4. Matrices

- matrix notation
- matrix operations
- complexity
- examples of matrices
- graphs
- convolution

Matrix

a *matrix* is a rectangular array of elements written as

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \dots & \vdots \\ A_{m1} & A_{m2} & \dots & A_{mn} \end{bmatrix}$$

- scalars in array are the *elements* (*entries*, *coefficients*, *components*)
- A_{ij} is the i, j element of A (i is row index, j is column index)
- *size* (*dimensions*) of the matrix is $m \times n$ = (#rows) $\times$ (#columns)

Example

$$A = \begin{bmatrix} 0 & 1 & -2.3 & 0.1 \\ 1.3 & 4 & -0.1 & 0 \\ 4.1 & -1 & 0 & 1.7 \end{bmatrix}$$

- $A_{23} = -0.1$
- a 3×4 matrix

Notes and conventions

Notes

- a matrix of size $m \times n$ is called an $m \times n$-matrix
- $\mathbb{R}^{m \times n}$ is set of $m \times n$ matrices with real elements
- we use $A_{i,j}$ when i or j are more than one digit
- two matrices with same size are equal if corresponding entries are all equal
- sometimes A_k is a matrix; in this case, we use $(A_k)_{ij}$ to denote its i, j element

Conventions

- matrices are typically denoted by capital letters
- parentheses are also used instead of rectangular brackets to represent a matrix
- often a_{ij} is used to denote the i, j element of A
- some authors use bold capital letter for matrices (*e.g.*, $\mathbf{A}, \boldsymbol{A}$)
- be prepared to figure out whether a symbol represents a matrix, vector, or a scalar

Matrix shapes

Scalar: a 1×1 matrix is a scalar

Row and column vectors

- a $1 \times n$ matrix is called a row vector
- an $n \times 1$ matrix is called a column vector (or just vector)

Tall, wide, square matrices: an $m \times n$ matrix is

- tall, skinny, or thin if $m > n$
- wide or fat if $m < n$
- square if $m = n$

Columns and rows

an $m \times n$ matrix can be viewed as a matrix with row/column vectors

Columns representation

$$A = [a_1 \ a_2 \ \cdots \ a_n] \qquad a_j = \begin{bmatrix} A_{1j} \\ \vdots \\ A_{mj} \end{bmatrix}$$

each a_j is an m-vector (the jth column of A)

Rows representation

$$A = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \qquad b_i = [A_{i1} \ \cdots \ A_{in}]$$

each b_i is a $1 \times n$ row vector (the ith row of A)

Block matrix and submatrices

- a *block* matrix is a rectangular array of matrices
- elements in the array are the *blocks* or *submatrices* of the block matrix

Example: a 2×2 block matrix

$$A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}$$

- submatrices can be referred to by their block row and column (C is $1, 2$ block of A)
- dimensions of the blocks must be compatible
- if the blocks are

$$B = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 2 & 3 \\ 5 & 4 & 7 \end{bmatrix}, \quad D = \begin{bmatrix} 1 \end{bmatrix}, \quad E = \begin{bmatrix} -1 & 6 & 0 \end{bmatrix}$$

then

$$A = \begin{bmatrix} 2 & 0 & 2 & 3 \\ 1 & 5 & 4 & 7 \\ 1 & -1 & 6 & 0 \end{bmatrix}$$

Slice of matrix

$$A_{p:q,r:s} = \begin{bmatrix} A_{pr} & A_{p,r+1} & \cdots & A_{ps} \\ A_{p+1,r} & A_{p+1,r+1} & \cdots & A_{p+1,s} \\ \vdots & \vdots & & \vdots \\ A_{qr} & A_{q,r+1} & \cdots & A_{qs} \end{bmatrix}$$

- an $(q - p + 1) \times (s - r + 1)$ matrix
- obtained by extracting from A elements in rows p to q and columns r to s
- from last page example, we have

$$A_{2:3,3:4} = \begin{bmatrix} 4 & 7 \\ 6 & 0 \end{bmatrix}$$

Special matrices

Zero matrix

- matrix with $A_{ij} = 0$ for all i, j
- notation: 0 or $0_{m \times n}$ (if dimension is not clear from context)

Identity matrix

- square matrix with $A_{ij} = 1$ if $i = j$ and $A_{ij} = 0$ if $i \neq j$
- notation: I or I_n (if dimension is not clear from context)
- columns of I_n are unit vectors $e_1, e_2, \ldots, e_n$; for example,

