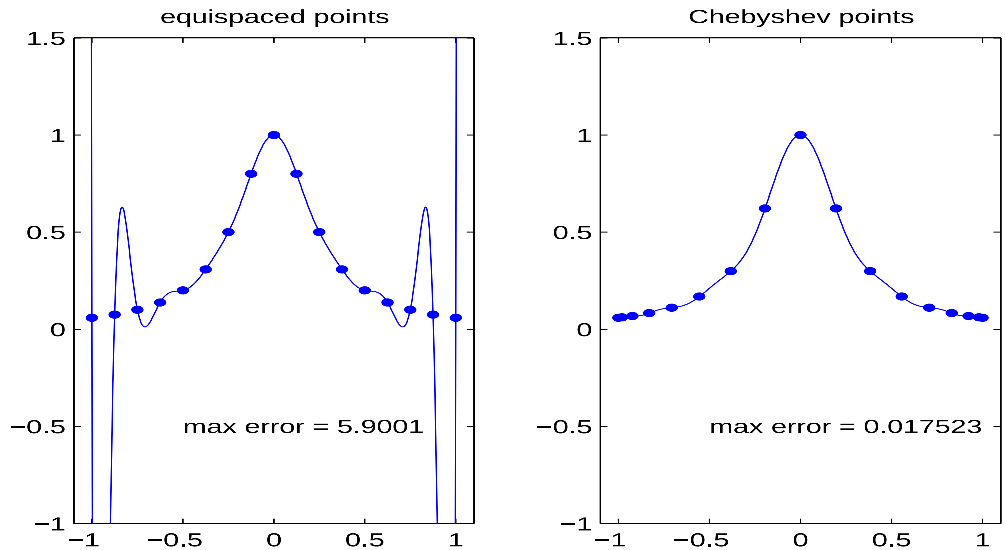
2, Interpolation, Curve Fitting, and Integration



- Polynomial interpolation and extrapolation
- Chebyshev approximation
- Padé approximation
- Fast Fourier Transform and Discrete Fourier Transform
- Trapezoidal and Simpson method for integration

EE.NTHU 國立清華大學 電機工程學系及研究所

Finite difference approximation

Second-order FD approximation for $u'(x_j)$

$$u'(x_j) \approx \frac{u_{j+1} - u_{j-1}}{2h}$$

in the matrix-vector form (with periodic boundary)

$$\begin{pmatrix} u_1' \\ \vdots \\ \\ u_j' \\ \vdots \\ \\ u_N' \end{pmatrix} = h^{-1} \begin{pmatrix} 0 & \frac{1}{2} & & & & & -\frac{1}{2} \\ -\frac{1}{2} & 0 & & & & & \\ & & & \ddots & & & \\ & & -\frac{1}{2} & 0 & \frac{1}{2} & & \\ & & & \ddots & & & \\ & & & & & 0 & \frac{1}{2} \\ \frac{1}{2} & & & & & -\frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ \\ u_j \\ \vdots \\ \\ u_N \end{pmatrix}$$

EE.NTHU 國立清華大學 電機工程學系及研究所

Finite difference approximation

Taylor's expansion for $u(x)$ at x_{j+1},

$$u(x_{j+1}) = u(x_j)+u'(x_j)(x_{j+1}-x_j)+\frac{u''(x_j)}{2}(x_{j+1}-x_j)^2+\frac{u'''(x_j)}{3!}(x_{j+1}-x_j)^3+\ldots,$$

one can approximate $u'(x_j)$ by

$$\begin{aligned} u'(x_j) &= \frac{u(x_{j+1})-u(x_j)}{x_{j+1}-x_j} - \frac{u''(x_j)}{2}(x_{j+1}-x_j) - \frac{u'''(x_j)}{3!}(x_{j+1}-x_j)^2 + \ldots, \\ &\approx \frac{u(x_{j+1})-u(x_j)}{x_{j+1}-x_j} + \mathbf{O}(\Delta x), \end{aligned}$$

by the same way the Taylor's expansion for $u(x)$ at x_{j-1},

$$u(x_{j-1}) = u(x_j)+u'(x_j)(x_{j-1}-x_j)+\frac{u''(x_j)}{2}(x_{j-1}-x_j)^2+\frac{u'''(x_j)}{3!}(x_{j-1}-x_j)^3+\ldots,$$

one can approximate $u'(x_j)$ by

$$\begin{aligned} u'(x_j) &= \frac{u(x_j)-u(x_{j-1})}{x_j-x_{j-1}} - \frac{u''(x_j)}{2}(x_j-x_{j-1}) + \frac{u'''(x_j)}{3!}(x_j-x_{j-1})^2 + \ldots, \\ &\approx \frac{u(x_j)-u(x_{j-1})}{x_j-x_{j-1}} + \mathbf{O}(\Delta x), \end{aligned}$$

Finite difference approximation

combine

$$u(x_{j+1}) = u(x_j) + u'(x_j)\Delta x + \frac{u''(x_j)}{2}(\Delta x)^2 + \frac{u'''(x_j)}{3!}(\Delta x)^3 + \dots,$$
$$u(x_{j-1}) = u(x_j) - u'(x_j)\Delta x + \frac{u''(x_j)}{2}(\Delta x)^2 - \frac{u'''(x_j)}{3!}(\Delta x)^3 + \dots,$$

one can approximate $u'(x_j)$ by

$$u'(x_j) = \frac{u(x_{j+1}) - u(x_{j-1})}{2\Delta x} - \frac{u'''(x_j)}{2 * 3!}(\Delta x)^2 + \dots,$$
$$\approx \frac{u(x_{j+1}) - u(x_{j-1})}{2\Delta x} + \mathbf{O}(\Delta x^2),$$

- 4th-order approximation
- Runge-Kutta method
- differential matrix

By local interpolation

For $j = 1, 2, \ldots, N$:

- Let p_j be the unique polynomial of degree ≤ 2 with $p_j(x_{j-1}) = u_{j-1}$,$p_j(x_j) = u_j$, and $p_j(x_{j+1}) = u_{j+1}$.
- $u'(x_j) = p_j'(x)$.

