EE263 Mid-term exam, November 2019

This is a 24 hour take-home exam. Please turn it in using Gradescope 24 hours after you receive it.

- You may use any books, notes, or computer programs (*e.g.*, Julia, Python or Matlab). You can Google anything. But **you may not discuss the exam with anyone** until Nov. 6, after everyone has taken the exam. The only exception is that you can ask the TAs or Professor Lall for clarification, by emailing to the staff email address `ee263-aut1920-staff@lists.stanford.edu`. We've tried pretty hard to make the exam unambiguous and clear, so we're unlikely to say much.

- Since you have 24 hours, we expect your solutions to be legible, neat, and clear. Do not hand in your rough notes, and please try to simplify your solutions as much as you can. We will deduct points from solutions that are technically correct, but much more complicated than they need to be.

- When a problem involves some computation, we do not want just the final answers. We want a clear discussion and justification of exactly what you did. Be sure to show us your verification that your computed solution satisfies whatever properties it is supposed to, at least up to numerical precision. For example, if you compute a vector x that is supposed to satisfy $Ax = b$ (say), show us the code that checks this, and the result. (This might be done by the code `norm(A*x-b)`; be sure to show us the result, which should be very small.) *We will not check your numerical solutions for you, in cases where there is more than one solution.*

- In the portion of your solutions where you explain the mathematical approach, you *cannot* refer to programming language operators, such as the backslash operator. (You can, of course, refer to inverses of matrices, or any other standard mathematical construct.)

- Some of the problems may be described in a practical setting, (imaging systems, energy consumption, population dynamics, or wireless communications). Others may discuss mathematical areas (Markov chains, differential equations). *You do not need to understand anything about these practical settings or mathematical areas to solve these problems.* We've taken special care to make sure all the information and math needed to solve the problem is given in the problem description.

- Some of the problems require you to download data files. These files can be found at the URL

    `http://ee263.stanford.edu/middlequestions.html`

There are no links on the course web page pointing there, so you'll have to type in the whole URL yourself. The data files are in `json` format. In Julia, use `readclassjson` to load the files. In Matlab, use `jason_helper.m` to load the files. These are available at the above web page.

- At the same web page as the data files we will post notification of any exam typos or problems. We recommend you check this site occasionally during the exam.

- Good luck!

1. **Tomography.** In tomography, a beam of X-rays is passed through a sample, and the total absorption along the path is measured. This process is repeated from many different angles, given a large number of measurements. If we divide the region containing the sample into volume cells (voxels), then the absorption along the path is proportional to the product of the absorption of the voxels that the beam passes through. Taking the logarithm makes this into a *linear* operation; the log of the absorption is proportional to the sum of the absorptions of the voxels that the beam passes through.

   In this question, we will look at a 2-dimensional version of tomography. We have a sample, and divide it up into a rectangular array of pixels. We'll represent the pixels as a matrix $X \in \mathbb{R}^{h \times w}$, and our array of pixels has height h and width w.

   We will shine m X-ray beams through this sample. Each beam is specified by its distance from the center of the sample, and its angle to the horizontal. If the beam has direction θ and distance from the center given by d, then we can calculate which pixels it is affected by. Specifically, pixel X_{ij} affects the beam if

   $$|(j - w/2)\sin\theta - (i - h/2)\cos\theta - d| < 2 \tag{1}$$

   For each beam there is a corresponding scalar measurement of log-absorption. This is given by the sum of the the log-absorption of the pixels affected by that beam.

   a) Draw a picture to interpret equation (1)

   b) Suppose we measure $y = Ax$ where $A \in \mathbb{R}^{m \times n}$. For recursive least squares, the standard algorithm is

   $$P(0) = 0 \in \mathbb{R}^{n \times n}$$
   $$q(0) = 0 \in \mathbb{R}^n$$
   for $k = 0, 1, \ldots,$
   $$P(k + 1) = P(k) + a_{k+1}a_{k+1}^{\mathsf{T}}$$
   $$q(k + 1) = q(k) + y_{k+1}a_{k+1}$$

   where a_i^{T} is the i'th row of A. Suppose we want to instead solve the regularized least-squares problem

   $$\text{minimize} \quad \|y - Ax\|^2 + \mu\|x\|^2$$

   How would you modify the recursive least squares algorithm to solve this problem?

   c) The file `tomodata.json` contains h, w, the height and width of the array of pixels in the sample. It also contains a matrix $L \in \mathbb{R}^{m \times 2}$, where each row corresponds to a measurement. The i'th row specifies the beam position, with $\theta_i = L_{i1}$ the angle of the i'th X-ray beam, and $d_i = L_{i2}$ the distance from the beam to the center. Finally it contains y, where y_i is the log-absorption of the beam.

   Use your modified recursive estimation algorithm, with $\mu = 0.1$, to recursively estimate the sample array X. Plot this (the log-absorption of each pixel), as an image, when
   $$k = 1, 10, 50, 100, 200, 500, 1000, 2000$$

Also plot the final estimate. Each plot should show an image of height h pixels and width w pixels, where the color of the (i, j) pixel is determined by X_{ij}. Larger values of X_{ij} should be shown as darker pixels.

