Numerical Methods

Xuemei Chen

University of San Francisco

(Math 375: Numerical Analysis in Fall 2017)

Contents

Preliminaries

Before this course, you need to be familiar with

- Calculus I
- Calculus II: especially Sequences and Series
- Linear Algebra
    1. Vectors, norm, dot product
    2. Linear Independence and Dependence
    3. Span, subspace, bases, dimension of a subspace/vector space
    4. Properties of orthogonal or orthonormal sets/bases; Gram-Schmidt algorithm
    5. Matrix multiplication; Determinant; Inverse
    6. Solving linear equations: algorithm, existence and uniqueness.
    7. Projections; Least square
    8. Eigenvalues and Eigenvectors
    9. SVD

Notations

- $\{x|$ descriptions of $x\}$ is the set of all x that meets the description after the ":" sign. For example, $\{x|x^2-1<0\}$ is the set of all number x such that $x^2-1<0$. We can solve this inequality further and see that $\{x|x^2-1<0\}=(-1,1)$

  Remark: Some books use colon instead of verticle bar as $\{x:$ descriptions of $x\}$
- $\in$: belongs to/is contained in. For example, $a\in\{x|x^2-1<0\}$ means that a is a number satisfying $a^2-1<0$.
- s.t.: such that

Chapter 1

Solving nonlinear equations

If you recall the equations that you are able to solve, you would realize that there are very few. You know how to solve $x^2 - x - 1 = 0$ using the famous quadratic formula, but have you ever wondered formulas for finding roots of polynomials whose degree is higher than 2? Unfortunately, even as simple as a polynomial equation $x^5 - x - 1 = 0$, a math Ph.D feels hopeless to find an exact answer by hand (so you shouldn't feel bad). In fact, there are no formula for polynomials of degree bigger than 4. Equations like $\cos x = x$? does not have a closed form solution either.

When we need to solve nonlinear equations that frequently occur in every aspect of sciences and applications, we use iterative methods.

1.1 Bisection Method

This is a simple, yet efficient method that can be covered in Calculus I as an application of the Intermediate Value Theorem (IVT).

Theorem 1.1 (Intermediate Value Theorem)**.** *If $f(x)$ is continuous on $[a, b]$, then for any m that is in between $f(a)$ and $f(b)$, there exists a number $c \in [a, b]$ such that $f(c) = m$.*

Take $f(x) = x^5 - x - 1$ as an example, whose graph is shown on the left of Figure 1.1. Since $f(1) = -1 < 0$ and $f(1.5) \approx 5.09 > 0$, then by IVT, there is $c \in [1, 1.5]$ such that $f(c) = 0$ and this c is exactly the root that we are looking for.

To have a more precise estimate of c, we evaluate f at the middle point $\dfrac{1 + 1.5}{2} = 1.25$. $f(1.25) \approx 0.8 > 0$. By IVT again, we conclude that the root $c \in [1, 1.25]$. We can keep bisecting the interval and eventually find the root to a sufficient precision. See Figure 1.1.

Note that the key of this Bisection method are:

1. The function needs to be continuous.
2. One needs to be given initial points a, b such that $f(a)f(b) < 0$.

1.2 Newton's Method

Newton's method is a little more sophisticated: it involves derivatives, but it is still Calc I material. This is usually how a calculator or a computer finds a root.

If we are to find the root of $y = f(x)$, we start with a random initial guess x_1. Consider the the tangent line to the curve $y = f(x)$, at $P_0 = (x_0, f(x_0))$. The idea behind Newton's method

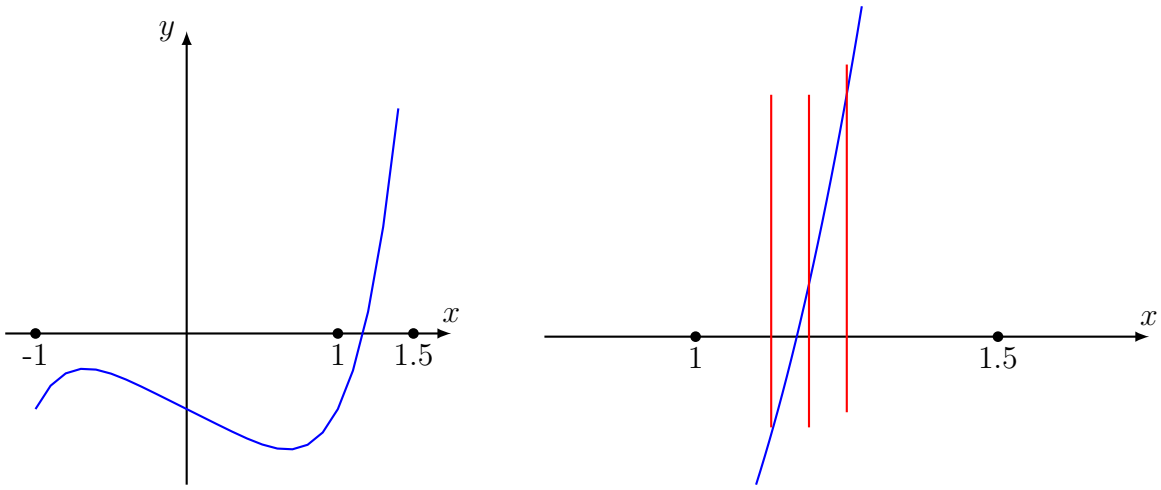


Figure 1.1: Bisection Method

is linearization. Since the tangent line (linearization of f) is close to f, then its x-intercept, x_2, is close to the x-intercept of the curve $y = f(x)$. We can easily find x_2.