Then the interpolant p_j is given by

$$p_j(x) = u_{j-1}a_{-1}(x) + u_j a_0(x) + u_{j+1}a_1(x)$$

where

$$\begin{aligned} a_{-1}(x) &= (x - x_j)(x - x_{j+1})/2h^2 \\ a_0(x) &= -(x - x_{j-1})(x - x_{j+1})/2h^2 \\ a_1(x) &= (x - x_{j-1})(x - x_j)/2h^2 \end{aligned}$$

Generalization to fourth-order

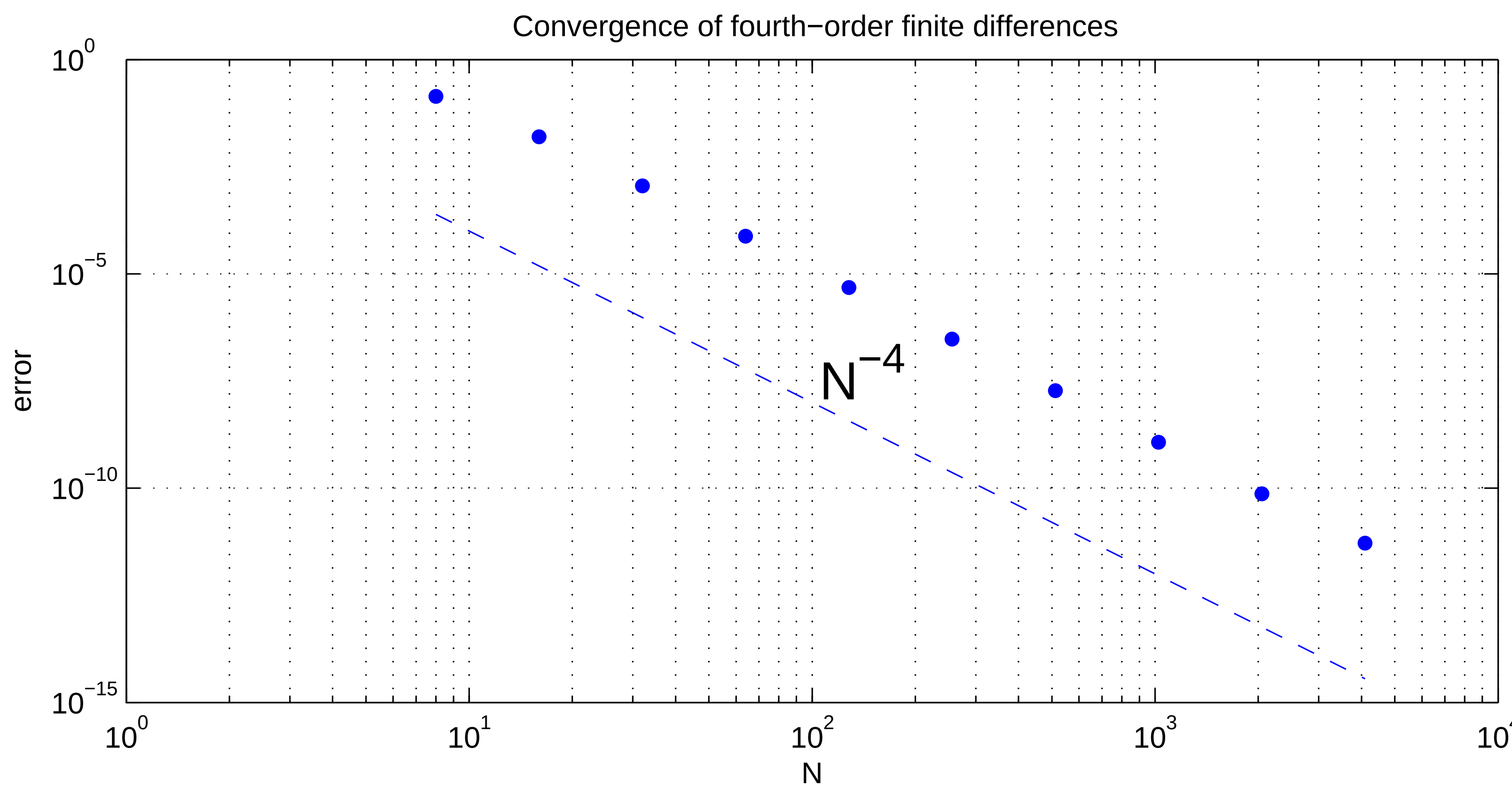
For $j = 1, 2, \ldots, N$:

- Let p_j be the unique polynomial of degree ≤ 2 with $p_j(x_{j\pm1}) = u_{j\pm1}$,$p_j(x_j) = u_j$, and $p_j(x_{j\pm2}) = u_{j\pm2}$.
- $u'(x_j) = p'_j(x)$.

$$\begin{pmatrix} u'_1 \\ \vdots \\ \\ u'_j \\ \vdots \\ \\ u'_N \end{pmatrix} = h^{-1} \begin{pmatrix} & & \ddots & & \\ & \ddots & -\frac{1}{12} & \ddots & \\ & \ddots & \frac{2}{3} & \ddots & \\ \frac{1}{12} & -\frac{2}{3} & 0 & \frac{2}{3} & -\frac{1}{12} \\ & \ddots & -\frac{2}{3} & \ddots & \\ & \ddots & \frac{1}{12} & \ddots & \\ & & \ddots & & \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ \\ u_j \\ \vdots \\ \\ u_N \end{pmatrix}$$

Test of the convergence: FD

$$u(x) = e^{\sin(x)}, Du(x) = u'(x).$$



Polynomial interpolation

For a given set of $N+1$ dat points

$$\{(x_0, y_0), (x_1, y_1), \ldots, (x_N, y_N)\},$$

we want to find the coefficients of an Nth-degree polynomial function to match them:

$$P_N(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_N x^N,$$

The coefficients can be obtained by solving the following system of linear equations,

$$\begin{aligned}
a_0 + x_0 a_1 + x_0^2 a_2 + \cdots + x_0^N a_N &= y_0, \\
a_0 + x_1 a_1 + x_1^2 a_2 + \cdots + x_1^N a_N &= y_1, \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\
a_0 + x_N a_1 + x_N^2 a_2 + \cdots + x_N^N a_N &= y_N.
\end{aligned}$$