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

Structured matrices

matrices with special patterns or structure arise in many applications

Diagonal matrix

- square with $A_{ij} = 0$ for $i \neq j$
- represented as $A = \mathrm{diag}(a_1, \ldots, a_n)$ where a_i are diagonal elements

$$\mathrm{diag}(0.2, -3, 1.2) = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 1.2 \end{bmatrix}$$

Lower triangular matrix: square with $A_{ij} = 0$ for $i < j$

$$\begin{bmatrix} 4 & 0 & 0 \\ 3 & -1 & 0 \\ -1 & 5 & -2 \end{bmatrix}, \quad \begin{bmatrix} 4 & 0 & 0 \\ 0 & -1 & 0 \\ -1 & 0 & -2 \end{bmatrix}$$

Upper triangular matrix: square with $A_{ij} = 0$ for $i > j$

(a triangular matrix is **unit** upper/lower triangular if $A_{ii} = 1$ for all i)

Sparse matrices

a matrix A is *sparse* if most (almost all) of its elements are zero

- $\mathbf{nnz}(A)$ is number of nonzero elements (typically order n or less)
- *density* is $\mathbf{nnz}(A)/(mn) \leq 1$
- densities of sparse matrices that arise in practice are typically small (*e.g.*, 10^{-2})
- can be stored and manipulated efficiently on a computer
- for example the triplet format:

$$\begin{array}{cr} (1,1) & 2.4000 \\ (1,2) & -3.0000 \\ (3,2) & 2.0000 \\ (2,3) & 1.3000 \\ (3,3) & -6.0000 \end{array}$$

  which means $A_{11} = 2.4$, $A_{3,2} = 2$, ...

Transpose of a matrix

transpose of an $m \times n$ matrix A is the $n \times m$ matrix:

$$A^T = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{m1} \\ A_{12} & A_{22} & \cdots & A_{m2} \\ \vdots & \vdots & & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{mn} \end{bmatrix}$$

- $(A^T)^T = A$
- the transpose of a block matrix (shown for a 2×2 block matrix)

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^T = \begin{bmatrix} A^T & C^T \\ B^T & D^T \end{bmatrix}$$

  - A, B, C, and D are matrices with compatible sizes
  - concept holds for any number of blocks

Symmetric matrices

a square matrix is *symmetric* if

$$A = A^T$$

- $A_{ij} = A_{ji}$
- examples

$$\begin{bmatrix} 4 & 3 & -2 \\ 3 & -1 & 5 \\ -2 & 5 & 0 \end{bmatrix}, \quad \begin{bmatrix} 4+3j & 3-2j & 0 \\ 3-2j & -j & -2j \\ 0 & -2j & 3 \end{bmatrix}$$

Outline

- matrix notation
- **matrix operations**
- complexity
- examples of matrices
- graphs
- convolution

Matrix addition

sum of two $m \times n$ matrices A and B

$$A + B = \begin{bmatrix} A_{11} + B_{11} & A_{12} + B_{12} & \cdots & A_{1n} + B_{1n} \\ A_{21} + B_{21} & A_{22} + B_{22} & \cdots & A_{2n} + B_{2n} \\ \vdots & \vdots & & \vdots \\ A_{m1} + B_{m1} & A_{m2} + B_{m2} & \cdots & A_{mn} + B_{mn} \end{bmatrix}$$

Properties

- *commutativity:* $A + B = B + A$
- *associativity:* $(A + B) + C = A + (B + C)$
- *addition with zero matrix:* $A + 0 = 0 + A = A$
- *transpose of sum:* $(A + B)^T = A^T + B^T$

Scalar-matrix multiplication

scalar-matrix product of $m \times n$ matrix A with scalar β

$$\beta A = \begin{bmatrix} \beta A_{11} & \beta A_{12} & \cdots & \beta A_{1n} \\ \beta A_{21} & \beta A_{22} & \cdots & \beta A_{2n} \\ \vdots & \vdots & & \vdots \\ \beta A_{m1} & \beta A_{m2} & \cdots & \beta A_{mn} \end{bmatrix}$$

Properties: for matrices A, B, scalars β, γ

- associativity: $(\beta\gamma)A = \beta(\gamma A)$
- *distributivity:* $(\beta + \gamma)A = \beta A + \gamma A$ and $\beta(A + B) = \beta A + \beta B$
- *transposition:* $(\beta A)^T = \beta A^T$

