Contents

List of Tables

List of Figures

List of Algorithms

Part I

Assignments

Assignment 1

Introduction

Exercise 1.1 — Metric space

Let X be a metric space

1. Show that $\varnothing$ and X are closed in X.

   Proof. According to the definition of openness, a subset U of X is *open* if

   $$\forall x \in U, \exists \varepsilon > 0 \text{ such that } (y \in X, d(x,y) < \varepsilon) \implies y \in U$$

   And the definition of closed set is a complement of an open set. Since $\varnothing$ does not have any elements, all statements with $x \in \varnothing$ is trivially true. Thus, $\varnothing$ is open, which implies its complement

   $$X \backslash \varnothing = X$$

   is closed. Then, we prove X is also open. According to the definition,

   $$\forall x \in X, \exists \varepsilon > 0 \text{ such that } (y \in X, d(x,y) < \varepsilon) \implies y \in X$$

   This is also trivially true since we have $y \in X \implies y \in X$. So X is open, and its complement

   $$X \backslash X = \varnothing$$

   is closed. □

2. Prove that the intersection of a collection of closed subsets of X is closed in X.

   Proof. By theorem, a metric space is a topological space, so we can directly apply the property that

   > Any union, finite or not, of open subsets of X is open.

   Then we take the complement and apply de Morgan's law to get: any intersection, finite or not, of closed subsets of X is closed. □

3. Prove that the union of a finite collection of closed subsets of X is closed in X.

Proof. Similarly, we quote another property of topological spaces

> Any intersection of finitely many open subsets of X is open.

Then we take the complement and apply de Morgan's law to get: the union of finitely many closed subsets of X is closed. □

Exercise 1.2 — Continuity

For each of the following question prove your result.

1. Give an example of function which is continuous but not uniformly continuous;

   Example: $f(x) = x^2$.

   Proof. Continuity can be proved by choosing $\delta = \frac{\varepsilon}{1+|x+x_0|}$ in the $\varepsilon - \delta$ definition.

   $$\begin{aligned} &\forall \varepsilon > 0, \exists \delta = \frac{\varepsilon}{1+|x+x_0|} > 0 \text{ such that } |x - x_0| < \delta \\ &\implies |f(x) - f(x_0)| = |x - x_0| \cdot |x + x_0| < \frac{\varepsilon}{1+|x+x_0|} \cdot |x + x_0| < \varepsilon \end{aligned}$$

   And the property of uniform continuous can be violated by choosing $x = \frac{\varepsilon}{\delta}, y = x + \frac{\delta}{2}$, which produces

   $$|x - y| = \frac{\delta}{2} < \delta \text{ while } |f(y) - f(x)| = \left| x\delta + \frac{\delta^2}{4} \right| = \left| \varepsilon + \frac{\delta^2}{4} \right| > \varepsilon$$

   So this is a continuous but not a uniformly continuous function. □

2. Give an example of function which is uniformly continuous but not Lipschitz continuous.

   Example: $f(x) = \sqrt{x}$ on its domain $\mathbb{R}^+$.

   Proof. Uniform continuity can be proved by choosing $\delta = \varepsilon^2$ in the $\varepsilon - \delta$ definition.

   $$\begin{aligned} &\forall \varepsilon > 0, \exists \delta = \varepsilon^2 > 0 \text{ such that } |x - y| < \delta \\ &\implies |f(x) - f(y)| = |\sqrt{x} - \sqrt{y}| \\ &\leq \sqrt{|\sqrt{x} - \sqrt{y}| \cdot |\sqrt{x} + \sqrt{y}|} \\ &= \sqrt{|x - y|} = \sqrt{\varepsilon^2} = \varepsilon \end{aligned}$$

   And the property of Lipschitz continuous can be violated by writing

   $$|f(x) - f(y)| = |\sqrt{x} - \sqrt{y}| = \frac{1}{\sqrt{x} + \sqrt{y}} |x - y|$$

By choosing $y = 0$, we need the Lipschitz constant to be the supreme

$$K \geq \sup_{x>0} \frac{1}{\sqrt{x}}$$

However,

$$\lim_{x \to 0^+} = +\infty$$

which means we can't find such a Lipschitz constant. So this is a uniformly continuous but not a Lipschitz continuous function. □

Exercise 1.3 — Cardinality

1. Prove that $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{Q}$ have the same number of elements.

   Proof. We can construct a bijective function from $\mathbb{N}$ to $\mathbb{Z}$:

   $$f(n) = (-1)^n \lfloor \frac{n+1}{2} \rfloor \quad n \in \mathbb{N}$$

   Injectivity is trivial because this function only returns an integer. Surjectivity is proved by

   $$n = \begin{cases} 4k & \to 2k \\ 4k+1 & \to -(2k+1) \\ 4k+2 & \to 2k+1 \\ 4k+3 & \to -2(k+1) \end{cases}$$

   which covers the positive even numbers, negative odd numbers, positive add numbers, negative even numbers and zero, whose union is $\mathbb{Z}$ and are disjoint. Thus, the mapping is both injective and surjective, which is bijective. Then, we can construct a bijective mapping $f : \mathbb{N}^* \times \mathbb{N}^* \to \mathbb{Q}^+$

   $$\begin{array}{ccccc} 1/1 & 1/2 & 1/3 & 1/4 & \cdots \\ 2/1 & 2/2 & 2/3 & 2/4 & \cdots \\ 3/1 & 3/2 & 3/3 & 3/4 & \cdots \\ 4/1 & 4/2 & 4/3 & 4/4 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{array}$$

   This mapping is trivially injective since it returns a fraction of two natural numbers. It is also surjective because every positive rational number can be written as a ratio of two integers. Then we can count diagonally to construct the bijective mapping $g : \mathbb{N}^* \to \mathbb{Q}^+$. And the negative integers can use the same mapping with negative signs on the range, so that all the negative rational numbers are mapped. Finally, we map the zero in $\mathbb{Z}$ to the zero in $\mathbb{Q}$ to construct the whole map, which gets the surjectivity. Thus, this map is also bijective. The first mapping concludes $\text{Card}(\mathbb{N}) = \text{Card}(\mathbb{Z})$, and the second mapping concludes $\text{Card}(\mathbb{Z}) = \text{Card}(\mathbb{Q})$. The combined result is

   $$\text{Card}(\mathbb{N}) = \text{Card}(\mathbb{Z}) = \text{Card}(\mathbb{Q})$$

   □

2. Prove that $\mathbb{R}$ has more elements than $\mathbb{N}$.

   Proof. We prove that the subset $[0,1] \subset \mathbb{R}$ has more elements than $\mathbb{N}$. Suppose we can construct a bijective map from $\mathbb{N}$ to $[0,1]$, then

   $$\begin{aligned} n_1 &\to 0.a_{11}a_{12}a_{13}a_{14}\cdots \\ n_2 &\to 0.a_{21}a_{22}a_{23}a_{24}\cdots \\ n_3 &\to 0.a_{31}a_{32}a_{33}a_{34}\cdots \\ &\vdots \end{aligned}$$

   where $a_{ij} \in \{0,1,2,3,4,5,6,7,8,9\}$. Then we can always find an element from $[0,1]$ that is not mapped to. Take $b_1 \neq a_{11}, b_2 \neq a_{22}, \cdots, b_i \neq a_{ii}, \cdots$, $0.b_1b_2\cdots b_i\cdots \in [0,1]$ differs from all the elements in the range of the mapping, so the mapping is not surjective. This contradicts to the bijection assumption. Thus the bijection does not exist, and $[0,1]$ has more elements than $\mathbb{N}$, which implies $\mathbb{R}$ has more elements than $\mathbb{N}$. □

3. Prove that $[0,1]$ has as many elements as $\mathbb{R}$.

   Proof. We find a bijective mapping $f : \mathbb{R} \to (0,1)$:

   $$f(x) = \frac{1}{2} + \frac{1}{\pi}\arctan(x)$$

   This mapping is monotone so that its inverse exists, which is clearly bijective. This implies $(0,1)$ has as many elements as $\mathbb{R}$, and adding two more elements $\{0,1\}$ to infinite set does not change the cardinality. Thus,

   $$\text{Card}([0,1]) = \text{Card}(\mathbb{R})$$

   □

Exercise 1.4 — Cauchy-Schwartz inequality

In the lecture the Cauchy-Schwartz inequality was proven for real numbers, prove it for the complex numbers.

Theorem 1.1 (Cauchy-Schwartz Inequality)**.** *Let V be an inner product space over $\mathbb{C}$ and $u, v \in V$. Then*

$$|\langle u, v\rangle| \leq ||u|| \cdot ||v||.$$

This equality occurs if and only if one of u, v is a scalar multiple of the other.

Proof. If $v = 0$, both sides are zeros, so the equality is trivially true. If $v \neq 0$, we define

$$\lambda = \frac{\langle u, v\rangle}{||v||^2}$$

Then,

$$\begin{aligned}
0 &\leq ||u - \lambda v||^2 \\
&= \langle u, u\rangle - \langle u, \lambda v\rangle - \langle \lambda v, u\rangle + \langle \lambda v, \lambda v\rangle \\
&= \langle u, u\rangle - \lambda\langle u, v\rangle - \bar{\lambda}\langle u, v\rangle + \lambda\bar{\lambda}\langle v, v\rangle \\
&= ||u||^2 - \frac{|\langle u, v\rangle|^2}{||v||^2} - \frac{|\langle u, v\rangle|^2}{||v||^2} + \frac{|\langle u, v\rangle|^2}{||v||^2} \\
&= ||u||^2 - \frac{|\langle u, v\rangle|^2}{||v||^2} \\
\implies & ||u||^2 \geq \frac{|\langle u, v\rangle|^2}{||v||^2}
\end{aligned}$$

Since all the squares are nonnegative, we get

$$||u||||v|| \geq |\langle u, v\rangle|^2$$

□

Exercise 1.5 — Linear algebra

1. Let f be a linear map from a vector space V_1 into a vector space V_2. Show that the dimension of V_1 is the sum of the dimensions of the kernel and of the image of f. This result is called the rank nullity theorem.

   Proof. Suppose $\{u_1, \ldots, u_m\}$ forms a basis of $\ker f$. We can extend this to form a basis of V_1: $\{u_1, \ldots, u_m, w_1, \ldots, w_n\}$. Since the dimension of $\ker f$ is m and the dimension of V_1 is $m+n$, it suffices to show that the dimension of image of f (V_2) is n. We can show that $\{f(w_1), \ldots, f(w_n)\}$ forms a basis form a basis of V_2. There exist unique scalars such that

   $$\begin{aligned}
   v &= a_1u_1 + \cdots a_mu_m + b_1w_1 + \cdots + b_nw_n \\
   &\implies f(v) = a_1f(u_1) + \cdots + a_mf(u_m) + b_1f(w_1) + \cdots + b_nf(w_n) \\
   &\implies f(v) = b_1f(w_1) + \cdots + b_nf(w_n)
   \end{aligned}$$

   Thus, $\{f(w_1), \cdots, f(w_n)\}$ spans V_2. Then, we have to show these vectors are linearly independent to ba a basis of V_2.

   $$\begin{aligned}
   c_1f(w_1) + \cdots + c_nf(w_n) = 0 &\iff f(c_1w_1 + \cdots + c_nw_n) = 0 \\
   &\implies c_1w_1 + \cdots + c_nw_n \in \ker f
   \end{aligned}$$

   Then since u_i span $\ker f$, there exists a set of scalars d_i such that

   $$c_1w_1 + \cdots + c_nw_n = d_1u_1 + \cdots + d_mu_m$$

   But since $\{u_1, \ldots, u_m, w_1, \ldots, w_n\}$ form a basis of V_1, all c_i, d_i must be zeros. Therefore, $\{f(w_1), \ldots, f(w_n)\}$ is linearly independent and forms a basis of V_2. This proves that the dimension of V_2 is n as desired. □

2. Prove that the composition of two linear maps is a linear map.

   Proof. Suppose the two linear maps are f and g, and a scalar $a \in \mathbb{K}$

   $$\begin{aligned} f(x+ay) &= f(x) + af(y) \\ g(f(x+ay)) &= g(f(x) + af(y)) = g(f(x)) + ag(f(y)) \end{aligned}$$

   This is exactly the linearity of the composition $g(f)$. □

3. Prove that the inverse of a linear map is a linear map.

   Proof. Suppose the linear map is f and its inverse is f^{-1}, and a scalar $a \in \mathbb{K}$.

   $$\begin{aligned} f(x+ay) &= f(x) + af(y) \\ f^{-1}(f(x+ay)) &= f^{-1}(f(x) + af(y)) \\ x + ay &= f^{-1}(f(x) + af(y) \end{aligned}$$

   But we can write x as $f^{-1}(f(x))$, and y as $f^{-1}(f(y))$.

   $$f^{-1}(f(x)) + af^{-1}(f(y)) = f^{-1}(f(x) + af(y))$$

   This is exactly the linearity of f^{-1}. □

4. Consider $\mathcal{C}^\infty(\mathbb{R})$, the set of the functions that are infinitely differentiable over $\mathbb{R}$. We equip $\mathcal{C}^\infty(\mathbb{R})$ of the supremum norm.

   (a) Show that $f(x) = \sin(nx)/n \in \mathcal{C}^\infty(\mathbb{R})$.

   Proof. We find the first four differentials

   $$\begin{aligned} \mathrm{d}f &= \cos(nx) \\ \mathrm{d}^2 f &= -n\sin(nx) \\ \mathrm{d}^3 f &= -n^2\cos(nx) \\ \mathrm{d}^4 f &= n^3\sin(nx) \end{aligned}$$

   Thus we get the recursive relation between the differentials.

   $$\mathrm{d}^{k+4} f = n^4 \mathrm{d}^k f$$

   Since the first four are all differentiable, all differentials are differentiable. □

   (b) Determine the differential of f.

Answer. According to the recursive relation,

$$\begin{aligned}
\mathrm{d}^{4k+1} f &= n^{4k} \cos(nx) \\
\mathrm{d}^{4k+2} f &= -n^{4k+1} \sin(nx) \\
\mathrm{d}^{4k+3} f &= -n^{4k+2} \cos(nx) \\
\mathrm{d}^{4k+4} f &= n^{4k+3} \sin(nx)
\end{aligned}$$

□

(c) Prove that differentiation, viewed as a linear map, is not continuous.

Proof. Take $x = 0$ and $y = \frac{\pi}{n}$, when $n \to \infty$, $|f(x) - f(y)| = 2n$ which goes to infinity. This contradicts to the continuity. □

(d) How does this relate to theorem 1.28?

Answer. This implies the differentiation does not have a finite norm. □

Exercise 1.6 — Pi

1. Write the pseudocode for

   (a) The approximation of π using polygons;

   Proof.

$$\pi = \lim_{n \to \infty} \frac{n}{2} \sin \frac{2\pi}{n}$$

□

   (b) The approximation of π using Machin's formula $\frac{\pi}{4} = 4 \arctan \frac{1}{5} - \arctan \frac{1}{239}$ and Taylor series;

   Proof.

$$\pi = 4 \sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1} \left[4 \cdot 5^{-(2k+1)} - 239^{-(2k+1)} \right]$$

□

2. Implement the two previous algorithms in MATLAB.

```
function [ p ] = PiApproximation1( n )
    p=4;
    t=(sqrt(2)-1); %t0
    a=t^2; %a0
    t=t/(1+sqrt(1+t^2)); %t1
    p=p-4*a; %p1
```

```
        for i=1:n
                a=2*t^3/(1-t^2);
                t=t/(1+sqrt(1+t^2));
                p=p-4*a*2^i;
        end
        end

        function [ p ] = PiApproximation2( n )
                p=0;
                for k=0:n
                        p=p+(-1)^k/(2*k+1)
            *(4*5^(-2*k-1)-239^(-2*k-1));
                end
                p=p*4;
        end
```

Assignment 2

Topology

Exercise 2.1 — Connected space

In this exercise we provide alternative definitions for a connected space. Let X be a metric space. We want to prove that the following conditions are equivalent

(i) The only subsets that are both open and closed in X are X and $\varnothing$;

(ii) It is impossible to write X as the union of two disjoint, non-empty open subsets;

(iii) It is impossible to write X as the union of two disjoint non-empty closed subsets;

(iv) There is no continuous, surjective application from X into $\{0, 1\} \subset \mathbb{R}$;

1. Prove that (i), (ii), and (iii) are equivalent.

   Proof. The sketch of the proof is to prove (i) $\iff$ (ii) and (i) $\iff$ (iii), then (i), (ii), (iii) are equivalent by transitivity.

(i) $\implies$ (ii) Assume it is possible to write X as the union of two disjoint, non-empty open subsets S_1 and S_2, *i.e.*,

$$S_1 \cup S_2 = X, S_1 \cap S_2 = \varnothing$$
$$S_1 \subset X, S_2 \subset X$$
$$S_1 \neq \varnothing, S_2 \neq \varnothing$$

By definition,

$$S_2 = X \backslash S_1$$

Since S_1 is open, its complement, S_2 is closed. But by definition, S_2 is also open. Thus, we find a subset S_2 with $S_2 \neq X, S_2 \neq \varnothing$ and is both open and closed, which causes contradiction to (i).

(ii) $\implies$ (i) Assume there is a subset S which is both open and closed with $S \subset X, S \neq \varnothing$. Since S is closed, its compliment $X \backslash S$ is open. By definition,

$$S \cup (X \backslash S) = X \quad S \cap (X \backslash S) = \varnothing$$

Since $X\backslash S$ is open, S is also open, we have found a way to write X as the union of two disjoint, non-empty open subsets, which causes contradiction to (ii).

(i) $\Longrightarrow$ (iii) Assume it is possible to write X as the union of two disjoint, non-empty closed subsets S_1 and S_2, *i.e.*,

$$S_1 \cup S_2 = X, S_1 \cap S_2 = \varnothing$$
$$S_1 \subset X, S_2 \subset X$$
$$S_1 \neq \varnothing, S_2 \neq \varnothing$$

By definition,

$$S_2 = X\backslash S_1$$

Since S_1 is closed, its complement, S_2 is open. But by definition, S_2 is also closed. Thus, we find a subset S_2 with $S_2 \neq X, S_2 \neq \varnothing$ and is both open and closed, which causes contradiction to (i).

(iii) $\Longrightarrow$ (i) Assume there is a subset S which is both open and closed with $S \subset X, S \neq \varnothing$. Since S is open, its compliment $X\backslash S$ is closed. By definition,

$$S \cup (X\backslash S) = X \quad S \cap (X\backslash S) = \varnothing$$

Since $X\backslash S$ is closed, S is also closed, we have found a way to write X as the union of two disjoint, non-empty closed subsets, which causes contradiction to (iii).

From the proofs above, we can conclude (i) $\Longleftrightarrow$ (ii) and (i) $\Longleftrightarrow$ (iii), which implies (i), (ii), (iii) are equivalent. □

2. Assume (iv) is not false and show that (iii) is false.

Proof. We can write $\{0, 1\}$ as the union of two disjoint non-empty closed subsets

$$\{0, 1\} = \{0\} \cup \{1\}$$

If there is a continuous, surjective function $f : X \to \{0, 1\}$, then we have

$$f^{-1}(\{0\}) \quad f^{-1}(\{1\})$$

are both closed in X. Since f is surjective, both of $f^{-1}(\{0\})$ and $f^{-1}(\{1\})$ are not empty. And since $\{0\}$ is complement to $\{1\}$ in the image of f,

$$f^{-1}\left(\{0\}^c\right) = \left(f^{-1}\{0\}\right)^c$$

Then, we can write X as the union of two disjoint non-empty closed subsets

$$X = f^{-1}(\{0\}) \cup f^{-1}(\{1\})$$

which cause contradiction to (iii). □

3. Assume (iii) is not false and show that (iv) is false.

Proof. Assume it is possible to write X as the union of two disjoint, non-empty closed subsets S_1 and S_2, *i.e.*,

$$S_1 \cup S_2 = X, S_1 \cap S_2 = \varnothing$$
$$S_1 \subset X, S_2 \subset X$$
$$S_1 \neq \varnothing, S_2 \neq \varnothing$$

By definition,

$$S_2 = X \backslash S_1$$

Then, we can construct a continuous function $f : X \to \{0, 1\}$

$$f(x) = \begin{cases} 0 & x \in S_1 \\ 1 & x \in S_2 \end{cases}$$

and it is indeed surjective. This causes contradiction to (iv). □

Exercise 2.2 — Intermediate value theorem

In this exercise we prove the intermediate value theorem presented in the lecture slides.

Theorem 2.1 (Intermediate Value Theorem)**.** *Let X be a connected metric space, and $a, b \in X$. If $f : X \to \mathbb{R}$ is continuous then it takes all the values between $f(a)$ and $f(b)$.*

1. Let X and Y be two metric spaces, A be a connected subset of X, and $f : X \to Y$ be a continuous map. Show that $f(A)$ is connected.
   Hint: use one of the characterizations of a connected set from exercise 1.

   Proof. Assume $f(A)$ is not connected, then it has a subset S which is both open and closed, and $S \neq \varnothing, S \neq f(A)$. Then we can write $f(A)$ as the union of two disjoint, non-empty open subsets

   $$f(A) = S \cup (f(A) \backslash S)$$

   By the definition of continuous function,

   $$\begin{aligned} &\text{open } S \subset f(A) \implies \text{open } f^{-1}(S) \subset A \\ &\text{open } S^c \subset f(A) \implies \text{open } f^{-1}(S^c) \subset A \implies \text{open } \left(f^{-1}(S)\right)^c \subset A \end{aligned}$$

   Then we can also write A as the union of two disjoint, non-empty open subsets

   $$A = f^{-1}(S) \cup \left(f^{-1}(S)\right)^c$$

   which leads to contradiction to the connectedness of A. □

2. Let A be a subset of $\mathbb{R}$. We want to prove that A is connected if and only if A is an interval.

(a) Show that it is true for the empty set, and for all the subsets of $\mathbb{R}$ composed of a single element.

Proof. By definition, $\varnothing$ is both closed and open, and the only subset that is both closed and open is itself, as well as $\varnothing$. Then, $\varnothing$ is connected by definition. For those subsets which only contain a single element, there are only two different subsets: themselves and $\varnothing$, which are both closed and open. Thus, the subsets of $\mathbb{R}$ that only contain one element are also connected by definition. □

(b) Assuming that A is not an interval prove that A is not connected.

Proof. The property of interval $S \subseteq \mathbb{R}$ ensures that if $x < y \in S$, and $z \in \mathbb{R}$ such that $x < z < y$, then $z \in S$. If A is not an interval, then $\exists x, y \in A$ and $z \in \mathbb{R}\backslash A$ such that $x < z < y$. Consider the sets $A_1 = A \cap (-\infty, z)$ and $A_2 = A \cap (z, +\infty)$. Then A_1 and A_2 are open by definition of the subset topology on A. Neither of them is empty since $x \in A_1$ and $y \in A_2$. They are disjoint since $A_1 \cap A_2 = (-\infty, z) \cap (z, +\infty) = \varnothing$. And their union is A, since $z \notin A$. Then, we can write $A_1|A_2$ us a separation of A. It follows by definition that A is disconnected. □

(c) We now prove the converse.

i Show that it suffices to prove that $\mathbb{R}$ is connected.

Proof. If a subset $A \subset \mathbb{R}$ is an interval, suppose it has a open and closed subset $T \subset A$ with

$$T \neq A \quad T \neq \varnothing$$

Then there exists an element $x \in T$ and an element $z \in A\backslash T$. Without loss of generality, assume $x < z$. Put $S = T \cap [x, z]$. Then S, being the intersection of two closed sets, is closed. It is also bounded above, since z is the upper bound. Since it is closed and has an upper bound, $p := \sup S \in S$. Since $p \in [x, z], p \leq z$ As $z \in A\backslash S$, $p \neq z$ and so $p < z$. Now T is also an open set and $p \in T$. So there exist $a < b \in A$ such that $p \in (a, b) \subseteq T$. Let t be such that $p < t < \min(b, z)$. So $t \in T$ and $t \in [p, z]$. Thus, $t \in T \cap [x, z] = S$. This is a contradiction since $t > p$ and p is the supremum of S. Hence, our assumption is false and consequently $T = A$ or $T = \varnothing$. So the only remaining case is $\mathbb{R}$. □

ii Let U be a subset of $\mathbb{R}$, different from $\mathbb{R}$, that is both open and closed in $\mathbb{R}$. Find a contradiction and conclude that $\mathbb{R}$ is connected.
Hint: observe that a closed and non-empty set having an infimum has a minimum.

Proof. If a proper non-empty subset $U \subset \mathbb{R}$ is closed, then it has a minimum. At the minimum, it does not have a neighborhood, which contradicts to the openness. If it has a neighborhood, then there is contradiction to the definition of minimum. Thus, a proper non-empty subset of $\mathbb{R}$ cannot be both open and

closed. Thus $\mathbb{R}$ has only two subsets that are both open and closed: $\varnothing$ and itself. It follows that $\mathbb{R}$ is connected by definition. □

3. Conclude the proof of the intermediate value theorem.

Proof. We have already proved the connectedness of $f(A)$. And we have just proved connected subsets of $\mathbb{R}$ are all intervals. Then $f(A)$ is an interval which contains both $f(a)$ and $f(b)$. And since it is an interval, it also contains all the values between them. □

Exercise 2.3 — Extreme value theorem

In this exercise we prove the extreme value theorem presented in the lecture slides.

Theorem 2.2 (Extreme Value Theorem)**.** *Let X be a metric space, $f : X \to \mathbb{R}$ a continuous map, and A be a non-empty compact subset of X. Then f is bounded and reaches both its upper and lower bounds.*

1. Let X be a metric space and A a subset of X.

   (a) Prove that if X is compact and A is closed in X, then A is a compact subset of X.

   Proof. Since X is compact, we can write it as

   $$X \subseteq \bigcup_{i \in I} U_i$$

   where U_i forms a finite open cover of X. Since A is a closed subset of X, $X \backslash A$ is an open subset of X, so we can take the union

   $$X \subseteq \left(\bigcup_{i \in I} U_i \right) \cup (X \backslash A)$$

   So there exists a finite subcovering $U_{i_1}, U_{i_2}, \cdots, U_{i_k}, X \backslash A$,

   $$X \subseteq U_{i_1} \cup U_{i_2} \cup \cdots \cup U_{i_k} \cup (X \backslash A)$$

   Therefore,

   $$A \subseteq U_{i_1} \cup U_{i_2} \cup \cdots \cup U_{i_k} \cup (X \backslash A)$$

   Since $A \cap (X \backslash A) = \varnothing$, we can reduce it to

   $$A \subseteq U_{i_1} \cup U_{i_2} \cup \cdots \cup U_{i_k}$$

   Hence, A has a finite subcovering and thus is compact. □

   (b) Show that if A is a compact subset of X then A is closed in X.

Proof. We shall show that A contains all its limit points and hence is closed. Let $p \in X \backslash A$, then for each $a \in A$, there exists open sets

$$U_a := B(a, d(a,p)/2) \quad V_a := B(p, d(a,p)/2)$$

such that $a \in U_a$, $p \in V_a$ and $U_a \cap V_a = \varnothing$. Then

$$A \subseteq \bigcup_{a \in A} U_a$$

As A is compact, there exist $a_1, a_2, \cdots, a_n$ in A such that

$$A \subseteq U_{a_1} \cup U_{a_2} \cup \cdots \cup U_{a_n}$$

Define

$$U := U_{a_1} \cup U_{a_2} \cup \cdots \cup U_{a_n}$$
$$V := V_{a_1} \cap V_{a_2} \cap \cdots \cap V_{a_n}$$

Then $p \in V$ and $V_a \cap U_a = \varnothing$ implies $V \cap U = \varnothing$, which in turn implies $V \cap A = \varnothing$. So p is not a limit point of A, and V is an open set containing p which does not intersect A. Hence A contains all of its limit points and therefore is closed. □

2. Prove that a subset of $\mathbb{R}$ is compact if and only if it is closed and bounded.

Proof. From the previous proof, we already have a compact subset of a metric space is closed. We just need it to be bounded. Assume a subset $A \subseteq \mathbb{R}$ is unbounded, then it can be written as

$$A \subseteq \bigcup_{n=1}^{\infty} (-n, n)$$

But $\{(-n, n) : n \in \mathbb{N}^*\}$ does not have any finite subcovering of A since it is unbounded. Therefore A is not compact, which leads to contradiction. Hence all compact subsets of $\mathbb{R}$ are bounded. Together with (a), we have all compact subsets of $\mathbb{R}$ are closed and bounded. The converse is a special case of the Heine-Borel theorem.

Theorem 2.3 (Heine-Borel Theorem)**.** *Every closed bounded subset of* $\mathbb{R}$ *is compact.*

Proof. If A is a closed bounded subset of $\mathbb{R}$, then $A \subseteq [a, b]$, for some $a < b \in \mathbb{R}$. Since the space $[a, b]$ is homeomorphic to $[0, 1]$, we only need to prove $[0, 1]$ is compact. Let $O_i, i \in I$ be any open covering of $[0, 1]$. Then $\forall x \in [0, 1], \exists O_i$ such that $x \in O_i$. As O_i is open and contains x, there exists an interval U_x, open in $[0, 1]$ such that $x \in U_x \subseteq O_i$. Now define a subset $S \subset [0, 1]$ as follows:

$$S = \left\{ z \;\middle|\; [0, z] \text{ can be covered by a finite number of the sets } U_x \right\}$$

so that

$$z \in S \implies [0, z] \subseteq U_{x_1} \cup U_{x_2} \cup \cdots \cup U_{x_n} \cup U_x$$

Now let $x \in S$ and $y \in U_x$. Without loss of generality, we assume $x \leq y$, then as U_x is an interval containing x and y, $[x, y] \subseteq U_x$.

$$[0, y] \subseteq U_{x_1} \cup U_{x_2} \cup \cdots \cup U_{x_n} \cup U_x$$

and hence $y \in S$. So for each $x \in [0, 1]$, $U_x \cap S = U_x$ or $U_x \cap S = \varnothing$. This implies that

$$S = \bigcup_{x \in S} U_x \quad \text{and} \quad [0, 1] \backslash S = \bigcup_{x \notin S} U_x$$

Thus we have that S is open in $[0, 1]$. But $[0, 1]$ is connected. Therefore, $S = [0, 1]$ or $S = \varnothing$. However, $0 \in S$, and so $S = [0, 1]$. That means $[0, 1]$ can be covered by finite number of U_x, so

$$[0, 1] \subseteq U_{x_1} \cup U_{x_2} \cup \cdots \cup U_{x_m}$$

But each U_{x_i} is contained in an $O_i, i \in I$. Hence,

$$[0, 1] \subseteq O_{i_1} \cup O_{i_2} \cup \cdots \cup O_{i_m}$$

and we have shown that $[0, 1]$ is compact, which indicates all the closed intervals $[a, b]$ are compact. As $[a, b]$ is compact and A is a closed subset, A is compact. □

□

3. Complete the proof of the extreme value theorem. We apply Theorem 2.79 from the lecture slides.

   Theorem 2.4 (Compactness and continuity)**.** *Let X and Y be two metric spaces and $f : X \to Y$ be a continuous function. Then for any compact subset $A \subset X$, $f(A) \subset Y$ is compact.*

   Since A is a non-empty compact subset of X, $f(A)$ is compact on $\mathbb{R}$. Thus, $f(A)$ is closed and bounded. Since it is bounded, it has a supremum and an infimum. Since it is closed, all the limit points are in $f(A)$. Thus, $\sup f(A) \in f(A)$ and $\inf f(A) \in f(A)$.

Exercise 2.4 — Rolle's theorem

Reasoning by induction and applying the extreme values theorem, prove Rolle's theorem.

Theorem 2.5 (Rolle's Theorem)**.** *Let f be a $C^n[a, b]$ function, for $a, b \in \mathbb{R}$, which has $n+1$ distinct roots in $[a, b]$. Then there exists $c \in (a, b)$ such that $f^{(n)}(c) = 0$.*

Proof. We first need Fermat's theorem.

Theorem 2.6 (Fermat's Theorem)**.** *Let f be defined on $[a, b]$ and let $c \in (a, b)$ be such that $f(c) \geq f(x)$ for all $x \in (a, b)$. Then either $f'(c) = 0$ or $f'(c)$ does not exist.*

Proof. Let $\{x_n\}$ be any sequence in (a, b) with $x_n \to c$ (and $x_n \neq c$) so $f(c) \geq f(x_n)$. Now whenever $c > x_n$, we have

$$\frac{f(x_n) - f(c)}{x_n - c} \geq 0$$

But whenever $c < x_n$, we have

$$\frac{f(x_n) - f(c)}{x_n - c} \leq 0$$

If $f'(c)$ exists then we must have $f'(c) = 0$. Otherwise, $f'(c)$ does not exist. □

We first prove the Rolle's theorem with the $n = 1$ case. If f is constant on $[a, b]$, then its derivative is zero and so any $c \in (a, b)$ satisfies the conclusion of the theorem. So we assume that f is not constant on $[a, b]$. By the Extreme Value Theorem, f attains an absolute maximum and an absolute minimum on $[a, b]$. Since f is not constant, at least one of these absolute extrema must occur at $c \in (a, b)$. Then since f is differentiable on (a, b), an application of Fermat's Theorem gives $f'(c) = 0$ as desired.

And Fermat's theorem can also be generalized by induction: assume f is at least k times differentiable, and replace all the f with $f^{(k)}$. Let $c \in (a, b)$ be such that $f^{(k)}(c) \geq f^{(k)}(x)$ for all $x \in (a, b)$. Then either $f^{(k+1)}(c) = 0$ or $f^{(k+1)}(c)$ does not exist.

Assume Rolle's theorem holds for $n = k$, then $\exists c_1 \in (a, b)$ such that $f^{(k)}(c_1) = 0$. If $f^{(k)}$ is constant on $[a, b]$, then its derivative is zero and so any $c \in (a, b)$ satisfies the conclusion of the theorem. So we assume that $f^{(k)}$ is not constant on $[a, b]$. By the Extreme Value Theorem, $f^{(k)}$ attains an absolute maximum and an absolute minimum on $[a, b]$. Since $f^{(k)}$ is not constant, at least one of these absolute extrema must occur at $c_2 \in (a, b)$. Then since f is $k+1$-times differentiable on (a, b), an application of Fermat's Theorem gives $f^{(k+1)}(c_2) = 0$ as desired.

