University of Toronto
Department of Electrical and Computer Engineering

ECE367 MATRIX ALGEBRA AND OPTIMIZATION

Problem Set #5

Autumn 2021

Prof. S. C. Draper **Due:** 11:59pm (Toronto time) Monday, 22 November 2021

Homework policy: Problem sets must be turned by the due date and time. Late problem sets will not be accepted. See the information sheet for further details. The course text "Optimization Models" is abbreviated as "OptM" and "Introduction to Applied Linear Algebra" as "IALA".

Problems are categorized as:

- **"Theory" problems:** These are mostly mathematical questions designed to give you deeper insight into the fundamentals of the ideas introduced in this class.
- **"Application" problems:** These questions are designed to expose you to the breadth of application of the ideas developed in class and to introduce you to useful numerical toolboxes. Problems of this sort often ask you to produce plots and discuss your results; said plots and discussions should be included in and form part of your submission – think of your submitted solution like a lab book. Your attached code simply provides back-up evidence.
- **"Optional" problems:** Optional problems provide extra practice or introduce interesting connections or extensions. They need not be turned in. I will assume you have reviewed and understood the solutions to the optional problems when designing the exams.

Hand-in procedure:

- **Initial submission:** Your initial submission of the "Theory" and "Application" questions must be submitted via Quercus upload by the due date. Click on the `Assignments` tab, then look for the `Initial submission` tab and upload under the correct problem set number.
- **Self-assessment:** After the problem set is due we will post solutions. You will have one week from the initial due date to submit a commented version of your assignment in which, using as a reference the posted solutions, you highlight your errors or omissions in red. Annotate the PDF you initially submitted. If you have not submitted a solution you cannot submit the self-assessment. To submit the self-assessment on Quercus, click on the `Assignments` tab, then look for the `Self-assessment` tab and upload under the correct problem set number.
- **Late problem sets are not accepted**
- **Grading:** Per the course handout problem sets are graded for completion only. Points are assigned to (i) Initial submission of theory part, (ii) Submission of application part, (iii) Self-assessment of theory part. The relative points breakdown is detailed in each problem set.

Points allocation

- Theory parts (initial assessment): 1 pt
- Application parts (initial assessment): 1 pt
- Theory parts (self-assessment): 1 pt

Problem categorization and main concepts covered

Theory

- SVD and ellipsoids: Problem 5.1
- Least squares: Problems 5.2-5.5

Application

Note: There are THREE application problems in this problem set.

⇒ You must **complete any ONE of the THREE application problems** to receive full credit.

Pick the one you find most interesting – the technical topics & application area for each are indicated below. Doing more than one of the following three problem is **optional**. That said, once posted, please review the solutions to all three.

- PCA (Eigenfaces): Problem 5.6
- Under-determined least squares (optimal control): Problem 5.7
- Over-determined least squares (computer-aided tomography): Problem 5.8

Optional

- None on this problem set

THEORY

Problem 5.1 (SVDs and ellipsoids)
This problem concerns the matrix

$$A = \frac{1}{\sqrt{10}} \begin{bmatrix} 5 & 0 \\ 3 & 4 \end{bmatrix}.$$

(a) Find the SVD of A, $A = U\Sigma V^T$, specifying the matrix of normalized left-singular vectors U, the matrix of normalized right-singular vectors U, and the matrix of (non-negative) singular values Σ. (Since A is square, $\tilde{\Sigma} = \Sigma$.)

(b) Use your results from part (a) to write the SVD in outer-product form, as $A = \sigma_1 u^{(1)}(v^{(1)})^T + \sigma_2 u^{(2)}(v^{(2)})^T$.

Now consider the action of A on a unit vector x such that $\|x\|_2 = 1$ (or, in part (e), $\|x\|_2 \leq 1$). In answering parts (c)–(e) it may help to use the outer-product form from part (b) to write

$$Ax = U\Sigma V^T x = U\Sigma\bar{x} = U \begin{bmatrix} \sigma_1\bar{x}_1 & 0 \\ 0 & \sigma_2\bar{x}_2 \end{bmatrix} = u^{(1)}\sigma_1\bar{x}_1 + u^{(2)}\sigma_2\bar{x}_2, \tag{1}$$

where $\bar{x} = V^T x$ and $\|\bar{x}\|_2 = 1$ since $\|x\|_2 = 1$ and V is an orthogonal matrix.