Solution.
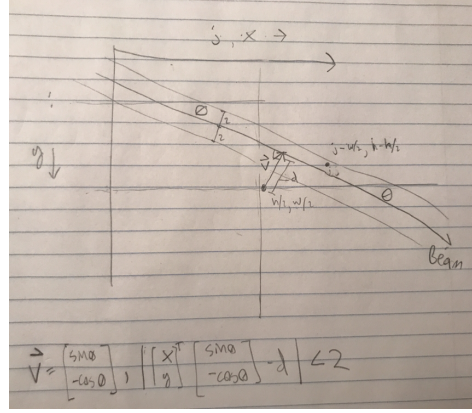
a) The equations indicates that the pixel, where x corresponds to $j - w/2$ and y corresponds to $i - h/2$ is within distance 1 of the line that makes an angle of θ with the horizontal, and is d away from the origin. Since $\|[-\sin\theta \ \cos\theta]^T\| = 1$, this is exactly given by the provided hyperplane expression.

You can also see this by writing the line in slope intercept form with slope $m = \frac{\sin\theta}{\cos\theta}$ and intercept $b = -\frac{d}{\cos\theta}$.

b) To modify the algorithm, simply let $P(0) = \mu I$. This is because from the formula for multi-objective least squares $\|Ax - y\|^2 + \mu\|Fx - g\|^2$ is

$$x = (A^T A + \mu F^T F)^{-1}(A^T y + \mu F^T g)$$

And in our case, $F = I, g = 0$. So the only thing that changes in our algorithm is we need a μI to be added to all of our rank one additions to A, hence the change of the initialization.

c) We observe that each log-absorption measurement is linear in the image as a vector input. So we let each row l of the data matrix A be a vector whose entries are one if and only if the pixel is affected:

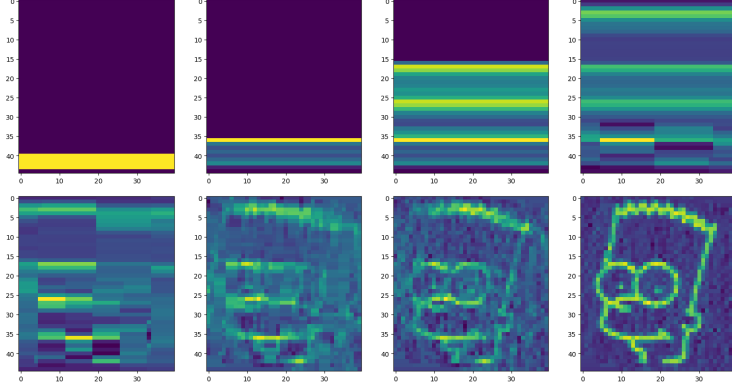$$|(j - w/2)\sin\theta_l - (i - h/2)\cos\theta_l - d_l| < 2$$

where we have unwound a single index k for the vectorized image into
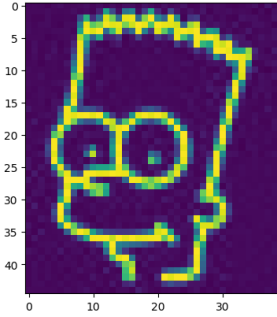
$$j = (k - 1)\%w + 1$$

$$i = (k - 1) \div w + 1$$

where the operations are division mod w and integer division.

Then, the solution after m examples is given by $\hat{X} = P(m)^{-1}q(m)$ using our algorithm from the previous part. The plots for the desired values of m follow, from left to right:


The final solution is:

2. **Magnetic resonance imaging.** In magnetic resonance imaging, the protons in hydrogen nuclei are excited by applying a magnetic field, and this excitation generates a radio frequency signal which is detected and used to make an image. Several magnetic fields are applied; a very strong uniform (or homogeneous) field, and additional weaker fields to cause atoms to resonate and provide spatial localization information.

The strong field must be uniform, *i.e.*, it must not vary with position. It may be as much as 7 T. For comparison, Earth's field is about 10^{-5} T. This question is about the design of such magnetic fields. The strength of the field means that the energy used is also a factor in the design.

