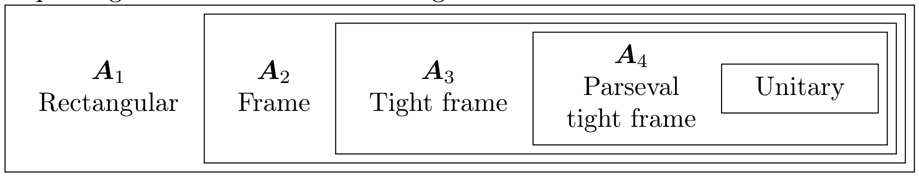
Pr. 1. (sol/hsj02)

(a) If $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}'$ then $\boldsymbol{A}'\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}'\boldsymbol{\Sigma}\boldsymbol{V}'$, 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 $\kappa(\boldsymbol{A}'\boldsymbol{A}) = \sigma_1^2/\sigma_N^2$.

(b) Here $\boldsymbol{A}'\boldsymbol{A} + \beta\boldsymbol{I} = \boldsymbol{V}\boldsymbol{\Sigma}'\boldsymbol{\Sigma}\boldsymbol{V}' + \beta\boldsymbol{I} = \boldsymbol{V}(\boldsymbol{\Sigma}'\boldsymbol{\Sigma} + \beta\boldsymbol{I})\boldsymbol{V}'$ so its singular values are $\sigma_n^2 + \beta$. Thus the condition number of $\boldsymbol{A}'\boldsymbol{A} + \beta\boldsymbol{I}$ is $\kappa(\boldsymbol{A}'\boldsymbol{A} + \beta\boldsymbol{I}) = \frac{\sigma_1^2+\beta}{\sigma_N^2+\beta}$.

Because $\beta > 0$, this condition number is always smaller than $\kappa(\boldsymbol{A}'\boldsymbol{A})$, so regularization improves conditioning.

Pr. 2. (sol/hsj44)
Repeating for reference the Venn diagram for frames:


$\boldsymbol{A}_1$ Rectangular
$\boldsymbol{A}_2$ Frame
$\boldsymbol{A}_3$ Tight frame
$\boldsymbol{A}_4$ Parseval tight frame
Unitary

(a) $\boldsymbol{A}_1 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} = \boldsymbol{B}_8$ is "wide" ($N \leq M$) but not a frame. An even simpler example is $\boldsymbol{A}_1 = \begin{bmatrix} 0 \end{bmatrix} = \boldsymbol{B}_0$.

$\boldsymbol{A}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} = \boldsymbol{B}_{11}$ is a frame but not a tight frame.

$\boldsymbol{A}_3 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \boldsymbol{B}_9$ is a tight frame but not a Parseval tight frame.

$\boldsymbol{A}_4 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \end{bmatrix} = \boldsymbol{B}_5$ is a Parseval tight frame but is not unitary.

(b) A diagonal matrix $\boldsymbol{D}$ is unitary iff $\boldsymbol{D}^{-1} = \boldsymbol{D}'$, which holds iff its diagonal elements satisfy $1/d_{jj} = d_{jj}^*$ or equivalently $1 = |d_{jj}|^2$. In other words, the diagonal elements all have unit magnitude, i.e., $d_{jj} = \exp(\imath\phi_j)$.

Pr. 3. (sol/hs115)

(a) Here we use the simple observation that the largest in magnitude eigenvalue of the matrix $\boldsymbol{B} = \boldsymbol{A} - \lambda_1(\boldsymbol{A})\boldsymbol{I}$ corresponds to 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 $\boldsymbol{v}_N$ associated with the smallest eigenvalue λ_N.