According to the principle of mathematical induction, Rolle's theorem holds for all $n \in \mathbb{N}^*$. □

Exercise 2.5 — Continuity

In this exercise we provide alternative characterizations for a continuous function. Let X and Y be two metric spaces, f be a function from X into Y, and $a \in X$. We want to prove that the following conditions are equivalent

(i) For all $\varepsilon \in \mathbb{R}_+^*$, there exists $\eta \in \mathbb{R}_+^*$ such that $f(B(a, \eta)) \subset B(f(a), \varepsilon)$;

(ii) For all $\varepsilon \in \mathbb{R}_+^*$, there exists $\eta \in \mathbb{R}_+^*$ such that $d(f(x), f(a)) < \varepsilon$ when $d(a, x) < \eta$, for $x \in X$;

(iii) For any neighborhood V of $f(a)$, there exists a neighborhood U of a such that $f(U) \subset V$;

(iv) For any neighborhood V of $f(a)$, $f^{-1}(V)$ is a neighborhood of a;

1. Show that (i) and (ii) are equivalent;

Proof. The open ball on a metric space (X, d) is defined to be

$$B_d(a, \eta) := \{x \in X \mid d(x, a) < \eta\}$$

while its image is

$$f(B_d(a, \eta)) := \{f(x) \in Y \mid d(x, a) < \eta\}$$

And it follows that

$$B_d(f(a), \varepsilon) = \{y \in Y \mid d(y, f(a)) < \varepsilon\}$$

Since $x \in X \implies f(x) \in Y$, we can replace the y with $f(x)$,

$$B_d(f(a), \varepsilon) = \{f(x) \in Y \mid d(f(x), f(a)) < \varepsilon\}$$

By the definition of subset relation,

$$f(B(a, \eta)) \subset B(f(a), \varepsilon) \implies (d(x, a) < \eta \implies d(y, f(a)) < \varepsilon)$$

The converse ((ii) $\implies$ (i)) is more tricky, since if we directly take the η in (ii) as the η in (i), we will get a $\subseteq$ relation according to definition. Thus, we need to take $\eta_1 = \eta_2/2$, where η_1 is the η in (i) and η_2 is the η in (ii). Then

$$(d(x, a) < \eta \implies d(y, f(a)) < \varepsilon) \implies f(B(a, \eta/2)) \subset f(B(a, \eta)) \subseteq B(f(a), \varepsilon)$$

□

2. Consider a neighborhood of $f(a)$ and prove that (i) implies (iii).

   Proof. This is trivial by taking $U = B(a, \eta)$ and $V = B(f(a), \varepsilon)$. □

3. Observe that if V is a neighborhood of $a \in X$, then any subset of X containing V is a neighborhood of a. Conclude that (iii) implies (iv).

   Proof. By the definition of subset relation,

   $$V \subseteq W \implies \forall a \in V, a \in W$$

   This shows that W is also a neighborhood of a. By writing U as $f^{-1}(f(U))$, we have

   $$U = f^{-1}(f(U)) \subset f^{-1}(V)$$

   Since U is a neighborhood of a, $f^{-1}(V)$ is also a neighborhood of a. □

4. Consider $V = B(f(a), \varepsilon)$, for some ε, and prove that (iv) implies (i).

   Proof. An open set S on a metric space (X, d) has the property that

   $$\forall x \in S, \exists B(x, \eta) \subset S$$

   By (iv), $f^{-1}(V) = f^{-1}(B(f(a), \varepsilon))$ is a neighborhood of a, which is an open set, and X is a metric space, we have already found an η for (i) to hold. □

Assignment 3

Integration

Exercise 3.1 — Cantor set

In this exercise we investigate a few properties of the Cantor set and Cantor function. Calling C_0 the compact $[0,1]$ we trisect it and remove the middle open part and define C_1 as $[0,1/3] \cup [2/3,1]$. Recursively repeating the process we construct an infinite sequence $C_0 \supset C_1 \supset \cdots \supset C_i \supset \cdots$. We define the *Cantor set* as the intersection of all the nested C_i, $C = \bigcap_{i=0}^{\infty} C_i$.

1. Prove that C is compact.

   Proof. By Assignment 1.1(2), the intersection of a collection (finite or infinite) of closed subsets of a metric space is closed. By definition of Cantor set, it is an intersection of closed intervals, and thus is closed. It is also bounded, since it has lower bound 0 and upper bound 1. By Heine-Borel theorem, every closed bounded subset of $\mathbb{R}$ is compact. Hence, the Cantor set is compact. □

2. Show that for any two elements $x < y \in C$, $\exists z \notin C$ such that $x < z < y$.

   Proof. We can write the Cantor set as

   $$[0,1] \setminus \bigcup_{n=1}^{\infty} \bigcup_{k=0}^{3^{n-1}-1} \left(\frac{3k+1}{3^n}, \frac{3k+2}{3^n} \right)$$

   Since $y > x$, we can find n such that

   $$m = \left\lfloor \log_3 \frac{2}{y-x} \right\rfloor + 1$$

   Then we have

   $$\frac{1}{3^m} < \frac{2}{3^m} < y - x$$

   or equivalently,

   $$3^m x < 3^m x + 1 < 3^m x + 2 < 3^m y$$

We select $j = \lfloor 3^{m-1}x \rfloor$, then, we have an interval such that

$$x < \frac{3j+1}{3^m} < \frac{3j+2}{3^m} < y$$

Hence, we can always find an interval Z with

$$Z = \left(\frac{3j+1}{3^m}, \frac{3j+2}{3^m}\right), \quad Z \cap C = \varnothing$$

and $\forall z \in Z, x < z < y, z \notin C$. □

3. We estimate how "large" the set C is with respect to λ, the Lebesgue measure.

   (a) What is the Lebesgue measure of a countable set?

   Answer. 0 □

   (b) Show that for $n \in \mathbb{N}, \lambda(C_n) = \left(\frac{2}{3}\right)^n$, and conclude that C has Lebesgue measure 0.

   Proof. We can prove by recursive definition. By removing the middle part of each interval, we have removed $\frac{1}{3}$ of the original length, so

   $$\lambda(C_{n+1}) = \frac{2}{3}\lambda(C_n)$$

   Since $\lambda(C_0) = 1$, we have proved $n \in \mathbb{N}, \lambda(C_n) = \left(\frac{2}{3}\right)^n$. By σ-additivity and Fubini's theorem, we can also interchange the order of limit and summation,

   $$\lambda(C) = \lambda\left(\lim_{n\to\infty} C_n\right) = \lim_{n\to\infty} \lambda(C_n) = \lim_{n\to\infty} \left(\frac{2}{3}\right)^n = 0$$

   □

4. We estimate how "large" the set C is with respect to cardinality.

   (a) Show that C is not empty.

   Proof. We can find some trivial elements in the Cantor set.

   $$\left\{0, 1, \frac{1}{3}, \frac{2}{3}\right\} \subset C$$

   So C is not empty. □

   (b) We write any $x \in [0, 1]$ using the ternary expansion

   $$x = \sum_{i=1}^{\infty} \frac{a_i}{3^i}, \quad \text{with } a_i \in \{0, 1, 2\}$$

   Describe the form of the x belonging to $C_i, i \in \mathbb{N}$.

Answer. $\forall i \in \mathbb{N}, a_i \neq 1$ □

(c) Using Cantor diagonal argument show that Cantor set is uncountable.

Proof. We prove that the Cantor set C has more elements than $\mathbb{N}$. Suppose we can construct a bijective map from $\mathbb{N}$ to C, then

$$\begin{aligned} n_1 &\to (0.a_{11}a_{12}a_{13}a_{14}\cdots)_3 \\ n_2 &\to (0.a_{21}a_{22}a_{23}a_{24}\cdots)_3 \\ n_3 &\to (0.a_{31}a_{32}a_{33}a_{34}\cdots)_3 \\ &\vdots \end{aligned}$$

where $a_{ij} \in \{0, 2\}$. Then we can always find an element from C that is not mapped to. Take $b_1 \neq a_{11}, b_2 \neq a_{22}, \cdots, b_i \neq a_{ii}, \cdots$, $0.b_1b_2\cdots b_i\cdots \in C$ differs from all the elements in the range of the mapping, so the mapping is not surjective. This contradicts to the bijection assumption. Thus the bijection does not exist, and C has more elements than $\mathbb{N}$, which implies it is uncountable. □

5. Confront the results from questions 3 and 4.

Answer. Although the Cantor set has Lebesgue measure 0, its cardinality is larger than $\aleph_0$. This is a very anti-intuition result. □

We now define the Cantor function following the construction process of the Cantor set. Let f_1 be $1/2$ over the "removed interval" $(1/3, 2/3)$ and linear on C_1. Then define f_2 to be $1/4$ and $3/4$ on the two removed intervals, to coincide with f_1 in $1/3$ and $2/3$, while being linear on the remaining four intervals. The process is carried on as the Cantor set is built, defining the Cantor function f_C.

6. Show that the $(f_n)_{n\in\mathbb{N}}$ define a sequence of monotonically increasing continuous functions.

Proof. By definition, the Cantor function $f_n(x)$ is constant on $[0, 1]\backslash C_n$, where C_n is recursively defined. Constant functions are trivially monotone and continuous. And $f_n(x)$ is linear on C_n, while linear functions are also continuous. And by definition, they have the same value at the conjunctions, which is also continuous. □

7. Show that $(f_n)_{n\in\mathbb{N}}$ converges uniformly to $f_C : [0, 1] \to [0, 1]$.

Proof. We apply the analytic expression of f_{n+1} by f_n:

$$f_{n+1}(x) = \begin{cases} \frac{1}{2}f_n(3x) & x \in \left[0, \frac{1}{3}\right] \\ \frac{1}{2} & x \in \left(\frac{1}{3}, \frac{2}{3}\right) \\ \frac{1}{2} + \frac{1}{2}f_n(3x-2) & x \in \left[\frac{2}{3}, 1\right] \end{cases}$$

By definition, we find

$$\max_{x\in[0,1]} |f_{n+1}(x) - f_n(x)| \le \frac{1}{2} \max_{x\in[0,1]} |f_n(x) - f_{n-1}(x)|, \quad \forall n \ge 1$$

where $f_0(x) = x$ on $[0,1]$ as an extended definition. This gives

$$\max_{x\in[0,1]} |f(x) - f_n(x)| \le 2^{-n+1} \max_{x\in[0,1]} |f_1(x) - f_0(x)|$$

Both f_0 and f_1 are bounded, so we get the uniform convergence of f_n to f_C. □

8. Prove that the Cantor function is

(a) Uniformly continuous;

Proof. We prove by definition. Let $\varepsilon > 0$ be given. Select $n = \lfloor -\log_2 \varepsilon \rfloor + 1$, and take $\delta = 1/3^n$.

$$\forall x, y \in [0,1] \text{ with } |x - y| < \delta = \frac{1}{3^n}$$

then the base-3 expansion of x and y agree for at least the first n terms, which implies

$$|f_C(x) - f_C(y)| \le \sum_{j=n+1}^{\infty} \frac{1}{2^j} = \frac{1}{2^n} < \varepsilon$$

This gives the uniform continuity of f_C on $[0,1]$. □

(b) Monotonically increasing (non-decreasing);

Proof. We can prove by the explicit form of the Cantor function

$$f_C(x) = f_C\left(\sum_{i=1}^{\infty} \frac{a_i}{3^i}\right) = \sum_{i=1}^{N_x} \frac{b_n}{2^n}$$

where N_x denotes the smallest i such that $a_i = 1$ if it exists or $N_x = \infty$ if there is no such a_i, and

$$b_n = \begin{cases} a_n/2 & n < N_x \\ 1 & n = N_x \end{cases}$$

which is the binary expansion of a number in $[0,1]$. This is a monotonely increaseing transformation from a_n to b_n for $n < N_x$, and monotonely non-decreasing for $n = N_x$. Thus, if $x < y$, then their tenary expansions must differ at some point N and at that point $a_{Nx} < a_{Ny}$. The monotocity is preserved by the mapping $a_n \to b_n$, and is also preserved by binary expansion. Thus, the combination, Cantor function, is monotonely non-decreasing. □

(c) Differentiable almost everywhere, with $f'(x) = 0$;

Proof. By definition, $f_C(x)$ is constant when $x \in [0,1]\backslash C$, and the constant function has $f'_C(x) = 0$. Since $[0,1]\backslash C$ has Lebesgue measure 1, it is almost everywhere differentiable. □

9. Prove that f_C is not absolutely continuous.

Proof. By selecting x_{n_k} as the left endpoints of C_n and y_{n_k} as the right endpoints, we are able to construct an example to break absolute continuity. We have already proved the length of C_n is $\left(\frac{2}{3}\right)^n$, so we can set $n = \lfloor \log_{2/3} \delta \rfloor + 1$ such that

$$\sum_{k=1}^{2^n} |y_{n_k} - x_{n_k}| < \delta$$

But we can select $\varepsilon = 0.5$, and

$$\sum_{k=1}^{2^n} |f_C(y_{n_k}) - f_C(x_{n_k})| = 1 > \varepsilon$$

Thus f_C is not absolutely continuous. □

Note: the Cantor function is one of the most simple function to be uniformly continuous but not absolutely continuous.

Exercise 3.2 — Taylor's theorem

Reasoning by induction and applying the fundamental theorem of calculus, prove the following theorems.

Theorem 3.1 (Taylor's theorem)**.** *Given a function f such that $f^{(n)}$ is absolutely continuous on the compact $[a,x] \subset \mathbb{R}$,*

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k + \frac{1}{n!}\int_a^x f^{(n+1)}(t)(x-t)^n \, \mathrm{d}t$$

Proof. We prove by induction and integration by parts. When, $n = 1$, this is the fundamental theorem of calculus.

$$f(x) = f(a) + \int_a^x f'(t) \, \mathrm{d}t$$

Assume this is true for n, then by integration by parts,

$$\begin{aligned}
f(x) &= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k + \frac{1}{n!}\int_a^x f^{(n+1)}(t)(x-t)^n \,\mathrm{d}t \\
&= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k - \frac{1}{n!}\int_a^x f^{(n+1)}(t)(x-t)^n \,\mathrm{d}(x-t) \\
&= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k - \frac{1}{n!(n+1)}\int_a^x f^{(n+1)}(t) \,\mathrm{d}(x-t)^{n+1} \\
&= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k - \frac{1}{(n+1)!}\left[f^{(n+1)}(t)(x-t)^{n+1}\Big|_a^x - \int_a^x (x-t)^{n+1} \,\mathrm{d}f^{(n+1)}(t)\right] \\
&= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k + \frac{f^{(n+1)}(a)(x-a)^{n+1}}{(n+1)!} + \frac{1}{(n+1)!}\int_a^x f^{(n+2)}(t)(x-t)^{n+1} \,\mathrm{d}t \\
&= \sum_{k=0}^{n+1} \frac{f^{(k)}(a)}{k!}(x-a)^k + \frac{1}{(n+1)!}\int_a^x f^{(n+2)}(t)(x-t)^{n+1} \,\mathrm{d}t
\end{aligned}$$

which proves the theorem is also true for $n+1$. By principles of mathematical induction, we have proved the theorem for all $n \in \mathbb{N}$. □

Theorem 3.2 (Taylor's theorem with the remainder in the Lagrange form)**.** *Calling $P_n(x)$ the polynomial part of $f(x)$ in Taylor's theorem, show the existence of $c \in [a, x]$ such that*

$$f(x) - P_n(x) = \frac{f^{n+1}(c)}{(n+1)!}(x-a)^{n+1}$$

Proof. For simplicity, we define

$$F(x) = f(x) - P_n(x) = \frac{f^{n+1}(c)}{(n+1)!}(x-a)^{n+1}$$

It is easy to observe

$$F(a) = F'(a) = \cdots = F^{(n)}(a) = 0$$

So it is reasonable to compare F with this function

$$G(x) = (x-a)^{n+1}$$

which also has

$$G(a) = G'(a) = \cdots = G^{(n)}(a) = 0$$

Applying Cauchy's Mean Value Theorem $n+1$ times, $\exists c_1, c_2, \cdots, c_{n+1} \in [a, x]$ such that

$$\begin{aligned}\frac{F(x)}{G(x)} &= \frac{F(x)-F(a)}{G(x)-G(a)} = \frac{F'(c_1)}{G'(c_1)} \\ &= \frac{F'(c_1)-F'(a)}{G'(c_1)-G'(a)} = \frac{F''(c_2)}{G''(c_2)} \\ &= \cdots = \frac{F^{(k)}(c_k)-F^{(k)}(a)}{G^{(k)}(c_1)-G^{(k)}(a)} \\ &= \frac{F^{(n+1)}(c_{n+1})}{G^{(n+1)}(c_{n+1})}\end{aligned}$$

which means

$$F(x) = \frac{F^{(n+1)}(c_{n+1})}{G^{(n+1)}(c_{n+1})} G(x)$$

Now we observe that

$$\begin{aligned}F^{(n+1)}(c_{n+1}) &= f^{(n+1)}(c_{n+1}) \\ G^{(n+1)}(c_{n+1}) &= (n+1)!\end{aligned}$$

This gives

$$F(x) = \frac{f^{(n+1)}(c_{n+1})}{(n+1)!}(x-a)^{n+1}$$

Set $c = c_{n+1}$, we get the desired result. □

Exercise 3.3 — Convergence of rationals to irrationals

1. Show that e is irrational.

   Proof. We first get the upper bound and lower bound of e. Since e is defined to be

   $$e := \sum_{n=0}^{\infty} \frac{1}{n!}$$

   We apply the telescoping method:

   $$\begin{aligned}e &= \sum_{n=0}^{\infty} \frac{1}{n!} > \sum_{n=0}^{1} \frac{1}{n!} = 1+1 = 2 \\ e &= \sum_{n=0}^{\infty} \frac{1}{n!} = 1+1+\sum_{n=2}^{\infty} \frac{1}{n!} < 1+1+\sum_{n=1}^{\infty} \frac{1}{2^n} = 3\end{aligned}$$

   So we get $2 < e < 3$. Assume e is rational, then we can write it as a fraction $\frac{a}{b}$, where $a, b \in \mathbb{N}^*$. Since e is not an integer, we have $b \geq 2$, and since $e > 1$, we also have $a > b$. We multiply the both sides of the definition of e by $b!$, ans split the sum to two parts

   $$\frac{a}{b} \cdot b! = \sum_{n=0}^{b} \frac{b!}{n!} + \sum_{n=b+1}^{\infty} \frac{b!}{n!}$$

The left side of the equation is $a(b-1)!$, which is an integer. The first sum is

$$\sum_{n=0}^{b} \frac{b!}{n!} = \sum_{n=0}^{b} \binom{b}{n} n!$$

which is also an integer. Thus,

$$\sum_{n=b+1}^{\infty} \frac{b!}{n!}$$

is an integer. However,

$$\sum_{n=b+1}^{\infty} \frac{b!}{n!} < \sum_{n=1}^{\infty} \frac{b!}{b! \cdot b^n} = \frac{1}{b} < 1$$

there is no such an integer between 0 and 1. Thus, the assumption that e is rational will cause contradiction. Hence, e is irrational. □

2. Show that the sequence $(u_n)_{n\in\mathbb{N}}$ defined by $u_n = \left(1+\frac{1}{n}\right)^n$ converges to e.

Proof. We first prove a lemma.

Lemma 3.1. *If a sequence* $(a_n)_n \in [0,1]$,

$$\prod_{i=1}^{n}(1-a_i) \geq 1 - \sum_{i=1}^{n} a_i$$

Proof. This is easily proved by mathematical induction. When $n = 1$, the claim becomes $1 - a_1 \geq 1 - a_1$, which is trivially true. Next, we assume it is true for n. If $\sum_{i=1}^{n+1} a_i > 1$, then

$$\prod_{i=1}^{n}(1-a_i) \geq 0 > 1 - \sum_{i=1}^{n+1} a_i$$

So we assume $\sum_{i=1}^{n+1} a_i \leq 1$. Then we automatically have $\sum_{i=1}^{n} a_i \leq 1$, and

$$\begin{aligned}
\prod_{i=1}^{n+1}(1-a_i) &= (1-a_{n+1})\prod_{i=1}^{n}(1-a_i) \\
&\geq (1-a_{n+1})\left(1-\sum_{i=1}^{n} a_i\right) \\
&= 1 - \left(a_{n+1} + \sum_{i=1}^{n} a_i\right) + a_{n+1}\sum_{i=1}^{n} a_i \\
&\geq 1 - \sum_{i=1}^{n+1} a_i
\end{aligned}$$

According to the principle of mathematical induction, we have the lemma proved. □

Apply binomial theorem, we can expand u_n as the infinite sum

$$u_n = \sum_{i=0}^{n} \binom{n}{k} \frac{1}{n^k} = \sum_{k=0}^{n} \frac{1}{k!} \cdot \frac{n!}{(n-k)!n^k}$$

when $k \geq 1$,

$$\frac{n!}{(n-k)!n^k} = \prod_{i=0}^{k-1} \left(1 - \frac{i}{n}\right)$$

Apply the lemma, we get the lower bound and the upper bound of the product,

$$1 - \sum_{i=0}^{k-1} \frac{i}{n} \leq \prod_{i=0}^{k-1} \left(1 - \frac{i}{n}\right) \leq 1$$

And the lower bound can be simplified to

$$1 - \sum_{i=0}^{k-1} \frac{i}{n} = 1 - \frac{k(k-1)}{2n}$$

Plug into u_n,

$$\sum_{k=0}^{n} \frac{1}{k!} \left(1 - \frac{k(k-1)}{2n}\right) \leq u_n \leq \sum_{k=0}^{n} \frac{1}{k!}$$

and the lower bound can be expanded to

$$\sum_{k=0}^{n} \frac{1}{k!} \left(1 - \frac{k(k-1)}{2n}\right) = \sum_{k=0}^{n} \frac{1}{k!} - \frac{1}{2n} \left(1 + 1 + \sum_{k=2}^{n} \frac{1}{(k-2)!}\right)$$

while its limit when $n \to \infty$ is

$$\lim_{n\to\infty} \left[\sum_{k=0}^{n} \frac{1}{k!} - \frac{1}{2n} \left(1 + 1 + \sum_{k=2}^{n} \frac{1}{(k-2)!}\right)\right] = \lim_{n\to\infty} \sum_{k=0}^{n} \frac{1}{k!} - \lim_{n\to\infty} \frac{e+2}{2n} = e$$

And the upper bound also has the limit e. According to the squeeze theorem, if a sequence is bounded by two sequences with the same limit, then the sequence also converges to that limit. Thus we conclude that

$$\lim_{n\to\infty} u_n = e$$

□

Assignment 4

Orthogonal Polynomials and Interpolation

Exercise 4.1 — Legendre polynomials

Let $(Q_n)_{n\in\mathbb{N}}$ be the sequence of polynomials defined by $Q_n(x) = \frac{1}{2^n n!}\left((x^2-1)^n\right)^{(n)}$. This sequence defines the Legendre polynomials.

1. Using the constant weight function $w(x) = 1$ over $(-1, 1)$, show that $(Q_n)_{n\in\mathbb{N}}$ defines a sequence of orthogonal polynomials.

   Proof. First, we want to prove the inner product of any Legendre polynomials with themselves are non-zero. Define

   $$I_n = \int_{-1}^{1} Q_n^2(x)\ \mathrm{d}x$$

   We can easily get the first Legendre polynomial

   $$Q_0(x) = 1, \quad I_0 = \int_{-1}^{1} \mathrm{d}x = 2$$

   So $I_0 \neq 0$ trivially holds for $n = 0$. By integration by parts,

   $$\begin{aligned} I_n &= \frac{1}{2^{2n}(n!)^2} \int_{-1}^{1} \frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right] \cdot \frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right]\ \mathrm{d}x \\ &= \frac{1}{2^{2n}(n!)^2} \int_{-1}^{1} \frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right] \cdot\ \mathrm{d}\left\{\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right]\right\} \\ &= \frac{1}{2^{2n}(n!)^2} \left\{ \frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right] \frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right]\Big|_{-1}^{1} \right. \\ &\quad \left. - \int_{-1}^{1} \frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right] \cdot \frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}}\left[\left(x^2-1\right)^n\right]\ \mathrm{d}x \right\} \end{aligned}$$

Calculation Mistakes Since the $\frac{\mathrm{d}^k}{\mathrm{d}x^k}\left[(x^2-1)^n\right]$ always has a factor (x^2-1) for $0 \leq k \leq n$, the first term is always zero, which means

$$I_n = \frac{-1}{2^{2n}(n!)^2}\int_{-1}^{1} \frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[(x^2-1)^n\right] \cdot \frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}}\left[(x^2-1)^n\right]\ \mathrm{d}x$$

By repeating n times of integration by part, we get

$$I_n = \frac{(-1)^n}{2^{2n}(n!)^2}\int_{-1}^{1} \left[(x^2-1)^n\right] \cdot \frac{\mathrm{d}^{2n}}{\mathrm{d}x^{2n}}\left[(x^2-1)^n\right]\ \mathrm{d}x$$

Expand $(x^2-1)^n$ by binomial theorem, and only x^{2n} has a nonzero $2n^{\text{th}}$ derivative.

$$\frac{\mathrm{d}^{2n}}{\mathrm{d}x^{2n}}\left[(x^2-1)^n\right] = (2n)!$$

Plug in this result and

$$I_n = \frac{(2n)!}{2^{2n}(n!)^2}\int_{-1}^{1}\left(1-x^2\right)^n\ \mathrm{d}x$$

Define the integral part as

$$K_n = \int_{-1}^{1}\left(1-x^2\right)^n\ \mathrm{d}x$$

Integration by parts (taking a factor 1 as the second part) gives

$$K_n = \int_{-1}^{1} 2nx^2\left(1-x^2\right)^{n-1}\ \mathrm{d}x$$

Writing $2nx^2$ as $2n-2n\left(1-x^2\right)$ we obtain

$$\begin{aligned} K_n &= 2n\int_{-1}^{1}\left(1-x^2\right)^{n-1}\ \mathrm{d}x - 2n\int_{-1}^{1}\left(1-x^2\right)^n\ \mathrm{d}x \\ &= 2nK_{n-1} - 2nK_n \end{aligned}$$

So we get the recursive relation

$$(2n+1)K_n = 2nK_{n-1}$$

Expend this relation down to K_0, we have

$$K_n = \frac{2n}{2n+1}\cdot\frac{2n-2}{2n-1}\cdots\frac{2}{3}\cdot K_0$$

It is easy to get $K_0 = 2$, and reorganizing the terms,

$$K_n = \frac{2^{2n+1}(n!)^2}{(2n+1)!}$$

By substitution and some simple calculation,

$$I_n = K_n \cdot \frac{(2n)!}{2^{2n}(n!)^2} = \frac{2}{2n+1}$$

Then, we need to prove the Legendre polynomials are mutually orthogonal. We introduce the following property of Legendre polynomials

Theorem 4.1 (Legendre differential equation)**.** *Legendre polynomials are solutions to the Legendre differential equation*

$$\left[\left(1-x^2\right)Q_n'(x)\right]' + n(n+1)Q_n(x) = 0$$

Proof. To prove that Legendre polynomials solve the Legendre differential equation, we need the differential rule for a product of two factors.

Theorem 4.2 (General Leibniz rule)**.** *f and g are n-times differentiable functions, then the product fg is also n-times differentiable and its n^{th} derivative is given by*

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n}[f(x)g(x)] = \sum_{i=0}^{n}\binom{n}{i}\frac{\mathrm{d}^{n-i}}{\mathrm{d}x^{n-i}}f(x)\cdot\frac{\mathrm{d}^i}{\mathrm{d}x^i}g(x)$$

where the zeroth derivative is defined to be the function itself.

Proof. We prove by induction. At $n = 1$, we get

$$(fg)'(x) = f'(x)g(x) + f(x)g'(x)$$

This is the basic Leibniz rule, so the equality holds for $n = 1$. We assume that the equality

$$(fg)^{(m)}(x) = \sum_{k=0}^{m}\binom{m}{k}f^{(m-k)}(x)g^{(k)}(x)$$

Therefore, at $m+1$ we get:

$$\begin{aligned}
(fg)^{(m+1)}(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\sum_{k=0}^{m}\binom{m}{k}f^{(m-k)}(x)g^{(k)}(x)\\
&= \sum_{k=0}^{m}\binom{m}{k}f^{(m-k)}(x)g^{(k+1)}(x)+\sum_{k=0}^{m}\binom{m}{k}f^{(m+1-k)}(x)g^{(k)}(x)\\
&= \sum_{k=1}^{m+1}\binom{m}{k-1}f^{(m+1-k)}(x)g^{(k)}(x)+\sum_{k=0}^{m}\binom{m}{k}f^{(m+1-k)}(x)g^{(k)}(x)\\
&= \binom{m}{m}f(x)g^{(m+1)}(x)+\sum_{k=1}^{m}\binom{m}{k-1}f^{(m+1-k)}(x)g^{(k)}(x)\\
&+ \sum_{k=1}^{m}\binom{m}{k}f^{(m+1-k)}(x)g^{(k)}(x)+\binom{m}{0}f^{(m+1)}(x)g(x)\\
&= \binom{m}{m}f(x)g^{(m+1)}(x)+\sum_{k=1}^{m}\left[\binom{m}{k-1}+\binom{m}{k}\right]f^{(m+1-k)}(x)g^{(k)}(x)\\
&+ \binom{m}{0}f^{(m+1)}(x)g(x)\\
&= \binom{m+1}{m+1}f(x)g^{(m+1)}(x)+\sum_{k=1}^{m}\binom{m+1}{k}f^{(m+1-k)}(x)g^{(k)}(x)\\
&+ \binom{m+1}{0}f^{(m+1)}(x)g(x)\\
&= \sum_{k=0}^{m+1}\binom{m+1}{k}f^{(m+1-k)}(x)g^{(k)}(x).
\end{aligned}$$

By principle of mathematical induction, this holds for all $n\in\mathbb{N}^*$. □

To prove that Legendre polynomials are solution to Legendre differential equation, let $y=(x^2-1)^n$, so that $y'=2nx\,(x^2-1)^{n-1}$, and

$$\left(x^2-1\right)y'-2nxy=0$$

Differentiate this expression $n+1$ times using the general Leibniz rule, we obtain

$$\left(x^2-1\right)\frac{\mathrm{d}^{n+2}y}{\mathrm{d}x^{n+2}}+2x(n+1)\frac{\mathrm{d}^{n+1}y}{\mathrm{d}x^{n+1}}+n(n+1)\frac{\mathrm{d}^{n}y}{\mathrm{d}x^{n}}-2n\left[x\frac{\mathrm{d}^{n+1}y}{\mathrm{d}x^{n+1}}+(x+1)\frac{\mathrm{d}^{n}y}{\mathrm{d}x^{n}}\right]=0$$

which reduces to

$$\left(x^2-1\right)\frac{\mathrm{d}^{n+2}}{\mathrm{d}x^{n+2}}y+2x\frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}}y-n(n+1)\frac{\mathrm{d}^{n}}{\mathrm{d}x^{n}}y=0$$

Dividing each side by $-2^n(n!)$, we get

$$\left(1-x^2\right)\frac{\mathrm{d}^{n+2}}{\mathrm{d}x^{n+2}}\frac{y}{2^n(n!)}+2x\frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}}\frac{y}{2^n(n!)}-n(n+1)\frac{\mathrm{d}^{n}}{\mathrm{d}x^{n}}\frac{y}{2^n(n!)}=0$$

Notice that

$$Q_n(x) = \frac{\mathrm{d}^n}{\mathrm{d}x^n}\frac{y}{2^n(n!)}$$

Apply the substitution, we obtain

$$\left(1-x^2\right)Q_n''(x) + 2xQ_n'(x) - n(n+1)Q_n(x) = 0$$

which is exactly

$$\left[\left(1-x^2\right)Q_n'(x)\right]' + n(n+1)Q_n(x) = 0$$

□

Now we can prove that two different Legendre polynomials are mutually orthogonal. With the Legendre differential equation, we multiply both sides by $Q_k(x)$, where $n \neq k$, and integrate it in $[-1, 1]$.

$$\int_{-1}^{1} Q_k(x)\left[\left(1-x^2\right)Q_n'(x)\right]'\,\mathrm{d}x + \int_{-1}^{1} Q_k(x)n(n+1)Q_n(x)\,\mathrm{d}x = 0 \tag{4.1}$$

We also have the Legendre differential equation for $Q_k(x)$:

$$\left[\left(1-x^2\right)Q_k'(x)\right]' + k(k+1)Q_k(x) = 0$$

Multiply both sides by $Q_n(x)$, where $n \neq k$, and integrate it in $[-1, 1]$, we get a dual form

$$\int_{-1}^{1} Q_n(x)\left[\left(1-x^2\right)Q_k'(x)\right]'\,\mathrm{d}x + \int_{-1}^{1} Q_n(x)k(k+1)Q_k(x)\,\mathrm{d}x = 0. \tag{4.2}$$

Integrate by part on first term of both (4.1) and (4.2), we get

$$\begin{cases} -\int_{-1}^{1} Q_k'(x)\left[\left(1-x^2\right)Q_n'(x)\right]\,\mathrm{d}x + \int_{-1}^{1} Q_k(x)n(n+1)Q_n(x)\,\mathrm{d}x = 0 \\ -\int_{-1}^{1} Q_n'(x)\left[\left(1-x^2\right)Q_k'(x)\right]\,\mathrm{d}x + \int_{-1}^{1} Q_n(x)k(k+1)Q_k(x)\,\mathrm{d}x = 0 \end{cases}$$

Notice that the first terms of the two equations are equal, we have the second terms equal, which means

$$n(n+1)\int_{-1}^{1} Q_k(x)Q_n(x)\,\mathrm{d}x = k(k+1)\int_{-1}^{1} Q_n(x)Q_k(x)\,\mathrm{d}x$$

Since we are considering different Legendre polynomials, we have $n \neq k$, so $n(n+1) \neq k(k+1)$. Then we conclude that

$$\int_{-1}^{1} Q_n(x)Q_k(x)\,\mathrm{d}x = 0$$

So the mutual orthogonality is also proved. □

2. Show that $Q_n(-x) = (-1)^n Q_n(x)$.

Proof. We expand $(x^2-1)^n$ by binomial theorem.

$$(x^2-1)^n = \sum_{i=0}^{n} \binom{n}{i} x^{2n}(-1)^{n-i}$$

Split the cases where n is even and n is odd.

$$\frac{\mathrm{d}^{2m+1}}{\mathrm{d}x^{2m+1}}(x^2-1)^{2m+1} = \sum_{i=m+1}^{2m+1} \binom{2m+1}{i} \frac{(2i)!}{(2i-2m-2)!} x^{2i-2m-1}(-1)^{2m+1-i}$$

$$\frac{\mathrm{d}^{2m}}{\mathrm{d}x^{2m}}(x^2-1)^{2m} = \sum_{i=m}^{2m} \binom{2m}{i} \frac{(2i)!}{(2i-2m)!} x^{2i-2m}(-1)^{2m-i}$$

Combine these two results,

$$Q_n(x) = \begin{cases} \frac{1}{2^{2m+1}(2m+1)!} \sum_{i=m+1}^{2m+1} \binom{2m+1}{i} \frac{(2i)!}{(2i-2m-2)!} x^{2(i-m)-1}(-1)^{2m+1-i} & n = 2m+1 \\ \frac{1}{2^{2m}(2m)!} \sum_{i=m}^{2m} \binom{2m}{i} \frac{(2i)!}{(2i-2m)!} x^{2(i-m)}(-1)^{2m-i} & n = 2m \end{cases}$$

We can see that when $n = 2m$, $Q_n(x)$ is a sum of x^{2k} with some coefficients, which means it is an even function. When $n = 2m+1$, $Q_n(x)$ is a sum of x^{2k+1} with some coefficients, which means it is an odd function. Since $Q_n(x)$ is even when n is even, and is odd when n is odd, we have

$$Q_n(-x) = (-1)^n Q_n(x)$$

□

3. Show that for any $x \in [-1,1]$, the Legendre polynomials obey the recurrence relation

$$(n+1)Q_{n+1}(x) = (2n+1)xQ_n(x) - nQ_{n-1}(x)$$

Proof. This can be proved with pure calculation, with the application of general Leibniz rule.

$$\begin{aligned} (n+1)Q_{n+1}(x) &= \frac{n+1}{2^{n+1}(n+1)!} \frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}} \left[(x^2-1)^{n+1} \right] \\ &= \frac{1}{2^{n+1}n!} \frac{\mathrm{d}^n}{\mathrm{d}x^n} \left[2x(n+1)(x^2-1)^n \right] \\ &= \frac{n+1}{2^n n!} \frac{\mathrm{d}^n}{\mathrm{d}x^n} \left[x(x^2-1)^n \right] \end{aligned}$$

Apply the general Leibniz rule on the differential,

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n} \left[x(x^2-1)^n \right] = x \frac{\mathrm{d}^n}{\mathrm{d}x^n} \left[(x^2-1)^n \right] + n \frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}} \left[(x^2-1)^n \right] \tag{4.3}$$

Apply the substitution to get

$$\begin{aligned}(n+1)Q_{n+1}(x) &= \frac{n+1}{2^n n!}x\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right] + \frac{n+1}{2^n n!}n\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right] \\ &= x(n+1)Q_n(x) + \frac{n(n+1)}{2^n n!}\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right] \\ &= x(2n+1)Q_n(x) - nxQ_n(x) - \frac{n}{2^n n!}x\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right]\end{aligned}$$

The last term without its coefficient is already in (4.3), so

$$x\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[\left(x^2-1\right)^n\right] = \frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[x\left(x^2-1\right)^n\right] - n\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right]$$

Apply substitution,

$$\begin{aligned}(n+1)Q_{n+1}(x) &= x(2n+1)Q_n(x) + \frac{n+1}{2^n(n-1)!}\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right] \\ &\quad - \frac{n}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[x\left(x^2-1\right)^n\right] + \frac{n}{2^n n!}\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[n\left(x^2-1\right)^n\right] \\ &= x(2n+1)Q_n(x) + \frac{1}{2^n(n-1)!} \\ &\quad \cdot \frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left\{(n-1)\left(x^2-1\right)^n - \frac{\mathrm{d}}{\mathrm{d}x}\left[x\left(x^2-1\right)^n\right] + n\left(x^2-1\right)^n\right\} \\ &= x(2n+1)Q_n(x) \\ &\quad + \frac{1}{2^n(n-1)!}\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[2n\left(x^2-1\right)\left(x^2-1\right)^{n-1} - 2x^2 n\left(x^2-1\right)^{n-1}\right] \\ &= x(2n+1)Q_n(x) - \frac{2n}{2^n(n-1)!}\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}}\left[\left(x^2-1\right)^n\right] \\ &= x(2n+1)Q_n(x) - nQ_{n-1}(x)\end{aligned}$$

This proves the recurrence relation of Legendre polynomials. □

4. Prove that

$$Q_n(x) = \sum_{i=0}^{n}(-1)^i\binom{n}{i}^2\left(\frac{1+x}{2}\right)^{n-i}\left(\frac{1-x}{2}\right)^i$$

Proof. This is a direct result from the general Leibniz rule.

$$Q_n(x) = \frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(x-1)^n(x+1)^n\right]$$

Apply the general Leibniz rule, where

$$f(x) = (x-1)^n \quad g(x) = (x+1)^n$$

And their higher order derivatives are

$$\frac{\mathrm{d}^{n-i}}{\mathrm{d}x^{n-i}}f(x) = n(n-1)\cdots(i+1)(x-1)^i = \frac{n!}{i!}(x-1)^i$$
$$\frac{\mathrm{d}^{i}}{\mathrm{d}x^{i}}g(x) = n(n-1)\cdots(n-i+1)(x-1)^i = \frac{n!}{(n-i)!}(x+1)^i$$

Plug these two results in the binomial expansion of Legendre polynomials,

$$\begin{aligned} Q_n(x) &= \frac{1}{2^n n!}\sum_{i=0}^{n}\binom{n}{i}\frac{n!}{i!}(x-1)^i\frac{n!}{(n-i)!}(x+1)^{n-i} \\ &= 2^{-n}\sum_{i=0}^{n}\binom{n}{i}^2(x-1)^i(x+1)^{n-i} \\ &= \sum_{i=0}^{n}(-1)^i\binom{n}{i}^2\left(\frac{1+x}{2}\right)^{n-i}\left(\frac{1-x}{2}\right)^{i} \end{aligned}$$

This proves the result. □

Exercise 4.2 — Interpolation

Let f be a continuous function for which we know the eight points defined in the following table.

x	-5	-1	0	1	3	5	10	10
$f(x)$	781	5	1	1	61	521	9091	19141

Determine $f(2)$.

