10.1 Finite Difference

Our goal is to compute the derivative of a nonlinear function $f(x)$. Since we do not know the explicit formula of the derivative in general, we must use numerical approach to approximate such value. Consider the derivative as the limit of difference operator:

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}.$$

It is natural to use a small enough h to approximate the limiting value for which $h \to 0$. To this end, we have the following options:

1. Forward difference formula: $f'(x) \approx \frac{f(x+h)-f(x)}{h}$.
2. Backward difference formula: $f'(x) \approx \frac{f(x)-f(x-h)}{h}$.

Let us examine the accuracy of finite difference approximation. Suppose $f \in C^2[a,b]$. The truncation error, namely the error between true derivative and finite difference approximation, can be estimated by Taylor expansion:

$$\varepsilon_t = f'(x) - \frac{f(x+h)-f(x)}{h} = f'(x) - \frac{f'(x)h + \frac{1}{2}f''(\xi)h^2}{h} = \frac{1}{2}f''(\xi)h = \mathcal{O}(h).$$

In an analogous way, the truncation error of backward difference formula is $\mathcal{O}(h)$.

Furthermore, suppose f has high order derivative, namely $f \in C^3[a,b]$, we can use centered difference formula:

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h}.$$

By expanding $f(x \pm h)$ to the third order, we obtain the truncation error:

$$\varepsilon_t = f'(x) - \frac{f(x+h)-f(x-h)}{2h} = f'(x) - \frac{2f'(x)h + \frac{1}{3}f'''(x)h^3}{2h} = \mathcal{O}(h^2).$$

Suppose $f \in C^{2q+1}[a,b]$, let

$$g(x) = \frac{1}{h}\sum_{t=-q}^{q} a_t f(x+th). \tag{10.1}$$

We can choose the coefficients a_t properly to obtain the truncation error $\varepsilon_p = f'(x) - g(x) = \mathcal{O}(h^{2q})$.

This can be illustrated using our polynomial interpolation. Let us denote $f_0(y) = f(x+y)$. Then forward difference can be viewed as the gradient of linear interpolant across $(0, f_0(0)$ and $(h, f_0(h))$. The centered difference can be viewed as the gradient of quadratic interpolant across $(-h, f_0(-h))$, $(0, f_0(0)$ and $(h, f_0(h))$. Using Lagrangian formula, we have

$$p(y) = \frac{(y-0)(y-h)}{(-h)(-2h)} f_0(-h) + \frac{(y+h)(y-h)}{h(-h)} f_0(0) + \frac{(y-0)(y+h)}{h(2h)} f_0(h).$$

Then $p'(0) = \frac{f_0(h) - f_0(-h)}{2h}$.

More generally, using $2q$-th order polynomial $p(y)$ to interpolate $(ih, f_0(ih))$, $i = -q, -q+1, ..., 0, 1, .., q$, we have $f_0(y) - p(y) = \mathcal{O}(f_0^{(p+1)}(\xi) h^{2q+1})$. Therefore, $f'(x) = f_0'(0) = p'(0) + \mathcal{O}(h^{2q})$.

It appears that we can take diminishing value of h to obtain arbitrarily accurate solution. However, this is not true because we still count on rounding error in computing (10.1). Recall that in error analysis, we have

$$f(x + \epsilon x) \approx f(x) + f(x)\kappa_f(x)\epsilon = f(x) + \mathcal{O}(\epsilon)$$

where $\epsilon \le \varepsilon_M$. The overall error of finite difference is

$$\varepsilon = \varepsilon_r + \varepsilon_t = \frac{\mathcal{O}(\varepsilon_M q)}{h} + \mathcal{O}(h^{2q}).$$

By optimizing the error bound w.r.t. h, we arrive at the optimal $h^* = \mathcal{O}(\varepsilon_M^{1/(1+2q)})$ and the error $\varepsilon = \mathcal{O}(\varepsilon_M^{2q/(1+2q)})$.

Example 1. Let us use finite difference to compute the derivative of $f(x) = \sin(e^{x+1})$ at $x = 0$. We know that the derivative is $f'(x) = \cos(e)e$. We compare the performance of forward difference and centered difference for $h = \frac{1}{4^k}$, $k = 1, 2, ..., 8$. Indeed, we observe that centered difference is more accurate than forward difference.

Next we further reduce h, such that $h = \frac{1}{4^k}$, $k = 7, 8, ..., 16$. We immediately observe that roundoff error starts to dominate as h is decreasing. However, centered difference still outperforms forward difference.