Lagrange polynomial

Lagrange polynomial

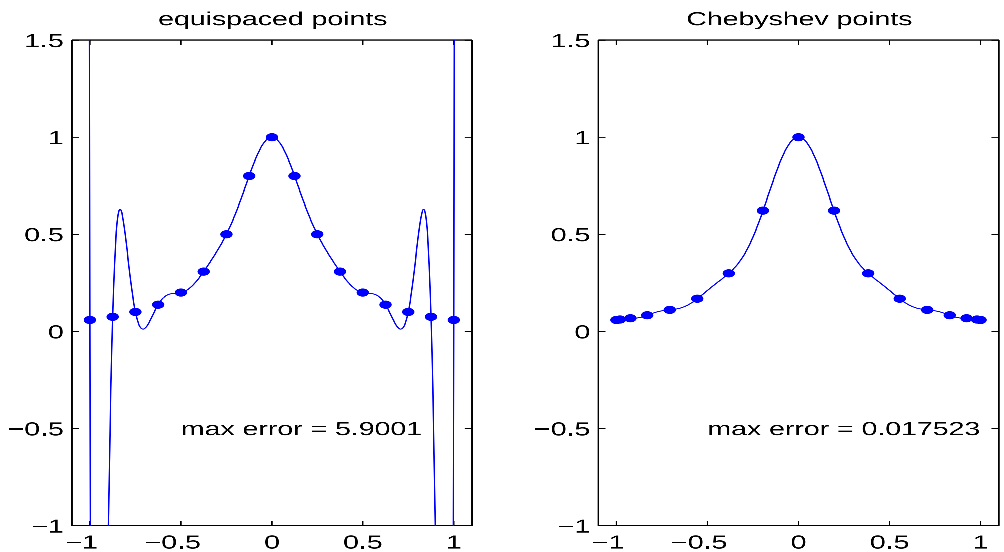
$$L_N(x) = y_0\frac{(x-x_1)(x-x_2)\cdots(x-x_N)}{(x_0-x_1)(x_0-x_2)\cdots(x_0-x_N)} + y_1\frac{(x-x_0)(x-x_2)\cdots(x-x_N)}{(x_1-x_0)(x_1-x_2)\cdots(x_1-x_N)}$$
$$+\cdots+y_N\frac{(x-x_0)(x-x_1)\cdots(x-x_{N-1})}{(x_N-x_0)(x_N-x_1)\cdots(x_N-x_{N-1})}$$

or

$$L_N(x) = \sum_{m=0}^{N} y_m L_{N,m}(x), \quad \text{with} \quad L_{N,m} = \prod_{k\neq m}^{N} \frac{x-x_k}{x_m-x_k}$$

```
function [L,LNm]=lagranp(x,y)
%Input : x=[x0 x1 ... xN], y=[y0 y1 ... yN]
N= length(x)-1; L=0; %the order of polynomial
for m=1:N+1
   P=1;
   for k=1:N+1
     if k~=m
      P=conv(P,poly(x(k)))/(x(m)-x(k));
     end
   end
   LNm(m,:)=P; L =L+y(m)*P; %Lagrange coefficient polynomial
end
```

Polynomial wiggle and Runge Phenomenon



increasing the degree of polynomial contributes little to reducing the approximation error,

- polynomial wiggle: the oscillation becomes large
- Runge phenomenon: the error gets bigger

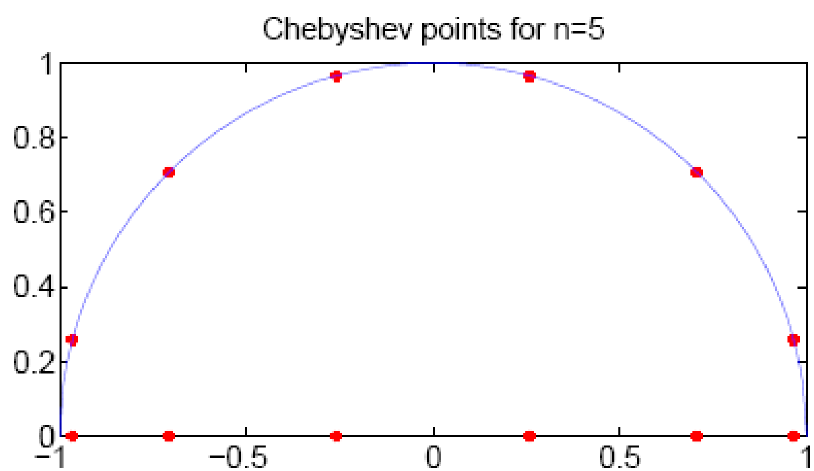
polynomials of degree 5 or above are seldom used.