(c) What is the unit input direction x, such that $\|x\|_2 = 1$, that leads to the greatest amplification (the largest $\|Ax\|_2$)? What is the amplification? What output direction Ax results from setting x equal to the input direction that yields the greatest amplification?

(d) What is the unit input direction x, such that $\|x\|_2 = 1$, that leads to the least amplification (the smallest $\|Ax\|_2$)? What is the amplification? What output direction Ax results from setting x equal to the input direction that yields the least amplification?

(e) Sketch the set $\{Ax \in \mathbb{R}^2 : \|x\|_2 \leq 1, x \in \mathbb{R}^2\}$. This set is the *image* of the unit ball under the linear map $f(x) = Ax$.

(f) Now, sketch the set $\{x \in \mathbb{R}^2 : \|Ax\|_2 \leq 1\}$. The logic is related to that in (c)-(e), but note that here you are asked to identify the inputs for which the corresponding *outputs* are constrained to the unit ball. This set is the *pre-image* of the unit ball under the linear map $f(x) = Ax$.

Problem 5.2 (Fitting functions: LS, weighted LS, regularized LS)
This problem concerns using least squares to solve the problem of fitting a function to data. In this problem we use variants of the least squares method to fit a variety of functions to a set of four data points:

$$\{(x_i, y_i)\}_{i=1}^4 = \{(0,4), (1,1), (2,4), (3,2)\}$$

In all these parts it is recommended to use a computer to calculate and plot the fits; it is also okay to sketch by hand. Please show your work *setting up* the various least squares problem that you will use the computer to solve numerically (and then to plot).

(a) Use least square to fit the "best" affine function $y = \alpha_0 + \alpha_1 x$ to the data. (Note here we are using the α_i as the parameters you want to solve for and (x_i, y_i) for the data; in lecture we used x_i to denote the parameters we wanted to solve for.) First, according to (unweighted) least squares, what are the best choices for α_0 and α_1? What are the resulting residuals $r_i = y_i - (\alpha_0 + \alpha_1 x_i)$ for $i \in [4]$? What is the norm-squared of the residual vector $\|r\|_2^2$ where $r = (r_1, r_2, r_3, r_4)$? Plot your affine function $f_{\text{affine}}(x) = \alpha_0 + \alpha_1 x$ over the range $-1 \leq x \leq 5$ and also plot the four data point (x_i, y_i) for comparison.

In parts (b)-(d) we consider fitting the data using the **Weighted Least Squares** criterion where the PSD (not necessarily diagonal) weighting matrix is

$$W = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{a+b}{2} & \frac{a-b}{2} & 0 \\ 0 & \frac{a-b}{2} & \frac{a+b}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

and where various choices for a and b are specified in the ensuing parts.

(b) Find the best fit line for $a = b = 3$. This is the diagonally-weighted case. What are the best choices for α_0 and α_1? What are the resulting residuals $r_i = y_i - (\alpha_0 + \alpha_1 x_i)$ for $i \in [4]$? What is the norm-squared of the residual vector $\|r\|_2^2$? Plot your affine function over the range $-1 \leq x \leq 5$ and also plot the four data point (x_i, y_i) for comparison. Finally, comment on the differences versus the line you found in (a).

In the following two parts we use a non-diagonal PSD weighted matrix W. To help develop an interpretation note that the eigen-decomposition of the 2×2 matrix in the middle of W is

$$\begin{bmatrix} \frac{a+b}{2} & \frac{a-b}{2} \\ \frac{a-b}{2} & \frac{a+b}{2} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

(c) Find the best fit line for $a = 4$ and $b = 2$. What are the best choices for α_0 and α_1? What are the resulting residuals $r_i = y_1 - (\alpha_0 + \alpha_1 x_i)$ for $i \in [4]$? What is the norm-squared of the residual vector $\|r\|_2^2$? Plot your affine function over the range $-1 \leq x \leq 5$ and also plot the four data point (x_i, y_i) for comparison. Comment on the differences versus the line you found in (b), focusing on how the current choice of a and b changes the residuals. (Note that the diagonal entries of the W matrix in part (c) match those of the W matrix in part (b), it is only the off-diagonals that have changed.)