Matrix-vector product

product of $m \times n$ matrix A with n-vector x

$$Ax = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n \\ A_{21}x_1 + A_{22}x_2 + \cdots + A_{2n}x_n \\ \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n \end{bmatrix} = \begin{bmatrix} b_1^T x \\ \vdots \\ b_m^T x \end{bmatrix}$$

- b_i^T is ith row of A
- dimensions must be compatible (number of columns of A equals the size of x)
- Ax is a linear combination of the columns of A:

$$Ax = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x_1 a_1 + x_2 a_2 + \cdots + x_n a_n$$

each a_i is an m-vector (ith column of A)

Properties of matrix-vector multiplication

for matrices A, B, vectors u, v and scalar β

- *associativity:* $(\beta A)u = A(\beta u) = \beta(Au)$ (we write βAu)
- *distributivity:* $A(u+v) = Au + Av$ and $(A+B)u = Au + Bu$
- *transposition:* $(Au)^T = u^T A^T$

General examples

- $0x = 0$, *i.e.*, multiplying by zero matrix gives zero
- $Ix = x$, *i.e.*, multiplying by identity matrix does nothing
- inner product a^Tb is matrix-vector product of $1 \times n$ matrix a^T and n-vector b
- $Ae_j = a_j$, the jth column of A [$(A^Te_i)^T = e_i^TA$ is ith row]
- the product $A\mathbf{1}$ is the sum of the columns of A
- for the $n \times n$ matrix

$$A = \begin{bmatrix} 1-1/n & -1/n & \cdots & -1/n \\ -1/n & 1-1/n & \cdots & -1/n \\ \vdots & & \cdots & \vdots \\ -1/n & -1/n & \cdots & 1-1/n \end{bmatrix},$$

  $\tilde{x} = Ax$ is de-meaned version of x

Difference matrix

$(n-1) \times n$ *difference matrix* is

$$D = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 & 0 \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ 0 & 0 & 0 & \cdots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}$$

$y = Dx$ is $(n-1)$-vector of differences of consecutive entries of x:

$$Dx = \begin{bmatrix} x_2 - x_1 \\ x_3 - x_2 \\ \vdots \\ x_n - x_{n-1} \end{bmatrix}$$

Running sum matrix

the $n \times n$ matrix

$$S = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ & & \ddots & \ddots & & \\ & & & \ddots & \ddots & \\ 1 & 1 & 1 & \cdots & 1 & 0 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{bmatrix}$$

is called the *running sum matrix*

the ith entry of the n-vector Sx is the sum of the first i entries of x:

$$Sx = \begin{bmatrix} x_1 \\ x_1 + x_2 \\ x_1 + x_2 + x_3 \\ \vdots \\ x_1 + \cdots + x_n \end{bmatrix}$$

Selectors

an $m \times n$ *selector matrix*: each row is a unit vector (transposed)

$$A = \begin{bmatrix} e_{k_1}^T \\ \vdots \\ e_{k_m}^T \end{bmatrix}$$

- $k_1, \ldots, k_m$ are integers in range $1, \ldots, n$
- Ax copies the k_ith entry of x into the ith entry:

$$Ax = (x_{k_1}, x_{k_2}, \ldots, x_{k_m})$$

Reverser matrix

$$A = \begin{bmatrix} e_n^T \\ \vdots \\ e_1^T \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}, \quad Ax = \begin{bmatrix} x_n \\ x_{n-1} \\ \vdots \\ x_2 \\ x_1 \end{bmatrix}$$

Circular shift matrix

$$A = \begin{bmatrix} e_n^T \\ e_1^T \\ \vdots \\ e_{n-1}^T \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \quad Ax = \begin{bmatrix} x_n \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix}$$

Down-sampling: the $m \times 2m$ matrix

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \quad Ax = \begin{bmatrix} x_1 \\ x_3 \\ \vdots \\ x_{2m-1} \end{bmatrix}$$

'down-samples' x by 2

Permutation matrices

- an $n \times n$ *permutation matrix* has exactly one entry of each row/column is one
- let $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ be a *permutation* (reordering) of $(1, 2, \ldots, n)$
- we associate with π the $n \times n$ permutation matrix A

$$A_{i\pi_i} = 1, \quad A_{ij} = 0 \quad \text{if} \quad j \neq \pi_i$$

- Ax is a permutation of the elements of x: $Ax = (x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n})$
- example: for permutation $\pi = (3, 1, 2)$, the associated permutation matrix is

