CMDA 3606 · MATHEMATICAL MODELING II

Problem Set 9

Posted 12 April 2018. Due at 5pm on Thursday, 18 April 2019.

Basic guidelines: Students may discuss the problems on this assignment, but each student must submit his or her individual writeup and code. (In particular, you *must write up your own individual MATLAB code.*) Students may consult class notes and other online resources for general information; cite all your sources and list those with whom you have discussed the problems.

1. [25 points: 5 points per part]

   At some point in your past, you learned that the dot product describes the *angle between two vectors*: For nonzero $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$,

   $$\mathbf{x}^T\mathbf{y} = \cos\angle(\mathbf{x},\mathbf{y})\|\mathbf{x}\|\|\mathbf{y}\|,$$

   which can be regrouped as

   $$\cos\angle(\mathbf{x},\mathbf{y}) = \left(\frac{\mathbf{x}}{\|\mathbf{x}\|}\right)^T\left(\frac{\mathbf{y}}{\|\mathbf{y}\|}\right). \qquad (*)$$

   Here we explore how to generalize this idea to understand the *angles between two subspaces.*

   (a) Consider the two 2-dimensional subspaces

   $$\mathcal{X} = \text{span}\left\{\begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}\right\}, \qquad \mathcal{Y} = \text{span}\left\{\begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix}\right\}.$$

   Each of these subspaces can be visualized as a plane in $\mathbb{R}^3$. What is their intersection? Describe, in geometrical language (e.g., in terms of the "x, y, and z axes"), how these planes are arranged relative to one another.

   We define the angles between two d-dimensional subspaces $\mathcal{X}$ and $\mathcal{Y}$ of $\mathbb{R}^n$ as follows:

   - Construct a matrix $\mathbf{Q}_\mathcal{X} \in \mathbb{R}^{n\times d}$ whose columns form an orthonormal basis for $\mathcal{X}$;
   - Construct a matrix $\mathbf{Q}_\mathcal{Y} \in \mathbb{R}^{n\times d}$ whose columns form an orthonormal basis for $\mathcal{Y}$;
   - Form the matrix $\mathbf{A} = \mathbf{Q}_\mathcal{X}^T\mathbf{Q}_\mathcal{Y}$;
   - Compute the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d$ of $\mathbf{A}$.
   - We define the jth smallest angle $\theta_j(\mathcal{X},\mathcal{Y})$ between $\mathcal{X}$ and $\mathcal{Y}$ to be

   $$\cos\theta_j(\mathcal{X},\mathcal{Y}) = \sigma_j.$$

   (b) Explain how this formula reduces to $(*)$ for the case $d = 1$.

   (c) Compute $\theta_1(\mathcal{X},\mathcal{Y})$ and $\theta_2(\mathcal{X},\mathcal{Y})$ for the subspaces $\mathcal{X}$ and $\mathcal{Y}$ in part (a).
   Does this answer make sense, compared to your geometric description in part (a)?

   (d) Compute $\theta_1(\mathcal{X},\mathcal{Z})$ and $\theta_2(\mathcal{X},\mathcal{Z})$, where $\mathcal{X}$ is again from part (a) and

   $$\mathcal{Z} = \text{span}\left\{\begin{bmatrix}0\\\sqrt{2}/2\\\sqrt{2}/2\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix}\right\}.$$

   (e) Now suppose $\mathcal{X}$ and $\mathcal{Y}$ are general d-dimensional subspaces, and let $\mathbf{u}_j$ and $\mathbf{v}_j$ denote the jth left and right singular vectors of $\mathbf{A}$. Using the definition $(*)$, what is the angle between the vectors $\mathbf{Q}_\mathcal{X}\mathbf{u}_j$ (which is in $\mathcal{X}$) and $\mathbf{Q}_\mathcal{Y}\mathbf{v}_j$ (which is in $\mathcal{Y}$)?

