

Lecture 14

Summary (what to remember)
exam

Numerical uncertainties

- Floating-point precision :
 - (17 digits 8-byte double, 8 digits float, machine-dependent)
 - Introduces Round-off errors
 - Not all numbers representable!
 - Strong and narrow poles: $E + \ln |E - E_{res}|$
 - Truncation errors
 - In square root: calculate correctly

$$q = -\frac{1}{2} \left(b + \operatorname{sgn}(b) \sqrt{b^2 - 4ac} \right)$$

$$x_1 = \frac{q}{a}, \quad x_2 = \frac{c}{q}$$

Round-off errors

- Narrow poles: maybe not representable, integration steps may not come close enough

$$x + \frac{\ln |x - \pi|}{2\pi}$$

- Derivatives: uncertainty in calculation ($f(x)$ and in $(x+h)$:

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}, \quad f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

- make exact measure h via $\text{temp}=x+h$, $h=\text{temp}-x$
- Still loses half the significant digits in this implementation

Numerical errors

- Truncation:
 - Functions calculated as series. Need to be cut off somewhere
- Stability: recurrent terms may lead to exponentially growing, exponentially damped, or stable errors (e.g. sqrt(terms))
 - Recursive steps. E.g. in Bessel function calculations. Hard to detect that an error blows up exponentially.
 - Change the order : use the recursion relation in the opposite direction!
 - In differential equations: use implicit differencing when necessary.

stability

- Numerical techniques:
 - Optimal accuracy via minimization of round-off and truncation errors
 - E.g. in matrix calculation: choose right pivot
 - Error control and error estimate
 - From number of terms, protection against underflow/overflow/division by zero etc.
- Numerical implementations:
 - Always Check/verify code. Either via calculation of an analytically known result, or via inspection:
 - Use several precisions. Do the results scale?
 - Change input parameters marginally. Do the answers behave linearly? If not, is that expected?
 - Use debugger

techniques

- Series acceleration:
 - Continuous fractions : converges in other range in complex plane. Usually very fast convergence
 - With integers: millions of digits can be calculated, as long as you make large integers. No round-off errors
 - Aitkens delta-squared process: geometric series
 - Euler transformation. Alternating series.
 - Clenshaw recurrence formula:
 - Applied in many techniques.
 - E.g. in Chebyshev calculation

interpolation

- Always use well-defined grid points:
 - Interpolation scheme depends on the grid points chosen!
 - Only switch at a grid point!
 - Higher order is not necessarily higher accuracy
 - Use existing algorithms: they will develop the calculation around the optimal point

