MA3105

Numerical Analysis

Autumn 2021

Satvik Saha
19MS154

Indian Institute of Science Education and Research, Kolkata,
Mohanpur, West Bengal, 741246, India.

Contents

1 Time complexity

1.1 Runtime cost

When designing or implementing an algorithm, we care about its efficiency – both in terms of execution time, and the use of resources. This gives us a rough way of comparing two algorithms. However, such metrics are architecture and language dependent; different machines, or the same program implemented in different programming languages, may consume different amounts of time or resources while executing the same algorithm. Thus, we seek a way of measuring the 'cost' in time for a given algorithm.

For example, we may look at each statement in a program, and associate a cost c_i with each of them. Consider the following statements.

```
one = 1;                        // c_1
two = 2;                        // c_2
three = 3;                      // c_3
```

The total cost of running these statements can be calculated as $T = c_1 + c_2 + c_3$, simply by adding up the cost of each statement. Similarly, consider the following loop construct.

```
sum = 0;                        // c_1
for (i = 0; i < n; i++)         // c_2
    sum += a[i];                // c_3
```

The total cost can be shown to be $T(n) = c_1 + c_2(n+1) + c_3 n$; this time, we must take into account the number of times a given statement is executed. Note that this is linear. Another example is as follows.

```
sum = 0;                        // c_1
for (i = 0; i < n; i++)         // c_2
    for (j = 0; j < n; j++)     // c_2
        sum += a[i][j];         // c_4
```

The total cost can be shown to be $T(n) = c_1 + c_2(n+1) + c_3 n(n+1) + c_4 n^2$. Note that this is quadratic. Finally, consider the following recursive call.

```
int factorial (int n) {         // c_1
    if (n == 0)                 // c_2
        return 1;               // c_3
    return n * factorial(n - 1);  // c_4
}

f = factorial(n);               // c_5
```

The cost can be shown to be $T(n) = c_5 + (c_1 + c_2)(n+1) + c_3 + c_4 n$. This turns out to be linear.

In all these cases, we care about our total cost as a function of the input size n. Moreover, we are interested mostly in the *growth* of our total cost; as our input size grows, the total cost can often be compared with some simple function of n. Thus, we can classify our cost functions in terms of their asymptotic growths.

1.2 Asymptotic growth

Definition 1.1. The set $O(g(n))$ denotes the class of functions f which are asymptotically bounded above by g. In other words, $f(n) \in O(g(n))$ if there exists $M > 0$ and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$|f(n)| \leq Mg(n).$$

This amounts to writing

$$\limsup_{n\to\infty} \frac{|f(n)|}{g(n)} < \infty.$$

Example. Consider a function defined by $f(n) = an + b$, where $a > 0$. Then, we can write $f(n) \in O(n)$. To see why, note that for all $n \geq 1$, we have

$$|f(n)| = |an + b| \leq an + |b| \leq (a + |b|)n.$$

Thus, setting $M = a + |b| > 0$ completes the proof.

Example. Consider a polynomial function defined by

$$f(n) = a_k n^k + a_{k-1} n^{k-1} + \cdots + a_1 n + a_0,$$

with some non-zero coefficient. Then, we can write $f(n) \in O(n^k)$. Like before, note that for all $n \geq 1$, we have

$$|f(n)| \leq \sum_{i=0}^{k} |a_i| n^i \leq \sum_{i=0}^{k} |a_i| n^k = (|a_k| + |a_{k-1}| + \cdots + |a_0|)n^k.$$

Thus, setting $M = |a_k| + \cdots + |a_0| > 0$ completes the proof.

Theorem 1.1. *If $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then*

$$f_1(n) + f_2(n) \in O(\max\{g_1(n), g_2(n)\}).$$

Definition 1.2. The set $\Omega(g(n))$ denotes the class of functions f are asymptotically bounded below by g. In other words, $f(n) \in \Omega(g(n))$ if there exists $M > 0$ and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$|f(n)| \geq Mg(n).$$

This amounts to writing

$$\liminf_{n\to\infty} \frac{f(n)}{g(n)} > 0.$$

Definition 1.3. The set $\Theta(g(n))$ denotes the class of functions f which are asymptotically bounded both above and below by g. In other words, $f(n) \in \Theta(g(n))$ if there exist $M_1, M_2 > 0$ and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$M_1 g(n) \leq |f(n)| \leq M_2 g(n).$$

This amounts to writing $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$.