The tangent line at the point $(x_0, f(x_0))$ is

$$y - f(x_0) = f'(x_0)(x - x_0).$$

Plugging in $y = 0$ to solve for the x-intercept:

$$0 - f(x_0) = f'(x_0)(x - x_0) \Longrightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

So $x_1 = x_0 - \dfrac{f(x_0)}{f'(x_0)}$. We use x_1 as the next approximation and keep repeating this process, as shown in Figure 1.2, where we gain this iteration formula

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, k \geq 0 \tag{1.1}$$

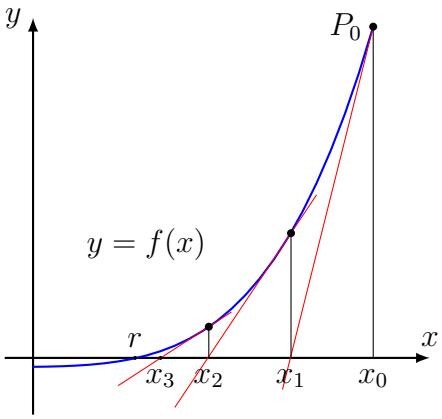


Figure 1.2: Newton's method

Example 1.2. Use Newton's method to approximate $\sqrt{2}$.

Let $f(x) = x^2 - 2$, which makes $\sqrt{2}$ a root of $f(x)$. $f'(x) = 2x$. Let $x_0 = 4$.

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{x_k}{2} + \frac{1}{x_k}$$

$x_1 = 2 + 1/4 = 9/4$
$x_2 = 9/8 + 4/9 = 113/72$
...

The two drawbacks of the Newton's method are: (1) It requires the knowledge of the derivative function; (2) It may not converge if you have an unlucky initial guess x_0. For example, if one picks $x_1 = 0$ for finding root of $y = x^5 - x - 1$, the sequence will not converge to the root (try to draw tangent lines on your own). However, we do have convergence if the initial guess is close enough.

Theorem 1.3 ([1, Theorem 4.3.1])**.** *If the second derivative of $f(x)$ is continuous, and x_0 is sufficiently close to a root r of f, then Newton's method converges to r and ultimately the convergence rate is quadratic.*

The next section introduces the secant method which is Newton's method without requiring derivatives.

1.3 Secant Method

The secant method is defined by taking $f'(x_n)$ to be

$$f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}.$$

So the iteration formula for the secant method is:

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}.$$

1.4 Fixed point methods

x is called a fixed point of $\varphi(x)$ if $\varphi(x) = x$.

Example 1.4. Find fixed point of $\varphi(x) = x^2$.

This is to solve $x = x^2 \Longrightarrow x(x-1) = 0 \Longrightarrow x = 1, 0$

What about finding fixed points of $\varphi(x) = e^{-x}$ since we can't solve $e^{-x} = x$ by hand anymore? A common way is to iterate using the formula

$$x_{k+1} = \varphi(x_k), \qquad k \geq 0$$

which in this example will be $x_0 = 0, x_1 = \varphi(x_0) = 1, x_2 = \varphi(x_1) = e^{-1}, ...$

Finding the fixed point of $\varphi(x)$ is equivalent to finding the root of $\varphi(x) - x$, which means we can always use Newton's method to find a fixed point. Using the same example, we let

$f(x) = e^{-x} - x$, then $f'(x) = -e^{-x} - 1$. So the Newton' iteration is $x_{k+1} = x_k + \dfrac{e^{-x_k} - x_k}{e^{-x_k} + 1}$. Still letting $x_0 = 0$, we have $x_1 = 0 + 1/2 = 1/2$, which is already a better approximation than the fixed point iteration.

Theorem 1.5. *Assume $\varphi \in C^1$ and $|\varphi'(x)| < 1$ in some interval $[r - \delta, r + \delta]$ where r is a fixed point of φ. If $x_0 \in [r - \delta, r + \delta]$ and we use iteration $x_{k+1} = \varphi(x_k)$, then $\lim_{k\to\infty} x_k = r$.*

Proof. See page 95 of [1]. □

Newton's method (1.1) can be considered as a fixed point iteration. To be specific, (1.1) is equivalent to $x_{k+1} = \varphi(x_k)$ with $\varphi(x) = x - \dfrac{f(x)}{f'(x)}$. We can apply Theorem 1.5 to pick a good initial point for Newton's method.