We would like to design a magnet made of a coil of wire, which consists of multiple circular loops, arranged as shown in the figure below.
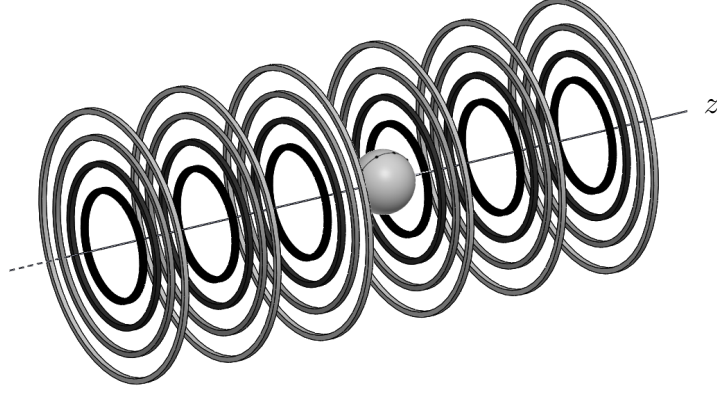
In order to make the magnetic field as uniform as possible, we will apply a different electric current to each loop. We have $n_z \times n_r$ loops. Since the loops have rotational symmetry about the z-axis, we need only look at the field within a slice. Such a slice is shown below. (In this figure, for illustration only, $n_z = 6$ and $n_r = 4$. In this problem we will use larger numbers.)
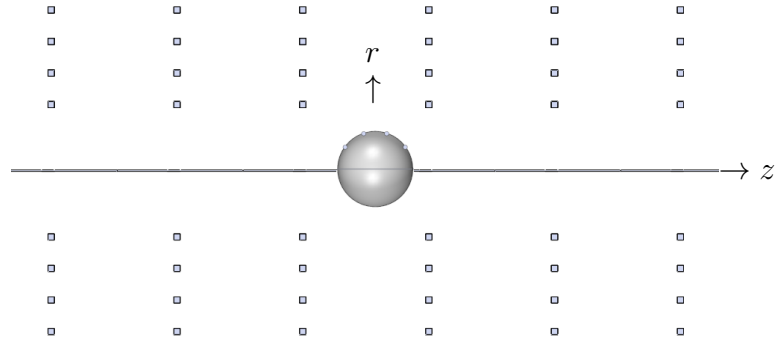
A point in the slice is described by r, the radial distance from the axis, and z, the axial distance from the origin.

The magnetic field is then a vector B, which varies with position. The radial component of B is small, and in this question we will neglect it. The magnetic field therefore always points axially. The magnetic field at a point (z, r) is specified by a scalar b.

For a single loop j ($j = 1, \dots, n_z n_r$) of wire of radius a_j, located in the axial direction at $\tilde{z}_j$, the axial component of the magnetic field $b_{z,r}^{(j)}$ at the point z, r is given by

$$b_{z,r}^{(j)} = \frac{x_j \mu_0}{2\pi \sqrt{q}} \left(\frac{(a_j^2 - r^2 - (z - \tilde{z}_j)^2) E(k)}{(a_j - r)^2 + (z - \tilde{z}_j)^2} + K(k) \right)$$

where $q = (a_j + r)^2 + (z - \tilde{z}_j)^2$. Here E and K are given by so-called elliptic integrals,

6

which are

$$K(k) = \int_0^{\pi/2} \frac{1}{\sqrt{1 - k^2 \sin^2 \theta}} \, d\theta \qquad E(k) = \int_0^{\pi/2} \sqrt{1 - k^2 \sin^2 \theta} \, d\theta$$

and $k = \sqrt{4a_j r/q}$. The elliptic integrals have no explicit algebraic expression, but may be evaluated numerically. In Julia, use

```
import Pkg
Pkg.add("Elliptic")
using Elliptic
K,E = ellipke(k*k)
```

In python, use

```
import scipy.special
K=scipy.special.ellipk(k*k)
E=scipy.special.ellipe(k*k)
```

In Matlab, use `[K,E] = ellipke(k*k)`. Note that all of these functions take k^2 as argument, not k. Here x_j is the current in the loop j, and μ_0 is the permeability constant. We will normalize and define $y_{z,r}^{(l)} = b_{z,r}^{(l)}/\mu_0$.

The magnetic field generated by a coil with multiple loops is the sum of the magnetic fields induced by each of the loops, $i.e.$, the total normalized magnetic field $y_{z,r}$ at location (z, r) is then defined as the sum of $y_{z,r}^{(j)}$ across all $n_z n_r$ loops. Let x_j be the current in loop j. We would like the resulting magnetic field to be uniform within a $target \ region$, which is the volume within which we want to image. The target region is the surface of a sphere of radius d. Since we have axial symmetry, we need only consider a target region which is a semicircle. We will divide it into m points (z_i, r_i) for $i = 1, \ldots, m$ where

$$z_i = d \cos \theta_i \qquad r_i = d \sin \theta_i$$

and $\theta_i = \pi(i - 1)/(m - 1)$. We will evaluate the magnetic field y_i at the point z_i, r_i for $i = 1, \ldots, m$.

The power P dissipated in the coil is proportional to the sum of the squares of the currents in the individual loops. We will normalize so that

$$P = \sum_{i=1}^{n_r n_z} x_i^2$$

The geometry of the coil is as follows. In a slice at fixed z, the coil has n_r equally spaced concentric loops of wire. The innermost loop has radius $r_{\min}$ and the outermost one has radius $r_{\max}$. Longitudinally, moving along the z axis, there are n_z equally spaced sets of n_r loops, with a total axial length of L, so each loop is distance $L/(n_z - 1)$ from it's

neighbor. So $\tilde{z}_j$ takes n_z values equally spaced between $-L/2$ and $L/2$, and a_j takes n_r values equally spaced between r_{min} and r_{max}.