Answer. We apply polynomial interpolation. Since there are 8 distinct points to pass through, we need a polynomial with degree at least 7. Assume the polynomial has form

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7$$

Then with the given points we have the following system of linear equations

$$\begin{bmatrix} x_1^0 & x_1^1 & x_1^2 & x_1^3 & x_1^4 & x_1^5 & x_1^6 & x_1^7 \\ x_2^0 & x_2^1 & x_2^2 & x_2^3 & x_2^4 & x_2^5 & x_2^6 & x_2^7 \\ x_3^0 & x_3^1 & x_3^2 & x_3^3 & x_3^4 & x_3^5 & x_3^6 & x_3^7 \\ x_4^0 & x_4^1 & x_4^2 & x_4^3 & x_4^4 & x_4^5 & x_4^6 & x_4^7 \\ x_5^0 & x_5^1 & x_5^2 & x_5^3 & x_5^4 & x_5^5 & x_5^6 & x_5^7 \\ x_6^0 & x_6^1 & x_6^2 & x_6^3 & x_6^4 & x_6^5 & x_6^6 & x_6^7 \\ x_7^0 & x_7^1 & x_7^2 & x_7^3 & x_7^4 & x_7^5 & x_7^6 & x_7^7 \\ x_8^0 & x_8^1 & x_8^2 & x_8^3 & x_8^4 & x_8^5 & x_8^6 & x_8^7 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \\ f(x_3) \\ f(x_4) \\ f(x_5) \\ f(x_6) \\ f(x_7) \end{bmatrix}$$

With numerical value,

$$\begin{bmatrix} 1 & -5 & 25 & -125 & 625 & -3125 & 15625 & -78125 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 9 & 27 & 81 & 243 & 729 & 2187 \\ 1 & 5 & 25 & 125 & 625 & 3125 & 15625 & 78125 \\ 1 & 10 & 100 & 1000 & 10000 & 100000 & 1000000 & 10000000 \\ 1 & 12 & 144 & 1728 & 20736 & 248832 & 2985984 & 35831808 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} = \begin{bmatrix} 781 \\ 5 \\ 1 \\ 1 \\ 61 \\ 521 \\ 9091 \\ 19141 \end{bmatrix}$$

Write it in the matrix form,

$$Xa = f$$

Then, we can solve for a as

$$a = X^{-1}f, \quad \text{or in MATLAB}^{\text{TM}} \text{ syntax, } a = X\backslash f$$

By MATLAB™, we get

$$a = (1, -1, 1, -1, 1, 0, 0, 0)^T$$

So the interpolating polynomial is

$$p(x) = 1 - x + x^2 - x^3 + x^4$$

Plug in $x = 2$, we get the estimate

$$f(2) \approx p(2) = 11$$

The MATLAB™ code is attached here.

```
x = [-5 -1 0 1 3 5 10 12];
X = [x.^0;x.^1;x.^2;x.^3;x.^4;x.^5;x.^6;x.^7]';
y = [781 5 1 1 61 521 9091 19141]';
X\y
```

□

Exercise 4.3 — Newton's form of the interpolation polynomial

Let f be a continuous function, P^n be its interpolation polynomial in the points $x_0, \cdots, x_n$.

1. Let $P^0(x) = f(x_0)$ be the interpolation polynomial in a single point x_0.

   (a) Show that for two points x_0 and x_1, $P^1(x) = P^0(x) + \frac{f(x_1)-f(x_0)}{x_1-x_0}(x - x_0)$.

   Proof. We evaluate $P^1(x)$ at x_0 and x_1.

$$P^1(x_0) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x_0 - x_0) = f(x_0)$$

$$P^1(x_1) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x_1 - x_0) = f(x_0) + f(x_1) - f(x_0) = f(x_1)$$

Obviously, $P^1(x)$ is a polynomial with degree of at most 1, so it represents a straight line, and two distinct points are enough to determine a unique $P^1(x)$. □

(b) Determine a polynomial R of degree at most two, such that $P^2(x) = P^1(x) + R(x)$, for three nodes x_0, x_1 and x_2.

Proof. Since $P^2(x)$ is an interpolation polynomial in the points x_0, x_1, x_2, we have $P^2(x_2) = f(x_2)$. Since $P^1(x)$ has already satisfied $P^1(x_0) = f(x_0)$ and $P^1(x_1) = f(x_1)$, we need $R(x_0) = R(x_1) = 0$. Thus, we construct $R(x)$ to have zeros at x_0 and x_1, with an undetermined coefficient a_2 to pass through $(x_2, f(x_2))$.

$$P^2(x) = P^1(x) + a_2(x - x_0)(x - x_1)$$

With boundary condition, we have

$$f(x_2) = P^1(x_2) + a_2(x_2 - x_0)(x_2 - x_1)$$

So we can solve for a_2

$$a_2 = \frac{f(x_2) - P^1(x_2)}{(x_2 - x_0)(x_2 - x_1)}$$

Hence, we obtain

$$R(x) = \frac{f(x_2) - P^1(x_2)}{(x_2 - x_0)(x_2 - x_1)}(x_2 - x_0)(x_2 - x_1)$$

□

(c) Prove by induction that

$$P^j(x) = P^{j-1}(x) + a_j \prod_{k=0}^{j-1}(x - x_k), \quad \text{where } a_j \text{ only depends on } x_0, \cdots x_j$$

Proof. For $j = 1$ and $j = 2$, we have already proved in the previous steps. Assume this holds for $j = n$, which means for all m such that $0 \le m \le n$, $P^n(x_m) = f(x_m)$. We need to prove

$$\forall m \text{ such that } 0 \le m \le n+1, P^n(x_m) + a_{n+1} \prod_{k=0}^{n}(x_m - x_k) = f(x_m)$$

This is obvious for $0 \le m \le n$, since the product always contains a zero factor and is zero. So we just need the condition for $m = n + 1$.

$$P^n(x_{n+1}) + a_{n+1} \prod_{k=0}^{n}(x_{n+1} - x_k) = f(x_{n+1})$$

Then we obtain the coefficient

$$a_{n+1} = \frac{f(x_{n+1}) - P^n(x_{n+1})}{\prod_{k=0}^{n}(x_{n+1} - x_k)}$$

This only depends on $x_0, \cdots, x_{n+1}$. So we have

$$P^{n+1}(x) := P^n(x) + \frac{f(x_{n+1}) - P^n(x_{n+1})}{\prod_{k=0}^{n}(x_{n+1} - x_k)} \prod_{k=0}^{n}(x - x_k)$$

which interpolates $f(x)$ in $x_0, \cdots, x_{n+1}$. This is a polynomial with degree at most $n+1$. By invertibility of Vandermonde matrix, we know that $n+1$ distinct points can uniquely determine a polynomial with degree at most $n+1$. So this definition exactly define the interpolation polynomial of $f(x)$. By the principle of mathematical induction, we have it true for all $j \in \mathbb{N}^*$. □

2. Denoting a_j by $f[x_0, \cdots, x_j]$, show that

$$P^n(x) = f(x_0) + \sum_{j=1}^{n} f[x_0, \cdots, x_j] \prod_{k=0}^{j-1}(x - x_k)$$

Proof. From the previous proof, we have already proved that

$$P^n(x) = P^{n-1}(x) + a_n \prod_{k=0}^{n-1}(x - x_k)$$

With this recursive relation, we can prove the proposition by induction. For $j = 0$, it has been proved that

$$P^0(x) = f(x_0)$$

Assume it is true for $j = n$, then for $n+1$,

$$P^{n+1}(x) = P^n(x) + a_{n+1} \prod_{k=0}^{n}(x - x_k)$$

By induction assumption,

$$P^n(x) = f(x_0) + \sum_{j=1}^{n} f[x_0, \cdots, x_j] \prod_{k=0}^{j-1}(x - x_k)$$

Apply the substitution,

$$\begin{aligned} P^{n+1}(x) &= f(x_0) + \sum_{j=1}^{n} f[x_0, \cdots, x_j] \prod_{k=0}^{j-1}(x - x_k) + a_{n+1} \prod_{k=0}^{n}(x - x_k) \\ &= f(x_0) + \sum_{j=1}^{n+1} f[x_0, \cdots, x_j] \prod_{k=0}^{j-1}(x - x_k) \end{aligned}$$

By the principle of mathematical induction, we have it true for all $j \in \mathbb{N}^*$. □

3. Prove that for any $k \in \mathbb{N}^*$,

$$\begin{cases} f[x_k] = f(x_k) \\ f[x_0, \cdots, x_k] = \frac{f[x_1, \cdots, x_k] - f[x_0, \cdots, x_{k-1}]}{x_k - x_0} \end{cases}$$

Proof. We prove by induction. For $n = 1$, we have proved in the previous step that

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

Suppose this is true for $k = n$, then according to the previous result, we have the interpolating polynomial with degree at most n passing $(x_0, f(x_0))$, $(x_1, f(x_1))$, $\cdots$, $(x_n, f(x_n))$

$$P_1^n(x) = f(x_0) + \sum_{j=1}^{n} f[x_0, \cdots, x_j] \prod_{k=0}^{j-1} (x - x_k)$$

When $k = n + 1$, we also have another polynomial of degree at most n interpolating $(x_1, f(x_1)), (x_2, f(x_2)), \cdots, (x_{n+1}, f(x_{n+1}))$

$$P_2^n(x) = f(x_1) + \sum_{j=2}^{n+1} f[x_1, \cdots, x_j] \prod_{k=1}^{j} (x - x_k)$$

We can express $P^{n+1}(x)$ by these two interpolating polynomials

$$P^{n+1}(x) = \frac{x - x_0}{x_{n+1} - x_0} P_2^n(x) + \frac{x_{n+1} - x}{x_{n+1} - x_0} P_1^n(x)$$

This is constructed to satisfy

$$P^{n+1}(x_i) = \begin{cases} P_1^n(x_0) = f(x_0) & i = 0 \\ f(x_i) & i = 1, \cdots, n \\ P_2^n(x) = f(x_{n+1}) & i = n + 1 \end{cases}$$

So this is indeed the interpolating polynomial of these $n+2$ points. And the coefficient of x^{n+1} is

$$\frac{f[x_1, \cdots, x_{n+1}] - f[x_0, \cdots, x_n]}{x_{n+1} - x_0}$$

By the principle of mathematical induction, we have it true for all $k \in \mathbb{N}^*$. □

4. Write the pseudocode of a clear algorithm to compute $P^n(x)$ when given $n + 1$ nodes $x_0, \cdots, x_n$ and the value of f at those nodes.

We now consider the case of equidistant nodes, *i.e.*, $x_i = x_0 + ih$, for any $0 \le i \le n$, and some $h \in \mathbb{R}^+$. Denoting $f[x_i]$ by f_i , $0 \le i \le n$, we recursively define the operator ∇ such that

$$\begin{cases} \nabla^0 f_i = f_i \\ \nabla^{k+1} f_i = \nabla^k f_{i+1} - \nabla^k f_i \end{cases}$$

Input: $n+1$ nodes: $(x_0, f(x_0)), \cdots, (x_n, f(x_n))$, $x_0, \cdots, x_n$ stored in vector x and $f(x_0), \cdots, f(x_n)$ stored in vector y
Output: Coefficients of $P^n(x)$: $a_0, \cdots, a_n$, stored in vector a

```
A=zeros(n+1,n+1);
for i=1:n+1 do
    A(i,1) = y(i);
end
a(1) = A(1,1);
for j=2:n+1 do
    for i=j:n+1 do
        A(i,j) = (A(i-1,j-1)-A(i,j-1))/(x(i-j+1)-x(i));
    end
    a(j) = A(j,j);
end
```

Algorithm 4.1: Newton's interpolation

5. Show that $\forall i, k, \in \mathbb{N}, f[x_i, \cdots, x_{i+k}] = \frac{1}{k!h^k}\nabla^k f_i$.

Proof. We prove by induction. When $k = 1$, we have already proved in the previous step that

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

The labeling of the points does not make a difference, and the difference between consecutive nodes is h, so

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{h}$$

Assume this holds for $k = m$, which means

$$f[x_i, x_{i+1}, \cdots, x_{i+m}] = \frac{1}{m!h^m}\nabla^m f_i$$

According to the previous proof, for $m + 1$,

$$\begin{aligned} f[x_i, x_{i+1}, \cdots, x_{i+m+1}] &= \frac{f[x_{i+1}, x_{i+2}, \cdots, x_{i+m+1}] - f[x_i, x_{i+2}, \cdots, xi + m]}{x_{i+m+1} - x_i} \\ &= \frac{\frac{\nabla^m f_{i+1}}{m!h^m} - \frac{\nabla^m f_i}{m!h^m}}{(m+1)h} \\ &= \frac{\nabla^{m+1} f_i}{(m+1)!h^{m+1}} \end{aligned}$$

By the principle of mathematical induction, we have it true for all $i, k \in \mathbb{N}$. □

6. Observing that $\binom{s}{k} = \frac{1}{k!}\prod_{j=0}^{k-1}(s-j)$, prove that

$$P^n(x) = f_0 + \sum_{k=1}^{n} \binom{s}{k} \nabla^k f_0, \quad \text{where } s = \frac{x - x_0}{h}$$

Proof. According to the previous proof,

$$\begin{aligned} P^n(x) &= f(x_0) + \sum_{k=1}^{n} f[x_0, \cdots, x_j] \prod_{j=0}^{k-1}(x - x_j) \\ &= f_0 + \sum_{k=1}^{n} f[x_0, \cdots, x_j] \prod_{j=0}^{k-1}(x - x_0 - jh) \\ &= f_0 + \sum_{k=1}^{n} f[x_0, \cdots, x_j] h^k \prod_{j=0}^{k-1}\left(\frac{x - x_0}{h} - j\right) \\ &= f_0 + \sum_{k=1}^{n} f[x_0, \cdots, x_j] k! h^k \cdot \frac{1}{k!}\prod_{j=0}^{k-1}(s - j) = f_0 + \sum_{k=1}^{n} \nabla^k f_0 \binom{s}{k} \end{aligned}$$

□

7. Write the pseudocode of an algorithm which takes a step h as input, a number of nodes, the value of f at each of those nodes, and a value x. The algorithm should return $P^n(x) \approx f(x)$.

Input: Step length h, number of nodes $n+1$, the value of f at those nodes, stored in vector y, and a value x

Output: Estimation of f at x using interpolation with polynomial of degree at most n, denoted as f_x

```
A=zeros(n+1,n+1);
for i=1:n+1 do
    A(i,1) = y(i);
end
a(1) = A(1,1);
for j=2:n+1 do
    for i=j:n+1 do
        A(i,j) = (A(i,j-1)-A(i-1,j-1))/((j-1)h);
    end
    a(j) = A(j,j);
end
f_x = 0;
for i=0:n do
    f_x = f_x + a(i)x^i
end
```

Algorithm 4.2: Newton's interpolation with evaluation

Assignment 5

Numerical Quadrature

Exercise 5.1 — Lebesgue constant for Chebyshev nodes

We recall that the Lebesgue number is defined by $\Lambda_n = \max_{x\in[a,b]} \sum_{i=0}^{n} |l_i(x)|$ and the Chebyshev polynomials by $T_n(x) = \cos(n \arccos(x))$. The roots $x_i, 0 \leq i \leq n$, of Chebyshev polynomials are given by $x_i = \cos\theta_i$, with $\theta_i = \frac{2i+1}{2(n+1)}\pi$.

1. Let L_i be the Lagrange polynomials associated to the node x_i.

   (a) Prove that
   $$L_i(x) = \frac{T_{n+1}(x)}{(x - x_i)T'_{n+1}(x)}.$$

   Proof. Since the Chebyshev polynomials have roots x_i, we can rewrite it as
   $$T_{n+1}(x) = c \cdot \prod_{k=0}^{n}(x - x_k).$$
   Then its derivative is
   $$T'_{n+1} = c \cdot \sum_{j=0}^{n} \left[\prod_{\substack{k=0 \\ k\neq j}}^{n}(x - x_k) \right].$$
   At $x = x_i$, the products are zero, so only the term without $x - x_i$ remains, *i.e.*,
   $$T'_{n+1} = c \cdot \prod_{\substack{k=0 \\ k\neq i}}^{n}(x - x_k).$$
   Then it is easy to get
   $$\frac{T_{n+1}(x)}{(x - x_i)T'_{n+1}(x_i)} = \frac{c \cdot \prod_{k=0}^{n}(x - x_k)}{(x - x_i) \cdot c \cdot \prod_{\substack{k=0 \\ k\neq i}}^{n}(x - x_k)} = \prod_{\substack{k=0 \\ k\neq i}}^{n} \frac{x - x_k}{x_i - x_k},$$
   which is $L_i(x)$, by definition. □

(b) Show that

$$T'_{n+1} = \frac{n+1}{\sqrt{1-\cos^2\theta}}\sin(n+1)\theta,$$

and

$$T'_{n+1}(x_k) = (-1)^k \frac{n+1}{\sin\theta_k}.$$

Proof. Apply chain rule on differentiation,

$$\begin{aligned} T'_{n+1}(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\cos[(n+1)\arccos x] = \frac{\mathrm{d}\cos t}{\mathrm{d}t}\cdot\frac{\mathrm{d}[(n+1)\arccos x]}{\mathrm{d}x} \\ &= -\sin[(n+1)\arccos x]\cdot\frac{-(n+1)}{\sqrt{1-x^2}} \\ &= \frac{n+1}{\sqrt{1-x^2}}\cdot\sin[(n+1)\arccos x]. \end{aligned}$$

Plug in $x_k = \cos\theta_k$,

$$\begin{aligned} T'_{n+1}(x_k) &= \frac{n+1}{\sin\theta_k}\cdot\sin\left[(n+1)\cdot\frac{2k+1}{2(n+1)}\cdot\pi\right] \\ &= \frac{n+1}{\sin\theta_k}\cdot\sin\left[(2k+1)\cdot\frac{\pi}{2}\right] \\ &= (-1)^k\frac{n+1}{\sin\theta_k}. \end{aligned}$$

□

(c) Conclude that

$$\sum_{i=0}^{n}|L_i(1)| \geq \frac{1}{n+1}\sum_{i=0}^{n}\cot\frac{\theta_i}{2}.$$

Proof. We apply the substitution from the result of previous questions.

$$\begin{aligned} L_i(1) &= \frac{T_{n+1}(1)}{(1-x_i)T'_{n+1}(x_i)} \\ &= \frac{1}{(1-\cos\theta_i)\cdot(-1)^i\frac{n+1}{\sin\theta_i}} \\ &= \frac{(-1)^{-i}}{n+1}\cdot\frac{\sin\theta_i}{1-\cos\theta_i} = \frac{(-1)^{-i}}{n+1}\cdot\cot\frac{\theta_i}{2}. \end{aligned}$$

So the sum of their absolute values has

$$\sum_{i=0}^{n}|L_i(1)| = \sum_{i=0}^{n}\left|\frac{(-1)^{-i}}{n+1}\cot\frac{\theta_i}{2}\right| = \frac{1}{n+1}\sum_{i=0}^{n}\left|\cot\frac{\theta_i}{2}\right| \geq \frac{1}{n+1}\sum_{i=0}^{n}\cot\frac{\theta_i}{2}.$$

□

2. Show that
$$\frac{1}{n+1}\sum_{i=0}^{n}\cot\frac{\theta_i}{2}\geq\frac{2}{\pi}\int_{\frac{\theta_0}{2}}^{\frac{\pi}{2}}\cot t\ \mathrm{d}t.$$

Proof. By the property of the definite integral, we can split the integral to several integrals,
$$\frac{2}{\pi}\int_{\frac{\theta_0}{2}}^{\frac{\pi}{2}}\cot t\ \mathrm{d}t=\sum_{i=0}^{n-1}\frac{2}{\pi}\int_{\frac{\theta_i}{2}}^{\frac{\theta_{i+1}}{2}}\cot t\ \mathrm{d}t+\frac{2}{\pi}\int_{\frac{\theta_n}{2}}^{\frac{\pi}{2}}\cot t\ \mathrm{d}t.$$
Since the cotangent function is monotonically decreasing on the interval $\left(0,\frac{\pi}{2}\right)$, the integral can be telescoped to have the left value on the whole interval,
$$\int_{\frac{\theta_i}{2}}^{\frac{\theta_{i+1}}{2}}\cot t\ \mathrm{d}t\leq\cot\frac{\theta_i}{2}\cdot\left(\frac{\theta_{i+1}}{2}-\frac{\theta_i}{2}\right)=\frac{\pi}{2(n+1)}\cdot\cot\frac{\theta_i}{2},$$
while the last term is
$$\frac{2}{\pi}\int_{\frac{\theta_n}{2}}^{\frac{\pi}{2}}\cot t\ \mathrm{d}t\leq\cot\frac{\theta_n}{2}\left(\frac{\pi}{2}-\frac{\theta_n}{2}\right)=\frac{\pi}{4(n+1)}\cot\frac{\theta_n}{2}\leq\frac{\pi}{2(n+1)}\cot\frac{\theta_n}{2}.$$
Combine these two results together,
$$\frac{2}{\pi}\int_{\frac{\theta_0}{2}}^{\frac{\pi}{2}}\cot t\ \mathrm{d}t\leq\sum_{i=0}^{n-1}\frac{2}{\pi}\cdot\frac{\pi}{2(n+1)}\cdot\cot\frac{\theta_i}{2}+\frac{2}{\pi}\cdot\frac{\pi}{2(n+1)}\cot\frac{\theta_n}{2}=\frac{1}{n+1}\sum_{i=0}^{n}\cot\frac{\theta_i}{2}$$
□

3. Using question 2, conclude that $\Lambda_n\geq\frac{2}{\pi}\ln n$.

Exercise 5.2 — Interpolation

Let $\mathcal{C}[a,b], a,b\in\mathbb{R}$, be the set of the continuous functions over $[a,b]$, endowed with the usual norm for uniform convergence, $||u||_\infty=\max_{x\in[a,b]}|u(x)|$. For some $n\in\mathbb{N}$ we define the collection of points $(x_k,y_k), k\in\{0,\cdots,n\}$, such that $a\leq x_0<y_0<x_1<y_1<\cdots<x_n<y_n\leq b$ and consider the following application
$$\begin{aligned}\phi:\mathcal{C}[a,b]&\to\mathbb{R}^{n+1}\\ f&\to(m_0(f),m_1(f),\cdots,m_n(f)),\end{aligned}$$
with for all $k\in\{0,\cdots,n\}, m_k(f)=\frac{f(x_k)+f(y_k)}{2}$.

1. Let $f\in\mathcal{C}[a,b]$ such tat $\phi(f)=0$. Show that $\forall k,\exists\xi_k\in[x_k,y_k]$ such that $f(\xi_k)=0$.
2. Prove that if ϕ is restricted to $\mathbb{R}_n[x]$, then ϕ is injective. Conclude on the existence of a unique polynomial $P_n\in\mathbb{R}_n[x]$ such that $\phi(P_n)=\phi(f)$.
3. Assuming $f\in\mathcal{C}^{n+1}[a,b]$, prove that
$$||f-P_n||_\infty=\frac{(b-a)^{n+1}}{(n+1)!}\sup_{x\in[a,b]}|f^{(n+1)}(x)|.$$

Exercise 5.3 — Trigonometric polynomials

Let $x \in [0,1]$ and $\theta \in [-\pi, \pi]$. For $n \in \mathbb{N}$, we denote by

$$T_n = \left\{ Q_n \;\middle|\; Q_n(\theta) = \frac{a_0}{\sqrt{2}} + \sum_{k=1}^{n} a_k \cos(k\theta) \right\}$$

the set of the trigonometric polynomials of degree less than n.

1. Prove that for any $0 \leq k \leq n, (\cos\theta)^k \in T_n$. Conclude that ϕ, which maps $P_n(x)$ into $Q_n(\theta) = P_n(\cos\theta)$ is a linear bijection from $\mathbb{R}_n[x]$ into T_n.

2. For $f \in \mathcal{C}^{n+1}[1,1]$, we define $F(\theta) = f(\cos\theta)$. Show the existence of $Q_n \in T_n$, such that $Q_n(\theta_i) = F(\theta_i)$, where $\theta_i = \frac{2i+1}{2n+1}\pi, 0 \leq i \leq \pi$.

3. Prove that finding $Q_n \in T_n$ in the previous question is equivalent to solving the linear system $La = b$, with $a = (a_0, \cdots, a_n)^T$ such that

$$Q_n(\theta) = \frac{a_0}{\sqrt{2}} + \sum_{k=1}^{n} a_k \cos(k\theta)$$

4. Show that for any $\theta \in (-\pi, \pi)$, there exists $\xi \in (-1,1)$, such that

$$F(\theta) - Q_n(\theta) = \frac{\cos(n+1)\theta}{2^n(n+1)!} f^{(n+1)}(\xi).$$

Assignment 6

Numerical Integration

Exercise 6.1 — Richardson extrapolation

In this exercise we investigate Richardson extrapolation, a sequence acceleration method which can be used to improve the rate of convergence of a quadrature formula.

Let $a_0 \in \mathbb{R}$ be a value to be computed and $A(t), t > 0$ be such that

1. $\lim_{t\to 0} A(t) = a_0$
2. For all $a \leq 0$, there exists $a_1, \cdots, a_n$, and c_{n+1} such that

$$A(t) = a_0 + \sum_{i=1}^{n} a_i t^i + R_{n+1}(t), \quad \text{with } |R_{n+1}(t)| \leq c_{n+1}(t);$$

Let $(A_n)_{n\in\mathbb{N}}$ be the sequence defined by

$$\begin{cases} A_0(t) & = A(t) \\ A_n(t) & = \frac{r^n A_{i-1}(t) - A_{i-1}(rt)}{r^n - 1}, \quad n \geq 1 \text{ and } r > 1 \text{ a constant.} \end{cases}$$

1. Prove that for all $n \in \mathbb{N}$, $A_n(t) = a_0 + O(t^{n+1})$.
2. Fixing $t_0 > 0$ and $r_0 > 1$, we define a sequence $(t_m)_{m\geq 0}$ such that $t_m = r_0^{-m} t_0$.
   (a) Show that when n is fixed then $\lim_{m\to\infty} A_n(t_m) = a_0$.
   (b) Show that $A_n(t_m) = a_0 + O(r_0^{-m(n+1)})$
3. For m and n two integers, we define a matrix M whose entry at column n and row m is $A_{m,n} = A_n(t_m)$. Write the pseudocode of a clear algorithm generating the matrix M and returning a_0.

Exercise 6.2 — Integration

We consider the quadrature formula

$$\int_a^b f(x)\ \mathrm{d}x \approx (b-a) f\left(\frac{a+b}{2}\right).$$

1. Why does this formula fall under Peano's method?
2. Determine Peano kernel for this formula, and show that it keeps a constant sign.
3. Conclude on the existance of $\xi \in [a,b]$, such that for any $f \in \mathcal{C}^2[a,b]$, the error is expressed as
$$E(f) = \frac{1}{24} f''(\xi)(b-a)^3.$$

Exercise 6.3 — Gauss' method

Let $(q_k)_{k\in\mathbb{N}}$ be a sequence of polynomials such that

$$q_k(x) = \frac{\sin(k+1)\theta}{\sin\theta}, \quad \text{with } x = \cos\theta.$$

1. Let $w(x) = \sqrt{1-x^2}$ be a function over $(-1, 1.$
   (a) Show that w is a weight function.
   (b) Show that the $(q_k)_{k\in\mathbb{N}}$ define a sequence of orthogonal polynomials for the weight function w.
   (c) Determine the orthonormal polynomials $(p_k)_k$ associated to $(q_k)_k$.
2. We consider the Gauss' method of order $2n+1$ defined by
$$\int_{-1}^{1} f(x)w(x)\,\mathrm{d}x \approx \sum_{k=0}^{n} A_k f(x_k).$$
   (a) Determine all the $x_k, 0 \le k \le n$.
   (b) Show that for all $0 \le k \le n$,
$$A_k = \frac{\pi}{n+2}\sin^2\frac{(k+1)\pi}{n+2}.$$
   (c) Assuming $f \in \mathcal{C}^{2n+2}[1,1]$, show that there exists $\xi(1,1)$, such that the error of the method is given by
$$E_n(f) = c \cdot \frac{f^{(2n+2)}(\xi)}{(2n+2)!}, \quad \text{where } c \text{ is a constant to be determined.}$$

Assignment 7

Iteration Methods

Exercise 7.1 — Lipschitz continuity vs. Differentiability

A function $f : \mathbb{R} \to \mathbb{R}$ is differentiable in some open interval $\mathcal{I} \subset \mathbb{R}$ if it is differentiable at every point of $\mathcal{I}$, and it is Lipschitz continuous if there is a constant $c \geq 0$ such that

$$|f(x_1) - f(x_2) \leq c|x_1 - x_2| \qquad \forall x_1, x_2 \in \mathcal{I}.$$

1. Give a function, with proof, that is differentiable but its derivative is not bounded in some open interval.

   Proof.
   $$f(x) = \tan x, \quad x \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$$
   is differentiable since its derivative is $f'(x) = \sec^2 x$, which is not bounded on its domain. □

2. Suppose f is differentiable and f' is bounded in some open interval. Prove that f is Lipschitz continuous in that interval.

   Proof. We first introduce Cauchy's Mean Value Theorem,

   Theorem 7.1 (Mean Value Theorem)**.** *If a function f is continuous on the closed interval $[a, b]$, and differentiable on the open interval (a, b), then there exists a point $\xi \in (a, b)$ such that*
   $$f'(\xi) = \frac{f(b) - f(a)}{b - a}.$$

   Proof. Define $g(x) = f(x) - rx$, where r is a constant. Since f is continuous on $[a, b]$ and differentiable on (a, b), the same is true for g. We now want to choose r
   $$\begin{aligned} g(a) = g(b) &\iff f(a) - ra = f(b) - rb \\ &\iff r(b - a) = f(b) - f(a) \\ &\iff r = \frac{f(b) - f(a)}{b - a}. \end{aligned}$$

By Rolle's theorem, since g is differentiable and $g(a) = g(b)$, there is some $\xi \in (a, b)$ for which $g'(\xi) = 0$ and thus

$$f'(\xi) = g'(\xi) + r = \frac{f(b) - f(a)}{b - a}$$

as required. □

Then, we can set the Lipschitz constant to be

$$c = \sup_{\xi \in (a,b)} |f'(\xi)|.$$

Since f' is bounded on (a, b), the Lipschitz constant is finite. Apply Cauchy's Mean Value Theorem, we get

$$\forall x_1, x_2 \in (a, b), \quad |f(x_1) - f(x_2)| = |f'(\xi)| \cdot |x_1 - x_2| \leq c \cdot |x_1 - x_2|,$$

which means f is Lipschitz continuous. □

3. Give a function, with proof, that is differentiable but not Lipschitz continuous in some open interval.

Proof. This is exactly the same as differentiable but the derivative not bounded.

$$f(x) = \sqrt[3]{x}, \quad x \in (-1, 1)$$

is differentiable since its derivative is $f'(x) = \frac{1}{3}x^{-\frac{2}{3}}$. But it is not Lipschitz continuous since

$$\frac{|f(x_1) - f(x_2)|}{|x_1 - x_2|} = \frac{1}{x_1^{\frac{2}{3}} + x_1^{\frac{1}{3}}x_2^{\frac{1}{3}} + x_2^{\frac{2}{3}}}$$

is not bounded on $x_1, x_2 \in (-1, 1)$ so that the Lipschitz constant does not exist. □

4. Give a function, with proof, that is Lipschitz continuous but not differentiable in some open interval.

Proof.

$$f(x) = |x|, \quad x \in (-1, 1)$$

is Lipschitz continuous with Lipschitz constant

$$c = \limsup_{x_1, x_2 \in (-1,1)} \left| \frac{f(x_1) - f(x_2)}{x_1 - x_2} \right| = 1.$$

But it is not differentiable at $x = 0$. □

Exercise 7.2 — Fixed point iteration convergence condition

A fixed point of a function $g(x)$ is a real number x^* such that

$$g(x^*) = x^*.$$

Assume the followings:

1. The function g and its derivative g' are continuous on $[a, b]$, *i.e.*, $g, g' \in \mathcal{C}[a, b]$.
2. The function g is bounded below by a and above by b, *i.e.*, $\forall x \in [a, b], g(x) \in [a, b]$.
3. The initial point x_0 is an interior point of $[a, b]$, *i.e.*, $x_0 \in (a, b)$.