(b) Computing the quadratic form $\boldsymbol{v}_N'\boldsymbol{A}\boldsymbol{v}_N = \boldsymbol{v}_N'\lambda_N\boldsymbol{v}_N = \lambda_N \|\boldsymbol{v}_N\|_2^2 = \lambda_N$ gives a numerical approximation to λ_N, the smallest eigenvalue of $\boldsymbol{A}$. The factor $\|\boldsymbol{v}_N\|_2^2$ is invariant to the "sign" of the eigenvector

Another option is to use the ratio $\lambda_N = [\boldsymbol{A}\boldsymbol{v}_N]_i/[\boldsymbol{v}_N]_i$, as long as $[\boldsymbol{v}_N]_i$ is nonzero. This approach can be more sensitive to numerical errors (dividing small numbers) so is not recommended. It too is invariant to the "sign". Grader: This approach earns full credit *only if* the student mentions $[\boldsymbol{v}_N]_i \neq 0$.

(c) If we do not know λ_1, then we *may* need two runs of the power method. Here is the effecient process.

(i) Apply the power method to $\boldsymbol{A}$ yielding an eigenvector $\boldsymbol{v}$ corresponding either to $|\lambda_1|$ or $|\lambda_N|$. Initially we do not know which one it is.

(ii) Compute $\lambda \triangleq \boldsymbol{v}'\boldsymbol{A}\boldsymbol{v}$ and look at its sign.

(iii) If $\lambda < 0$, then it must be the case that $\lambda = \lambda_N$ and $|\lambda_1| < |\lambda| = |\lambda_N|$, so $\boldsymbol{v} = \boldsymbol{v}_N$ and we are done.

(iv) If $\lambda \geq 0$ then we know that $\lambda = \lambda_1$ and $-\lambda < \lambda_N < 0$. We then apply the power iteration to $\boldsymbol{B} \triangleq \boldsymbol{A} - \lambda\boldsymbol{I}$, which will have eigenvalues in the interval $[\lambda_N - \lambda_1, 0]$ and the resulting vector will be $\boldsymbol{v}_N$.

Pr. 4. (sol/hsj43)
Given training data $(\boldsymbol{x}_n, y_n), n = 1, \ldots, N$ consisting of pairs of features $\boldsymbol{x}_n \in \mathbb{R}^M$ and responses $y_n \in \mathbb{R}$, we train a linear "artificial neuron" to minimize the average MSE loss by solving the following optimization problem:

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} L(\boldsymbol{w}), \quad L(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^N \|y_n - \boldsymbol{w}'\boldsymbol{x}_n\|_2^2 = \frac{1}{N} \|\boldsymbol{y}' - \boldsymbol{w}'\boldsymbol{X}\|_{\mathrm{F}}^2 = \frac{1}{N} \|\boldsymbol{y} - \boldsymbol{X}'\boldsymbol{w}\|_2^2,$$

where $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \ldots & \boldsymbol{x}_N \end{bmatrix} \in \mathbb{R}^{M \times N}$ and $\boldsymbol{y} = (y_1, \ldots, y_N)' \in \mathbb{R}^N$.
Assuming $\boldsymbol{X}$ has full row rank, the LS solution is simply

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}')^+ \boldsymbol{y} = (\boldsymbol{X}\boldsymbol{X}')^{-1}\boldsymbol{X}\boldsymbol{y} = \boldsymbol{K}_x^{-1}\boldsymbol{K}_{yx}'$$

where $\boldsymbol{K}_x = \frac{1}{N}\sum_{n=1}^N \boldsymbol{x}_n\boldsymbol{x}_n'$ and $\boldsymbol{K}_{yx} = \frac{1}{N}\sum_{n=1}^N y_n\boldsymbol{x}_n'$.
We need $N \geq M$ for there to be any chance that $\boldsymbol{K}_x$ is invertible. Otherwise we need a pseudo-inverse or some form of regularization.

Pr. 5. (sol/hsj72)

(a) We already know that $\boldsymbol{B}'\boldsymbol{B} \succeq \boldsymbol{0}$ so to show $\boldsymbol{B}'\boldsymbol{B} \succ \boldsymbol{0}$ we simply must show that $\boldsymbol{x} \neq \boldsymbol{0} \Rightarrow \boldsymbol{x}'\boldsymbol{B}'\boldsymbol{B}\boldsymbol{x} \neq 0$. Suppose there is a nonzero $\boldsymbol{x}$ such that $\boldsymbol{x}'\boldsymbol{B}'\boldsymbol{B}\boldsymbol{x} = 0$. Then $\|\boldsymbol{B}\boldsymbol{x}\| = 0$ so $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{0}$, but this would contradict the linear independence of the columns of $\boldsymbol{B}$.

