Numerical differentiation

$$f'(x)=\frac{f(x+h)-f(x)}{h}+O(h)$$ - forward difference

$$f'(x)=\frac{f(x)-f(x-h)}{h}+O(h)$$ - backward difference

$$f'(x)=\frac{f(x+h/2)-f(x-h/2)}{h}+O(h^2)$$ - centred difference

Generally, centred difference is preferred. Exception: function evaluation is costly and f(x) is already available.

Second derivative $$f''(x)\approx\frac{f'(x+h/2)-f'(x-h/2)}{h}\approx\frac{f(x+h)-2f(x)+f(x-h)}{h^2}$$

Accuracy cannot reach machine precision. Can we do better?

Accuracy cannot reach machine precision. Can we do better?

Yes, by extrapolation.

$$f'(x)=\frac{f(x+h/2)-f(x-h/2)}{h}+O(h^2) \qquad F(x;h)\equiv\frac{f(x+h/2)-f(x-h/2)}{h}$$

$f'(x)=F(x;h)+C_1h^2+\ldots$ Neglecting other terms, linear function of h^2

Calculate for 2 values of *h* for not too small *h*, draw a straight line.

Don't have to stop there. $f'(x)=F(x;h)+C_1h^2+C_2h^4+C_3h^6+\ldots$

Get enough points, find a polynomial that goes through them, put h=0.

How do we do this in general?

Interpolation and extrapolation

Interpolation

Suppose there is some function f(x), but we only know its values at some points $x_0 < x_1 < ... < x_N$. Can we predict its value at some other point x?

Interpolation, if $x_0 < x < x_N$, extrapolation otherwise. Extrapolation is in general less reliable, but the methods are the same.

Interpolating function. Goes through the specified points with abscissas x_0 , x_1, ..., x_N (called interpolation nodes), i.e., coincides with f(x) at these points.

Another related problem is approximation. Suppose we know the function, but it is too complicated. We want to find a simpler approximate one. Like interpolation, but with the additional freedom that we can choose the interpolation nodes.

Data fitting: want a simple function that approximates a set of data points, but does not necessarily go through them. Will be considered in Comp. Phys. II.

Need to build an interpolating function. Obviously, infinitely many possibilities.

Can restrict based on some knowledge about f(x), or just choose something simple that does the job.

Polynomial interpolation

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$$

n is the order of the polynomial.

Has some nice properties:

1. If we have N+1 points with abscissas x_0 , x_1, ..., x_N , there is always one and only one polynomial of order N that goes through all these points. Then such restricted interpolation problem **is** unique.

Existence: by induction. Suppose there is such polynomial P_{N-1} going through the first N-1 points. Add to it $c(x-x_0)...(x-x_{N-1})$, adjust c.

Uniqueness: suppose there are two, P_N and Q_N. The difference will have N+1 roots, but that is impossible.

2. Weierstrass approximation theorem: any continuous function can be approximated as accurately as desired by a polynomial of a sufficiently high degree.

Note, though, this is a theorem about approximation, not interpolation. Do not get to choose the interpolation nodes.

How does one find the interpolating polynomial?

$$a_n x_0^n + a_{n-1} x_0^{n-1} + \ldots + a_1 x_0 + a_0 = y_0$$
$$a_n x_1^n + a_{n-1} x_1^{n-1} + \ldots + a_1 x_1 + a_0 = y_1$$

.....

$$a_n x_N^n + a_{n-1} x_N^{n-1} + \ldots + a_1 x_N + a_0 = y_N$$

All x and y are known, the unknowns are a_i. A linear system of N+1 eqs. with N+1 unknowns.

The matrix of the coefficients is called the Vandermonde matrix.

Useful, if one really needs the coefficients of the polynomial explicitly, but otherwise, there are much simpler approaches.

Lagrange form of the interpolating polynomial

Consider $(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_N)$. Order N.

Zero at all nodes, except x_i.

$$f(x_i)\frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_N)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_N)}$$ is $f(x_i)$ at x_i, 0 at other nodes

The sum of such terms over all i from 0 to N is the solution, and it is unique.

$$P_N(x)=\sum_{i=0}^{N} f(x_i)\prod_{j\neq i}\frac{x-x_j}{x_i-x_j}.$$

There are other forms that are more convenient in some cases, e.g., Newton's divided differences interpolating polynomial.