$$(x - 100)^{12} \Rightarrow 10^{-16} \text{ error } (x \approx 100 \pm 1)$$

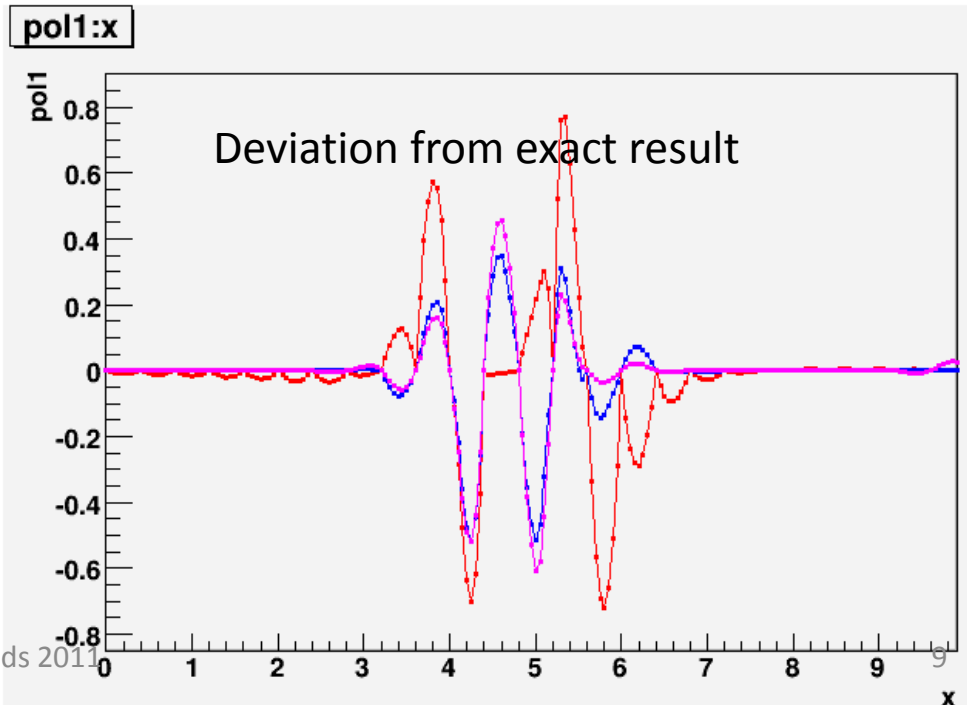
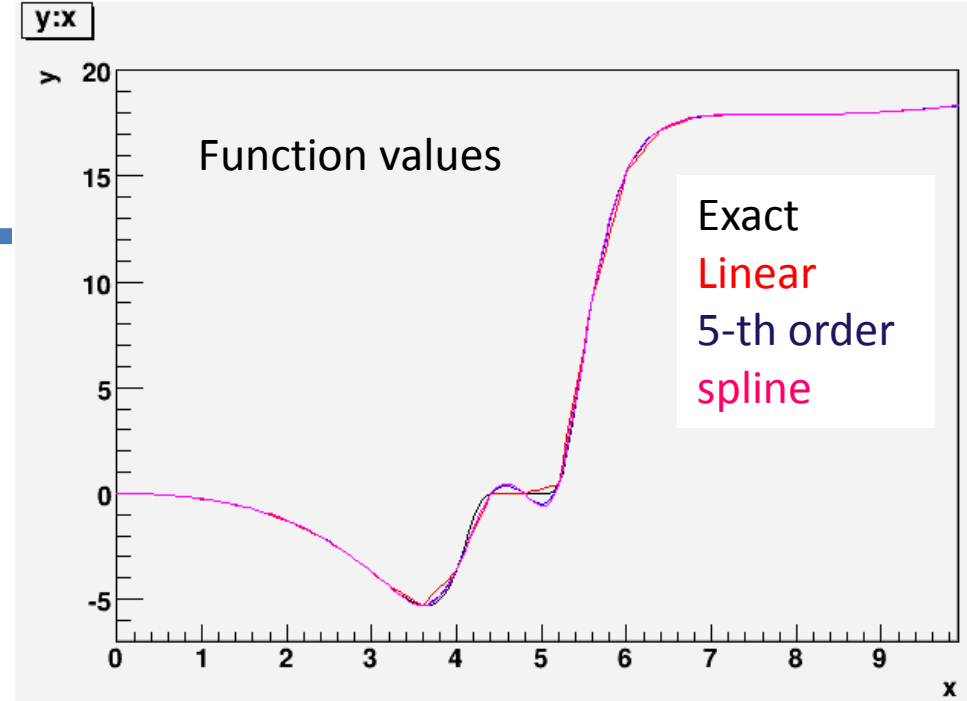
$$\sum_k \binom{12}{k} x^k (-100)^{12-k} \Rightarrow 10^8 \text{ error}$$

interpolation

- Spline: continuous derivatives. Also problematic when the derivatives are very small!
- Higher-order better for smooth functions, lower-order better for strongly-varying derivatives.
- Large deviations (0.5) found for perfectly smooth function in the interior in exercise 2,
- Large deviations ($1e5$) for 25-th order polynomial through 25 points.

