Pr. 1. (sol/hs065)

$$\begin{aligned}
\boldsymbol{A} &= \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \\
&= \begin{bmatrix} \boldsymbol{q}_1 & \dots, \boldsymbol{q}_k & -\boldsymbol{q}_{k+1} & \dots & -\boldsymbol{q}_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & & & & \\ & \ddots & & & & \\ & & \lambda_k & & & \\ & & & -\lambda_{k+1} & & \\ & & & & \ddots & \\ & & & & & -\lambda_n \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_1 & \dots & \boldsymbol{q}_k & \boldsymbol{q}_{k+1} & \dots & \boldsymbol{q}_n \end{bmatrix} \\
&= \tilde{\boldsymbol{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\boldsymbol{V}}^T.
\end{aligned}$$

It can be easily checked that $\tilde{\boldsymbol{U}}$ and $\tilde{\boldsymbol{V}}$ are orthogonal matrices, and that $\tilde{\boldsymbol{\Sigma}}$ is a diagonal matrix with real, positive entries along its diagonal. However, these diagonal elements are not necessarily in descending order. Let $\sigma_1, \sigma_2, \dots, \sigma_n$ be the sequence $\{\lambda_1, \dots, \lambda_k, -\lambda_{k+1}, \dots, -\lambda_n\}$ sorted in descending order. That is, $\sigma_1 = \max\{\lambda_1, \dots, \lambda_k, -\lambda_{k+1}, \dots, -\lambda_n\}$, $\sigma_2 = \max\{\lambda_1, \dots, \lambda_k, -\lambda_{k+1}, \dots, -\lambda_n\} \backslash \{\sigma_1\}$, and so on. Let $\boldsymbol{u}_{\sigma_i}, \boldsymbol{v}_{\sigma_i}$ be the eigenvectors of $\boldsymbol{A}$ such that

$$\boldsymbol{u}_{\sigma_i} = \begin{cases} \boldsymbol{q}_j & \text{if } \sigma_i = \lambda_j, \quad j \in \{1, \dots, k\}, \\ -\boldsymbol{q}_j & \text{if } \sigma_i = -\lambda_j, \quad j \in \{k+1, \dots, n\}, \end{cases}$$

and

$$\boldsymbol{v}_{\sigma_i} = \begin{cases} \boldsymbol{q}_j & \text{if } \sigma_i = \lambda_j, \quad j \in \{1, \dots, k\}, \\ \boldsymbol{q}_j & \text{if } \sigma_i = -\lambda_j, \quad j \in \{k+1, \dots, n\}. \end{cases}$$

Then an SVD of $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{u}_{\sigma_1} & \dots & \boldsymbol{u}_{\sigma_n} \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{\sigma_1} & \dots & \boldsymbol{v}_{\sigma_n} \end{bmatrix} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T,$$

since $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal (they are just column rearranged versions of $\tilde{\boldsymbol{U}}$ and $\tilde{\boldsymbol{V}}$), and $\boldsymbol{\Sigma}$ is a diagonal matrix, with real, positive entries along the diagonal, sorted in descending order.
Here we have flipped the signs on the negative eigenvalues, and the eigenvectors corresponding to the negative eigenvalues in the left–most matrix of the expression $\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$. Alternatively, one could change signs of the appropriate vectors the right–most matrix of the same expression.

Pr. 2. (sol/hs032)
Here we have $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}^T \in \mathbb{R}^{n \times n}$.

(a) Let $\boldsymbol{v}_i$ denote an eigenvector of $\boldsymbol{B}$ associated with the eigenvalue λ_i for $i = 1 \dots n$. Then by definition we have:

$$\boldsymbol{B}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}.$$

Substituting $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}^T$ into the above expression yields:

$$(\boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}^T)\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i.$$

Rearranging terms (to help solve for $\boldsymbol{v}_i$) yields:

$$(\boldsymbol{A} - \lambda_i \boldsymbol{I})\boldsymbol{v}_i = -\boldsymbol{x}\boldsymbol{x}^T \boldsymbol{v}_i.$$

Noting that $c_i \triangleq -\boldsymbol{x}^T \boldsymbol{v}_i$ is a scalar:

$$(\boldsymbol{A} - \lambda_i \boldsymbol{I})\boldsymbol{v}_i = c_i \boldsymbol{x}.$$

Let us rearrange the terms one more time. Assuming that $\boldsymbol{A} - \lambda_i \boldsymbol{I}$ is invertible (which will be true so long as $\boldsymbol{B}$ and $\boldsymbol{A}$ do not share any common eigenvalues), we have that

$$\boldsymbol{v}_i = c_i (\boldsymbol{A} - \lambda_i \boldsymbol{I})^{-1} \boldsymbol{x},$$

or equivalently that

$$\boldsymbol{v}_i \propto (\boldsymbol{A} - \lambda_i \boldsymbol{I})^{-1} \boldsymbol{x}.$$