For a typical medical MRI machine, we have geometry

$$r_{min} = 0.3 \qquad r_{max} = 0.4 \qquad L = 1.02$$

(with distances in meters) and we will design a coil with $n_z = 20$ and $n_r = 5$. We will control the field at $m = 1000$ target points, and have target of radius $d = 0.225$. The objective of this question is to choose how much current to apply to each coil, to achieve a magnetic field y which is as uniform as possible, and subject to a power dissipation limit $P < P_{max}$, where $P_{max} = 0.02$. (We have normalized both power and magnetic field units, but the other parameters are realistic.)

a) The relationship between the vector of target field strengths y and coil currents x is linear, and so $y = Ax$ for some matrix A. What are the dimensions of this matrix. Give an expression for A_{ij}.

b) We would like to achieve a uniform magnetic field. The quantity commonly used to describe this is the *field uniformity coefficient*, given by

$$F = \sum_i (y_i - b_0)^2$$

Here b_0 is the desired field strength; we will work with $b_0 = 1$. We would like F to be small, but there is a trade off between F and power dissipated P. Plot the optimal trade-off curve.

c) Find the coil currents x that minimize F subject to the requirement that $P < P_{max}$. Plot this as a n_z by n_r pixel image, with color proportional to absolute value of current.

d) For the choice of current in the previous part, plot the magnetic field y as a function of θ.

Solution.

a) For $i = 1, \ldots, m$,

$$\begin{aligned}
y_i &= \sum_{l=1}^{n_z n_r} y_{z_i r_i}^{(l)} \\
&= \sum_{l=1}^{n_z n_r} b_{z_i r_i}^{(l)} / \mu_0 \\
&= Ax
\end{aligned}$$

where the $A \in \mathbb{R}^{m \times n_z n_r}$. The expression for A_{ij} is

$$A_{ij} = \frac{1}{2\pi\sqrt{(a_j+r_i)^2+(z_i-\tilde{z}_j)^2}} \left(\frac{(a_j^2-r_i^2-(z_i-\tilde{z}_j)^2)E(k_{ij})}{(a_l-r_i)^2+(z_i-\tilde{z}_j)^2} + K(k_{ij}) \right)$$

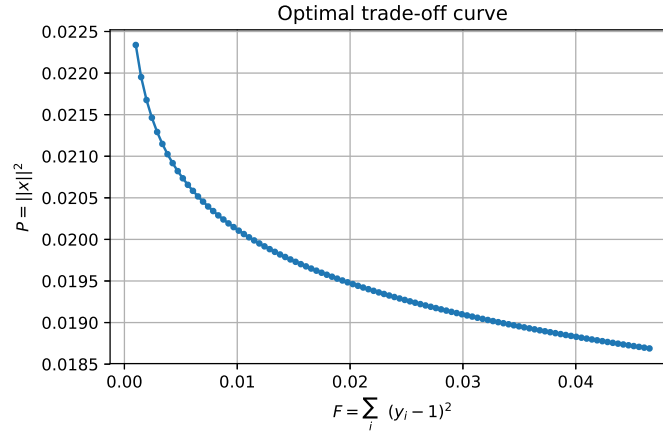where $k_{ij} = \sqrt{4a_j r_i / q_i}$.

b) The problem is

$$\text{minimize} \quad ||y - b_0||^2 + \lambda ||x||^2$$

where b_0 is a vector of ones and λ is the parameter giving relative weight between F and P. This is multi-objective least squares problem and we can derive the closed form solution as following:

$$||y - b_0||^2 + \lambda ||x||^2 = \left|\left| \begin{bmatrix} A \\ \sqrt{\lambda}I \end{bmatrix} x - \begin{bmatrix} b_0 \\ 0 \end{bmatrix} \right|\right|^2 = ||\tilde{A}x - \tilde{y}||^2$$

we have $x = (\tilde{A}^T \tilde{A})^{-1} \tilde{A}^T \tilde{y}$

c) From previous part, we find the minimum F that meets the constraint $P < P_{\text{max}}$. Then, we reshape it and plot as n_z by n_r pixel image. The choice $\lambda = 12.38$ satisfies this.
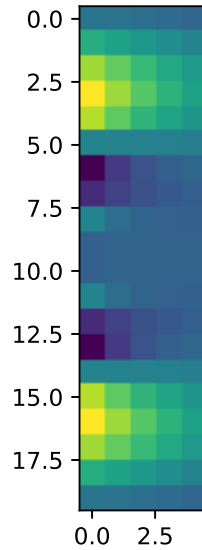

d) The plot is

3. Left inverses (or not). Consider the following matrix:

$$A = \begin{bmatrix} 6 & 1 & 5 \\ 7 & 1 & 3 \\ 7 & 5 & 9 \\ 4 & 8 & 9 \\ 4 & 1 & 3 \end{bmatrix}$$

Recall that a matrix B is a *left inverse* of A if $BA = I$. For each of the following questions, explicitly find a matrix B that fits the criterion. If it is not possible, say so and explain why. If it is possible, turn in the matrix and the code you wrote verifying your that your matrix fits the criterion (print the code's results out to the terminal). Explain how you reached your answer.