Example 1.6. We are trying to use Newton's method to find the root of $f(x) = x^3 - 1$. The Newton's iteration is $x_{k+1} = \varphi(x_k)$, where $\varphi(x) = x - \dfrac{f(x)}{f'(x)} = x - \dfrac{x^3 - 1}{3x^2} = \dfrac{2}{3}x + \dfrac{1}{3x^2}$. $\varphi'(x) = \dfrac{2}{3} - \dfrac{2}{3}x^{-3}$.

$$|\varphi'(x)| < 1 \Longrightarrow \left|\frac{2}{3} - \frac{2}{3}x^{-3}\right| < 1 \Longrightarrow |\frac{1}{x^3} - 1| < \frac{3}{2} \Longrightarrow -\frac{3}{2} < \frac{1}{x^3} - 1 < \frac{3}{2} \Longrightarrow x^3 > \frac{2}{5} \text{ or } x^3 < -2$$

We can pick x_0 from the interval $(\left(\dfrac{2}{5}\right)^{1/3}, 2 - \left(\dfrac{2}{5}\right)^{1/3})$. Notice that the interval has to be centered around 1.

Remark 1.7. Note that Theorem 1.5 is only a sufficient condition. Even if $|\varphi'(x_0)| < 1$, x_0 can be still a fine initial point. For example, any x_0 can work in Example 1.6.

Chapter 1 Exercises

* means extra credit problems

1 Use Newton's method to approximate $\sqrt{2}$. Let $x_0 = 1$. Compute 2 iterates only.

2 Prove that Newton's method will converge to 0 given any initial value x_0 if we are solving $x^2 = 0$.

3 Write down the first three iterates of the secand method for solving $x^2 - 3 = 0$, starting with $x_0 = 0$ and $x_1 = 1$.

4 We can compute 1/3 by solving $f(x) = 0$ with $f(x) = x^{-1} - 3$.

(a) Write down the Newton iteration for this problem, and compute by hand the first 2 Newton iterates for approximating 1/3, starting with $x_0 = 0.5$.

(b) What happens if you start with $x_0 = 1$?

(c) *In the case of (b), show that the iterates $x_k \to -\infty$ as $k \to \infty$.

(d) Use the theory of fixed point iteration to determine an interval about $1/3$ from which Newton's method will converge to $1/3$.

5 Let function $\varphi(x) = (x^2 + 4)/5$.

(a) Find the fixed point(s) of $\varphi(x)$.

(b) Would the fixed point iteration, $x_{k+1} = \varphi(x_k)$, converge to a fixed point in the interval $[0, 2]$ for all initial gueses $x_0 \in [0, 2]$?

(c) *Find a function $f(x)$ such that its Newton iterations are $x_{k+1} = \varphi(x_k)$. (Hint: You need to solve a separable differential equation (Calc II material))

6 Compare Bisection method and Newton's method. List their pros and cons. Think of a way to combine Bisection method and Newton's method to overcome the drawbacks of the Newton's method.

Chapter 2

Floating point arithmetic

Try add up 0.1 and 0.2 in Python, you will get

```
>>> 0.1 + 0.2
0.30000000000000004
```

This can cause serious issues and create bugs. See Section 5.1 of [1]. The above error is caused by rounding in floating point numbers, with which numerical analysis is traditionally concerned.

Floating point numbers are represented in terms of a base β, a precision p, and an exponent e. For example, in base 10, 0.34 can be represented as 3.4×10^{-1} or 34×10^{-2}. In order to guarantee uniqueness, we take a floating point number to have the specific form.

$$\pm (d_0 + d_1\beta^{-1} + \cdots + d_{p-1}\beta^{-(p-1)})\beta^e, \tag{2.1}$$

where each d_i is an integer in $[0, \beta)$ and $d_0 \neq 0$. Such a representation is said to be a *normalized floating point number*. Here are a few examples where $\beta = 10, p = 5$ and e is in the range $-10 \leq e \leq 10$:

$$1.2345 \times 10^8, \qquad 3.4000 \times 10^{-2}, \qquad -5.6780 \times 10^5$$

In these examples, we are using base $\beta = 10$ to ease into the material. Computers work more naturally in binary (base 2). **Please review binary representation**. When $\beta = 2$, each digit d_i is 0 or 1 in equation (2.1). To review, we have

- $11.0101_2 = 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} = 2 + 1 + 0.25 + 0.0625 = 3.3125$
- $1100_2 = 1.1_2 \times 2^3 = (1 \cdot 2^0 + 1 \cdot 2^{-1}) \cdot 2^3 = 12$
- $0.01_2 = 1.00_2 \times 2^{-2} = 0.25$

The number 0.1 has an exact representation in base 10. However, 0.1 does not have a finite binary expansion. In fact

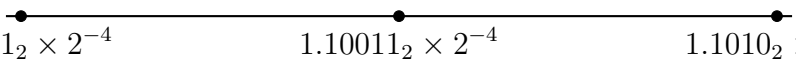
$$0.1 = 0.0\overline{0011}_2$$

meaning $0.1 = 2^{-4} + 2^{-5} + 2^{-8} + 2^{-9} + \cdots$. The representation has to be rounded at some point, and this is causing the computation inaccuracy at the beginning of this section.

There are usually 4 rounding models: Rounding down, Rounding up, Rounding towards 0, and rounding to nearest. The default IEEE standard is rounding to nearest, that is, either round down or round up, whichever is closer. In case of a tie, it is the one whose least significant (rightmost) bit is 0.

For convenience of illustration, we use examples where we retain 4 digits after the binary point. And we shall use the round to nearest model.

