}^{(1)})\|\mathbf{a}_2^{(1)}\|_2 = -0.04 \tag{37}$$

so that, finally, we have

$$H_2 H_1 A = R = \begin{pmatrix} -1.85 & -1.94 \\ 0 & -0.04 \\ 0 & 0 \end{pmatrix}. \tag{38}$$

We can observe that the matrix R is full rank, therefore it is possible to use the QR factorization to solve the least square problem. It is important to stress that, as $A^T A$ is indefinite in 3 digit precision, using Cholesky is not possible.

Now we want to compute $Q = [\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3]$ and, in order to do it, we use the following relations:

$$\begin{aligned} \mathbf{q}_1 &= H_1 H_2 \mathbf{e}_1 = \mathbf{e}_1 - 2(\mathbf{u}_1^T\mathbf{e}_1)\mathbf{u}_1 \\ \mathbf{q}_2 &= H_1 H_2 \mathbf{e}_2 = H_1(\mathbf{e}_2 - 2(\mathbf{u}_2^T\mathbf{e}_2)\mathbf{u}_2) \\ &= \mathbf{e}_2 - 2(\mathbf{u}_2^T\mathbf{e}_2)\mathbf{u}_2 - 2(\mathbf{u}_1^T\mathbf{e}_2)\mathbf{u}_1 + 4(\mathbf{u}_2^T\mathbf{e}_2)(\mathbf{u}_1^T\mathbf{u}_2)\mathbf{u}_1. \\ \mathbf{q}_3 &= H_1 H_2 \mathbf{e}_3 = H_1(\mathbf{e}_3 - 2(\mathbf{u}_2^T\mathbf{e}_3)\mathbf{u}_2) \\ &= \mathbf{e}_3 - 2(\mathbf{u}_2^T\mathbf{e}_3)\mathbf{u}_2 - 2(\mathbf{u}_1^T\mathbf{e}_3)\mathbf{u}_1 + 4(\mathbf{u}_2^T\mathbf{e}_3)(\mathbf{u}_1^T\mathbf{u}_2)\mathbf{u}_1 \end{aligned} \tag{39}$$

Using these relations, we found the approximated matrix Q, that is:

$$Q = \begin{bmatrix} -0.578 & 0.572 & 0.572 \\ -0.578 & 0.212 & -0.788 \\ -0.578 & -0.788 & 0.212 \end{bmatrix}. \tag{40}$$

Finally, we can verify that, multiplying Q and R and approximating as before, we recover the original matrix A. It is important to point out that the columns of Q are not perfectly orthonormal.

(4) Let $A \in \mathbb{R}^{m\times n}$, with $\operatorname{rank}(A) = n$, and let $\mathbf{b} \in \mathbb{R}^m$. Let $A = Q_1R_1$ be a reduced QR decomposition of A, where $Q_1 \in \mathbb{R}^{m\times n}$ has orthonormal columns and $R_1 \in \mathbb{R}^{n\times n}$ is upper triangular and nonsingular. Show that a reduced QR factorization of the augmented matrix $A_+ = [A\ \mathbf{b}]$ is given by:

$$A_+ = [Q_1\ \mathbf{q}_{n+1}] \begin{bmatrix} R_1 & \mathbf{z} \\ \mathbf{0}^T & \rho \end{bmatrix} \tag{41}$$

where $\mathbf{z} = Q_1^T\mathbf{b}$. Also, show that $|\rho| = \|\mathbf{b} - A\mathbf{x}^*\|_2$ where $\mathbf{x}^*$ is the solution to the least squares problem $\|\mathbf{b} - A\mathbf{x}\|_2 = \min$.

We start recalling that any matrix has a QR factorization, therefore the matrix A_+ must have it. In particular we know that $A_+ = Q_+R_+$, where Q_+ has orthonormal columns and R_+ is upper triangular. In order to find the value of $\mathbf{z}$ and ρ, we compute the product Q_+R_+ and impose that it is equal to A_+:

$$[Q_1\ \mathbf{q}_{n+1}] \begin{bmatrix} R_1 & \mathbf{z} \\ \mathbf{0}^T & \rho \end{bmatrix} = [Q_1R_1\ Q_1\mathbf{z} + \rho\mathbf{q}_{n+1}] = [A\ \mathbf{b}], \tag{42}$$

therefore we must have

$$\mathbf{b} = Q_1\mathbf{z} + \rho\mathbf{q}_{n+1}. \tag{43}$$

Multiplying both members of the last equation for Q_1^T we find that:

$$Q_1^T\mathbf{b} = Q_1^TQ_1\mathbf{z} + \rho Q_1^T\mathbf{q}_{n+1} = \mathbf{z} \tag{44}$$

where we used that the columns of Q_1 are orthonormal and $\mathbf{q}_{n+1}$ is orthogonal to all the columns of Q_1.

