Lecture 13, Oct 20, 2023

Additional Linear Algebra Topics

Positive Definiteness

Definition

A matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ is *positive semidefinite* if

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} \geq 0$$

$\boldsymbol{B}$ is *positive definite* if

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{x} \neq \boldsymbol{0} \implies \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} > 0$$

- $\boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x}$ is referred to as the *quadratic form*, which is the matrix version of x^2
- For any $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\boldsymbol{A}^T \boldsymbol{A}$ is positive semi-definite; $\boldsymbol{A}^T \boldsymbol{A}$ is positive definite if and only if $\boldsymbol{A}$ is full rank
  - $\boldsymbol{x}^T (\boldsymbol{A}^T \boldsymbol{A}) \boldsymbol{x} = (\boldsymbol{A}\boldsymbol{x})^T (\boldsymbol{A}\boldsymbol{x}) = |\boldsymbol{A}\boldsymbol{x}|_2^2 \geq 0$
  - If $\boldsymbol{A}$ has linearly independent columns, then $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0} \implies \boldsymbol{x} = \boldsymbol{0}$, so $\boldsymbol{x}^T (\boldsymbol{A}^T \boldsymbol{A}) \boldsymbol{x} = (\boldsymbol{A}\boldsymbol{x})^T (\boldsymbol{A}\boldsymbol{x}) = \boldsymbol{0}$ only when $\boldsymbol{x} = \boldsymbol{0}$; this goes both ways
- The eigenvalues of a positive semi-definite matrix are always greater than or equal to zero; for positive definite matrices all eigenvalues are strictly positive
- Positive definite matrices come up often as inertia matrices or covariance matrices

Orthogonality

Definition

A set of vectors $\{ \boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n \}$ is *orthonormal* if and only if

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

That is, each vector has norm 1 and is orthogonal to every other vector.

A square matrix whose columns are orthonormal is called an *orthogonal* matrix.

- Let $\boldsymbol{Q}$ be orthogonal, then:
  - $\boldsymbol{Q}^T \boldsymbol{Q} = \boldsymbol{1}$, and so $\boldsymbol{Q}^T = \boldsymbol{Q}^{-1}$
  - Applying $\boldsymbol{Q}$ has a linear transformation will not affect the length of a vector or the angle between two vectors; this means $\boldsymbol{Q}$ is an *isometry*
    - $\|\boldsymbol{Q}\boldsymbol{x}\|_2^2 = \boldsymbol{x}^T \boldsymbol{Q}^T \boldsymbol{Q} \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{x} = \|\boldsymbol{x}\|_2^2$
    - $(\boldsymbol{Q}\boldsymbol{x}) \cdot (\boldsymbol{Q}\boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{Q}^T \boldsymbol{Q} \boldsymbol{y} = \boldsymbol{x}^T \boldsymbol{y} = \boldsymbol{x} \cdot \boldsymbol{y}$
- An orthogonal matrix can rotate vectors but not scale them; all rotation matrices are orthogonal

Solving Linear Systems

- Solving systems in the form of $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is a common problem
- However solving by $\boldsymbol{A}^{-1}\boldsymbol{b}$ is almost never a good idea since $\boldsymbol{A}^{-1}$ can be expensive to compute, reduces solution accuracy, and is less efficient since a sparse $\boldsymbol{A}$ will have a dense $\boldsymbol{A}^{-1}$
- Gaussian elimination works for any $\boldsymbol{A}$ and $\boldsymbol{b}$, but we can only achieve $O(n^3)$ for $\boldsymbol{A} \in \mathbb{R}^{n \times n}$; to get better performance, we can exploit the structure of a matrix (e.g. sparse/dense, triangular, Hermitian, etc)
  - Simplest case: $\boldsymbol{A}$ diagonal, which we can solve in $O(n)$

    - If $\boldsymbol{A}$ is upper or lower triangular, we can solve in $O(n^2)$; we can use each row to solve for exactly a single element of $\boldsymbol{x}$
    - For a sparse matrix, if we can split it up into blocks, we can solve for each block individually
- Gaussian elimination is equivalent to first factorizing $\boldsymbol{Ax} = \boldsymbol{LUx} = \boldsymbol{b}$ and then solving $\boldsymbol{Ly} = \boldsymbol{b}, \boldsymbol{Ux} = \boldsymbol{y}$, where $\boldsymbol{L}$ is lower triangular and $\boldsymbol{U}$ is upper triangular
    - If we want to reuse $\boldsymbol{A}$ with different values of $\boldsymbol{b}$, we can prefactorize $\boldsymbol{A}$ and reuse the factors to save time
- In practice, use `x = np.linalg.solve(A, b)` in Python or `x = A \ b` in MATLAB
- Example: solve $\begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
    - We could do this by Gaussian elimination:
        * $\begin{bmatrix} 1 & 2 & 1 \\ 2 & 5 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 2 & 1 \\ 0 & 1 & -1 \end{bmatrix} \to \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$
        * $x_2 = -1 \implies x_1 = 3$
    - Note the first row operation was $\left(I - 2 \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \right) \boldsymbol{A} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$
        * $\begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$
        * Notice the matrix multiplying $\boldsymbol{A}$ is lower triangular and the result is upper triangular; now we can invert the first matrix to get a form of $\boldsymbol{A} = \boldsymbol{LU}$, since the inverse of a lower triangular matrix is also lower triangular
    - LU factorization:
        * $\begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$
        * Solve first $\begin{bmatrix} 1 \\ 0 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ \implies & y \end{bmatrix}_1 = 1, y_2 = -1$
        * Now we can solve $\begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \boldsymbol{y} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, which gives us the same result