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

  multiplying a 3-vector by A re-orders its entries: $Ax = (x_3, x_1, x_2)$

Matrix multiplication

product of $m \times n$ matrix A and $n \times p$ matrix B

$$C = AB$$

is the $m \times p$ matrix with i, j element

$$C_{ij} = A_{i1}B_{1j} + A_{i2}B_{2j} + \cdots + A_{in}B_{nj}$$

- to get C_{ij} : move along ith row of A, jth column of B
- dimensions must be compatible:

$$\#\text{columns in } A = \#\text{rows in } B$$

- example:

$$\begin{bmatrix} -1.5 & 3 & 2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 \\ 0 & -2 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 3.5 & -4.5 \\ -1 & 1 \end{bmatrix}$$

Special cases of matrix multiplication

- scalar-vector product (with scalar on right!) $x\alpha$
- inner product $a^T b$
- matrix-vector multiplication Ax
- outer product of m-vector a and n-vector b

$$ab^T = \begin{bmatrix} a_1 b_1 & a_1 b_2 & \cdots & a_1 b_n \\ a_2 b_1 & a_2 b_2 & \cdots & a_2 b_n \\ \vdots & \vdots & & \vdots \\ a_m b_1 & a_m b_2 & \cdots & a_m b_n \end{bmatrix}$$

- multiplication by identity $AI = A$ and $IA = A$
- matrix power: multiplication of matrix with itself p times: $A^p = AA \cdots A$

Properties of matrix-matrix product

- associativity: $(AB)C = A(BC)$, so we write ABC
- associativity: with scalar multiplication: $(\gamma A)B = \gamma(AB) = \gamma AB$
- distributivity with sum:

$$A(B+C) = AB + AC, \quad (A+B)C = AC + BC$$

- transpose of product:

$$(AB)^T = B^T A^T$$

- **not** commutative: $AB \neq BA$ in general; for example,

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \neq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

  there are exceptions, e.g., $AI = IA$ for square A

Product of block matrices

block-matrices can be multiplied as regular matrices

Example: product of two 2×2 block matrices

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} W & Y \\ X & Z \end{bmatrix} = \begin{bmatrix} AW + BX & AY + BZ \\ CW + DX & CY + DZ \end{bmatrix}$$

if the dimensions of the blocks are compatible

Column and row representations

Column representation

- A is $m \times p$, B is $p \times n$ with columns b_i

$$AB = A\left[\begin{array}{cccc} b_1 & b_2 & \cdots & b_n \end{array}\right] = \left[\begin{array}{cccc} Ab_1 & Ab_2 & \cdots & Ab_n \end{array}\right]$$

- so AB is 'batch' multiply of A times columns of B

Row representation

- with a_i^T the rows of A

$$AB = \left[\begin{array}{c} a_1^T B \\ a_2^T B \\ \vdots \\ a_m^T B \end{array}\right] = \left[\begin{array}{c} (B^T a_1)^T \\ (B^T a_2)^T \\ \vdots \\ (B^T a_m)^T \end{array}\right]$$

- row i is $(B^T a_i)^T$

Inner and outer product representations

Inner product representation

- A is $m \times p$ with rows a_i^T, B is $p \times n$ with columns b_i

$$AB = \begin{bmatrix} a_1^T b_1 & a_1^T b_2 & \cdots & a_1^T b_n \\ a_2^T b_1 & a_2^T b_2 & \cdots & a_2^T b_n \\ \vdots & \vdots & & \vdots \\ a_m^T b_1 & a_m^T b_2 & \cdots & a_m^T b_n \end{bmatrix}$$

- entry ij is $a_i^T b_j$

Outer product representation

- a_i columns of A, b_i^T rows of B
- then we can express the product matrix AB as a sum of outer products:

$$AB = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \begin{bmatrix} b_1^T \\ \vdots \\ b_n^T \end{bmatrix} = a_1 b_1^T + \cdots + a_n b_n^T$$

Frobenius norm

the *Frobenius norm* of an $m \times n$ matrix A is

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

- agrees with vector norm when $n = 1$
- in MATLAB: `norm(A,'fro')`
- distance between two matrices: $\|A - B\|_F$
- satisfies norm properties:
  - $\|\alpha A\| = |\alpha| \|A\|$
  - $\|A + B\| \leq \|A\| + \|B\|$
  - $\|A\| \geq 0$
  - $\|A\| = 0$ only if $A = 0$
- additional properties:
  - $\|A\|_F = \|A^T\|_F = \sqrt{\|a_1\|^2 + \cdots + \|a_n\|^2}$, a_j is jth column of A
  - $\|AB\|_F \leq \|A\|_F \|B\|_F$

