Algorithmique numérique - ANUM

Academic year 2022-23

Additional Notes

aka. notes to take with u for exam

Instructor : Prof. Stef Graillat

written by Hayato Ishida

*This note only contains additional information to lecture slides. This note may be useful only if use with lecture slides.

Contents

I. Topic 1: Introduction to MATLAB and to floating-point arithmetic

Rounding

- *Machine numbers*: Real numbers that are exactly representable in a given floating-number system.

- In other cases, it is approximated by a "nearby" floating-point number. This approximation is called *rounding* and the error caused by this approximation is called *rounding error*.

- The most accurate rounding method is *rounding to nearest* but it is more expensive to implement. It is also a default set by IEEE.

Cancellation*

- An source of rounding error that occurs with subtraction.

- It occurs when subtracting two nearby numbers. the most significant digits in operand matches and cancel each other

- Suppose a and b already contain some rounding error, perform subtraction on these number leads to large error. Example : consider three floating-number $a = 1.22, b = 3.34, c = 2.27$ and perform $b^2 - 4ac$. The exact result of this is 0.0292, however, with rounding: $b^2 = 11.1556$ round to 11.2 and $4ac = 11.0776$ round to 11.1, then result of subtraction is 0.1. *note that this is just a example and result might differ in different settings.

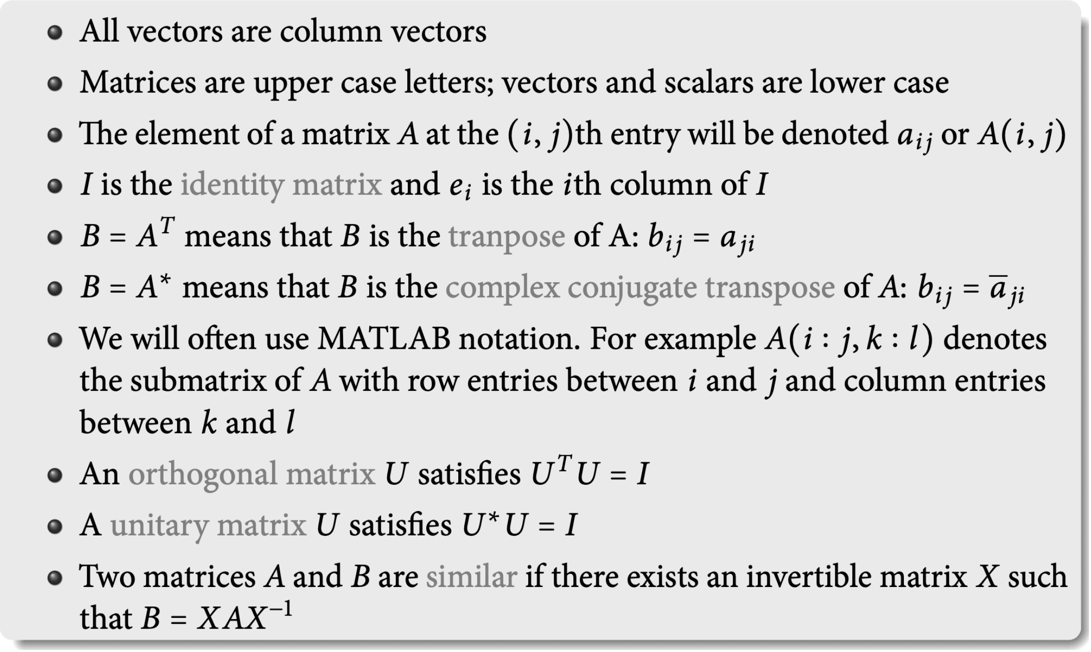
Absorption

- with floating-point we lose some algebratic property of real number. For instance, we might face the situation where:

$$(a + b) + c \neq a + (b + c)$$

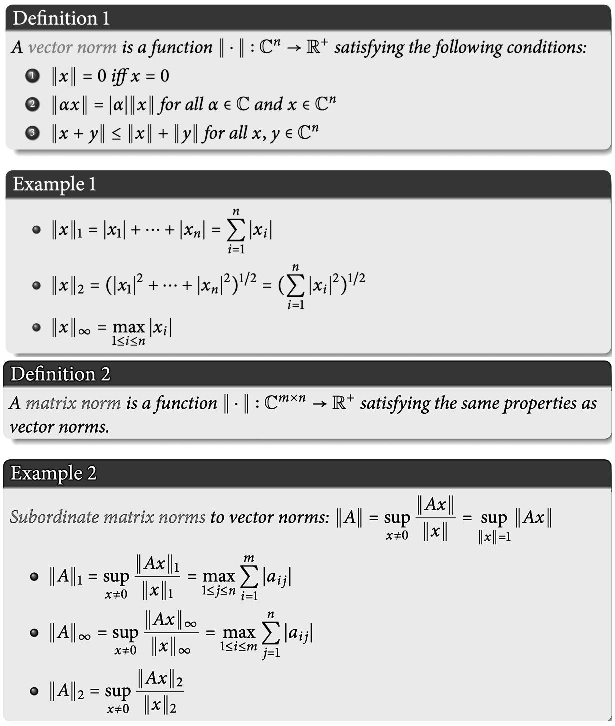
II. Topic 2: Matrix computation

Figure 1: Notations



- All vectors are column vectors
- Matrices are upper case letters; vectors and scalars are lower case
- The element of a matrix A at the (i, j)th entry will be denoted a_{ij} or $A(i, j)$
- I is the identity matrix and e_i is the ith column of I
- $B = A^T$ means that B is the tranpose of A: $b_{ij} = a_{ji}$
- $B = A^*$ means that B is the complex conjugate transpose of A: $b_{ij} = \overline{a}_{ji}$
- We will often use MATLAB notation. For example $A(i : j, k : l)$ denotes the submatrix of A with row entries between i and j and column entries between k and l
- An orthogonal matrix U satisfies $U^T U = I$
- A unitary matrix U satisfies $U^* U = I$
- Two matrices A and B are similar if there exists an invertible matrix X such that $B = XAX^{-1}$

Figure 2: Norms



Definition 1

A vector norm is a function $\| \cdot \| : \mathbb{C}^n \to \mathbb{R}^+$ satisfying the following conditions:

1. $\|x\| = 0$ *iff* $x = 0$
2. $\|\alpha x\| = |\alpha| \|x\|$ *for all* $\alpha \in \mathbb{C}$ *and* $x \in \mathbb{C}^n$
3. $\|x + y\| \leq \|x\| + \|y\|$ *for all* $x, y \in \mathbb{C}^n$

Example 1

- $\|x\|_1 = |x_1| + \cdots + |x_n| = \sum_{i=1}^{n} |x_i|$
- $\|x\|_2 = (|x_1|^2 + \cdots + |x_n|^2)^{1/2} = (\sum_{i=1}^{n} |x_i|^2)^{1/2}$
- $\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$

Definition 2

A matrix norm is a function $\| \cdot \| : \mathbb{C}^{m \times n} \to \mathbb{R}^+$ satisfying the same properties as vector norms.

Example 2

Subordinate matrix norms to vector norms: $\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sup_{\|x\|=1} \|Ax\|$

- $\|A\|_1 = \sup_{x \neq 0} \frac{\|Ax\|_1}{\|x\|_1} = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |a_{ij}|$
- $\|A\|_\infty = \sup_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty} = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |a_{ij}|$
- $\|A\|_2 = \sup_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}$

Dense matrix: matrix that does not have a large number of zero element.

Sparse matrix: matrix that has large number of zero elements.

*there is no specific number of zero elements to define these two type of matrix.

- Transpose: $a_{ij} \to a_{ji}^T$

Three types of classical problem

- liner system
- least-square problem
- eigenvalue/ eigenvector

Unit triangular matrix: triangular matrix with 1s on the main diagonal. Upper unit triangular matrix, and lower unit triangular matrix.

Unit matrix: Identity matrix.