Show the followings are true.

1. If $0 \leq |g'(x)| < 1, \forall x \in [a, b]$, the fixed-point iteration will converge to the unique fixed point $x^* \in [a, b]$.

   Proof. Define the Lipschitz constant to be

   $$c = \sup_{\xi \in (a,b)} g'(\xi).$$

   It comes from definition that $0 \leq c < 1$. Define $h(x) = g(x) - x$, and it is monotonically decreasing since

   $$h'(x) = g'(x) - 1 \in (-2, 0), \quad \forall x \in [a, b].$$

   Thus, $h(x) = 0$ has at most one solution, and so is $g(x) = x$. This proves the uniqueness of x^*. Then apply Cauchy's Mean Value Theorem,

   $$|g(x_{k+1}) - g(x_k)| = |g'(\xi)| \cdot |x_{k+1} - x_k| \leq c \cdot |x_{k+1} - x_k|.$$

   Apply the iteration relation,

   $$x_{k+1} = g(x_k), \qquad x_k = g(x_{x-1}),$$

   we get

   $$|g(x_{k+1}) - g(x_k)| \leq c \cdot |g(x_k) - g(x_{k-1})|.$$

   Such a recursive relation deduces

   $$|g(x_{k+1}) - g(x_k)| \leq c^k \cdot |g(x_1) - g(x_0)|.$$

   Notice that $g(x)$ is bounded on $[a, b]$, so $|g(x_1) - g(x_0)|$ is finite, and with $0 \leq c < 1$,

   $$\lim_{k \to \infty} |g(x_{k+1}) - g(x_k)| \leq |g(x_1) - g(x_0)| \cdot \lim_{k \to \infty} c^k = 0.$$

   Furthermore, the iteration relation also implies

   $$\lim_{k \to \infty} (x_{k+2} - x_{k+1}) = 0.$$

   Together with the boundedness of x_n and g, we conclude that the sequence x_n is convergent to some $x^* \in [a, b]$. □

2. If $|g'(x)| > 1, \forall x \in [a, b]$, then the fixed-point iteration will never converge to x^*.

Proof. Since $g'(x)$ is continuous on the interval $[a, b]$,

$$|g'(x)| > 1 \implies (\forall x \in [a, b], g'(x) > 1) \quad \text{or} \quad (\forall x \in [a, b], g'(x) < -1)$$

So we discuss over these two cases.

- $g'(x) > 1$: define $h(x) = g(x) - x$, and it is monotonically increasing since

$$h'(x) = g'(x) - 1 > 0.$$

  Thus, $h'(x) = 0$ has at most one solution, and so is $g(x) = x$.

- $g'(x) < -1$: define $h(x) = g(x) + x$, and it is monotonically decreasing since

$$h'(x) = g'(x) + 1 < 0.$$

  Thus, $h'(x) = 0$ has at most one solution, and so is $g(x) = x$.

In both cases, if the solution x^* exists, then it is unique. Define the Lipschitz constant to be

$$c = \inf_{\xi \in (a,b)} |g'(\xi)|.$$

It comes from definition that $c > 1$. Then apply Cauchy's Mean Value Theorem,

$$|g(x_{k+1}) - g(x_k)| = |g'(\xi)| \cdot |x_{k+1} - x_k| \geq c \cdot |x_{k+1} - x_k|.$$

Apply the iteration relation,

$$x_{k+1} = g(x_k), \qquad x_k = g(x_{x-1}),$$

we get

$$|g(x_{k+1}) - g(x_k)| \geq c \cdot |g(x_k) - g(x_{k-1})|.$$

Such a recursive relation deduces

$$|g(x_{k+1}) - g(x_k)| \geq c^k \cdot |g(x_1) - g(x_0)| = c^{k+1}|x_1 - x_0|.$$

Since x^* is unique, if $x_0 \neq x^*$, then $x_1 = g(x_0) \neq x_0$. Then, with $c > 1$,

$$\lim_{k\to\infty} |g(x_{k+1}) - g(x_k)| \geq |x_1 - x_0| \lim_{k\to\infty} c^{k+1} = +\infty$$

which implies that the iteration does not converge. □

Exercise 7.3 — Root finding

Choose a suitable numerical method to find the smallest positive root and the second smallest positive root of the equation

$$\tan x = 4x$$

correct to 3 decimal places. Explain your choice.

Answer. Newton's method converges quadratically if the function has nonzero derivative at the root, which is faster than bisection method. Even if it has zero derivative at the root, Newton's method still converges linearly, which is not worse than bisection method. So we choose Newton's method. We first define the function in MATLAB™ style.

```
f = @(x) tan(x)-4*x;
df = @(x) 1/(cos(x)^2)-4;
```

The initial guess for the first positive root is near $\frac{\pi}{2}$, while the second is near $\frac{3\pi}{2}$. Using the MATLAB™ function provided, we perform the iteration with the following command:

```
newton(f,df,1.57,1e-4,1e-3,100)
newton(f,df,4.7,1e-4,1e-3,100)
```

The results are recorded in Table 7.1.

(a) Finding the first positive root

n	x_n	E_n	Relative E_n
1	1.5692	7.9235e-04	0.0001
2	1.5676	0.0016	0.0020
3	1.5645	0.0031	0.0040
4	1.5585	0.0060	0.0077
5	1.5472	0.0113	0.0147
6	1.5270	0.0202	0.0265
7	1.4947	0.0323	0.0432
8	1.4525	0.0422	0.0581
9	1.4141	0.0384	0.0543
10	1.3959	0.0181	0.0260
11	1.3933	0.0026	0.0038
12	1.3932	4.5560e-05	6.5397e-05

(b) Finding the second positive root

n	x_n	E_n	Relative E_n
1	4.6905	0.0095	0.0041
2	4.6776	0.0129	0.0055
3	4.6654	0.0122	0.0052
4	4.6596	0.0058	0.0025
5	4.6588	8.1071e-04	3.4803e-04
6	4.6588	1.2777e-05	5.4849e-06

Table 7.1: Newton's Method on Iteration

□

Exercise 7.4 — Order and rate of convergence

Suppose that the sequence $\{a_n\}$ converges to the number L, and there is a constant $0 < \lambda < \infty$ such that

$$\lim_{n\to\infty} \frac{|a_{n+1} - L|}{|a_n - L|^\alpha} = \lambda,$$

then the sequence is said to converge to L with *order of convergence* α. The constant λ is called the *asymptotic error*. A positive sequence $\{E_n\}$ is said to has an order of at least α and a *rate* of at most λ if there is a sequence $\{a_n\}$ that has an order α and a rate of λ such that

$$E_n \leq a_n \quad \forall n \in \mathbb{N}^*.$$

1. Find the order of convergence and the rate of convergence of the sequence

$$a_n = \frac{1}{n} \quad \forall n \geq 1$$

Answer. First, we evaluate the sequence limit.

$$L = \lim_{n\to\infty} \frac{1}{n} = 0.$$

Then,

$$\lim_{n\to\infty} \frac{|a_{n+1}|}{|a_n|} = 1$$

So the order of the convergence is 1, and the rate of the convergence is also 1. □

2. Examine the order of convergence and the rate of convergence of $\{b_n\}$ where

$$b_{2k} = \frac{1}{\ln k} \quad \text{and} \quad b_{2k+1} = \frac{1}{k} \quad \forall k \geq 1.$$

Answer. First, we evaluate the sequence limit.

$$L = \lim_{n\to\infty} \max\left(\frac{1}{\ln(n/2)}, \frac{2}{n}\right) = 0.$$

Notice that

$$\forall \alpha \in \mathbb{R}, \qquad \lim_{n\to\infty} \frac{\ln^\alpha n}{n} = 0 \quad \text{and} \quad \lim_{n\to\infty} \frac{n}{\ln^\alpha n} = +\infty.$$

Then,

$$\lim_{2k\to\infty} \frac{|b_{2k+1}|}{|b_{2k}|} = \lim_{k\to\infty} \frac{\ln k}{k} = 0$$
$$\lim_{2k\to\infty} \frac{|b_{2k+2}|}{|b_{2k+1}|} = \lim_{k\to\infty} \frac{k+1}{\ln k} = +\infty$$

Since the two subsequence of b_n have two different limits, the limit of b_n does not exist. So neither the order of the convergence nor the rate of the convergence exists. □

3. Use the precise definition above to show the method of fixed-point iteration leads an error sequence that has at least linear convergence.

Proof. Suppose the fixed point for function g is

$$g(x^*) = x^*.$$

The iteration gives

$$\forall n \in \mathbb{N}^*, \quad x_{n+1} = g(x_n).$$

If the iteration converges, according to 7.2, we have

$$\lim_{n\to\infty} x_n = x^*.$$

The linear convergence comes directly from the definition of Lipschitz continuity. Lipschitz continuity ensures that

$$|g(x_n) - g(x^*)| \leq c \cdot |x_n - x^*|,$$

where $0 < c < 1$ is the Lipschitz constant. Apply substitutions, we obtain

$$|x_{n+1} - x^*| \leq c \cdot |x_n - x^*|$$

Apply the definition of error term,

$$E_{n+1} \leq c \cdot E_n.$$

Since this is true for all $n \in \mathbb{N}^*$, we have the limit

$$\lim_{n\to\infty} \frac{E_{n+1}}{E_n} \leq c.$$

So the linear convergence is proved. □

4. Find the asymptotic error constant for the method of fixed-point iteration when it has at least quadratic convergence.

Proof. Since $g : \mathbb{R} \to \mathbb{R}$ is required to be Lipschitz continuous, it is absolutely continuous, and therefore is almost everywhere differentiable. Apply the Taylor's expansion at x^*,

$$g(x_n) = g(x^* + (x_n - x^*)) = g(x^*) + g'(x^*)(x_n - x^*) + \frac{1}{2}g''(x^*)(x_n - x^*)^2 + o\left((x_n - x^*)^2\right)$$

If the convergence is at least quadratic, the first derivative $g'(x^*) = 0$, so the Taylor expansion is reduced to

$$x_{n+1} - x^* = \frac{1}{2}g''(x^*)(x_n - x^*)^2 + o\left((x_n - x^*)^2\right)$$

Observe that $x_n - x^*$ is the error term E_n, and by substitution,

$$\lim_{n\to\infty} \frac{E_{n+1}}{E_n} = g'(x^*).$$

So the convergence rate is

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \frac{1}{2}|g'(x^*)|.$$

□

5. Show the order of convergence of the secant method is $\frac{\sqrt{5}+1}{2}$.

Proof. Apply the definition of secant method.

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

Rewrite it in the form of error term, *i.e.*, apply the substitution $x_n = x_* + E_n$, we get

$$E_{n+1} = E_n - \frac{f(x^* + E_n)(E_n - E_{n-1})}{f(x^* + E_n) - f(x^* + E_{n-1})}. \tag{7.1}$$

Assume $f \in \mathcal{C}^2$, we apply Taylor's expansion up to the quadratic term,

$$f(x^* + E) \approx f(x^*) + f'(x^*)E + \frac{f''(x^*)}{2} \cdot E^2.$$

The subscript is ignored since Taylor's expansion applies to all $E_n, n \in \mathbb{N}^*$ with the assumption that E_n are small. Since $f(x^*) = 0$ by definition of root,

$$f(x^* + E_n) \approx f'(x^*)E_n \left(1 + \frac{f''(x^*)}{2f'(x^*)} \cdot E_n\right).$$

So the denominator in (7.1) can be approximated by

$$f(x^* + E_n) - f(x^* + E_{n-1}) \approx f'(x^*)(E_n - E_{n-1}) \left[1 + \frac{f''(x^*)}{2f'(x^*)} \cdot (E_n + E_{n-1})\right].$$

For simplicity, define

$$M = \frac{f''(x^*)}{2f'(x^*)}.$$

Then equation (7.1) can be simplified to

$$\begin{aligned} E_{n+1} &\approx E_n - \frac{E_n f'(x^*)(1 + ME_n)(E_n - E_{n-1})}{f'(x^*)(E_n - E_{n-1})[1 + M(E_n + E_{n-1})]} \\ &= E_n - \frac{E_n(1 + ME_n)}{1 + M(E_n + E_{n-1})} \\ &= \frac{M}{1 + M(E_n + E_{n-1})} E_n E_{n-1} \end{aligned}$$

Again, apply the assumption that E_n are small, then

$$\frac{1}{1 + M(E_n + E_{n-1})} \approx 1,$$

and thus

$$E_{n+1} = ME_nE_{n-1}. \tag{7.2}$$

By definition of convergence rate,

$$\lim_{n\to\infty}\frac{|E_{n+1}|}{|E_n|^\alpha}=\lambda.$$

Then, we have the approximated first order recursive relation

$$\forall n>N,\quad |E_{n+1}|\approx\lambda|E_n|^\alpha.$$

Apply this approximation to the second order recursive relation (7.2),

$$\lambda|E_n|^\alpha=|M|\cdot|E_n|\cdot|E_{n-1}|,$$

which can be written in another first order recursive relation

$$|E_n|^{\alpha-1}=\frac{|M|}{\lambda}\cdot|E_{n-1}|.$$

Taking the $(\alpha-1)^{\text{th}}$ root, we have the recursive relation in the original form

$$|E_n|=\left(\frac{|M|}{\lambda}\right)^{\frac{1}{\alpha-1}}\cdot|E_{n-1}|^{\frac{1}{\alpha-1}}$$

Since we only focus on the order of convergence, not the rate, we omit the equation for the coefficient. Then, we require

$$\alpha=\frac{1}{\alpha-1},$$

which has a root

$$\alpha=\frac{\sqrt{5}+1}{2}.$$

The negative root is omitted since it does not satisfy the convergence condition. □

Exercise 7.5 — Order of roots

Suppose that $f(x)$ and its derivatives $f'(x),\cdots,f^{(m)}(x)$ are defined and continuous on an interval containing x^*, we say that $f(x)=0$ has a *root of order* m at $x=x^*$ if and only if

$$f(x^*)=0,f'(x^*)=0,f''(x^*)=0,\cdots,f^{(m-1)}(x^*)=0,\text{ and }f^{(m)}(x^*)\neq0.$$

The positive integer m is known as the *multiplicity of the root*. A root of order $m=1$ is often called a *simple root*, and if $m>1$, it is called a *multiple root*.

1. Prove by induction that there exists a continuous function $h(x)$ so that $f(x)$ can be expressed as the product

$$f(x)=(x-x^*)^m h(x),\quad\text{where}\quad h(x^*)\neq0$$

if the equation $f(x)$ has a root of order m at $x=x^*$.

Proof. Actually, induction is not required to prove the statement. We can define

$$h(x) = \frac{f(x)}{(x-x^*)^m}, \quad \forall x \in \mathbb{R}\backslash\{x^*\}.$$

This function is obviously continuous at $\mathbb{R}\backslash\{x^*\}$. So we need the continuity at $x = x^*$. Apply the L'Hospital rule, h has a limit at $x = x^*$

$$\lim_{x\to x^*} \frac{f(x)}{(x-x^*)^m} = \lim_{x\to x^*} \frac{f^{(m)}(x)}{[(x-x^*)^m]^{(m)}} = \frac{f^{(m)}(x)}{m!} \neq 0$$

by definition. So we complete the definition of h:

$$h(x) = \begin{cases} \frac{f(x)}{(x-x^*)^m}, & x \in \mathbb{R}\backslash\{x^*\} \\ \frac{f^{(m)}(x^*)}{m!}, & x = x^* \end{cases}.$$

This is continuous at x^*, since the limit of the function equals to the function value. It is easy to verify

$$f(x) = (x-x^*)^m h(x).$$

And $h(x^*) = f^{(m)}(x^*) \neq 0$ is also satisfied. So we are able to find such an $h(x)$ for all $m \in \mathbb{N}^*$. □

2. Show the following modified Newton's iteration will produce a sequence that converges quadratically to x^*

$$x_k = x_{k-1} - m \cdot \frac{f(x_{k-1})}{f'(x_{k-1})}$$

if the original Newton's method produces a sequence converges linearly to the root $x = x^*$ of order $m > 1$.

Proof. Write $f(x) = (x-x^*)^m h(x)$, then

$$x_k = x_{k-1} - m \cdot \frac{f(x_{k-1})}{f'(x_{k-1})} = x_{k-1} - \frac{(x_{k-1}-x^*)h(x_{k-1})}{h(x_{k-1}) + \frac{1}{m}(x_{k-1}-x^*)h'(x_{k-1})}.$$

Define the error term $E_k = x_k - x^*$, then

$$E_k = E_{k-1}\left(1 - \frac{1}{1 + \frac{1}{m}E_{k-1}\frac{h'(x^*+E_{k-1})}{h(x^*+E_{k-1})}}\right).$$

A first order approximation gives

$$E_k \approx E_{k-1}^2 \cdot \frac{h'(x^*+E_{k-1})}{m \cdot h(x^*+E_{k-1})}.$$

So the quadratic convergence is proved. □

3. In practice, it is unlikely that m is known, in those cases the following version of modified Newton's method can be applied to accelerate convergence

$$x_k = x_{k-1} - \frac{u(x_{k-1})}{u'(x_{k-1})}, \quad \text{where} \quad u(x) = \frac{f(x)}{f'(x)}$$

Explain how this helps to speed up convergence.

Answer. If a polynomial $f(x)$ has a root of multiplicity m at a point x_0, then its derivative $f'(x)$ has a root of multiplicity $m-1$ at x_0. Thus we can obtain a function

$$u(x) = \frac{f(x)}{f'(x)}$$

that has all simple roots at all of f's roots. □

4. Zero is a root of multiplicity of 3 for the function

$$f(x) = \sin(x^3).$$

Start with $x_0 = 1$ and compute $x_1, x_2, \cdots, x_6$ first by using the iteration formula in part (b) and then by using the iteration formula in part (c). What can you conclude based on the values that you have computed?

Answer. Using the MATLAB™ function provided, we perform the iteration with the following command:

```
f = @(x) sin(x^3);
df = @(x) 3*x^2*cos(x^3);
newton_Modified(f,df,1,3,1e-10,1e-10,100)
u = @(x) tan(x^3)/(3*x^2);
du = @(x) sec(x^3)^2 - (2*tan(x^3))/(3*x^3);
newton_Improved(f,u,du,1,1e-20,1e-20,6)
```

The results are recorded in Table 7.2. It's easy to observe that both methods converge

(a) Modified with $m = 3$

n	x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
1	-0.557407724654902	1.557407724654902
2	0.005640692477577	0.563048417132479
3	-6.071532165918825e-17	0.005640692477577
4	0	6.071532165918825e-17
5	NaN	NaN
6	NaN	NaN

(b) Improved

x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
0.782537832379215	0.217462167620785
0.265581322231385	0.516956510147830
1.862855151222864e-04	0.265395036716262
2.710505431213761e-20	1.862855151222864e-04
3.009265538105056e-36	2.710505431213761e-20
3.340955887615245e-52	3.009265538105056e-36

Table 7.2: Comparison of Newton's Method on Iteration

rapidly, and the method in part (b) happens to reach the exact solution and terminates. □

5. Compare Newton's method and the modified version of it in part (c) by finding all the roots of the following polynomial correct to 6 decimal places.

$$f(x) = x^5 - 11x^4 + 46x^3 - 90x^2 + 81x - 27$$

Proof. Using the MATLAB™ function provided, we perform the iteration with the following command:

```
f = @(x) x^5-11*x^4+46*x^3-90*x^2+81*x-27;
df = @(x) 5*x^4-44*x^3+138*x^2-180*x+81;
u = @(x) f(x)/df(x);
du = @(x) (5*x^2-18*x+21)/(5*x-9)^2;
newton_Modified(f,df,1.2,1,1e-11,1e-11,100)
newton_Improved(f,u,du,1.2,1e-20,1e-20,6)
newton_Modified(f,df,3.01,1,1e-20,1e-80,100)
newton_Improved(f,u,du,3.02,1e-80,1e-40,40)
```

The results are recorded in Table 7.3 and 7.4. Obviously, the improved method which only have simple roots converges much faster. □

(a) Original Newton's iteration

n	x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
1	1.079999999999995	0.120000000000005
2	1.037333333333312	0.042666666666683
3	1.018118414918420	0.019214918414892
4	1.008933250933831	0.009185163984589
5	1.004436361406744	0.004496889527087
6	1.002210759058684	0.002225602348059
7	1.001103541654488	0.001107217404196
8	1.000551313519913	5.522281345746727e-04
9	1.000275542702326	2.757708175875617e-04
10	1.000137742869052	1.377998332738883e-04
11	1.000068864318036	6.887855101522789e-05
12	1.000034430381744	3.443393629209979e-05
13	1.000017214700528	1.721568121615391e-05
14	1.000008607270650	8.607429878404460e-06
15	1.000004303573453	4.303697196395007e-06
16	1.000002151611564	2.151961889129694e-06
17	1.000001075651434	1.075960130103582e-06

(b) Improved

x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
1.036363636363630	0.163636363636370
1.001028277634942	0.035335358728688
1.000000793830820	0.001027483804122
1.000000000564978	7.932658419029792e-07

Table 7.3: Comparison of Newton's Method on Iteration at a Root of Order 2

Exercise 7.6 — Problem with root finding methods

Provide an example, if exists, of root-finding problems that satisfy the following criteria:

(a) Original Newton's iteration

n	x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
1	3.006677685778723	0.003322314221277
2	3.004456717701055	0.002220968077668
3	3.002973344359819	0.001483373341236
4	3.001983210661162	9.901336986564147e-04
5	3.001322573018363	6.606376427993332e-04
6	3.000881914953126	4.406580652367431e-04
7	3.000588039669541	2.938752835852654e-04
8	3.000392066202243	1.959734672976055e-04
9	3.000261381083893	1.306851183500157e-04
10	3.000174327057305	8.705402658826245e-05
11	3.000116433888668	5.789316863724636e-05
12	3.000076956251177	3.947763749057032e-05
13	3.000053363008282	2.359324289535891e-05
14	3.000035897764118	1.746524416423867e-05
15	3.000028546336471	7.351427647073194e-06
16	3.000002389324648	2.615701182318020e-05
17	3.000831576720999	8.291873963517382e-04
18	3.000554470371992	2.771063490074610e-04
19	3.000369712689852	1.847576821396579e-04
20	3.000246556354566	1.231563352859233e-04
21	3.000164218617164	8.233773740240480e-05
22	3.000109602674366	5.461594279765336e-05
23	3.000073526984020	3.607569034658198e-05
24	3.000046805273550	2.672171046969041e-05
25	3.000033832581067	1.297269248246025e-05
26	3.000031763455728	2.069125339509981e-06
27	3.000027068531880	4.694923848358457e-06
28	3.000010906132086	1.616239979318479e-05

(b) Improved

x_n	$E_n = \lvert x_n - x_{n-1}\rvert$
2.999869302453229	0.020130697546771
2.999997846418646	1.285439654177267e-04
2.983092132985680	0.016905713432966
2.999903080879523	0.016810947893843
2.999999154222051	9.607334252725863e-05
2.921051830315649	0.078947323906402
2.997750655427219	0.076698825111570
2.999998308704638	0.002247653277418
2.992547985979292	0.007450322725346
2.999981350990154	0.007433365010863
3.000103925190753	1.225742005983577e-04
3.000003933963204	9.999122754900114e-05

Table 7.4: Comparison of Newton's Method on Iteration at a Root of Order 3

1. It can be solved by Bisection but not by using fixed-point iteration.

   Answer. By previous result, we only need the function to have Lipschitz constant $c > 1$. A typical example is
   $$f(x) = 2|x|.$$
   □

2. It can be solved by using fixed-point iteration but not by Newton's method.

   Answer. Fixed-point iteration requires Lipschitz continuity, while Newton's method requires differentiable. So we can construct a piecewise Lipschitz continuous function
   $$f(x) = \begin{cases} \frac{|x|}{2} & |x| \le \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} \le |x| \le 1 \\ \frac{|x|}{2} - \frac{1}{4} & |x| \ge 1 \end{cases}$$
   Such function has Lipschitz constant $c = \frac{1}{2}$ on $\mathbb{R}$, so it will be solved by fixed-point iteration at any initial guess. However, if the initial guess x_0 has $|x_0| > 1$, then
   $$x_1 = x_0 - \frac{\frac{|x_0|}{2} - \frac{1}{4}}{\operatorname{sgn}(x_0) \cdot \frac{1}{2}} = \frac{1}{2} \cdot \operatorname{sgn}(x_0)$$
   But the function is not differentiable at $\pm\frac{1}{2}$, so the process cannot continue. If the initial guess has $\frac{1}{2} < |x_0| < 1$, then $f'(x_0) = 0$, and the process cannot continue either. At critical points $|x_0| = 1$ or $|x_0| = \frac{1}{2}$, the function is not differentiable. □

Exercise 7.7 — Complex root finding for analytic functions

Let $f : \mathbb{C} \to \mathbb{C}$ be analytic. The following iteration formula can be used to solve $f(z) = 0$.
$$z_{k+1} = z_k - \frac{f(z_k)}{f'(z_k) - \frac{f''(z_k)}{2f'(x_k)} \cdot f(z_k)}$$
where $f'(z)$ and $f''(z)$ are the usual complex derivatives of $f(z)$.

1. For the following function
   $$f(z) = z^7 - 1$$
   depict the relationship between the initial point $z_0 = x_0 + iy_0$ and the number of iteration required for the sequence to be considered to have converged, *i.e.*,
   $$|z_k - z_{k-1}|^2 < \varepsilon \quad \text{where } \varepsilon = 0.0001.$$

2. What happens to the relationship in part (a) if we use the criterion
   $$\left||z_k|^2 - |z_{k-1}|^2\right| < \varepsilon \quad \text{where } \varepsilon = 0.0001.$$

Assignment 8

Optimization

Exercise 8.1 — Convexity and Quasi-convexity

The negative of a *convex* function is known to be a *concave* function. The negative of a *quasi-convex* function is known to be a *quasi-concave* function. For each of the following functions determine whether it is convex, concave, quasi-convex, or quasi-concave.

1. The function $f : \mathbb{R} \to \mathbb{R}$ given by

$$f(x) = e^x - 1.$$

Answer. The second derivative

$$f''(x) = e^x > 0, \quad \forall x \in \mathbb{R},$$

so it is convex, and it comes from definition that it is also quasi-convex. Thus, it is not concave, and not quasi-concave. □

2. The function $f : \mathbb{R}^2_{++} \to \mathbb{R}$ given by

$$f(x_1, x_2) = x_1 x_2.$$

Answer. The Hessian matrix of f is

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

where the eigenvalues are $\lambda_1 = 1, \lambda_2 = -1$. So the function is neither convex nor concave, nor is it quasi-convex or quasi-concave. □

3. The function $f : \mathbb{R}^2_{++} \to \mathbb{R}$ given by

$$f(x_1, x_2) = \frac{1}{x_1 x_2}.$$

Answer. The Hessian matrix of f is

$$\begin{pmatrix} \frac{2}{x_1^3 x_2} & \frac{1}{x_1^2 x_2^2} \\ \frac{1}{x_1^2 x_2^2} & \frac{2}{x_1 x_2^3} \end{pmatrix},$$

which is positive definite for $x_1, x_2 \in \mathbb{R}^+$. Thus, f is convex on its domain, and it follows from definition that it is also quasi-convex. Thus, it is not concave, and not quasi-concave. □

4. The function $f : \mathbb{R}^2_{++} \to \mathbb{R}$ given by

$$f(x_1, x_2) = \frac{x_1}{x_2}.$$

Proof. The Hessian matrix of f is

$$\begin{pmatrix} 0 & -\frac{1}{x_2^2} \\ -\frac{1}{x_2^2} & \frac{2x_1}{x_2^3} \end{pmatrix}.$$

And the eigenvalue of it is

$$\lambda_{1,2} = \frac{x_1 \pm \sqrt{x_1^2 + x_2^2}}{x_2^3}.$$

So $\lambda_1 > 0, \lambda_2 < 0, \forall x_1, x_2 \in \mathbb{R}^+$, which means the matrix is always indefinite. So f is neither convex nor concave, nor is it quasi-convex or quasi-concave. □

5. The function $f : \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}$ given by

$$f(x_1, x_2) = \frac{x_1^2}{x_2}.$$

Answer. The Hessian matrix of f is

$$\begin{pmatrix} \frac{2}{x_2} & -\frac{2x_1}{x_2^2} \\ -\frac{2x_1}{x_2^2} & \frac{2x_1^2}{x_2^3} \end{pmatrix},$$

which is negative semidefinite. So f is neither convex nor concave, nor is it quasi-convex or quasi-concave. □

6. The function $f : \mathbb{R}^2_{++} \to \mathbb{R}$ given by

$$f(x_1, x_2) = x_1^\alpha x_2^{1-\alpha}.$$

$$f(x_1, x_2) = x_1 x_2.$$

Answer. The Hessian matrix of f is

$$\begin{pmatrix} \alpha(\alpha-1)x_1^{\alpha-2}x_2^{1-\alpha} & \alpha(1-\alpha)x_1^{\alpha-1}x_2^{-\alpha} \\ \alpha(1-\alpha)x_1^{\alpha-1}x_2^{-\alpha} & \alpha(\alpha-1)x_1^{\alpha}x_2^{-\alpha-1} \end{pmatrix},$$

where the eigenvalues are $\lambda_1 = \alpha(\alpha-1)x_1^\alpha x_2^{-\alpha}\left(x_1^{-2}x_2 + x_2^{-1}\right), \lambda_2 = 0$, which makes the matrix semidefinite. So f is neither convex nor concave, nor is it quasi-convex or quasi-concave. □

Exercise 8.2 — Extrema and Convexity in higher dimension

Let $f : \mathbb{R}^n \to \mathbb{R}$. Prove the following theorems.

1. If f is continuously differentiable in an open neighborhood of a local minimizer x^*, then the gradient $\nabla f(x^*) = 0$.
2. If the Hessian H of f is continuous in an open neighborhood of a local minimizer x^*, then the gradient $\nabla f(x^*) = 0$ and the Hessian H at $x = x^*$ is *positive semi-definite*.
3. If f is differentiable and convex, then x^* is a global minimum if and only if $\nabla f(x^*) = 0$.
4. If f is twice differentiable, then f is convex if and only if f is *positive semi-definite* for all $x \in \mathbb{R}$.

Exercise 8.3 — Approximation width

Let $f_0, f_1, \cdots, f_n : \mathbb{R} \to \mathbb{R}$ be continuous functions. Consider the problem of approximating f_0 as a linear combination of $f_1, \cdots, f_n$. For $\alpha \in \mathbb{R}^n$, we say that

$$f = \alpha_1 f_1 + \cdots + \alpha_n f_n$$

approximates f_0 with tolerance $\epsilon > 0$ over the interval $[0, T]$ if

$$|f(t) - f_0(t)| \leq \epsilon \quad \forall t \in [0, T].$$

For a fixed tolerance $\epsilon > 0$, we define the *approximation width* as the largest T such that f approximates f_0 over the interval $[0, T]$, that is

$$W(\alpha) = \sup \{T | \forall 0 \leq t \leq T, \quad |f(t) - f_0(t)| \leq \epsilon\}$$

Show that W is quasi-concave.

Exercise 8.4 — Numerical methods to find global extrema

Choose a suitable numerical method to find the global minimum of the function

$$f(x) = \frac{\sin \frac{1}{x}}{(x - 2\pi)^2 + \pi}$$

correct to 3 decimal places. Explain your choice.

Answer. Since f is a combination of elementary function, $f \in \mathcal{C}^\infty(\mathbb{R}\backslash\{0\})$. So we try Newton's method to find the root of f', since the global has the property

$$f'(x) = 0, \quad f''(x) > 0.$$

Since there are infinitely many local minimum and local maximum in the interval $[-1, 1]$, we first omit the second derivative. After finding a root of the first derivative, we check if

the function value is negative or positive. If positive, it must be a maximum, and we will try again. The numerator has the property $\sin\frac{1}{x} < 0, \forall x \in \left(\frac{1}{2\pi}, \frac{1}{\pi}\right)$. And the denominator is monotonically decreasing in this interval. So there is a local minimum $x^* \in \left(\frac{1}{2\pi}, \frac{1}{\pi}\right)$. Furthermore, all the roots of $\sin\frac{1}{x}$ are in the interval $\left[-\frac{1}{\pi}, \frac{1}{\pi}\right]$, while x^* also minimizes the positive denominator. Thus we deduce that the local minimum $x^* \in \left(\frac{1}{2\pi}, \frac{1}{\pi}\right)$ is the global minimum. Then, we evaluate the first derivative of f:

$$f'(x) = -\frac{\cos\left(\frac{1}{x}\right)}{x^2\left((x-2\pi)^2+\pi\right)} - \frac{2(x-2\pi)\sin\left(\frac{1}{x}\right)}{\left((x-2\pi)^2+\pi\right)^2}.$$

The zero derivative gives

$$2(2\pi - x)x^2 \sin\left(\frac{1}{x}\right) + \left(-x^2 + 4\pi x - 4\pi^2 - \pi\right)\cos\left(\frac{1}{x}\right) = 0.$$

Define the left hand side as $g(x)$, and its derivative is

$$g'(x) = -\frac{(6x^4 - 8\pi x^3 + x^2 - 4\pi x + 4\pi^2 + \pi)\sin\left(\frac{1}{x}\right)}{x^2}$$

Apply Newton's root-finding iteration with initial guess $x_0 = \frac{1}{1.5\pi}$, we get

$$x^* \approx 0.213.$$

Indeed, $\sin\frac{1}{x^*} \approx -0.9998$, which provides a minimum for sure. By previous deduction, we are sure it is the global minimum, and $\min_x f(x) \approx -0.025$. □

Exercise 8.5 — Cubic approximation

Suppose f is unimodal and differentiable. Consider the following algorithm which explicitly uses f_0 when we know the minimum point $x \in [a_0, b_0]$. The essential idea is to approximate the function f on the interval $[a_{k-1}, b_{k-1}]$ with a cubic polynomial of the form

$$P(x) = \alpha_{k-1}(x - a_{k-1})^3 + \beta_{k-1}(x - a_{k-1})^2 + \gamma_{k-1}(x - a_{k-1}) + \rho_{k-1},$$

which has the same value and derivative as f at the endpoints a_{k-1} and b_{k-1}. Then using the minimum point c_{k-1} of the cubic polynomial to tell how to squeeze the interval $[a_{k-1}, b_{k-1}]$ to $[a_k, b_k]$.