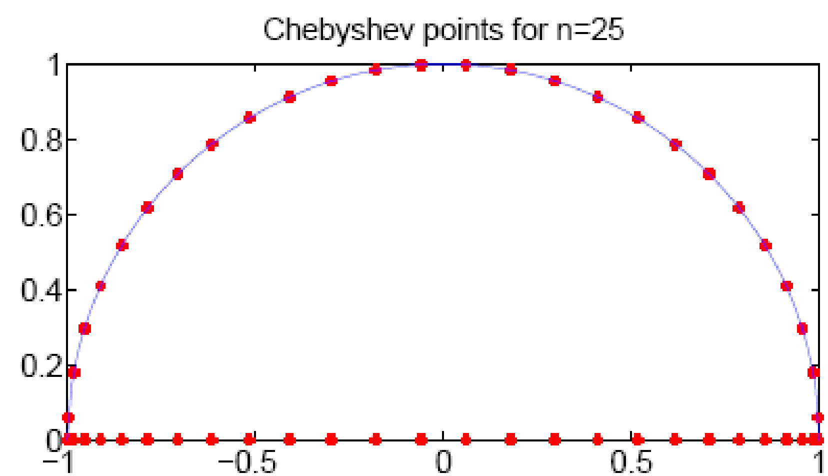
Chebyshev polynomial

in the normalized interval $[-1, +1]$,

$$x_j = \cos \frac{j\pi}{N}, \quad \text{for} \quad j = 0, 1, \ldots, N$$

there points are called *Chebyshev points*



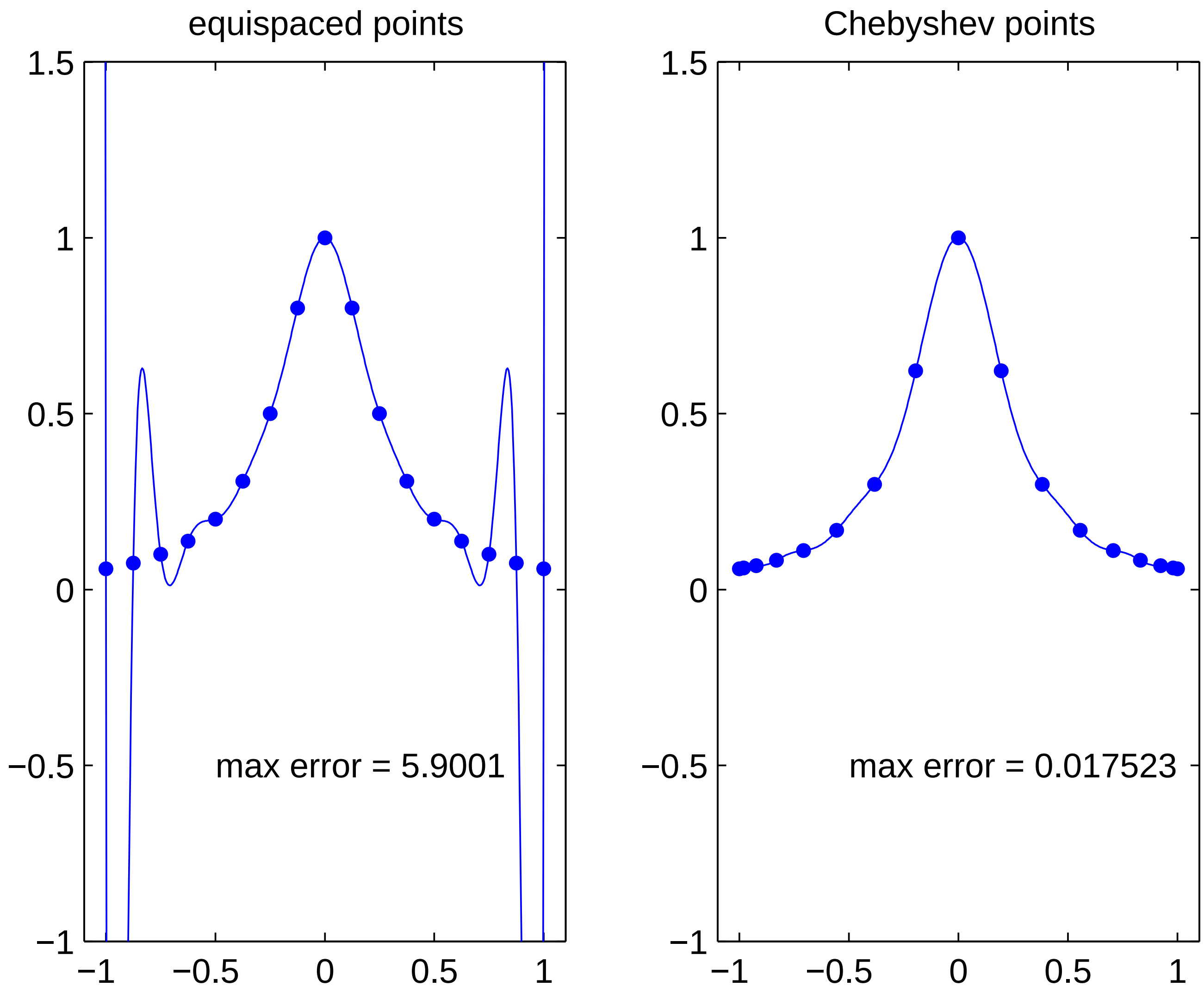


Example: Interpolation

$$u(x) = \frac{1}{1 + 16x^2}$$

```
N = 16;
xx = -1.01:.005:1.01; clf
for i = 1:2
  if i==1, s = 'equispaced points'; x = -1 + 2*(0:N)/N; end
  if i==2, s = 'Chebyshev points';  x = cos(pi*(0:N)/N); end
  subplot(1,2,i)
  u = 1./(1+16*x.^2);
  uu = 1./(1+16*xx.^2);
  p = polyfit(x,u,N);                     % interpolation
  pp = polyval(p,xx);                     % evaluation of interpolant
  plot(x,u,'.','markersize',13)
  line(xx,pp)
  axis([-1.1 1.1 -1 1.5]), title(s)
  error = norm(uu-pp,inf);
  text(-.5,-.5,['max error = ' num2str(error)])
end
```

Example: Interpolation



Padé approximation

A Padé approximation (of a specific order) tries to approximate a function $f(x)$ around a point x_0 by a rational function

$$P_{M,N}(x - x_0) = \frac{Q_M(x - x_0)}{D_N(x - x_0)} \quad \text{with} \quad M = N \quad \text{or} \quad M = N + 1$$

$$= \frac{q_0 + q_1(x - x_0) + q_2(x - x_0)^2 + \cdots + q_M(x - x_0)^M}{1 + d_1(x - x_0) + d_2(x - x_0)^2 + \cdots + d_M(x - x_0)^N}$$

where $f(x_0), f'(x_0), f^{(2)}(x_0), \ldots, f^{(M+N)}(x_0)$ are known.
Taylor expansion of $f(x)$ at $x = x_0$ to degree $M + N$ is

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{f^{(2)}(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(M+N)}(x_0)}{(M+N)!}(x - x_0)^{M+N}$$

$$= a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \cdots + a_{M+N}(x - x_0)^{M+N}$$

$$= \frac{Q_M(x - x_0)}{D_N(x - x_0)}$$

$$= \frac{q_0 + q_1(x - x_0) + q_2(x - x_0)^2 + \cdots + q_M(x - x_0)^M}{1 + d_1(x - x_0) + d_2(x - x_0)^2 + \cdots + d_M(x - x_0)^N}$$

Padé approximation

One can find the coefficients of $Q_M(x-x_0)$ and $D_N(x-x_0)$,

$$
\begin{aligned}
&[a_0 + a_1(x-x_0) + a_2(x-x_0)^2 + \cdots + a_{M+N}(x-x_0)^{M+N}] \\
&\times[1 + d_1(x-x_0) + d_2(x-x_0)^2 + \cdots + d_M(x-x_0)^N] \\
&= [q_0 + q_1(x-x_0) + q_2(x-x_0)^2 + \cdots + q_M(x-x_0)^M]
\end{aligned}
$$

by solving the following equations, assume $x_0 = 0$:

$$
\begin{array}{llllll}
a_0 & & & & & = q_0 \\
a_1 & +a_0 d_1 & & & & = q_1 \\
a_2 & +a_1 d_1 & +a_0 d_2 & & & = q_2 \\
\cdots & \cdots & \cdots & \cdots & & \cdots \\
a_M & +a_{M-1} d_1 & +a_{M-2} d_2 & \cdots & +a_{M-N} d_N & = q_M \\
a_{M+1} & +a_M d_1 & +a_{M-1} d_2 & \cdots & +a_{M-N+1} d_N & = 0 \\
a_{M+2} & +a_{M+1} d_1 & +a_M d_2 & \cdots & +a_{M-N+2} d_N & = 0 \\
\cdots & \cdots & \cdots & \cdots & & \cdots \\
a_{M+N} & +a_{M+N-1} d_1 & +a_{M+N-2} d_2 & \cdots & +a_M d_N & = 0
\end{array}
$$

Example: Padé approximation

Padé approximation for $f(x) = e^x$ at $x_0 = 0$ with $M = 3, N = 2$:

- Taylor expansion at $x = 0$ to degree $M + N = 5$,

$$\begin{aligned} f(x) &= e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 + \cdots \\ &\approx a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 \end{aligned}$$

- solve

$$\begin{aligned} a_4 + a_3 d_1 + a_2 d_2 &= 0, \\ a_3 + a_2 d_1 + a_1 d_2 &= 0, \end{aligned}$$

  one can find the solutions $d_1 = -2/5$ and $d_2 = 1/20$.