Permutation matrix: permutation matrix on left side corresponding to row permutation, and right side corresponding to column permutation

LU decomposition

- Gaussian elimination algorithm

- given matrix A of size $n \times n$, $PA = LU$
  - P: permutation matrix
  - L: unit lower triangular matrix
  - U: upper triangular matrix
- In matlab, [L,U,P] = lu(A) or [PtL, U] = lu(A) to calculate $P^T L$ and U.
- cost: $n^3/3$ multiplication
- use to solve liner system
  - $Ax = b$, given A and b, decompose A as $PA = LU$. $PAx = LUx = Pb$
  - let $y = Ux$, then $Ly = Pb$
  - In MatLab, it is $x = A \setminus b$

Symmetric positive-define

Matrix A $n \times n$, $A^T = A$ and $x^T Ax > 0$for all $x \neq 0$.

Choleskie decomposition

$A = LL^T$, L: lower triangular matrix

$$\begin{bmatrix} A_{1,1} & A_{1,2:n} \\ A_{2:n,1} & A_{2:n,2:n} \end{bmatrix} = \begin{bmatrix} R_{1,1} & 0 \\ R^T_{1,2:n} & R^T_{2:n,2:n} \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2:n} \\ 0 & R_{2:n,2:n} \end{bmatrix} = \begin{bmatrix} R^2_{1,1} & R_{1,1}R_{1,2:n} \\ R_{1,1}R^T_{1,2:n} & R^T_{1,2:n}R_{1,2:n} + R^T_{2:n,2:n}R_{2:n,2:n} \end{bmatrix}$$

$A = LDL^T$

where L: unit lower triangular matrix

D: diagonal matrix (elements only on the diagonal)

$$\begin{bmatrix} A_{1,1} & A_{1,2:n} \\ A_{2:n,1} & A_{2:n,2:n} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ R^T_{1,2:n} & 1 \end{bmatrix} \begin{bmatrix} R^2_{1,1} & 0 \\ 0 & \end{bmatrix} \begin{bmatrix} 1 & R_{1,2:n} \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} R^2_{1,1} & R_{1,1}R_{1,2:n} \\ R_{1,1}R^T_{1,2:n} & R^T_{1,2:n}R_{1,2:n} + R^T_{2:n,2:n}R_{2:n,2:n} \end{bmatrix}$$

- If matrix A is symmetric positive-definite, it is more convenient to use the Choleskie decomposition.

- cost: hlf of the LU decompostion.
- MatLab : chol(A)
- If A is nonsingular (invertible) thhen the liner system $Ax = b$ has a unique solution.
- If A is singular then x is a solution of $Ax = b$ if b can be written as a liner combination of ome columns of A. In this case, every vector $x + y$ is a solution if $Ay = 0$.
- Nonsingular vs Singular
  - Nonsingular: invertible, $det(A) \neq 0$
  - Singular: $det(A) = 0$

Using Gaussian elimination, you can find determinant easily. Determinant is, determinant of corresponding upper triangular matrix, which is the product of digonal elements.

Sensitivity of the solution of a linear system

Assuming that we perturb the system such that we now need to solve.

$(A + \triangle A)y = b + \triangle b$

We want to know to which distance the solution y of the perturbed system is from the solution x of the intial system

Let, $\varepsilon_A = \frac{\|\triangle A\|}{\|A\|}$, $\varepsilon_b = \frac{\|\triangle b\|}{\|b\|}$, $\kappa = \|A\|\|A^{-1}\|$ (condition number of matrix A)

If, $\kappa\varepsilon_A < 1$, $\frac{\|x-y\|}{\|x\|} \leq \frac{\kappa}{1-\kappa\varepsilon_A}(\varepsilon_A + \varepsilon_b)$

QR decomposition

$A = QR$, there are two possible case of QR.

- $A(m \times n, m \geq n) = Q(m \times m)R(m \times n)$ or $Q(m \times n)R(n \times n)$

Three QR decomposition algorithm.

- Givens rotation (good for matrix A, $m \times m$
- Gram-Schmidt oethogonalization (good for matrix A, $m \times n$
- Householder reflections

*Gram-Schmidt is easiest/ less complicated according to Jermy....