Example 2.1. $0.1 = 1.1001100\cdots_2 \times 2^{-4}$. But we have to round this number so that there are only 4 digits after the binary point. Rounding up will be $u = 1.1010_2 \times 2^{-4}$. Rounding down will be $d = 1.1001_2 \times 2^{-4}$. To see which one is nearest, we can compute the midpoint of u and d.



0.1 is clearly bigger than the middle point, so we pick $1.1010_2 \times 2^{-4}$ with rounding to the nearest.

Example 2.2 (Addition)**.** Suppose we retain 4 digits after the binary point. To add up two numbers with different powers, we first need to adjust one of the numbers to align the significands (use bigger power), as:

$1.1000_2 \times 2^1 \oplus 1.1001_2 \times 2^{-1} = 1.1000_2 \times 2^1 \oplus fl(0.011001_2) \times 2^1 = 1.1000_2 \times 2^1 \oplus 0.0110_2 \times 2^1 = fl(1.1110_2) \times 2^1 = 1.1110_2 \times 2^1$.

Almost all machines today use IEEE-754 floating point arithmetic, and almost all platforms map Python floats to IEEE-754 double precision. This standard has 64 bits, with 1 bit for the sign, 11 for the exponent, and 52 for the significand.

Use the IEEE-754 floating point number to represent,

$$0.1 = +1.\underbrace{1001100110011001100110011001100110011001100110011010_2}_{52 \text{ digits}} \times 2^{-4},$$

which is approximately `0.1000000000000000055511151231257827021181583404541015625` in decimal.

2.1 Fundamental Axiom of Floating Point Arithmetic

If we use $fl(x)$ to represent the floating point number of x, then we always obey that $fl(x) = x(1+\epsilon)$ such that $|\epsilon| \leq \epsilon_m$, where ϵ_m is the machine precision. This can also be expressed as $\left|\dfrac{fl(x)-x}{x}\right| \leq \epsilon_m$, and interpreted as "relative error is bounded by the machine precision."

Example 2.3. Let us use base 10 with 1 digit precision as an example, in which case the machine precision is $\epsilon_m = 10^{-1}$. If we use the rounding to nearest method,

1.23 becomes 1.2. Relative error=$\left|\dfrac{1.23-1.2}{1.21}\right| \approx 0.02$

10.61, is first expressed as 1.061*10, then round to 1.1*10. Relative error=$\left|\dfrac{10.61-11}{10.61}\right| \approx 0.04$

Not only the rounding needs to have small relative error, all the operations $(+, -\times, \div)$ should be stable in the same way:

Fundamental axiom of floating point arithmetic: Let $*$ be one of the operations $(+, -, \times, \div)$, and let $\circledast$ be its floating point analogue, then for all **floating point numbers** x, y, we have that

$$x \circledast y = (x * y)(1+\epsilon), \text{ for some } \epsilon \text{ such that } |\epsilon| \leq \epsilon_m.$$

or equivalently

$$\left|\frac{x \circledast y - x * y}{x * y}\right| \leq \epsilon_m.$$

Let us check quickly check this axiom with the operation + on one example (insanity check).

Example 2.4. We assume a machine, base 2, can only have 4 digits precision after the binary point, rounding to the nearest. Let $x = 1.1010_2 \times 2^{-4}$ and $y = 1.1010_2 \times 2^{-3}$ be two floating point numbers. (x is approximately 0.1 by Example 2.1 and y is approximately 0.2 in decimal.)

Similar to Example 2.2, we first rewrite x as $0.11010_2 \times 2^{-3}$, which rounds to $0.1110_2 \times 2^{-3}$ (break the tie by picking the one whose right most bit is 0).

$x \oplus y = fl(0.1110_2 \times 2^{-3} + 1.1010_2 \times 2^{-3}) = fl(10.1000_2 \times 2^{-3}) = 1.0100_2 \times 2^{-2} = 0.3125$.

The relative error for this addition is

$$\left|\frac{x \oplus y - (x+y)}{x+y}\right| = \left|\frac{1.0100_2 \times 2^{-2} - 1.001110_2 \times 2^{-2}}{1.001110_2 \times 2^{-2}}\right| = \frac{0.000010_2}{1.001110_2} \approx 0.0256 < 2^{-4} = \epsilon_m.$$

Example 2.5. If we assume a machine, base 2, can only have 4 digits precision after the binary point. This is how it adds up 0.1 and 0.2.

Step 1: 0.1 is rounded to $1.1010_2 \times 2^{-4} = fl(0.1)$.

Step 2: 0.2 is rounded to $1.1010_2 \times 2^{-3} = fl(0.2)$.

Step 3: $fl(0.1) \oplus fl(0.2) = 0.3125$ as shown in Example 2.4.

The overall relative error is $0.0125/0.3 \approx 0.042$.