- solve the coefficients for q_i, $i = 0, 1, 2, 3$,

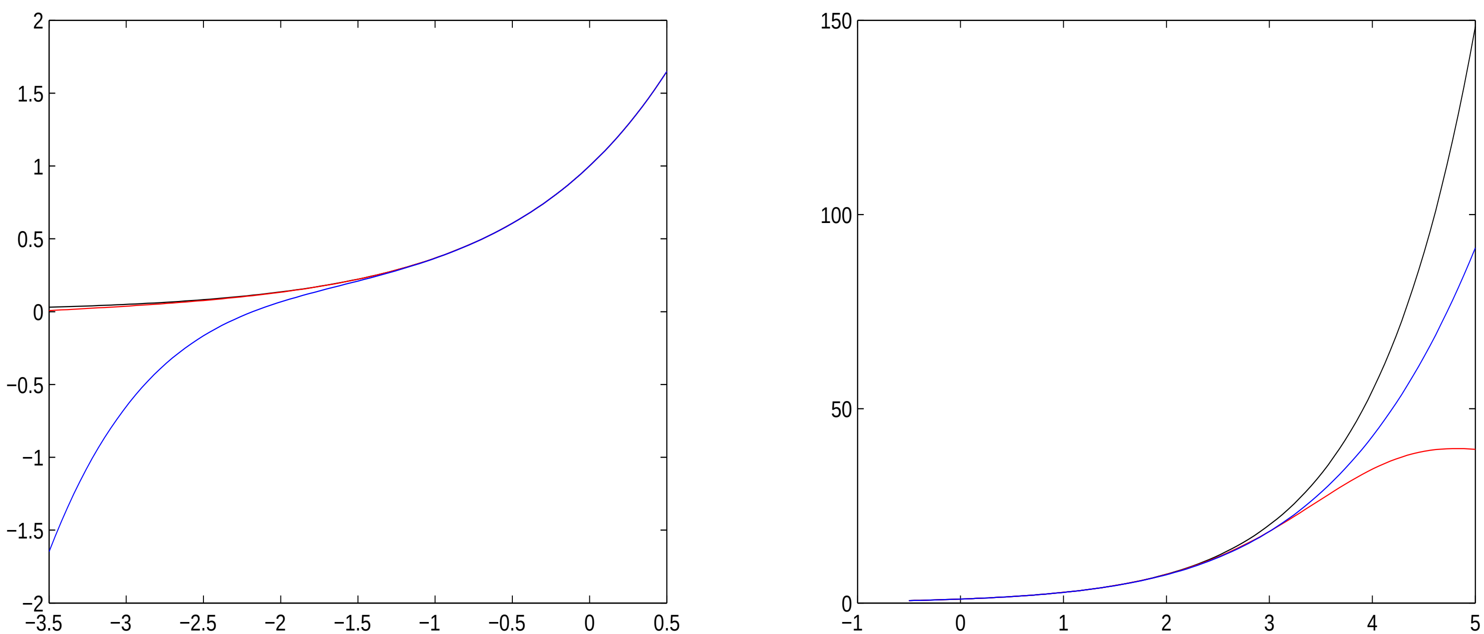
$$\begin{aligned} q_0 &= 1, \quad q_1 = 3/5 \\ q_2 &= 3/20, \quad q_3 = 1/60 \end{aligned}$$

Example: Padé approximation

Padé approximation for $f(x) = e^x$ at $x_0 = 0$:

$$p_{3,2} = \frac{q_0 + q_1 x + q_2 x^2 + q_3 x^3}{1 + d_1 x + d_2 x^2},$$

$$= \frac{(1 + (3/5)x + (3/20x^2 + (1/60)x^3}{1 + (-2/5)x + (1/20)x^2}$$



black: e^x; red: $P_{3,2}(x)$; blue: Taylor expansion to degree 5.

Padé approximation in Beam Propagation method

Wave equation:

$$\frac{\partial u}{\partial z} = i\bar{k}(\sqrt{1+P}-1)u \approx \frac{N_m(P)}{D_n(P)},$$

where P is an operator defined by

$$P \equiv \frac{1}{\bar{k}^2}[\frac{\partial^2}{\partial x^2} + (k^2 - \bar{k}^2)].$$

Padé rational expansion:

$$\sqrt{1+P}-1 = \frac{N_m(P)}{D_n(P)},$$

Padé approximants are given by

$$\text{Padé}(1,0): \quad \frac{N_1(P)}{D_0(P)} = \frac{\frac{P}{2}}{1},$$

$$\text{Padé}(1,1): \quad \frac{N_1(P)}{D_1(P)} = \frac{\frac{P}{2}}{1+\frac{P}{4}},$$

$$\text{Padé}(2,2): \quad \frac{N_2(P)}{D_2(P)} = \frac{\frac{P}{2}+\frac{P^2}{4}}{1+\frac{3P}{4}+\frac{P^2}{16}}.$$

EE.NTHU 國立清華大學 電機工程學系及研究所

Two-dimensional interpolation

- bilinear interpolation in 1D: given points $(x_{m-1}, f(x_{m-1}))$ and $(x_m, f(x_m))$,

$$f(x) = \frac{x - x_m}{x_{m-1} - x_m} f(x_{m-1}) + \frac{x - x_{m-1}}{x_m - x_{m-1}} f(x_m)$$

- bilinear interpolation in 2D:

$$\begin{aligned} z(x,y) &= \frac{y - y_n}{y_{n-1} - y_n} z(x, y_{n-1}) + \frac{y - y_{n-1}}{y_n - y_{n-1}} z(x, y_n) \\ &= \frac{y - y_n}{y_{n-1} - y_n} [\frac{x - x_n}{x_{n-1} - x_n} z(x_{n-1}, y_{n-1}) + \frac{x_n - x_{n-1}}{x_n - x_{n-1}} z(x_n, y_{n-1})] \\ &+ \frac{y - y_{n-1}}{y_n - y_{n-1}} [\frac{x - x_n}{x_{n-1} - x_n} z(x_{n-1}, y_n) + \frac{x - x_{n-1}}{x_n - x_n} z(x_n, y_{n-1})] \end{aligned}$$

Numerical integration

The numerical integration of a function $f(x)$ over some interval $[a,b]$ is a weighted sum of the sunction values at the sample points (nodes),

$$\int_a^b f(x)\mathrm{d}x \approx \sum_{k=0}^{N} w_k f(x_k), \quad \text{with} \quad a = x_0 < x_1 \cdots < x_N = b,$$

- midpoint rule:

$$\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x \approx \Delta x f(\frac{x_k + x_{k+1}}{2}),$$

where $\Delta x = x_{k+1} - x_k$

EE.NTHU 國立清華大學 電機工程學系及研究所

Trapezoidal and Simpson's method for integration

- trapezoidal rule:

$$\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x \approx \frac{\Delta x}{2}[f(x_k) + f(x_{k+1})] + \mathbf{O}(\Delta x^3),$$

- Simpson's rule:

$$\int_{x_{k-1}}^{x_{k+1}} f(x)\mathrm{d}x \approx \frac{\Delta x}{3}[f(x_{k-1})+4f(x_k)+f(x_{k+1})]+\mathbf{O}(\Delta x^5),$$