Now we consider the complete QR factorization of A_+:

$$A_+ = QR = [Q_1\ Q_2] \left[\begin{array}{cc} R_1 & \mathbf{z} \\ \mathbf{0}^T & \rho \\ \hline \multicolumn{2}{c}{\mathbf{0}} \end{array}\right] = [Q_1R_1\ \ Q_1\mathbf{z} + Q_2(\rho, 0, \dots, 0)^T] \tag{45}$$

obtaining the relation

$$\mathbf{b} = Q_1\mathbf{z} + Q_2(\rho, 0, \dots, 0)^T \tag{46}$$

and, therefore, multiplying both sides for Q_2^T:

$$Q_2^T\mathbf{b} = \rho(1, 0, \dots, 0)^T. \tag{47}$$

Now we define the residual $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{b} - A\mathbf{x}^* \tag{48}$$

where $\mathbf{x}^*$ is the solution to the least square problem, and compute:

$$\|\mathbf{r}\|_2^2 = \|Q^T\mathbf{r}\|_2^2 = \| \begin{bmatrix} Q_1^T\mathbf{b} \\ Q_2^T\mathbf{b} \end{bmatrix} - \begin{bmatrix} Q_1^TA\mathbf{x}^* \\ Q_2^TA\mathbf{x}^* \end{bmatrix} \|_2^2 = \|Q_1^T\mathbf{b} - R_1\mathbf{x}^*\|_2^2 + \|Q_2^T\mathbf{b}\|_2^2 \tag{49}$$

where we used the invariance of the 2-norm with respect to unitary transformations. Knowing that $\mathbf{x}^* = R_1^{-1}Q_1^T\mathbf{b}$ we obtain:

$$\|\mathbf{r}\|_2 = \|Q_2^T\mathbf{b}\|_2 = \|\rho(1, 0, \dots, 0)^T\|_2 = |\rho| \tag{50}$$

that concludes the proof. □

Problem 3

(1) Let $A \in \mathbb{R}^{m\times n}$, with $\text{rank}(A) = n$. The singular value decomposition of A can be written as $A = U\Sigma V^T$ or, equivalently, as $A = \sum_{i=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$, where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ represent the singular values of A, while $\{\mathbf{u}_i\}_{i=1,\ldots,n}$ and $\{\mathbf{v}_i\}_{i=1,\ldots,n}$ are, respectively, left and right singular vectors of A.

Express the singular values and singular vectors of the following matrices in terms of those of A.

(a) $(A^TA)^{-1}$

$$(A^TA)^{-1} = (V^T)^{-1}\Sigma^{-2}V^{-1} = V\Sigma^{-2}V^T$$

In order to obtain this expression we used that $U^TU = \mathbb{1}$ and that $V^{-1} = V^T$, since U and V are orthogonal. We can conclude that both left and right singular vectors of $(A^TA)^{-1}$ are equal to the right singular vectors of A, while singular values of the former matrix are equal to those of A raised to the -2 power. Note that, since singular values of A are in increasing order and in this expression they are raised to a negative power, they must appear in reverse order.

(b) $(A^TA)^{-1}A^T$

$$(A^TA)^{-1}A^T = V\Sigma^{-2}V^TV\Sigma U^T = V\Sigma^{-1}U^T$$

Analogously to the previous case, we know that $V^TV = \mathbb{1}$. Here we see that left singular vectors of $(A^TA)^{-1}A^T$ are equal to the right singular vectors of A, right singular vectors of $(A^TA)^{-1}A^T$ are equal to the left singular vectors of A, while singular values of this matrix are equal to the inverse of singular values of A.

