Errors, Stability, Interpolation

Prateek Sharma (prateek@physics.iisc.ernet.in)
Office: D2-08

most of the material in the course is adapted from *Numerical Recipes*

Errors: round-off vs. truncation errors

Integer Representation

integers are represented exactly (range is machine dependent)
integer*4: 32 bits
integer*8: 64 bits (recommended with large integers)

0 _ _1011011_ _ 91

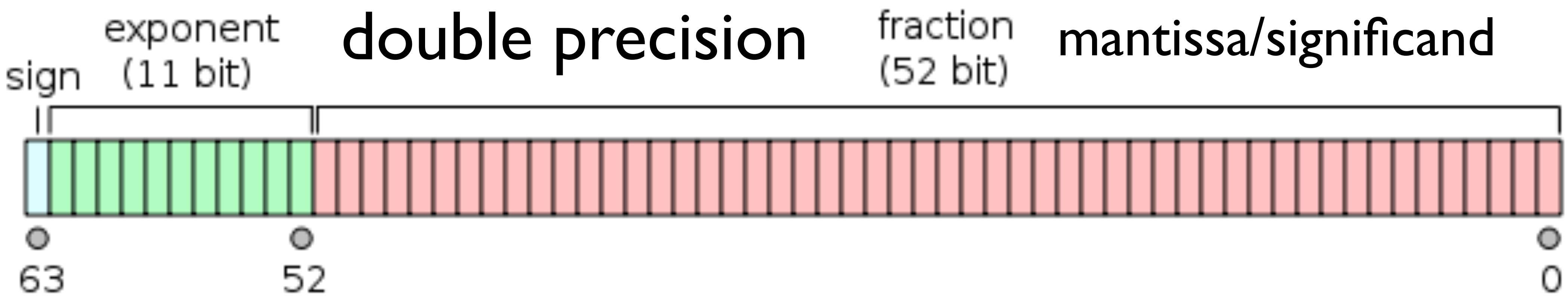
sign bit (0: + nos., 0) value

for integer*4: 31 bits to represent value from -2^{31} (-2147483648) to $2^{31}-1$ (2147483647)

Real Numbers

represented with floating-point (decimal point is floating)

double precision mantissa/significand



e.g., 152.6e5 => 0.1526e8

sign: 0(+), exponent: 8, mantissa: 1526

(of course these are internally represented as binary)

largest/smallest number that can be represented:

$$2^{\pm(2^{\wedge}10-1)} \sim 2^{\pm 1023} \sim 10^{\pm 308}$$

precision: $2^{52} \sim 4 \times 10^{15}$; thus DP stores ~16 places of a decimal number precisely; precision lost beyond that many digits

Machine Precision

smallest number represented in DP: 10^{-308}
What's the precision? Is it10^{-308}? No. it is 16 decimal places.

$1+10^{-16}=1$ in DP!
subtracting almost equal nos. result in loss of precision, e.g.,
1.2345678901234567 - 1.2345678998765432 =
-9.753086~~56059326~~x10^{-09}

dominated by *round-off error*

$x^2+bx+c=0$; $x = -b/2 \pm (b^2-4c)^{1/2}/2$ What if $c<<b^2$?

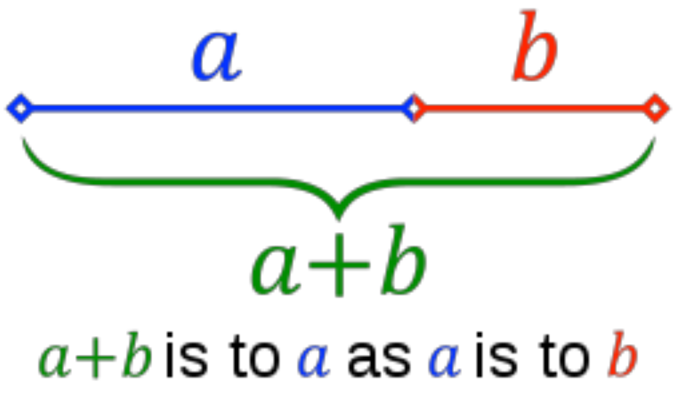
precision not lost in multiplication/division

Round-Off Errors

16 decimal places of precision is more than enough for most, but not all, applications.
High precision is required for, e.g., long term evolution of the solar system.

most of numerical analysis would remain even with infinite precision!
problem is not round-off errors but *numerical stability*
even tiny round-off errors grow rapidly if algorithm is not numerically stable

golden mean: $(a+b)/a=a/b=1/\Phi$
$\Phi^2+\Phi-1=0 \Rightarrow \Phi = (-1\pm5^{1/2})/2 = 0.618034, -1.618034$



recursive formula for powers of Φ: $\Phi^{n+1}=\Phi^{n-1}-\Phi^n$ w. $\Phi^0=1$, $\Phi^1=0.618034$