To get unit-norm eigenvectors we must have that:

$$\boldsymbol{v}_i = \frac{(\boldsymbol{A} - \lambda_i \boldsymbol{I})^{-1} \boldsymbol{x}}{\|(\boldsymbol{A} - \lambda_i \boldsymbol{I})^{-1} \boldsymbol{x}\|_2},$$

which is the desired expression for an eigenvector of $\boldsymbol{B}$ in terms of $\boldsymbol{A}$, $\boldsymbol{x}$ and λ_i. Note that a key part of the derivation involves recognizing that $c_i = -\boldsymbol{x}^T\boldsymbol{v}_i$ is a just a scale (or normalization) factor and we do not have to solve for it explicitly.

Furthermore, the only time some of the eigenvalues of $\boldsymbol{B}$ will equal the eigenvalues of $\boldsymbol{A}$ will be if $\boldsymbol{x}$ is collinear with any of the eigenvectors of $\boldsymbol{B}$. This is a special case but other than that we have a complete solution given by the above expression.

(b) To prove orthogonality:

$$\begin{aligned}
\boldsymbol{v}_j^T\boldsymbol{v}_k &\propto \boldsymbol{x}^T(\boldsymbol{A}-\lambda_j\boldsymbol{I})^{-1}(\boldsymbol{A}-\lambda_k\boldsymbol{I})^{-1}\boldsymbol{x} \\
&= \sum_{i=1}^n \frac{x_i^2}{(\lambda_j - a_{ii})(\lambda_k - a_{jj})} \\
&= \sum_{i=1}^n \frac{1}{(\lambda_k-\lambda_j)}\left(\frac{x_i^2}{\lambda_j - a_{ii}} - \frac{x_i^2}{\lambda_k - a_{ii}}\right) \\
&= \frac{1}{(\lambda_k-\lambda_j)}\left(\sum_{i=1}^n \frac{x_i^2}{\lambda_j - a_{ii}} - \sum_{i=1}^n \frac{x_i^2}{\lambda_k - a_{ii}}\right). \qquad (1)
\end{aligned}$$

But from previous HW, for every eigenvalue λ of $\boldsymbol{B}$, $\sum_{i=1}^n \frac{x_i^2}{\lambda - a_{ii}} = 1$. Therefore (1) becomes $\frac{1}{(\lambda_k-\lambda_j)}(1-1) = 0$, showing that the eigenvectors are orthogonal.

Pr. 3. (sol/hs031)

Given $\boldsymbol{A} = \boldsymbol{x}\boldsymbol{x}^T + \boldsymbol{y}\boldsymbol{y}^T$.

The rank of $\boldsymbol{A}$ is at most two. It is equal to zero when $\boldsymbol{x}\boldsymbol{x}^T = -\boldsymbol{y}\boldsymbol{y}^T$ and equals one when $\boldsymbol{x}$ is collinear with $\boldsymbol{y}$. We treat the setting where the rank of $\boldsymbol{A}$ is two. In other words, $\boldsymbol{x}$ and $\boldsymbol{y}$ are linearly independent pairs of vectors.

We first compute the eigenvalues of $\boldsymbol{A}$. Let $\boldsymbol{Z} = \begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}$ so that we can write $\boldsymbol{A}$ as:

$$\boldsymbol{A} = \begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}^T = \boldsymbol{Z}\boldsymbol{Z}^T.$$

If $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{A}$ associated with the eigenvalue $z \neq 0$, then

$$\boldsymbol{Z}\boldsymbol{Z}^T\boldsymbol{v} = z\boldsymbol{v}.$$

Thus:

$$\begin{aligned}
&\boldsymbol{Z}^T\boldsymbol{Z}\boldsymbol{Z}^T\boldsymbol{v} = z\boldsymbol{Z}^T\boldsymbol{v}, \\
\Longrightarrow\ &(\boldsymbol{Z}^T\boldsymbol{Z})\underbrace{(\boldsymbol{Z}^T\boldsymbol{v})}_{\widetilde{\boldsymbol{v}}} = z\underbrace{(\boldsymbol{Z}^T\boldsymbol{v})}_{=\widetilde{\boldsymbol{v}}} \\
\Longrightarrow\ &(\boldsymbol{Z}^T\boldsymbol{Z})\widetilde{\boldsymbol{v}} = z\widetilde{\boldsymbol{v}} \\
\Longrightarrow\ &z \text{ is an eigenvalue of } \boldsymbol{M} \triangleq \boldsymbol{Z}^T\boldsymbol{Z} = \begin{bmatrix}\|\boldsymbol{x}\|_2^2 & \rho \\ \rho & \|\boldsymbol{y}\|_2^2\end{bmatrix},
\end{aligned}$$