Given's rotation

We assume all matrices are real ones. A simple orthogonal matrix, a rotation, can be used to introduce one zero at a time into a real matrix.

Given matrix G: written as

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

where, $c^2 + s^2 = 1$, (c and s have the geometric interpretation of the cosine and sine of an angle θ). A vector multiply by G rotate through an angle θ.

G_{ij}: denote and $n \times n$, identity matrix with its ith and jth rows modified to include the Given rotation. It modifies all rows except ith and jth rows.

Algorithm:

Define Given's matrix again,

- $g_{k,k} = 1$, for all $1 \leq k \leq m,\ k \neq i, j$
- $g_{i,i} = g_{j,j} = c$, $g_{j,i} = -g_{i,j} = s$ with $c^2 + s^2 = 1$.

for $i = 3, j = 1$,

$$G_{i,j} = \begin{pmatrix} c & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ -s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

c and s are defined as

$$c = \frac{r_{j,j}}{\sqrt{r_{j,j}^2 + r_{i,j}^2}}, \quad s = \frac{r_{i,j}}{\sqrt{r_{j,j}^2 + r_{i,j}^2}}$$

Algorithm 1: Given's

```
Input: A real matrix A of size m × n
Output: Its QR decomposition
$Q \leftarrow Id_m$;
$R \leftarrow A$;
for $j = 1$to $n$ do
    for $i = j + 1$ to $m$ do
        $R = G_{i,j}R$;
        $Q = QG_{i,j}^T$;
return $Q$, $R$
```

Gram-Schmid:

From the columns $[a_1, ..., a_n]$ of the matrix A, we create an orthonormal basis $\{q_1, ..., 1_n\}$ and save the coefficients that accomplish this goal in an upper triangular matrix R.

Algorithm 2: Gram-Schmidt's

```
Input: A matrix A of size m × n
Output: Its QR decomposition
$r_{1,1} \leftarrow \|a_1\|$;
$q_1 \leftarrow \frac{a_1}{r_{1,1}}$;
for $j = 2$to $n$ do
    $q_j = a_j$;
    for $i = 1$ to j-1 do
        $r_{i,j} = q_i^* q_j$;
        $q_j = q_j - r_{i,j} q_i$;
    $r_{j,j} = \|q_j\|$;
    $q_j = \frac{q_j}{r_{j,j}}$
return Q, R
```

Cost of QR decompositions

- Given's: $2mn^2 - 2/3n^3$ multiplications
- Gram: mn^2 multiplications
- Householder: $mn^2 - 1/3n^3$ multiplications

In MatLab $[Q, R] = qr(A)$ for A of size $m \times n$ with $m \geq n$

- qr(A,0) returns the compact matrix Q (although the full is computed with Householder reflections)
- QR can get the basis for the range of a full-rank matrix A (the first n columns of Q) and the null-space of A^* (the last $m - n$ columns of Q).
- QR can be used to solve linear least squares problems
- Least square problems: $Ax = b$, if $A = QR$ then $Rx = Q^*b$

- SVD can solve most of problems but it is quite expensive. It referred as Swiss knife

- Between dense and sparse, there is no specified boundry.

Algebraic properties of SVD:

- SVD can be rewritten as follow by writing $V = [v_1, v_2, ..., v_n]^T$, and: $U = [u_1, u_2, ..., u_n]^T$:

$$A = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

Perron-Frobenius theorem:

(1) If $A \in \mathbb{R}^{n\times n}$ is nonnegative then

- $\rho(A)$ is an eigenvalue of A.
- there is a nonnegative eigenvector x such that $Ax = \rho(A)x$.

(2) If $A \in \mathbb{R}^{n\times n}$ is nonnegative and irreducible then

- $\rho(A)$ is an eigenvalue of A.
- $\rho(A) > 0$
- there is a nonnegative eigenvector x such that $Ax = \rho(A)x$.
- $\rho(A)$ is a simple eigenvalue.

(3) If $A \in \mathbb{R}^{n\times n}$ is positive then (2) holds and, in addition, $|\lambda| < \rho(A)$ for any eigenvalue λ with $\lambda \neq \rho(A)$.