Outline

- matrix notation
- matrix operations
- **complexity**
- examples of matrices
- graphs
- convolution

Complexity of matrix operations

Addition and scalar multiplication

- addition $A + B$ requires mn flops (for $m \times n$ matrices)
- scalar multiplication requires requires mn
- less for sparse matrices
- transpose requires zero flops

Matrix-vector multiplication (for n-vector x and $m \times n$ matrix A)

- $y = Ax$ requires $(2n - 1)m$ flops or simply $2mn$
- m elements in y; each element requires an inner product of length n
- approximately $2mn$ for large n
- flop count is lower for structured matrices
  - A diagonal: n flops
  - A lower triangular: $1 + 3 + 5 + \cdots + 2n - 1 = n^2$ flops
  - A sparse: #flops $\ll 2mn$

Matrix-matrix product product of $m \times n$ matrix A and $n \times p$ matrix B:

$$C = AB$$

requires $mp(2n-1)$ flops

- mp elements in C; each element requires an inner product of length n
- approximately $2mnp$ for large n

Outline

- matrix notation
- matrix operations
- complexity
- **examples of matrices**
- graphs
- convolution

Matrix examples

Images

- $m \times n$ matrix denote a monochrome (black and white) image
- X_{ij} is i, j pixel value in a monochrome image

Rainfall data

- $m \times n$ matrix A gives the rainfall at m different locations on n consecutive days
- A_{ij} is rainfall at location i on day j

Multiple asset returns

- $T \times n$ matrix R gives the returns of n assets over T periods
- R_{ij} is return of asset j in period i
- jth column of R is a T-vector that is the return time series for asset j

Matrix-vector product examples

Return matrix

- R is $T \times n$ matrix of asset returns (returns of n assets over T periods)
- R_{ij} is return of asset j in period i (say, in percentage)
- n-vector w gives investments in the assets (*e.g.*, $w_4 = 0.15$ means that 15% of the total portfolio value is held in asset 4)
- T-vector Rw is time series of the portfolio return over periods $1, \ldots, T$

Image cropping

- MN-vector x is image, with its entries giving the pixel values in specific order
- y is the $(M/2) \times (N/2)$ image that is the upper left corner (cropped version)
- we have $y = Ax$, where A is an $(MN/4) \times (MN)$ selector matrix
- ith row of A is $e_{k_i}^T$, k_i is index of the pixel in x that corresponds to ith pixel in y

Feature matrix

- $X = [x_1 \ \cdots \ x_N]$ is $n \times N$ feature matrix
- column x_j is feature n-vector for object or example j
- X_{ij} is value of feature i for example j
- n-vector w is weight vector
- $s = X^T w$ is vector of scores for each example; $s_j = x_j^T w$

Cost of production

production inputs (materials, parts, labor,...) are combined to make products

- x_j is price per unit of production of input j
- A_{ij} is units of production of input j required to manufacture one unit of product i
- $y = Ax$ is production cost (y_i is production cost per unit of product i)
- ith row of A is bill of materials for unit of product i

Signal power in wireless system

- n transmitter/receiver pairs
- transmitter j transmits to receiver j (and, unintentionally, to the other receivers)
- p_j is power of jth transmitter
- s_i is received signal power of ith receiver
- z_i is received interference power of ith receiver
- G_{ij} is path gain from transmitter j to receiver i
- we have $s = Ap$, $z = Bp$, where

$$A_{ij} = \begin{cases} G_{ii} & i = j \\ 0 & i \neq j \end{cases} \qquad B_{ij} = \begin{cases} 0 & i = j \\ G_{ij} & i \neq j \end{cases}$$

- A is diagonal; B has zero diagonal (ideally, A is 'large', B is 'small')

Vandermonde matrix

- polynomial of degree $n-1$ or less with coefficients $x_1, x_2, \ldots, x_n$:

$$p(t) = x_1 + x_2 t + x_3 t^2 + \cdots + x_n t^{n-1}$$

- values of $p(t)$ at m points $t_1, \ldots, t_m$:

$$\begin{bmatrix} p(t_1) \\ p(t_2) \\ \vdots \\ p(t_m) \end{bmatrix} = \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ 1 & t_2 & \cdots & t_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & t_m & \cdots & t_m^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$
$$= Ax$$

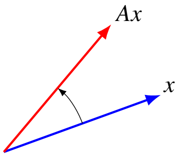
  the matrix A is called a *Vandermonde matrix*