where $\rho = \boldsymbol{x}'\boldsymbol{y}$. Solving the quadratic formula for $\det(\boldsymbol{Z}'\boldsymbol{Z} - z\boldsymbol{I}) = 0$ gives the eigenvalues

$$z_{1,2} = \frac{\|\boldsymbol{x}\|_2^2 + \|\boldsymbol{y}\|_2^2 \pm \sqrt{(\|\boldsymbol{x}\|_2^2 - \|\boldsymbol{y}\|_2^2)^2 + 4\rho^2}}{2}.$$

The remaining $n - r$ eigenvalues must be zero since the rank of $\boldsymbol{A}$ is at most two.

If $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{A}$, then $\boldsymbol{A}\boldsymbol{v} = (\boldsymbol{x}^T\boldsymbol{v})\,\boldsymbol{x} + (\boldsymbol{y}^T\boldsymbol{v})\,\boldsymbol{y} = \lambda\boldsymbol{v}$. Thus $\boldsymbol{v}$ has to be in $\operatorname{span}(\{\boldsymbol{x},\boldsymbol{y}\}) = \mathcal{R}(\boldsymbol{Z})$. Let $\boldsymbol{v} = \begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\begin{bmatrix}\beta_1 \\ \beta_2\end{bmatrix} = \boldsymbol{Z}\boldsymbol{\beta}$, for some $\boldsymbol{\beta} \in \mathbb{R}^2$. Working with the eigenvalue/eigenvector definition we have that:

$$\begin{aligned}
&\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}^T \\ \boldsymbol{y}^T\end{bmatrix}\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta} = \lambda\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta} \\
&\qquad\Longrightarrow \begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{M}\boldsymbol{\beta} = \lambda\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta} \\
&\qquad\Longrightarrow \begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}(\boldsymbol{M}\boldsymbol{\beta} - \lambda\boldsymbol{\beta}) = \boldsymbol{0}. \qquad (2)
\end{aligned}$$

For equation (2) to hold we must have that $\boldsymbol{M\beta} - \lambda\boldsymbol{\beta} = 0$. Why? Because, we assumed that $\boldsymbol{x}$ and $\boldsymbol{y}$ were linearly independent so the only way (2) is true is if $\boldsymbol{M\beta} - \lambda\boldsymbol{\beta} = 0 \in \mathbb{R}^{2\times 1}$. Thus we have that

$$\boxed{\boldsymbol{M\beta} = \lambda\boldsymbol{\beta}}$$

or equivalently, that $\boldsymbol{\beta} \in \mathbb{R}^{2\times 1}$ is an *eigenvector* of $\boldsymbol{M}$ corresponding to the eigenvalue λ.
We need the eigenvectors of $\boldsymbol{ZZ}^T$ to be unit-norm, so we normalize them. Thus the unit-norm eigenvectors of $\boldsymbol{ZZ}^T$, which are exactly equal to the eigenvectors of $\boldsymbol{A}$, are simply:

$$\widetilde{\boldsymbol{u}}_1 = \frac{\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta}_1}{\|\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta}_1\|_2}, \qquad \text{and } \widetilde{\boldsymbol{u}}_2 = \frac{\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta}_2}{\|\begin{bmatrix}\boldsymbol{x} & \boldsymbol{y}\end{bmatrix}\boldsymbol{\beta}_2\|_2},$$

where $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ denote the eigenvectors of $\boldsymbol{M}$ corresponding to the eigenvalues z_1 and z_2 above.
The point of this exercise is to make you see these computations in matrix-vector terms. Knowing that $\boldsymbol{A}$ has rank 2 means that there is a 2×2 matrix lurking within the problem whose solution will yield the eigenvalues and the eigenvectors. This is precisely what this "slick" solution does. Do not worry if you did not get this right away; by the end of this semester you will start seeing these computations in this clean way and have at your disposal an extremely powerful and versatile technique for "seeing" the structure in the kinds of matrix-valued problems that routinely arise in modern signal processing.