Stochastic matrix:

A matrix $P \in \mathbb{R}^{n\times m}$ said to be stochastic if:

$P \geq 0$ and,

$$\sum_{j=1}^{n} P_{ij} = 1 \quad \text{all} \quad i = 1, 2, ..., n$$

III. Topic 3: Introduction to optimization

- Given function $f : \mathbb{R}^n \to \mathbb{R}$and set $S \subset \mathbb{R}^n$, find $x^* \in S$ such that $f(x^*) \leq f(x)$ for all $x \in S$. x^* is called minimizer or minimum of f on S.
- It suffices to consider only minimization since maximum of f is minimum $-f$
- Objective function f is usually differentiable

- Constraint set S is defined by a system of equations and inequalities, which may be linear or nonlinear
- Points $x \in S$ are called feasible points.
- If $S = \mathbb{R}^n$, problem is unconstrained

General continuous optimization problem:

$$min\, f(x) \text{ subject to } g(x) = 0 \text{ and } h(x) \leq 0$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$, $h : \mathbb{R}^n \to \mathbb{R}^p$
It is said to be linear programming if all functions are linear. If one of the functions is nonlinear, it is said to be nonlinear programming.

- $x^* \in S$ is global minimum if $f(x^*) \leq f(x)$ for all $x \in S$
- $x^* \in S$ is local minimum if $f(x^*) \leq f(x)$ for all feasible x in some neighborhood of x^*.
- Finding, or even verifying, global minimum is difficult, in general. Most optimization methods are designed to find local minimum, which may or may not be global minimum. If global minimum is desired, one can try several widely separated starting points and see if all produce same result. For some problems, such as linear programming, global optimization is more tractable

Existence of minimum

- If f is continues on closed and bounded set $S \subset \mathbb{R}^n$, then f has a global minimum on S. But if S is not closed or is unbounded, then f may have no local/global minimum on S.
- Continues function f on an unbounded set $S$is coercive if, $\lim_{\|x\| \to +\infty} f(x) = +\infty$. i.e. $f(x)$ must be large whenever $\|x\|$ is large
- If f is coercive on closed, unbounded set $S \subset \mathbb{R}^n$ then f has global minimum on S

Bounded: there exists some kind of boundaries/constraints upon the function. $\iff$ Unbounded/ ex. upper boundary and lower boundary.
Closed: contains all of its boundary points.
Open: not include points on the boundary.
Level set

- A level set for a function $f : S \subset \mathbb{R}^n \to \mathbb{R}$is the set of all points in S for which f has some given constant value

- Given $\gamma \in \mathbb{R}$, the sublevel set is: $L_\gamma = \{x \in S : f(x) \leq \gamma\}$

- If a continuous function f on $S \subset \mathbb{R}^n$ has a nonempty sublevel set that is closed and bounded, then f has global minimum on S

- If S is unbounded, then f is coercive on S iff all of its sublevel sets are bounded

Uniquness of minimum:

- A set S is convex if it contains a line segment between any two of its points.
  $\forall x, y \in S, \forall \alpha \in [0; 1], \alpha x + (1 - \alpha)y \in S$

- A function $f : S \subset \mathbb{R}^n \to \mathbb{R}$ is convex on a convex set S if
  $\forall x, y \in S, \forall \alpha \in]0; 1[, f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$.

- A function $f : S \subset \mathbb{R}^n \to \mathbb{R}$ is strictly convex on a convex set S if
  $\forall x, y \in S, \forall \alpha \in]0; 1[, f(\alpha x + (1 - \alpha)y) < \alpha f(x) + (1 - \alpha)f(y)$

- Any local minimum of convex function f on convex set $S \subset \mathbb{R}^n$ is global minimum of f on S.

- Any local minimum of a strictly convex function f on convex set $S \subset \mathbb{R}^n$ is unique global minimum of f on S.