Numerical Instability

$\Phi^{n+1}=\Phi^{n-1}-\Phi^{n}$ w. $\Phi^{0}=1$

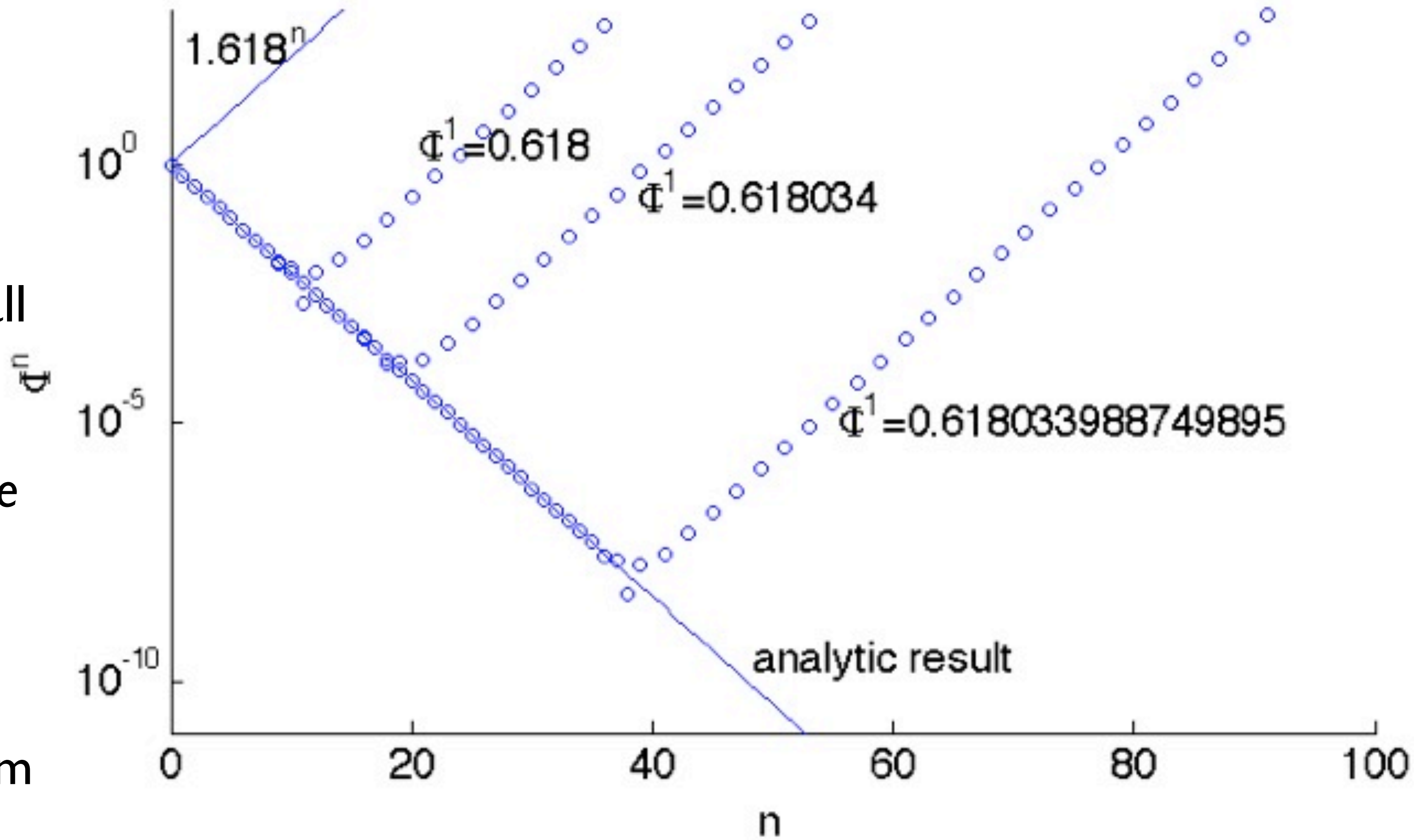
Lessons:

while RO errors are small, not small enough!