(b) $\boldsymbol{A} \succ \boldsymbol{0}$ means that all eigenvalues are positive, hence nonzero, so the matrix is invertible.

(c) $\boldsymbol{x}'(\boldsymbol{A} + \boldsymbol{B})\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{A}\boldsymbol{x} + \boldsymbol{x}'\boldsymbol{B}\boldsymbol{x} \geq 0$ for all $\boldsymbol{x}$.

(d) $\boldsymbol{x}'(\boldsymbol{A} + \boldsymbol{B})\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{A}\boldsymbol{x} + \boldsymbol{x}'\boldsymbol{B}\boldsymbol{x} > 0$ for all $\boldsymbol{x} \neq \boldsymbol{0}$ because $\boldsymbol{A}$ is positive definite.

(e) Because $\boldsymbol{B}$ has full column rank, $\boldsymbol{B}'\boldsymbol{B} \succ \boldsymbol{0}$, so $\boldsymbol{A}'\boldsymbol{A} + \boldsymbol{B}'\boldsymbol{B}$ is positive definite by the previous property and thus invertible by an earlier subproblem.

(f) It suffices to show that $\boldsymbol{A}'\boldsymbol{A} + \boldsymbol{B}'\boldsymbol{B} \succ \boldsymbol{0}$ when $\mathcal{N}(\boldsymbol{A}) \cap \mathcal{N}(\boldsymbol{B}) = \{\boldsymbol{0}\}$. We already have shown that $\boldsymbol{A}'\boldsymbol{A} + \boldsymbol{B}'\boldsymbol{B} \succeq \boldsymbol{0}$ so we just need to verify that $\boldsymbol{x}'(\boldsymbol{A}'\boldsymbol{A} + \boldsymbol{B}'\boldsymbol{B})\boldsymbol{x} \neq 0$ for any $\boldsymbol{x} \neq \boldsymbol{0}$.

Suppose the contrary, *i.e.*, that $\boldsymbol{x}'(\boldsymbol{A}'\boldsymbol{A} + \boldsymbol{B}'\boldsymbol{B})\boldsymbol{x} = 0$ for some $\boldsymbol{x} \neq \boldsymbol{0}$. then it follows that $\|\boldsymbol{A}\boldsymbol{x}\| = 0$ and $\|\boldsymbol{B}\boldsymbol{x}\| = 0$ so $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$ and $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{0}$. But this means that $\boldsymbol{x}$ is in the null space of both $\boldsymbol{A}$ and $\boldsymbol{B}$ and that contradicts the assumption that $\mathcal{N}(\boldsymbol{A}) \cap \mathcal{N}(\boldsymbol{B}) = \{\boldsymbol{0}\}$.

(g) $\boldsymbol{B} \succ \boldsymbol{0}$ implies that $\boldsymbol{B}$ is invertible, so if $\boldsymbol{x}$ is any nonzero vector, then $\boldsymbol{y} = \boldsymbol{B}\boldsymbol{x}$ is nonzero, because otherwise $\boldsymbol{B}$ would be singular.

For any nonzero vector $\boldsymbol{y}$, $\boldsymbol{A} \succ \boldsymbol{0} \Rightarrow \boldsymbol{y}'\boldsymbol{A}\boldsymbol{y} > 0$ by definition of a positive definite matrix.
Thus $\boldsymbol{A} \succ \boldsymbol{0},\ \boldsymbol{B} \succ \boldsymbol{0} \Rightarrow \boldsymbol{x}'\boldsymbol{B}'\boldsymbol{A}\boldsymbol{B}\boldsymbol{x} > 0,\ \forall \boldsymbol{x} \neq \boldsymbol{0} \Rightarrow \boldsymbol{B}\boldsymbol{A}\boldsymbol{B} \succ \boldsymbol{0}$.