(d) Find the best fit line for $a = 2$ and $b = 4$. What are the best choices for α_0 and α_1? What are the resulting residuals $r_i = y_i - (\alpha_0 + \alpha_1 x_i)$ for $i \in [4]$? What is the norm-squared of the residual vector $\|r\|_2^2$? Plot your affine function over the range $-1 \leq x \leq 5$ and also plot the four data point (x_i, y_i) for comparison. Comment on the differences versus the line you found in (c), again focusing on the role played by the off-diagonal entries of W.

As was mentioned in class, to fit a function to data using the least-squares method, the function need not be linear (or affine), what is important is that the function is linear in its coefficients. In this part you use least squares to fit quadratic and cubic functions to the data.

(e) Use the least-squares method to fit the quadratic function $f_{\text{quad}}(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2$ and the cubic function $f_{\text{cubic}}(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3$ to the data. For each function what are the best choices for the α_i, what are the resulting residuals, and what are the $\|r\|_2^2$? Compare the norm of the residuals to each other and to what you found in part (a).

Finally, we turn to regularized least squares, concentrating on a cubic fit. Here

$$\alpha^* = \arg\min_{\alpha \in \mathbb{R}^4} \sum_{i=1}^{4} \Big(y_i - (\alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \alpha_3 x_i^3) \Big)^2 + \gamma \|\alpha\|_2^2,$$

where $f_{\text{cubic},\gamma}(x) = \alpha_0^* + \alpha_1^* x + \alpha_2^* x^2 + \alpha_3^* x^3$.

(f) Find the regularized cubic fit $f_{\text{cubic},\gamma}(x)$ for $\gamma = 0.05$ and $\gamma = 0.5$. Sketch or plot $f_{\text{cubic},\gamma}$ for the interval $-2 \leq x \leq 6$ for $\gamma = 0$, $\gamma = 0.05$ and $\gamma = 0.5$. Note that the plot of $f_{\text{cubic},0}$ is the same as that which you plotted in part (e).

Problem 5.3 (Lotka's law and least squares)
OptM Problem 6.3. (Hint: Think about doing a variable transformation to make the problem look more like a least-square problem.)

Problem 5.4 (ℓ_2 regularized least squares), from a previous exam

This problem considers optimization problems of the form

$$\min_{x \in \mathbb{R}^2} \|Ax - y\|_2^2 + \gamma \|x\|_2^2 \tag{2}$$

where $\gamma \in \mathbb{R}$ and $\gamma \geq 0$. In this problem we denote the x that minimizes (2) as x_γ^*.

(a) First consider the case where

$$A = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 2 & 1 \end{bmatrix}, \qquad y = \begin{bmatrix} 4 \\ -2 \\ 5 \end{bmatrix}, \qquad \gamma = 0.$$

Find x_0^*, the optimal variable that solves (2) for the parameters given. (To be doubly clear, $x_0^* = x_\gamma^*|_{\gamma=0}$, which is read as "$x_\gamma^*$ evaluated at $\gamma = 0$".)

(b) Re-derive the following relation derived in class, *fully justifying* your steps:

$$\|Ax - y\|_2^2 = \|Ax_0^* - y\|_2^2 + \|A(x - x_0^*)\|_2^2,$$

again note that $x_0^* = x_\gamma^*|_{\gamma=0}$.

Parts (c) and (d) of this problem concern Figure 1. In Figure 1 we plot two *possible* paths for x_γ^* as a function of γ for the same values of A and y that are specified in part (a).

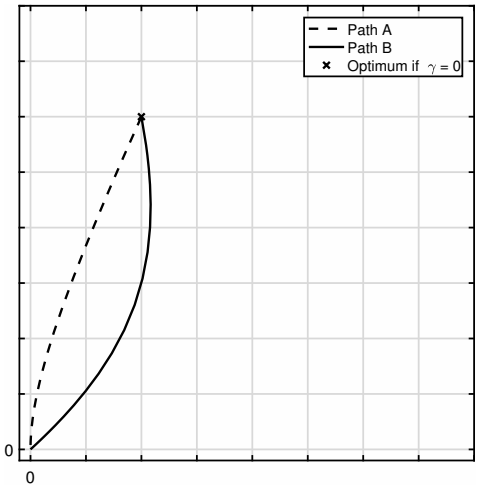

Figure 1: Path followed by x_γ^* as a function of γ.