To sum up, if we use $m(\cdot)$ to indiate the machine output using floating pont numbers, then $m(x+y) = fl(x) \oplus fl(y) = (x(1+\epsilon_1) + y(1+\epsilon_2))(1+\epsilon_3)$

Chapter 2 Exercises

1. Convert binary to decimal.
   (a) 110_2
   (b) 11.0101_2
2. Directly add and subtract in binary (No rounding involved).
   (a) $0.111_2 + 11.101_2$
   (b) $1.0101_2 - 1.10_2$
   (c) $1.1010_2 \times 2^{-4} + 1.1010_2 \times 2^{-3}$ as in Example 2.4. Convert it to decimal with a calculator.
3. Compute the **normalized** floating point numbers
   (a) 10.3456, base 10, retain 4 digits after point, rounding up.
   (b) 1.0101_2, base 2, retain 2 digits after point, rounding up.
   (c) 1.0101_2, base 2, retain 2 digits after point, rounding down.
   (d) 1.0101_2, base 2, retain 2 digits after point, rounding to the nearest (decide which one is closer by finding the midpoint of answers in (b) and (c)).

4. Compute the machine result of 0.1+0.2 if we retain 2 digits precision after the binary point.

5. *In Python

```
>>>  (0.1 + 0.2) + 0.3
0.6000000000000001
>>>  0.1 + (0.2 + 0.3)
0.6
```

   Explain this phenomenon with a more limited machine where we only retain 2 digits after the binary point.

6. This problem is from Chapter 5 problem 15 of [1].

   In the 1991 Gulf War, the Patriot misslie defense system failed due to roundoff error. The troubles stemmed from a computer that performed the tracking calculations with an internal clock whose integer values in tenths of a second were converted to seconds by multiplying by a 24-bit binary approximation to 0.1:

$$0.1 \approx 0.00011001100110011001100_2$$

   (a) Convert the binary number above to a fraction. Call it x.

   (b) Compute the absolute difference between 0.1 and x.

   (c) What is the time error in seconds after 100 hours of operation (i.e., the value of $|360,000 - 3,600,000x|$)?

   On February 25, 1991, a Patriot battery system, which was to protet the Dhahran Air Base, had been operating for over 100 consecutive hours. The roundoff error caused the system not to track an incoming Scud missle, which slipped through the defense system and detonated on US Army barracks, killing 28 American soldiers.

Chapter 3

Linear Algebra Essentials

3.1 Vector

The letters towards the end of the alphabet, like x, y, z, usually indicate vectors. They are considered as column vectors unless otherwise stated. For a vector x, x_i will denote its ith coordinate. The square brackets $[\cdot]$ is for matrices. Since vectors are a special kind of matrices, we have

- $x = [x_1, x_2 \cdots, x_n]$ is a row vector.
- $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ is a column vector.
- $x = [x_1, x_2 \cdots, x_n]^T$ is a column vector.
- $x = (x_1, x_2 \cdots, x_n)$ is a column vector. Yes, if we use parenthesis, it is a column vector!

In consideration of saving space, you will see the last two expressions often.

Let $x = (x_1, x_2 \cdots, x_n)$ and $y = (y_1, y_2, \cdots, y_n)$ be two real vectors in $\mathbb{R}^n$, then the *inner product* of x and y is defined as

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i.$$

Notice $\langle x, y \rangle = x^T y$ (or $y^T x$).

Two vectors x, y are called *orthogonal* if $\langle x, y \rangle = 0$. x is called a unit (norm) vector if $\|x\| = 1$. It is obvious that $\langle x, x \rangle = \|x\|_2^2$. We use the notion $\|\cdot\|_2$ here because in this course, we will learn a more general notion of norm.

For several vectors $\{v_1, v_2, \cdots, v_n\}$, they are called

- *Orthogonal* if $v_i^T v_j = 0, \forall\ i \neq j$.
- *Orthonormal* if $\begin{cases} v_i^T v_j = 0, & \forall\ i \neq j \\ v_i^T v_i = 1, & \forall\ 1 \leq i \leq n \end{cases}$

Definition 3.1. [Vector norm] A *norm* for a vector is a function $\|\cdot\|$ satisfying
(i) $\|v\| \geq 0$ for all vector v, with equality if and only if $v = 0$ (positive definite);
(ii) $\|\alpha v\| = |\alpha|\|v\|$ for any scalar α and any vector v (scalable);
(iii) $\|v + w\| \leq \|v\| + \|w\|$ for all vector v, w (triangle inequality).

The most common norm is probably the class of ℓ_p norms. For any $p \geq 1$, define:

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

Among the ℓ_p norm class, the frequently used ones are the ℓ_1 norm:

$$\|x\|_1 := \sum_{i=1}^{n} |x_i|,$$

the ∞-norm

$$\|x\|_\infty = \max_i |x_i|,$$

and of course the ℓ_2 norm (Euclidean norm) that we are very familiar with.

Example 3.2. Let $x = (1, 1, -2)$, then $\|x\|_2 = \sqrt{6}, \|x\|_1 = 4, \|x\|_\infty = 1$.

3.2 Matrix

$A = (a_{ij}) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \\ \vdots \\ r_m^T \end{bmatrix} = [c_1, c_2, \cdots, c_n]$ is an $m \times n$ matrix, where r_i^T is the ith row of this matrix and c_j is the jth column of this matrix.

Let $x = [x_1, \cdots, x_n]^T$, then there are three ways to view the multiplication Ax.

(1) $Ax = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{mn}x_n \end{bmatrix}$

(2) $Ax = \begin{bmatrix} r_1^T \\ r_2^T \\ \vdots \\ r_m^T \end{bmatrix} x = \begin{bmatrix} r_1^T x \\ r_2^T x \\ \vdots \\ r_m^T x \end{bmatrix}$.