- Ax maps coefficients of polynomial to function values

Geometric transformations

Rotation in a plane

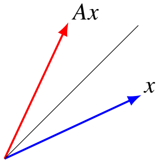
$$A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$



$y = Ax$ is x rotated counterclockwise over an angle θ

Reflection

$$y = \begin{bmatrix} \cos(2\theta) & \sin(2\theta) \\ \sin(2\theta) & -\cos(2\theta) \end{bmatrix} x$$



$y = Ax$ is the vector obtained by reflecting x through the line that passes through the origin, inclined θ radians with respect to horizontal

Finding the geometric matrix

- when a geometric transformation is represented by matrix vector multiplication
- a simple method to find the matrix is to find its columns
- the ith column is the vector $a_i = Ae_i$

Example: consider clockwise rotation by 90° in 2-D

- rotating the vector $e_1 = (1, 0)$ by 90° gives $(0, -1)$
- rotating $e_2 = (0, 1)$ by 90° gives $(1, 0)$
- so rotation by 90° is given by

$$y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x$$

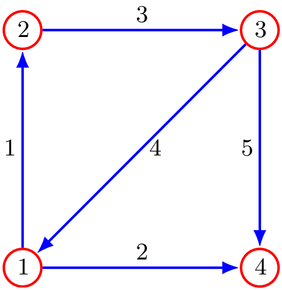
Outline

- matrix notation
- matrix operations
- complexity
- examples of matrices
- **graphs**
- convolution

Incidence matrix

- *directed graph* consists of m vertices (nodes), n directed edges (arcs, branches)
- *incidence matrix* is $m \times n$ matrix A with

$$A_{ij} = \begin{cases} 1 & \text{if edge } j \text{ point to node } i \\ -1 & \text{if edge } j \text{ point from node } i \\ 0 & \text{otherwise} \end{cases}$$



$$A = \begin{bmatrix} -1 & -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Flow conservation

- graph is used to represent a network
- through which some quantity such as electricity, water, or heat flows
- assume n-vector x gives flows along the edges
- $x_j > 0$ means flow follows edge direction
- Ax is m-vector that gives the total or net flows
- $(Ax)_i$ is the net flow into node i (flows in node i minus flows out)

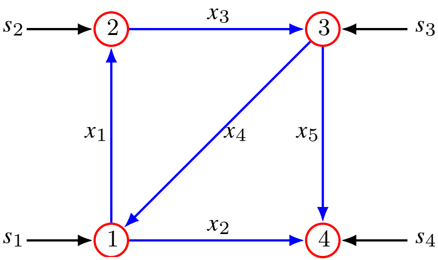
$$(Ax)_i = \sum_{\substack{\text{edge } j \text{ enters} \\ \text{node } i}} x_j - \sum_{\substack{\text{edge } j \text{ leaves} \\ \text{node } i}} x_j$$

- can include external source flows $Ax + s$, s_i is flow entering/leaving node i

Kirchhoff's current law

n-vector $x = (x_1, x_2, \ldots, x_n)$ with x_j the *current* through branch j

$(Ax)_i =$ total current arriving at node i (excluding sources)

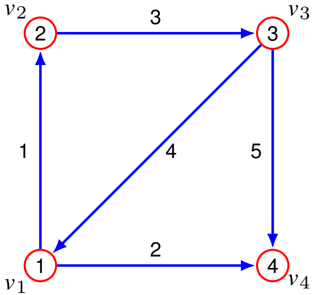


$$Ax + s = \begin{bmatrix} -x_1 - x_2 + x_4 + s_1 \\ x_1 - x_3 + s_2 \\ x_3 - x_4 - x_5 + s_3 \\ x_2 + x_5 + s_4 \end{bmatrix}$$

Node potentials

m-vector $v = (v_1, v_2, \ldots, v_m)$ with v_i the *potential* value at node i

$$(A^T v)_j = v_k - v_l \text{ if edge } j \text{ goes from node } l \text{ to } k$$



$$A^T v = \begin{bmatrix} v_2 - v_1 \\ v_4 - v_1 \\ v_3 - v_2 \\ v_1 - v_3 \\ v_4 - v_3 \end{bmatrix}$$

if v_i are node voltages in a circuit, then $(A^T v)_j =$ (negative) voltage across branch j