(c) Make a sketch of Fig. 1 in your solutions. To your sketch add some level sets of the objective of (2) for the case where $\gamma = 0$. You must show below work that mathematically justifies your sketches of the level sets.

(d) *You must base your answer to this part on your answer to part (c)*:

Which path does x_γ^* follow as one increases γ? In the space provided below both clearly indicate *which* path (either the upper, dashed, "Path A" or the lower, solid, "Path B") you think x_γ^* follows and provide an explanation for your choice *based on your answer to part (c)*.

Problem 5.5 (Solving least squares problems using the Moore-Penrose pseudoinverse) In this problem you derive the result that the Moore-Penrose pseudoinverse can be used to solve least squares problems for overdetermined, underdetermined, and rank-deficient systems. Recall that least square problems consider the system of linear equations $Ax = b$ where $A \in \mathbb{R}^{m,n}$ and $\text{rank}(A) = r$. If the compact SVD of A is $A = U_r \Sigma V_r^T$ where $\Sigma \in \mathbb{R}^{r,r}$ is a positive definite diagonal matrix of (non-zero) singular values, $U_r \in \mathbb{R}^{m,r}$ and $V_r \in \mathbb{R}^{n,r}$ each contain orthonormal columns, then the Moore-Penrose pseudoinverse is $A^\dagger = V_r \Sigma^{-1} U_r^T$.

(a) Recall that the *overdetermined* least squares problem considers an $A \in \mathbb{R}^{m,n}$ when $m > n$ (more constraints in the y vector than parameters in the x vector). Here the objective is to find an x that minimizes $\|Ax - y\|_2$. Show that an optimal solution is $x^* = A^\dagger y$. To show this recall that an optimal solution x^* must satisfy the "normal" equations $A^T A x^* = A^T y$. Verify that $x^* = A^\dagger y$ satisfies the normal equations.

(b) Recall that the *underdetermined* least squares problem considers an $A \in \mathbb{R}^{m,n}$ when $m < n$ (fewer constraints in the y vector than parameters in the x vector). Here the objective is to find the x that minimizes $\|x\|_2$ while satisfying $Ax = y$ (equivalently, satisfying $\|Ax - y\|_2 = 0$). Show that the optimal solution $x^* = A^\dagger y$. To show this recall that the optimal solution x^* must satisfy two conditions: (i) $x^* \in \mathcal{R}(A^T)$ and (ii) $Ax^* = y$. Verify that $x^* = A^\dagger y$ satisfies (i) all the time. Under what conditions does x^* satisfy condition (ii)? (Hint, think about the rank of A.)

In the above two parts we haven't explicitly considered the role of the rank r of the A matrix. But we also note that the only place where the rank of A comes into the discussion of parts (a) and (b) is in the discussion of condition (ii) of part (b).

Recall that $r \leq \min\{m, n\}$. When $r = \min\{m, n\}$ the A matrix is full column rank in the overdetermined problem and is full row rank in the underdetermined problem. In both these full-rank cases x^* has the simple expression presented in class. Now we consider what happens when $r < \min\{m, n\}$.

(c) In this part consider the situation where $\text{rank}(A) = r < m < n$. This is a "rank-deficient" underdetermined least squares problem. If we set $x^* = A^\dagger y$, what characteristics do Ax^* and $\|x^*\|_2$ satisfy? (Hint: this is a type of hybrid problem that at the same time can have characteristics of both overdetermined and underdetermined least squares.)

APPLICATIONS

NOTE: The requirement is to complete any **ONE** of the three following "application" problems.