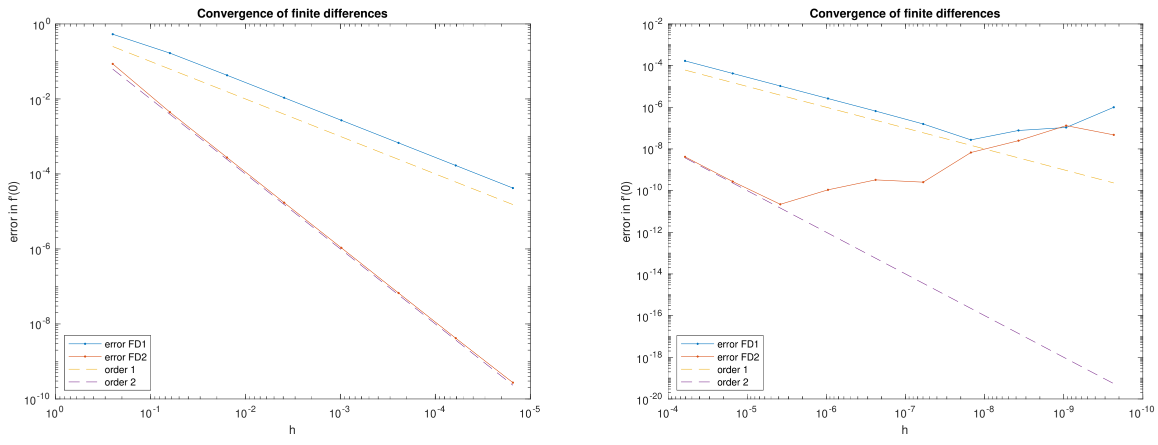

Figure 10.1: Convergence of ε_t for $h = \frac{1}{4^k}$. Left $k = 1, 2, ..., 8$. Right: $k = -7, -8, ..., -16$.

10.2 Integration

Our next goal is numerical integration. Similar to computing the derivative, since the integration may not have closed-form formula, numerical tool is our most trustworthy friend. Let us denote the integral

$$I(f) = \int_a^b f(x)dx.$$

and its numerical counterpart:

$$Q(f) = \sum_{k=0}^{M} w_k f(x_k)$$

where $a = x_0 < x_1 \cdots < x_M = b$ are the finite samples. The truncation error is defined as $\varepsilon_t = |I(f) - Q(f)|$.

Our strategy is to replace $f(x)$ by some known function for which integration is easy to compute. The most natural choice is to interpolate $f(x)$ by piecewise polynomial $p(x)$ and integrate on $p(x)$. The resulting formula is called *Newton-Cotes formula.*

Let us consider piecewise linear interpolant. Using cardinal function, we have $p(x) = \sum_{i=1}^{n} y_i h_i(x)$ where h_k is the hat function.

$$h_k(x) = \begin{cases} \frac{x-x_{k-1}}{x_k-x_{k-1}} & \text{if } x \in [x_{k-1}, x_k] \\ \frac{x_{k+1}-x}{x_{k+1}-x_k} & \text{if } x \in [x_k, x_{k+1}] \\ 0 & \text{o.w} \end{cases}, \quad 1 \le k \le n-1$$

Also $h_0(x) = \frac{x_1-x}{x_1-x_0}$ and $h_n = \frac{x-x_{n-1}}{x_n-x_{n-1}}$. It is obvious that $\int h_k(x)\,dx = \frac{1}{2}(x_{k+1} - x_{k-1})$. Therefore

$$\int_a^b p(x)\,dx = \frac{1}{2}y_0(x_1 - x_0) + \frac{1}{2}\sum_{i=1}^{n-1} y_i(x_{i+1} - x_{i-1}) + \frac{1}{2}y_n(x_n - x_{n-1}).$$

Suppose x_i are equally spaced, then $x_i = a + ih$, $h = \frac{b-a}{n}$. Using the notation $y_i = f(x_i)$, we have the *trapezoid formula*:

$$T_n(f) = \int_a^b p(x)\,dx = \frac{b-a}{n}\Big[\frac{1}{2}f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}f(x_n)\Big].$$

Since $\|f(x) - p(x)\|_\infty = \mathcal{O}(h^2)$, we have the truncation error

$$\varepsilon_t = |\int_a^b f(x) - p(x)\,dx| \le (b-a)\|f(x) - p(x)\|_\infty = \mathcal{O}((b-a)^3 n^{-2}).$$

Example 2 (Trapezoid formula)**.** We compute the integral $\int_0^2 e^{sin 7x}\,dx$. We use the following Matlab command to compute the accurate result.

```
I = integral(f,a,b,'abstol',1e-14,'reltol',1e-14);
```

We plot the function and the error of integration by trapezoid formula in Figure 10.2.

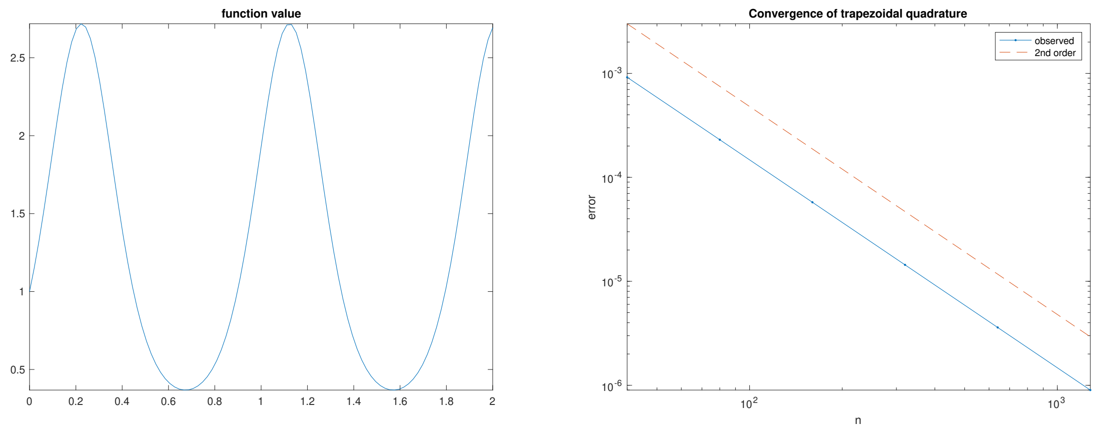

Figure 10.2: Left: plot of $e^{\sin 7x}$. Right: Convergence of error w.r.t. interpolation points number n