For example, if you were asked "Find x such that $x + 5 > 3$", then the below would be a proper terminal output:

Plot y vs θ
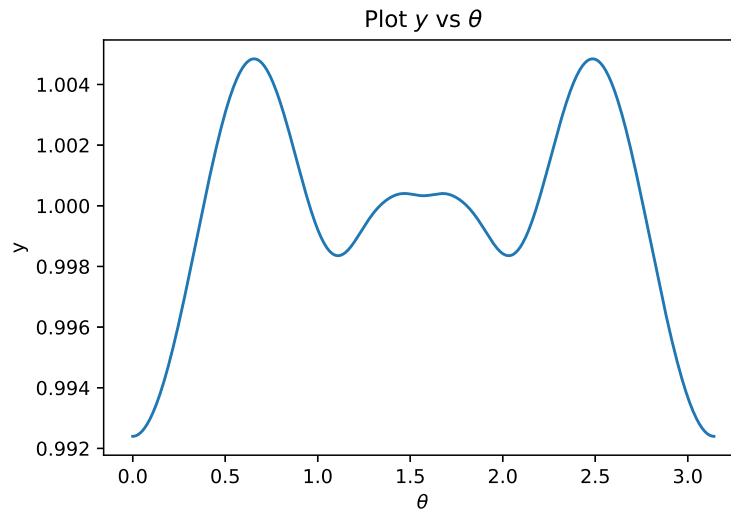

```
               _
       _  _ _(_)_     |  Documentation: https://docs.julialang.org
      (_)     | (_) (_)    |
       _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
      | | | | | | |/ _` |  |
      | | |_| | | | (_| |  |  Version 1.0.3 (2018-12-18)
     _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
    |__/                   |

[julia> x = 0
0

[julia> x + 5 > 3
 true

julia> █
```

a) Find B such that $BA = I$.

b) Find B such that $BA = I$ and B is lower-triangular.

c) Find B such that $BA = I$ and $B_{11} = 0$.

d) Find B such that $BA = I$ and B is rank 2.

e) Find B such that

$$B = \operatorname*{argmin}_{X} \|X\|_F \text{ subject to } XA = I.$$

Here $\|X\|_F$ is the Frobenius norm, defined for $X \in \mathbb{R}^{m \times n}$ by

$$\|X\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} X_{ij}^2 \right)$$

11

f) Find B such that $BA = I$ and all other left inverses of A have smaller norm.

Solution. We can split this problem up into three sub-blocks of linear equations. Let

$$B = \begin{bmatrix} B_1^T \\ B_2^T \\ B_3^T \end{bmatrix}.$$

Then we can rewrite $BA = I$ as the following three systems of equations

$$A^T B_1 = (1,0,0), \quad A^T B_2 = (0,1,0), \quad A^T B_3 = (0,0,1).$$

This will be useful for parts b and c.

a) $B = (A^T A)^{-1} A^T$ is a immediate solution. For this case, we have that $(A^T A)^{-1} A^T$ is equal to

```
[-0.07078492   0.24875504 -0.07280057   0.01968224   0.02857482]
[-0.27620346   0.28094617 -0.15467157   0.2250415   -0.03171686]
[ 0.25278634  -0.35736305  0.19800806  -0.11216505   0.01185677]
```

b) Impossible. We can immediately see from $A^T B_1 = (1,0,0)$ that we'd have $6(B_1)_1 = 1$ and $(B_1)_1 = 0$, since $(B_1)_2 = (B_1)_3 = (B_1)_4 = (B_1)_5 = 0$ from the definition of a lower-triangular matrix.

c) We can solve this by letting B_2 and B_3 being the same B_2 and B_3 in part a, and letting $B_1 = (0, \tilde{b})$, with $\tilde{b}$ being the solution of

$$(A_{2:5,1:5})^T \tilde{b} = (1,0,0).$$

There are many other methods that are valid. For completeness, here's another valid answer.

```
B= [0   2.59181292e-01 -1.23586736e-01   4.94346943e-02 -3.67251674e-02]
  [-3.29844088e-01   2.89696356e-01 -8.84339885e-02   1.91394384e-01 -4.88373955e-02
  [ 3.08697561e-01 -3.60421477e-01   1.39070257e-01 -8.11350947e-02 5.45338910e-03]
```

d) Impossible. First, we have that $\text{rank}(A) = 3$. (The student should verify or at least note that this is true.)

If $\text{rank}(B) = 2$, then the mapping $B \to BA$ would produce a rank 2 matrix. Regardless of what this matrix is, it is never I, which is trivially rank 3.

e) The solution to this part is simply $(A^T A)^{-1} A^T$. This problem was to test the student to see if they understood what "least-norm" means.

f) Impossible. We are essentially solving the optimization problem

$$\max_{X} \|X\|_F \text{ subject to } XA = I.$$

The set $\{X \mid XA = I\}$ is a hyperplane, so there exist infinitely many X that satisfy $XA = I$. Additionally since $A \neq \mathbf{0}$, X can become arbitrarily large. Therefore, the solution to this optimization problem is unbounded, and there does not exist an a B such that $BA = I$ and all other left inverses of A have smaller norm.

(We note that no knowledge of advanced concepts in optimization, *i.e.*, concepts not introduced in this course, were necessary to solve this part.)