Problem 5.6 (Eigenfaces and ℓ_2 projection)
In this problem you will familiarize with the concept of Eigenfaces and its uses. Download the dataset `yalefaces.mat` from the course website. This dataset consists of 32×32 gray scale images of faces taken from the *Extended Yale Face Database B* (`http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html`). Load the dataset into your MATLAB environment by executing the command `load('yalefaces.mat')`. You will see a new variable `M` of dimension $32 \times 32 \times 2414$ which consists of 2414 grayscale images, 32×32 pixels each. The pixel values of the images range from 0 to 255. You can view the loaded images by making use of MATLAB built-in functions `imshow` or `imagesc`. As an example, the first image of the dataset can be displayed by executing `imshow(M(:,:,1)/255)`.

Let N be the number of images in the dataset and let $d = 1024$, the total number of pixels in each image. An image can be thought of as a matrix with 32 columns and 32 rows consisting of entries in the range $[0, 255]$. In this exercise we consider the images in vector form. Let $\mathcal{J}_i$ be the ith image in the dataset where $i \in [N]$. We formulate a column vector $x^{(i)} \in \mathbb{R}^d$ by flattening the matrix that makes up $\mathcal{J}_i$. In our case we stack all 32 columns vertically to form a d-dimensional vector $x^{(i)}$. In MATLAB you can do this efficiently using the command `reshape`. The 'average face' vector of the dataset $\bar{x}$ can be computed as $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}$. We can (roughly) visualize the $x^{(i)}$s as forming a cloud of data points where the cloud is centered at $\bar{x}$ in d-dimensional space. Translate the cloud center to the origin by subtracting $\bar{x}$ from each $x^{(i)}$. Let the resulting vectors be denoted as $\bar{x}^{(i)} = x^{(i)} - \bar{x}$, which we will refer to as centered image vectors. Construct a matrix $X \in \mathbb{R}^{d\times N}$ by concatenating all the centered vectors $\bar{x}^{(i)}$, i.e., $X = \left[\bar{x}^{(1)}, \bar{x}^{(2)}, \ldots, \bar{x}^{(N)}\right]$. The matrix $C = XX^T$ is N times the covariance matrix of the data samples $x^{(i)}, i \in [N]$.

(a) In class you learned about singular value decomposition (SVD) and eigendecomposition. What is the connection between the singular values of X and the eigenvalues of C? What is the connection between the left-singular vectors of X and the eigenvectors of C? Make sure to describe the reasoning behind your answers, e.g., by describing the singular or eigen decompositions of each matrix.

(b) Compute the eigenvalue/eigenvector pairs of C and arrange them in decreasing order of the magnitude of the eigenvalues. Let the jth pair in the ordered set be denoted as $\lambda_j \in \mathbb{R}, v^{(j)} \in \mathbb{R}^d$ for $j \in [d]$. You may find the MATLAB command `svd` or `eig` helpful. Comment whether the eigenvalues are real and briefly explain why. Plot $\log \lambda_j$ against $j \in [d]$. Include your plot in your solution.

(c) The set of eigenvectors you computed in the previous part form a basis for the centered image vectors. Reshape each eigenvector $v^{(j)}$ to obtain a 32×32 matrix. These matrices can be considered as images and they are known as *eigenfaces*. Include plots of two sets of eigenfaces, those corresponding to the largest 10, and those corresponding to the smallest 10, eigenvalues. Do you observe any difference between the two sets of eigenfaces you plotted? If so briefly explain the reason for this difference.

(d) In this part, consider the images $\mathcal{J}_i$ for $i \in \{1, 1076, 2043\}$. Let $\mathcal{B}_j = \{v^{(1)}, v^{(2)}, \ldots, v^{(j)}\}$ where $j = \{2^1, 2^2, 2^3, \ldots, 2^{10}\}$, i.e., j indexes sets each consisting of a different number of the eigenvectors. Let us denote the ℓ_2 projection of $\bar{x}^{(i)}$ onto the basis vector set $\mathcal{B}_j$ as $\bar{y}^{(i,j)}$. Compute $\bar{y}^{(i,j)}$ for the given i, j pairs using MATLAB. (I.e., do this using your numerical tool and not by hand.) Note that $\bar{y}^{(i,j)}$ vectors are computed using the centered image vectors. Translate these vectors back (un-center them) by the cloud center shift $\bar{x}$ to get the image vectors $y^{(i,j)} = \bar{y}^{(i,j)} + \bar{x}$. Reshape the $y^{(i,j)}$ vectors into 32×32 matrices and plot them as images. Note that you will need to plot 30 images and these can be compactly plotted using `subplot` command in MATLAB.