Pr. 4. (sol/hs121)

(a) Suppose $\boldsymbol{A}$ is $p \times m$, $\boldsymbol{X}$ is $m \times n$, and $\boldsymbol{B}$ is $q \times n$. Let us represent these matrices as

$$\boldsymbol{X} = \begin{bmatrix}\boldsymbol{x}_1 & \dots & \boldsymbol{x}_n\end{bmatrix}, \quad \boldsymbol{B}^T = \begin{bmatrix}\boldsymbol{b}_1 & \dots & \boldsymbol{b}_q\end{bmatrix}.$$

Using these definitions, we have

$$\boldsymbol{AXB}^T = \boldsymbol{A}\begin{bmatrix}\boldsymbol{x}_1 & \dots & \boldsymbol{x}_n\end{bmatrix}\begin{bmatrix}\boldsymbol{b}_1 & \dots & \boldsymbol{b}_q\end{bmatrix} = \boldsymbol{A}\left[\left(\sum_{k=1}^n \boldsymbol{b}_{1k}\boldsymbol{x}_k\right)\right]$$

$$= \begin{bmatrix}\boldsymbol{A}\left(\sum_{k=1}^n b_{1k}\boldsymbol{x}_k\right) & \dots & \boldsymbol{A}\left(\sum_{k=1}^n b_{qk}\boldsymbol{x}_k\right)\end{bmatrix} \Longrightarrow \operatorname{vec}(\boldsymbol{AXB}^T) = \begin{bmatrix}\boldsymbol{A}\left(\sum_{k=1}^n b_{1k}\boldsymbol{x}_k\right) \\ \vdots \\ \boldsymbol{A}\left(\sum_{k=1}^n b_{qk}\boldsymbol{x}_k\right)\end{bmatrix}.$$

On the other hand,

$$(\boldsymbol{B} \otimes \boldsymbol{A})\operatorname{vec}(\boldsymbol{X}) = \begin{bmatrix} b_{11}\boldsymbol{A} & \dots & b_{1n}\boldsymbol{A} \\ \vdots & \ddots & \vdots \\ b_{q1}\boldsymbol{A} & \dots & b_{qn}\boldsymbol{A}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n\end{bmatrix} = \begin{bmatrix}\sum_{k=1}^n b_{1k}\boldsymbol{A}\boldsymbol{x}_k \\ \vdots \\ \sum_{k=1}^n b_{qk}\boldsymbol{A}\boldsymbol{x}_k\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}\left(\sum_{k=1}^n b_{1k}\boldsymbol{x}_k\right) \\ \vdots \\ \boldsymbol{A}\left(\sum_{k=1}^n b_{qk}\boldsymbol{x}_k\right)\end{bmatrix}.$$

This establishes the result.

(b) $\boldsymbol{F}_M\boldsymbol{X}$ multiplies each column of $\boldsymbol{X}$ by $\boldsymbol{F}_M$, so it computes the M-point 1D DFT of each column of $\boldsymbol{X}$. Therefore, $(\boldsymbol{F}_N\boldsymbol{X}^T)^T = \boldsymbol{X}\boldsymbol{F}_N^T$ computes the N-point 1D DFT of each row of $\boldsymbol{X}$. Combining these operations yields $\boldsymbol{F}_X = \boldsymbol{F}_M\boldsymbol{X}\boldsymbol{F}_N^T$. Note that we use transpose, not Hermitian transpose (even for complex-valued data). Applying (a) yields

$$\operatorname{vec}(\boldsymbol{F}_X) = (\boldsymbol{F}_N \otimes \boldsymbol{F}_M)\operatorname{vec}(\boldsymbol{X}).$$

(c) Observe that

$$\frac{1}{MN}\boldsymbol{F}_M^H\boldsymbol{F}_X\overline{\boldsymbol{F}_N} = \underbrace{\left(\frac{1}{M}\boldsymbol{F}_M^H\boldsymbol{F}_M\right)}_{=\boldsymbol{I}_M}\boldsymbol{X}\underbrace{\left(\frac{1}{N}\boldsymbol{F}_N^T\overline{\boldsymbol{F}_N}\right)}_{=:\boldsymbol{Y}},$$

where $\boldsymbol{Y} = \frac{1}{N}\boldsymbol{F}_N^T\overline{\boldsymbol{F}_N} = \overline{\left(\frac{1}{N}\overline{\boldsymbol{F}_N}^T\boldsymbol{F}_N\right)} = \overline{\left(\frac{1}{N}\boldsymbol{F}_N^H\boldsymbol{F}_N\right)} = \overline{\boldsymbol{I}_N} = \boldsymbol{I}_N$. Applying (a):

$$\operatorname{vec}(\boldsymbol{X}) = \frac{1}{MN}(\boldsymbol{F}_N \otimes \boldsymbol{F}_M)^H\operatorname{vec}(\boldsymbol{F}_X) = \frac{1}{MN}(\boldsymbol{F}_N^H \otimes \boldsymbol{F}_M^H)\operatorname{vec}(\boldsymbol{F}_X).$$