Simpson's method for integration

Change the interval $x = \{x_k - \Delta x, x_k, x_k + \Delta_x\}$ to $t = \{-\Delta x, 0, \Delta x\}$ by $t = x - x_k$, then the second-degree polynomial to fit $f(t)$ is

$$P_2(t) = c_2 t^2 + c_1 t + c_0,$$

with the coefficients

$$\begin{aligned} c_0 &= f_k, \\ c_1 &= \frac{f_{k+1} - f_{k-1}}{2\Delta x}, \\ c_2 &= \frac{1}{2\Delta x^2}(f_{k+1} + f_{k-1} - 2f_k). \end{aligned}$$

Integrate the second-degree polynomial $f(x)$ from $t = -\Delta x$ to $t = \Delta x$,

$$\int_{-\Delta x}^{\Delta x} P_2(t)\mathrm{d}t = \frac{\Delta x}{3}(f_{k-1} + 4f_k + f_{k+1}).$$

This is the Simpson integration formula.

Taylor's expansion of integration

For

$$
\begin{aligned}
g(x) &= \int_{x_k}^{x} f(t)\mathrm{d}t, \\
&= g(x_k) + g'(x_k)\Delta x + \frac{1}{2!}g^{(2)}(x_k)\Delta x^2 + \frac{1}{3!}g^{(3)}(x_k)\Delta x^3 + \cdots \\
&= g(x_k) + f(x_k)\Delta x + \frac{1}{2!}f'(x_k)\Delta x^2 + \frac{1}{3!}f^{(2)}(x_k)\Delta x^3 + \cdots
\end{aligned}
$$

where $(x - x_k) = \Delta x$. Similarily

$$
\begin{aligned}
\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x &= g(x_k) + f(x_k)\Delta x + \frac{1}{2!}f'(x_k)\Delta x^2 + \frac{1}{3!}f^{(2)}(x_k)\Delta x^3 + \cdots \\
\int_{x_k}^{x_{k-1}} f(x)\mathrm{d}x &= g(x_k) - f(x_k)\Delta x + \frac{1}{2!}f'(x_k)\Delta x^2 - \frac{1}{3!}f^{(2)}(x_k)\Delta x^3 + \cdots ,
\end{aligned}
$$

where $g(x_k) = 0$. Then

$$
\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x + \int_{x_{k-1}}^{x_k} f(x)\mathrm{d}x = 2[f(x_k)\Delta x + \frac{1}{3!}f^{(2)}(x_k)\Delta x^3 + \frac{1}{5!}f^{(4)}(x_k)\Delta x^5 + \cdots]
$$

EE.NTHU 國立清華大學 電機工程學系及研究所

Taylor's expansion of integration

$$\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x = f(x_k)\Delta x + \frac{1}{2^2 \cdot 3!} f^{(2)}(x_k)\Delta x^3 + \frac{1}{2^4 \cdot 5!} f^{(4)}(x_k)\Delta x^5 + \cdots]$$

- The error of trapezoidal rule:

$$\int_{x_k}^{x_{k+1}} f(x)\mathrm{d}x - \frac{\Delta x}{2}[f(x_k) + f(x_{k+1})] = \mathbf{O}(\Delta x^3) + \ldots,$$

- The error of Simpson's rule:

$$\int_{x_{k-1}}^{x_{k+1}} f(x)\mathrm{d}x - \frac{\Delta x}{3}[f(x_{k-1}) + 4f(x_k) + f(x_{k+1})] = \mathbf{O}(\Delta x^5) + \ldots$$

Fourier transform

Fourier transform equation

$$\mathbf{H}(f) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f t} \mathrm{d}t,$$
$$h(t) = \int_{-\infty}^{\infty} \mathbf{H}(f) e^{-2\pi i f t} \mathrm{d}f.$$

with $\omega = 2\pi f$, one has

$$\mathbf{H}(\omega) = \int_{-\infty}^{\infty} h(t) e^{i\omega t} \mathrm{d}t,$$
$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathbf{H}(\omega) e^{-i\omega t} \mathrm{d}\omega.$$

Properties of Fourier transform

- if $h(t)$ is real, then $\mathbf{H}(-\omega) = \mathbf{H}^*(\omega)$.
- if $h(t)$ is even, then $\mathbf{H}(-\omega) = \mathbf{H}(\omega)$, i.e., $\mathbf{H}(\omega)$ is even.
- if $h(t)$ is odd, then $\mathbf{H}(-\omega) = -\mathbf{H}(\omega)$, i.e., $\mathbf{H}(\omega)$ is odd.
- Time scaling, $h(a\,t) \leftrightarrow \frac{1}{|a|}\mathbf{H}(\frac{\omega}{a})$.
- Frequency scaling, $\frac{1}{|b|}h(\frac{t}{b}) \leftrightarrow \mathbf{H}(b\,\omega)$.
- Time shifting, $h(t-t_0) \leftrightarrow \mathbf{H}(\omega)e^{i\omega t_0}$.
- Frequency shifting, $h(t)e^{-i\omega_0 t} \leftrightarrow \mathbf{H}(\omega-\omega_0)$.

Convolution theorem A convolution of two time functions, $f(t)$ and $g(t)$, is defined,

$$g \otimes f = \int_{-\infty}^{\infty} g(\tau) f(t-\tau) \mathrm{d}\tau,$$

the Fourier transform of the convolution is

$$\int_{-\infty}^{\infty} g \otimes f \, e^{i\omega t} \mathrm{d}t = \mathbf{G}(\omega)\mathbf{F}(\omega).$$

EE.NTHU 國立清華大學 電機工程學系及研究所

Sampling theorem and aliasing

- Sampled data:

$$h_n = h(n\Delta), \quad n = \ldots, -3, -2, -1, 0, 1, 2, 3, \ldots$$

where the reciprocal of the time interval Δ is called the *sampling rate*.