(e) In this part you will learn how the eigenfaces can be used for the task of *face recognition*.

Consider the two sets of indices $\mathcal{I}_1 = \{1, 2, 7\}$ and $\mathcal{I}_2 = \{2043, 2044, 2045\}$. The faces in the set $\mathcal{J}_i$ for $i \in \mathcal{I}_1$ belong to one person and those for $i \in \mathcal{I}_2$ belong to a second person. We have carefully picked the image indices so that the corresponding images are taken under similar lighting conditions. Consider $\mathcal{B}_{25}$ where $\mathcal{B}_j$ is defined as in part (d). Compute the projection coefficients obtained by projecting $\bar{x}^{(i)}, i \in \mathcal{I}_1 \cup \mathcal{I}_2$ onto the eigenvectors in $\mathcal{B}_{25}$. Let $c^{(i)} \in \mathbb{R}^{25}$ be the vector that consists of coefficients obtained for ith image. The vector $c^{(i)}$ can be thought of as a representation of the corresponding original image $\mathcal{J}_i$. Intuitively, images that belong to same person should have similar coefficient vectors. To verify this, compute the pairwise Euclidean distances between the $c^{(i)}$ vectors. Tabulate the values and comment whether the distance between any two $c^{(i)}$ vectors that belong to the same person is smaller than those belonging to the other person. Briefly explain how you can use this to build a simple face recognition scheme.

Problem 5.7 (Optimal control of a unit mass)
Consider a unit mass with position $x(t)$ and velocity $\dot{x}(t)$ subject to force $f(t)$, where the force is piecewise constant over intervals of duration one second, i.e., $f(t) = p_n$ for $n - 1 < t \leq n$, $n = 1, \cdots, 10$; we consider the system for 10 seconds in total. Ignore friction. Assume the mass has zero initial position and velocity, i.e., $x(0) = \dot{x}(0) = 0$.

(a) Derive the "state-space" equations that describe a discrete-time version of the dynamics of this system. In particular, derive relationships between $x(n)$ and $\dot{x}(n)$ in terms of $x(n-1)$, $\dot{x}(n-1)$, and the driving force p_n, for each $n \in \{1, 2, \ldots, 10\}$. (You will need to use your knowledge of basic physics to determine these relationships.) The two equations you derive should be

$$x(n) = x(n-1) + \dot{x}(n-1) + (1/2)p_n,$$
$$\dot{x}(n) = \dot{x}(n-1) + p_n,$$

which you can then stack up in vector form to form the state-space equations

$$\begin{bmatrix} x(n) \\ \dot{x}(n) \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} x(n-1) \\ \dot{x}(n-1) \end{bmatrix} + \underbrace{\begin{bmatrix} 1/2 \\ 1 \end{bmatrix}}_{b} p_n. \tag{3}$$

We note that this continuous time system can only be discretized exactly because in this system the control effort $f(t)$ is held constant over each time interval. (At this point we recommend you remind yourself of the discussion of state-space models of linear dynamical systems in OptM Exercise 3.4.)

(b) Find the p_n, $n \in [10]$ that minimizes

$$\sum_{n=1}^{10} p_n^2$$

subject to the following specifications: $x(10) = 1$, $\dot{x}(10) = 0$. Plot the optimal f, the resulting x and $\dot{x}$. (I.e., plot $f(t)$, $x(t)$, and $\dot{x}(t)$ for $0 \leq t \leq 10$.) Give a short intuitive explanation of what you see.

(c) Suppose that we add one more specification $x(5) = 0$, i.e., we require the mass to be at position 0 at time 5. Plot the optimal f, the resulting x and $\dot{x}$. Give a short intuitive explanation of what you see.