(3) $Ax = [c_1, c_2, \cdots, c_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \sum_{i=1}^{n} x_i c_i$. Ax is a linear combination of columns of A.

The null space or kernel of A is defined as

$$N(A) = \{x : Ax = 0\}$$

Given $A \in \mathbb{R}^{n\times n}$, if $Ax = \lambda x$ for some $x \neq 0$, then x is an eigenvector of A, and λ is the corresponding eigenvalue. For example, we have $\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = 3\begin{bmatrix} 1 \\ 1 \end{bmatrix}$, which means that $[x, 1]^T$ is an eigenvector of $A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$, with the corresponding eigenvalue 3.

To figure out all the eigenvalues and its corresponding eigenvectors of A, we need to first find roots of $\det(A - \lambda I)$, and second find the null space of $A - \lambda I$ for each root.

Example 3.3. Find eigenvalues and eigenvectors of $A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$.

$\det \begin{bmatrix} 1-\lambda & 2 \\ 2 & 1-\lambda \end{bmatrix} = (1-\lambda)^2 - 4 = (1-\lambda-2)(1-\lambda+2) = (-1-\lambda)(3-\lambda)$.

$r_1 = -1$, solve $\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} x = 0$, $u_1 = [1, -1]^T$.

$\lambda_2 = 3$, solve $\begin{bmatrix} -2 & 2 \\ 2 & -2 \end{bmatrix} x = 0, u_2 = [1, 1]^T$.

$A \in \mathbb{R}^{n\times n}$ is symmetric if $A = A^T$.

$A \in \mathbb{R}^{n\times n}$ is called positive definite if it is symmetric and all eigenvalues are positive.

$A \in \mathbb{R}^{n\times n}$ is called positive semidefinite if it is symmetric and all eigenvalues are nonnegative.

$A \in \mathbb{R}^{n\times n}$ is called diagonalizable if we can find n independent eigenvectors. (This is not the official definition of diagonalizability, but it's an equivalent one.) A symmetric matrix must be diagonalizable. Figure 3.1 shows the ownerships of these matrices.

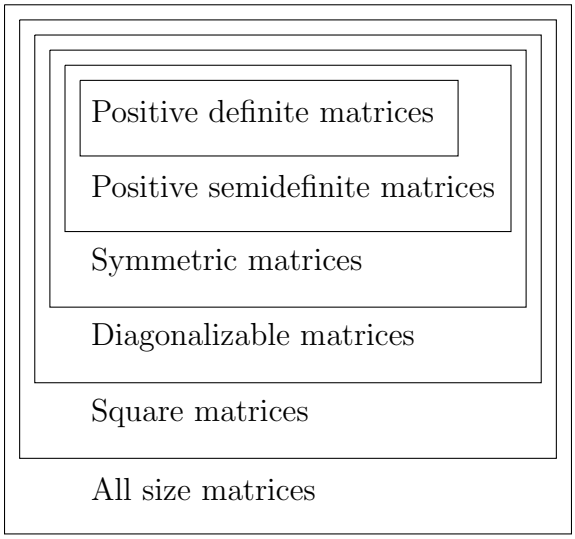


Figure 3.1: Matrices

3.3 Basis

Basis is a very important concept for a vector space. $\{b_1, b_2, \cdots, b_K\}$ is a basis of the vector space V if $\text{span}\{b_1, \cdots, b_K\} = V$ and $\{b_1, b_2, \cdots, b_K\}$ are linearly independent. The vectors in a basis

can be considered the building blocks of all vectors in a vector space. For example, using the columns of the identity matrix, $\left\{ e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, e_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots, e_K = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \right\}$, which is called the *canonical basis of* $\mathbb{R}^K$, we are able to generate all vectors in $\mathbb{R}^K$:

$$\text{For any } x = (x_1, \cdots, x_K) \in \mathbb{R}^K, \quad x = x_1 e_1 + \cdots x_K e_K.$$

Moreover, this "generation" (i.e. the coefficients in front of e_i) is unique due to independence. So another definition of basis is: $\{b_1, b_2, \cdots, b_K\}$ is a basis of the vector space V if every vector in V can be uniquely written as a linear combination of $\{b_1, b_2, \cdots, b_K\}$. Moreover,

$$\dim V = \text{number of vectors in a basis of } V$$

Example 3.4. $\{(1,2,0),(1,1,0)\}$ is a basis of the xy plane in $\mathbb{R}^3$.

Example 3.5. $\{(1,2),(1,1)\}$ is a basis of $\mathbb{R}^2$. $\{\frac{1}{\sqrt{2}}(1,1), \frac{1}{\sqrt{2}}(-1,1)\}$ is another basis of $\mathbb{R}^2$.

Notation: Sometimes we will also put the basis vectors as columns of a matrix B and call B is a basis (of some vector space). For example, the identity matrix $I = [e_1, \cdots, e_K]$ is the canonical basis of $\mathbb{R}^K$. $\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ is a basis of $\mathbb{R}^2$ as claimed in Example 3.5.