Another class of notation uses the idea of dominated growth.

Definition 1.4. The set $o(g(n))$ denotes the class of functions f which are asymptotically dominated by g. In other words, $f(n) \in o(g(n))$ if for all $M > 0$, there exists $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$|f(n)| < Mg(n).$$

This amounts to writing

$$\lim_{n\to\infty} \frac{|f(n)|}{g(n)} = 0.$$

Definition 1.5. The set $\omega(g(n))$ denotes the class of functions f which asymptotically dominate g. In other words, $f(n) \in \omega(g(n))$ if for all $M > 0$, there exists $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$|f(n)| > Mg(n).$$

This amounts to writing

$$\lim_{n\to\infty} \frac{|f(n)|}{g(n)} = \infty.$$

Definition 1.6. We say that $f(n) \sim g(n)$ if f is asymptotically equal to g. In other words, $f(n) \sim g(n)$ if for all $\epsilon > 0$, there exists $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$,

$$\left| \frac{f(n)}{g(n)} - 1 \right| < \epsilon.$$

This amounts to writing

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1.$$

We often abuse notation and treat the following as equivalent.

$$T(n) \in O(g(n)), \qquad T(n) = O(g(n)).$$

2 Root finding methods

Consider an equation of the form $f(x) = 0$, where $f\colon [a, b] \to \mathbb{R}$ is given. We wish to solve this equation, i.e. find the roots of f.

Note that for *arbitrary* functions, this task is impossible. To see this, consider a function f which assumes the value 1 on $[0, 1] \setminus \{\alpha\}$ and $f(\alpha) = 0$, for some $\alpha \in [0, 1]$. There is no way of pinpointing α without checking f at every point in $[0, 1]$. Besides, a computer cannot reasonably store real numbers with arbitrary precision.

Thus, we direct our attention towards *continuous* functions f. We only seek sufficiently accurate approximations of its root $\alpha \in (a, b)$.

Theorem 2.1 (Intermediate Value Theorem). *Let $f \colon [a, b] \to \mathbb{R}$ be continuous. If $f(a)f(b) < 0$, then there exists $\alpha \in (a, b)$ such that $f(\alpha) = 0$.*

2.1 Tabulation method

To identify the location of a root of f on an interval $I = [a, b]$, we subdivide I into n subintervals $[x_i, x_{i+1}]$ where $x_i = a + (b - a)i/n$. Now, we simply apply the Intermediate Value Theorem to f on each of these intervals. If $f(x_i)f(x_{i+1}) < 0$, then f has a root somewhere in (x_i, x_{i+1}). Note that the error in our approximation is on the order of $|b - a|/n$. The precision of this method can be improved by increasing n.

To reach a degree of approximation ϵ, we must iterate n times, where

$$n > \frac{b - a}{\epsilon}.$$

2.2 Bisection method

Here, we first verify that $f(a)f(b) < 0$, thus ensuring that f has a root within (a, b). Now, set $x_1 = a + (b - a)/2$ and apply the Intermediate Value Theorem on the subintervals $[a, x_1]$ and $[x_1, b]$. One of these *must* contain a root of f. Note that if $f(x_1) = 0$, we are done; otherwise, let $I_1 = [a_1, b_1]$ be the subinterval containing the root. Repeat the above process, obtaining successive subintervals I_n with lengths $|b - a|/2^n$. The error in our approximation is of this order, and can be controlled by stopping at appropriately large n.

The quantity $x_{n+1} = (a_n + b_n)/2$ is a good approximation for the actual root α since we know that $x_{n+1}, \alpha \in [a_n, b_n]$, so

$$|x_{n+1} - \alpha| \leq |b_n - a_n| = 2^{-n}|b - a| \to 0.$$

To reach a degree of approximation ϵ, we must iterate n times, where

$$n > \log_2 \frac{b - a}{\epsilon}.$$

2.3 Newton-Raphson method

Assuming that f is twice differentiable, use Taylor's theorem to write

$$f(x) = f(x_0) + f'(x)(x - x_0) + \frac{1}{2}f''(c)(x - x_0)^2$$

for all $x \in [a, b]$, where c is between x and x_0. The first two terms represent the tangent line to f, drawn at $(x_0, f(x_0))$. Now, define

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Note that this is the point at which the tangent line to f at x_0 cuts the x-axis. We have implicitly assumed that $f'(x_0) \neq 0$. In this manner, create the sequence of points

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

We wish to show that $x_n \to \alpha$, under certain circumstances.