Problem 5.8 (CAT scan imaging)

In this problem you will obtain a basic understanding of the math behind computer aided tomography (CAT) scanning. Before proceeding we recommend you read the Example 6.6 (*CAT scan imaging*) in OptM to obtain a basic understanding about the CAT scan. In the following parts, we assume the same meaning for y, A and x as in Example 6.6. Download the file `scanImage.p` from the course website. Although you will observe that the file is encrypted, this file defines a MATLAB function called `scanImage(M)` that takes in one optional argument and produces one output. You can execute this function just the same way you execute other MATLAB functions. This function encapsulates a "virtual CAT scanner" that is able to scan 2-d images. Consider a grayscale image of height h and width w. Such an image can be represented as a $h \times w$ matrix M with values between 0 and 255. Let $n = hw$ be the total number of pixels (in CAT-scan terminology this is the number of "voxels", as described in OptM). If you execute the function with M as the argument, it will return a vector $y \in \mathbb{R}^m$ which is the scan result. Similar to OptM Example 6.6, y is the vector of log-intensity ratios, and m corresponds to the number of beams used for the scan. Note that for the provided scanner, you have no control over the number of beams used or how beams are positioned across the image. However, the dimension m and the positioning of beams is a function only of h and w. This means that any two images which are of the same dimensions will be scanned with identical beam setups.

Similar to OptM, let us denote the vectorized image M by the n-dimensional column vector x. The input x can be related to the output y of the scanner by matrix $A \in \mathbb{R}^{m \times n}$ as $y = Ax$. In OptM Example 6.6, A is computed analytically, using geometric properties of the beam setup. For this numerical problem, even if you knew the positions (and angles) of the beams, computing A analytically would be ... tedious. We propose an alternative approach to estimate A. Observe that

$$[y_1\, y_2\, \cdots\, y_m]^T = \left[a^{(1)}\, a^{(2)}\, \cdots\, a^{(n)}\right] [x_1\, x_2\, \cdots\, x_n]^T\,, \tag{4}$$

where $a^{(i)}$ is the ith column of A. In the provided scanner, you have control over the input image. Consider the case where you obtain an image M_1 by setting the first pixel (the $(1,1)$ coordinate of M) to 1 and all others to 0. The vectorized representation of M_1 will be $[1, 0, \cdots, 0]^T$. If you put M_1 through the scanner, the resulting y should be equal to $a^{(1)}$ as per (4). This way, by turning each pixel on while all others are off, you can estimate each of the columns in A. As you may guess, this is not how things work with real-world CAT scanners, but eases our development herein.

(a) Now, using the method described in the text, write a MATLAB script to estimate A when $h = 50$ and $w = 50$. In this case you will observe the dimension of the scanner output, i.e., m is 1950. Since A is an $m \times n$ matrix, we can visualize A by treating it as an image. Use the MATLAB function `imshow(A, [])` to display A as an image and include a scaled down version of the image with your answers. Note that if `imshow(A, [])` displays a mostly black image, `imshow(-A, [])` will display a mostly white image.

(b) Till this point, you used the provided scanner function `scanImage(·)` to get the outputs for known images, yielding an estimate for A. In this part, you are given the scanner output y of an unknown image M_{un}, which has dimensions $h = 50$ and $w = 50$. The scanner output y for the image M_{un} can be obtained by executing `y=scanImage`, without passing an input to the function. (We use an encrpyted function so that in this part you need to determine an estimate of the image.) Since y and A are known, you can solve for an x that satisfies $y = Ax$. You will find that $\text{rank}\, A < m < n$ so this is an underdetermined and rank-deficient system of equations. However, the matrix A may also have non-zero singular values that are extremely close to zero which, to avoid numerical instabilities, you will need to assume to be zero. To choose an appropriate effective rank to work with, plot the r singular values σ_i of A versus their index $i \in [r]$. Visually inspect your plot to choose an effective minimum non-zero singular value. After solving for x, obtain the image M_{un} and include with your answers. What is the hidden message in the scanned image?

In a real-world CAT scanner, the number of voxels (n) usually is much larger than the number of beams (m) used to produce the vector y. Rather than least squares the 'inverse Radon transform' is typically used to solve for the vector x. Note that the number of voxels considered is proportional to the resolution of the image obtained. A typical CAT scanner provides around 0.5mm resolution, which means two adjacent voxels are around 0.5mm apart.

OPTIONAL