Algorithms must be numerically stable

Quiz:

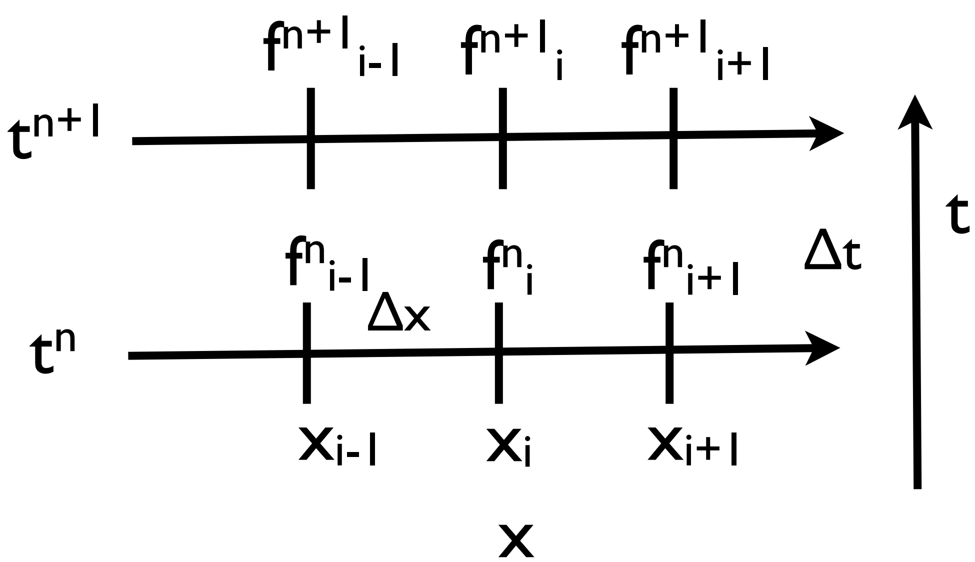
At what n will iteration deviate from the analytic result?



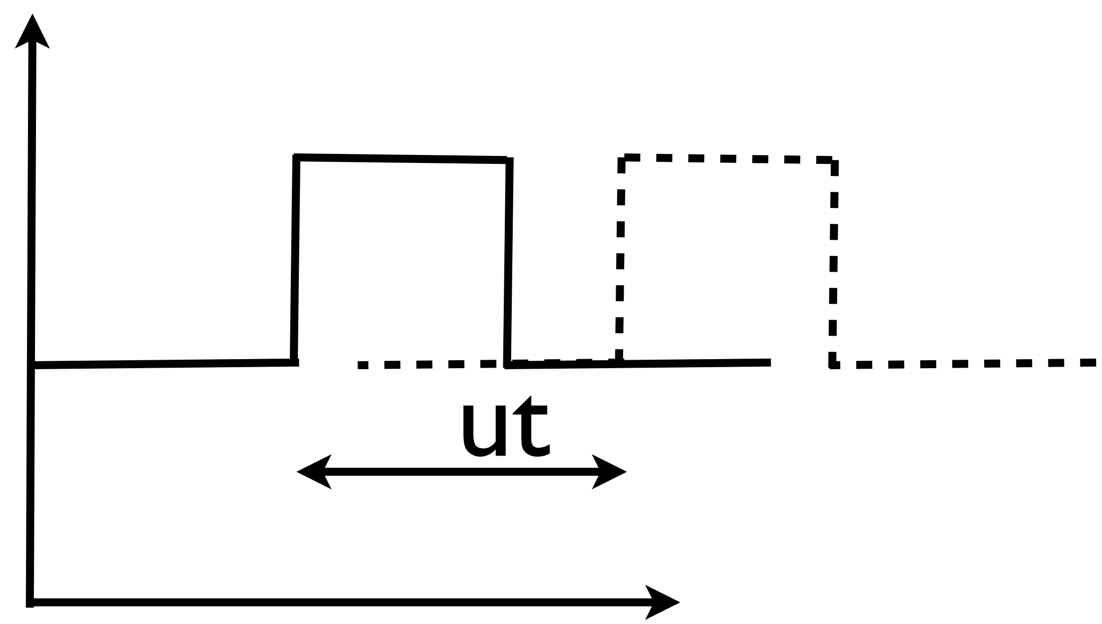
Advection Equation

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0$$

u=constant, advection equation solution $f(x,t) = f(x-ut,t=0)$



evolve the solution in time



FTCS finite difference formula:

$$\frac{f_i^{n+1} - f_i^n}{\Delta t} + u\frac{f_{i+1}^n - f_{i-1}^n}{2\Delta x} = 0$$

FTCS is unstable!
von-Neumann stability analysis

FTCS: forward in time centered in space

VNSA: linear analysis of difference eqs. w. Fourier modes

$$f_i^n = Ce^{-iwt^n + ikx_i} \qquad \frac{f_i^{n+1} - f_i^n}{\Delta t} + u\frac{f_{i+1}^n - f_{i-1}^n}{2\Delta x} = 0$$

$$f_i^{n+1} - f_i^n = Ce^{ikx_i}(e^{-iwt^{n+1}} - e^{-iwt^n}) = Ce^{(ikx_i - iwt^n)}(e^{-iw\Delta t} - 1)$$

$$\frac{(f_{i+1}^n - f_{i-1}^n)}{2\Delta x} = Ce^{-iwt^n}\frac{(e^{ikx_{i+1}} - e^{ikx_{i-1}})}{2\Delta x} = Ce^{(-iwt^n + ikx_i)}\frac{(e^{ik\Delta x} - e^{-ik\Delta x})}{2\Delta x}$$

