

EE263 Homework 7  
Fall 2025  
due Thursday 11/13, at 11:59 PM

**15.2290. Matrix gain compared with entry size.** A matrix can have all entries large and yet have small gain in some directions, that is, it can have a small  $\sigma_{\min}$ . For example,

$$A = \begin{bmatrix} 10^6 & 10^6 \\ 10^6 & 10^6 \end{bmatrix}$$

has “large” entries while  $\|A[1 \ -1]^T\| = 0$ . Can a matrix have all entries small and yet have a large gain in some direction, that is, a large  $\sigma_{\max}$ ? Suppose, for example, that  $|A_{ij}| \leq \epsilon$  for  $1 \leq i, j \leq n$ . What can you say about  $\sigma_{\max}(A)$ ?

**15.2300. Frobenius norm of a matrix.** The Frobenius norm of a matrix  $A \in \mathbb{R}^{n \times n}$  is defined as  $\|A\|_F = \sqrt{\text{trace } A^T A}$ . (Recall trace is the trace of a matrix, *i.e.*, the sum of the diagonal entries.)

a) Show that

$$\|A\|_F = \left( \sum_{i,j} |A_{ij}|^2 \right)^{1/2}.$$

Thus the Frobenius norm is simply the Euclidean norm of the matrix when it is considered as an element of  $\mathbb{R}^{n^2}$ . Note also that it is much easier to compute the Frobenius norm of a matrix than the (spectral) norm (*i.e.*, maximum singular value).

b) Show that if  $U$  and  $V$  are orthogonal, then  $\|UA\|_F = \|AV\|_F = \|A\|_F$ . Thus the Frobenius norm is not changed by a pre- or post- orthogonal transformation.

c) Show that  $\|A\|_F = \sqrt{\sigma_1^2 + \dots + \sigma_r^2}$ , where  $\sigma_1, \dots, \sigma_r$  are the singular values of  $A$ . Then show that  $\sigma_{\max}(A) \leq \|A\|_F \leq \sqrt{r} \sigma_{\max}(A)$ . In particular,  $\|Ax\| \leq \|A\|_F \|x\|$  for all  $x$ .

**15.2350. Two representations of an ellipsoid.** In the lectures, we saw two different ways of representing an ellipsoid, centered at 0, with non-zero volume. The first uses a quadratic form:

$$\mathcal{E}_1 = \left\{ x \mid x^T S x \leq 1 \right\},$$

with  $S^T = S > 0$ . The second is as the image of a unit ball under a linear mapping:

$$\mathcal{E}_2 = \{ y \mid y = Ax, \|x\| \leq 1 \},$$

with  $\det(A) \neq 0$ .

a) Given  $S$ , explain how to find an  $A$  so that  $\mathcal{E}_1 = \mathcal{E}_2$ .

b) Given  $A$ , explain how to find an  $S$  so that  $\mathcal{E}_1 = \mathcal{E}_2$ .

c) What about uniqueness? Given  $S$ , explain how to find *all*  $A$  that yield  $\mathcal{E}_1 = \mathcal{E}_2$ . Given  $A$ , explain how to find *all*  $S$  that yield  $\mathcal{E}_1 = \mathcal{E}_2$ .

**15.2470. Uncovering a hidden linear explanation.** Consider a set of vectors  $y_1, \dots, y_N \in \mathbb{R}^n$ , which might represent a collection of measurements or other data. Suppose we have

$$y_i \approx Ax_i + b, \quad i = 1, \dots, N,$$

where  $A \in \mathbb{R}^{n \times m}$ ,  $x_i \in \mathbb{R}^m$ , and  $b \in \mathbb{R}^n$ , with  $m < n$ . (Our main interest is in the case when  $N$  is much larger than  $n$ , and  $m$  is smaller than  $n$ .) Then we say that  $y = Ax + b$  is a *linear explanation* of the data  $y$ . We refer to  $x$  as the vector of *factors* or *underlying causes* of the data  $y$ . For example, suppose  $N = 500$ ,  $n = 30$ , and  $m = 5$ . In this case we have 500 vectors; each vector  $y_i$  consists of 30 scalar measurements or data points. But these 30-dimensional data points can be ‘explained’ by a much smaller set of 5 ‘factors’ (the components of  $x_i$ ). This problem is about uncovering, or discovering, a linear explanation of a set of data, given only the data. In other words, we are given  $y_1, \dots, y_N$ , and the goal is to find  $m$ ,  $A$ ,  $b$ , and  $x_1, \dots, x_N$  so that  $y_i \approx Ax_i + b$ . To judge the accuracy of a proposed explanation, we’ll use the RMS explanation error, *i.e.*,