Simplest example: linear interpolation

$$P_1(x)=f(x_0)\frac{x-x_1}{x_0-x_1}+f(x_1)\frac{x-x_0}{x_1-x_0}$$

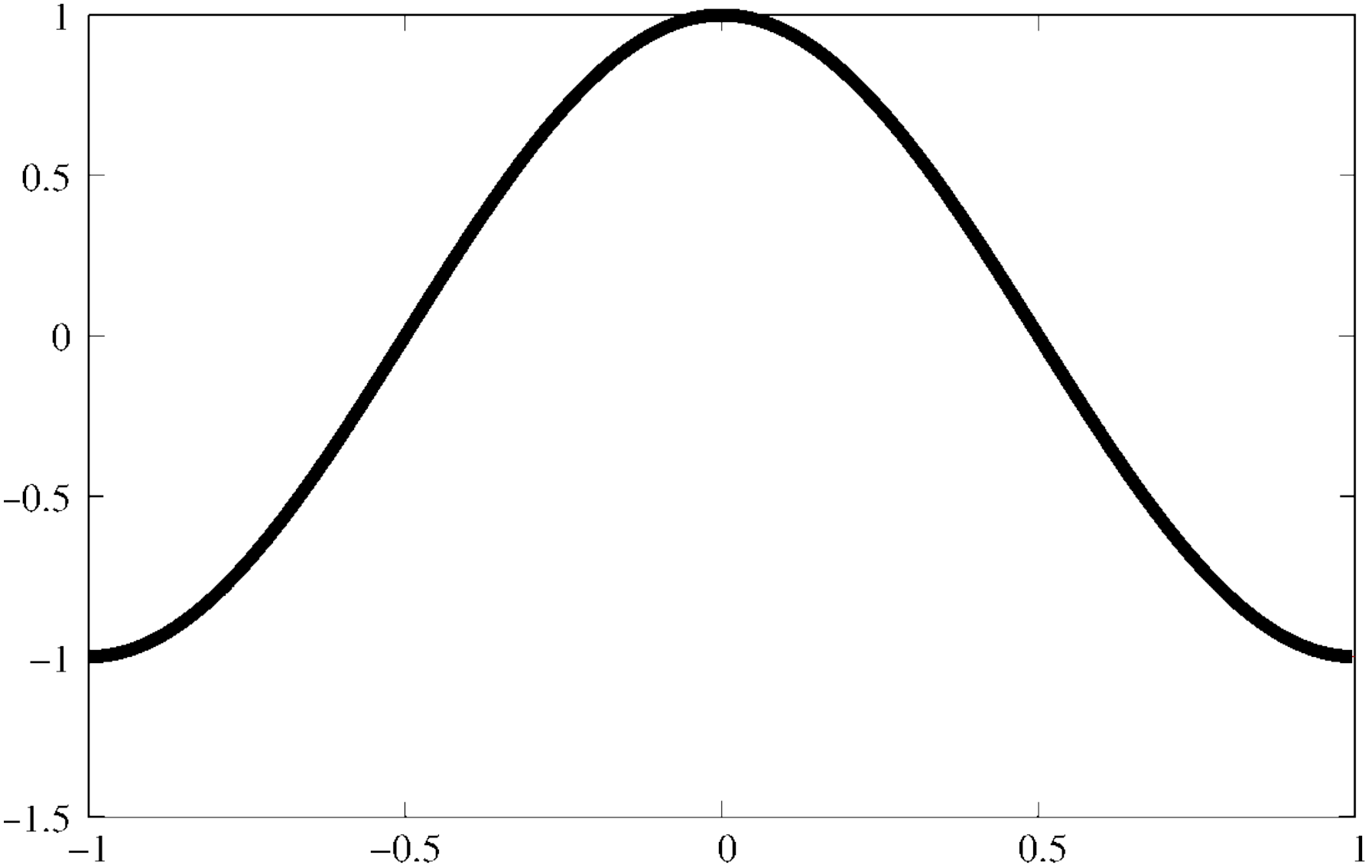
Accuracy

$$f(x)-P_N(x)=\frac{1}{(N+1)!}f^{(N+1)}(\xi_N)\prod_{i=0}^{N}(x-x_i).$$

Suppose N is fixed. If the length of the interval is x_N-x_0=h, then $\prod_{i=0}^{N}(x-x_i)\sim h^{N+1}$