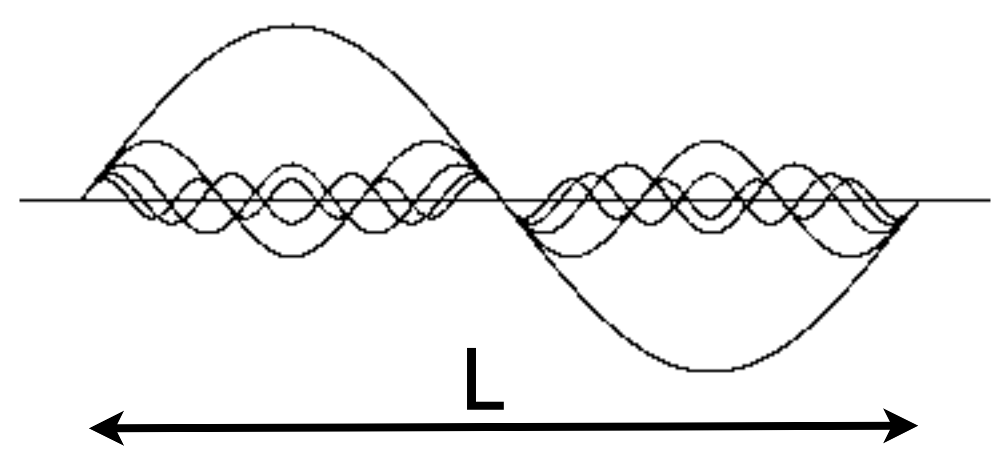
VNSA contd.

amplification factor $r \equiv e^{-iw\Delta t}$, so FTCS eq. becomes:

$$r = 1 - iku\Delta t \sin(k\Delta x) \quad \text{so} \quad |r| = \sqrt{1 + k^2 u^2 \Delta t^2 \sin^2(k\Delta x)} \geq 1$$

FTCS is unconditionally unstable!

for numerical scheme to be stable
all modes in the box should have $|r|<1$
i.e., there should be no growing mode



$k=2\pi n/L$, $n=1,2,..,L/2\Delta x$

$k_{Ny}=\pi/\Delta x$

Even tiny RO error can't handle numerical instability
we'll have much more on this once we
come to ODEs and PDEs

nonlinear stability much more difficult to prove!

Truncation Error

appears when the continuous problem $\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0$ is discretized; for smooth f(x,t)

Taylor Expansion:

$$f_i^{n+1} = f_i^n + \left(\frac{\partial f}{\partial t}\right)_i^n \Delta t + \left(\frac{\partial^2 f}{\partial t^2}\right)_i^n \Delta t^2/2 + \ldots$$

$$f_{i+1}^n = f_i^n + \left(\frac{\partial f}{\partial x}\right)_i^n \Delta x + \left(\frac{\partial^2 f}{\partial x^2}\right)_i^n \Delta x^2/2 + \ldots$$

first order accurate in time

$$\left(\frac{\partial f}{\partial t}\right)_i^n = (f_i^{n+1} - f_i^n)/\Delta t + \mathcal{O}(\Delta t)$$

second order accurate in space

$$\left(\frac{\partial f}{\partial x}\right)_i^n = (f_{i+1}^n - f_{i-1}^n)/2\Delta x + \mathcal{O}(\Delta x^2)$$

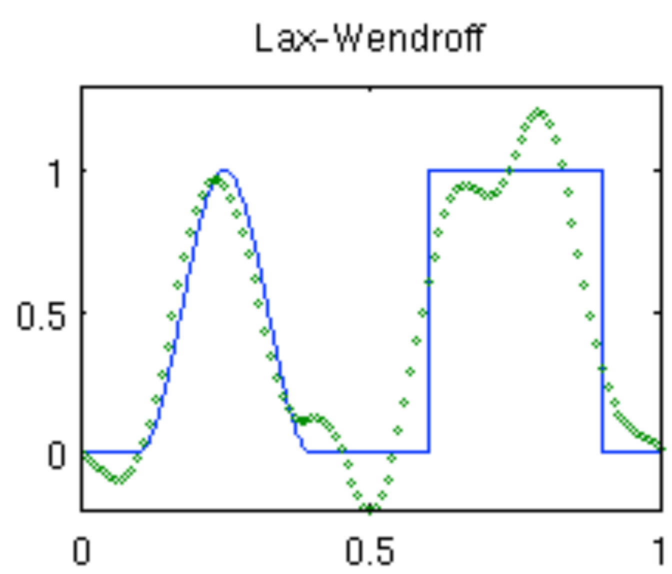
$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta x^2) = 0$$