$$J = \left( \frac{1}{N} \sum_{i=1}^N \|y_i - Ax_i - b\|^2 \right)^{1/2}.$$

One rather simple linear explanation of the data is obtained with  $x_i = y_i$ ,  $A = I$ , and  $b = 0$ . In other words, the data explains itself! In this case, of course, we have  $y_i = Ax_i + b$ , so the RMS explanation error is zero. But this is not what we’re after. To be a useful explanation, we want to have  $m$  substantially smaller than  $n$ , *i.e.*, substantially fewer factors than the dimension of the original data (and for this smaller dimension, we’ll accept a nonzero, but hopefully small, value of  $J$ .) Generally, we want  $m$ , the number of factors in the explanation, to be as small as possible, subject to the constraint that  $J$  is not too large. Even if we fix the number of factors as  $m$ , a linear explanation of a set of data is not unique. Suppose  $A$ ,  $b$ , and  $x_1, \dots, x_N$  is a linear explanation of our data, with  $x_i \in \mathbb{R}^m$ . Then we can multiply the matrix  $A$  by two (say), and divide each vector  $x_i$  by two, and we have another linear explanation of the original data. More generally, let  $F \in \mathbb{R}^{m \times m}$  be invertible, and  $g \in \mathbb{R}^m$ . Then we have

$$y_i \approx Ax_i + b = (AF^{-1})(Fx_i + g) + (b - AF^{-1}g).$$

Thus,

$$\tilde{A} = AF^{-1}, \quad \tilde{b} = b - AF^{-1}g, \quad \tilde{x}_1 = Fx_1 + g, \quad \dots, \quad \tilde{x}_N = Fx_N + g$$

is another equally good linear explanation of the data. In other words, we can apply any affine (*i.e.*, linear plus constant) mapping to the underlying factors  $x_i$ , and generate another equally good explanation of the original data by appropriately adjusting  $A$  and  $b$ . To standardize or normalize the linear explanation, it is usually assumed that

$$\frac{1}{N} \sum_{i=1}^N x_i = 0, \quad \frac{1}{N} \sum_{i=1}^N x_i x_i^\top = I.$$

In other words, the underlying factors have an average value zero, and unit sample covariance. (You don’t need to know what covariance is — it’s just a definition here.) Finally, the problem.

- a) Explain clearly how you would find a hidden linear explanation for a set of data  $y_1, \dots, y_N$ . Be sure to say how you find  $m$ , the dimension of the underlying causes, the matrix  $A$ ,

the vector  $b$ , and the vectors  $x_1, \dots, x_N$ . Explain clearly why the vectors  $x_1, \dots, x_N$  have the required properties.

- b) Carry out your method on the data in the file `linexp_data.json` available on the course web site. The file gives the matrix  $Y = [y_1 \cdots y_N]$ . Verify that  $y_i \approx Ax_i + b$  by calculating the norm of the error vector,  $\|y_i - Ax_i - b\|$ , for  $i = 1, \dots, N$ . Sort these norms in descending order and plot them. (This gives a good picture of the distribution of explanation errors.) By explicit (Julia) computation verify that the vectors  $x_1, \dots, x_N$  obtained, have the required properties.

**16.2570. Alternating projections for low rank matrix completion.** In the *low rank matrix completion problem*, you are given *some* of the entries of a matrix, along with an upper bound on its rank; you are to guess or estimate the remaining entries. This arises in several applications, one of which is described at the end of this problem. This question investigates a heuristic method for the low rank matrix completion problem.

You are told that  $A \in \mathbb{R}^{m \times n}$  has rank  $\leq r$ , and that  $A_{ij} = A_{ij}^{\text{known}}$  for  $(i, j) \in \mathcal{K}$ , where  $\mathcal{K} \subseteq \{1, \dots, m\} \times \{1, \dots, n\}$  is the set of indices of the known entries. (You are given  $A_{ij}^{\text{known}}$  for  $(i, j) \in \mathcal{K}$ .) We let  $p = |\mathcal{K}|$  denote the number of known entries. You are to estimate or guess the entries  $A_{ij}$ , for  $(i, j) \notin \mathcal{K}$ .