- Sampling theorem and Aliasing: for any sampling interval Δ, there is a special frequency f_c,

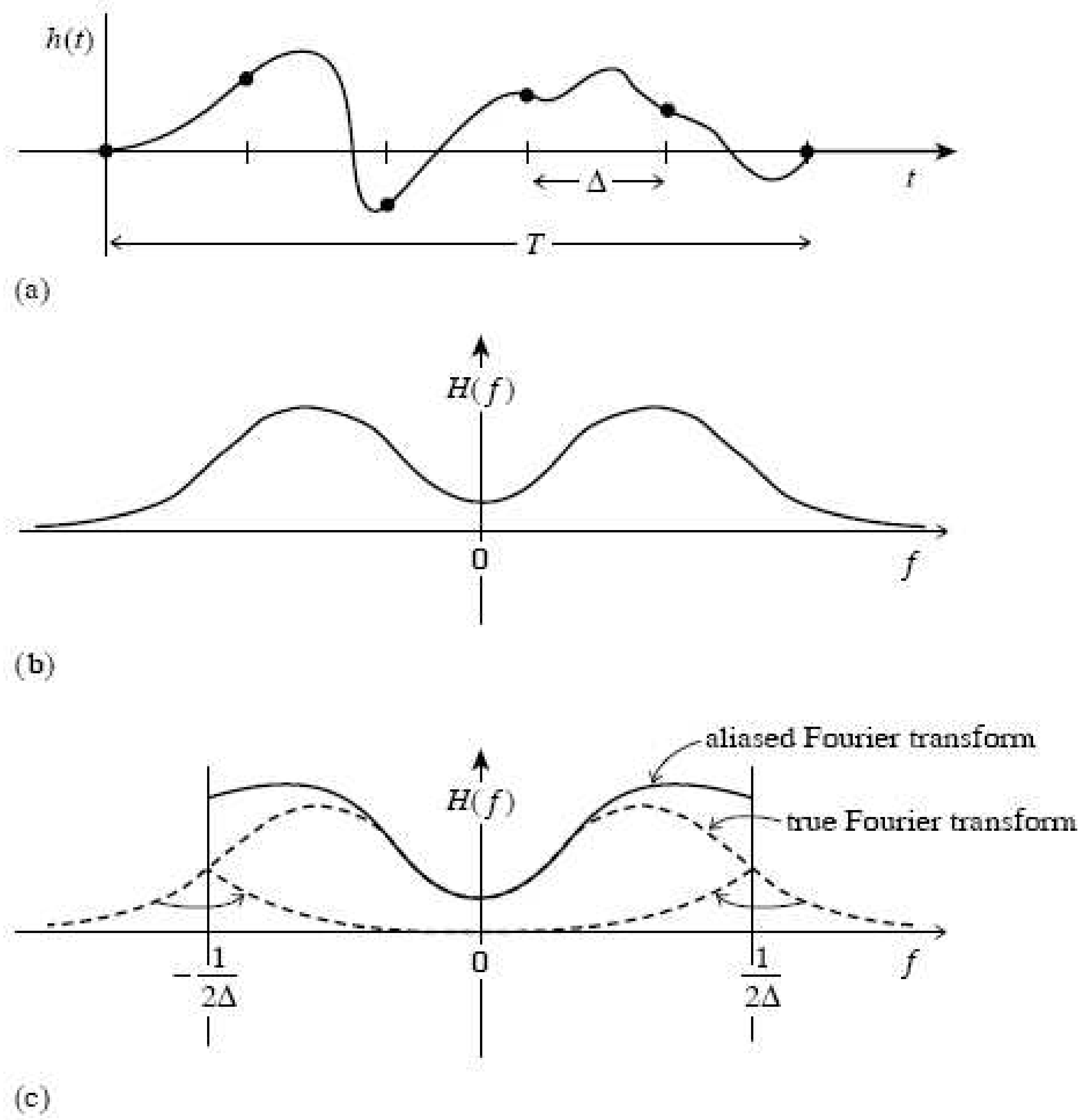
$$f_c \equiv \frac{1}{2\Delta},$$

where f_c is called the *Nyquist critical frequency*.

- The function $h(t)$ is **completely determined** by its samples h_n only when sampled at an interval Δ happens to be *bandwidth limited* to frequencies smaller in magnitude than f_c.

- Aliasing: sampling a continuous function that is *not* bandwidth limted to less than the Nyquist critical frequency.

Aliasing



h(t)
Δ
t
T
(a)
H(f)
0
f
(b)
aliased Fourier transform
true Fourier transform
H(f)
0
f
−1/2Δ
1/2Δ
(c)

Discrete Fourier Transform

- in *time* domain, N consecutive sampled values, (suppose N is even),

$$h_k \equiv h(t_k), \quad t_k \equiv k\Delta, \quad k = 0, 1, 2, \ldots, N-1$$

- in *frequency* domain,

$$f_n \equiv \frac{n}{N\Delta}, \quad n = -\frac{N}{2}, \ldots, \frac{N}{2},$$

  there are $N+1$ values, but $f_{-N/2} = f_{N/2}$.

- discrete Fourier transform

$$\begin{aligned} \mathbf{H}(f_n) &= \int_{-\infty}^{\infty} f(t) e^{2\pi i f_n t} \mathrm{d}t \\ &\approx \sum_{k=0}^{N-1} h_k e^{2\pi i f_n t_k} \Delta \\ &= \Delta \sum_{k=0}^{N-1} h_k e^{2\pi i k n/N} \end{aligned}$$

Discrete Fourier Transform

- The discrete Fourier transform maps N complex numbers (the h_k's) into N complex numbers (the H_n's), where H_n is

$$H_n \equiv \sum_{k=0}^{N-1} h_k e^{2\pi ikn/N}$$

- if n in H_n varies from 0 to $N-1$:
  - $n = 0$ corresponds to the zero frequency.
  - $1 \leq n \leq N/2 - 1$ corresponds to the positive frequencies $0 < f < f_c$.
  - $N/2 + 1 \leq n \leq N - 1$ corresponds to the positive frequencies $-f_c < f < 0$.
  - $n = N/2$ corresponds to both $f = f_c$ and $f = -f_c$.

- the discrete *inverse* Fourier transform is

$$h_k = \frac{1}{N} \sum_{n=0}^{N-1} H_n e^{-2\pi ikn/N}$$

Fast Fourier Transform

- the discrete Fourier transform

$$H_n \equiv \sum_{k=0}^{N-1} h_k e^{2\pi i k n/N} = \sum_{k=0}^{N-1} W^{nk} h_k,$$

where $W \equiv e^{2\pi i/N}$. DFT requires N^2 complex multiplications, and **O**(N^2) process.