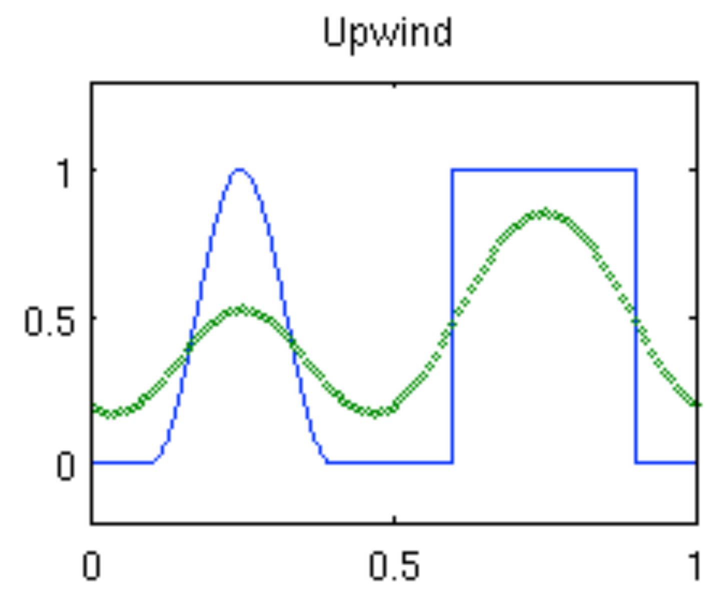
consistent as Δx ,$\Delta t \to 0$

Truncation vs RO errors

truncation error controlled by programmer; choose a more accurate method!
RO error is fixed (16 decimal places in DP); less control
typically truncation error>>round-off error; e.g., $\Delta x \sim 10^{-3}$ 2^{nd} order TE$\sim 10^{-6}$

order of accuracy not the sole metric
stability, robustness, mathematical properties more crucial





Amplitude vs Phase Errors

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0$$

true solution for a Fourier mode: w=ku

$$f \propto e^{ik(x-ut)} \quad r_{true} = e^{-iku\Delta t}$$

$$r = 1 - iku\Delta t \sin(k\Delta x) \; = |r|e^{i\phi}$$

amplitude error: $|r_{true}|/|r|-1$, phase error: $\varphi_{true}/\varphi-1$ (normalized)

recall for FCTS: $|r| = \sqrt{1 + k^2u^2\Delta t^2 \sin^2(k\Delta x)} \geq 1$

amplitude error results in growth in amplitude and phase error introduces phase shift in the solution relative to the true solution.

How to measure error?

$$L_p = \left(\frac{1}{N} \sum_{i=1}^{N} |F_i - F|^p \right)^{1/p}$$

L_p error

F_i: discrete approx. solution at x_i, F: correct solution at x_i

$L_1 < L_2 < L_3 \ldots\ldots < L_\infty$

weighted to min. point-wise error

weighted to max. point-wise error

What if we don't know the correct solution? can use the solution with half the step-size: *Richardson error*

$$L_p = \left(\frac{1}{N} \sum_{i=1}^{N} |F_i^{\Delta x} - F_i^{\Delta x/2}|^p \right)^{1/p}$$

Richardson Extrapolation: $A - A(h) = a_n h^n + O(h^m),\ a_n \neq 0,\ m > n$ $h=\Delta x$

$$R(h) = A(h/2) + \frac{A(h/2) - A(h)}{2^n - 1} = \frac{2^n A(h/2) - A(h)}{2^n - 1}$$

accurate to $O(h^m)$

Modified Eq.

the equation that is *really* being solved

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta x^2) = 0$$

lets write the next order terms:

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + \frac{\partial^2 f}{\partial t^2}\Delta t/2 + \frac{\partial^3 f}{\partial x^3}u\Delta x^2/3 + \mathcal{O}(\Delta t^2) + \mathcal{O}(\Delta x^4) = 0$$

$$\frac{\partial f}{\partial t} = -u\frac{\partial f}{\partial x} + \mathcal{O}(\Delta t) => \frac{\partial^2 f}{\partial t^2} = u^2\frac{\partial^2 f}{\partial x^2} + \mathcal{O}(\Delta t)$$

modified eq., just keeping the lowest order term:

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = -(u^2\Delta t/2)\frac{\partial^2 f}{\partial x^2}$$

anti-diffusion w. D=$u^2\Delta t/2$
responsible for amplitude error!

derivatives w. even (odd) powers: diffusive/amplitude (dispersive) error
(easy to see in Fourier space)

Fundamental Thm in NA, here

for a consistent finite difference method for a well-posed linear initial value problem, the method is convergent if and only if it is stable.