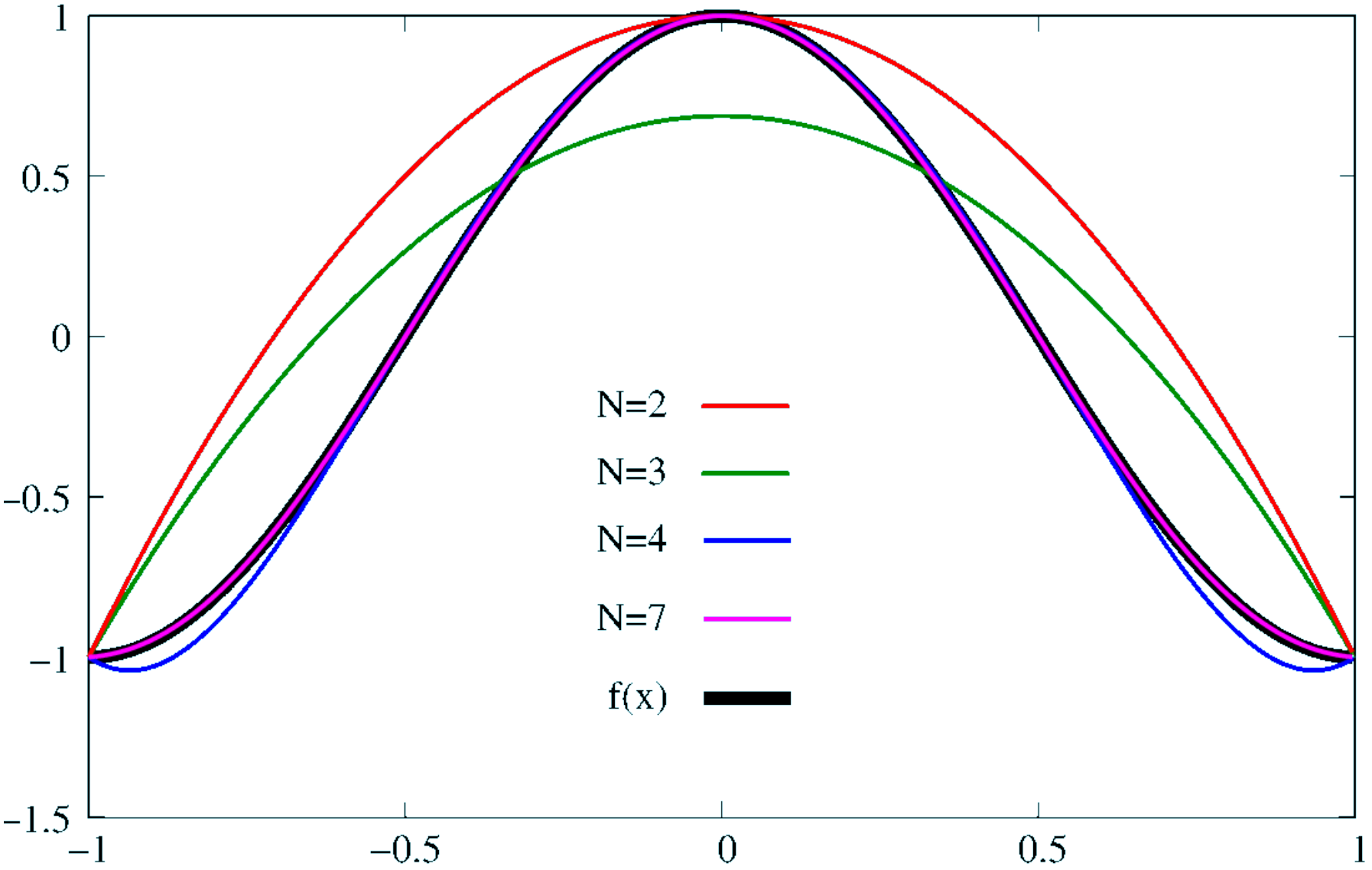
Example 1

$$f(x)=\cos(\pi x),\quad x\in[-1,1]$$



Example 1

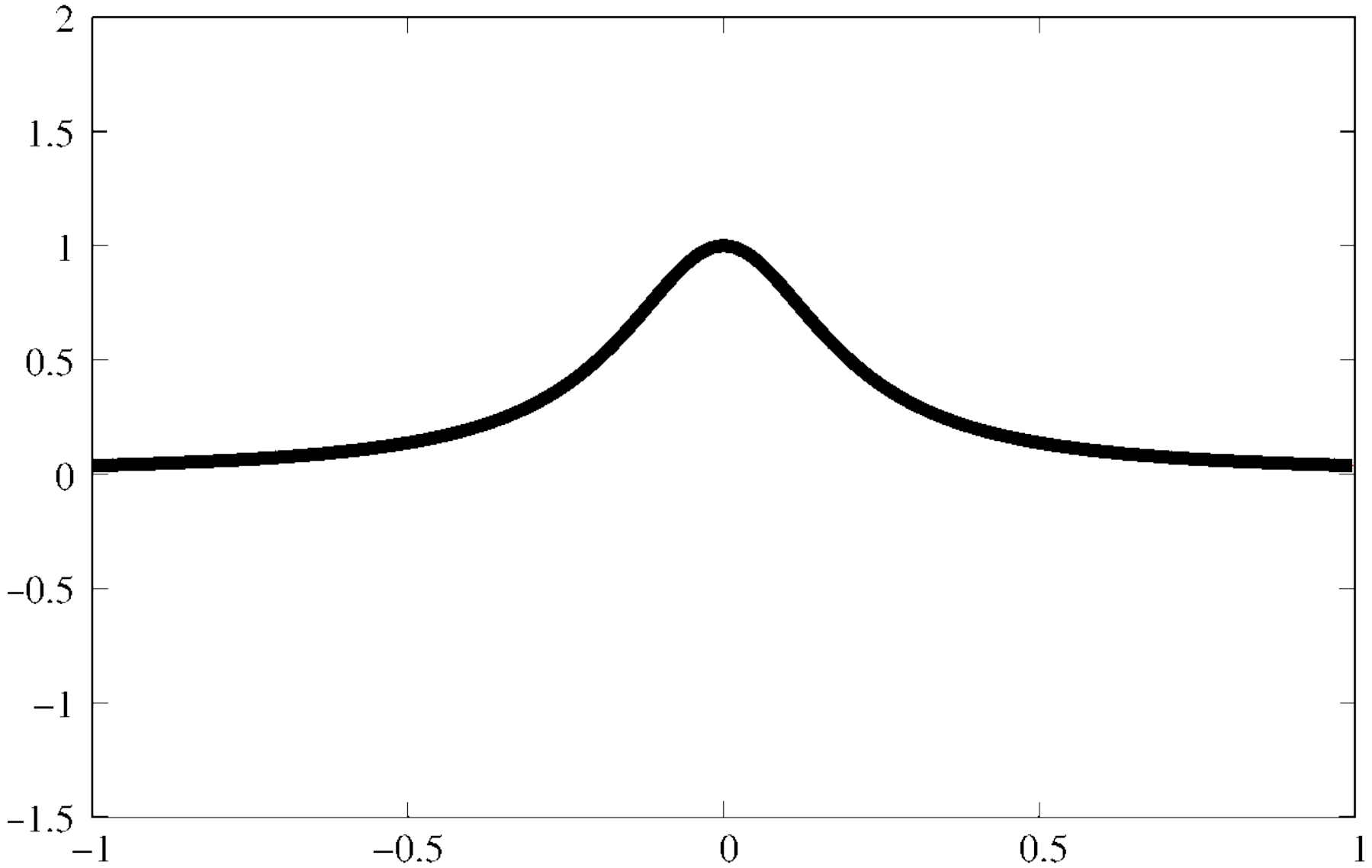
$$f(x)=\cos(\pi x),\quad x\in[-1,1]$$



Converges rapidly

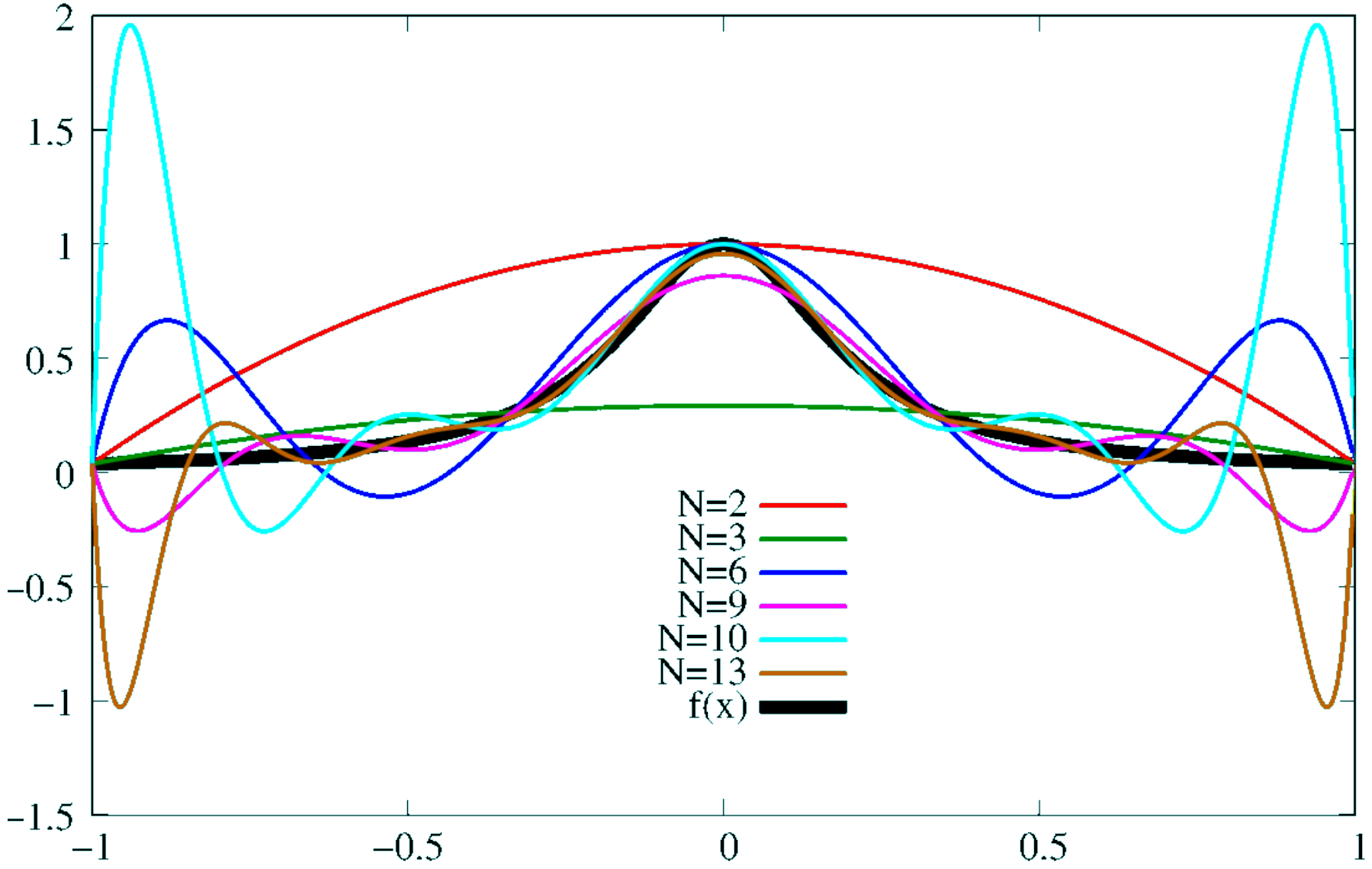
Example 2

$$f(x) = 1/(1+25x^2), \quad x \in [-1,1]$$



Example 2

$$f(x)=1/(1+25x^2),\quad x\in[-1,1]$$



Diverges as N increases. Runge phenomenon.

What is the difference?

$$f(x)-P_N(x)=\frac{1}{(N+1)!}f^{(N+1)}(\xi_N)\prod_{i=0}^{N}(x-x_i).$$

For cos πx, $f^{(N+1)}\sim\pi^{N+1}$ The error decreases rapidly with *N*.

But for $f(x)=1/(1+25x^2),$ there is a factor ~ N!

Essentially, related to the radius of convergence of the Taylor series.

Infinite for cos, but for $1/(1+25x^2)$, there are poles at $\pm i/5$.

Higher-order methods are not necessarily always better than lower-order methods. Only true for small enough steps/intervals, and even then assuming that the function is differentiable enough times, etc.

Can we do better? Yes, in principle, if we are allowed to vary the nodes. The optimal set of nodes is different for each function and impossible to find. But there is a very good "nearly optimal" universal set of nodes, called Chebyshev nodes.

$$f(x)-P_N(x)=\frac{1}{(N+1)!}f^{(N+1)}(\xi_N)\prod_{i=0}^{N}(x-x_i).$$

Minimize the error by adjusting x_i. ξ_N depends on them as well in an unknown way. But can minimize the product – the result is independent of f(x).

For [-1,1], the product is minimized by the roots of Chebyshev polynomials of the 1st kind.

$T_n(x)=\cos(n\arccos x)$ Looks trigonometric, but, in fact, these are polynomials