Definition 2.1 (Order of convergence)**.** Let $x_n \to \alpha$. We say that this convergence is of order $p \geq 1$ if

$$\lim_{n\to\infty} \frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^p} > 0.$$

Theorem 2.2. *Let f be a real function on $[\alpha - \delta, \alpha + \delta]$ such that*

1. $f(\alpha) = 0$.
2. *f is twice differentiable, with non-zero derivatives.*
3. *f'' is continuous.*
4. *$|f''(x)/f'(y)| \leq M$ for all x, y.*

If $x_0 \in [\alpha - h, \alpha + h]$ where $h = \min\{\delta, 1/M\}$, then the Newton-Raphson sequence generated by x_0 converges to the root α quadratically.

Proof. Pick $x_n \in [\alpha - h, \alpha + h]$. Using Taylor's theorem,

$$f(\alpha) = f(x_n) + f'(x_n)(\alpha - x_n) + \frac{1}{2}f''(c)(\alpha - x_n)^2.$$

Also note that $f(\alpha) = 0$, and $x_n - x_{n+1} = f(x_n)/f'(x_n)$. Thus, dividing by $f'(x_n)$ and substituting gives

$$\alpha - x_{n+1} = -\frac{1}{2}\frac{f''(c)}{f'(x_n)}(\alpha - x_n)^2.$$

Using our estimates on $f''(c)/f'(x_n)$ and x_n along with $h \leq 1/M$, we see that

$$|\alpha - x_{n+1}| \leq \frac{1}{2}Mh|\alpha - x_n| \leq \frac{1}{2}|\alpha - x_n|.$$

Indeed, we have shown that

$$|\alpha - x_n| \leq \frac{1}{2^n}|\alpha - x_0|,$$

which directly gives the convergence $x_n \to \alpha$. Furthermore, we have

$$\frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^2} = \frac{1}{2}\left|\frac{f''(c)}{f'(x_n)}\right| \leq \frac{1}{2}M,$$

hence taking the limit $n \to \infty$ proves that the convergence is quadratic. □

Corollary 2.2.1. *Suppose that f satisfies the conditions of the previous theorem, along with $f' > 0$ and $f'' > 0$ on some interval $[\alpha, x]$. Then, the Newton-Raphson sequence generated by $x_0 \in [\alpha, x]$ converges to the root α quadratically.*

Remark. The convexity of f means that the tangent drawn at x_n lies below the curve, and hence cuts the x-axis between α and x_n.

Theorem 2.3. *If α is a multiple root of f such that $f(\alpha) = 0$, $f'(\alpha) = 0$, $f''(\alpha) \neq 0$, then the Newton-Raphson sequence converges to α linearly under suitable conditions.*

Proof. Use Rolle's Theorem to replace $f'(x_n) = f'(x_n) - f'(\alpha) = f''(a)(x_n - \alpha)$. □

2.4 Secant method

The chief difference between this method as Newton's method is that we approximate the tangent with a secant, i.e. perform an approximation of the derivative,

$$f'(x)h \approx f(x+h) - f(x)$$

for small h. Thus, our iterations proceed as

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

Theorem 2.4. *Let f be a real function on $[a, b]$ such that*

1. *$f(\alpha) = 0$ where $\alpha \in (a, b)$.*
2. *f is continuously differentiable, with non-zero derivatives.*

Then, there exists $\delta > 0$ such that the sequence generated by the secant method converges to α when $x_0, x_1 \in (\alpha - \delta, \alpha + \delta)$.

Proof. Consider

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

Now, use the Mean Value Theorem to write $f(x_n) = f(x_n) - f(\alpha) = f'(\xi)(x_n - \alpha)$ for some ξ between α and x_n. Similarly, write $f(x_n) - f(x_{n-1}) = f'(\zeta)(x_n - x_{n-1})$ for some ζ between x_n and x_{n-1}. Thus,

$$\alpha - x_{n-1} = \alpha - x_n + \frac{f'(\xi)(x_n - \alpha)}{f'(\zeta)} = (\alpha - x_n)\left(1 - \frac{f'(\xi)}{f'(\zeta)}\right).$$