Well-Posed: unique solution exists, solution depends continuously on data
not well posed called ill-posed; e.g., anti-diffusion eq.

IVP: $f(0) \longrightarrow f(t)$

consistency + stability = convergence

Convergence: better & better agreement with solution as $\Delta x, \Delta t \longrightarrow 0$

Stability: already about VNSA; nonlinear stability is tough to prove

Consistent: solving the correct eq. as $\Delta x, \Delta t \longrightarrow 0$

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta x^2) = 0$$

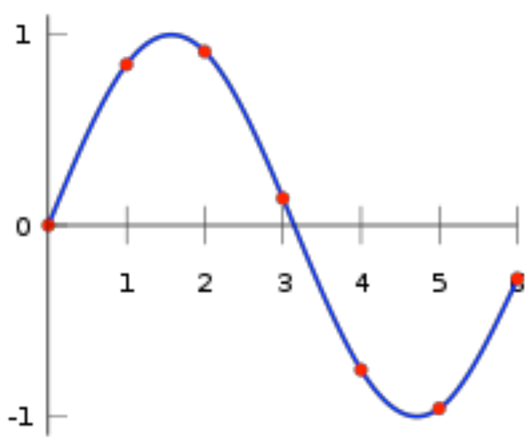
Interpolation & Extrapolation

given f_i at x_i find $f(x)$; $x_1<x<x_n$ interpolation

x outside the range: extrapolation

(cousin of data-fitting)

Lagrange interpolation: unique *polynomial* of degree N-1 through N pts.



$$L(x) := \sum_{j=0}^{k} y_j \ell_j(x) \qquad \mathcal{O}(N^2)$$

$$\ell_j(x) := \prod_{\substack{0\le m\le k \\ m\neq j}} \frac{x-x_m}{x_j-x_m} = \frac{(x-x_0)}{(x_j-x_0)} \cdots \frac{(x-x_{j-1})}{(x_j-x_{j-1})} \frac{(x-x_{j+1})}{(x_j-x_{j+1})} \cdots \frac{(x-x_k)}{(x_j-x_k)}.$$

Neville's Algorithm:

$$\begin{array}{lllll}
x_1: & y_1 = P_1 & & & \\
 & & P_{12} & & \\
x_2: & y_2 = P_2 & & P_{123} & \\
 & & P_{23} & & P_{1234} \\
x_3: & y_3 = P_3 & & P_{234} & \\
 & & P_{34} & & \\
x_4: & y_4 = P_4 & & &
\end{array}$$

constructing a higher order polynomial recursively

$$P_{i(i+1)\dots(i+m)} = \frac{(x-x_{i+m})P_{i(i+1)\dots(i+m-1)} + (x_i - x)P_{(i+1)(i+2)\dots(i+m)}}{x_i - x_{i+m}}$$

Barycentric interpolation: O(N)