Dirichlet energy

$\|A^T v\|^2$ is the sum of squared potential differences

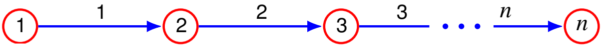
$$\|A^T v\|^2 = \sum_{\text{edges } i \to j} (v_j - v_i)^2$$

- called *Dirichlet energy*
- $\mathcal{D}(v)$ is small when potential values of neighboring nodes are similar
- used as a measure of non-smoothness (roughness) of node potentials on a graph

Example: for the graph on the previous page

$$\|A^T v\|^2 = (v_2 - v_1)^2 + (v_4 - v_1)^2 + (v_3 - v_2)^2 + (v_1 - v_3)^2 + (v_4 - v_3)^2$$

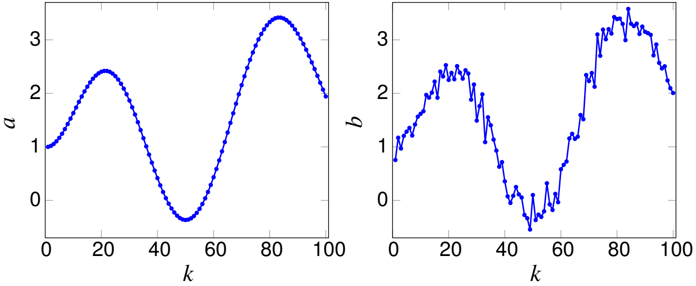
Chain graph



- the $n \times (n-1)$ incidence matrix is the transpose of the difference matrix D
- Dirichlet energy:

$$\mathcal{D}(v) = \|Dv\|^2 = (v_2 - v_1)^2 + \cdots + (v_n - v_{n-1})^2$$

- used as a measure of the non-smoothness time series



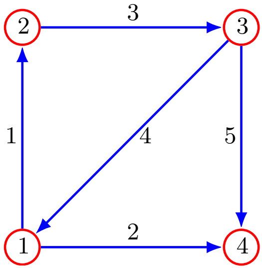
$\mathcal{D}(a) = 1.14$ and $\mathcal{D}(b) = 8.99$

Graph Laplacian

if A is incidence matrix, matrix $L = AA^T$ is the *Laplacian* of the graph

$$L_{ij} = \begin{cases} \text{degree of node} & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and vertices } i \text{ and } j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

the *degree* of a node is the number of edges incident to it



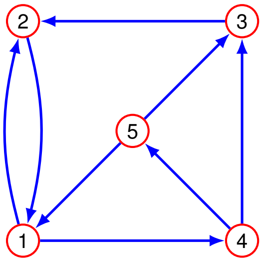
$$L = AA^T = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

- assume there are no self-loops and at most one edge between any two vertices
- we have $\mathcal{D}(v) = \|A^T v\|^2 = v^T L v$ (sometimes called *Laplacian quadratic form*)

Adjacency matrix of directed graph

adjacency matrix of directed graph is the $n \times n$ matrix A with:

$$A_{ij} = \begin{cases} 1 & \text{if edge from node } j \text{ to node } i \\ 0 & \text{otherwise} \end{cases}$$



$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- can describe a *relation* between n objects $\mathcal{R}$ ($A_{ij} = 1$ if $(i, j) \in \mathcal{R}$)
- can be defined in reverse; $A_{ij} = 1$ means a directed edge from $i \to j$

Paths in directed graph

square of adjacency matrix:

$$(A^2)_{ij} = \sum_{k=1}^{n} A_{ik}A_{kj}$$

- each term is either zero, or one when $j \rightarrow k$ and $k \rightarrow i$
- $(A^2)_{ij}$ is number of paths of length 2 from j to i
- more generally, $(A^\ell)_{ij}$ = number of paths of length ℓ from j to i
- for the example,

$$A^2 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 2 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad A^3 = \begin{bmatrix} 1 & 1 & 0 & 1 & 2 \\ 2 & 0 & 1 & 2 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

e.g., there are two paths of length two from 5 to 2

Outline

- matrix notation
- matrix operations
- complexity
- examples of matrices
- graphs
- **convolution**

Convolution

convolution between n-vector a and m-vector b is the $(n+m-1)$-vector

$$c_k = (a * b)_k = \sum_{\substack{\text{all } i,j \text{ with} \\ i+j=k+1}} a_i b_j, \quad k = 1, \ldots, n+m-1$$