Pr. 5. (sol/hsj02)

(a) If $\boldsymbol{A} = \boldsymbol{U\Sigma V}'$ then $\boldsymbol{A}'\boldsymbol{A} = \boldsymbol{V\Sigma}'\boldsymbol{\Sigma V}'$, so singular values of $\boldsymbol{A}'\boldsymbol{A}$ are the squares of those of $\boldsymbol{A}$. Thus the condition number of $\boldsymbol{A}'\boldsymbol{A}$ is σ_1^2/σ_N^2.

(b) Here $\boldsymbol{A}'\boldsymbol{A}+\beta\boldsymbol{I} = \boldsymbol{V\Sigma}'\boldsymbol{\Sigma V}'+\beta\boldsymbol{I} = \boldsymbol{V}(\boldsymbol{\Sigma}'\boldsymbol{\Sigma}+\beta\boldsymbol{I})\boldsymbol{V}'$ so its singular values are $\sigma_n^2+\beta$. Thus the condition number of $\boldsymbol{A}'\boldsymbol{A}+\beta\boldsymbol{I}$ is $\frac{\sigma_1^2+\beta}{\sigma_N^2+\beta}$.
Because $\beta > 0$, this condition number is always smaller and in part (a), so the regularized version is better conditioned.

Pr. 6. (sol/hsj03)
If $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{T}$, then $\boldsymbol{Tx} = \lambda\boldsymbol{x}$ so $\boldsymbol{T}^3\boldsymbol{x} = \lambda^3\boldsymbol{x}$. But here $\boldsymbol{T}^3 = \boldsymbol{I}$ so $\boldsymbol{x} = \lambda^3\boldsymbol{x}$, which means $\lambda^3 = 1$ so $\lambda \in \{1, \exp(\pm\imath 2\pi/3)\}$.

Pr. 7. (sol/hs115)

(a) Here we use the simple observation that the largest in magnitude eigenvalue of the matrix $\boldsymbol{B} = \boldsymbol{A} - \lambda_1(\boldsymbol{A})\boldsymbol{I}$ is the smallest eigenvalue of $\boldsymbol{A}$ because the largest eigenvalue of $\boldsymbol{B}$ will now exactly equal zero! So we can apply the power method to $\boldsymbol{B}$ to obtain the eigenvector $\boldsymbol{x}$ associated with the smallest eigenvalue. Computing the quadratic form $\boldsymbol{x}^T\boldsymbol{Ax}$ will give us an approximation of the smallest eigenvalue of $\boldsymbol{A}$.

(b) If we do not know λ_1 then we need to estimate it first.
If the largest eigenvalue of $\boldsymbol{A}$ is also its largest in magnitude then we apply the power method to estimate λ_1 by first computing an approximation of the eigenvector $\boldsymbol{x}$ via the power method and then approximating the largest eigenvalue of $\boldsymbol{A}$ via the quadratic form $\boldsymbol{x}^T\boldsymbol{Ax}$. We then compute $\boldsymbol{B}$ and repeat the procedure described earlier.
If, however, the largest eigenvalue of $\boldsymbol{A}$ is not its largest in magnitude, then it means that by running the power method and computing $\boldsymbol{x}^T\boldsymbol{Ax}$, we have computed the smallest eigenvalue of $\boldsymbol{A}$ instead. So, if we were to run the power method on the matrix $\boldsymbol{B}$, we will have computed the eigenvector $\boldsymbol{x}$ associated with the largest eigenvalue of $\boldsymbol{A}$. We can resolve this ambiguity by computing the quadratic form $\boldsymbol{x}^T\boldsymbol{Ax}$ for each of the two eigenvectors thus computed via the power method, and the smaller of the two quadratic forms will be the smallest eigenvalue.

Pr. 8. (sol/hs120)

(a) Here $f(\boldsymbol{x}) = \boldsymbol{A}^T(\boldsymbol{Ax}-\boldsymbol{b})$, so that

$$\begin{aligned}
\|f(\boldsymbol{x}) - f(\boldsymbol{y})\| &= \|\boldsymbol{A}^T(\boldsymbol{Ax}-\boldsymbol{b}) - \boldsymbol{A}^T(\boldsymbol{Ay}-\boldsymbol{b})\| = \|\boldsymbol{A}^T\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{y})\| \\
&\leq \|\boldsymbol{A}^T\boldsymbol{A}\|_2 \cdot \|\boldsymbol{x}-\boldsymbol{y}\|_2 = \sigma_1^2(\boldsymbol{A})\|\boldsymbol{x}-\boldsymbol{y}\| \qquad (\text{ since } \|\boldsymbol{A}^T\boldsymbol{A}\|_2 = \sigma_1(\boldsymbol{A}^T\boldsymbol{A}) = \sigma_1(\boldsymbol{A})^2),
\end{aligned}$$

and so $\boldsymbol{A}^T(\boldsymbol{Ax}-\boldsymbol{b})$ is a Lipschitz function with Lipschitz constant $\sigma_1^2(\boldsymbol{A})$.