Matrix and Vector Norms

Definition

A general *vector norm* is any function $\|\cdot\| : \mathbb{R}^n \mapsto [0, \infty)$ which satisfies the following conditions:

1. $\|\boldsymbol{x}\| = 0 \iff \boldsymbol{x} = \boldsymbol{0}$
2. $\forall c \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^n, \|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$
3. $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$

- $\|\boldsymbol{x}\| \geq 0$ follows from these conditions
- As with the Euclidean norm, norms encode some notion of "length"
- Typical vector norms:
    - The p-norm for $p \geq 1$ is defined as $\|\boldsymbol{x}\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$
        * The 2-norm, or Euclidean norm, is an example of a p-norm
        * If we constrain $\|\boldsymbol{x}\|_p = 1$, we get boxes of various shapes; e.g. a 1-norm is a rotated square in 2D, 2-norm is a circle in 2D, and infinity norm is a square in 2D; all other norms are somewhere in between
    - The ∞-norm (infinity norm) is defined as $\|\boldsymbol{x}\|_\infty = \max(|x_1|, |x_2|, \ldots, |x_n|)$
- All p-norms for $p \geq 1$ (including the ∞-norm) are convex

Definition

The *matrix norm* on $\mathbb{R}^{m\times n}$ induced by a vector norm $\|\cdot\|$ is given by

$$\|\boldsymbol{A}\| = \max\left\{\, \|\boldsymbol{Ax}\| \mid \|\boldsymbol{x}\| = 1 \,\right\}$$

Or equivalently

$$\|\boldsymbol{A}\| = \max_{\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{x}\neq 0} \frac{\|\boldsymbol{Ax}\|}{\|\boldsymbol{x}\|}$$

- The induced vector norm is essentially the maximum norm of a unit vector after multiplying by $\boldsymbol{A}$
- This makes the property that $\|\boldsymbol{Ax}\| \leq \|\boldsymbol{A}\|\|\boldsymbol{x}\|$
- Typical matrix norms:
  - 1-norm: $\|\boldsymbol{A}\|_1 = \max_{i\leq j\leq n} \sum_{i=1}^{m} |a_{ij}|$
    * This is equivalent to the maximum column sum
  - 2-norm: $\|\boldsymbol{A}\|_2 = \max\left\{ \sqrt{\lambda} \,\middle|\, \exists \boldsymbol{x} \in \mathbb{R}^n \text{ s.t. } \boldsymbol{A}^T\boldsymbol{Ax} = \lambda\boldsymbol{x} \right\}$
    * This is the square root of the largest eigenvalue of $\boldsymbol{A}^T\boldsymbol{A}$ (intuitively we can interpret this as the largest eigenvalue of $\boldsymbol{A}$)
    * Sometimes called the *spectral radius* of $\boldsymbol{A}$
  - Frobenius norm: $\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\operatorname{tr} \boldsymbol{A}^T\boldsymbol{A}}$
    * $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_F$ always holds
  - ∞-norm (infinity norm): $\|\boldsymbol{A}\|_\infty = \max_{1\leq i\leq m} \sum_{j=1}^{n} |a_{ij}|$
    * This is the maximum row sum
    * Proof:
      - $\|\boldsymbol{A}\|_\infty = \max\left\{\, \|\boldsymbol{Ax}\|_\infty \mid \|\boldsymbol{x}\|_\infty = 1 \,\right\}$
      - Note $[\boldsymbol{Ax}]_i = \sum_j a_{ij}x_j$, so $\|\boldsymbol{Ax}\|_\infty = \max_i \left| \sum_j a_{ij}x_j \right| \leq \max_i \sum_j |a_{ij}||x_j| \leq \max_i \sum_j |a_{ij}| x_{max} = \max_i \sum_j |a_{ij}| \|\boldsymbol{x}\|_\infty = \max_i \sum_j |a_{ij}|$
      - We can show the bound the other way by selecting x_j in a special way (see posted notes)

Condition Number

Definition

The *condition number* of $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ with respect to a given norm $\|\cdot\|$ is

$$\operatorname{cond} \boldsymbol{A} = \|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|$$

if $\boldsymbol{A}$ is non-invertible, $\operatorname{cond} \boldsymbol{A} = \infty$ by definition.

- Recall that conditioning describes how a small error in the input propagates to an error in the output
- For a matrix, we ask the question: for finding $\boldsymbol{x}$ such that $\boldsymbol{Ax} = \boldsymbol{b}$, how does the solution $\boldsymbol{x}$ change if we make a small change to the matrices $\boldsymbol{A}$ and $\boldsymbol{b}$?
  - We can derive the relative condition number to be $|\varepsilon|\|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|$, where ε is some input error
- e.g. if we used the 2-norm, we would essentially get the ratio between the largest eigenvalue and the smallest eigenvalue; intuitively if these two eigenvalues are different, we will see more error since the system is stiff