Pr. 6. (sol/hs103)

(a) Let $\boldsymbol{b} \triangleq \text{vec}(\boldsymbol{B}) = \begin{bmatrix} \boldsymbol{B}_{[:,1]} \\ \boldsymbol{B}_{[:,2]} \\ \vdots \\ \boldsymbol{B}_{[:,N]} \end{bmatrix} \in \mathbb{F}^{M \times N}$. Similarly let $\boldsymbol{a}_k \triangleq \text{vec}(\boldsymbol{A}_k)$ for $k = 1, \ldots, K$.

For any matrix $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F = \|\text{vec}(\boldsymbol{A})\|_2$, since the squared Frobenius norm is the sum of squares of each element within the matrix, which is equal to the sum of squares of each element in the vec of the matrix. Therefore,

$$\underset{x_1,\ldots,x_K}{\arg\min} \left\| \sum_{k=1}^K x_k \boldsymbol{A}_k - \boldsymbol{B} \right\|_{\text{F}} = \underset{x_1,\ldots,x_K}{\arg\min} \left\| \text{vec}\left(\sum_{k=1}^K x_k \boldsymbol{A}_k - \boldsymbol{B} \right) \right\|_2 = \underset{x_1,\ldots,x_K}{\arg\min} \left\| \sum_{k=1}^K x_k \text{vec}(\boldsymbol{A}_k) - \text{vec}(\boldsymbol{B}) \right\|_2$$
$$= \underset{\boldsymbol{x}}{\arg\min} \| \underbrace{\begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \ldots & \boldsymbol{a}_K \end{bmatrix}}_{\triangleq \widetilde{\boldsymbol{A}}} \boldsymbol{x} - \boldsymbol{b} \|_2.$$

The problem is now reduced to the ordinary least squares formulation, and the solution is given by $\hat{\boldsymbol{x}} = \left(\widetilde{\boldsymbol{A}}\right)^+ \boldsymbol{b}$.

(b) The solution $\hat{\boldsymbol{x}}$ is unique iff $\widetilde{\boldsymbol{A}}$ has full column rank, *i.e.*, the vectors $\{\boldsymbol{a}_k\}$ are linearly independent.

(c) For $K = 3$ one can use `xh = [vec(A1) vec(A2) vec(A3)] \ vec(B)`

(d) (Not graded)
For the general K case, I would store the $\boldsymbol{A}_k$ matrices in an array of matrices like `As = [A1, A2, A3]` and then use `Julia`'s broadcasting and splatting for remarkably concise code:
`xh = hcat(vec.(As)...)  \ vec(B)`

Pr. 7. (sol/hsj31)

(a) By inspection, $\{\boldsymbol{e}_1, \boldsymbol{e}_4\}$ is an orthonormal basis for the null space of $\boldsymbol{Z}$, because $\boldsymbol{Z}\boldsymbol{e}_1 = \boldsymbol{Z} = \boldsymbol{e}_4 = \boldsymbol{0}$ and the set is orthonormal.

(b) By inspection, $\{\boldsymbol{e}_2, \boldsymbol{e}_3\}$ is an orthonormal basis for $\mathcal{N}^\perp(\boldsymbol{Z})$, the orthogonal complement of the null space of $\boldsymbol{Z}$.

(c) The projection of $\boldsymbol{x}$ onto $\mathcal{N}^\perp(\boldsymbol{Z})$ is $\boldsymbol{P}^\perp_{\mathcal{N}(\boldsymbol{Z})}\boldsymbol{x} = \boldsymbol{Q}\boldsymbol{Q}'\boldsymbol{x} = \boldsymbol{Q}\boldsymbol{Q}' \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 3 \\ 0 \end{bmatrix}$, where $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{e}_2 & \boldsymbol{e}_3 \end{bmatrix}$ is an orthormal basis matrix for $\mathcal{N}^\perp(\boldsymbol{Z})$.