1. Discuss why if $f(c_{k-1}) > 0$, then we shall set $a_k = a_{k-1}$ and $b_k = c_{k-1}$, else we shall set $a_k = c_{k-1}$ and $b_k = b_{k-1}$.

2. Show that
$$c_{k-1} = a_{k-1} + \frac{-\beta_{k-1} + \sqrt{\beta_{k-1}^2 - 3\alpha_{k-1}\gamma_{k-1}}}{3\alpha_{k-1}}.$$

3. Find formulas for $\alpha_k, \beta_k, \gamma_k$ and ρ_k in terms of $\alpha_{k-1}, \beta_{k-1}, \gamma_{k-1}$ and ρ_{k-1}.

4. Use this method to find the minimum of

$$f(x) = e^x + 2x + \frac{x^2}{2}$$

on the interval $[-2.4, -1.6]$ correct to 5 decimal places.

Answer. $c^* \approx -2.12003$. □

5. Compare this method with Newton's method in terms of assumptions about f and convergence rate.

6. Compare this method with Golden Section Search in terms assumptions about f and convergence rate.

Exercise 8.6 — Constrained Extrema

Consider the following function

$$f(x, y) = e^x \left(4x^2 + 2y^2 + 4xy + 2y + 1\right).$$

Produce an informative MATLAB™ plot of $g(y) = \min_x f(x, y)$, which gives the minimum of $f(x, y)$ as y changes.

Answer. Since f is a combination of elementary function, $f \in \mathcal{C}^\infty(\mathbb{R})$. Thus, a necessary and sufficient condition for the minimum of f when x changes is

$$\partial_x f = 0, \quad \partial_{xx} f > 0.$$

So we evaluate the first and second derivative of f:

$$\begin{aligned} \partial_x f &= \left(4x^2 + 4xy + 8x + 2y^2 + 6y + 1\right) e^x, \\ \partial_{xx} f &= \left(4x^2 + 4xy + 16x + 2y^2 + 10y + 9\right) e^x. \end{aligned}$$

Since $e^x \neq 0$, we only need

$$\begin{aligned} 4x^2 + 4xy + 8x + 2y^2 + 6y + 1 &= 0, \\ 4x^2 + 4xy + 16x + 2y^2 + 10y + 9 &> 0. \end{aligned}$$

Plug in the first equality to the second inequality, we get

$$\begin{aligned} 4x^2 + 4xy + 8x + 2y^2 + 6y + 1 &= 0, \\ 2x + y + 2 &> 0. \end{aligned}$$

Regard the first equality as a quadratic equation of x while treat y as a parameter, we can find the two roots:

$$x = \frac{-(y+2) \pm \sqrt{(1-y)(y+3)}}{2}.$$

A simple transformation gives

$$2x + y + 2 = \pm\sqrt{(1-y)(y+3)},$$

so the inequality requires us to take the positive root. f can also be simplified to

$$f(x, y) = -4(2x + y)e^x.$$

Apply the substitution, we get the explicit form of g:

$$\begin{aligned} g(y) &= f\left(\frac{-(y+2) + \sqrt{(1-y)(y+3)}}{2}, y\right) \\ &= 4e^{\frac{1}{2}\left(\sqrt{(1-y)(y+3)} - y - 2\right)}\left(2 - \sqrt{(1-y)(y+3)}\right), \end{aligned}$$

and the natural domain is $y \in [-3, 1]$. The plot of g is given in Figure 8.1. □

Exercise 8.7 — Application

The length, L, of the longest ladder that can pass around the corner of two corridors depends on the angle α shown in Figure 8.2. Produce a MATLAB™ plot of L versus α ranging from 45° to 135° by first solving a minimisation problem using numerical methods that we have discussed so far.

Answer. Observe that

$$L(\alpha, \beta) = \frac{1}{\sin\beta} + \frac{1}{\sin(\alpha + \beta)}.$$

Regard α as a parameter, and β as a variable, we need to find

$$l(\alpha) = \min_{\beta} L(\alpha, \beta).$$

So we require

$$\partial_\beta L(\alpha, \beta) = 0, \quad \partial_{\beta\beta} L(\alpha, \beta) > 0.$$

Plug in the function, we find that when

$$\beta = \frac{\pi - \alpha}{2},$$

the condition is satisfied. Thus,

$$l(\alpha) = L\left(\alpha, \frac{\pi - \alpha}{2}\right) = \frac{1}{\sin\frac{\pi-\alpha}{2}} + \frac{1}{\sin\frac{\pi+\alpha}{2}}.$$

The graph of this function on the desired domain is plotted in Figure 8.3. □

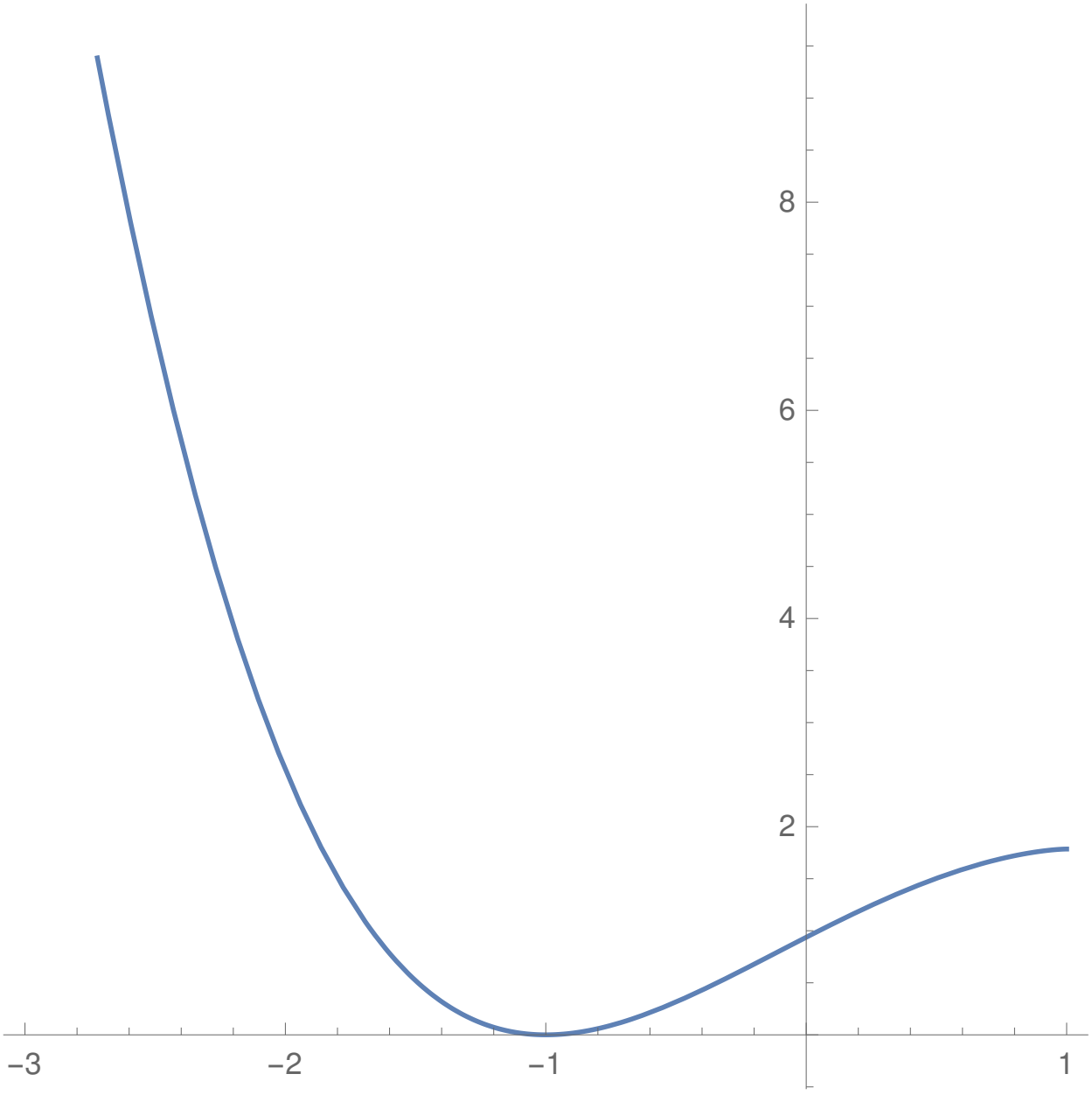

Figure 8.1: Plot of $g(y) = \min_x f(x, y)$

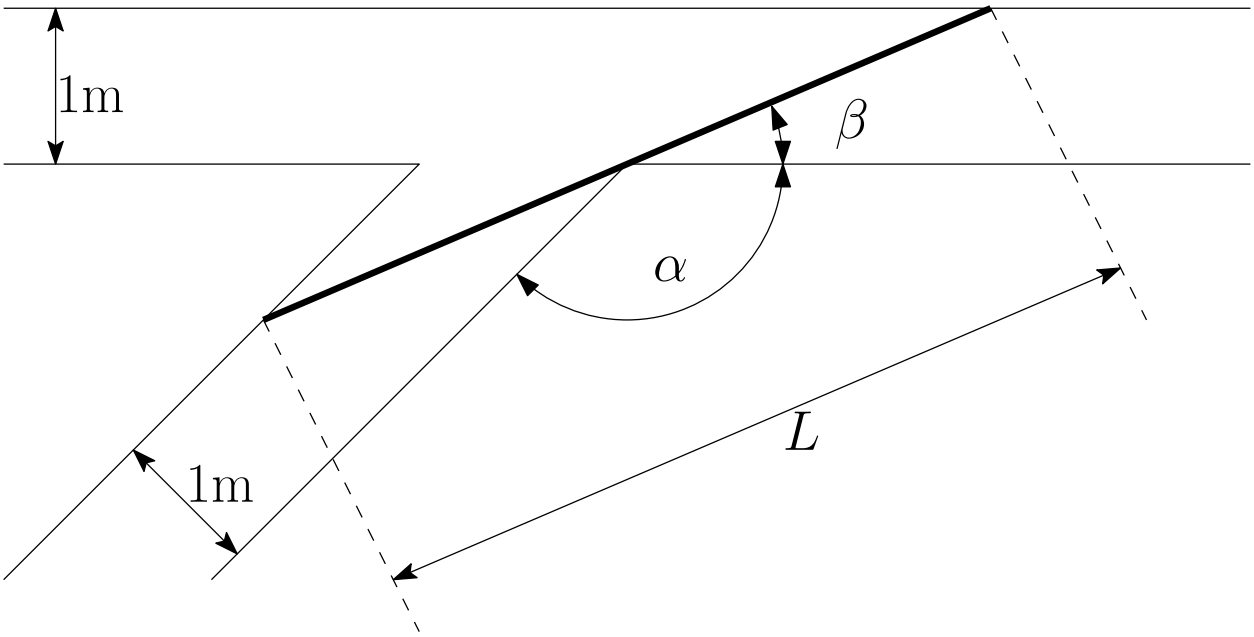

Figure 8.2: Illustration of the situation

Exercise 8.8 — Method of simulated annealing

A popular global optimization algorithm for difficult functions, especially if there are many local minima, is called the *method of simulated annealing*. It involves no derivatives or an initial guess that needs to be sufficiently close to the minimum. Suppose $f : \mathbb{R}^n \to \mathbb{R}$ has a global minimum at x and the k-iteration x_k has been computed. It iterates by the following scheme:

1. Generating a number of random points $u_1, u_2, \cdots, u_m$ in a large neighborhood of x_k.
2. Computing $f(u_1), f(u_2), \cdots, f(u_m)$.
3. Finding the index j such that $f(u_j) = \min\{f(u_1), f(u_2), \cdots, f(u_m)\}$.
4. Assigning $x_{k+1} = u_j$ if $f(x_k) > f(u_j)$. Otherwise assigning a probability

$$p_i = \frac{\exp\{\alpha[f(x_k) - f(u_i)]\}}{\sum_{\ell=1}^{m} \exp\{\alpha[f(x_k) - f(u_\ell)]\}}, \quad \alpha > 0,$$

   to u_i for each $i = 1, \cdots, m$. Then making a random choice among $u_1, u_2, \cdots, u_m$ according to the probabilities p_i and assigning this randomly chosen u_ℓ to be x_{k+1}.

With some minor modifications, this can be used for function $Q : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is any set. For example, in the *traveling salesman problem*, $\mathcal{X}$ is the set of all permutations of a set of integers. Consider the *Euclidean traveling salesman problem* (ETSP): given a set of points in $\mathbb{R}^2$ representing positions of cities on a map, we wish to visit each city exactly once while minimizing the total distance traveled.

1. Implement simulated annealing to solve ETSP with different α.

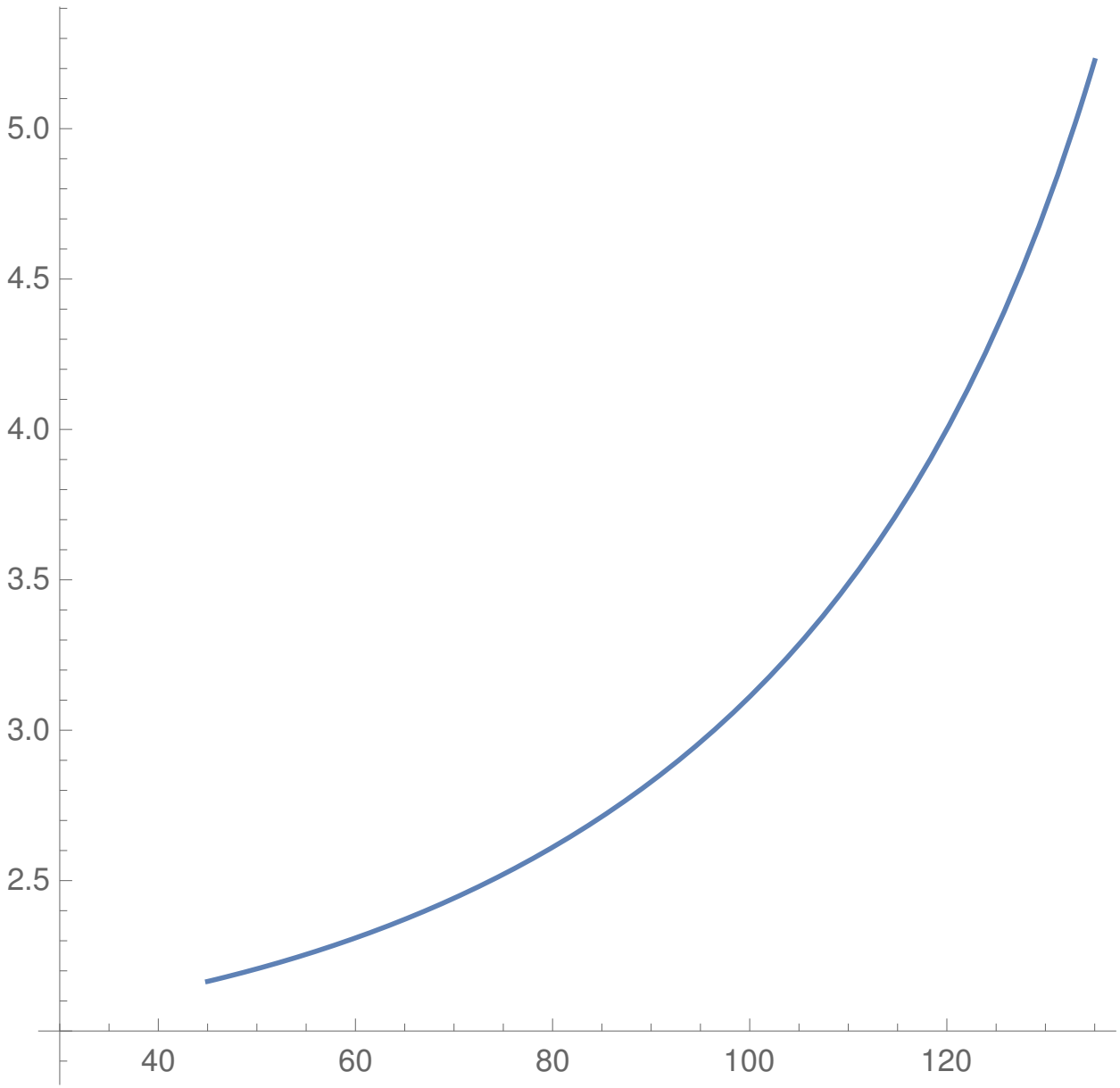

Figure 8.3: Maximum length to pass the corridors. (β in degrees)

2. Propose another optimization method to solve ETSP. Analyze how its efficiency would compare to that of simulated annealing.

Assignment 9

Differential Equations

Exercise 9.1 — Uniqueness of initial value problem

Theorem 9.1 (Uniqueness of initial value problem)**.** *If Φ is continuous and satisfies a Lipschitz condition in y on the set*

$$\mathcal{D} = \big\{(t, y) \;\big|\; t_0 \le t \le T, y \in \mathbb{R}\big\},$$

then

$$\dot{y} = \Phi(t, y), \quad y(t_0) = y_0, \text{ where } t_0 \le t \le T$$

has a unique solution.

1. Show Φ satisfies a Lipschitz condition in y on $\mathcal{A}$ with Lipschitz constant c if $\mathcal{A}$ is convex and there exists a $c > 0$ such that

$$|\partial_y \Phi(t, y)| \le c, \quad \forall (t, y) \in \mathcal{A}$$

2. Show for any constants t_0 and T, the set $\mathcal{D}$ is convex.

3. Use the above to show the following IVP has a unique solution.

$$\dot{y} = \frac{4t^3 y}{1 + t^4}, \quad y(0) = 1, \quad t \in [0, 1].$$

Proof.

$$|\Phi(t, y_1) - \Phi(t, y_2)| = \frac{4|t|^3}{1 + t^4} \cdot |y_1 - y_2|.$$

So we can find a Lipschitz constant L as

$$L \ge \sup_{t \in [0,1]} \frac{4t^3}{1 + t^4} = 2.$$

So we proved that $\Phi(t, y)$ is Lipschitz continuous in y. According to the uniqueness of initial value problem, this IVP has a unique solution. □

4. Do you think it is a good idea to solve the following IVP numerically?

$$\dot{y} = 1 + y^2, \quad y(0) = 0, \quad t \in [0, 3].$$

Justify your answer. Show Euler's method is going to fail miserably for this IVP.

Answer. No. The function $\Phi(y) = 1 + y^2$ is not Lipschitz continuous on $\mathbb{R}$, so Euler's method will fail. In practice, if we choose $h = 1$, then $\hat{y}(3) = 13$. If $h = 0.5$, then $\hat{y}(3) = 218.1344$. Table 9.1 shows the estimation of $y(3)$ from Euler's method with $h \to 0$. Clearly, when $h \to 0$, $\hat{y}(3)$ diverges quickly. □

Steps	$\hat{y}(3)$
1	3
2	6.375000000000000
3	13
4	28.404234389774501
5	71.073212236802789
6	2.181344382569777e+02
7	8.937891513340965e+02
8	5.455818281020568e+03
9	5.729541345952547e+04

Table 9.1: Divergence of Euler's method on non-Lipschitz functions

Exercise 9.2 — Initial value problem

Consider the following IVP

$$\dot{y} = \arctan(y), \quad y(0) = y_0, \quad t_0 \le t \le T$$

1. Find a Lipschitz constant for $\arctan(y)$.

Answer. Obviously,

$$L \le \max_y \frac{\mathrm{d}}{\mathrm{d}y} \arctan(y) = \max_y \frac{1}{1 + y^2} = 1.$$

So a Lipschitz constant can be $L = 1$. □

2. Find an upper bound on $|\ddot{y}|$ without solving the IVP.

Answer. Differentiate both sides of the equation,

$$\ddot{y} = \frac{1}{1 + y^2},$$

so an upper bound is

$$\sup_{y \in \mathbb{R}} \frac{1}{1 + y^2} = 1.$$

□

3. Find an upper bound on the absolute global error

$$|e_k| = |\hat{y}_k - y(t_k)|,$$

where $\hat{y}_k$ is the Euler's approximation to $y(t_k)$, in terms of step size and t_k.

Answer. Since $\Phi(y) = \arctan(y) \in \mathcal{C}^2(\mathbb{R})$, we have

$$|e_k| \leq h \cdot \frac{\sup_t |\ddot{y}|}{2c} \left[e^{c(T-t_0)} - 1\right].$$

According to the previous result, the second derivative is bounded by 1, and the Lipschitz constant is also $c = 1$. Thus, the global error is bounded by

$$|e_k| \leq \frac{h}{2} \left[e^{T-t_0} - 1\right].$$

□

Exercise 9.3 — Comparison of different methods

Solve the following IVP using the step size $h = 1$

$$\dot{y} = \left(2 + 0.01t^2\right) y, \quad y(0) = 4, \quad t \in [0, 15].$$

1. By Euler's method.
2. By the backward Euler's method.
3. By the second-order Taylor's method.
4. By the Heun's method.
5. By the two-step Adams-Bashforth method.
6. It was mentioned in class that Heun's method, which is derived by applying the trapezoidal rule

$$\int_a^b f(x)\, \mathrm{d}x \approx \frac{1}{2}(b-a)\left(f(a) + f(b)\right)$$

is one the simplest form of Runge-Kutta method. The other simple second-order Runge-Kutta method, which is also known as the modified Euler's method, uses the mid-point rule

$$\int_a^b f(x)\, \mathrm{d}x \approx (b-a) f\left(\frac{a+b}{2}\right).$$

Use this information to derive this second-order Runge-Kutta method. Write a piece of pseudocode for it, then implement it to solve the above IVP.

7. The most widely used Runge-Kutta method is a fourth-order Runge-Kutta method, which uses four sequential evaluations of Φ during each time step, *i.e.*, it has four stages. Similar to the previous two Runge-Kutta, it can be understood from a quadrature rule. In this case, Simpson's rule:

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{b-a}{6} f\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right].$$

This scheme proceeds as follows:

$$\begin{aligned}
&\hat{y}_0 = y_0 \\
&\hat{y}_n = \hat{y}_{n-1} + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
\text{where}\quad &\Phi_1 = \Phi(t_{k-1}, \hat{y}_{k-1}) \\
&\Phi_2 = \Phi\left(t_{k-1} + \frac{h}{2}, \hat{y}_{k-1} + \frac{h}{2}\Phi_1\right) \\
&\Phi_3 = \Phi\left(t_{k-1} + \frac{h}{2}, \hat{y}_{k-1} + \frac{h}{2}\Phi_2\right) \\
&\Phi_4 = \Phi\left(t_{k-1} + h, \hat{y}_{k-1} + h\Phi_3\right)
\end{aligned}$$

Use this fourth-order Runge-Kutta method to solve the above IVP.

8. Compare all of the above approximations to the exact solution by plotting them on the same graph.

9. Use the approximation from Euler's method to find the value of y at $t = 9.625$ by interpolation in Newton's form.

Answer. The table of y values solved by Euler's method, backward Euler's method, second-order Taylor's method, Heun's method, two-step Adams-Bashforth method are second-order Runge-Kutta method based on mid-point rule are listed in Table 9.2. And the result from 4th order Runge-Kutta method is in Table 9.3. All the solutions are plotted in Figure 9.1. Since this function increases too fast, the normal plot cannot present much inforamtion. Thus, it is in log scale. Even in log scale, the function increases cubically, since the exact solution is

$$y(t) = 4e^{\frac{t^3}{300}+2t}$$

From Figure 9.1, we can see that 4^{th} order Runge-Kutta method is the most accurate. Second-order Taylor's method, Heun's method, two-step Adams-Bashforth method are nearly the same. Two-step Adams-Bashforth method is worse. Euler's method is much worse, and backward Euler's method is the worst, since it has negative values and cannot be shown on the graph. By interpolation, the function value at 9.625 is estimated to be

$$\hat{y}(9.625) = 340846.7993708422.$$

□

t	$\hat{y}(t)$
0	4
1	12
2	36.1200000000000
3	109.804800000000
4	339.296832000000
5	1072.17798912000
6	3484.57846464000
7	11708.1836411904
8	40861.5609077545
9	148736.081704226
10	566684.471293103
11	2266737.88517241
12	9542966.49657585
13	42370771.2447968
14	198718917.138097
15	985645829.004960

(a) Euler's method

t	$\hat{y}(t)$
0	4
1	-3.960396039603961
2	3.808073115003809
3	-3.493645059636522
4	3.011762982445278
5	-2.409410385956222
6	1.771625283791340
7	-1.189010257578080
8	0.725006254620781
9	-0.400555941779437
10	0.200277970889718
11	-0.090623516239692
12	0.037140785344136
13	-0.013806983399307
14	0.004664521418685
15	-0.001435237359595

(b) Backward Euler's method

t	$\hat{y}(t)$
0	4
1	20
2	100.801000000000
3	518.197780800000
4	2748.54693925224
5	15207.1605054948
6	88676.7546976665
7	550221.527548081
8	3664502.88454660
9	26402010.3825813
10	207204297.583017
11	1781956959.21395
12	16878785415.5225
13	176835659041.346
14	2056253885115.72
15	26609570276505.6

(c) 2nd order Taylor's method

t	$\hat{y}(t)$
0	4
1	20.0600000000000
2	101.808512000000
3	529.078475161600
4	2847.60616901476
5	16046.2607623982
6	95635.7141438932
7	608549.176240422
8	4169657.24576412
9	30998065.8964596
10	251704295.079252
11	2245202312.10693
12	22106711005.4672
13	241224009149.457
14	2926336699793.90
15	39564072181213.5

(d) Heun's method

t	$\hat{y}(t)$
0	4
1	20.0200000000000
2	76.3803000000000
3	289.983918000000
4	1121.17559493000
5	4450.75132819320
6	18261.1674183208
7	77898.6048349593
8	347301.716339914
9	1625632.75002645
10	8019236.52581924
11	41821786.8782187
12	231164835.908113
13	1356851421.25443
14	8469420520.13577
15	56274387537.5278

(e) Adams-Bashforth method

t	$\hat{y}(t)$
0	4
1	20.0200000000000
2	101.203352250000
3	522.841818561562
4	2792.24326755076
5	15584.0681248543
6	91833.9914469929
7	576813.891978135
8	3895116.05992360
9	28497448.1176130
10	227424240.920712
11	1991383509.56199
12	19227031815.4658
13	205536970107.329
14	2440754089236.66
15	32280071169495.0

(f) 2nd order Runge-Kutta method based on mid-point rule

Table 9.2: Comparison of different methods to solve ODE

Input: function Φ, endpoints t_0, T, initial condition y_0, number of steps$n$
Output: t contains $n+1$ equally spaced mesh points, starting from t_0 to T.
y contains y_0 and approximations to the solution at the corresponding t

```
Runge-Kutta-2nd-order-mid-point (Φ, t_0, T, y_0):
h = (T − t_0)/n;
t[0] = t_0;
y[0] = y_0;
for i = 1 : n do
    y[i] = y[i − 1] + h · Φ(t[i − 1] + h/2, y[i − 1] + h/2 · Φ(t[i − 1], y[i − 1]));
end
return (t, y)
```

Algorithm 9.1: 2^{nd} order Runge-Kutta method based on mid-point rule

t	$\hat{y}(t)$
0	4
1	28.0867667083333
2	200.811414623582
3	1488.36689161515
4	11639.8029402688
5	97724.8224291606
6	895709.119604343
7	9107895.98380737
8	104325021.246580
9	1365504205.66836
10	20695551888.4852
11	367596581036.202
12	7735004488651.82
13	194653712486918
14	5906440410736200
15	217588947569641000

Table 9.3: 4^{th} order Runge-Kutta method

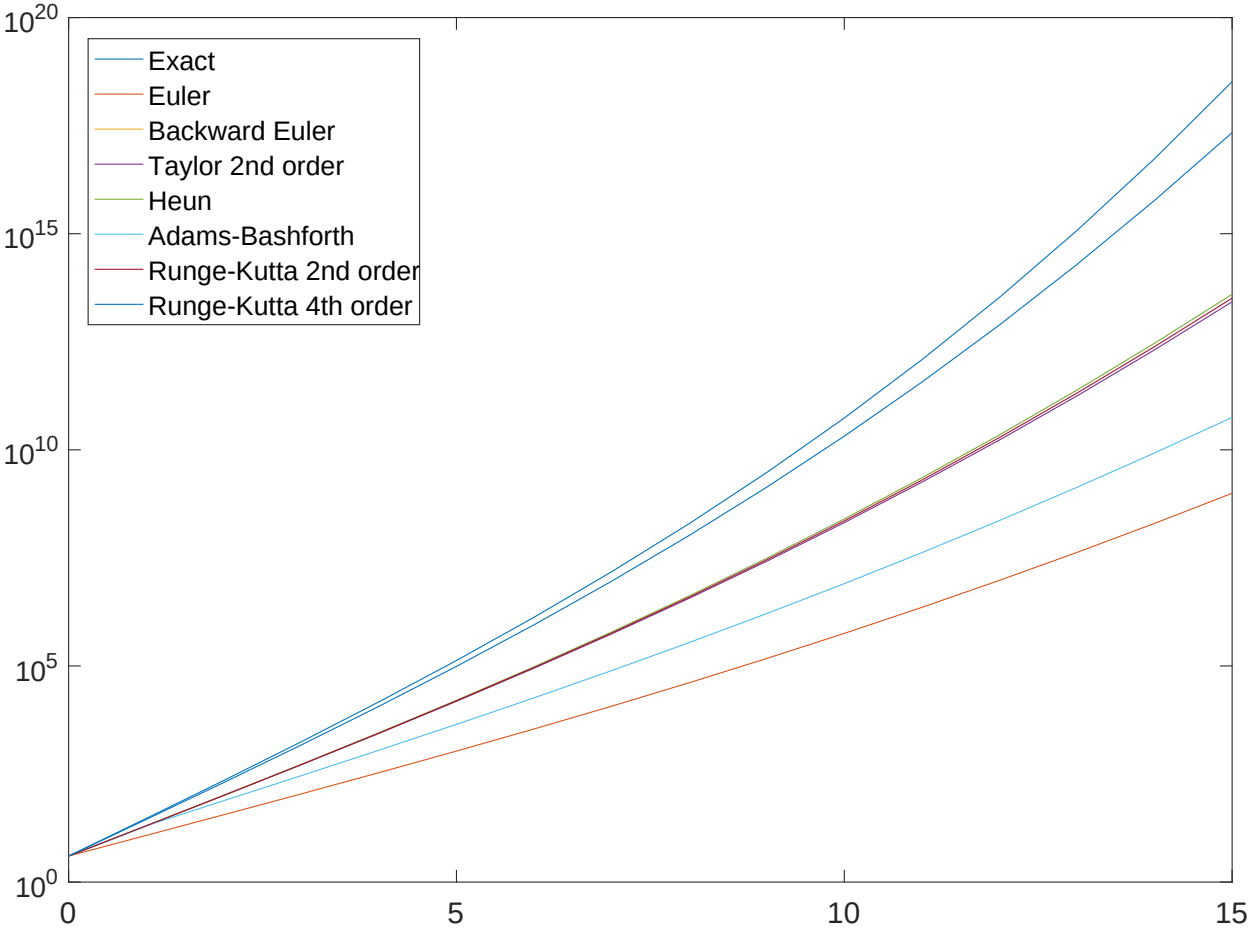

Figure 9.1: Comparison of different methods to solve ODE

Exercise 9.4 — Classic fourth-order Runge-Kutta method

Use the classic fourth-order Runge-Kutta method to find the numerical solution of the following higher-order differential equation, and compare the results to the exact solution.

$$t^3\dddot{y} + t^2\ddot{y} - 2t\dot{y} + 2y = 8t^3 - 2, \quad y(1) = 2, \quad \dot{y}(1) = 8, \quad \ddot{y}(1) = 6$$

for $1 \leq t \leq 2$ with $h = 0.1$. The exact solution is

$$y = -\frac{1}{t} - 1 + 2t + t^2 + t^3.$$

Answer. We alter the dimension of parameters in the code from 4th order Runge-Kutta method to get

```
f=@(t,y) [y(2); y(3); -y(3)/t+2*y(2)/t^2-2*y(1)/t
   ^3+8-2/t^3];
M=100;
h=1/M;
T=zeros(1,M+1);
Y=zeros(3,M+1);
T=1:h:2;
Y(:,1)=[2;8;6];
for j=1:M
        k1=h*f(T(j),Y(:,j));
        k2=h*f(T(j)+h/2,Y(:,j)+k1/2);
        k3=h*f(T(j)+h/2,Y(:,j)+k2/2);
        k4=h*f(T(j)+h,Y(:,j)+k3);
        Y(:,j+1)=Y(:,j)+(k1+2*k2+2*k3+k4)/6;
end
```

The results are shown in Table 9.4. The error is so small that we cannot distinguish them on the plot. So the results are only shown in the table. □

Exercise 9.5 — Shoot method and Finite-different method

Solve the following boundary value problem on the domain $\left[\frac{\pi}{4}, \frac{\pi}{3}\right]$ with $h = \frac{\pi}{60}$

$$\ddot{y} = -\left(2\dot{y}^3 + \dot{y}y^2\right)\sec t, \quad y\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt[4]{2}}, \quad y\left(\frac{\pi}{3}\right) = \frac{\sqrt[4]{12}}{2}.$$

1. By the shoot method.
2. By the Finite-difference method.

Answer. The results are shown in Table 9.5 □

t	$\hat{y}(t)$	$y(t)$	$\lvert\hat{y}(t) - y(t)\rvert$
1.0	2	2	0
1.1	2.83190807328222	2.83190909090909	1.01762686854912e-06
1.2	3.73466631271768	3.73466666666667	3.53948986564490e-07
1.3	4.71777047274397	4.71776923076923	1.24197473549970e-06
1.4	5.78971769910921	5.78971428571428	3.41339492493375e-06
1.5	6.95833931590518	6.95833333333333	5.98257184769579e-06
1.6	8.23100885799572	8.23100000000000	8.85799571470614e-06
1.7	9.61477669743981	9.61476470588235	1.19915574572360e-05
1.8	11.1164598022167	11.1164444444444	1.53577722734610e-05
1.9	12.7427031538000	12.7426842105263	1.89432736927841e-05
2.0	14.5000227413135	14.5000000000000	2.27413135327481e-05

Table 9.4: 4^{th} order Runge-Kutta method on higher order ODE

t	$\hat{y}(t)$
$\frac{15}{60}\pi$	0.840896
$\frac{16}{60}\pi$	0.862058
$\frac{17}{60}\pi$	0.881559
$\frac{18}{60}\pi$	0.899454
$\frac{19}{60}\pi$	0.91579
$\frac{20}{60}\pi$	0.930605

Table 9.5: Shoot method

Exercise 9.6 — Boundary value problem

Solve the following BVP on the domain $[0, 1]$ by using its variational form

$$\ddot{y} + \dot{y} + t = 0, \quad y(0) = 0, \quad y(1) = 0.$$

1. Assume a linear hat basis for the solution.

2. Assume a cubic polynomial basis for the solution.

Answer. The exact solution is

$$y(t) = -\frac{-e(t-1)^2 + e^{1-t} + (t-2)t}{2 - 2e}.$$

□

Exercise 9.7 — Problem of Runge Kutta method

Find the following integral using the 4-order Runge Kutta method.

$$\int_0^1 \left(\frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6 \right) \, \mathrm{d}x.$$

Is equally spaced x_k the best option for this problem?