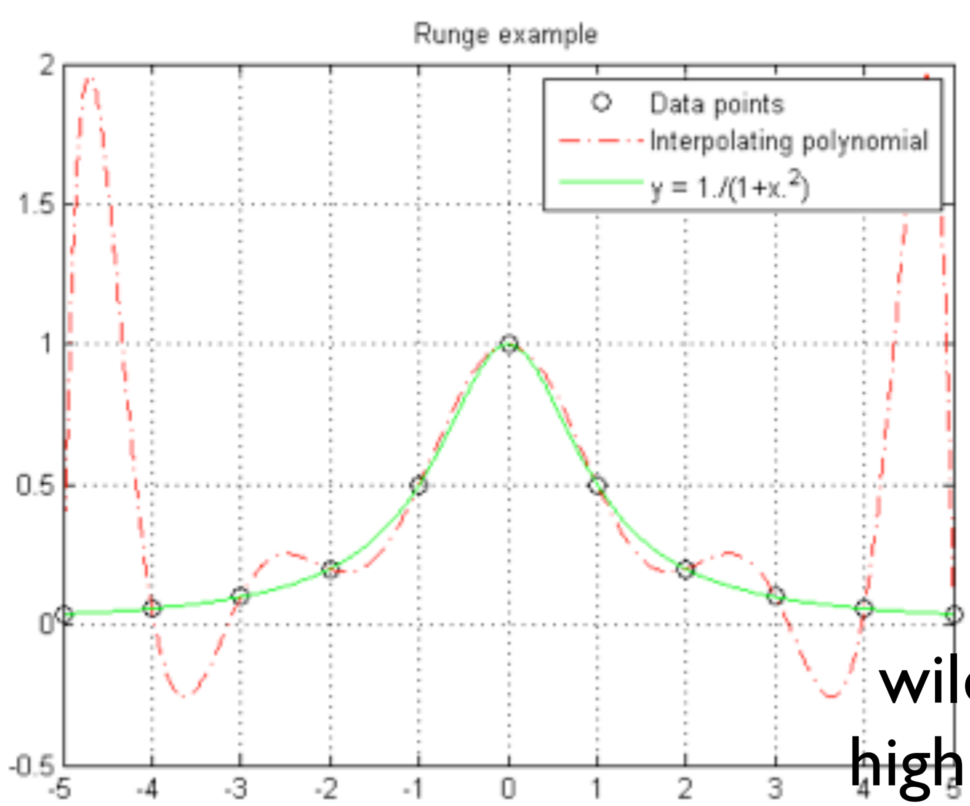
Rational interpolation

$$\frac{P_\mu(x)}{Q_\nu(x)} = \frac{p_0 + p_1 x + \cdots + p_\mu x^\mu}{q_0 + q_1 x + \cdots + q_\nu x^\nu} \qquad m+1 = \mu + \nu + 1$$

useful for functions with poles

which function to use for interpolations depends on nature of data

high order polynomial interpolation not always best!



wild oscillations with
high order polynomials
:Runge
phenomenon

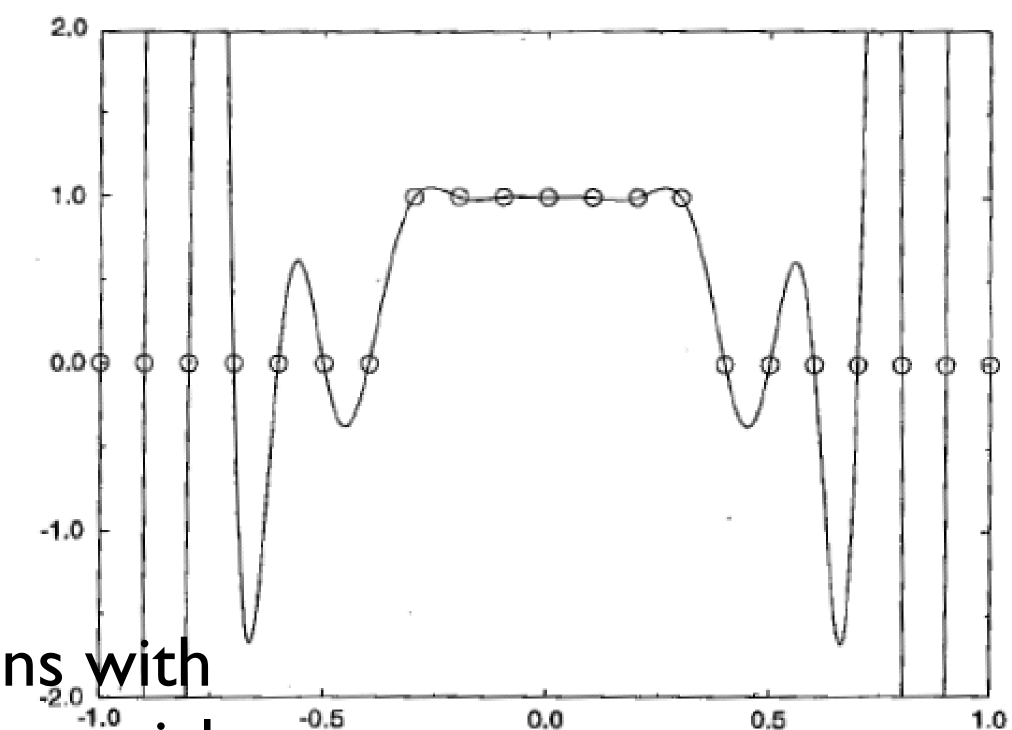
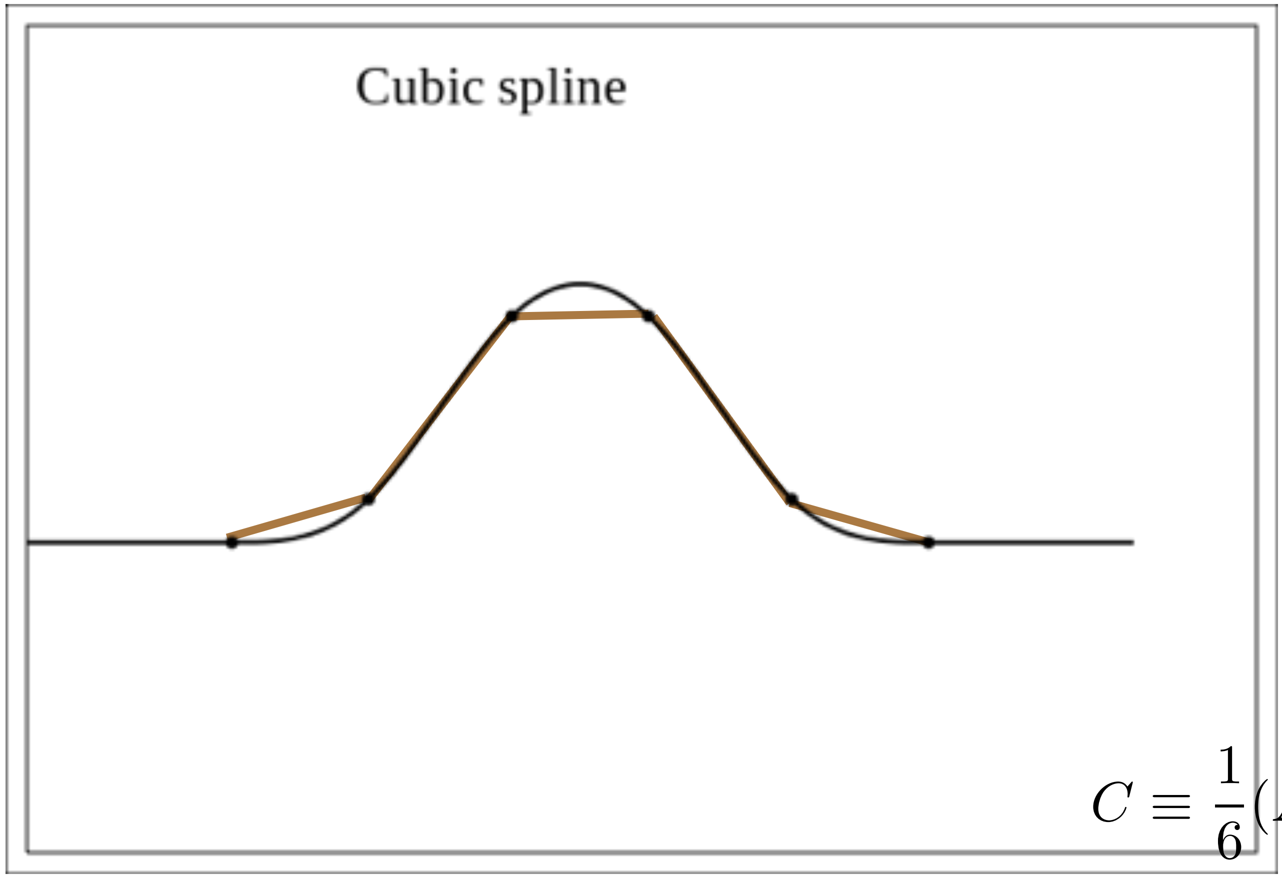