(c) $A(A^TA)^{-1}$

$$A(A^TA)^{-1} = U\Sigma V^TV\Sigma^{-2}V^{-T} = U\Sigma^{-1}V^T$$

Note that this result can also be obtained from the previous case, by noting that $A(A^TA)^{-1} = ((A^TA)^{-1}A^T)^T$. We can see that left and right singular vectors of $(A^TA)^{-1}$ coincide with that of A, while singular values of this matrix are equal to the inverse of singular values of A.

(d) $A(A^TA)^{-1}A^T$

$$A(A^TA)^{-1}A^T = U\Sigma^{-1}V^TV\Sigma U^T = UU^T$$

In this case, both left and right singular vectors of $A(A^TA)^{-1}A^T$ are equal to left singular vectors of A, while singular values of this matrix are all equal to 1.

(2) Given the following matrix:

$$A = \begin{bmatrix} 1 & 2 \\ 0 & 2 \end{bmatrix}, \tag{51}$$

its singular values can be computed by considering the square root of the eigenvalues of the matrix A^TA, which has the form:

$$A^TA = \begin{bmatrix} 1 & 2 \\ 2 & 8 \end{bmatrix}. \tag{52}$$

Hence, by imposing that $p(\lambda) = \det(A^TA - \lambda\mathbb{1}) = 0$, we find the two eigenvalues of A^TA and, therefore, the two singular values of A, which are:

$$\sigma_{1,2} = \sqrt{\frac{9 \pm \sqrt{65}}{2}}. \tag{53}$$

Finally, knowing that the spectral condition number is defined as $k_2(A) = \frac{\sigma_{\max}}{\sigma_{\min}}$, we can find the condition number of A:

$$k_2(A) = \sqrt{\frac{9+\sqrt{65}}{9-\sqrt{65}}} \simeq 4.26. \tag{54}$$

We want now to see how the unit ball is modified by the transformation described by the matrix A. In other words, we want to see what is the image of a vector $\mathbf{x} = (\cos(\theta), \sin(\theta))$, parametrizing the unit ball on a plane, when considering the linear transformation $\mathbf{y} = A\mathbf{x}$. The vector $\mathbf{y}$ under the transformation A assumes the form:

$$\mathbf{y} = (\cos(\theta) + 2\sin(\theta), 2\sin(\theta)) \quad \theta \in [0, 2\pi)\,.$$

Graphically, we have:

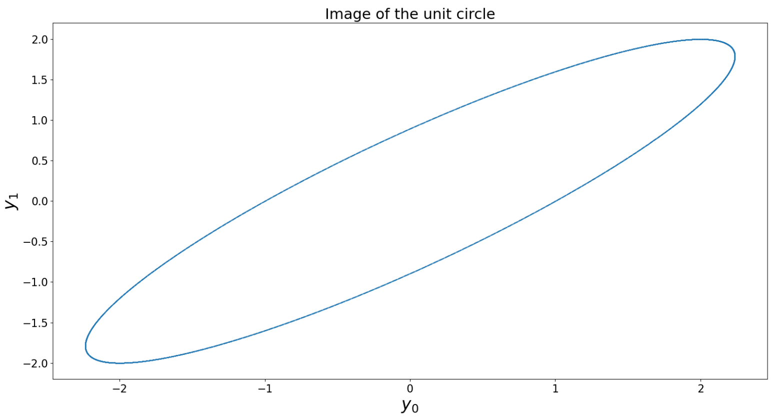


Figure 2: Image of the unit circle

(3) jvejkvbkdjf

(4) Given a matrix

$$A = \begin{bmatrix} -4 & -2 & -4 & -2 \\ 2 & -2 & 2 & 1 \\ -800 & 200 & -800 & -401 \end{bmatrix}. \tag{55}$$

we want to compute the the singular values of A, the Moore-Penrose pseudoinverse of A and its spectral condition number. We report in the following the Python script that we used to do it.

```python
import numpy as np

A = np.array([[-4.,-2.,-4.,-2.],[2.,-2.,2.,1.],[-800,200,-800,-401]])
U, singular_values, V_transpose = np.linalg.svd(A, compute_uv=True)
pseudoinverse = np.linalg.pinv(A)
spectral_cond_num = np.linalg.cond(A)
print(f'singular values of A = {singular_values}')
print(f'pseudoinverse of A = {pseudoinverse}')
print(f'spectral condition number of A = {spectral_cond_num}')
```