(b) Now let $f(\boldsymbol{X}) = \sigma_1(\boldsymbol{X})$. Because $\boldsymbol{X} = \boldsymbol{Y} - (\boldsymbol{Y}-\boldsymbol{X})$:

$$\begin{aligned}
\sigma_1(\boldsymbol{X}) &= \operatorname*{arg\,max}_{\|\boldsymbol{u}\|_2=1} \|\boldsymbol{Xu}\|_2 = \operatorname*{arg\,max}_{\|\boldsymbol{u}\|_2=1} \|[\boldsymbol{Y} - (\boldsymbol{Y}-\boldsymbol{X})]\boldsymbol{u}\|_2 \\
&\leq \operatorname*{arg\,max}_{\|\boldsymbol{u}\|_2=1} (\|\boldsymbol{Yu}\|_2 + \|(\boldsymbol{Y}-\boldsymbol{X})\boldsymbol{u}\|_2) \qquad (\text{by the triangle inequality }) \\
&\leq \left(\operatorname*{arg\,max}_{\|\boldsymbol{u}\|_2=1} \|\boldsymbol{Yu}\|_2\right) + \left(\operatorname*{arg\,max}_{\|\boldsymbol{u}\|_2=1} \|(\boldsymbol{Y}-\boldsymbol{X})\boldsymbol{u}\|_2\right) \\
&= \sigma_1(\boldsymbol{Y}) + \sigma_1(\boldsymbol{Y}-\boldsymbol{X}) = \sigma_1(\boldsymbol{Y}) + \sigma_1(\boldsymbol{X}-\boldsymbol{Y}) \qquad (\text{since } \sigma_1(\boldsymbol{X}-\boldsymbol{Y}) = \sigma_1(\boldsymbol{Y}-\boldsymbol{X})).
\end{aligned}$$

Thus we have established that $\sigma_1(\boldsymbol{X}) - \sigma_1(\boldsymbol{Y}) \leq \sigma_1(\boldsymbol{X}-\boldsymbol{Y})$.
Swapping $\boldsymbol{X}$ and $\boldsymbol{Y}$ in this inequality gives us $\sigma_1(\boldsymbol{Y}) - \sigma_1(\boldsymbol{X}) \leq \sigma_1(\boldsymbol{X}-\boldsymbol{Y})$.
Combining the two inequalities and using the inequality from an earlier problem shows

$$|\sigma_1(\boldsymbol{X}) - \sigma_1(\boldsymbol{Y})| \leq \sigma_1(\boldsymbol{X}-\boldsymbol{Y}) \leq \|\boldsymbol{X}-\boldsymbol{Y}\|_F.$$

Thus $\sigma_1(\cdot)$ is a 1-Lipschitz function with respect to the spectral norm and the Frobenius norm.

Pr. 9. (sol/hs077)
The output is plotted in Figure 1, for one realization of the system. Since $\mu \in (0, \frac{2}{\sigma_1^2(A)})$, we see that the iterations converge to the least squares solution.