First-order optimally condition:

- For function of one variable, we can find extremum by differentiating function and setting derivative to zero

- Generalization to function of n variables is to find critical point, i.e., solution of nonlinear system

$$\nabla f(x) = 0, \;\; \nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix}$$

- For continuously differentiable any interior point x^* of S at which f has local minimum must be a critical point of f. But not all critical points are minima: they can also be maxima or saddle points

Second-order optimary condition:

- For twice continuously differentiable $f : S \subset \mathbb{R}^n \to \mathbb{R}$ we can distinguish among critical points by considering Hessian matrix $H_f(x)$ ddefined by:

  $$(H_f(x))_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \quad \text{this is } \underline{\text{symetric}}$$

  Also, $H_f(x) = \nabla^2 f(x)$

- a=At crtical points x^*, if $H_f(x^*)$ is....
  - positive definite, then x^* is a minimum of f
  - negative definite, then x^* is a maximum of f
  - indefinite, then x^* is saddle point of f
  - singular, then various pathological situations are possible. (singular: one of eigenvalue is 0, and not invertible)

III.1. One-dimention

Unimodality:

For minimizing the function of one variable, we need a "bracket" for the solution analogous to sign change for the nonlinear equation. Real-valued function f is unimodal on interval $[a, b]$ if there is unique $x^* \in [a, b]$ such that $f(x^*)$is a minimum of f on$[a, b]$, f isstrictly decreasing for $x \leq x^*$ and is strictly increasing for $x^* \leq x$. Unimodality enables discarding portions of interval based on sample function values, analogous to interval bisection.

Golden section search

Algorithm 3: Golden section search

```
$\tau = (\sqrt{5}$-1)/2 ($\approx 0.618$ and $\tau - 1 \approx 0.382$));
$x_1 = a + (1-\tau)(b-a); f_1 = f(x_1)$ ;
$x_2 = a + \tau(b-a); f_2 = f(x_2)$ ;
while $(b$-$a)>tol$ do
    if $f_1 > f_2$ then
        $a = x_1$;
        $x_1 = x_2$;
        $f_1 = f_2$;
        $x_2 = a + \tau(b-a)$;
        $f_2 = f(x_2)$
    else
        $b = x_2$;
        $x_2 = x_1$;
        $f_2 = f_1$;
        $x_1 = a + (1-\tau)(b-a)$;
        $f_1 = f(x_1)$
```

- By comparing two points on f, we can discard one side.
  - f is unimodal on $[a, b]$, and two points x_1, x_2 within $[a, b]$ and $x_1 < x_2$.
  - Comparing $f(x_1)$ and $f(x_2)$, we can discard $[a, x_1)$ or $(x_2, b]$
- Repeat this process and only one new function evaluation is required for each iteration.
- fraction of where to evaluate the function is fixed. $\tau = (\sqrt{5}-1)/2 \approx 0.618, 1-\tau \approx 0.382$

III.2. Dimension $n \leq 2$

Pattern search method

Nelder-Meade algorithm

Steepest Descent method

$f : \mathbb{R}^n \to \mathbb{R}$, real-value function of n real variables.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

- $-\nabla f(x)$ points downhill towards lower values of f.
- α is a line search parameter that determines how far to go in a given direction $-\nabla f(x)$
- zigzag towards a solution, so slow, convergence rate is linear.

IV. Topic 4: Nonlinear equation

With a given function f, we seek x value such that $f(x) = 0$. A solution x is called a root of the equation or a zero of the function f. Two important cases:

- Single nonlinear equation is one unknown, where $f : \mathbb{R} \to \mathbb{R}$. A solution is a scalar x satisfying $f(x) = 0$.
- System of n nonlinear equation in n unknowns, where $f : \mathbb{R}^n \to \mathbb{R}^n$. A solution is a vector x for which $f(x) = 0$.

Existence and uniqueness of solutions:

- If $f : \mathbb{R} \to \mathbb{R}$ is continuous and $sign(f(a)) \neq sign(f(b))$ then the Intermediate Value Theorem implies that there exists $x^* \in [a, b]$ such that $f(x^*) = 0$. There is no simple analog for dimensions $n > 1$.