(d) (1) Using: `Y = ones(3,3); (U,s,V) = svd(Y); V0=V[:,2:end]` an orthonormal basis for the null space of $\boldsymbol{Y}$ is $\left\{ \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} / \sqrt{2}, \begin{bmatrix} 2 \\ -1 \\ -1 \end{bmatrix} / \sqrt{6} \right\}$.
(Of course none of these bases are unique, so there are many possible correct answers.)

(2) An orthonormal basis for the orthogonal complement of the null space of $\boldsymbol{Y}$ is $\{\boldsymbol{v}_1\}$, where $\boldsymbol{v}_1 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$.

(3) The projection of $\boldsymbol{w}$ onto the orthogonal complement of the null space of $\boldsymbol{Y}$ is $\boldsymbol{P}^\perp_{\mathcal{N}(\boldsymbol{Y})}\boldsymbol{w} = \boldsymbol{v}_1(\boldsymbol{v}_1'\boldsymbol{w}) = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$

(e) A possible `Julia` implementation is

```julia
using LinearAlgebra: svd, rank, Diagonal

"""
    R = orthcompnull(A, X)
```

```
Project each column of `X` onto the orthogonal complement of the null space
of the input matrix `A`.

In:
* `A` `M × N` matrix
* `X` vector of length `N`, or matrix with `N` rows and many columns

Out:
* `R` : vector or matrix of size ??? (you determine this)

For full credit, your solution should be computationally efficient!
"""
function orthcompnull(A, X)

    (_, s, V) = svd(A)
    r = rank(Diagonal(s)) # avoids redundant SVD call

    # orthonormal basis for orthogonal complement of the null space of A:
    Vr = V[:,1:r]

    return Vr * (Vr' * X) # output size is N by the number of columns of X
end
```

The above solution uses a `svd` call. The SVD itself calls a **QR decomposition** that is related to **Gram-Schmidt orthogonalization.** It is likely that an even more efficient solution exists by using the QR decomposition directly to find the basis $\boldsymbol{V}_r$ for the orthogonal complement of the null space of $\boldsymbol{A}$. A QR approach is not required for full credit because we have not covered the QR decomposition.

Grader: solutions must have appropriate parentheses like `Vr * (Vr' * X)` to earn full credit. Deduct 1 point for either `(Vr * Vr') * X` or `Vr * Vr' * X`.
Grader: accept solutions that use QR instead of SVD, as long as no `inv` or `pinv` or other expensive operations are used and as long as the basis matrix is used efficiently with parentheses.
Because $\mathcal{N}^{\perp}(\boldsymbol{A}) = \mathcal{R}(\boldsymbol{A}')$, having a basis $\boldsymbol{V}_r$ for the range space of $\boldsymbol{A}'$ suffices. The following code also passes.

```
function orthcompnull(A,x)
    (Q,~) = qr(A') # QR approach to getting basis for range(A')
    return Q * (Q' * x);
end
```

Pr. 8. (sol/hs087)

(a) A possible `Julia` implementation is

```
"""
    x = lsngd(A, b ; x0 = zeros(size(A,2)), nIters = 200, mu = 0)

Perform Nesterov-accelerated gradient descent to solve the LS problem
``\\argmin_x 0.5 \\| A x - b \\|_2``

In:
- `A` `m × n` matrix
- `b` vector of length `m`

Option:
- `x0` initial starting vector (of length `n`) to use; default 0 vector.
- `nIters` number of iterations to perform; default 200.
- `mu` step size, must satisfy ``0 < \\mu \\leq 1 / \\sigma_1(A)^2``
to guarantee convergence, where ``\\sigma_1(A)`` is the first (largest) singular value.
Ch.5 will explain a default value for `mu`.

Out:
`x` vector of length `n` containing the approximate solution

"""
```

```
function lsngd(A::AbstractMatrix{<:Number}, b::AbstractVector{<:Number} ;
    x0::AbstractVector{<:Number} = zeros(eltype(b), size(A,2)),
    nIters::Int = 200, mu::Real = 0)

    if (mu == 0) # use the following default value:
        mu = 1. / (maximum(sum(abs.(A),dims=1)) * maximum(sum(abs.(A),dims=2)))
    end

    # Nesterov-accelerated gradient descent
    t = 1
    x = x0
    z = x0
    for _ in 1:nIters
        tLast = t
        t = 0.5 * (1 + sqrt(1 + 4 * t^2)) # t update
        xLast = x
        x = z - mu * (A' * (A * z - b)) # x update
        z = x + ((tLast - 1) / t) * (x - xLast) # z update (momentum)
    end

    return x
end
```