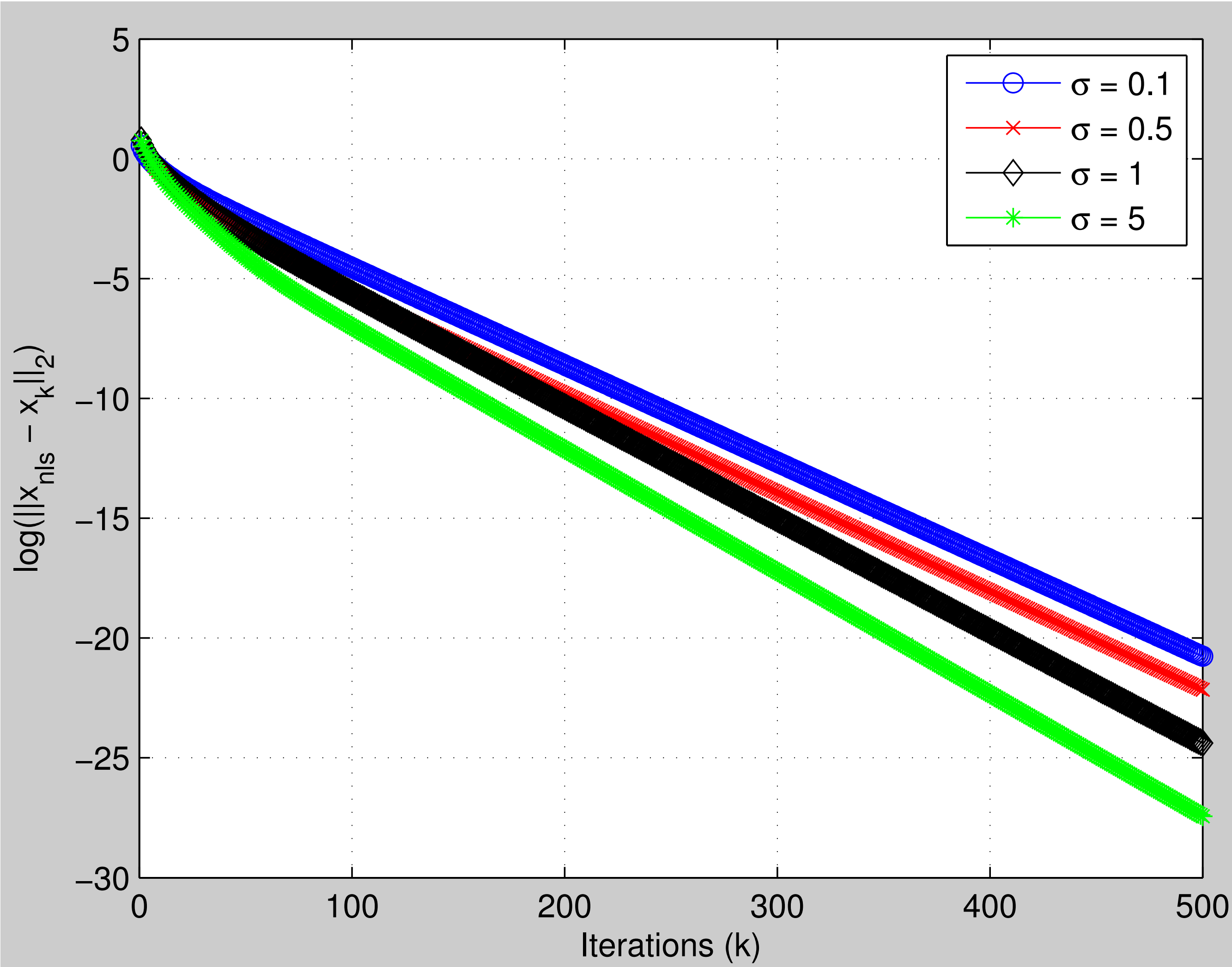
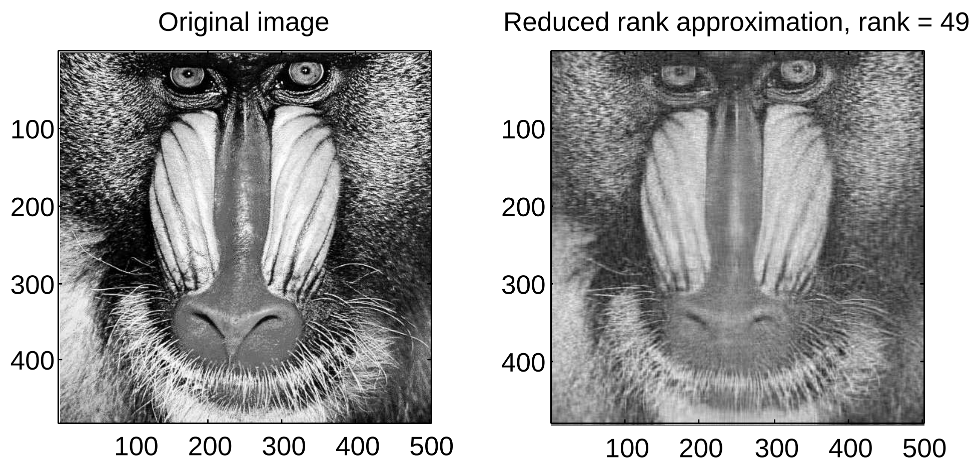