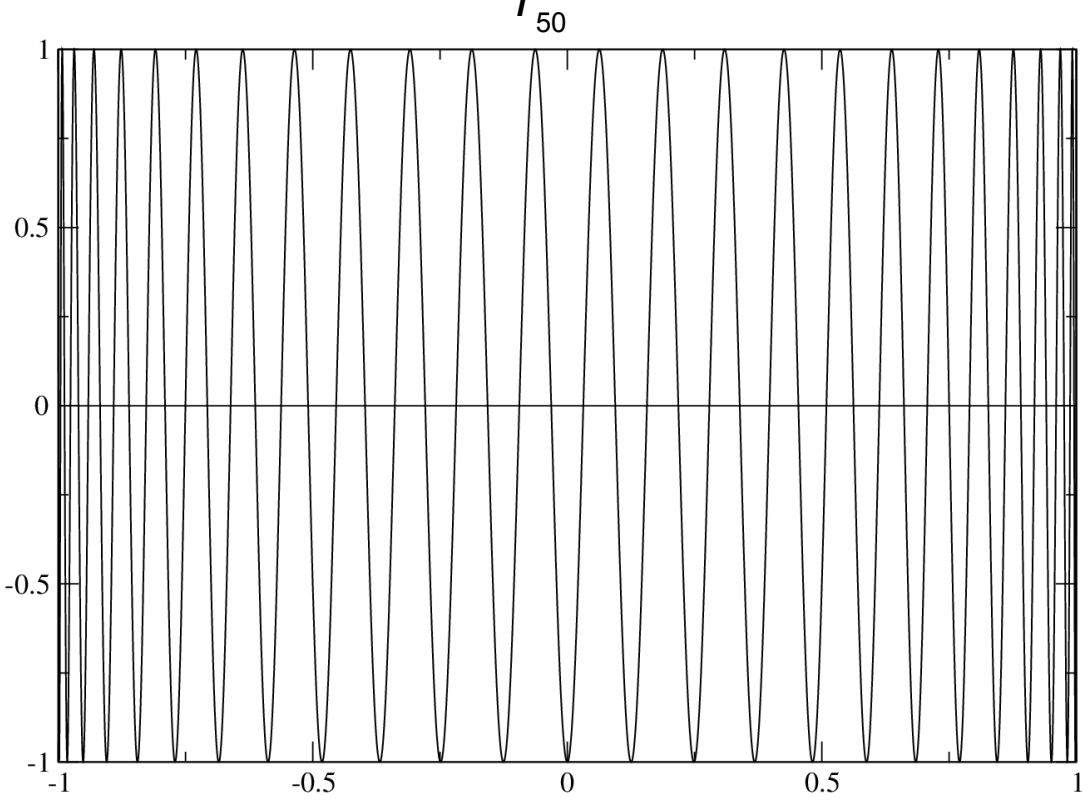
$T_0=1;\ \ T_1=x;\ \ T_{n+1}=2xT_n-T_{n-1}$

Roots

cos(n arccos x) = 0

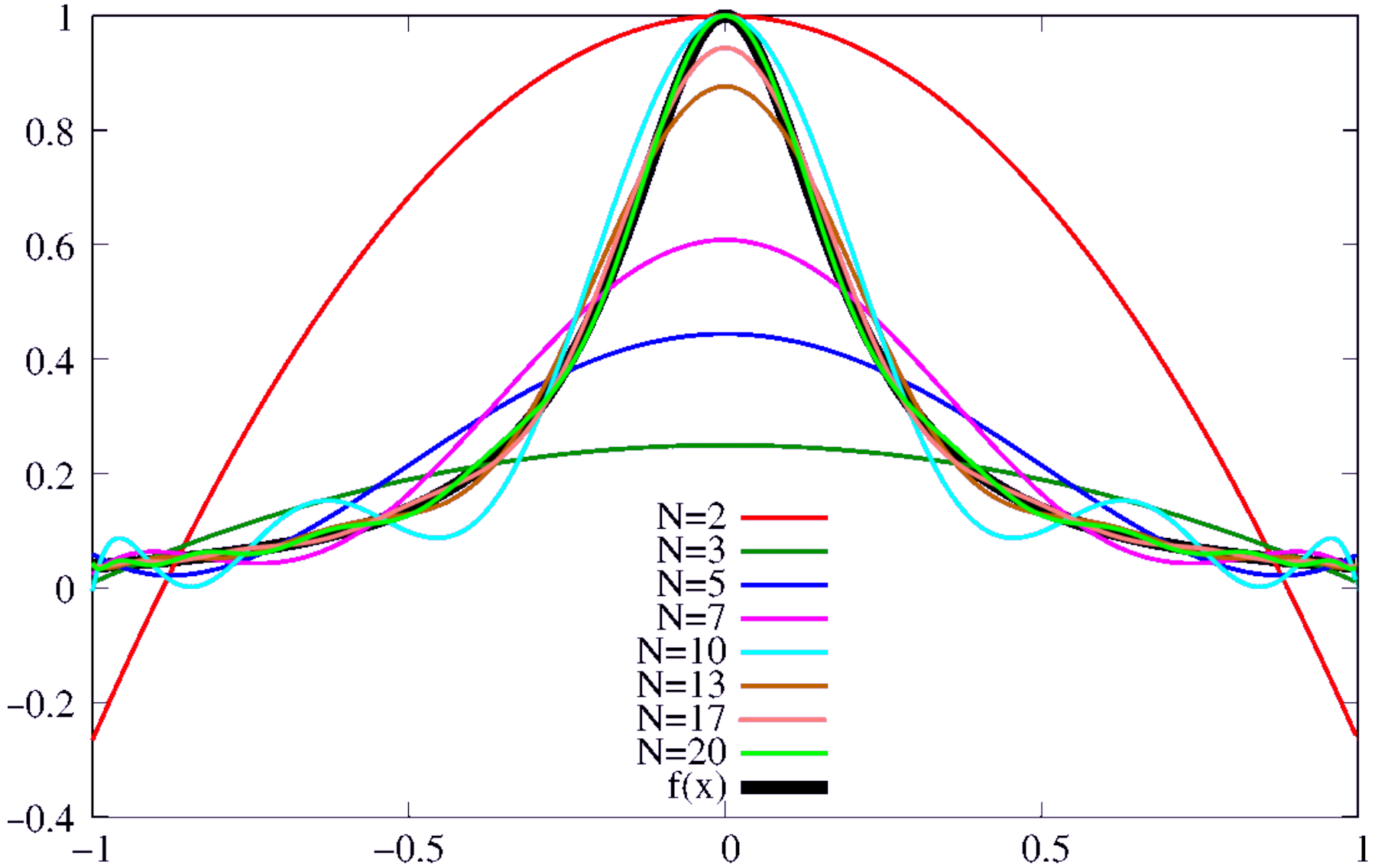
$$x_j=\cos\left(\frac{\pi(2j-1)}{2n}\right),\quad j=1,\ldots,n$$