We want $|1 - f'(\xi)/f'(\zeta)| < 1$. Since $f'(\alpha) \neq 0$, there is a δ-neighbourhood of α where $3f'(\alpha)/4 < f'(x) < 5f'(\alpha)/4$ (without loss of generality) using the continuity of f'. Thus, whenever $x_0, x_1 \in (\alpha - \delta, \alpha + \delta)$, we have ξ, ζ belonging to the same neighbourhood. This gives $3/5 < f'(\zeta)/f'(\xi) < 5/3$. This gives

$$-\frac{2}{3} < 1 - \frac{f'(\xi)}{f'(\zeta)} < \frac{2}{5}.$$

In other words, $|1 - f'(\xi)/f'(\zeta)| < 2/3$, so

$$|\alpha - x_{n+1}| < \frac{2}{3}|\alpha - x_n|,$$

which directly gives $x_n \to \alpha$.

The order of convergence turns out to be $\varphi = (1 + \sqrt{5})/2$. To show this, we want

$$\lim_{n\to\infty} \frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^{\varphi}} > 0.$$

Assume that $f'(\alpha) > 0$, $f''(\alpha) > 0$. First, we will show that

$$\lim_{n\to\infty} \frac{|\alpha - x_{n+1}|}{|\alpha - x_n||\alpha - x_{n-1}|} = \frac{f''(\alpha)}{2f'(\alpha)}.$$

Denote the quantity in the limit as $\psi(x_n, x_{n-1})$. We examine the equivalent limit

$$\lim_{x_{n-1}\to\alpha} \lim_{x_n\to\alpha} \psi(x_n, x_{n-1}).$$

Like before, write

$$\alpha - x_{n+1} = (\alpha - x_n)\left(1 - \frac{f'(\xi)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}\right),$$

hence

$$\frac{\alpha - x_{n+1}}{(\alpha - x_n)(\alpha - x_{n-1})} = \frac{1}{\alpha - x_{n-1}}\left[1 - \frac{f'(\xi)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}\right].$$

Thus,

$$\begin{aligned}\lim_{x_n\to\alpha} \psi(x_n, x_{n-1}) &= \frac{1}{\alpha - x_{n-1}}\left[1 + \frac{f'(\alpha)(\alpha - x_{n-1})}{f(x_{n-1})}\right] \\ &= \frac{f(x_{n-1}) + f'(\alpha)(\alpha - x_{n-1})}{f(x_{n-1})(\alpha - x_{n-1})}.\end{aligned}$$

Use Taylor's Theorem to approximate

$$f(x_{n-1}) = f(\alpha) + f'(\alpha)(x_{n-1} - \alpha) + \frac{1}{2}f''(\eta)(x_{n-1} - \alpha)^2,$$

giving

$$\lim_{x_n\to\alpha} \psi(x_n, x_{n-1}) = \frac{f''(\eta)(\alpha - x_{n-1})^2}{2f(x_{n-1})(\alpha - x_{n-1})},$$

and use the Mean Value Theorem to write $f(x_{n-1}) = f'(\kappa)(x_{n-1} - \alpha)$ giving

$$\lim_{x_n\to\alpha} \psi(x_n, x_{n-1}) = -\frac{f''(\eta)}{2f'(\kappa)},$$

This gives

$$\lim_{x_{n-1}\to\alpha} \lim_{x_n\to\alpha} |\psi(x_n, x_{n-1})| = \frac{f''(\alpha)}{2f'(\alpha)} = C.$$

Now, suppose that

$$\lim_{n\to\infty} \frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^q} = A > 0.$$

Dividing, we have

$$\lim_{n\to\infty} \frac{|\alpha - x_n|^{q-1}}{|\alpha - x_{n-1}|} = \frac{C}{A}, \qquad \lim_{n\to\infty} \frac{|\alpha - x_n|}{|\alpha - x_{n-1}|^{1/(q-1)}} = \left(\frac{C}{A}\right)^{1/(q-1)} > 0.$$

For q to be minimal, we must have $1/(q-1) = q$, or q is the golden ratio φ. □

2.5 Fixed point method

Note that a root of f is simply a fixed point of $f + x$.

Theorem 2.5. *Let $f\colon [a, b] \to [a, b]$ be continuous. Then, f has a fixed point $\beta \in [a, b]$, $f(\beta) = \beta$.*

Thus, let $f\colon [a, b] \to [a, b]$ be continuous. Define the fixed point sequence $x_{n+1} = f(x_n)$, seeded by some $x_0 \in [a, b]$. Note that if this sequence converges with $x_n \to \beta$, then β is a fixed point of f.