4. **Solving optimization problems via least squares.** In this problem, we explore an approach for solving unconstrained smooth optimization problems using linearization and least squares. Formally, consider a continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, we want to find $w \in \mathbb{R}^n$ to minimize $f(w)$ over $w \in \mathbb{R}^n$.

   a) Suppose that $f(w)$ is quadratic, *i.e.*, $f(w) = \frac{1}{2}w^T H w + g^T w$, for some matrix $H \in \mathbb{R}^{n \times n}$ and vector $g \in \mathbb{R}^n$. Given m vectors $w_1, w_2, \ldots, w_m \in \mathbb{R}^n$, consider their linear combination $\tilde{w} = \sum_{i=1}^{m} \alpha_i w_i$, where $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$ is the coefficients of the linear combination.

      i. Find an expression for $\nabla f(\tilde{w})$ (*i.e.*, the gradient of f evaluated at $\tilde{w}$) using H, g, w_i $(i = 1, \ldots, m)$ and α.

      ii. Assuming that $g \neq 0$, find a necessary and sufficient condition on α such that

      $$\nabla f(\tilde{w}) = \sum_{i=1}^{m} \alpha_i \nabla f(w_i)$$

      iii. The gradient descent algorithm updates w to $w' = F(w)$, where the mapping $F : \mathbb{R}^n \to \mathbb{R}^n$ is defined as $F(w) = w - \eta \nabla f(w)$, where $\eta > 0$ is a parameter. Show that under the condition you found in problem 2-a)ii above, for the quadratic f in this question

      $$F(\tilde{w}) = \sum_{i=1}^{m} \alpha_i F(w_i)$$

   b) Here we introduce Anderson mixing for finding the minimum of $f(w)$ over w. Anderson mixing is an iterative algorithm, which attempts to find a minimum of f by minimizes the norm of its gradient. We no longer assume f is quadratic.

      Specifically, we start with m random points $w_1, w_2, \ldots, w_m \in \mathbb{R}^n$ and view them as the first m iterations. Then at iteration $k \geq m$, given m previous iterates

$w_{k-m+1}, w_{k-m+2}, \ldots, w_k$, we define

$$w_{k+1} = \tilde{w} = \sum_{i=1}^{m} \alpha_i \left(w_{k-m+i} - \eta \nabla f(w_{k-m+i}) \right),$$

where $\alpha \in \mathbb{R}^m$ is chosen to minimize the Euclidean norm of $\sum_{i=1}^{m} \alpha_i \nabla f(w_{k-m+i})$, (which would equal $\nabla f(w_{k+1})$ if f were quadratic), subject to the condition you found in problem 2-a)ii above.

Notice that when f is quadratic, w_{k+1} can be equivalently obtained by applying the gradient descent map F in problem 2-a)iii above to $\tilde{w}_{k+1} = \sum_{i=1}^{m} \alpha_i w_{k-m+i}$.

Suppose that $m \leq n$, find an explicit expression for the solution α to the k-th iteration subproblem described above. In your solution no minimization/maximization can appear, but you can use linear algebra operations like matrix inverses and multiplication. (*Hint*: Feel free to assume that certain the matrices appearing in your solution are full-rank.)

c) This approach is used in machine learning to minimize the cost function when solving certain regression problems. We'll look here at a very simple example, with just one data point(!), just to verify that the method works. The regression problem we would like to solve is to minimize

$$f(w) = -y \log \sigma(w^T x) - (1 - y) \log(\sigma(-w^T x)) + \frac{\lambda}{2} \|w\|^2$$

over $w \in \mathbb{R}^n$. Here $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$ are given, and are not variables. The function $\sigma : \mathbb{R} \to \mathbb{R}$ is the sigmoid, defined to be

$$\sigma(s) = 1/(1 + e^{-s})$$

Then the gradient of f is

$$\nabla f(w) = (\sigma(w^T x) - y)x + \lambda w$$

Let $m = 3$ and $n = 6$. Run the iteration procedure described in 2-b) above for $k = 100$ iterations (*i.e.*, the final output would be w_{101}). The initialization of w_1, w_2, w_3, the data x and y, as well as the hyper-parameters λ and η are below. Plot $f(w_k)$ against k for $k = 4, \ldots, 101$, and report the smallest value of f that you found (up to 3 digits after the decimal point). We have $y = 1$, $\lambda = 0.01$, $\eta = 0.1$, and

$$w_1 = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \end{bmatrix} \quad w_2 = \begin{bmatrix} 0.2 \\ 0.1 \\ 0.3 \\ 0.1 \\ 0.1 \\ 0.01 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.2 \\ 0.5 \\ 0.1 \\ 0.3 \\ 0.2 \\ -0.7 \end{bmatrix} \quad x = \begin{bmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \end{bmatrix}$$

14

Solution.

a)   i. We have $\nabla f(w) = \frac{1}{2}(H + H^T)w + g$, and hence $\nabla f(\tilde{w}) = \frac{1}{2}(H + H^T)\tilde{w} + g = \frac{1}{2}(H + H^T)\sum_{i=1}^{m} \alpha_i w_i + g = \frac{1}{2}(H + H^T)W\alpha + g$, where $W = (w_1, \ldots, w_m)$ is an n-by-m matrix.

     ii. We have $\nabla f(\tilde{w}) = \frac{1}{2}(H + H^T)W\alpha + g = \sum_{i=1}^{m} \alpha_i \nabla f(w_i) = \sum_{i=1}^{m} \alpha_i(\frac{1}{2}(H + H^T)w_i + g) = \frac{1}{2}(H + H^T)W\alpha + (\sum_{i=1}^{m} \alpha_i)\, g$ if and only if $\sum_{i=1}^{m} \alpha_i = 1$ given that $g \neq 0$.