Answer. We can define

$$y(t) = \int_0^t \left(\frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6 \right) \, \mathrm{d}x.$$

Then,

$$\dot{y} = \frac{1}{(t-0.3)^2 + 0.01} + \frac{1}{(t-0.9)^2 + 0.04} - 6,$$

with initial value

$$y(0) = 0.$$

And the integral is the function value $y(1)$. While the exact evalutation is

$$y(1) = 6 + 5\,(\pi - \arctan(4)) + 10\,(\arctan(3) + \arctan(7)) \approx 29.85832540.$$

This means the 4^{th} order Runge-Kutta method performs quite well on this integral evaluation. And equally spaced t_k is a good option for this problem. □

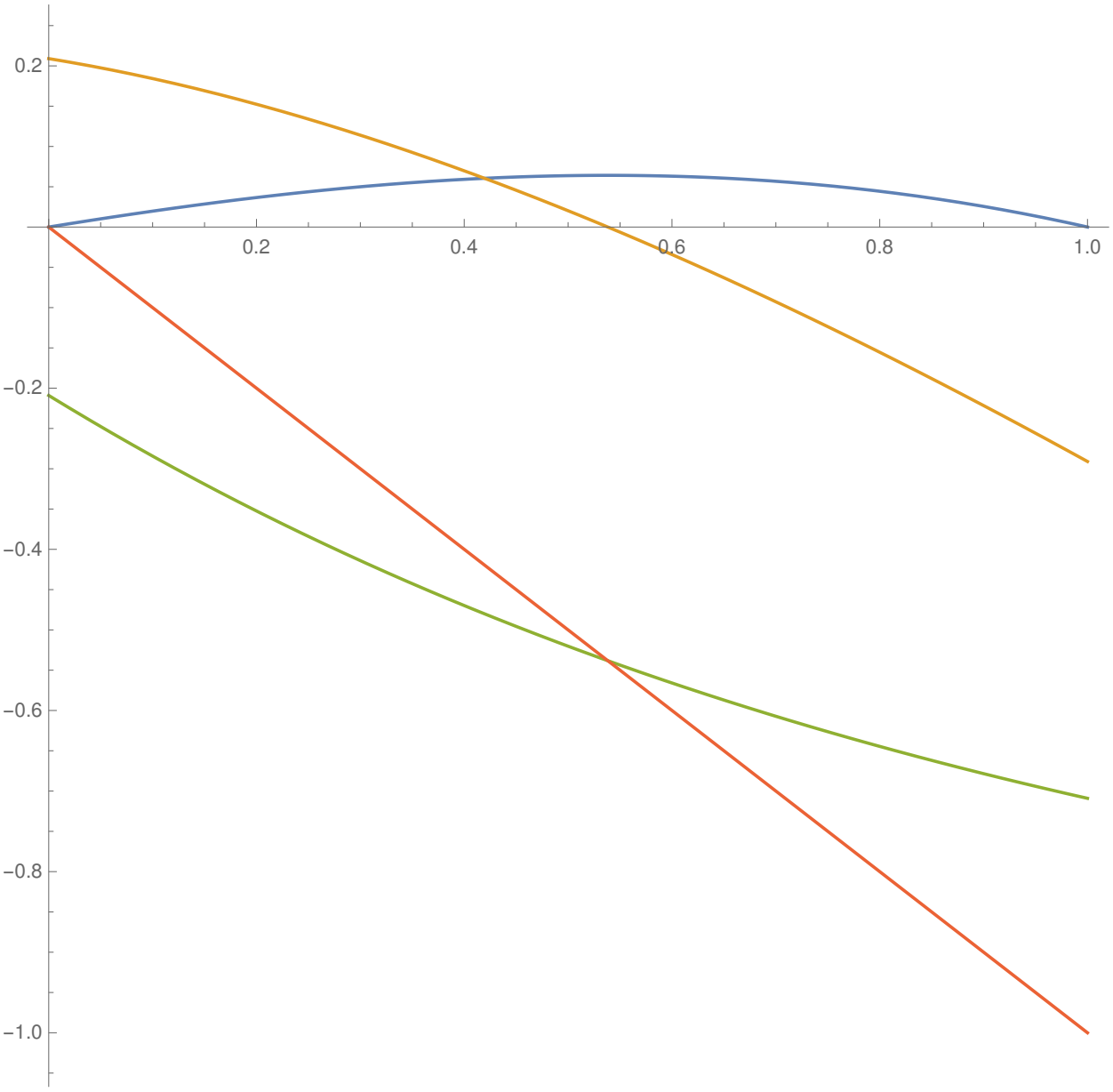

Figure 9.2: Boundary value problem

k	$\hat{y}(t_k)$	k	$\hat{y}(t_k)$	k	$\hat{y}(t_k)$
0	0	34	14.877298001	68	24.1409143884
1	0.0549797183	35	15.6774453655	69	24.2601131419
2	0.1167721427	36	16.4148371731	70	24.3822804222
3	0.1860023452	37	17.0885571363	71	24.5079280952
4	0.2633712757	38	17.7003918965	72	24.6375505499
5	0.3496668534	39	18.2538982241	73	24.771621151
6	0.4457768536	40	18.7536264937	74	24.9105871396
7	0.5527038756	41	19.2045445897	75	25.0548628559
8	0.6715826983	42	19.6116489556	76	25.2048211968
9	0.8037003482	43	19.9797253277	77	25.360783288
10	0.9505191924	44	20.3132179408	78	25.5230064513
11	1.1137033127	45	20.6161717165	79	25.6916707009
12	1.2951482879	46	20.892220427	80	25.8668641907
13	1.4970142441	47	21.1446018307	81	26.0485682705
14	1.7217615646	48	21.3761871463	82	26.2366430615
15	1.9721878412	49	21.5895168706	83	26.4308147107
16	2.2514633309	50	21.78683812	84	26.6306656795
17	2.5631600984	51	21.970140759	85	26.8356295135
18	2.9112668454	52	22.1411908987	86	27.0449914724
19	3.3001767971	53	22.301561155	87	27.2578961188
20	3.7346296374	54	22.4526575251	88	27.4733624733
21	4.2195804869	55	22.5957430065	89	27.6903066514
22	4.7599604317	56	22.7319582098	90	27.9075711014
23	5.3602873909	57	22.8623392735	91	28.1239587749
24	6.0240898629	58	22.9878333929	92	28.3382699313
25	6.7531302607	59	23.1093122629	93	28.5493389215
26	7.5464719933	60	23.2275837006	94	28.7560682986
27	8.3995307747	61	23.3434016808	95	28.9574579555
28	9.3033666064	62	23.4574749778	96	29.1526276186
29	10.2445480798	63	23.570474573	97	29.3408318159
30	11.2058657471	64	23.6830399489	98	29.5214672327
31	12.1679335644	65	23.7957843571	99	29.6940730575
32	13.1113672895	66	23.9092991145	100	29.858325415
33	14.0189628874	67	24.0241569495		

Table 9.6: 4^{th} order Runge-Kutta method on integral evaluation

Exercise 9.8 — Variational form

Consider the following BVP on the domain $[1, 3]$

$$x^3 y^{(4)} + 6x^2 y^{(3)} + 6xy'' - 10x = 0.$$

The boundary conditions are

$$y(1) = y(3) = y'(1) = y'(3) = 0.$$

1. Find its variational form.
2. Solve it using its variational form.
3. Compare your solution and the derivative of your solution with the exact solution and its derivative obtained by writing the differential equation as

$$\left(x^3 y''\right)'' = 10x.$$

Answer. The exact solution is

$$y(x) = \frac{5\left[x^3(\ln 9 - 2) + x^2(12 - 13\ln 3) + 4x\ln x + x(20\ln 3 - 22) + 12 - 9\ln 3\right]}{12x(\ln 3 - 1)}.$$

And the derivative is

$$y'(x) = \frac{5\left(x^2 - 4x + 3\right)\left[4x(\ln 3 - 1) - 4 + 3\ln 3\right]}{12x^2(\ln 3 - 1)}$$

□

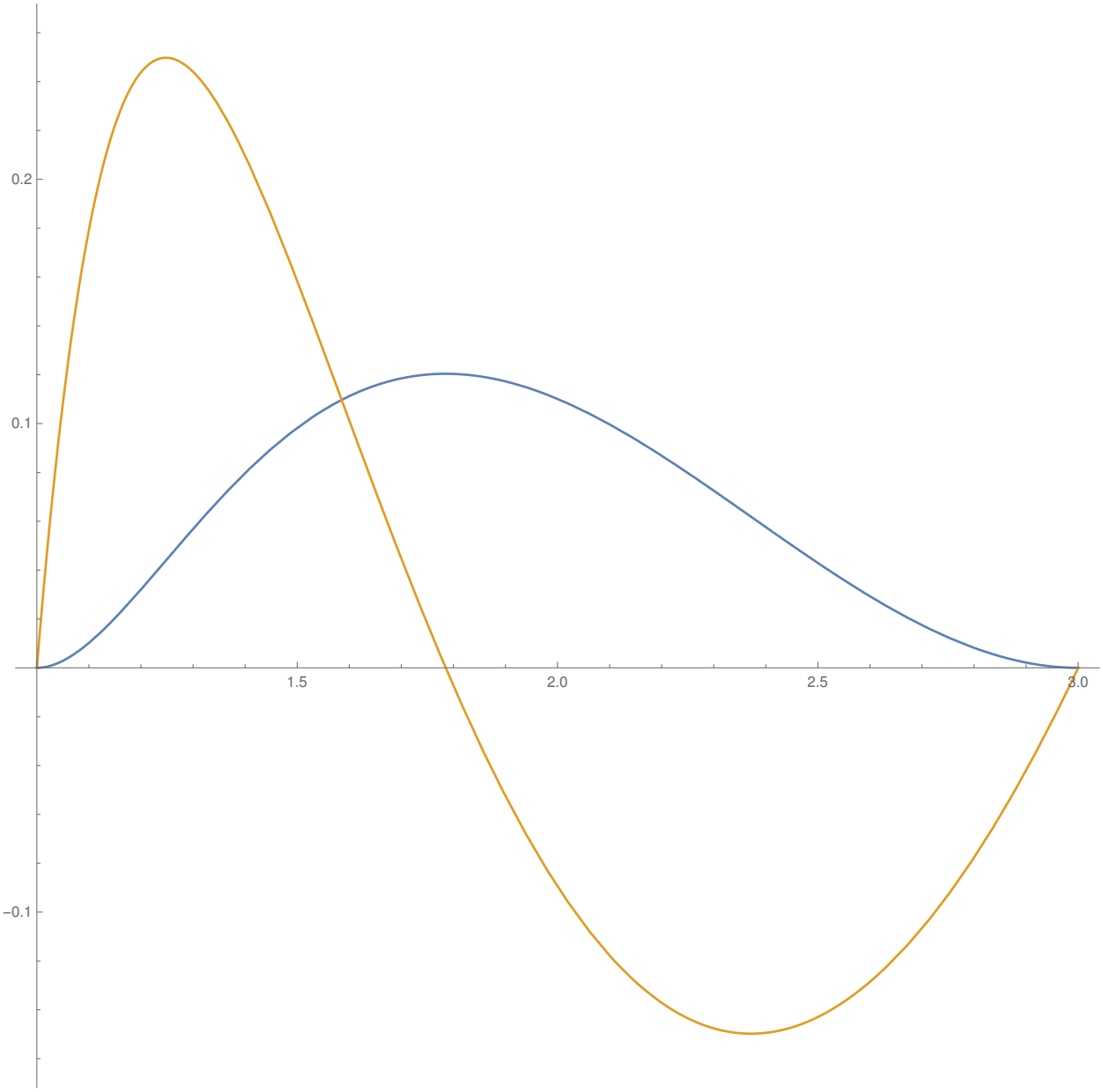

Figure 9.3: Variational form

Part II

Labs

Lab 1

Interpolation

Task 1

This question is to provide you with some experience on Lagrange interpolation, and some MATLAB™ commands you might find useful when comes polynomial approximation.

(a) Use the given M-function lagrinterpol to perform a Lagrange interpolation to estimate the y-value at $x = 4.5$ for the following data.

x	1	2	3	4	5	6
y	66	66	65	64	63	63

(b) Study and write down the output of the following command.

```
poly([2,3,5,7,9])
```

(c) ind the coefficient for the cubic term x^3 of the polynomial

$$f(x) = (x-2)(x-7)(x-1)(x-8)(x-2)(x-8)$$

(d) Study and write down the output of the following commands.

```
a = [ 2, 7, 1, 8, 2, 8];
P = poly([a(1), a(2), a(3)]);
Q = poly([a(4), a(5), a(6)]);
conv(P, Q)
```

(e) Study and write down the output of the following commands.

```
coeff = [a(1), a(2), a(3)]; % vector a from
   above
polyval(coeff, [a(4), a(5), a(6)])
```

(f) Write an M-function which outputs the coefficients of the derivative function of the interpolating polynomial for a set of distinct points (x_i, y_i). Use your function on the data in part (a) to obtain a plot of the derivative function over $[1, 6]$. Write down the coefficients.

Task 2

This question is to use the given M-functions lagrinterpolde and newpoly to study and compare Lagrange's form of interpolation with Newton's form.

(a) Study and write down the output of the following commands.

```
clear all;
close all;
N = 5;
xdata = linspace(0,1000, N+1);
ydata = randn([1,N+1]);
Cnewton = newpoly(xdata,ydata)
Clagrange = lagrinterpolde(xdata,ydata)
```

(b) Study and run the following commands. What can you conclude from it?

```
N = 100;
xdata = linspace(0,1000, N+1);
ydata = randn([1,N+1]);
tic
Cnewton = newpoly(xdata,ydata);
toc
tic
Clagrange = lagrinterpolde(xdata,ydata);
toc
```

(c) Run the following commands. What do the two graphs suggest?

```
N = 10;
xdata = linspace(0,1000, N+1);
rng(0)
ydata = randn([1,N+1]);
Cnewton = newpoly(xdata,ydata);
Clagrange = lagrinterpolde(xdata,ydata);
x_plot_points = linspace(0,1000,500);
plot(xdata, ydata,'.')
hold on
plot(x_plot_points, polyval(Cnewton,
   x_plot_points ),'b')
plot(x_plot_points, polyval(Clagrange,
   x_plot_points), '-.r')
hold off
N = 30;
xdata = linspace(0,1000, N+1);
rng(0)
```

```
ydata = randn([1,N+1]);
Cnewton = newpoly(xdata,ydata);
Clagrange = lagrinterpolde(xdata,ydata);
x_plot_points = linspace(0,1000,500);
plot(xdata, ydata,'.')
hold on
plot(x_plot_points, polyval(Cnewton,
   x_plot_points ),'b')
plot(x_plot_points, polyval(Clagrange,
   x_plot_points), '-.r')
hold off
```

Task 3

In this question, we will investigate so-called the Runge's Phenomenon, and employ what you have learnt in class to minimize it. Consider the following so-called Runge's function

$$f(x) = \frac{1}{1+25x^2}, \quad x \in [-1, 1].$$

(a) Study and write down the output of the following commands.

```
clear all;
close all;
syms x
f = symfun(1/(1+25*x^2), x);
N = 10;
xdata = linspace(-1,1,N+1);
ydata = double(f(xdata));
polyfit(xdata,ydata,N)
```

(b) Construct a polynomial of degree 20 that minimises the sum of squared distances to approximate f using *equally spaced nodes*. Then instead of degree 20, do it again using a 40th degree polynomial. Plot both polynomials with f on the same graph. Write down what you see in your graph. Use the following restriction.

```
axis([-1 1 -5 5]);
```

(c) Now instead of using *equally spaced nodes*, do part (b) again using Chebyshev nodes.

(d) Use interpolating polynomials, of degrees $n = 5, \cdots, 20$ on equally spaced nodes using Newton's form, to approximate $f(x)$. Plot all approximations on the same graph. Use the following restriction.

```
axis([-1 1 0 1]);
```

(e) Now repeat part (d) using Chebyshev nodes.

(f) Suppose evaluations of $y = f(x)$ are very inaccurate due to random errors in y. Shall we use interpolation to approximate $f(x)$? Briefly explain your rationale.

Task 4

1. Find a rational function of the following form

$$g(x) = \frac{a + bx}{c + dx}$$

that interpolates $y = \arctan(x)$ at the points $x = 1, 2, 3$. Plot g with $y = \arctan(x)$.

2. Consider the function

$$h(x) = \sin x + \sin 5x, \quad x \in [0, 2\pi].$$

Study and then use the Matlab command spline to find an approximation for $h(x)$. Briefly outline the strength of this method in comparison with other methods.

Lab 2

Intergration

Task 1

In this question, we will investigate the convergence of some common quadrature rules.

(a) Study and write down the output of the following commands.

```
syms x
fsym = symfun(1/(1+25*x^2), x); %Symbolic f
fsym( -1:1:1 )
fano = @(x) 1./(1+25*x.^2); %Anonymous f
fano( -1:1:1 )
exactf = int(fsym,-1,1);
format longE
eval(exactf)
format short
```

(b) Write an M-function to implement the so-called midpoint quadrature rule

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=1}^{n}(x_k - x_{k-1}) f\left(\frac{x_{k-1}+x_k}{2}\right),$$

where $a = x_0$ and $b = x_n$. The code for you m-file might look like the following:

```
function quad = midpointquad( f, a, b, n)
%-----------------------------------------
%MIDPOINTQUAD Midpoint Quadrature Rule
% Input
% f integrand
% a, b upper and lower limits of integration
% n the number of equally spaced subintervals
% Output
% quad quadrature value
%
```

```
% If f is defined as an M-function use the @
   notation to call
% quad=midpointquad(@f,a,b,n)
% If f is defined as an anonymous/symbolic
   function then simply call
% quad=midpointquad(f,a,b,n)
%-------------------------------------------
xpts = linspace( ??? ) ;
h = ??? ;
xmidpts = zeros(1, ??? );
for i = 1: ???
        xmidpts(i) = ??? ;
end
quad = h * sum ( ??? );
```

(c) Run the following commands and study the output of it.

```
N = 5;
nvec = (2 * 10 .^ (1:1:N));
result = zeros(1, N);
error = zeros(1,N);
for i = 1:N
        result(i) = midpointquad(fano,-1,1,nvec
           (i));
        error(i) = eval(result(i) - exactf);
end
format longE
T = table( categorical(nvec'), categorical(2./
   nvec'), result', error', 'VariableNames', {'n
   ' 'h' 'Midpoint' 'Error'})
format short
```

Use the output to estimate the convergence rate of the midpoint rule in terms of h.

(d) Write an M-function to implement the so-called trapezoidal quadrature rule

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=1}^{n}(x_k - x_{k-1})\frac{f(x_{k-1}) + f(x_k)}{2},$$

using equally spaced subintervals, where $a = x_0$ and $b = x_n$. Name this M-function `trapezoidalquad`, then estimate the convergence rate of the trapezoidal rule in terms of h using Runge's function.

(e) Write an M-function to implement the so-called Simpson's quadrature rule

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=1}^{n}\frac{x_k - x_{k-1}}{6}\left[f(x_{k-1}) + 4\left(\frac{x_{k-1} + x_k}{2}\right) + f(x_k)\right],$$

using equally spaced subintervals. Name this M-function `simpsonquad`, then estimate the convergence rate of the Simpson's rule in terms of h using Runge's function.

(f) Consider applying the midpoint quadrature rule to the following integral

$$\int_0^1 \ln x \, \mathrm{d}x = -1.$$

Estimate the rate of convergence of the midpoint quadrature rule using this integral.

Task 2

In this question, we will test Newton-Cotes, Clenshaw-Curtis, Gaussian quadrature rules.

(a) Use the given M-function newpoly to construct interpolating polynomials in Newton's form of degree $n = 1, 2, 3, 4, 5, 10, 15, 20$ on equally spaced nodes for Gaussian function

$$f(x) = e^{x^2}$$

then integrate the polynomials symbolically over $[-1, 1]$ to approximate

$$\int_{-1}^1 e^{x^2} \, \mathrm{d}x.$$

Note this approach does NOT break $[-1, 1]$ into subintervals.

(b) Repeat part (a), but this time, do it for Runge's function

$$\int_{-1}^1 \frac{1}{1 + 25x^2} \, \mathrm{d}x.$$

(c) Repeat part (b), but this time, do it using Chebyshev nodes instead. Study and compare the accuracy of it with the approximation in part (b).

(d) Repeat part (b), but this time, do it using the given M-function fclencurt and lgwt, which output the Clenshaw-Curtis and Gauss-Legendre nodes and weights, respectively. Compare the accuracy of those two quadrature rules with the approximations in part (b) and part (c).

Task 3

This question is to investigate the so-called degree of exactness of a quadrature rule.

(a) Run the following commands and study the output of it

```
clear all
syms x
N = 5;
miderror = zeros(1,N);
traperror = zeros(1,N);
simerror = zeros(1,N);
for i = 1:N
        fsym = symfun(i*x^(i-1), x);
        miderror(i) = midpointquad(fsym,0,1,1)
           - 1;
        traperror(i) = trapezoidalquad(fsym
           ,0,1,1) - 1;
        simerror(i) = simpsonquad(fsym,0,1,1) -
            1 ;
end
format long
monomial = {'$1$';'$2x$';'$3x^2$';'$4x^3$';'$5x
   ^4$'};
T = table(monomial , miderror', traperror',
   simerror', 'VariableNames', {'func' '
   MidpointError' 'TrapezoidalError' '
   SimpsonError' })
format short
```

What can you conclude from the output?

(b) Do a similar investigation for Clenshaw-Curtis and Gauss-Legendre.

Task 4

This question is to combine and extend what we have learnt about quadrature rules so far.

1. Based on Task 1-3, propose an algorithm by writing an M-function that performs a numerical integration, and test it using the following integral

$$\int_{-1}^{1} \frac{1}{1+25x^2}\,\mathrm{d}x.$$

2. Suggest two additional techniques that can be used to numerically solve the following integrals, or integrals that have similar features.

$$\int_{0}^{\infty} \frac{1}{1+25x^2}\,\mathrm{d}x \quad \text{and} \quad \int_{-1}^{1} \sqrt{\left|x-\frac{1}{2}\right|}\,\mathrm{d}x.$$

Lab 3

Root Finding

Task 1

In this question, we will continue our study on how to write an M-function.

(a) Study the M-function myplot and write down what the variable varargin in an M-function does.

```
n = 20; xmin = -2; xmax = 2;
x = linspace( xmin, xmax, n);
y = x.^2 - 2;
myplot(x, y,'$$y=x^2-\ln(x^2+2)$$', 'color', 'b
   ', 'linestyle', '-.')
```

(b) Write down what the variables nargin, argout and varargout in an M-function do.

(c) Study and write down the output of the following commands.

```
n = 5; a = 1; b = 2;
f = @(x) x.^2 - log(x.^2+2);
xvec = zeros(1,n);
for j = 1:n
        xvec(j) = (a+b)/2;
        if f(a)*f(xvec(j)) < 0
                b = xvec(j);
        else
                a = xvec(j);
        end
end
format long
table( (1:n)', xvec', 'VariableNames', {'n', '
   x_n'})
format short
```

(d) Study the M-function naiveFP, then add comments to the main body of it.

(e) Run the following, then explain why fixed point iteration can be used here.

```
clear all; close all;
g = @(x) sqrt(log(x.^2+2));
xvec = naiveFP(g, 1, 1e-6, 10, 'Numerical
   solutions to $x^2=\ln(x^2+2)$', 'b--o');
```

(f) The absolute error, or simply error, is defined as

$$E_k = |x_k - x^*|,$$

where x is the true root. The relative error is defined as

$$\epsilon_k = \left| \frac{x_k - x^*}{x^*} \right|,$$

In practice, the true root is not known, we estimate the errors by

$$E_k = |x_k - x_{k-1}| \quad \text{and} \quad \epsilon_k = \left| \frac{x_k - x_{n-1}}{x_{n-1}} \right|$$

Estimate the errors in part (e). Write down your estimates below. Based on this empirical data, does the sequence $\{x_k\}$ converge linearly or quadratically?

(g) Instead of the regular fixed point iteration $x_{k+1} = g(x_k)$, implement

$$\begin{aligned} \alpha &= g(x_k), \quad \beta = g(\alpha), \\ x_{k+1} &= x_k - \frac{(\alpha - x_k)^2}{\beta - 2\alpha + x_k} \\ &= \frac{\beta x_k - \alpha^2}{\beta - 2\alpha + x_k}, \quad \text{for } k = 0, 1, \cdots, n \end{aligned}$$

Does this iteration scheme converges fast?

(h) Add a warning message to your code for part (g) for dividing a small number.

Task 2

Use what you have learnt in Task 1 to find the root of

$$f(x) = \tan(x) - (x - 0.4)(x - 0.7)$$

near $x = 0$. Use $tol = 10^{-10}$ and $nmax = 50$.

Task 3

Consider the function

$$f(x) = e^x - 4x^2.$$

(a) Plot $f(x)$ over various intervals to confirm that the function has 3 roots:

$$x_1^* \in (-1, 0), \quad x_2^* \in (0, 1), \quad x_3^* \in (4, 4.5).$$

(b) Study and use the M-function bisect to find x_1^*.

(c) Study and use the M-function newton to find x_1^*.

(d) Study and use the M-function fixpt to find x_1^*.

Task 4

Newton's method converges slowly when derivative/s are zero at the root. In assignment 7, there is a question on how to speed up Newton's method when the order m is known in advance. However, this information is unlikely to be available in practice. This question illustrates how to estimate the order m. Consider

$$f(x) = (x-2)^3 e^x, \quad x \in [0, 3].$$

(a) Plot $f(x)$ and its derivatives over the interval $[0, 3]$ to determine the order.

(b) Implement the following iteration scheme for the function

$$m_k = \frac{x_{k-1} - x_{k-2}}{2x_{k-1} - x_k - x_{k-2}}, \quad k \geq 2,$$

where $\{x_k\}$ are the sequence computed by Newton's method. What do you notice?

Task 5

In MATLAB™ there exist functions for finding roots of a function: roots, fzero, fsolve. Consider the following function

$$f(x) = x^2 - \sin(x + 1).$$

(a) Use the M-function newton to find the root/s of f.

(b) Use the M-function secant to find the root/s of f.

(c) Study and fzero to find the root/s.

Task 6

Find y that satisfies the following equation

$$0.7 = 1 - \frac{2}{a}\left[\frac{1}{a}\int_0^a \frac{x}{\exp(x)-1}\,\mathrm{d}x + \frac{a}{6} - 1\right].$$

State and compare your solution with the solution from fsolve. Explain your method.

Lab 4

Differential Equations

Task 1

In this question we compare the convergence of a few methods we have done in class.

$$\dot{y} = \frac{y+t}{y-t}, \quad y(0) = 1.$$

(a) MATLAB™ can solve simple differential equations symbolically. For example,

```
equation = 'Dy = (y+t)/(y-t)';
initial = 'y(0) = 1';
y = dsolve(equation, initial, 't');
pretty(y)
x = linspace( 0, 1, 20);
z = eval(vectorize(y));
plot(x,z)
```

Run the above commands and write down the exact solution below.

(b) Study and use the M-function butcher and euler to estimate the convergence rate of Euler's method for the above IVP over the interval $[0, 1]$. In addition, compute the error as the difference between your approximation and the exact solution at $t = 1$.

(c) Repeat part (b), but this time do it for Heun's method. The M-function heun is given.

(d) Repeat to part (b), but this time do it for Aadms-Bashforth's method. The M-function ab2 is given.

(e) Repeat to part (b), but this time do it for the classic 4-order Runge Kutta's method. The M-function rk4 is given.

(f) Repeat to part (b), but this time do it for the classic 4-order Taylor's method. The M-function taylor is given.

Task 2

MATLAB™ has its implementation of Runge-Kutta. This question looks at the build-in functions ode23 and ode45, which implement versions of 2nd/3rd-order Runge-Kutta and 4th/5th-order Runge Kutta, respectively.

(a)) Study and use ode23 to solve the following

$$\dot{y} = \frac{y+t}{y-t}, \quad y(0) = 1.$$

Compare and comment the convergence with the methods we considered in Task 1.

(b) Write an M-function for the vector-valued function

$$\Phi(t, x) = \begin{bmatrix} 10(x_2 - x_1) \\ 28x_1 - x_2 - x_1x_3 \\ -8/3x_3 + x_2x_2 \end{bmatrix}$$

The input of your function shall be a scalar t and a vector in $\mathbb{R}^3$. The output of your function shall be a column vector.

(c) Study and use ode45 to solve the following IVP over $[0, 20]$.

$$\dot{x} = \Phi(x), \quad x(0) = \begin{bmatrix} -8 \\ 8 \\ 27 \end{bmatrix}.$$

(d) Let X denote the matrix that contains approximations you have found in part (c) for x_1, x_2 and x_3 as its columns at various tk values. Plot the following

```
plot(X(:,1), X(:,2))
plot(X(:,1), X(:,3))
plot(X(:,2), X(:,3))
subplot(3,1,1)
plot(t,X(:,1))
subplot(3,1,2)
plot(t,X(:,2))
subplot(3,1,3)
plot(t,X(:,3))
```

What do those graphs suggest? What happens if we change the initial condition x_0?

Task 3

There are IVPs that defeat the explicit Adams-Bashforth and Runge-Kutta methods. In fact, for such problems, the higher order method perform even more poorly than the low order methods. These problems are call "stiff" ODEs. Consider

$$y' = \lambda(-y + \sin x), \quad y(0) = 0.$$

whose exact solution is

$$y(x) = \frac{\lambda}{1+\lambda^2} e^{-\lambda x} + \frac{\lambda^2}{1+\lambda^2} \sin x - \frac{\lambda}{1+\lambda^2} \cos x.$$

(a) Run the following commands. The M-functions stiff2ode and stiff2solution are given.

```
clear all
h = 0.1; % mesh size
[x,y] = meshgrid ( 0:h:2*pi, -1:h:1 );
px = ones ( size ( x ) );
py = stiff2ode ( x, y );
quiver ( x, y, px, py )
axis tight equal
hold on
x1=(0:h:2*pi);
y1=stiff2solution(x1);
plot(x1,y1,'r')
hold off
```

What does the direction field seem to suggest? What happens if λ increases? Describe your understanding of stiffness.

(b) Use euler to solve the IVP when $\lambda = 10000$ for 40 steps over the interval $[0, 2\pi]$. Comment your solution and provide an explanation of it.

(c) Use rk4 to solve the IVP when $\lambda = 10000$ for 40 steps over the interval $[0, 2\pi]$. Is it better than Euler's? What happens if we increase the number of steps?

(d) Study and use the build-in function ode15s to solve the IVP when $\lambda = 10000$. Write down the estimated value of $y(2\pi)$.

Task 4

Consider the following boundary value problem.

$$\ddot{y} - 3\dot{y} + 2y = 0, \quad y(0) = 0, \quad y(1) = 10.$$

(a) Study and use the built-in function bvp4c to solve the above BVP over $[0, 1]$. Write the estimated value of $y(1/\pi)$.

(b) Write an M-function to solve the above BVP using the shooting method.

(c) Write an M-function to solve the above BVP using the finite-difference.

Part III

Projects

Project 1

Linear Prediction of Speech

Guangting Yu, University of Michigan - Shanghai Jiao Tong University Joint Institute, Minhang, Shanghai, 200135, China

Abstract

This paper finds an efficient algorithm to predict the speech signal based on the previously recorded speech signal. The prediction is linear in that the prediction signal can be written as a linear combination of previous signals, while the coefficients are the core of the prediction. Obtaining the best coefficients is equivalent to solving a system of linear equations, where Cholesky decomposition is applied to achieve the goal. The pseudocode and MATLAB™ code of the Cholesky decomposition are implemented.

1 Introductory Background

Speech production is the result of an excitation signal generated by the contraction of the lungs when they expel air. It is then modified by resonances when passing through the trachea, the vocal cords, the mouth cavity, as well as various muscles. The excitation signal is either created by the opening and closing of the vocal cords, or by a continuous flow of air. Introduced in the early 1960s by Fant, *the source-filter* model assumes that the glottis and vocal tract are fully uncoupled. This initial idea was reused to develop the *Linear Predictive* (LP) model for speech production.[**dutoit**]

In this model the speech signal is the output $y[n]$ of an *all-pole filter*[1] $1/A(z)$ excited by $x[n]$. Calling $Y(z)$ and $X(z)$ the Z-transform of the speech and excitation, respectively, the model is described by

$$Y(z) = \frac{X(z)}{1 - \sum_{i=0}^{p} a_i z^{-i}} = \frac{X(z)}{A_p}. \tag{1.1}$$

Applying the inverse Z-transform to this equation we observe that the speech can be

[1] A filter whose frequency response function goes infinite at specific frequencies

linearly predicted from the previous p samples and some excitation:

$$y[n] = x[n] + \sum_{i=1}^{p} a_i y[n-i]. \tag{1.2}$$

[**cc12**]

2 Model Construction and Simplification

Our goal is to explain as much as possible of $y[n]$ through the a_i , *i.e.*, we look at $x[n]$ as an error, and we strive at rendering it as small and simple as possible. For the sake of clarity we therefore rename $x[n]$ into $e[n]$. The question we want to answer is how to select the a_i such as to minimize the energy

$$E = \sum_{m=-\infty}^{\infty} e^2[m]. \tag{1.3}$$

Plug in equation (1.2), we have the error expressed by an infinite sum

$$E = \sum_{m=-\infty}^{\infty} \left(y[m] - \sum_{i=1}^{p} a_i y[m-i] \right)^2.$$

However, infinite sum cannot be computed, and it needs to be truncated. This can be achieved by applying the covariance method, which consists in windowing the error

$$E_n = \sum_{m=n}^{n+N-1} \left(y[m] - \sum_{i=1}^{p} a_i y[m-i] \right)^2. \tag{1.4}$$

Furthermore, we observe the recursive relationship between these terms

$$\sum_{m=-\infty}^{\infty} y[m]y[m-k] = \sum_{m=-\infty}^{\infty} \left(x[m] + \sum_{i=1}^{p} a_i y[m-i] \right) y[m-k], \quad k = 1, \cdots, p. \tag{1.5}$$

We want to make the prediction unbiased, so the error should have zero mean.

$$\sum_{m=-\infty}^{+\infty} x[m] = 0.$$

Furthermorre, we also want the error be independent to the signal

$$\sum_{m=-\infty}^{+\infty} x[m]y[m-k] = 0.$$

Then, equation (1.5) can be simplified to

$$\sum_{m=-\infty}^{\infty} y[m]y[m-k] = \sum_{i=1}^{p} a_i \sum_{m=-\infty}^{\infty} y[m-i]y[m-k], \quad k = 1, \cdots, p. \tag{1.6}$$

Again, we apply the truncation to avoid infinite calculation,

$$\sum_{m=n}^{n+N-1} y[m]y[m-k] = \sum_{i=1}^{p} a_i \sum_{m=n}^{n+N-1} y[m-i]y[m-k], \quad k = 1, \cdots, p. \tag{1.7}$$

Define

$$\phi_n(k,i) = \sum_{m=n}^{n+N-1} y[m-k]y[m-i]. \tag{1.8}$$

Then, equation (1.7) can be written in

$$\phi_n(k,0) = \sum_{i=0}^{p} a_i \phi_n(k,i), \quad k = 1, \cdots, p. \tag{1.9}$$

This can be further written in the form of linear equation

$$\begin{pmatrix} \phi(1,0) \\ \vdots \\ \phi(p,0) \end{pmatrix} = \begin{pmatrix} \phi(1,1) & \cdots & \phi(1,p) \\ \vdots & & \vdots \\ \phi(p,1) & \cdots & \phi(p,p) \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix}. \tag{1.10}$$

Solving for $a_i, 1 \leq i \leq p$ requires to invert the matrix Φ. This can be achieved through Gauss elimination. However, we will use Cholesky decomposition 1.6 in this paper instead.