Remark 3.6. $B' = \{b_1, \cdots, b_K\}$ and $B = [b_1, \cdots, b_K]$ can mean the same basis, so we will abuse the notation and equate B' and B when convenient. But make sure you understand that there is a difference between [and {. { indicates that it is a set, so B' means a set of K vectors. [is for matrix so B is a matrix whose columns are b_i.

There are infinitely many bases of a vector space, and some are better at representing vectors than others. In Example 3.5, the second basis $U_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ is a better basis because it is an *orthonormal basis*.

Definition 3.7. $U = \{u_1, \cdots, u_K\}$ is an orthonormal basis (ONB) of V if it is a basis of V and the vectors are orthonormal.

U is an ONB if and only if $U^T U = I_K$: the $K \times K$ identity matrix. This is a good way to check whether a matrix is an ONB. (Check U_2 is an ONB for yourself.)

Given a basis $B = \{b_1, \cdots, b_K\}$, we know x can be written as a linear combination of these basis vectors: $x = \sum_{i=1}^{K} c_i b_i$. In order to figure out the coefficients c_i, we need to solve a linear system (solve c from $Bc = x$). This can be annoying. With an ONB, it is a lot better and we simply have

$$\text{If } U = \{u_1, \cdots, u_K\} \text{ is an ONB and } x = \sum_{i=1}^{K} c_i u_i, \text{ then } c_i = u_i^T x = \langle u_i, x \rangle. \tag{3.1}$$

3.4 Matrix norm

Chapter 4

Appendix: Singular Value Decomposition

A positive semidefinite (PSD) matrix is a symmetric matrix whose eigenvalues are nonnegative. If A is positive semidefinite, then A can be diagonalized as

$$A = QDQ^{-1} = QDQ^T, \tag{4.1}$$

where Q is an orthonormal matrix, and D is a diagonal matrix whose diagonals are the eigenvalues of A (hence all nonnegative).

This decomposition where the diagonals of D are all nonnegative numbers is a prilividge of PSD matrices. The Singular Value Decomposition (SVD) tries to generalize the process of (4.1) to all matrices of any size, of course with some compromise.

Definition 4.1. Given any matrix A, whose size is $m \times n$, A can be factored as

$$A = U\Sigma V^T,$$

where U is an $m \times m$ orthonormal matrix, V is an $n \times n$ orthonormal matrix, and Σ is an $m \times n$ diagonal matrix. The entries on the diagonal of Σ are called the *singular values* of A. The singular values are always nonnegative. The singular value are often listed in descending order: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{m,n\}} \geq 0$.

Example 4.2. $$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boxed{2} & 0 \\ 0 & \boxed{0} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^T.$$

The singular values are 2,0.

Example 4.3. $$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \boxed{\sqrt{2}} & 0 \\ 0 & \boxed{\sqrt{2}} \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}^T.$$

The singular values are $\sqrt{2}, \sqrt{2}$, which are quite different from its eigenvalues (they are complex numbers). In general, singular values are very different from eigenvalues unless it is a symmetric matrix (see Example 4.6).

Example 4.4. $\begin{bmatrix} 0 & 0 & -2 & -2 \\ 1.5 & 1.5 & 2.5 & 2.5 \\ -3 & -3 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -1/3 & 2/3 & -2/3 \\ 2/3 & -1/3 & -2/3 \\ -2/3 & -2/3 & -1/3 \end{bmatrix} \begin{bmatrix} \boxed{6} & 0 & 0 & 0 \\ 0 & \boxed{3} & 0 & 0 \\ 0 & 0 & \boxed{0} & 0 \end{bmatrix} \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & -0.5 & -0.5 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & -0.5 \end{bmatrix}^T$.

The singular values are 6,3,0.

It can be easily proven that **the rank of A will always equal to the number of positive singular values**. The columns of U and V also play an important role. Suppose A has rank r, then its nonzero singular values can be expressed descendingly as $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. The SVD can be expressed as

$$A = [u_1, u_2, \cdots, u_m] \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \sigma_r & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} [v_1, v_2 \cdots, v_n]^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T.$$

The SVD in Example 4.4 can therefore be rewritten as

$$\begin{bmatrix} 0 & 0 & -2 & -2 \\ 1.5 & 1.5 & 2.5 & 2.5 \\ -3 & -3 & -1 & -1 \end{bmatrix} = 6 \begin{bmatrix} -1/3 \\ 2/3 \\ -2/3 \end{bmatrix} \begin{bmatrix} 1/2 & 1/2 & 1/2 & 1/2 \end{bmatrix} + 3 \begin{bmatrix} 2/3 \\ -1/3 \\ -2/3 \end{bmatrix} \begin{bmatrix} 1/2 & 1/2 & -1/2 & -1/2 \end{bmatrix}.$$

The SVD of a matrix A can pretty much tell every piece of useful information of A.

1. rank(A)=number of nonzero singular values of A=r.
2. $C(A) = \text{span}\{u_1, u_2, \cdots, u_r\}$.
3. $R(A) = \text{span}\{v_1, v_2, \cdots, v_r\}$.
4. $N(A) = \text{span}\{v_{r+1}, v_{r+2}, \cdots, v_n\}$.