Multiplicity:

- If $f(x^*) = f'(x^*) = f''(x^*) = ... = f^{m-1}(x^*) = 0$ but $f^{(m)}(x^*) \neq 0$, then we way that the root x^* has multiplicity m. If $m = 1$, we say that x^* is a simple root

Sensitivity and conditioning:

- The conditioning of root finding problem is opposite to that for evaluating function
- The absolute condition number of root finding a problem for root x^* of $f : \mathbb{R}^N \to \mathbb{R}^n$ is $1/|f'(x^*)|$.
- A root is ill-conditioned if its tangent line is nearly horizontal. In particular, a multiple root is ill-conditioned.

- Absolute condition number of root finding problem for root x^* of $\|J_f^{-1}(x^*)\|$ where J_f is the Jacobian of f,

$$J_f(x) = \{\partial f_i(x)/\partial x_j\}$$

A root is ill-conditioned if the Jacobian matrix is nearly singular

- What do we mean by approximate solution $\hat{x}$ to nonlinear system?

- "small residual": $\|f(\hat{x})\| \approx 0$

  the closeness to the true solution: $\|\hat{x} - x^*\| \approx 0$

- A small residual implies an accurate solution only if the problem is well-conditioned

Convergence rate:

- For a general iterative method, we define the error at iteration k by

$$e_k = x_k - x^*$$

x_k: approximation of the solution. x^*: solution

- The sequence (x_k) is said to converge with a rate r if

$$lim_{k \to +\infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$$

for constant $C \geq 0$

- some interesting special cases:
  - $r = 1$: linear
  - $r > 1$: superlinear
  - $r = 2$: quadratic

Convergence rate	Gain in digits per iteration
linear	constant
superlinear	increasing
quadratic	double

V. Topic 5: Newton's method

Newton's method has a quadratic convergence rate in general and requires an initial point close enough to the solution to converge.

- A local quadratic approximation is truncated Taylor series:

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2$$

- By differentiation, the minimum of this quadratic function of h is given by:

$$h = -f'(x)/f''(x)$$

- Iteration scheme:

$$x_{k+1} = x_k - f'(x_k)/f''(x_k)$$

  is Newton's method for solving nonlinear equation $f'(x) = 0$ Newton's method for finding minimum normally has a quadratic convergence rate,
  but must be started close enough to the solution to converge

Finite difference method:

- We want to approximate the gradient by finite difference

$$(\nabla f(x))_i \approx \frac{f(x+te_i) - f(x)}{t}$$

  for small t and e_i the $i\,th$ standard unit vector

- Small accuracy on the gradient. It is better to use automatic differentiation methods: it is a set of techniques to evaluate the derivative of a function specified by a computer program.

- We know that a minimum x^* satisfies $\nabla f(x^*) = 0$. We can then solve the equation$\nabla f(x) = 0$ with Newton's method.

- We can also approximate f by:

$$f(x+h) \approx f(x) + \nabla f(x)h + \frac{1}{2}h^T H_f(x)h$$

and minimize the quadratic approximation with respect to h

- Iteration:

$$x_{k+1} = x_k - H_f^{-1}(x_k)\nabla f(x_k)$$

  We solve...

$$H_f(x_k)s_k = -\nabla f(x_k)$$

  then compute

$$x_{k+1} = x_k + s_k$$

- quadratic convergence rate

- In principle, the line search parameter is unnecessary with Newton's method since quadratic model determines the length, as well as direction, of step to next approximate solution.

- When started far from solution, however, it may still be advisable to perform line search along direction of Newton step s_k to make method more robust.

- Once iterates are near solution, then $\alpha_k = 1$ should suffice for subsequent iterations

- If objective function f has continuous second partial derivatives, then Hessianmatrix H_f is symmetric,and near minimum it is positive definite. Thus, linear system for step to next iterate can be solved in only about half of work required for LU factorization using Cholesky factorization.

- Far from minimum, $H_f(x_k)$ may not be positive definite, so Newton step sk may not be descent direction for function, i.e., we may not have $\nabla f(x_k)^T s_k < 0$. In this case, an alternative descent direction can be computed, such as negative gradient or direction of negative curvature, and then perform line search