The results we obtained are reported below:

```
singular values of A = [1.21689895e+03 3.30829410e+00 4.21538860e-03]
pseudoinverse of A = [[-2.50833333e+01  5.00833333e+01  2.50000000e-01]
                      [-1.66666667e-01 -3.33333333e-01  7.57226996e-16]
                      [-2.50833333e+01  5.00833333e+01  2.50000000e-01]
                      [ 1.00000000e+02 -2.00000000e+02 -1.00000000e+00]]
spectral condition number of A = 288680.1350686104
```

Given the obtained results, we can say that $\text{rank}(A) = 3$.

(5) The best rank-k approximation of a matrix A, in the Frobenius norm, is obtained by computing the singular value decomposition of A and then by considering only the first k terms in that expression. In other words, given a matrix A written in terms of its singular value decomposition as $A = \sum_{i=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$, its best rank-k approximation is given by $\tilde{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$. Below is reported the Python code that we used to obtain the best rank-1 and rank-2 approximations of the matrix A mentioned in the previous point of this exercise.

```python
A_rank_1 = singular_values[0]*np.outer(U[:,0],V_transpose[0])
A_rank_2 = A_rank_1 + singular_values[1]*np.outer(U[:,1],V_transpose[1])
spectral_cond_num_A_rank_2 = singular_values[0]/singular_values[1]
print(f'A_rank_1 = {A_rank_1}')
print(f'A_rank_2 = {A_rank_2}')
print(f'Spectral condition number of A_rank_2 =
 ↪ {spectral_cond_num_A_rank_2}')
```

We obtained the following two matrices:

```
A_rank_1 = [[-3.67478811e+00  9.18652344e-01 -3.67478811e+00 -1.84198740e+00]
            [ 2.16152803e+00 -5.40355726e-01  2.16152803e+00  1.08346585e+00]
            [-8.00001057e+02  1.99990537e+02 -8.00001057e+02 -4.01000500e+02]]

A_rank_2 = [[  -3.99955481   -2.00000178   -3.99955481   -2.00177721]
            [   1.99910978   -1.99999645    1.99910978    1.00355377]
            [-800.00000445  200.00000002 -800.00000445 -400.99998223]]

Spectral condition number of A_rank_2 = 367.8327598758788
```