3 Mathematical Basis

3.1 Linear Algebra

Definition 1.1 (Hermitian matrix)**.** A Matrix $A \in \mathbb{C}^{n\times n}$ is *Hermitian* if it is equal to the conjugate transpose of itself, *i.e.*,

$$A = \bar{A}^T.$$

Definition 1.2 (Eigenvalues ad eigenvectors)**.** A number $\lambda \in \mathbb{C}$ is called an *eigenvalue* of $A \in \mathbb{C}^{n\times n}$ if there exists $v \in \mathbb{C}^n$ such that $v \neq 0$ and

$$Av = \lambda v.$$

Such a vector v is called an *eigenvector*.

Definition 1.3 (Spectrum)**.** The *spectrum* of $A \in \mathbb{C}^{n\times n}$, denoted as σ_A, is the set of all the eigenvalues of A, *i.e.*,

$$\sigma_A \coloneqq \{\lambda \mid \lambda \text{ is a eigenvalue of } A\}$$

Definition 1.4 (Spectrum radius)**.** The *spectrum radius* of $A \in \mathbb{C}^{n\times n}$, denoted as $\rho(A)$, is the largest absolute value of all its eigenvalues, *i.e.*,

$$\rho(A) \coloneqq \max_{\lambda\in\sigma_A} |\lambda|.$$

Theorem 1.1. *If a matrix $A \in \mathbb{C}^{n\times n}$ is Hermitian, then $\sigma_A \subset \mathbb{R}$.*

Proof. Take any eigenvalue $\lambda \in \sigma_A$ and its corresponding eigenvector $v \in \mathbb{C}^n$. By definition,

$$Av = \lambda v.$$

Multiply both sides by the conjugate transpose of v,

$$\bar{v}^T Av = \lambda \bar{v}^T v. \tag{1.11}$$

The left hand side of the equation has the conjugate transpose

$$\overline{\bar{v}^T Av}^T = \bar{v}^T \bar{A}^T v.$$

Since A is Hermitian, we plug in $\bar{A}^T = A$ on the right hand side and

$$\overline{\bar{v}^T Av}^T = \bar{v}^T Av,$$

which implies that $\bar{v}^T Av$ is also Hermitian. Furthermore, $\bar{v}^T Av \in \mathbb{C}^{1\times 1}$ implies that it is a real number. Then, according to equation (1.11)

$$\lambda = \frac{\bar{v}^T Av}{\bar{v}^T v}.$$

The denominator is the inner product of v with itself defined on $\mathbb{C}^n$, so $\bar{v}^T v \in \mathbb{R}_*^+$ since $v \neq 0$. Both the numerator and the denominator are real, so the quotient λ is also real. Since λ is arbitrarily selected from σ_A, all elements in σ_A are real. Furthermore, σ_A has at most n elements, so it is a proper subset of $\mathbb{R}$. □

Theorem 1.2. *If $A, B \in \mathbb{C}^{n\times n}$, then $\sigma_{AB} = \sigma_{BA}$.*

Proof. Select $\lambda \in \sigma_{AB}$.

If $\lambda = 0$, then there exists $v \in \mathbb{C}^n, v \neq 0$ such that $ABv = \lambda v = 0$. So the matrix AB is not full rank, and thus $\det(AB) = 0$. Then $\det(BA) = \det(AB) = 0$, which implies BA is not full rank either. Thus, there exists a nonzero vector $u \in \mathbb{C}^n$ such that $BAu = 0$. This means 0 is also an eigenvalue of BA.

If $\lambda \neq 0$, then there exists $v \in \mathbb{C}^n, v \neq 0$ such that $ABv = \lambda v$. Then, multiply both sides by B,

$$BABv = \lambda Bv.$$

This also means BA has λ as its eigenvalue with corresponding eigenvector Bv. Combine both cases, $\lambda \in \sigma_{BA}$. Since λ is arbitrarily selected from σ_{AB}, we have $\sigma_{AB} \subseteq \sigma_{BA}$. Exchange the name of A and B, we also have $\sigma_{AB} \supseteq \sigma_{BA}$. Therefore, $\sigma_{AB} = \sigma_{BA}$. This directly implies $\rho(AB) = \rho(BA)$. □

Definition 1.5 (Positive definite)**.** Let $A \in \mathbb{R}^{n\times n}$ be a symmetrix matrix, *i.e.*, $A = A^T$. It is *positive definite* if and only if

$$\forall x \in \mathbb{R}^n, x \neq 0 \qquad x^T Ax > 0.$$

Theorem 1.3. *Let $A \in \mathbb{R}^{n\times n}$ be a symmetrix matrix, i.e., $A = A^T$. Then A is positive definite if and only if $\sigma_A \subset \mathbb{R}_*^+$, i.e., all its eigenvalues are positive.*

Proof. Select $\lambda \in \sigma_A$ and its corresponding eigenvector $v \in \mathbb{R}^n, v \neq 0$. The equation (1.11) for $\mathbb{R}^{n\times n}$ deduces

$$\lambda = \frac{v^T A v}{v^T v}.$$

Since $v \neq 0$, its inner product of itself $v^T v > 0$. Then,

$$v^T A v > 0 \iff \lambda = \frac{v^T A v}{v^T v} > 0.$$

□

Definition 1.6 (Lower triangular matrix)**.** A matrix $A \in \mathbb{C}^{n\times n}$ is *lower triangular* if all the elements above its main diagonal are zero, *i.e.*,

$$\forall a_{ij} \in A, i < j \implies a_{ij} = 0.$$

Theorem 1.4 (Determinant of lower triangular matrix)**.** *The determinant of a lower triangular matrix $A \in \mathbb{C}^{n\times n}$ is the product of its main diagonal elements, i.e.,*

$$\det(A) = \prod_{i=1}^{n} a_{ii}.$$

Proof. This is a direct result from cofactor expansion of determinant. And Gauss elimination also gives the same result. □

3.2 System of Linear Equations

The system of linear equations

$$\begin{cases} a_{11}x_1 + \cdots + a_{1n}x_n = b_1 \\ \qquad\qquad \vdots \\ a_{n1}x_1 + \cdots + a_{nn}x_n = b_n \end{cases} \tag{1.12}$$

can be written in matrix form

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{1.13}$$

with a simplified experssion

$$AX = B.$$

Theorem 1.5. *The solution to the system of linear equations*

$$AX = B$$

where $A \in \mathbb{R}^{n\times n}, X, B \in \mathbb{R}^n$ is unique if nad only if A is invertible.

Proof. If A is invertible, we can find X by multiplying both sides by its inverse,

$$A^{-1}AX = A^{-1}B \implies X = A^{-1}B,$$

which is unique. If A is not invertible, then the kernel (null space) of A has nonzero dimension, and its range is proper subspace of $\mathbb{R}^n$.

- $B \in \text{range}(A)$, then the solution exists by definition, but it is not unique. If X is a solution to the system of equations, then we can select a nontrivial element $v \in \text{null}(A)$ with $v \neq 0$ and a nontrivial scalar $c \in \mathbb{R}, c \neq 0$ so that

  $$A(X + cv) = AX + cAv = AX + 0 = B,$$

  which means $X + cv$ is also a solution, and uniqness is violated.

- $B \notin \text{range}(A)$, then the solution doesn't exist by definition.

□

Theorem 1.6 (Cholesky decomposition)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric and positive definite matrix. Then, there exists a unique lower triangular matrix L with positive diagonal entries such that*

$$A = LL^T,$$

where the entries l_{ij} can be computed as follows: $l_{11} = \sqrt{a_{11}}$, and for $i = 2\cdots n$,

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}}{l_{jj}}, \quad j = 1\cdots, i-1, \tag{1.14}$$

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}. \tag{1.15}$$

Proof. We prove this by induction on the dimension of the matrix A. For $n = 1$, the result is trivially true, since a positive scalar has a positive square root. Assume this statement holds for $n = k$, then there exists an lower triangular matrix L_k such that

$$A_k = L_k L_k^T.$$

Since A_{k+1} is symmetric, we can partition it as

$$A_{k+1} = \begin{pmatrix} A_k & v \\ v^T & \alpha \end{pmatrix}$$

with $\alpha \in \mathbb{R}_*^+, v \in \mathbb{R}^k$. We need a lower trianglar factorization in the form

$$A_{k+1} = L_{k+1}L_{k+1}^T = \begin{pmatrix} L_k & 0 \\ l^T & \beta \end{pmatrix} \begin{pmatrix} L_k^T & l \\ 0 & \beta \end{pmatrix}.$$

Then we get two equations to satisfy

$$\begin{aligned} L_k l &= v, \\ l^T l + \beta^2 &= \alpha, \end{aligned}$$

which directly gives the solution

$$l = L_k^{-1} v, \tag{1.16}$$

$$\beta = \sqrt{\alpha - l^T l}. \tag{1.17}$$

Here, we also need to prove that L_k is invertible and $\beta \in \mathbb{R}_*^+$. According to Theorem 1.4, we know that the determinant of L_k is the product of its diagonal elements, and according to the induction hypothesis, they are all positive. So the product is also positive, which means L_k is invertible, and thus (1.16) gives a well defined vector. Furthermore, the determinant of the product of two matrices is the product of the determinants of these two matrices, *i.e.*,

$$\det(A_{k+1}) = \det(L_{k+1}) \cdot \det(L_{k+1}^T) = \beta^2 \det(L_k)^2 > 0,$$

where the positiveness of $\det(A_{k+1})$ comes from the induction hypothesis that it is positive definite, so that its determinant is the product of all its eigenvalues. With $\det(L_k) > 0$, we obtain $\beta^2 > 0$, which proves that (1.17) defines a positive real number. So the existence of the Cholesky decomposition is already proved, and it is easy to observe that (1.15) is equivalent to (1.17), and (1.14) is equivalent to (1.16). [**tam37**] □

3.3 Functional Analysis

Definition 1.7 (Norm)**.** Given a vector space V over a subfield $\mathbb{F} \subseteq \mathbb{C}$, a *norm* on V is a function $||\cdot|| : V \to \mathbb{R}$ with the following properties:

For all $c \in \mathbb{F}$ and all $u, v \in V$,

1. $||v|| \geq 0$; Nonnegative
2. $||v|| = 0 \implies v = 0$; Positive
3. $||cv|| = |c| \cdot ||v||$; Homogeneous
4. $||u + v|| \leq ||u|| + ||v||$. Subadditive

The subadditivity is also referred to as the triangle inequality.

Definition 1.8 (Matrix norm)**.** A *matrix norm* is a function $||\cdot|| : \mathbb{C}^{n\times n} \to \mathbb{R}$ if for all $A, B \in \mathbb{C}^{n\times n}$, it satisfies the following five properties.

1. $||A|| \geq 0$; Nonnegative
2. $||A|| = 0 \implies A = 0$; Positive
3. $||cA|| = |c| \cdot ||A||$; Homogeneous
4. $||A + B|| \leq ||A|| + ||B||$; Subadditive
5. $||AB|| \leq ||A|| \cdot ||B||$. Submultiplicative

[**horn**]

Theorem 1.7. *The set of functions on $A \in \mathbb{C}^{m\times n}$ with $a_{ij} \in A, \forall i \in [1,m]\cap\mathbb{N}, j \in [1,n]\cap\mathbb{N}$,*

$$N_p(A) := \begin{cases} \left(\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^p\right)^{\frac{1}{p}} & p \in \mathbb{N}^* \\ \max\limits_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}| & p = +\infty \end{cases} \tag{1.18}$$

define a set of norms.

Proof. We separate the cases where $p = +\infty$ and $p \in \mathbb{N}^*$.

- For $p = +\infty$:

  1. Nonnegative:
  $$N_\infty(A) = \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}| \geq 0.$$
  2. Positive:
  $$N_\infty(A) = 0 \implies \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}| = 0 \implies \forall i \in [1,m]\cap\mathbb{N}, j \in [1,n]\cap\mathbb{N}, |a_{ij}| = 0,$$
  which means all the entries of A are 0 and thus $A = 0$.
  3. Homogeneous:
  $$N_\infty(cA) = \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|c\cdot a_{ij}| = \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|c|\cdot|a_{ij}| = |c|\max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}| = |c|\cdot N_\infty(A).$$
  4. Subadditive:
  $$\begin{aligned} N_\infty(A+B) &= \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}+b_{ij}| \leq \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}(|a_{ij}|+|b_{ij}|) \\ &\leq \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|a_{ij}| + \max_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}|b_{ij}| = N_\infty(A) + N_\infty(B). \end{aligned}$$

- For $p \in \mathbb{N}^*$, only property 1,2 and 3 of the definition will be proved. The property 4 will be proved in the following context by three famous theorems: Young's inequality 1.8, Hölder's inequality 1.9 and Minkowski's inequality 1.10.

1. Nonnegative:
$$N_p(A) = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p \right)^{\frac{1}{p}} \geq 0.$$

2. Positive:
$$N_p(A) = 0 \implies \sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p = 0 \implies \forall i \in [1, m] \cap \mathbb{N}, j \in [1, n] \cap \mathbb{N}, |a_{ij}| = 0,$$
which means all the entries of A are 0 and thus $A = 0$.

3. Homogeneous:
$$N_p(cA) = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |c \cdot a_{ij}|^p \right)^{\frac{1}{p}} = \left(|c|^p \cdot \sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p \right)^{\frac{1}{p}} = |c| \cdot N_p(A).$$

4. As for subadditivity, we need to prove
$$N_p(A + B) \leq N_p(A) + N_p(B),$$
which can be explicitly expressed as
$$\left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij} + b_{ij}|^p \right)^{\frac{1}{p}} \leq \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p \right)^{\frac{1}{p}} + \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |b_{ij}|^p \right)^{\frac{1}{p}}.$$
The dimension of the matrix does not matter here, so that we can compress it to one dimensional sequences,
$$\left[\sum_{i=1}^{k} |a_i + b_i|^p \right]^{\frac{1}{p}} \leq \left(\sum_{i=1}^{k} |a_i|^p \right)^{\frac{1}{p}} + \left(\sum_{i=1}^{k} |b_i|^p \right)^{\frac{1}{p}}, \tag{1.19}$$
where $k \in \mathbb{N}^*$. And we can also extend the domain to $p \in (1, +\infty)$, without the constraint of being an integer. This is the discrete sum case of Minkowski's inequality , which will be proved in the following context, with the support of Hölder's inequality.

□

Theorem 1.8 (Young's inequality)**.** *Let $p, q \in \mathbb{R}_*^+$ be strictly positive real numbers such that $\frac{1}{p} + \frac{1}{q} = 1$. Then, for any $a, b \in \mathbb{R}^+$,*
$$ab \leq \frac{a^p}{p} + \frac{b^q}{q}. \tag{1.20}$$
Equality occurs if and only if
$$b = a^{p-1}.$$

Proof. We write the left hand side of the inequality as

$$ab = \exp\left(\frac{1}{p}\ln a^p + \frac{1}{q}\ln b^q\right).$$

Since the exponential function is strictly positive, we apply Jensen's inequality,

$$\exp\left(\frac{1}{p}\ln a^p + \frac{1}{q}\ln b^q\right) \le \frac{1}{p}\exp\left(\ln a^p\right) + \frac{1}{q}\exp\left(\ln b^q\right).$$

Apply the substitution and simplification, we get

$$ab \le \frac{a^p}{p} + \frac{b^q}{q}.$$

□

Theorem 1.9 (Hölder's inequality)**.** *Let $p, q \in (1, +\infty)$ be strictly positive real numbers such that $\frac{1}{p} + \frac{1}{q} = 1$. Two sequences of positive real numbers $a_i, b_i \in \mathbb{R}^+, \forall i \in [1, n] \cap \mathbb{N}$ satisfies*

$$\sum_{i=1}^{n} a_i b_i \le \left(\sum_{i=1}^{n} a_i^p\right)^{\frac{1}{p}} \left(\sum_{i=1}^{n} b_i^q\right)^{\frac{1}{q}}. \tag{1.21}$$

Equality occurs if and only if $(\forall i, a_i = 0)$ or $(\forall i, b_i = 0)$ or $(\exists c_1, c_2 \in \mathbb{R}_*^+ \forall i, c_1 a_i^p = c_2 b_i^q)$.

Proof. If $\forall i, a_i = 0$ or $\forall i, b_i = 0$, both sides of the inequality are zero, and equality occurs trivially. So we consider the case where the right hand side of the inequality is greater than 0. Then it is equivalent to prove

$$\frac{\sum_{i=1}^{n} a_i b_i}{(\sum_{i=1}^{n} a_i^p)^{\frac{1}{p}} (\sum_{i=1}^{n} b_i^q)^{\frac{1}{q}}} \le 1.$$

Then, we define

$$x_j \coloneqq \frac{a_j}{(\sum_{i=1}^{n} a_i^p)^{\frac{1}{p}}}, \quad y_j \coloneqq \frac{b_j}{(\sum_{i=1}^{n} b_i^q)^{\frac{1}{q}}}, \quad \forall j \in [1, n] \cap \mathbb{N}.$$

Then it is equivalent to prove

$$\sum_{j=1}^{n} x_j y_j \le 1.$$

Apply Young's inequality, we have

$$x_j y_j \le \frac{x_j^p}{p} + \frac{y_j^q}{q}, \quad \forall j \in [1, n] \cap \mathbb{N}.$$

So their sum satisfies

$$\sum_{j=1}^{n} x_j y_j \le \sum_{j=1}^{n} \left(\frac{x_j^p}{p} + \frac{y_j^q}{q}\right) = \frac{1}{p}\sum_{j=1}^{n} x_j^p + \frac{1}{q}\sum_{j=1}^{n} y_j^q.$$

According to the definition,

$$\sum_{j=1}^{n} x_j^p = \frac{\sum_{j=1}^{n} a_j^p}{\sum_{i=1}^{n} a_i^p} = 1.$$

Similarly,

$$\sum_{j=1}^{n} y_j^q = \frac{\sum_{j=1}^{n} b_j^q}{\sum_{i=1}^{n} b_i^q} = 1.$$

Therefore,

$$\frac{\sum_{i=1}^{n} a_i b_i}{(\sum_{i=1}^{n} a_i^p)^{\frac{1}{p}} (\sum_{i=1}^{n} b_i^q)^{\frac{1}{q}}} = \sum_{j=1}^{n} x_j y_j \leq \frac{1}{p} + \frac{1}{q} = 1.$$

[**utx**] □

Theorem 1.10 (Minkowski's inequality)**.** *Let $p \in (1, +\infty)$ be strictly positive real number. Two sequences of positive real numbers $a_i, b_i \in \mathbb{R}^+, \forall i \in [1, n] \cap \mathbb{N}$ satisfies*

$$\left[\sum_{i=1}^{n} (a_i + b_i)^p\right]^{\frac{1}{p}} \leq \left(\sum_{i=1}^{n} a_i^p\right)^{\frac{1}{p}} + \left(\sum_{i=1}^{n} b_i^p\right)^{\frac{1}{p}}. \tag{1.22}$$

Proof. Define

$$q \coloneqq \frac{p}{p-1}$$

such that $\frac{1}{p} + \frac{1}{q} = 1$ is satisfied. It follows from the Hölder's inequality that

$$\begin{aligned}
\sum_{i=1}^{n} (a_i + b_i)^p &= \sum_{i=1}^{n} a_i (a_i + b_i)^{p-1} + \sum_{i=1}^{n} b_i (a_i + b_i)^{p-1} \\
&\leq \left(\sum_{i=1}^{n} a_i^p\right)^{\frac{1}{p}} \left[\sum_{i=1}^{n} (a_i + b_i)^p\right]^{\frac{1}{q}} + \left(\sum_{i=1}^{n} b_i^p\right)^{\frac{1}{p}} \left[\sum_{i=1}^{n} (a_i + b_i)^p\right]^{\frac{1}{q}} \\
&= \left[\left(\sum_{i=1}^{n} a_i^p\right)^{\frac{1}{p}} + \left(\sum_{i=1}^{n} b_i^p\right)^{\frac{1}{p}}\right] \left[\sum_{i=1}^{n} (a_i + b_i)^p\right]^{\frac{1}{q}}
\end{aligned}$$

If $\sum_{i=1}^{n} (a_i + b_i)^p = 0$, the equality occurs trivially. So we consider the case where $\sum_{i=1}^{n} (a_i + b_i)^p > 0$, where we can divide both sides by $[\sum_{i=1}^{n} (a_i + b_i)^p]^{1/q}$ without changing the direction of the inequality. Notice that $1 - \frac{1}{q} = \frac{1}{p}$, we get

$$\left[\sum_{i=1}^{n} (a_i + b_i)^p\right]^{\frac{1}{p}} \leq \left(\sum_{i=1}^{n} a_i^p\right)^{\frac{1}{p}} + \left(\sum_{i=1}^{n} b_i^p\right)^{\frac{1}{p}}.$$

[**utx**] □

With Minkowski's inequality, it is easy to prove (1.19), and thus equation (1.18) defines a family of matrix norm, since the triangle inequality required by the norm is a special case of Minkowski's inequality.

Definition 1.9 (Induced matrix norm)**.** The *matrix norm induced by* N_i with $i \in \mathbb{N}^*$ for real matrices $||\cdot||_i : \mathbb{R}^{m\times n} \to \mathbb{R}^+$ is defined to be

$$||A||_i := \sup_{\substack{x\neq 0\\ x\in\mathbb{R}^n}} \frac{N_i(Ax)}{N_i(x)} = \sup_{\substack{N_i(x)=1\\ x\in\mathbb{R}^n}} N_i(Ax) \tag{1.23}$$

Definition 1.10 (Condition number)**.** If a matrix $A \in \mathbb{R}^{n\times n}$ is invertible, then we can define the *condition number* of A denoted by $\kappa_i(A)$ as

$$\kappa_i(A) := ||A||_i||A^{-1}||_i. \tag{1.24}$$

The condition number indicates the invertibility of a matrix. Matrices with small condition number are called well-conditioned, whereas those with a large condition number ill-conditioned. In particular, they evaluate the quality of the Cholesky decomposition on matrices. [**tam39**]

4 Numerical Algorithms

4.1 Cholesky Decomposition

By Theorem 1.6, we can write algorithms to implement Cholesky decomposition.

Input: Positive definite square matrix $A \in \mathbb{R}^{n\times n}$
Output: Lower triangular matrix L such that $LL^T = A$

```
for k=1:n-1 do
    A(k,k) = sqrt(A(k,k));
    for j=k+1:n do
        A(j,k) = A(j,k)/A(k,k);
    end
    for j=k+1:n do
        for m=j:n do
            A(m,j) = A(m,j) - A(m,k) · A(j,k);
        end
    end
end
A(n,n) = sqrt(A(n,n));
L = (a_ij)_{j≤i};
```

Algorithm 1.1: Cholesky Decomposition[**tam37**]

4.2 Complexity Analysis

The Cholesky decomposition algorithm has time complexity $O(n^3)$, since there are three contained loop, each of which are of complexity $O(n)$.

A Appendix

A.1 MATLAB™ Code for Cholesky Decomposition

```
function [L] = Cholesky(A)
[n,n]=size(A);
for k=1:n-1
    A(k,k)=sqrt(A(k,k));
    A(k+1:n,k)=A(k+1:n,k)/A(k,k);
    for j=k+1:n,
        A(j:n,j)=A(j:n,j)-A(j:n,k)*A(j,k);
    end
end
A(n,n)=sqrt(A(n,n));
L = tril(A);
return
end
```

Project 2

Multidimensional Root-finding and Optimization

Guangting Yu, University of Michigan - Shanghai Jiao Tong University Joint Institute, Minhang, Shanghai, 200135, China

Abstract

This paper explores numerical methods in higher dimensions, including the fixed-point iteration, Newton's method and secant method. A robust schema which involves the combination of these methods is introduced to find roots of higher dimensional functions. Applications to system of equations in $\mathbb{R}^2$ and $\mathbb{R}^3$ are studied to illustrate these methods. An application to non-linear regression is optimized by using the schema. The performances such as convergence and sensitivity with respect to the initial guess of these methods are also studied and compared.

1 Introduction

Finding roots of functions in higher dimensions is similar to the methods in one dimension. In this paper, our goal is to solve the following problems, including two systems of equations and a non-linear optimization from the non-linear regression model.

1.1 System of Equations

The first system is in $\mathbb{R}^2$.

$$\begin{aligned} x_1^2 - x_2 &= 0 \\ x_1^2 + x_2^2 &= 1 \end{aligned} \tag{2.1}$$

The second system is in $\mathbb{R}^3$

$$\begin{aligned} 3x_1 - \cos(x_2 x_3) &= \frac{1}{2} \\ x_1^2 - 625x_2^2 &= \frac{1}{4} \\ \exp(-x_1 x_2) + 20x_3 &= 1 - \frac{10}{3}\pi \end{aligned} \tag{2.2}$$

1.2 Non-linear Regression

The following function denote some physical relationship between the variables y and x.

$$y = \alpha \exp\left(\frac{\beta}{x+\gamma}\right),$$

where α, β, γ are unknown parameters in $\mathbb{R}$. Suppose the following data are obtained from an experiment, and we want to find the best estimate for α, β, γ by minimizing the sum of square errors

$$E = \sum_{i=1}^{n} \left[y_i - \alpha \exp\left(\frac{\beta}{x_i+\gamma}\right) \right]^2. \tag{2.3}$$

The data is given in

x	50	55	60	65	70	75	80	85
y	34780	28610	23650	19630	16370	13720	11540	9744
x	90	95	100	105	110	115	120	125
y	8261	7030	6005	5147	4427	3820	3307	2872

Table 2.1: Data for non-linear regression

2 Analysis

2.1 Reduction of Dimensions

The reduction of dimensions is implemented by the idea to apply substitutions to reduce the number of equations and unknowns. The best scenario is that the system of equations can be reduced to only one equation with only one unknown. Then the problem can be solved by the methods in one dimension.

Specifically, the system in $\mathbb{R}^2$ and the system in $\mathbb{R}^3$ can both be reduced to one dimension. In the system in $\mathbb{R}^2$, we apply the substitution

$$x_2 = x_1^2,$$

and the system is reduced to

$$x_1^4 + x_1^2 - 1 = 0, \tag{2.4}$$

which can be solved algebraically. The real roots are

$$x_1 = \pm\sqrt{\frac{\sqrt{5}-1}{2}}, \quad x_2 = \frac{\sqrt{5}-1}{2}.$$

Two other roots exist, but x_1 is a complex number, so we don't list them here, since the numerical methods we discuss cannot find them. In the system in $\mathbb{R}^3$, we apply the trigonometric substitution

$$\begin{aligned} x_1 &= \frac{1}{2}\sec t \\ x_2 &= \frac{1}{50}\tan t \\ x_3 &= \frac{1}{20} - \frac{\pi}{6} - \frac{1}{20}\exp\left(-\frac{\sec t \tan t}{100}\right) \end{aligned}.$$

Then the system of equations is reduced to

$$3\sec t - 2\cos\left\{\frac{\tan t}{1000}\left[1 - \frac{10}{3}\pi - \exp\left(-\frac{\sec t \tan t}{100}\right)\right]\right\} - 1 = 0. \tag{2.5}$$

Furthermore, this is periodic with period 2π, so we just need to find roots in the interval $[-\pi, \pi]$.

2.2 Generalization of Methods in Single Dimension

In general, the reduction of dimensions does not work, since there are equations that cannot be transformed to express a variable in explicit form. Thus, the numerical methods are still needed to solve the higher dimensional system of equations. Here, we just generalize three methods to higher dimensions: fixed-point iteration, Newton's method, and secant method.

Fixed-point iteration

Suppose we have the function

$$F(x) = G(x) - x, \quad G : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}^n,$$

where $F(x) = 0$ is the system of functions to solve. The the fixed-point iteration

$$x_0 \in \mathcal{D}, \quad x_k = G(x_{k-1}), \quad k = 1, 2, 3, \cdots$$

converges to the root x^* where $F(x^*) = 0$ if G has the Lipschitz constant $0 \leq c < 1$. Here the Lipschitz constant is defined to be a constant that satisfies

$$||J_G(x_1) - J_G(x_2)|| \leq c||x_1 - x_2|| \quad \forall x_1, x_2 \in \mathcal{D},$$

where J_G denotes the Jacobian of G.

Newton's method

Suppose we have the system of functions $F(x) = 0$ to solve, where $F : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}^n$. Then if the series of vectors with $x_0 \in \mathcal{D}$,

$$x_{k+1} = x_k - [J_F(x_k)]^{-1} F(x_k), \quad k = 1, 2, 3, \cdots$$

converges, then it converges to a root x^* such that $F(x^*) = 0$. Again, J_F denotes the Jacobian of F.

Secant method

The secant method in higher dimension is more complicated than the one dimension case. Suppose we have the system of functions $F(x) = 0$ to solve, where $F : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}^n$. The secant method is the following iteration, with the initial guesses x_0 and x_1 and the initial matrix A_1 defined to be

$$A_1 \coloneqq J_F(x_0) + \frac{[F(x_1) - F(x_0) - J_F(x_0)(x_1 - x_0)]\,(x_1 - x_0)^T}{||x_1 - x_0||_2^2}$$

And the iteration for $k = 1, 2, 3, \cdots$ is

$$\begin{aligned} s_k &= x_k - x_{k-1} \\ y_k &= F(x_k) - F(x_{k-1}) \\ A_k &= A_{k-1} + \frac{y_k - A_{k-1} s_k}{||s_k||_2^2} s_k^T \\ w_k &= A_k^{-1} F(x_k) \\ x_{k+1} &= x_k - w_k. \end{aligned}$$

2.3 Transforming Optimization to System of Equations

If the function $f : \mathbb{R}^n \to \mathbb{R}$ is sufficiently smooth and convex in the absence of any constraint, we could turn the optimization problem into a root-finding problem by finding x^* such that

$$\nabla f(x^*) = 0.$$

In the non-linear regression, minimizing the sum of square errors E is equivalent to finding the root of the system of equations

$$\begin{aligned} \partial_\alpha E &= 0 \\ \partial_\beta E &= 0 \\ \partial_\gamma E &= 0 \end{aligned} \tag{2.6}$$

2.4 Combination and Construction of Robust Method

In one dimension case, Newton's methods has the problem that the initial guess x_0 is critical to the convergence of the method. By contrast, the fixed-point iteration is sure to converge

on the domain where the Lipschitz constant $c < 1$, but there is no guarantee that the given system has such a property. So transformation such as dividing by a constant can create a new function with smaller Lipschitz constant without changing the root, but determining this constant is also equivalent to finding the supreme of the Jacobian matrix, which is an optimization question and the dimension does not get reduced.

Thus, in order to find a root of system of equations, the attempt by all three methods should be tried.

3 Results

3.1 Solving System of Equations

The solution to (2.5) is easy to obtain. Start with initial guess $t = 0$ and the solution happens to be found. Thus the solution to the system in $\mathbb{R}^3$ is

$$x^* = \begin{pmatrix} \frac{1}{2} \\ 0 \\ -\frac{\pi}{6} \end{pmatrix}.$$

And the solution is also unique.

Fixed-point Iteration

Apply fixed-point iteration to reduced system near a root,

$$G\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1^2 + x_1 - x_2 \\ x_1^2 + x_2^2 + x_2 - 1 \end{pmatrix},$$

we have the log in Table 2.2.

Obviously, it does not converge. This is a direct result from the norm of the Jacobian matrix $||J_G||_2 = 3.05847254037959 > 1$, which means the Lipschitz constant is too large for the fixed-point iteration to converge.

Newton's method

Apply Newton's method to reduced system near a root,

$$F\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1^2 - x_2 \\ x_1^2 + x_2^2 - 1 \end{pmatrix},$$

with its Jacobian matrix

$$J_F\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2x_1 & -1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

we have the log in Table 2.3a and 2.3b, which correspond to the root with positive x_1 and the root with negative x_1. Since we have already obtained the accurate solution, we can compare the numerical results with and and find the error.

n	$\hat{x}_1$	$\hat{x}_2$
1	0.786000000000000	0.618000000000000
2	0.785796000000000	0.617720000000000
3	0.785551353616000	0.616773352016000
4	0.785868930767931	0.614273648940982
5	0.789185258173280	0.609195741070583
6	0.802802888820723	0.603128563727145
7	0.844166803392477	0.611385106409613
8	0.945399288932737	0.697794446698978
9	1.14138465774828	1.07849135205744
10	1.36565224263401	2.54439388546328
11	0.686264404982040	9.88334017765749
12	-8.72611693913009	107.034712078502
13	-39.6157121822585	11638.6094186416
14	-10108.8204791163	135470436.213763
15	-33292293.5552410	1.83522393256060e+16
16	-1.72438625487299e+16	3.36804688264319e+32
17	-3.94538926648279e+31	1.13437398036825e+65
18	-1.11880788390417e+65	1.28680432733650e+130
19	-3.50732462503746e+128	1.65586537685195e+260
20	-1.65463524424941e+260	Inf

Table 2.2: Log of fixed-point iteration on the reduced system

n	$\hat{x}_1$	$\hat{x}_2$	Error in $\hat{x}_1$	Error in $\hat{x}_2$
1	10	1	9.21384862224258	0.381966011250105
2	5.03333333333333	0.666666666666667	4.24718195557591	0.0486326779167718
3	2.57816146326080	0.619047619047619	1.79201008550338	0.00101363029772439
4	1.40894026266226	0.618034447821682	0.622788884904833	4.59071787028975e-07
5	0.923795962641458	0.618033988749989	0.137644584884035	9.41469124882133e-14
6	0.796405823822399	0.618033988749895	0.0102544460649757	1.11022302462516e-16
7	0.786217395396267	0.618033988749895	6.60176388432854e-05	0
8	0.786151380529131	0.618033988749895	2.77170719709119e-09	0
9	0.786151377757423	0.618033988749895	0	0

(a) Log of Newton's method on the reduced system near the positive root

n	$\hat{x}_1$	$\hat{x}_2$	Error in $\hat{x}_1$	Error in $\hat{x}_2$
1	-10	2	9.21384862224258	1.38196601125011
2	-5.05000000000000	1	4.26384862224258	0.381966011250105
3	-2.59100660066007	0.666666666666667	1.80485522290264	0.0486326779167718
4	-1.41496413437228	0.619047619047619	0.628812756614854	0.00101363029772406
5	-0.925874333395720	0.618034447821682	0.139722955638297	4.59071786917953e-07
6	-0.796694117537667	0.618033988749989	0.0105427397802439	9.42579347906758e-14
7	-0.786221134367663	0.618033988749895	6.97566102392244e-05	0
8	-0.786151380851963	0.618033988749895	3.09453951352623e-09	0
9	-0.786151377757423	0.618033988749895	0	0

(b) Log of Newton's method on the reduced system near the negative root

Table 2.3: Log of Newton's method on the reduced system

Secant method

Apply secant method to reduced system near a root, we have the log in Table 2.4a and 2.4b, which correspond to the root with positive x_1 and the root with negative x_1.

3.2 Solving Non-Linear Regression from System of Equations

The system (2.6) has a very large Lipschitz constant, so the fixed-point iteration does not converge. And the initial guess for Newton's method and secant method that leads to the convergence is hard to obtain. Thus, we first apply brute force method to scan a subset of $\mathbb{R}^3$. The best guess is near

$$\alpha = 1.5, \quad \beta = 2400, \quad \gamma = 200$$

Then apply the secant method with this initial guess with some variation. Although the secant method still does not converge, it produces some results that are better than the initial guess. Thus, by scanning the log of the secant method, we find some better initial guesses for the secant method. Then apply the secant method again, better results still occur in the log, although the secant method still does not converge. Repeat this processes for many times, we can get the following results, which leads to the decreasing of the sum of square errors.