Denser near the boundaries



Guaranteed to converge for differentiable functions, but may be slow.

E.g., for $f(x) = 1/(1+25x^2)$:



Alternative: piecewise-polynomial interpolation

Linear interpolation between all pairs of adjacent points.

Can use higher-degree pieces by using more points.

But in either case the interpolating function is not smooth.

Spline interpolation

Originally, a flexible strip for drawing smooth curves between given points

A mathematical spline consists of polynomial pieces of degree k and has k-1 continuous derivatives at the nodes.

By far the most popular is cubic spline (k=3).

N+1 nodes, N intervals. 4N unknown coefficients.

At each internal node, 2 conditions for the value of the polynomial (1 left, 1 right) and 2 matching conditions for the derivatives (1^{st} and 2^{nd}). At each of the 2 outer nodes, 1 condition for the value only. 4(N-1)+2=4N-2 conditions.

2 additional conditions. E.g., zero 2^{nd} derivatives at edges (natural spline).

$$s_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i, \quad x \in [x_i, x_{i+1}]$$

Suppose the value at x_i is y_i , second derivative is z_i , and x_{i+1}-x_i=h_i

$$d_i = y_i \qquad\qquad z_i = 2b_i \Rightarrow b_i = z_i/2$$

$$z_{i+1} = 6a_i h_i + 2b_i = 6a_i h_i + z_i \Rightarrow a_i = \frac{z_{i+1} - z_i}{6h_i}.$$

$$d_{i+1} = a_i h_i^3 + b_i h_i^2 + c_i h_i + d_i = \frac{z_{i+1} - z_i}{6} h_i^2 + \frac{z_i}{2} h_i^2 + c_i h_i + y_i = y_{i+1} \Rightarrow$$

$$c_i = \frac{y_{i+1} - y_i}{h_i} - \frac{z_i}{3} h_i - \frac{z_{i+1}}{6} h_i$$

Equate first derivatives at x_i:

$$3a_{i-1} h_{i-1}^2 + 2b_{i-1} h_{i-1} + c_{i-1} = c_i,$$

$$\frac{z_i - z_{i-1}}{2} h_{i-1} + z_{i-1} h_{i-1} + \frac{y_i - y_{i-1}}{h_{i-1}} - \frac{z_{i-1}}{3} h_{i-1} - \frac{z_i}{6} h_{i-1} = \frac{y_{i+1} - y_i}{h_i} - \frac{z_i}{3} h_i - \frac{z_{i+1}}{6} h_i,$$

$$\frac{h_{i-1}}{6} z_{i-1} + \frac{h_{i-1} + h_i}{3} z_i + \frac{h_i}{6} z_{i+1} = \frac{y_{i+1} - y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}$$

A tridiagonal linear system – easy to solve.