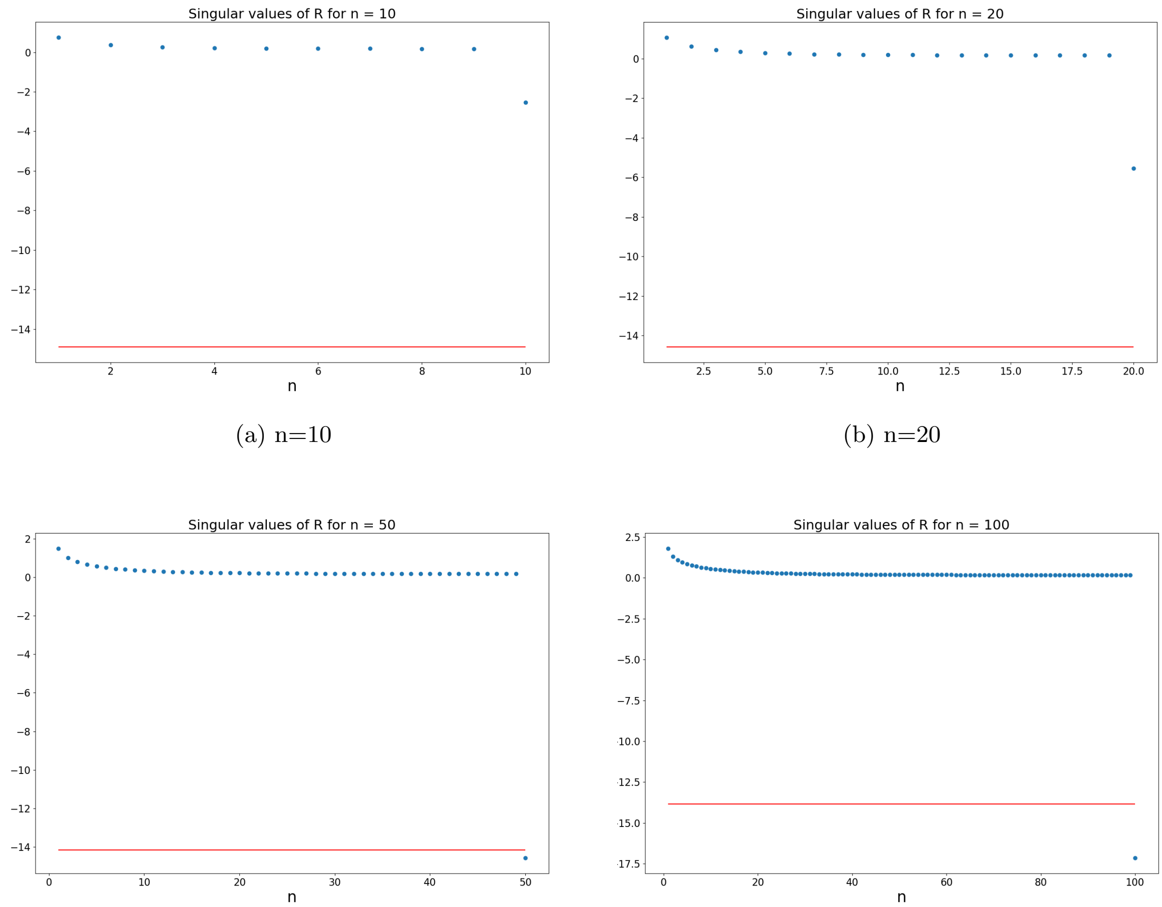
As we can see, the best rank-1 approximation of A returns a matrix that is already quite similar to the initial one, and the approximation improves when considering the best rank-2 approximation of A. This can be seen, in a more quantitative way, by considering the Frobenius norm of the difference between the approximated matrix and the initial matrix, both for the rank-1 and rank-2 approximation. By doing this, one obtains:

```
Frobenious norm between A and A_rank_1 = 3.308296788282531
Frobenious norm between A and A_rank_2 = 0.004215388599497665
```

(6) Consider an upper triangular matrix matrix $R = (r_{ij})$, whose entries are given by $r_{ii} = 1$ and $r_{ij} = -1$ for $j > i$. We defined a function `R_matrix(n)` that allows us to compute this matrix for a given dimension n:

```python
def R_matrix(n):
 R = np.triu(-np.ones((n,n)),k=1) + np.eye(n)
 return R
```

Note that we used an upper triangular mask to obtain the entries r_{ij}. Once derived R, we computed its singular values for $n = 10, 20, 50, 100$, as required. The plot of the singular values are reported below.



(a) n=10

(b) n=20

(c) n=50

(d) n=100

Figure 3: Singular values of the matrix R defined at the beginning of this exercise for $n = 10, 20, 50, 100$.

It can be observed that for $n = 50$ and for $n = 100$ the last singular value becomes smaller than the threshold value fixed by $u\sigma_1$, where u is the machine precision. This means that, when this happens, the matrix R becomes numerically singular.

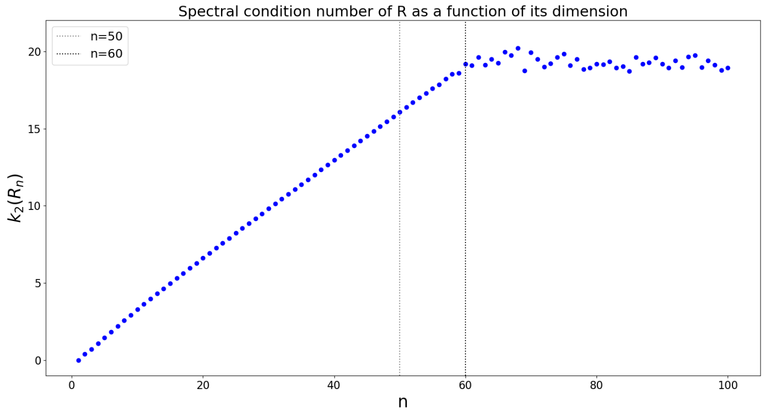


Figure 4: Spectral condition number of R as a function of its dimension.