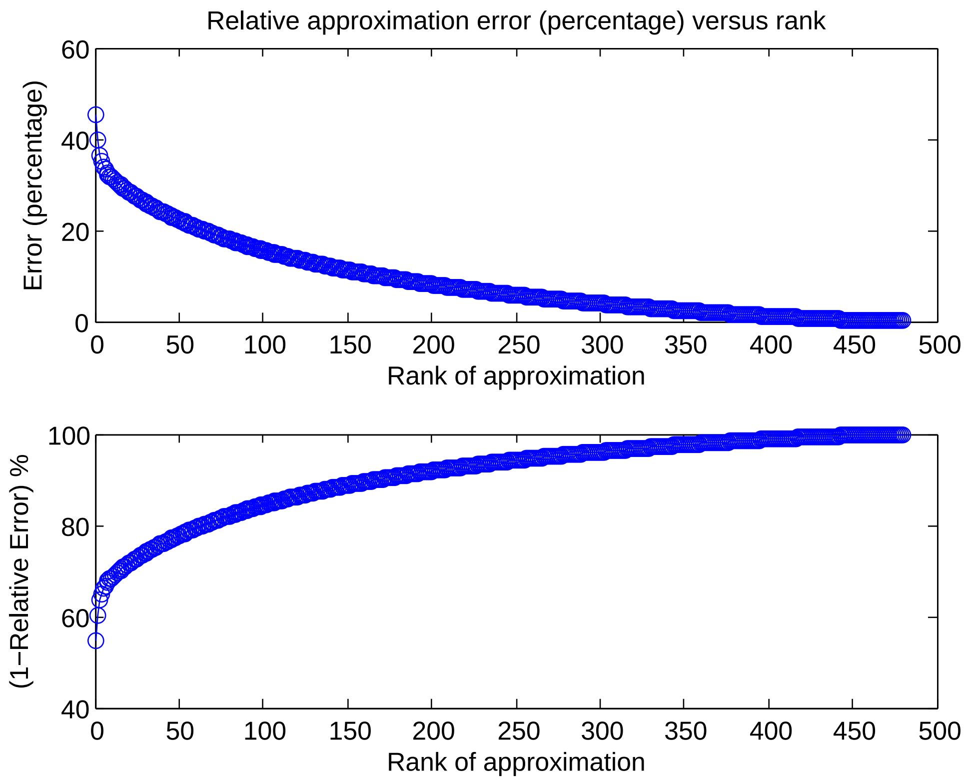
FIGURE 1. Problem 5: Convergence of Iterative Least Squares solution x_k to the true solution $\widehat{x} = A^{\dagger}b$.

Pr. 10. (sol/hs042)

(a) The idea is to view the image as a matrix $\boldsymbol{A} = \sum_{i=1}^{n} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$ where in this case $n = 480$. We know that the optimal (with respect to any unitarily invariant norm) rank-k approximation to $\boldsymbol{A}$ is $\boldsymbol{A}_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$. This approximation requires $1+480+500 = 981$ real numbers for each additional term. The original matrix stores $480 \times 500 = 240000$. One-fifth of this total is 48000. Thus the number of terms that gives 48000 real numbers is $48.93 \approx 49$.
Note that if you just took one fifth of the number of terms in the full image, that would give you 96 terms, but that would entail a roughly 40% reduction as opposed to the 20%.
In general, for an $m \times n$ matrix, the rank r approximation requires $r \times (1 + m + n)$ real numbers. Hence, for a given compression fraction p, we want to choose the maximum value of r such that $\frac{r(1+m+n)}{mn} \leq p$. I.e., r can be chosen as the *floor* of $\frac{p \times mn}{1+m+n}$.



(A) true image versus low rank image.



(B) relative error in approximation.

A possible `Julia` implementation is

```
function compress_image(A, p)
#
# Syntax:       Ac, r = compress_image(A, p)
#
# Inputs:       A is an m x n matrix
#
#               p is a scalar in (0, 1]
#
# Outputs:      Ac is an m x n matrix containing a compressed version of A that
#               can be represented using at most (100 * p)% as many bits
#               required to represent A
#
#               r is the rank of Ac
#
```

```
    # Parse inputs
    m, n = size(A)

    # Compute compression factor
    r = Int64(floor(p * m * n / (m + n + 1)))

    # Compute compressed image
    U, s, V = svd(A, thin=true)
    Ac = U[:, 1:r] * diagm(s[1:r]) * V[:, 1:r]'

    return Ac, r
end
```

(b) The MIT logo without the lettering has rank 4 if the background white space is represented as a zero in the matrix; otherwise it is rank 5. Note that the singular values σ_k for $k \geq 6$ are not numerically 0, but that they are very, very small. Either way it is nearly perfectly compressible with a low-rank approximation whereas the logo with the lettering is full rank and so the approximation error decreases as the approximation rank increases. Being able to spot low-rank patterns is a valuable skill in data and computational science!