To further improve the accuracy, we can use piecewise high order polynomial for interpolation, which nevertheless, requires solving linear system. Another approach to improve the accuracy is via *Richardson Extrapolation*, which combines several approximations of a certain quantity in a smart way to yield a more accurate approximation. We will not dive into details, but consider a specific version, namely the *Romberg integration*, which extrapolates the trapezoid formula to obtain even higher order error term. First, we show that the error of trapezoid formula is of even degree:

$$\varepsilon_t(n) = I(f) - T_n(f) = a_1 n^{-2} + a_2 n^{-4} + a_3 n^{-6} + \cdots + a_k n^{-2k} + \cdots . \tag{10.2}$$

This property is implied by the *Euler-MacLaurin* formula. Replacing $n \to 2n$, we have

$$\varepsilon_t(2n) = I(f) - T_{2n}(f) = a_1 \frac{1}{4} n^{-2} + \frac{1}{16} a_2 n^{-4} + \frac{1}{2^6} a_3 n^{-6}$$

Multiplying both sides by 4, we have

$$4\varepsilon_t(2n) = a_1 n^{-2} + \frac{1}{4} a_2 n^{-4} + \frac{1}{2^4} a_3 n^{-6}. \tag{10.3}$$

Putting (10.2) and (10.3) together, we have

$$4\varepsilon_t(2n) - \varepsilon_t(n) = 3I(f) - 4T_{2n}(f) + T_n(f) = \mathcal{O}(n^{-4}).$$

Equivalently, we have

$$I(f) = \frac{1}{3}\big[4T_{2n}(f) - T_n(f)\big] + \mathcal{O}(n^{-4}).$$

The term

$$S_{2n}(f) = \frac{1}{3}\big[4T_{2n}(f) - T_n(f)\big] \tag{10.4}$$

is called the *Simpson rule*. The simpliest case with $n = 1$ is

$$S_2(f) = \frac{1}{6}\big[f(a) + 4f(\frac{a+b}{2}) + f(b)\big]. \tag{10.5}$$

Repeating the extrapolation, we have

$$R_{4n}(f) = \frac{1}{15}\big[16S_{4n}(f) - S_{2n}(f)\big]. \tag{10.6}$$

This term has truncation error of $\mathcal{O}(n^{-6})$.

Adaptive integration In the earlier improvement, $\{x_n\}$ are equally spaced. However, for some problems part of the curve changes more rapidly than the other part, it is thus more reasonable to place interpolating points unevenly. We can adaptively choose the interpolation grids depending on the error.

We highlight a simple divide and conquer procedure to compute $\int_{a_0}^{b_0} f(x)\,dx$. For sake of notation, we use $S[a,b]$ to denote the Simpson formula (10.5) on interval $[a,b]$, that is $S[a,b] = \frac{1}{6}\left[f(a) + 4f(\frac{a+b}{2}) + f(b)\right]$.

Algorithm outline The basic idea is to check whether the error on the current interval passes a given test or not, if it fails, we divide the interval into two parts and continue integration on them separately. We maintain all the unexamined interval and target error in a queue.

1. Initially we have $[a_0, b_0, \varepsilon_0]$. $\mathcal{A} = 0$.
2. Loop over the following steps until queue is empty.
   (a) Pop up the current triplet $[a, b, \varepsilon]$, denote $c = \frac{a+b}{2}$.
      i. if $\mathcal{T}(a, b, \varepsilon)$ is false, then insert $[a, c, \frac{\varepsilon}{2}]$ and $[c, b, \frac{\varepsilon}{2}]$ to the end of the queue.
      ii. if $\mathcal{T}(a, b, \varepsilon)$ is true, $\mathcal{A} = \mathcal{A} + S[a,c] + S[b,c] - S[a,c]$.

Test $\mathcal{T}(a, b, \varepsilon)$. We want to estimate the error of $S[a,b]$. However, as we do not know $I(f) = \int_a^b f(x)\,dx$ a priori, we must use other empirical value to estimate it. For example, we insert $c = \frac{a+b}{2}$ and return true if

$$\frac{1}{10}|S[a,c] + S[c,b] - S[a,b]| \le \varepsilon. \tag{10.7}$$

It can be deduced from our earlier analysis that $I(f) - S[a,c] - S[c,b] = \mathcal{O}((b-a)^5)$. Hence when $a \approx b$, (10.7) will be a good estimate of integration error.