(b) Figure 1 compares sequences of $\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}\|$ versus k for standard gradient descent and Nesterov-accelerated gradient descent.
Grader: only the NGD curve is required for this part.

(c) For step sizes $\mu = \{0.25, 0.5, 0.75, 1\}/\sigma_1^2(A)$, clearly Nesterov-accelerated gradient descent converges faster than standard gradient descent, although the convergence is not necessarily monotone.

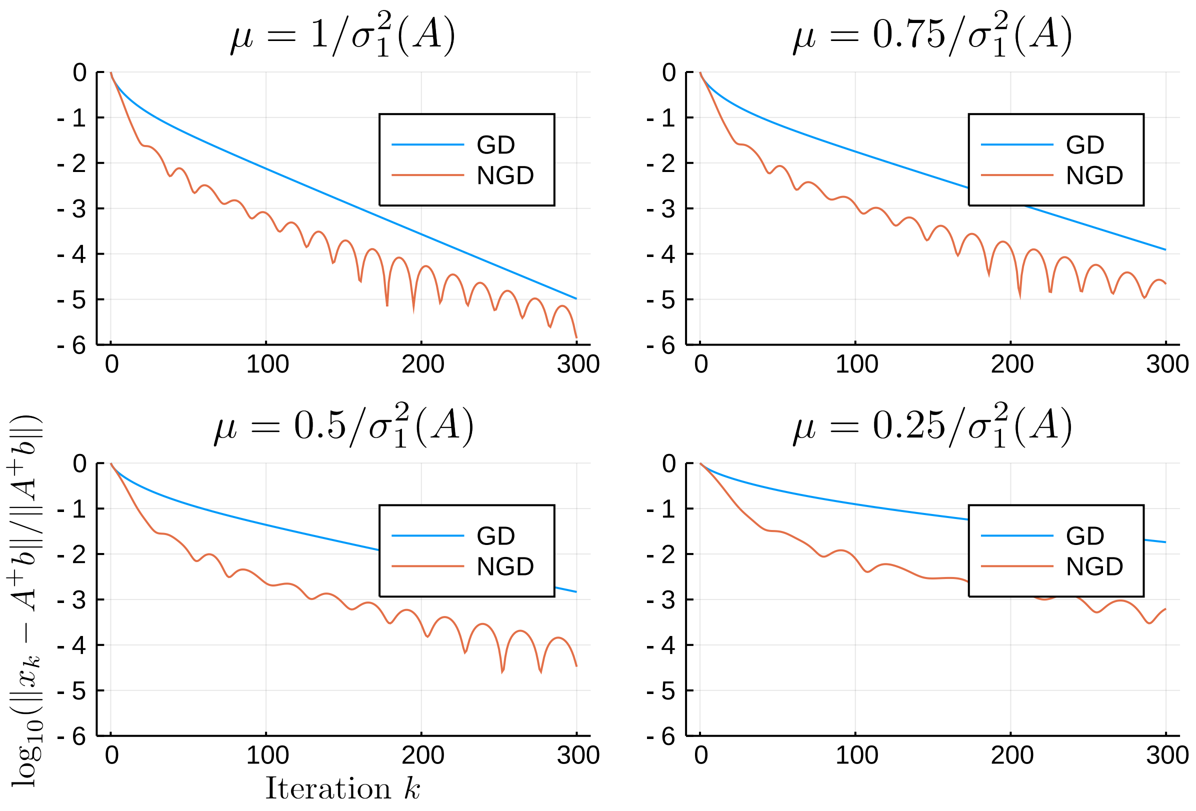


Figure 1: Nesterov-accelerated vs. standard gradient descent for a least squares problem, for four values of step size μ.