Interpolating functions other than (piecewise) polynomial

Rational interpolation. A ratio of 2 polynomials: $\frac{a_m x^m + a_{m-1} x^{m-1} + ... + a_1 x + a_0}{b_n x^n + b_{n-1} x^{n-1} + ... + b_1 x + 1}$

Works better than polynomial for functions with poles.

A related concept is Padé approximation. Suppose you are given a truncated Taylor series. A rational function with the same truncated Taylor series is often a better approximation than the original series.

Multidimensional interpolation: for a rectangular grid, do interpolation along grid lines in one direction, then interpolate in orthogonal direction(s).

http://en.wikipedia.org/wiki/Multivariate_interpolation

Numerical integration

Calculate the definite integral $\int_a^b f(x)\,dx.$ Here a and b can be finite or Infinite, but let's assume for now they are finite.

Riemann's definition: partition the interval $a = x_0 < x_1 < \ldots < x_N = b$.

$$\int_a^b f(x)\,dx = \lim_{N\to\infty} \sum_{i=1}^{N} f(x_i^*)(x_i - x_{i-1}), \quad x_i^* \in [x_{i-1}, x_i]$$

(assumes that the lengths of all subintervals approach 0)

For a finite N, this sum is a numerical method for calculating the integral approximately.

Where is the best location of x_i^* inside the subinterval? Consider a single subinterval.

$$\int_{x_{i-1}}^{x_i} f(x)\,dx \approx f(x_i^*)(x_i - x_{i-1})$$

We want this approximation to be as accurate as possible.

$$\int_{x_{i-1}}^{x_i} f(x)\,dx \approx f(x_i^*)(x_i - x_{i-1})$$

$$f(x) = f(x_i^*) + f'(x_i^*)(x - x_i^*) + f''(x_i^*)(x - x_i^*)^2/2 + O((x - x_i^*)^3)$$

$$\int_{x_{i-1}}^{x_i} f(x)\,dx = \int_{x_{i-1}}^{x_i} f(x_i^*)\,dx + \int_{x_{i-1}}^{x_i} f'(x_i^*)(x - x_i^*)\,dx$$

$$+ \int_{x_{i-1}}^{x_i} [f''(x_i^*)(x - x_i^*)^2/2]\,dx + \int_{x_{i-1}}^{x_i} O((x - x_i^*)^3)\,dx =$$

$$f(x_i^*)(x_i - x_{i-1}) + f'(x_i^*)[(x_i - x_i^*)^2 - (x_{i-1} - x_i^*)^2] + O((x_i - x_{i-1})^3)$$

The second term vanishes when $(x_i - x_i^*)^2 = (x_{i-1} - x_i^*)^2 \Rightarrow x_i^* = [x_{i-1} + x_i]/2$

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{N} f([x_{i-1} + x_i]/2)(x_i - x_{i-1})$$ – midpoint rule

Note the error from one subinterval is cubic. This is the local error. The sign depends on the convexity of the function. Errors from different subintervals add up. Since the number of subintervals is ~1/(width of 1 interval), the total global error is quadratic in the width of the subinterval (or ~$1/N^2$).

Another approach: do piecewise polynomial interpolation and integrate each polynomial piece analytically.

$a = x_0 < x_1 < \ldots < x_N = b$. Simplest: linear interpolation.

For subinterval $[x_{i-1},x_i]$, $f(x) = f(x_{i-1})\frac{x-x_i}{x_{i-1}-x_i} + f(x_i)\frac{x-x_{i-1}}{x_i-x_{i-1}} + \frac{f''(\xi_i)}{2}(x-x_{i-1})(x-x_i)$

$$\int_{x_{i-1}}^{x_i}\left[f(x_{i-1})\frac{x-x_i}{x_{i-1}-x_i} + f(x_i)\frac{x-x_{i-1}}{x_i-x_{i-1}} + \frac{f''(\xi_i)}{2}(x-x_{i-1})(x-x_i)\right]dx =$$

$$[f(x_{i-1})+f(x_i)](x_i-x_{i-1})/2 + f''(\tilde{\xi}_i)O([x_i-x_{i-1}]^3).$$

$$\int_a^b f(x)\,dx = \sum_{i=1}^{N}[f(x_{i-1})+f(x_i)](x_i-x_{i-1})/2 + O(1/N^2)$$

If all the subintervals are equal ($x_i - x_{i-1} = h$),

$$\int_a^b f(x)\,dx = \sum_{i=1}^{N}[f(x_{i-1})+f(x_i)]h/2 + O(h^2) =$$

$$\left[\frac{1}{2}f(x_0)+f(x_1)+f(x_2)+\ldots f(x_{N-2})+f(x_{N-1})+\frac{1}{2}f(x_N)\right]h + O(h^2)$$