Definition 2.2. A function $f\colon [a, b] \to \mathbb{R}$ is said to be a contraction if there exists $L \in (0, 1)$ such that $|f(x) - f(y)| \le L|x - y|$ for all $x, y \in [a, b]$.

Remark. Note that f is Lipschitz continuous. If f is also differentiable, then $|f'| < 1$.

Theorem 2.6. *Let $f\colon [a, b] \to [a, b]$ be a contraction map. Then, any fixed point sequence converges to the unique fixed point of f.*

Proof. First, we show that f has at most one fixed point. Let β_1, β_2 be fixed points of f. Then, $|f(\beta_1) - f(\beta_2)| \le L|\beta_1 - \beta_2|$ where $L \in (0, 1)$. This forces $\beta_1 = \beta_2$. Thus, f has a unique fixed point in $[a, b]$.

Let $\{x_n\}$ be a fixed point iteration. Then,

$$|x_{n+1} - \beta| = |f(x_n) - f(\beta)| \le L|x_n - \beta|,$$

which directly gives $x_n \to \beta$. □

3 Interpolation

3.1 Lagrange interpolation

Theorem 3.1. *Let $x_1, \dots, x_n \in \mathbb{R}$ be distinct, and let $y_1, \dots, y_n \in \mathbb{R}$. Then, the following polynomial of degree $n - 1$ satisfies $p(x_i) = y_i$.*

$$p(x) = \sum_{i=1}^{n} \prod_{j \neq i} \frac{x - x_j}{x_i - x_j} y_i.$$

Furthermore, this choice of p is unique.

Proof. The polynomials

$$p_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$$

satisfy $p_i(x_j) = \delta_{ij}$. These p_i form a basis of $\mathscr{P}^{n-1}$, the space of polynomials of degree at most $n - 1$. □

Theorem 3.2. *Let $f\colon [a,b] \to \mathbb{R}$ be n times differentiable, and let p be the Lagrange interpolating polynomial of f on the points $x_1, \dots, x_n$. Then, for any $x \in [a,b]$, there exists $\xi \in (a,b)$ such that*

$$f(x) - p(x) = \frac{f^{(n)}(\xi)}{n!} \prod_i (x - x_i)$$

Proof. This is clear when $x = x_i$. Suppose that $x \neq x_i$ for any i. Define

$$g\colon [a,b] \to \mathbb{R}, \qquad g(t) = f(t) - p(t) - (f(x) - p(x)) \prod_i \frac{t - x_i}{x - x_i}$$

We see that each $g(x_i) = 0$, as well as $g(x) = 0$, hence g has $n+1$ distinct roots. Hence, g' has exactly n distinct roots, and continuing in this fashion, $g^{(n)}$ has one root. Set ξ such that $g^{(n)}(\xi) = 0$. On the other hand,

$$g^{(n)}(\xi) = f^{(n)}(\xi) - n!(f(x) - p(x)) \prod_i \frac{1}{x - x_i}.$$

□

3.2 Newton's divided difference

Theorem 3.3. *Let $x_1, \dots, x_n \in \mathbb{R}$ be distinct, and let $y_1, \dots, y_n \in \mathbb{R}$. Define the divided difference recursively as*

$$\Delta(x_i) = y_i, \qquad \Delta(x_i, \dots, x_j) = \frac{\Delta(x_{i+1}, \dots, x_j) - \Delta(x_i, \dots, x_{j-1})}{x_j - x_i}.$$

Further denote

$$\Delta^k = \Delta(x_1, \dots, x_k).$$

Then, the following polynomial of degree $n-1$ interpolates the given data.

$$p(x) = \Delta^1 + (x - x_1)\Delta^2 + (x - x_1)(x - x_2)\Delta^3 + \cdots + (x - x_1)\cdots(x - x_{n-1})\Delta^n.$$

Remark. We already know that this must be identical to the Lagrange interpolating polynomial, hence all its properties carry over.

Remark. The divided difference $\Delta(x_1, \dots, x_k)$ is independent of the order of $x_1, \dots, x_k$.

4 Numerical integration

Let $f\colon [a,b] \to \mathbb{R}$ be continuously differentiable. We wish to approximate the value of the integral

$$\int_a^b f(x)\, dx.$$

The main way of doing this is to approximate the curve f using rectangles, trapeziums, parabolas, or even higher degree polynomials.