So the least square estimator is

$$\begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \\ \hat{\gamma} \end{pmatrix} = \begin{pmatrix} 1.53306388436911 \\ 2390.91074802608 \\ 188.153946322168 \end{pmatrix}. \tag{2.7}$$

4 Discussion

The solutions to the system in $\mathbb{R}^2$ and the system in $\mathbb{R}^3$ are very accurate, and here Newton's method works well because the dimension is reduced to one. And we can see Newton's method converges faster than secant method.

However, in solving the non-linear regression, Newton's method is too sensitive to the initial value. So the secant method is more useful. If we plug in the least square estimator in 2.7 and plot the residual, we get Figure 2.1.

This has nearly the same shape as

$$r = 100(x - 50) \exp\left(-\frac{x - 50}{12}\right),$$

which is totally different from the regression model

$$y = \alpha \exp\left(\frac{\beta}{x + \gamma}\right).$$

Thus, we can accept this residual.

n	$\hat{x}_1$	$\hat{x}_2$	Error in $\hat{x}_1$	Error in $\hat{x}_2$
1	0	0	0.786151377757423	0.618033988749895
2	1	1	0.213848622242577	0.381966011250105
3	-4.44089209850063e-16	0	0.786151377757424	0.618033988749895
4	0.500000000000000	0.500000000000000	0.286151377757424	0.118033988749895
5	1.50000000000000	0.500000000000000	0.713848622242578	0.118033988749895
6	0.699999999999999	0.700000000000000	0.0861513777574243	0.0819660112501053
7	0.743589743589743	0.561965811965812	0.0425616341676802	0.0560681767840830
8	0.769144654957387	0.625648271932884	0.0170067228000363	0.00761428318298951
9	0.782820422528377	0.620602567694648	0.00333095522904614	0.00256857894475282
10	0.786153389406297	0.618018179176911	2.01164887381200e-06	1.58095729836383e-05
11	0.786151765747616	0.618034153917164	3.87990192751708e-07	1.65167268750821e-07
12	0.786151367024367	0.618033984293740	1.07330563325903e-08	4.45615500055396e-09
13	0.786151377758197	0.618033988750217	7.73936470466197e-13	3.21631610233908e-13
14	0.786151377757423	0.618033988749895	0	0

(a) Log of secant method on the reduced system near the positive root

n	$\hat{x}_1$	$\hat{x}_2$	Error in $\hat{x}_1$	Error in $\hat{x}_2$
1	0	0	0.786151377757423	0.618033988749895
2	-1	-1	0.213848622242577	1.61803398874990
3	4.44089209850063e-16	0	0.786151377757424	0.618033988749895
4	-1.50000000000000	0.500000000000000	0.713848622242577	0.118033988749895
5	-1.04545454545455	1.40909090909091	0.259303167697122	0.791056920341014
6	-0.257542512342293	0.255896873285793	0.528608865415130	0.362137115464102
7	-0.569665310647485	0.530595677775853	0.216486067109938	0.0874383109740420
8	-0.920368915521390	0.702578943370071	0.134217537763967	0.0845449546201756
9	-0.763353387213641	0.617164988992850	0.0227979905437824	0.000868999757044686
10	-0.783613611581945	0.620001304298971	0.00253776617547874	0.00196731554907614
11	-0.786044731382020	0.618570694693166	0.000106646375403474	0.000536705943271265
12	-0.786127124995423	0.618117862137955	2.42527620001320e-05	8.38733880598186e-05
13	-0.786151822880213	0.618032593821515	4.45122789183294e-07	1.39492837991639e-06
14	-0.786151375839576	0.618033994871750	1.91784688130525e-09	6.12185457882219e-09
15	-0.786151377755821	0.618033988755150	1.60205182453410e-12	5.25468557555087e-12
16	-0.786151377757421	0.618033988749901	1.99840144432528e-15	6.43929354282591e-15
17	-0.786151377757423	0.618033988749895	0	1.11022302462516e-16

(b) Log of secant method on the reduced system near the negative root

Table 2.4: Log of secant method on the reduced system

$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$	E
1.5	2395	188	697959.028219363
1.51	2393	188	697670.840678421
1.53	2390	188	695776.006172582
1.53	2390	188	695776.006172582
1.53484769689784	2390.21685511258	188.112408803811	695578.502806576
1.53445741417831	2390.37238318836	188.121169919855	695460.800868128
1.53484769689784	2390.21685511258	188.112408803811	695578.502806576
1.53445741417831	2390.37238318836	188.121169919855	695460.800868135
1.53442660258461	2390.39392709258	188.123741057149	695459.210678605
1.53403657463910	2390.54569210021	188.132250404586	695343.574246659
1.53442660258461	2390.39392709258	188.123741057149	695459.210678599
1.53403657463910	2390.54569210021	188.132250404586	695343.574246657
1.53395060236349	2390.57619051588	188.134672703047	695330.058657119
1.53355930237206	2390.72624233791	188.143059536112	695215.380065222
1.53442660258461	2390.39392709258	188.123741057149	695459.210678599
1.53403657463910	2390.54569210021	188.132250404586	695343.574246657
1.53395060236349	2390.57619051588	188.134672703047	695330.058657119
1.53355930237206	2390.72624233791	188.143059536112	695215.380065222
1.53355930237206	2390.72624233791	188.143059536112	695215.380065222
1.53395060236349	2390.57619051588	188.134672703047	695330.058657126
1.53355930237206	2390.72624233791	188.143059536112	695215.380065222
1.53345656414647	2390.76171934672	188.145635434856	695197.249358202
1.53306388436911	2390.91074802608	188.153946322168	695083.149638547
1.53306388436911	2390.91074802608	188.153946322168	695083.149638545

Table 2.5: Log of the robust combination of secant method and brute-force method

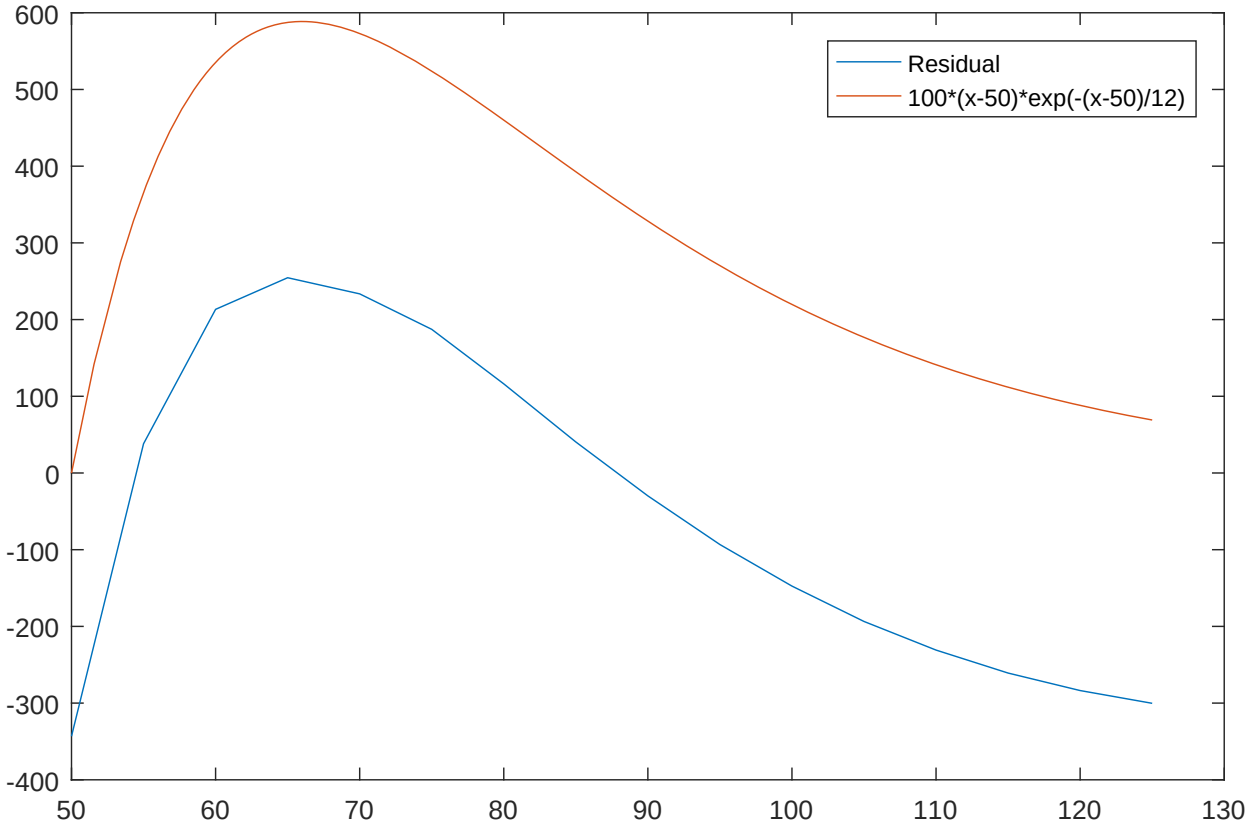

Figure 2.1: The residual of the regression with the obtained estimators

5 Conclusions

When fixed-point iteration does not converge, secant method is more stable to find some good results, although it converges slower than Newton's method. Furthermore, when all the three numerical methods do not converge, the brute-force method is a reliable alternative, and scanning the log of the output is also a good method, although the are slow and attempt to solve by a computer science approach instead of a numerical method approach. The results from brute-force method and scanning the log can be good initial guess for Newton's method and secant method.

A Appendix

A.1 MATLAB™ Function Implemention of the Numerical Methods

Fixed-point iteration

```
function [ P ] = fixedpoint( g,p0,tol,max1 )

P=zeros(max(size(p0)),max1);
P(:,1)= p0;

for k=2:max1
        P(:,k)=g(P(:,k-1));
    dif=P(:,k)-P(:,k-1);
        err=sqrt(dif'*dif);
    if (err<tol)
        break;
    end
end

P=P(:,1:k);

end
```

Newton's method

```
function [ X ] = newton_Highdim( f,Jf,p0,delta,epsilon,max1
   )

X=[];
for k=1:max1
        X=[X p0];
        p1=p0-Jf(p0)\f(p0);
        err=(p1-p0)'*(p1-p0);
        p0=p1;
        y=f(p0);
        if (err<delta)||(y'*y<epsilon)
                break
        end
end

end
```

Secant method

```
function [ X ] = secant_Highdim(f,Jf,x0,x1,delta,epsilon,
   max1)

X=[];


s=x1-x0;
y=f(x1)-f(x0);
A=Jf(x0)+(y-Jf(x0)*s)*s'/norm(s,2)^2;
Ainv=inv(A);
w=A\y;
X=[X x0];
x0=x1;
x1=x1-w;


for k=2:max1
    if(s'*s<delta)||(y'*y<epsilon)
        break
    end
    s=x1-x0;
    y=f(x1)-f(x0);
    A=A-(y-A*s)*s'/norm(s,2)^2;
    Ainv=Ainv+(s-Ainv*y)*s'*Ainv/(s'*Ainv*y);
    w=Ainv*f(x1);
    X=[X x0];
    x0=x1;
    x1=x1-w;
end

X=[X x1];

end
```

A.2 MATLAB™ log of the project

```
%%%%%% Task 1

% Fixed-point iteration
g=@(x)[x(1)^2+x(1)-x(2);x(1)^2+x(2)^2+x(2)-1];
P=fixedpoint(g,[0.786;0.618],1e-6,100);


% Why fixed-point fail
Jg=[sqrt(2*(sqrt(5)-1))+1, -1; sqrt(2*(sqrt(5)-1)), sqrt(5)
   ];
norm(Jg,2)

% Newton's method for higher dimension
f=@(x) [x(1)^2-x(2);x(1)^2+x(2)^2-1];
Jf=@(x) [ 2*x(1), -1; 2*x(1), 2*x(2)];
X1=newton_Highdim( f,Jf,[10;1],1e-300,1e-300,10);
X1=newton_Highdim( f,Jf,[-10;2],1e-300,1e-300,10);


% Secant method for higher dimension
X=secant_Highdim(f,Jf,[0;0],[1;1],1e-300,1e-300,20);
X=secant_Highdim(f,Jf,[0;0],[-1;-1],1e-30,1e-30,30);
[abs(X(1,:)+sqrt((sqrt(5)-1)/2))' abs(X(2,:)-(sqrt(5)-1)/2)
   ']

% loglog(abs(X(1,:)-sqrt((sqrt(5)-1)/2)),abs(X(2,:)-(sqrt
   (5)-1)/2))


%%%%%% Task 2
f=@(t) 3*sec(t)-2*cos(tan(t)/1000*(1-10*pi/3-exp(-sec(t)*
   tan(t)/100)))-1;
df=@(t) 3*tan(t)*sec(t)+(sec(t)*exp(-(1/100)*tan(t)*sec(t))
   *(3*tan(t)^3+3*tan(t)*sec(t)^2-100*sec(t)*((10*pi-3)*exp
   (1/100*tan(t)*sec(t))+3))*sin((tan(t)*(-exp(-(1/100)*tan
   (t)*sec(t))+1-(10*pi)/3))/1000))/150000;
newton(f,df,1,1e-16,1e-16,10)
%
%
% fplot(@(t) df(t),[-1,1])

% fplot(@(t) 3*sec(t)-2*cos(tan(t)/1000*(1-10*pi/3-exp(-sec
   (t)*tan(t)/100)))-1,[-1.3,1.3])
```

```
% axis([-1.3 1.3 -2 1]);


% syms t
% g=sec(t)-2*cos(tan(t)/1000*(1-10*pi/3-exp(-sec(t)*tan(t)
   /100)));
% dg=@(t) sin(t)/cos(t)^2 + 2*sin((tan(t)*(exp(-tan(t)
   /(100*cos(t))) ...
%     + 5332248173269055/562949953421312))/1000)*(((exp(-
   tan(t)/(100*cos(t))) ...
%     + 5332248173269055/562949953421312)*(tan(t)^2 + 1))
   /1000 -...
%     (exp(-tan(t)/(100*cos(t)))*tan(t)*((tan(t)^2 + 1)
   /(100*cos(t)) +...
%     (sin(t)*tan(t))/(100*cos(t)^2)))/1000);




% fplot(@(t) sin(t)/cos(t)^2 + 2*sin((tan(t)*(exp(-tan(t)
   /(100*cos(t))) ...
%     + 5332248173269055/562949953421312))/1000)*(((exp(-
   tan(t)/(100*cos(t))) ...
%     + 5332248173269055/562949953421312)*(tan(t)^2 + 1))
   /1000 -...
%     (exp(-tan(t)/(100*cos(t)))*tan(t)*((tan(t)^2 + 1)
   /(100*cos(t)) +...
%     (sin(t)*tan(t))/(100*cos(t)^2)))/1000),[-1.3,1.3])
% axis([-1.3 1.3 -2 1]);
% hold on
% fplot(@(t)  tan(t)*sec(t)+(sec(t)*exp(-(1/100)*tan(t)*sec
   (t))*(3*tan(t)^3+3*tan(t)*sec(t)^2-100*sec(t)*((10*pi-3)
   *exp(1/100*tan(t)*sec(t))+3))*sin((tan(t)*(-exp(-(1/100)
   *tan(t)*sec(t))+1-(10*pi)/3))/1000))/150000,[-1.3,1.3])
% hold off



%%%%%% Task 3




x=50:5:125;
```

```
y=[34780 28610 23650 19630 16370 13720 11540 9744 8261 7030
    6005 5147 4427 3820 3307 2872];

syms a b c;
et=a*exp(b./(x+c))'; % estimate
E=((y'-et)'*(y'-et)); % error
% E=a^2*(et'*et)-2*a*(y*et)+y*y';

double(subs(E,{a,b,c},{1.53,2390,288}))
temp=7e05;

cand=[];
for i=1.50:0.01:1.53
        for j=2350:2400
                for k=180:195
                        if double(subs(E,{a,b,c},{i,j,k}))<
                           temp
                                temp=double(subs(E,{a,b,c
                                   },{i,j,k}))
                                cand=[cand [i;j;k;temp]];
                                double(subs(g,{a,b,c},{i,j,
                                   k}))
                        end
                end
        end
end

dy=y-exp(2390.91074802608./(x+188.153946322168))
   *1.53306388436911;
set(gca(), 'LooseInset', get(gca(), 'TightInset'));
plot(x,(dy))
hold on
fplot(@(x) 100*(x-50)*exp(-(x-50)/16),[50,125])
hold off

legend('Residual','100*(x-50)*exp(-(x-50)/12)','Location','
   NorthEast')



% plot(x+5501464534.50766,2245360093.83860./log(y
   /8265.91321007004))
% plot(x,log(y/1.6),x,2500./(x+200))

% hold on
```

```
% for i=1:10
%     %plot(x./25,100./(log(y)+i));
%     %plot(x,exp(2000./(x+200)))
% end
% hold off









g=gradient(E,[a,b,c]);
H=hessian(E,[a,b,c]);


%Secant method


x0=[1.53395060236349;2390.57619051588;188.134672703047];
x1=[1.53355930237206;2390.72624233791;188.143059536112];

X=[];


s=x1-x0;
z=double(subs(g,{a,b,c},{x1(1),x1(2),x1(3)})-subs(g,{a,b,c
   },{x0(1),x0(2),x0(3)}));
Jf=double(subs(H,{a,b,c},{x0(1),x0(2),x0(3)}));
A=Jf+(z-Jf*s)*s'/norm(s,2)^2;
w=A\z;
X=[X x0];
x0=x1;
x1=x1-w;


for k=1:100
    if(s'*s<1e-10)||(z'*z<1e-10)
        break
    end
    s=x1-x0;
    z=double(subs(g,{a,b,c},{x1(1),x1(2),x1(3)})-subs(g,{a,
       b,c},{x0(1),x0(2),x0(3)}));
```

```
    Jf=double(subs(H,{a,b,c},{x0(1),x0(2),x0(3)}));
    A=Jf+(z-Jf*s)*s'/norm(s,2)^2;
    w=A\double(subs(g,{a,b,c},{x1(1),x1(2),x1(3)}));
    X=[X x0];
    x0=x1;
    x1=x1-w;
end

X=[X x1];









Newton Always Fail
p0=[1.53306388436911;2390.91074802608;188.153946322168];
for k=1:100
        X=[X p0];
        p1=p0-double(subs(H,{a,b,c},{p0(1),p0(2),p0(3)}))\
           double(subs(g,{a,b,c},{p0(1),p0(2),p0(3)}));
        err=(p1-p0)'*(p1-p0);
        p0=p1;
    y=double(subs(g,{a,b,c},{p0(1),p0(2),p0(3)}));
        if (err<1e-16||y'*y<1e-16)
                break
        end
end


for i=1:27
    if double(subs(E,{a,b,c},{X(1,i),X(2,i),X(3,i)}))<temp
        temp=double(subs(E,{a,b,c},{X(1,i),X(2,i),X(3,i)}))
        cand=[cand [X(1,i);X(2,i);X(3,i);temp]];
        double(subs(g,{a,b,c},{X(1,i),X(2,i),X(3,i)}))
    end
end
```

Part IV

External Resources

External Resource 1

MATLAB™ Codes

1 Interpolation

1.1 Lagrange polynomial interpolation

```
function [p]=lagrinterpol(x,y,z)
%-------------------------------------------------
%LAGRINTERPOL   Lagrange polynomial interpolation
% Inputs
%   x   vector of interpolation nodes
%   y   vector of funtion values at x
%   z   vector of points at which the interpolating
   polynomial need to be
%       evalauted
% Outpus
%   p   vector of polynomial values at z
%% NUMERICAL METHODS: MATLAB Programs, Mathews & Fink 2004
%-------------------------------------------------
[m n] = size(y);
for j = 1:m
    a (:,1) = y (j,:)';
    for i = 2:n
        a (i:n,i) = ( a(i:n,i-1)-a(i-1,i-1) )./(x(i:n)-x(i
           -1))';
    end
    p(j,:) = a(n,n).*(z-x(n-1)) + a(n-1,n-1);
    for i = 2:n-1
        p(j,:) = p(j,:).*(z-x(n-i))+a(n-i,n-i);
    end
end
return
```

Coefficients of Lagrange polynomial interpolation

```
function [C, C1] = lagrinterpolde(X,Y)
%-------------------------------------------------------
%lagrinterpolde   Coefficients of Lagrange polynomial
   interpolation
%                and coefficients of the first derivative
   of it
% Inputs
%   X   vector of interpolation nodes
%   Y   vector of funtion values at X
% Outpus
%   C   vector of coefficients of p
%   C1  vector of coefficients of p'
%% NUMERICAL METHODS: MATLAB Programs, Mathews & Fink 2004
%-------------------------------------------------------
w = length(X);
n = w - 1;
L = zeros(w,w);

for k = 1:n+1
    V = 1;
    for j = 1:n+1
        if k~=j
            V = conv(V, poly(X(j)))/(X(k)-X(j));
        end
    end
    L(k,:) = V;
end

C = Y*L;
C1 = C .* (n:-1:0);
C1 = C1(1:n);
```

1.2 Newton polynomial interpolation

```
function [C,D] = newpoly(X,Y)
%----------------------------------------
%newpoly   Construction of the collocation polynomial.
%          The method is Newton interpolation.
% Inputs
%   X   vector of abscissas
%   Y   vector of ordinates
% Return
```

```
%   C   coefficient list for the Newton polynomial
%   D   divided difference table
%
%% NUMERICAL METHODS: MATLAB Programs, Mathews & Fink 2004
%----------------------------------------

n = length(X);
D = zeros(n,n);
D(:,1) = Y';
for j=2:n,
  for k=j:n,
      D(k,j) = (D(k,j-1)-D(k-1,j-1))/(X(k)-X(k-j+1));
  end
end
C = D(n,n);
for k=(n-1):-1:1,
  C = conv(C,poly(X(k)));
  m = length(C);
  C(m) = C(m) + D(k,k);
end
```

2 Numerical Quadrature

2.1 Fast Clenshaw Curtis Quadrature

```
function [x,w]=fclencurt(N1,a,b)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% fclencurt.m - Fast Clenshaw Curtis Quadrature
%
% Compute the N nodes and weights for Clenshaw-Curtis
% Quadrature on the interval [a,b]. Unlike Gauss
% quadratures, Clenshaw-Curtis is only exact for
% polynomials up to order N, however, using the FFT
% algorithm, the weights and nodes are computed in linear
% time. This script will calculate for N=2^20+1 (1048577
% points) in about 5 seconds on a normal laptop computer.
%
% Written by: Greg von Winckel - 02/12/2005
% Copyright (c) 2009, Greg von Winckel
% All rights reserved.%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
19
20 N=N1-1; bma=b-a;
21 c=zeros(N1,2);
22 c(1:2:N1,1)=(2./[1 1-(2:2:N).^2 ])'; c(2,2)=1;
23 f=real(ifft([c(1:N1,:);c(N:-1:2,:)]));
24 w=bma*([f(1,1); 2*f(2:N,1); f(N1,1)])/2;
25 x=0.5*((b+a)+N*bma*f(1:N1,2));
```

2.2 Legendre-Gauss Quadrature

```
1  function [x,w]=lgwt(N,a,b)
2  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
3  % lgwt.m
4  %
5  % This script is for computing definite integrals using
       Legendre-Gauss
6  % Quadrature. Computes the Legendre-Gauss nodes and weights
       on an interval
7  % [a,b] with truncation order N
8  %
9  % Suppose you have a continuous function f(x) which is
       defined on [a,b]
10 % which you can evaluate at any x in [a,b]. Simply evaluate
       it at all of
11 % the values contained in the x vector to obtain a vector f
       . Then compute
12 % the definite integral using sum(f.*w);
13 %
14 % Written by Greg von Winckel - 02/25/2004
15 % Copyright (c) 2009, Greg von Winckel
16 % All rights reserved.
17 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
18
19 N=N-1;
20 N1=N+1; N2=N+2;
21
22 xu=linspace(-1,1,N1)';
23
24 % Initial guess
25 y=cos((2*(0:N)'+1)*pi/(2*N+2))+(0.27/N1)*sin(pi*xu*N/N2);
26
27 % Legendre-Gauss Vandermonde Matrix
28 L=zeros(N1,N2);
29
```

```
% Derivative of LGVM
Lp=zeros(N1,N2);

% Compute the zeros of the N+1 Legendre Polynomial
% using the recursion relation and the Newton-Raphson
   method

y0=2;

% Iterate until new points are uniformly within epsilon of
   old points
while max(abs(y-y0))>eps


    L(:,1)=1;
    Lp(:,1)=0;

    L(:,2)=y;
    Lp(:,2)=1;

    for k=2:N1
        L(:,k+1)=( (2*k-1)*y.*L(:,k)-(k-1)*L(:,k-1) )/k;
    end

    Lp=(N2)*( L(:,N1)-y.*L(:,N2) )./(1-y.^2);

    y0=y;
    y=y0-L(:,N2)./Lp;

end

% Linear map from[-1,1] to [a,b]
x=(a*(1-y)+b*(1+y))/2;

% Compute the weights
w=(b-a)./((1-y.^2).*Lp.^2)*(N2/N1)^2;
```

3 Root Finding

3.1 Bisection method

```
function [c,err,yc]=bisect(f,a,b,delta)

```

```
%Input    - f is the function
%                 - a and b are the left and right endpoints
%                 - delta is the tolerance
%Output - c is the zero
%                 - yc= f(c)
%                 - err is the error estimate for c

%If f is defined as an M-file function use the @ notation
% call [c,err,yc]=bisect(@f,a,b,delta).
%If f is defined as an anonymous function use the
% call [c,err,yc]=bisect(f,a,b,delta).

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

ya=f(a);
yb=f(b);
if ya*yb > 0,return,end
max1=1+round((log(b-a)-log(delta))/log(2));
for k=1:max1
        c=(a+b)/2;
        yc=f(c);
        if yc==0
                a=c;
                b=c;
        elseif yb*yc>0
                b=c;
                yb=yc;
        else
                a=c;
                ya=yc;
        end
        if b-a < delta, break,end
end

c=(a+b)/2;
err=abs(b-a);
yc=f(c);
```

3.2 Newton's method

```
function [p0,err,k,y]=newton(f,df,p0,delta,epsilon,max1)

%Input    - f is the object function
%            - df is the derivative of f
%            - p0 is the initial approximation to a zero of
    f
%                - delta is the tolerance for p0
%                - epsilon is the tolerance for the
   function values y
%                - max1 is the maximum number of iterations
%Output - p0 is the Newton-Raphson approximation to the
   zero
%                - err is the error estimate for p0
%                - k is the number of iterations
%                - y is the function value f(p0)

%If f and df are defined as M-file functions use the @
   notation
% call [p0,err,k,y]=newton(@f,@df,p0,delta,epsilon,max1).
%If f and df are defined as anonymous functions use the
% call  [p0,err,k,y]=newton(f,df,p0,delta,epsilon,max1).


%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

for k=1:max1
        p1=p0-f(p0)/df(p0);
        err=abs(p1-p0);
        relerr=2*err/(abs(p1)+delta);
        p0=p1;
        y=f(p0);
        if (err<delta)||(relerr<delta)||(abs(y)<epsilon)
                break
        end
end
```

3.3 Secant method

```
function [p1,err,k,y]=secant(f,p0,p1,delta,epsilon,max1)

%Input    - f is the object function
%             - p0 and p1 are the initial approximations to
   a zero of f
%                - delta is the tolerance for p1
%                - epsilon is the tolerance for the
   function values y
%                - max1 is the maximum number of iterations
%Output - p1 is the secant method approximation to the zero
%                - err is the error estimate for p1
%                - k is the number of iterations
%                - y is the function value f(p1)

%If f is defined as an M-file function use the @ notation
% call [c,err,yc]=bisect(@f,a,b,delta).
%If f is defined as an anonymous function use the
% call [c,err,yc]=bisect(f,a,b,delta).

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

for k=1:max1
        p2=p1-f(p1)*(p1-p0)/(f(p1)-f(p0));
        err=abs(p2-p1);
        relerr=2*err/(abs(p2)+delta);
        p0=p1;
        p1=p2;
        y=f(p1);
        if (err<delta)|(relerr<delta)|(abs(y)<epsilon),
           break,end
end
```

3.4 Fixed point iteration

```
function [k,p,err,P] = fixpt(g,p0,tol,max1)
```

```
2
3  %Input   -  g is the iteration function
4  %           -  p0 is the initial guess for the fixed-point
5  %           -  tol is the tolerance
6  %           -  max1 is the maximum number of iterations
7  %Output-  k is the number of iterations
8  %           -  p is the approximation to the fixed-point
9  %           - err is the error in the approximation
10 %           -  P' contains the sequence {pn}
11
12 %If g is defined as an M-file function use the @ notation
13 % call [k,p,err,P]=fixedpoint(@g,p0,tol,max1).
14 %If g is defined as an anonymous function use the
15 % call [k,p,err,P]=fixedpoint(g,p0,tol,max1).
16
17 %  NUMERICAL METHODS: Matlab Programs
18 % (c) 2004 by John H. Mathews and Kurtis D. Fink
19 %  Complementary Software to accompany the textbook:
20 %  NUMERICAL METHODS: Using Matlab, Fourth Edition
21 %  ISBN: 0-13-065248-2
22 %  Prentice-Hall Pub. Inc.
23 %  One Lake Street
24 %  Upper Saddle River, NJ 07458
25
26 P(1)= p0;
27
28 for k=2:max1
29         P(k)=g(P(k-1));
30         err=abs(P(k)-P(k-1));
31         relerr=err/(abs(P(k))+eps);
32         p=P(k);
33         if (err<tol) | (relerr<tol),break;end
34 end
35
36 P=P';
```

4 Differential Equation

4.1 Euler's method

```
1  function E=euler(f,a,b,ya,M)
2
3  %Input    - y'=f is the function
```

```
%             - a and b are the left and right endpoints
%             - ya is the initial condition y(a)
%             - M is the number of steps
%Output - E=[T' Y'] where T is the vector of abscissas and
%             - Y is the vector of ordinates

%If f is defined as an M-file function use the @ notation
% call E=euler(@f,a,b,ya,M).
%If f is defined as an anonymous function use the
% call E=euler(f,a,b,ya,M).

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

h=(b-a)/M;
T=zeros(1,M+1);
Y=zeros(1,M+1);
T=a:h:b;
Y(1)=ya;

for j=1:M
   Y(j+1)=Y(j)+h*f(T(j),Y(j));
end

E=[T' Y'];
```

4.2 Heun's method

```
function H=heun(f,a,b,ya,M)

%Input    - y'= f is the function
%             - a and b are the left and right endpoints
%             - ya is the initial condition y(a)
%             - M is the number of steps
%Output - H=[T' Y'] where T is the vector of abscissas and
%             - Y is the vector of ordinates

%If f is defined as an M-file function use the @ notation
```

```
% call E=heun(@f,a,b,ya,M).
%If f is defined as an anonymous function use the
% call E=heun(f,a,b,ya,M).

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

h=(b-a)/M;
T=zeros(1,M+1);
Y=zeros(1,M+1);
T=a:h:b;
Y(1)=ya;
for j=1:M
   k1=f(T(j),Y(j));
   k2=f(T(j+1),Y(j)+h*k1);
   Y(j+1)=Y(j)+(h/2)*(k1+k2);
end

H=[T' Y'];
```

4.3 Aadms-Bashforth's method

```
function [ x, y ] = ab2 ( f, xRange, yInitial, numSteps )
% [ x, y ] = ab2 ( f, xRange, yInitial, numSteps ) uses
% Adams-Bashforth second-order method to solve a system
% of first-order ODEs y?=f(x,y).
% f = name of an m-file with signature
% yprime = f(x,y)
% to compute the right side of the ODE as a row vector
%
% xRange = [x1,x2] where the solution is sought on x1<=x<=
   x2
% yInitial = column vector of initial values for y at x1
% numSteps = number of equally-sized steps to take from x1
   to x2
% x = column vector of values of x
% y = matrix whose k-th row is the approximate solution at
   x(k).
```

```
x=zeros(numSteps+1,1);
x(1) = xRange(1);
h = ( xRange(2) - xRange(1) ) / numSteps;
y(1,:) = transpose(yInitial);
yprime(1,:) = transpose(feval ( f, x(1), y(1,:) ));
k = 1;
xhalf = x(k) + 0.5 * h;
yhalf = y(k,:) + 0.5 * h * yprime(k,:);
yprime1 = transpose(feval ( f, xhalf, yhalf ));
x(k+1) = x(k) + h;
y(k+1,:) = y(k,:) + h * yprime1;
yprime(k+1,:) = transpose(feval ( f, x(k+1), y(k+1,:) ));
for k = 2 : numSteps
x(k+1) = x(k) + h;
y(k+1,:) = y(k,:) + ...
h * ( 3.0 * yprime(k,:) - yprime(k-1,:) ) / 2.0;
if k<numSteps
yprime(k+1,:) = transpose(feval ( f, x(k+1), y(k+1,:) ));
end
end
```

4.4 4-order Runge Kutta's method

```
function R=rk4(f,a,b,ya,M)

%Input    - f is the function
%             - a and b are the left and right endpoints
%             - ya is the initial condition y(a)
%             - M is the number of steps
%Output - R = [T' Y'] where T is the vector of abscissas
%               and Y is the vector of ordinates

%If f is an M-file function call R=rk4(@f,a,b,ya,M).
%If f is an anonymous function call R=rk4(f,a,b,ya,M).

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

```

```
h=(b-a)/M;
T=zeros(1,M+1);
Y=zeros(1,M+1);
T=a:h:b;
Y(1)=ya;
for j=1:M
    k1=h*f(T(j),Y(j));
    k2=h*f(T(j)+h/2,Y(j)+k1/2);
    k3=h*f(T(j)+h/2,Y(j)+k2/2);
    k4=h*f(T(j)+h,Y(j)+k3);
    Y(j+1)=Y(j)+(k1+2*k2+2*k3+k4)/6;
end

R=[T' Y'];
```

4.5 4-order Taylor's method

```
function T4=taylor(df,a,b,ya,M)

%Input     - df=[y' y'' y''' y''''] where y'=f(t,y)
%             - a and b are the left and right endpoints
%             - ya is the initial condition y(a)
%             - M is the number of steps
%Output - T4=[T' Y'] where T is the vector of abscissas and
%             - Y is the vector of ordinates

%If df is an M-file function call T4=taylor(@df,a,b,ya,M).
%If df is an anonymous function call T4=taylor(df,a,b,ya,M)
    .

%  NUMERICAL METHODS: Matlab Programs
% (c) 2004 by John H. Mathews and Kurtis D. Fink
%  Complementary Software to accompany the textbook:
%  NUMERICAL METHODS: Using Matlab, Fourth Edition
%  ISBN: 0-13-065248-2
%  Prentice-Hall Pub. Inc.
%  One Lake Street
%  Upper Saddle River, NJ 07458

h=(b-a)/M;
T=zeros(1,M+1);
Y=zeros(1,M+1);
T=a:h:b;
Y(1)=ya;
```

```
27
28 for j=1:M
29     D=df(T(j),Y(j));
30     Y(j+1)=Y(j)+h*(D(1)+h*(D(2)/2+h*(D(3)/6+h*D(4)/24)));
31 end
32
33 T4=[T' Y'];
```