     iii. By problem 2-a)ii above, we have

$$F(\tilde{w}) = \tilde{w} - \eta \nabla f(\tilde{w}) = \sum_{i=1}^{m} \alpha_i w_i - \eta \sum_{i=1}^{m} \alpha_i \nabla f(w_i)$$
$$= \sum_{i=1}^{m} \alpha_i(w_i - \eta \nabla f(w_i)) = \sum_{i=1}^{m} \alpha_i F(w_i).$$

b) We formulate the problem firstly as a constrained least-squares problem:

$$
\begin{array}{ll}
\text{minimize} & \|G\alpha\|_2 \\
\text{subject to} & \sum_{i=1}^{m} \alpha_i = 1,
\end{array}
\tag{2}
$$

where $G = (\nabla f(w_{k-m+1}), \ldots, \nabla f(w_k))$ is a matrix in $\mathbb{R}^{n \times m}$. Now we rewrite the above problem as an unconstrained least-squares problem in variable $\gamma \in \mathbb{R}^{m-1}$:

$$\text{minimize} \quad \|Y\gamma - g\|_2, \tag{3}$$

where $g = \nabla f(w_k)$, $Y = (y_{k-m+1}, \ldots, y_{k-1})$, with $y_j = \nabla f(w_{j+1}) - \nabla f(w_j)$, and γ is connected to α via

$$\alpha_1 = \gamma_1, \ \alpha_i = \gamma_i - \gamma_{i-1}, \ i = 2, \ldots, m-1, \alpha_m = 1 - \gamma_{m-1}. \tag{4}$$

Since we have assumed that $n \geq m$, assuming that Y has full column rank, then the least squares solution to γ is $\gamma = (Y^T Y)^{-1} Y^T g$, and then α is obtained from γ via (4).

c) The plot is shown in Figure 1. The objective value found is 0.206 (which also coincides with the CVXPY solution, btw).

5. **Signal reconstruction from cosine basis functions.** Let $f(t)$ be a continuous-time signal on $[0, 1]$. Our model for $f(t)$ is that it lies in the span of certain cosine basis functions, *i.e.*,

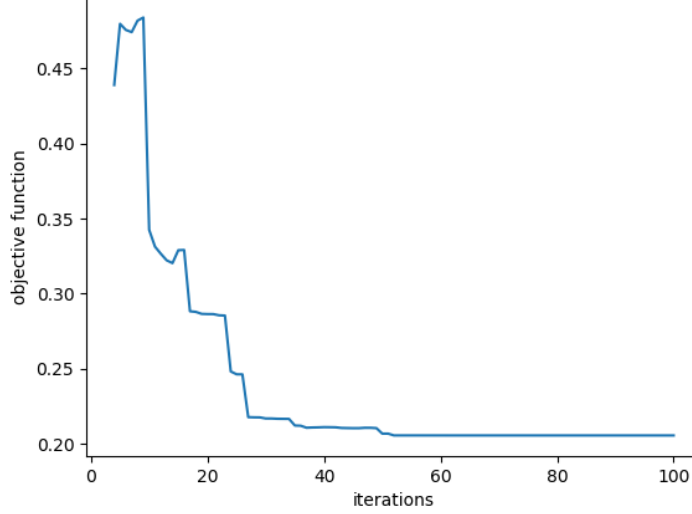$$f(t) = \sum_{i=0}^{N-1} x_i \psi_i(t),$$

Figure 1: Convergence of the regularized logistic objective

where $x \in \mathbb{R}^N$ and

$$\psi_i(t) = \begin{cases} 1 & i = 0 \\ \sqrt{2}\cos(\pi i t) & i = 1, 2, 3, \ldots. \end{cases}$$

Given M arbitrary points $t_1, \ldots, t_M$ on $[0, 1]$, we make $2M$ measurements. The first M measurements are the derivative of $f(t)$ at the t_m:

$$y_m = f'(t_m), \quad m = 1, \ldots, M.$$

The second M measurements are the integral of $f(t)$ up to time t_m:

$$y_m = \int_0^{t_{m-M}} f(q)dq, \quad m = M + 1, \ldots, 2M.$$

a) Show that it is possible to write $y \in \mathbb{R}^{2M}$ as a system of linear equations in $x \in \mathbb{R}^N$ (i.e., $y = Ax$ for some A). Find A explicitly for the particular case of $M = 3$ and $N = 4$.

b) You have been provided with 900 values in `cosinefit.json` corresponding to $t_1, \ldots, t_{300}$,
$f'(t_1), \ldots, f'(t_{300})$, and $\int_0^{t_1} f(q)dq, \ldots, \int_0^{t_{300}} f(q)dq$. Given that $f(t)$ lies in the span of the first 10 cosine basis functions, find $f(t)$. For your answer, turn in a plot of $f(t)$ in $[0, 1]$, and report the coefficients (i.e., $x \in \mathbb{R}^{10}$) to the nearest 0.001.