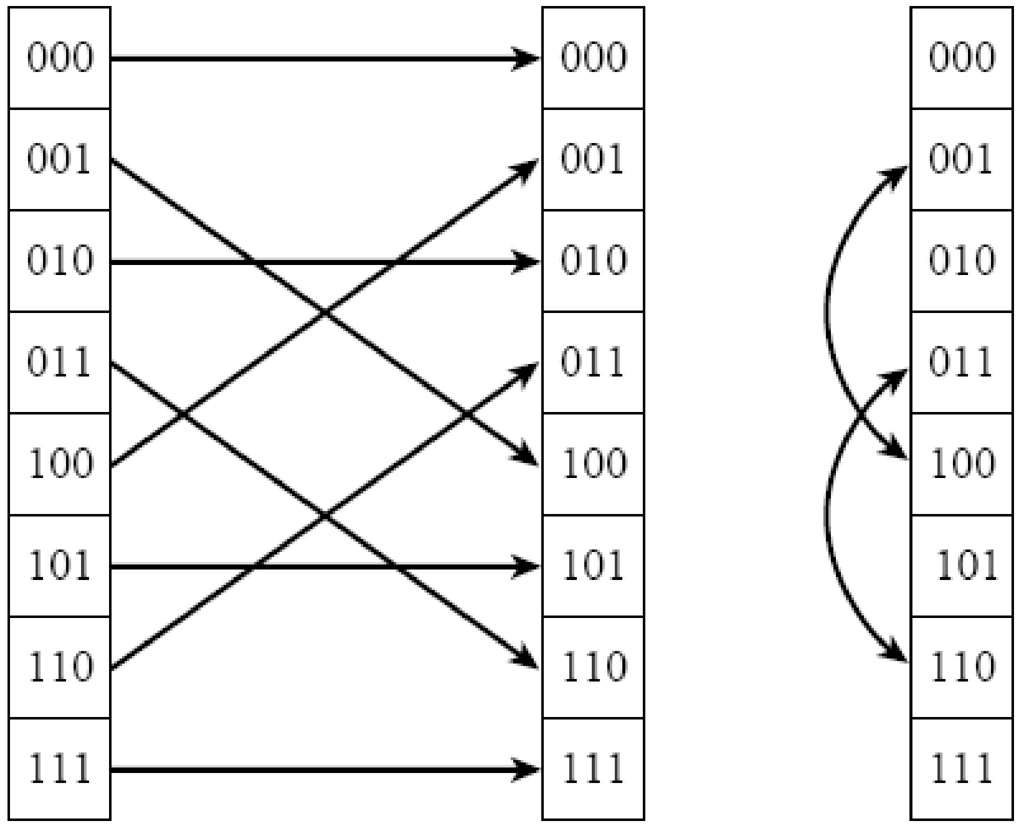
- fast Fourier transform, FFT, split the data into even- and odd-numbered points,

$$\begin{aligned} H_n &= \sum_{k=0}^{N-1} h_k e^{2\pi i k n/N} \\ &= \sum_{k=0}^{N/2-1} h_{2k} e^{2\pi i 2kn/N} + \sum_{k=0}^{N/2-1} h_{2k+1} e^{2\pi i (2k+1)n/N} \\ &= \sum_{k=0}^{N/2-1} h_{2k} e^{2\pi i k n/(N/2)} + W^n \sum_{k=0}^{N/2-1} h_{2k+1} e^{2\pi i k n/(N/2)} \\ &= H_n^{\text{even}} + W^n H_n^{\text{odd}} \end{aligned}$$

with **O**$(N \log_2 N)$ operations only.

Structure of FFT algorithm

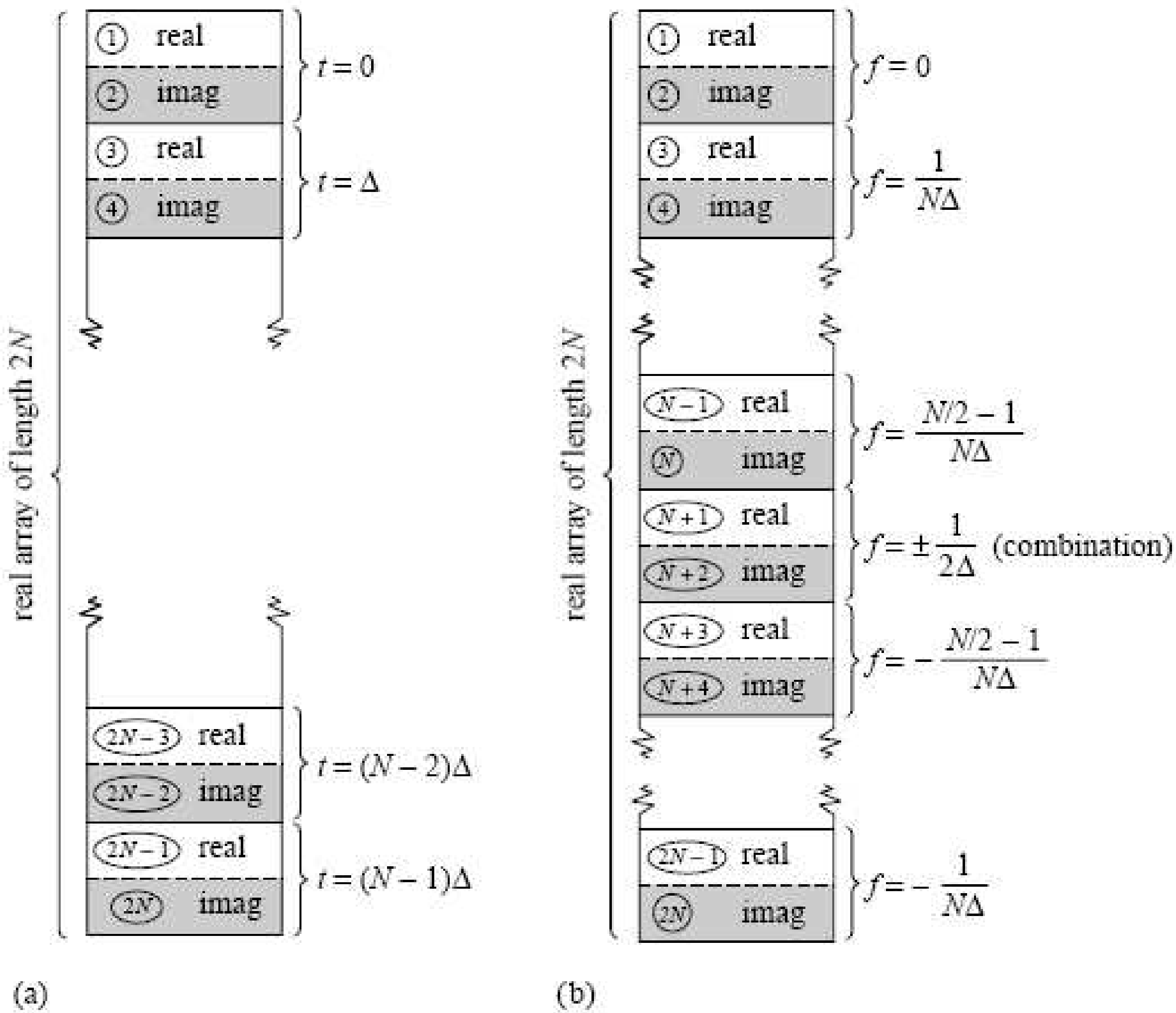
bit-reversed order,



N is a power of 2.

Structure of FFT algorithm

data output of FFT



N is a power of 2.

FFT: example

$$x(t) = \sin(1.5\pi t) + 0.5\cos(3\pi t),$$