- for example with $a = (a_1, a_2, a_3, a_4)$, $b = (b_1, b_2, b_3)$, we have

$$\begin{aligned} c_1 &= a_1 b_1 \\ c_2 &= a_1 b_2 + a_2 b_1 \\ c_3 &= a_1 b_3 + a_2 b_2 + a_3 b_1 \\ c_4 &= a_2 b_3 + a_3 b_2 + a_4 b_1 \\ c_5 &= a_3 b_3 + a_4 b_2 \\ c_6 &= a_4 b_3 \end{aligned}$$

- example: $(1, 0, -1) * (2, 1, -1) = (2, 1, -3, -1, 1)$
- arises in many applications and contexts

Interpretation and properties

Interpretation: if a and b are the coefficients of polynomials

$$p(x) = a_1 + a_2 x + \cdots + a_n x^{n-1}, \quad q(x) = b_1 + b_2 x + \cdots + b_m x^{m-1}$$

then $c = a * b$ gives the coefficients of the product polynomial

$$p(x)q(x) = c_1 + c_2 x + c_3 x^2 + \cdots + c_{n+m-1} x^{n+m-2}$$

Properties

- symmetric: $a * b = b * a$
- associative: $(a * b) * c = a * (b * c)$
- if $a * b = 0$ then $a = 0$ or $b = 0$

these properties follow directly from the polynomial product interpretation

Convolution as matrix-vector product

for fixed a (or b) the convolution can be expressed as matrix-vector product of b (or a)

$$c = a * b = T(b)a = T(a)b$$

for matrices $T(a)$ and $T(b)$

- example: for 4-vector a and a 3-vector b,

$$T(b) = \begin{bmatrix} b_1 & 0 & 0 & 0 \\ b_2 & b_1 & 0 & 0 \\ b_3 & b_2 & b_1 & 0 \\ 0 & b_3 & b_2 & b_1 \\ 0 & 0 & b_3 & b_2 \\ 0 & 0 & 0 & b_3 \end{bmatrix}, \quad T(a) = \begin{bmatrix} a_1 & 0 & 0 \\ a_2 & a_1 & 0 \\ a_3 & a_2 & a_1 \\ a_4 & a_3 & a_2 \\ 0 & a_4 & a_3 \\ 0 & 0 & a_4 \end{bmatrix}$$

- $T(b)$ is a *Toeplitz* matrix (values on diagonals are equal)
- columns of $T(a)$ are shifted versions of a padded with zeros

Examples

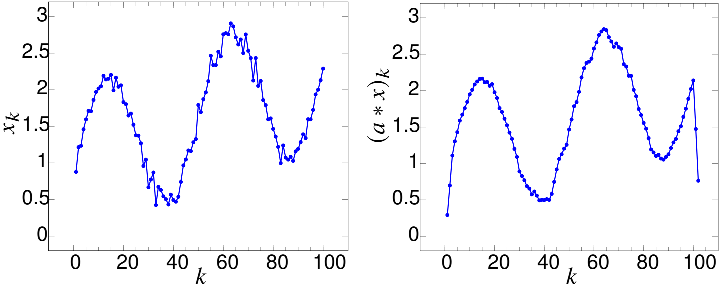
Moving average of a time series

- n-vector x represents a time series
- the 3-period moving average of the time series is the time series

$$y_k = (1/3)\,(x_k + x_{k-1} + x_{k-2}), \quad k = 1, 2, \ldots, n+2$$

(with x_k interpreted as zero for $k < 1$ and $k > n$)

- can be expressed as a convolution $y = a * x$ with $a = (1/3, 1/3, 1/3)$



Audio filtering

- x is audio signal
- a is a vector called filter coefficients
- $y = a * x$ is filtered audio signal
- example: audio tone controls

Communication channel

- u signal transmitted over some channel (electrical, radio, optical,...)
- receiver receives $y = c * u$
- c is channel *impulse response*

Input-output convolution system

many systems with input u and output y can be modeled as convolution $y = h * u$

- h is called the *system impulse response*
- for m-vector u input, n-vector h, we can express $(m + n - 1)$-vector y output,

$$y_i = \sum_{j=1}^{n} u_{i-j+1} h_j$$

  (interpreting u_k as zero for $k < n$ or $k > n$)

- interpretation: output y_i at time i is a linear combination of $u_i, \ldots, u_{i-n+1}$
- h_3 determines current output's dependency on input from two time steps ago

References and further readings

- S. Boyd and L. Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares,* Cambridge University Press, 2018.
- L. Vandenberghe. *EE133A lecture notes,* University of California, Los Angeles. (`http://www.seas.ucla.edu/~vandenbe/ee133a.html`)