16

Solution.

a) We have

$$f'(t) = \sum_{i=0}^{N-1} x_i \psi_i'(t)$$

$$= (\psi_0'(t), \psi_1'(t), \ldots, \psi_{N-1}'(t))^T x$$

and

$$\int_0^t f(q)dq = \int_0^t \sum_{i=0}^{N-1} x_i \psi_i(q)dq$$

$$= \sum_{i=0}^{N-1} x_i \int_0^t \psi_i(q)dq$$

$$= \left(\int_0^t \psi_0(q)dq, \int_0^t \psi_1(q)dq, \ldots, \int_0^t \psi_{N-1}(q)dq \right)^T x.$$

Computing $\psi_i'(t_m)$ and $\int_0^{t_m} \psi_i(q)dq$ for $i = 0, \ldots, N-1$ and $m = 1, \ldots, M$, and stacking the results yields us

$$A = \begin{bmatrix} 0 & -\sqrt{2}\pi \sin(\pi t_1) & -2\sqrt{2}\pi \sin(2\pi t_1) & -3\sqrt{2}\pi \sin(3\pi t_1) \\ 0 & -\sqrt{2}\pi \sin(\pi t_2) & -2\sqrt{2}\pi \sin(2\pi t_2) & -3\sqrt{2}\pi \sin(3\pi t_2) \\ 0 & -\sqrt{2}\pi \sin(\pi t_3) & -2\sqrt{2}\pi \sin(2\pi t_3) & -3\sqrt{2}\pi \sin(3\pi t_3) \\ t_1 & (\sqrt{2}/\pi) \sin(\pi t_1) & (\sqrt{2}/2\pi) \sin(2\pi t_1) & (\sqrt{2}/3\pi) \sin(3\pi t_1) \\ t_2 & (\sqrt{2}/\pi) \sin(\pi t_2) & (\sqrt{2}/2\pi) \sin(2\pi t_2) & (\sqrt{2}/3\pi) \sin(3\pi t_2) \\ t_3 & (\sqrt{2}/\pi) \sin(\pi t_3) & (\sqrt{2}/2\pi) \sin(2\pi t_3) & (\sqrt{2}/3\pi) \sin(3\pi t_3) \end{bmatrix}.$$

b) For this part, we simply use the results of part a and expand it to $M = 600$ and $N = 10$.
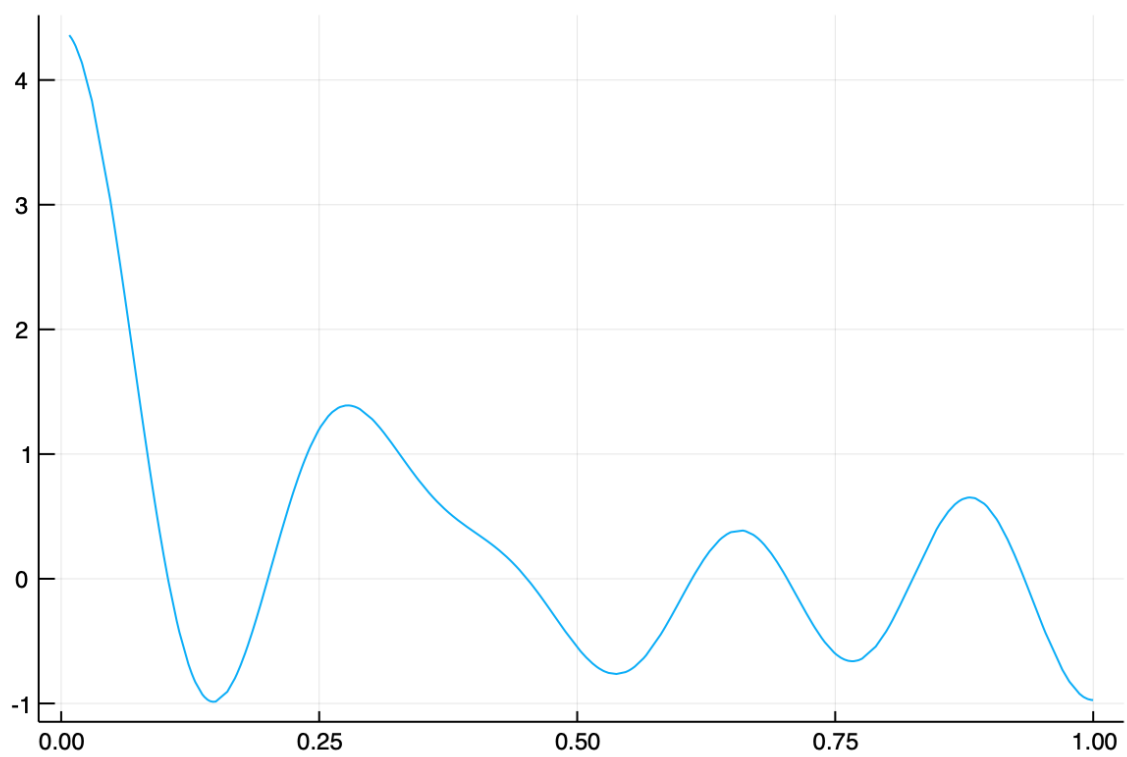
$$x = (0.3, 0.5, 0.3, 0.1, 0.1, 0.4, 0.5, 0.5, 0.1, 0.4).$$

Figure 2: Plot of $f(t)$ with $0 \leq t \leq 1$.