Review for Midterm 2

Contents

Preliminaries

Two Types of Errors

- absolute error
- relative error

Floating-Point Numbers

- binary scientific notation:

$$\pm\left(1+\frac{b_1}{2}+\frac{b_2}{2^2}+\cdots+\frac{b_d}{2^d}\right)2^E,$$

  where b_i is 0 or 1 and E is an integer.
    - d determines the *resolution*
    - the range of E determines the *scope* or *extent*
- IEEE Standard (double-precision; 64 bits)
    - $d = 52$ and $-1022 \leqslant E \leqslant 1023$
    - $\boxed{\text{eps}} = 2^{-52} \approx 2 \times 10^{-16}$
    - `realmin`, `realmax`

Floating-Point Numbers (cont')

- Key features
  - On any interval of the form $[2^E, 2^{E+1})$, there are 2^d evenly-spaced f-p numbers.
  - The spacing between two adjacent f-p numbers in $[2^E, 2^{E+1})$ is $2^{E-d} = 2^E \boxed{\text{eps}}$.
  - The gap between 1 and the next f-p number is $\boxed{\text{eps}}$, the machine epsilon.
  - Representation error (in relative sense) is bounded by $\frac{1}{2}\boxed{\text{eps}}$.

Conditioning (of a problem)

- The condition number measures the ratio of error in the result (or output) to error in the data (or input).
- Recall the definition of condition number $\kappa_f(x)$
- A large condition number implies that the error in a result may be much greater than the round-off error used to compute it.
- Catastrophic cancellation is one of the most common sources of loss of precision.

Stability (of an algorithm)

- When an algorithm produces much more error than can be explained by the condition number, the algorithm is unstable.

Square Linear Systems

Polynomial Interpolation

- Polynomial interpolation leads to a square linear system of equations with a Vandermonde matrix.

Gaussian Elimination and (P)LU Factorization

- A triangular linear system is solved by backward substitution or forward elimination.
- A general linear system is solved by Gaussian elimination.
- Gaussian elimination (with partial pivoting) is equivalent to (P)LU factorization.
- Solving a triangular linear system of size $n \times n$ takes $\sim n^2$ flops.
- PLU factorization takes $\underline{\sim \frac{2}{3}n^{\overset{3}{\cancel{2}}}}$ flops.

Norms

A *norm* generalizes the notion of length for vectors and matrices.

- **Vector *p*-norm**

$$\|\mathbf{v}\|_p = \left(\sum_{i=1}^{n} |b_i|^p\right)^{1/p}, \quad p \in [1, \infty)$$

  and

$$\|\mathbf{v}\|_\infty = \max_i |v_i|$$

- **Matrix *p*-norm** (induced)

$$\|A\|_p = \max_{\|\mathbf{x}\|_p = 1} \|A\mathbf{x}\|_p, \quad p \in [1, \infty]$$

- **Frobenius norm** (non-induced)

$$\|A\|_F = \left(\sum_i \sum_j |a_{i,j}|^2\right)^{1/2}$$

- MATLAB: `norm` can calculate both vector and matrix norms

Row and Column Operations

Various row and column operations can be emulated by matrix multiplications.
("Left-multiplication for row actions, right-multiplication for column actions")

- row/column extraction (unit vector)
- row/column swap (elementary permutation matrix)
- row/column rearrangement (permutation matrix)
- row replacement $R_i \to R_i + cR_j$ (Gaussian transformation matrix)

Conditioning/Stability

- Partial pivoting is needed for numerical stability.
- The matrix condition number is equal to the condition number of solving a linear system of equations.

Programming Notes

- Built-in functionalities
  - backslash (\)
  - `lu`
  - `norm`
  - `cond`, `condest`, `linsolve`
- Demonstration/Instructional codes
  - `backsub` and `forelim`
  - `GEnp` and `GEpp`
  - `mylu` and `myplu`

Overdetermined Linear Systems

Polynomial Approximation

- The most common solution to overdetermined systems is obtained by *least squares*, which minimizes the 2-norm of the residual vector.
- Least squares is used to find fitting functions that depend linearly on the unknown parameters.
- Equivalence of the LLS problem and the normal equation
  - linear algebra proof
  - calculus proof

QR Factorization

- Orthogonal sets of vectors are preferred to nonorthogonal ones in computing. (no catastrophic cancellation)
- Matrices with orthonormal columns and orthogonal matrices enjoy many *nice* analytical properties.
- QR factorization plays a role in LLS similar to that of LU factorization in square linear systems.

- $Q^{-1} = Q^T \leftarrow Q^T Q = Q Q^T = I$
- $\|Q\vec{x}\|_2 = \|\vec{x}\|_2$
- $\|Q\|_2 = 1$ $\quad\quad \kappa_2(Q) = \|Q\|_2 \|Q^{-1}\|_2$
- $\mathrm{cond}_2(Q) = \kappa_2(Q) = 1$

Two Types of QR Factorization

For $A \in \mathbb{R}^{m \times n}$, $m \geqslant n$:

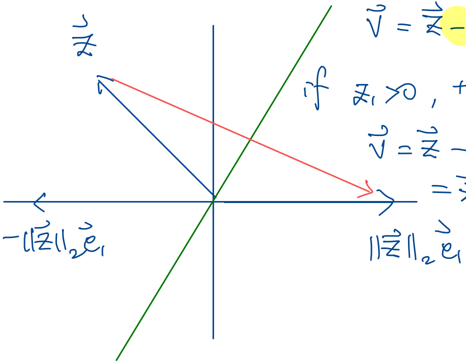
- Thick QR factorization: $A = QR$
  - $Q \in \mathbb{R}^{m \times m}$ orthogonal
  - $R \in \mathbb{R}^{m \times n}$ upper triangular
  - obtained by using successive Householder transformation matrices for *triangularization*
- Thin: $A = \widehat{Q}\widehat{R}$
  - $\widehat{Q} \in \mathbb{R}^{m \times n}$ orthonormal columns
  - $\widehat{R} \in \mathbb{R}^{n \times n}$ upper triangular
  - obtained by Gram-Schmidt *orthonormalization* procedure

Householder Transformation Matrices

- A Householder transformation matrix H (associated with a vector $\mathbf{z}$) is a *reflection* matrix which is
  - symmetric,
  - orthogonal, and
  - transforms $\mathbf{z}$ to $\pm \|\mathbf{z}\|_2 \, \mathbf{e}_1$.

$$\vec{v} = \|\vec{z}\|_2 \vec{e}_1 - \vec{z}$$

$$H = I - 2\frac{\vec{v}\vec{v}^T}{\vec{v}^T\vec{v}}$$



if $z_1 < 0$, then

$$\vec{v} = \vec{z} - \|\vec{z}\| \vec{e}_1$$

if $z_1 > 0$, then

$$\vec{v} = \vec{z} - (-\|\vec{z}\| \vec{e}_1)$$
$$= \vec{z} + \|\vec{z}\| \vec{e}_1.$$

3 (c) Show that $H\vec{z} = \|\vec{z}\|_2 \vec{e}_1$. (Recall: $\vec{v} = \|\vec{z}\|_2 \vec{e}_1 - \vec{z}$.)

Soln

Note that

$$\vec{v}^T \vec{v} = (\|\vec{z}\| \vec{e}_1 - \vec{z})^T (\|\vec{z}\| \vec{e}_1 - \vec{z})$$

$$= (\|\vec{z}\| \vec{e}_1^T - \vec{z}^T)(\|\vec{z}\| \vec{e}_1 - \vec{z})$$

$$= \|\vec{z}\|^2 \underbrace{\vec{e}_1^T \vec{e}_1}_{=1} - \|\vec{z}\| \boxed{\vec{e}_1^T \vec{z}} - \|\vec{z}\| \boxed{\vec{z}^T \vec{e}_1} + \underbrace{\vec{z}^T \vec{z}}_{= \|\vec{z}\|^2}$$

same

$$= \underline{2\|\vec{z}\|^2 - 2\|\vec{z}\| \vec{z}^T \vec{e}_1}$$

Scratch

$$H = I - \frac{2\vec{v}\vec{v}^T}{\vec{v}^T\vec{v}}$$

$$H\vec{z} = \left(I - \frac{2\vec{v}\vec{v}^T}{\vec{v}^T\vec{v}}\right)\vec{z}$$

$$= \vec{z} - \frac{2\vec{v}\vec{v}^T\vec{z}}{\vec{v}^T\vec{v}}$$ ✓

Programming Notes

- Built-in functionalities
  - backslash (\)
  - `qr`
- Demonstration/Instructional codes
  - `lsqrfact`: solving least squares using QR
  - `gs`: Gram-Schmidt (for homework)

Visualization of matrix p-norm

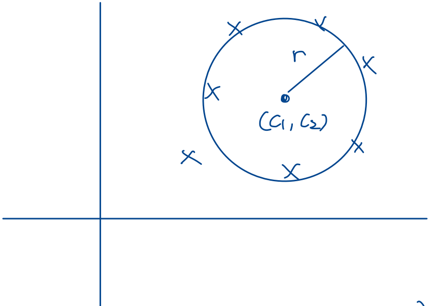
- Unit vectors

- $$\|A\|_p := \max_{\vec{x} \neq 0} \frac{\|A\vec{x}\|_p}{\|\vec{x}\|_p} = \max_{\|\vec{x}\|_p = 1} \|A\vec{x}\|_p$$

Fitting a circle

Given data $\{(x_k, y_k) : k = 1, \cdots, m\}$.

Find c_1, c_2, and r as in



$$(x - c_1)^2 + (y - c_2)^2 = r^2$$

$$x^2 - 2x c_1 + c_1^2 + y^2 - 2y c_2 + c_2^2 = r^2$$

$$x^2 + y^2 = 2x c_1 + 2y c_2 + \underbrace{r^2 - c_1^2 - c_2^2}_{\substack{\parallel \\ c_3}}$$

Note $c_3 = r^2 - c_1^2 - c_2^2$

$$\Rightarrow r = \sqrt{c_1^2 + c_2^2 + c_3}$$

$$\begin{bmatrix} 2x_1 & 2y_1 & 1 \\ 2x_2 & 2y_2 & 1 \\ \vdots & \vdots & \vdots \\ 2x_m & 2y_m & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 \\ \vdots \\ x_m^2 + y_m^2 \end{bmatrix}$$

<u>Parametric Rep'n</u>

$$\begin{cases} x(\theta) = c_1 + r\cos(\theta) \\ y(\theta) = c_2 + r\sin(\theta) \end{cases} \quad \text{for } \theta \in [0, 2\pi]$$