Pr. 10. (sol/hsx02)
This will be graded on Canvas, not gradescope. Thank for your submission. These practice problems will be a valuable resource for this class and future classes.

Non-graded problem(s) below

Pr. 11. (sol/hsj03)
If $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{T}$, then $\boldsymbol{T}\boldsymbol{x} = \lambda \boldsymbol{x}$ 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. 12. (sol/hsj5a)
For $\boldsymbol{A} = \begin{bmatrix}1\\1\\1\end{bmatrix}$, an orthonormal basis for $\mathcal{R}(\boldsymbol{A})$ is simply $\boldsymbol{U}_1 = \begin{bmatrix}1\\1\\1\end{bmatrix}/\sqrt{3}$.
For $\boldsymbol{B} = \begin{bmatrix}2 & 0\\2 & 3\\0 & 0\end{bmatrix}$, an orthonormal basis for $\mathcal{R}(\boldsymbol{B})$ is simply $\boldsymbol{U}_2 = \begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}$.
Thus $\cos\theta = \|\boldsymbol{U}_1'\boldsymbol{U}_2\|_2 = \left\|\begin{bmatrix}1 & 1\end{bmatrix}/\sqrt{3}\right\|_2 = \sqrt{2/3}$.

Pr. 13. (sol/hs032)
Here we have $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}' \in \mathbb{R}^{n\times n}$.

(a) Let $\boldsymbol{v}_i$ denote an eigenvector of $\boldsymbol{B}$ associated with the eigenvalue λ_i for $i = 1 \ldots n$. Then by definition we have:
$$\boldsymbol{B}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}.$$
Substituting $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}'$ into the above expression yields:
$$(\boldsymbol{A} + \boldsymbol{x}\boldsymbol{x}')\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i.$$
Rearranging terms (to help solve for $\boldsymbol{v}_i$) yields:
$$(\boldsymbol{A} - \lambda_i \boldsymbol{I})\boldsymbol{v}_i = -\boldsymbol{x}\boldsymbol{x}'\boldsymbol{v}_i.$$
Noting that $c_i \triangleq -\boldsymbol{x}'\boldsymbol{v}_i$ 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 all eigenvectors corresponding to λ_i satisfy
$$\boldsymbol{v}_i \propto (\boldsymbol{A} - \lambda_i \boldsymbol{I})^{-1}\boldsymbol{x},$$
expressing eigenvectors of $\boldsymbol{B}$ in terms of $\boldsymbol{A}$, $\boldsymbol{x}$ and λ_i.
A key part of the derivation is recognizing that $c_i = -\boldsymbol{x}'\boldsymbol{v}_i$ 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 the above expression provides the complete solution.

(b) To prove orthogonality:
$$\begin{aligned}\boldsymbol{v}_j'\boldsymbol{v}_k \propto \boldsymbol{x}'(\boldsymbol{A} - \lambda_j \boldsymbol{I})^{-1}(\boldsymbol{A} - \lambda_k \boldsymbol{I})^{-1}\boldsymbol{x} &= \sum_{i=1}^n \frac{x_i^2}{(\lambda_j - a_{ii})(\lambda_k - a_{jj})}\\ &= \sum_{i=1}^n \frac{1}{(\lambda_k - \lambda_j)}\left(\frac{x_i^2}{\lambda_j - a_{ii}} - \frac{x_i^2}{\lambda_k - a_{ii}}\right)\\ &= \frac{1}{(\lambda_k - \lambda_j)}\left(\sum_{i=1}^n \frac{x_i^2}{\lambda_j - a_{ii}} - \sum_{i=1}^n \frac{x_i^2}{\lambda_k - a_{ii}}\right). \qquad (1)\end{aligned}$$
But from a previous HW problem, for every eigenvalue λ of $\boldsymbol{B}$, $\sum_{i=1}^n \frac{x_i^2}{\lambda - a_{ii}} = 1$.