4.1 Newton-Cotes formula

Perform Lagrange interpolation of f on the points $x_1, \dots, x_n$, and write

$$\int_a^b f(x)\, dx \approx \sum_{i=1}^{n} f(x_i) \int_a^b p_i(x)\, dx.$$

4.2 Midpoint rule

Here, we use the midpoints $m_i = (x_i + x_{i+1})/2$, and write

$$\int_a^b f(x)\,dx \approx \Delta x\left[f(m_1) + \cdots + f(m_{n-1})\right].$$

4.3 Trapezoidal rule

Perform linear interpolations of f at equal intervals Δx and write

$$\int_a^b f(x)\,dx \approx \frac{1}{2}\Delta x\left[f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n)\right].$$

It can be shown that there exists ξ between a and b such that the error in this approximation is

$$-\frac{(b-a)^3}{12(n-1)^2}f''(\xi).$$

4.4 Simpson's rule

Perform quadratic interpolations of f, and write

$$\int_a^b f(x)\,dx \approx \frac{1}{3}\Delta x\left[f(x_1) + 4f(x_2) + 2f(x_3) + \cdots + 4f(x_{n-2}) + 2f(x_{n-1}) + f(x_n)\right].$$

Note that we need odd n. For each arc between a_i, b_i, we have used the area under the interpolating quadratic,

$$\int_{a_i}^{b_i} f(x)\,dx \approx \frac{1}{6}(b_i - a_i)\left[f(a_i) + 4f\left(\frac{a_i + b_i}{2}\right) + f(b_i)\right].$$

It can be shown that there exists ξ between a and b such that the error in this approximation is

$$-\frac{(b-a)^5}{180(n-1)^4}f^{(4)}(\xi).$$

5 Ordinary differential equations

Consider the initial value problem

$$y'(x) = f(x, y), \qquad y(x_0) = y_0,$$

where f is a continuous function on some open subset of $\mathbb{R}^2$. We are looking for a differentiable function y on an open neighbourhood of x_0, where each $(x, y(x))$ is in the domain of f.

5.1 Picard iterates

Any solution must satisfy

$$y(x) = y_0 + \int_{x_0}^x f(t, y(t))\,dt.$$

We may iterate $y_0(x) = y_0$, and

$$y_{n+1}(x) = y_0 + \int_{x_0}^x f(t, y_n(t))\,dt.$$

It can be shown that the Picard iterates do indeed converge to a solution of the given ODE.

Theorem 5.1 (Picard-Lindelöf theorem)**.** *Let f be uniformly Lipschitz continuous in y. Then the initial value problem has a unique solution y on some neighbourhood of x_0.*

5.2 Euler's method

Assume that a solution y exists on $[x_0, x_M]$. Construct the n evenly spaced mesh points $x_0, x_1, x_2, \ldots, x_n$, where $x_n = x_M$. Setting $h = \Delta x$, we can approximate

$$y(x_{k+1}) = y(x_k + h) \approx y(x_k) + hy'(x_k) = y(x_k) + hf(x_k, y(x_k)).$$

This gives an iterative scheme to approximate $y(x)$ on these mesh points.

5.3 Trapezoidal method

Here, we use perform the iterations

$$y(x_{n+1}) \approx y(x_n) + \frac{1}{2}h(f(x_n, y(x_n)) + f(x_{n+1}, y(x_{n+1}))).$$

In order to use this implicit relation, we can use Newton's method or fixed point iteration.

5.4 Runge-Kutta methods

The second order Runge-Kutta method uses the following iterations.

$$y_{n+1} = y_n + h(ak_1 + bk_2), \qquad k_1 = f(x_n, y_n), \qquad k_2 = f(x_n + \alpha h, y_n + \beta hk_1).$$

Additionally, we choose $\beta = \alpha$, $a = 1 - 1/2\alpha$, $b = 1 - a$.

The fourth order Runge-Kutta method uses the iterations

$$y_{n+1} = y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4),$$

$$k_1 = f(x_n, y_n), \qquad k_2 = f(x_{n+\frac{1}{2}}, y_n + \frac{1}{2}hk_1),$$

$$k_3 = f(x_{n+\frac{1}{2}}, y_n + \frac{1}{2}hk_2), \qquad k_4 = f(x_{n+1}, y_n + hk_2).$$