Use Example 4.4 again, the matrix is rank 2, and its column space has a basis $\{u_1, u_2\} = \{\frac{1}{3}[-1, 2, -2]^T, \frac{1}{3}[2, -1, -2]^T\}$. Its row space has a basis $\{v_1, v_2\} = \{\frac{1}{2}[1, 1, 1, 1]^T, \frac{1}{2}[1, 1, -1, -1]^T\}$. Its null space has a basis $\{v_3, v_3\} = \{\frac{1}{2}[1, -1, -1, 1]^T, \frac{1}{2}[1, -1, 1, -1]^T\}$.

Given the following SVD,

$$A = \begin{bmatrix} \frac{1}{\sqrt{5}} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{20}} \\ \frac{1}{\sqrt{5}} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{\sqrt{20}} \\ \frac{1}{\sqrt{5}} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{20}} \\ \frac{1}{\sqrt{5}} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{\sqrt{20}} \\ \frac{1}{\sqrt{5}} & 0 & 0 & 0 & \frac{-4}{\sqrt{20}} \end{bmatrix} \begin{bmatrix} 10 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.01 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}^T$$

We know that rank(A)=3, but it seems like A is almost rank 2 because the third singular value 0.01 can be neglected. Indeed, $A_2 = U \begin{bmatrix} 10 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$ is called the rank-2 approximation of A. After computation,

$$A = \begin{bmatrix} 3.52 & 2.81 & 0.005 \\ 3.52 & 2.81 & -0.005 \\ 2.81 & 3.52 & -0.005 \\ 2.81 & 3.52 & 0.005 \\ 3.16 & 3.16 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 3.52 & 2.81 & 0 \\ 3.52 & 2.81 & 0 \\ 2.81 & 3.52 & 0 \\ 2.81 & 3.52 & 0 \\ 3.16 & 3.16 & 0 \end{bmatrix}.$$

A_2 is very close to A as expected. The difference can sometimes be considered noise in applications. In combination with the earlier discussion, we can say that the columns of A are approximately on span$\{u_1, u_2\}$ = span$\{[1,1,1,1,1]^T, [1,1,-1,-1,0]^T\}$. The rows of A are approximately on span$\{v_1, v_2\}$ = span$\{[1,1,0]^T, [1,-1,0]^T\}$.

Notice the second singular value is also pretty small compared to 10, so we can approximate A by its rank-1 approximation $A_1 = U \begin{bmatrix} 10 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = \begin{bmatrix} 3.16 & 3.16 & 0 \\ 3.16 & 3.16 & 0 \\ 3.16 & 3.16 & 0 \\ 3.16 & 3.16 & 0 \\ 3.16 & 3.16 & 0 \end{bmatrix}$. The error is bigger, but perhaps still acceptable in some circumstances. In this case, we can say that the columns of A are approximately on the line span$\{u_1\}$ = span$\{[1,1,1,1,1]^T\}$. The rows of A are approximately on the line span$\{v_1\}$ = span$\{[1,1,0]^T\}$.

In general, If $A = U \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \sigma_r & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$, then given any $s < r$,

$$A_s = U \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_s & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix} V^T = \sum_{i=1}^{s} \sigma_i u_i v_i^T$$

is called the closest rank-s approximation of A.

It is not an easy task to compute the SVD of A, but it is easy to compute its singular values. **For any matrix A, the singular valules of A are the square roots of eigenvalues of A^TA (or AA^T).**

Example 4.5. Compute the singular values of ...

Moreover, when A is symmetric, we can easily compute its SVD.

Example 4.6. Compute the SVD of $A = \begin{bmatrix} -7 & 6 \\ 6 & 2 \end{bmatrix}$ and find its rank-1 approximation.

Step 1: Find the eigenvalues and orthonormal eigenvectors of A

$r_1 = -10, u_1 = \frac{1}{\sqrt{5}}[-2, 1]^T$.

$r_2 = 5, u_2 = \frac{1}{\sqrt{5}}[1, 2]^T$.

Step 2: Diagonalize A with an orthonormal U.

$$A = UDU^T = \frac{1}{\sqrt{5}} \begin{bmatrix} -2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} -10 & 0 \\ 0 & 5 \end{bmatrix} \frac{1}{\sqrt{5}} \begin{bmatrix} -2 & 1 \\ 1 & 2 \end{bmatrix}^T \tag{4.2}$$

Step 3: SVD

Equation (4.2) is not SVD because we need diagonals to be all positive. In order to get 10, we can simply change u_1 to $-u_1$.

$$A = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} 10 & 0 \\ 0 & 5 \end{bmatrix} \frac{1}{\sqrt{5}} \begin{bmatrix} -2 & 1 \\ 1 & 2 \end{bmatrix}^T.$$

Finally, rank-1 approximation is $\frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} 10 & 0 \\ 0 & 0 \end{bmatrix} \frac{1}{\sqrt{5}} \begin{bmatrix} -2 & 1 \\ 1 & 2 \end{bmatrix}^T = \begin{bmatrix} -8 & 4 \\ 4 & -2 \end{bmatrix}$

4.1 Data compression/dimension reduction

Bibliography

[1] Anne, Greenbaum, and Timothy P. Chartier. *Numerical methods: design, analysis, and computer implementation of algorithms*. Princeton University Press, 2012.