   (f) Suppose the first p singular values of $\mathbf{Q}_\mathcal{X}^T\mathbf{Q}_\mathcal{Y}$ are equal to one, $\sigma_1 = \cdots = \sigma_p = 1$.
   Describe the vectors at the intersection of $\mathcal{X}$ and $\mathcal{Y}$.

2. [38 points: 7 points each for (a),(b),(d),(e); 10 point each for (c)]

At its kth step, an iterative method for solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ constructs an approximate solution $\mathbf{x}_k$. With this iterate we associate the *residual* vector

$$\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k.$$

A successful method for solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ will drive $\mathbf{r}_k \to \mathbf{0}$ as quickly as possible.

Lewis Fry Richardson proposed that $\mathbf{x}_k$ be constructed as follows Let $\mathbf{x}_0 = \mathbf{0}$ so that $\mathbf{r}_0 = \mathbf{b}$. Then for a fixed constant c, define

$$\mathbf{x}_{k+1} = \mathbf{x}_k + c\mathbf{r}_k.$$

(a) Show that you can write $\mathbf{r}_k = (\mathbf{I} - c\mathbf{A})^k\mathbf{b}$.

(b) Suppose that $\mathbf{A}$ is diagonalizable, $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} = \sum_{j=1}^{n} \lambda_j \mathbf{v}_j \widehat{\mathbf{v}}_j^*$.
Show that $\mathbf{I} - c\mathbf{A}$ has the same eigenvectors as $\mathbf{A}$. What are the eigenvalues of $\mathbf{I} - c\mathbf{A}$?

(c) Suppose further that all eigenvalues λ of $\mathbf{A}$ satisfy $1 \le \lambda \le 2$.
To ensure that $\mathbf{r}_k \to \mathbf{0}$, all eigenvalues of $\mathbf{I} - c\mathbf{A}$ must have absolute value less than one.
For what real numbers c will $\mathbf{r}_k \to \mathbf{0}$?
What optimal choice of c will drive $\mathbf{r}_k \to \mathbf{0}$ most rapidly (making the largest absolute value of the eigenvalues of $\mathbf{I} - c\mathbf{A}$ as small as possible, knowing only that $1 \le \lambda \le 2$)?

For the rest of the problem, consider this 500×500 matrix $\mathbf{A}$ and accompanying right-hand side:

$$\mathbf{A} = \frac{1}{4}\begin{bmatrix} 6 & 1 & & \\ 1 & 6 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 6 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix},$$

where the unspecified entries of $\mathbf{A}$ are zero. The eigenvalues of this matrix satisfy $1 \le \lambda \le 2$. (You do not need to prove this, but you could do so by recognizing this matrix as a modification of one you studied on Problem Set 5.)

(d) Produce a `semilogy` plot showing $\|\mathbf{r}_k\|$ (vertical axis) versus k (horizontal axis) for the optimal c you determined in part (c).

(e) The polynomial $\psi_k(z) = (1 - cz)^k$ should be small on the interval $1 \le z \le 2$, and satisfy $\psi_k(0) = 1$. Confirm these properties by producing a plot showing $\psi_k(z)$ (vertical axis) versus z (horizontal axis) for $-1 \le z \le 3$ for $k = 1, 2, 3, 4, 5$. (Show all five graphs on the same plot.) Use `axis([-3 3 -.5 1.5])` to crop the axes to the most relevant region.

3. [37 points: 7 points each for (a),(b),(c); 16 points for (d); +5 bonus for (e)]

The barcode example on Problem Set 8 studied image blurring in one dimension. On this problem, we explore the two-dimensional version of this problem. (To be sure, the approach we set up here is rather idealistic; indeed books are written about image deblurring. This problem is meant to give you a taste for a much larger area, and to help you see how large the problems quickly become.)

As discussed in class, we regard a square image as a function of two variables, $f(t^{(1)}, t^{(2)})$, where $t^{(1)}, t^{(2)} \in [0, 1]$: the value of $f(t^{(1)}, t^{(2)})$ represents the intensity of gray value at the vertical point $t^{(1)}$ and the horizontal point $t^{(2)}$ in the image.