You will use an alternating projection method to find an estimate  $\hat{A}$  of  $A$ . After choosing an initial point  $\hat{A}^{(0)}$ , that has the known correct entries (*i.e.*,  $\hat{A}_{ij}^{(0)} = A_{ij}^{\text{known}}$  for  $(i, j) \in \mathcal{K}$ ), you will alternate between two projections. For  $k = 0, 1, \dots$  you carry out the following steps:

- *Project to the closest matrix satisfying the rank constraint.* Set  $\tilde{A}^{(k)}$  to be the matrix of rank  $\leq r$  that is closest to  $\hat{A}^{(k)}$  in Frobenius norm, *i.e.*, that minimizes

$$\|\tilde{A}^{(k)} - \hat{A}^{(k)}\|_F = \left( \sum_{i=1}^m \sum_{j=1}^n (\tilde{A}_{ij}^{(k)} - \hat{A}_{ij}^{(k)})^2 \right)^{1/2}$$

subject to  $\text{rank}(\tilde{A}^{(k)}) \leq r$ .

- *Project to the closest matrix with the known entries.* Set  $\hat{A}^{(k+1)}$  to be the matrix with the given known entries that is closest to  $\tilde{A}^{(k)}$  in Frobenius norm, *i.e.*, that minimizes

$$\|\hat{A}^{(k+1)} - \tilde{A}^{(k)}\|_F = \left( \sum_{i=1}^m \sum_{j=1}^n (\hat{A}_{ij}^{(k+1)} - \tilde{A}_{ij}^{(k)})^2 \right)^{1/2}$$

subject to  $\hat{A}_{ij}^{(k+1)} = A_{ij}^{\text{known}}$  for  $(i, j) \in \mathcal{K}$ .

This is a heuristic method: It can fail to converge at all, or it can converge to different limit points, depending on the starting point. But it often works well.

- a) Clearly explain how to perform each of these projections. We will subtract points for technically correct, but overly complicated methods. Do not use any Julia notation in your answer.

- b) Use 300 steps of the alternating projections algorithm to find a low rank matrix completion estimate for the problem defined in `low_rank_matrix_completion.json`. This file defines the rank upper bound  $r$ , the dimensions  $m$  and  $n$ , and the known matrix entries. The known matrix indices are given as a  $p \times 2$  matrix  $K$ , with each row giving  $(i, j)$  for one known entry. The  $p$ -vector `Aknown` gives the corresponding known values.

Initialize your method as follows. Let  $\mu$  denote the mean of all the known entries. Set  $\hat{A}_{ij}^{(0)} = A_{ij}^{\text{known}}$  for  $(i, j) \in \mathcal{K}$ , and  $\hat{A}_{ij}^{(0)} = \mu$  for  $(i, j) \notin \mathcal{K}$ .

To judge the performance of the algorithm, the data file also gives the actual matrix  $A$  as `Atrue`. (Of course in applications, you would not have access to the matrix  $A$ !) Plot  $\|\hat{A}^{(k)} - A\|_F$ , for  $k = 1, \dots, 300$ .

Make a very brief comment about how well the algorithm worked on this data set. You can allude to the fact that you are given only around one sixth of the entries of  $A$ .

**Remark.** *None of this is needed to solve the problem; it is only for your amusement and interest.* Algorithms like this can be used for problems like the *Netflix challenge*. Here  $A_{ij}$  represents the rating user  $i$  gives (or would give) to movie  $j$ . We have access to some of the ratings, and want to predict other ratings before they are given. (This would allow us to make recommendations, for example.) It is reasonable to assume (and is confirmed with real data) that ratings matrices like  $A$  have (approximately) low rank. This can be interpreted as meaning that a user's rating is (mostly) determined by a relatively small number of factors. The entries in the  $k$ th left singular vector tell us how much the user ratings are influenced (positively or negatively) by factor  $k$ ; the entries in the  $k$ th right singular vector tell us how much of factor  $k$  (positive or negative) is in each movie.