Figure 8.5 Twentieth-order polynomial interpolation for a square wave.

Splines



cubic spline: smooth in f', continuous f'', both inside the interval and at boundaries

$$y = Ay_j + By_{j+1}$$

$$A \equiv \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B \equiv 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

$$y = Ay_j + By_{j+1} + Cy_j'' + Dy_{j+1}''$$

$$C \equiv \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2 \qquad D \equiv \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

What about y''? obtained from smoothness of y'

$$\frac{d^2y}{dx^2} = Ay_j'' + By_{j+1}''$$

$$\frac{dy}{dx} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3A^2 - 1}{6}(x_{j+1} - x_j)y_j'' + \frac{3B^2 - 1}{6}(x_{j+1} - x_j)y_{j+1}''$$

N-2 eqs. for N unknowns; tridiagonal system

$$\frac{x_j - x_{j-1}}{6}y_{j-1}'' + \frac{x_{j+1} - x_{j-1}}{3}y_j'' + \frac{x_{j+1} - x_j}{6}y_{j+1}'' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

Tridiagonal Systems

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i,$$

$$\begin{bmatrix} b_1 & c_1 & & & 0 \\ a_2 & b_2 & c_2 & & \\ & a_3 & b_3 & \ddots & \\ & & \ddots & \ddots & c_{n-1} \\ 0 & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix}.$$

can be solved in O(n) operations
not O(n^3)!

Forward Elimination:

for k = 2 step until n do

$$m = \frac{a_k}{b_{k-1}}$$
$$b_k = b_k - m c_{k-1}$$
$$d_k = d_k - m d_{k-1}$$

end loop (k)

Backward Substitution:

$$x_n = \frac{d_n}{b_n}$$

for k = n-1 stepdown until 1 do

$$x_k = \frac{d_k - c_k x_{k+1}}{b_k}$$

end loop (k)