When we acquire the image through a camera, the device effectively blurs the true image, which is modeled by a *convolution*. The $(s^{(1)}, s^{(2)})$ point in our blurred image b is given by

$$b(s^{(1)}, s^{(2)}) = \int_0^1 \int_0^1 K\left(\begin{bmatrix} s^{(1)} \\ s^{(2)} \end{bmatrix}, \begin{bmatrix} t^{(1)} \\ t^{(2)} \end{bmatrix}\right) f(t^{(1)}, t^{(2)})\, \mathrm{d}t^{(1)}\, \mathrm{d}t^{(2)}.$$

Of course, you know that digital images are not functions of continuous variables, but rather are stored as discrete $n \times n$ arrays of pixels, which we regard as a matrix: $f_{j,k} = f(t_j, t_k)$ is the value of the (j,k) pixel, corresponding to

$$t_j = \frac{j - 1/2}{n}, \qquad t_k = \frac{k - 1/2}{n}.$$

For a grayscale image, $f_{j,k}$ will be an integer between 0 (black) and 255 (white).

Given that we only know $f(t^{(1)}, t^{(2)})$ at discrete values of $t^{(1)}$ and $t^{(2)}$, we naturally discretize the double integral for $b(s^{(1)}, s^{(2)})$ using two applications of the same midpoint rule we used for the barcode example in Problem Set 8:

$$b_{\ell,m} = \frac{1}{n^2} \sum_{j=1}^{n} \sum_{k=1}^{n} K\left(\begin{bmatrix} s_\ell \\ s_m \end{bmatrix}, \begin{bmatrix} t_j \\ t_k \end{bmatrix} \right) f_{j,k}$$

for

$$s_j = \frac{j - 1/2}{n}, \qquad s_k = \frac{k - 1/2}{n}.$$

To do linear algebra, we arrange this in matrix form. For example, for $n = 3$ we have

$$\left[\begin{array}{c} b_{1,1} \\ b_{2,1} \\ b_{3,1} \\ \hline b_{1,2} \\ b_{2,2} \\ b_{3,2} \\ \hline b_{1,3} \\ b_{2,3} \\ b_{3,3} \end{array}\right] = \left[\begin{array}{ccc|ccc|ccc} K_{1,1,1,1} & K_{2,1,1,1} & K_{3,1,1,1} & K_{1,2,1,1} & K_{2,2,1,1} & K_{3,2,1,1} & K_{1,3,1,1} & K_{2,3,1,1} & K_{3,3,1,1} \\ K_{1,1,2,1} & K_{2,1,2,1} & K_{3,1,2,1} & K_{1,2,2,1} & K_{2,2,2,1} & K_{3,2,2,1} & K_{1,3,2,1} & K_{2,3,2,1} & K_{3,3,2,1} \\ K_{1,1,3,1} & K_{2,1,3,1} & K_{3,1,3,1} & K_{1,2,3,1} & K_{2,2,3,1} & K_{3,2,3,1} & K_{1,3,3,1} & K_{2,3,3,1} & K_{3,3,3,1} \\ \hline K_{1,1,1,2} & K_{2,1,1,2} & K_{3,1,1,2} & K_{1,2,1,2} & K_{2,2,1,2} & K_{3,2,1,2} & K_{1,3,1,2} & K_{2,3,1,2} & K_{3,3,1,2} \\ K_{1,1,2,2} & K_{2,1,2,2} & K_{3,1,2,2} & K_{1,2,2,2} & K_{2,2,2,2} & K_{3,2,2,2} & K_{1,3,2,2} & K_{2,3,2,2} & K_{3,3,2,2} \\ K_{1,1,3,2} & K_{2,1,3,2} & K_{3,1,3,2} & K_{1,2,3,2} & K_{2,2,3,2} & K_{3,2,3,2} & K_{1,3,3,2} & K_{2,3,3,2} & K_{3,3,3,2} \\ \hline K_{1,1,1,3} & K_{2,1,1,3} & K_{3,1,1,3} & K_{1,2,1,3} & K_{2,2,1,3} & K_{3,2,1,3} & K_{1,3,1,3} & K_{2,3,1,3} & K_{3,3,1,3} \\ K_{1,1,2,3} & K_{2,1,2,3} & K_{3,1,2,3} & K_{1,2,2,3} & K_{2,2,2,3} & K_{3,2,2,3} & K_{1,3,2,3} & K_{2,3,2,3} & K_{3,3,2,3} \\ K_{1,1,3,3} & K_{2,1,3,3} & K_{3,1,3,3} & K_{1,2,3,3} & K_{2,2,3,3} & K_{3,2,3,3} & K_{1,3,3,3} & K_{2,3,3,3} & K_{3,3,3,3} \end{array}\right] \left[\begin{array}{c} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ \hline f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ \hline f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{array}\right],$$