Thus (1) becomes $\frac{1}{(\lambda_k - \lambda_j)}(1 - 1) = 0$, showing that the eigenvectors are orthogonal.

Pr. 14. (sol/hs031)
Given $\boldsymbol{A} = \boldsymbol{x}\boldsymbol{x}' + \boldsymbol{y}\boldsymbol{y}'$.
The rank of $\boldsymbol{A}$ is at most two. It is equal to zero when $\boldsymbol{x}\boldsymbol{x}' = -\boldsymbol{y}\boldsymbol{y}'$ 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 a linearly independent pair 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:

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

The nonzero eigenvalues of $\boldsymbol{Z}\boldsymbol{Z}'$ are the same as those of $\boldsymbol{M} \triangleq \boldsymbol{Z}'\boldsymbol{Z} = \begin{bmatrix}\|\boldsymbol{x}\|_2^2 & \rho \\ \rho & \|\boldsymbol{y}\|_2^2\end{bmatrix}$, 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 $\boldsymbol{A}\boldsymbol{v} = (\boldsymbol{x}'\boldsymbol{v})\,\boldsymbol{x} + (\boldsymbol{y}'\boldsymbol{v})\,\boldsymbol{y} = \lambda\boldsymbol{v}$. 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 $\boldsymbol{\beta} \in \mathbb{F}^2$. Working with the eigenvalue/eigenvector definition we have that:

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

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

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

or equivalently, that $\boldsymbol{\beta} \in \mathbb{F}^2$ is an *eigenvector* of $\boldsymbol{M}$ corresponding to the eigenvalue λ.
Often we prefer to work with unit-norm eigenvectors of $\boldsymbol{Z}\boldsymbol{Z}'$, so we normalize them. Thus the unit-norm eigenvectors of $\boldsymbol{Z}\boldsymbol{Z}'$, which are exactly equal to the eigenvectors of $\boldsymbol{A}$, are simply (to within $\mathrm{e}^{\imath\phi}$ factors):

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

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. 15. (sol/hsj42)

(a) $\boldsymbol{A}' = \boldsymbol{V}_r\boldsymbol{\Sigma}_r\boldsymbol{U}_r' \Rightarrow \mathcal{N}(\boldsymbol{A}') = \mathcal{R}(\boldsymbol{U}_0) \Rightarrow \mathcal{N}^\perp(\boldsymbol{A}') = \mathcal{R}(\boldsymbol{U}_r)$

(b) $\boldsymbol{A}' = \boldsymbol{V}_r\boldsymbol{\Sigma}_r\boldsymbol{U}_r' \Rightarrow \mathcal{R}(\boldsymbol{A}') = \mathcal{R}(\boldsymbol{V}_r)$

(c) $\boldsymbol{A}^+ = \boldsymbol{V}_r\boldsymbol{\Sigma}_r^{-1}\boldsymbol{U}_r' \Rightarrow \boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{U}_r\boldsymbol{U}_r' \Rightarrow \mathcal{R}(\boldsymbol{A}\boldsymbol{A}^+) = \mathcal{R}(\boldsymbol{U}_r) \Rightarrow \mathcal{R}^\perp(\boldsymbol{A}\boldsymbol{A}^+) = \mathcal{R}(\boldsymbol{U}_0)$

(d) $\boldsymbol{A}^+ = \boldsymbol{V}_r\boldsymbol{\Sigma}_r^{-1}\boldsymbol{U}_r' \Rightarrow \mathcal{N}(\boldsymbol{A}^+) = \mathcal{R}(\boldsymbol{U}_0)$

(e) $\boldsymbol{A}^+ = \boldsymbol{V}_r\boldsymbol{\Sigma}_r^{-1}\boldsymbol{U}_r' \Rightarrow \boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{V}_r\boldsymbol{V}_r' \Rightarrow \mathcal{R}(\boldsymbol{A}^+\boldsymbol{A}) = \mathcal{R}(\boldsymbol{V}_r)$