where

$$K_{j,k,\ell,m} = \frac{1}{n^2} K\left(\begin{bmatrix} s_\ell \\ s_m \end{bmatrix}, \begin{bmatrix} t_j \\ t_k \end{bmatrix} \right).$$

We can abbreviate this as $\mathbf{b} = \mathbf{K}\mathbf{f}$.

Let us discuss how to work with images in MATLAB. Suppose we have an $n \times n$ matrix stored in `F`. You can visualize this image via:

```
imagesc(F)
colormap gray
axis equal, axis([1 n 1 n])
```

To go from an image matrix `F` $\in \mathbb{R}^{n \times n}$ to the vectorized version `f` $\in \mathbb{R}^{n^2 \times 1}$, use:

```
f = reshape(F,n*n,1);
```

Now you can operate on `f` now using matrix computations. To apply the blurred version `b`:

```
b = K * f;
```

Use `reshape` again to transform `b` $\in \mathbb{R}^{n^2 \times 1}$ into an image `B` $\in \mathbb{R}^{n \times n}$:

```
B = reshape(b,n,n);
```

This MATLAB code runs the forward problem: if we know `F`, this would blur the image to get `B`. We are faced with the much harder *inverse problem*: our hardware acquires the blurred image `B` (potentially polluted with noise), and we want to find the *unblurred* image `F`.

To help with this problem, download three files from the class website: `blur2d.m`, `hokiebird.mat`, `mystery_plate.mat`. The first of these functions creates the matrix `K` for the Gaussian blurring kernel

$$K_{j,k,\ell,m} = \frac{1}{n^2} \frac{\mathrm{e}^{-\left((s_\ell - t_j)^2 + (s_m - t_k)^2\right)/z}}{\pi z}$$

(with the tails truncated to add zeros to the blurring matrix). The examples below use $n = 60$, so $K \in \mathbb{R}^{3600\times 3600}$. Your computations will likely be a bit slow.

(a) Set $n = 60$ and $z = 0.01$, and generate `K = blur2d(60,0.01);`
Produce a `semilogy` plot of the singular values of `K`.

(b) Run `load hokiebird.mat`, which contains the original image `F` and a noisy, blurred image. Use the `imagesc` command, as illustrated above, to visualize both `F` and `B`.

(c) Use `frec = K\b` to attempt to recover the original image from the noisy, blurred version `b = reshape(B,n*n,1)`. Produce an `imagesc` plot of the recovered version `Frec = reshape(frec,n,n)`.

(d) To get a better result, apply Tikhonov regularization (or the truncated SVD) to the blurred vector `b = reshape(B,n*n,1)`. Do your best to recover something like the original image **F**. Describe your experiments and include a plot or two. (You do not need to produce an L-curve: each experiment will likely take a few seconds of computation.)

(e) +5 bonus points: A mysterious vehicle has spotted in the vicinity of the CMDA Collaboration Center, but we only have a blurry image of the license plate: `load mystery_plate.mat` will load the image into the variable `B`. This image was acquired using the same blur matrix in (a), plus some pollution from noise. Can you use regularization to recover the correct license plate?