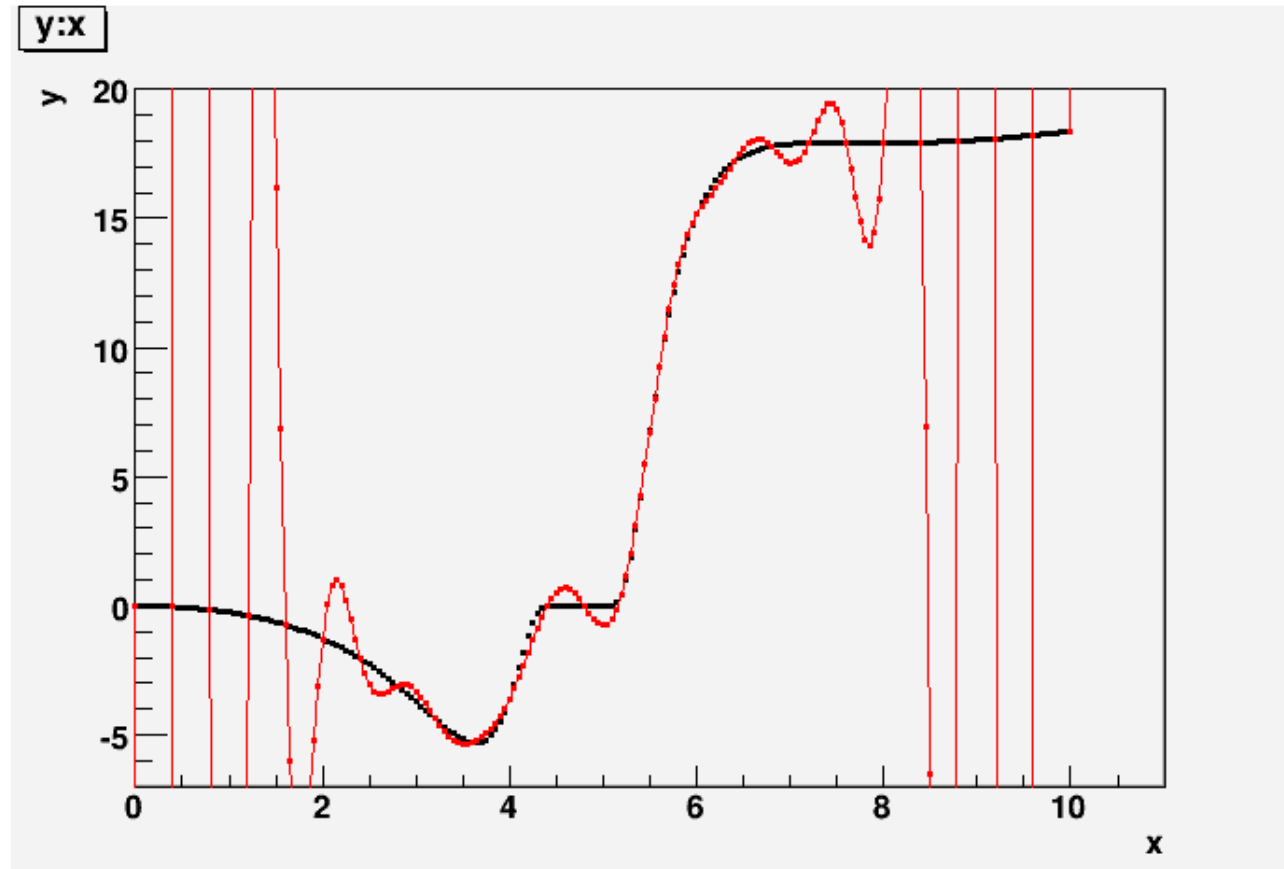
Exercise 2

- Example code on the web
- Spline and 5-th order polynomial have problems interpolating close to $x=4.75$
 - High-order derivatives are 0, Taylor expansion fails
- Linear interpolation gives worse result in region where the second-order derivative is sizable
- Interpolation errors of ~ 0.4 for spline (and 5-th order polynomial) while y -values are small, x -spacing is small, and only 1 zero crossing present.



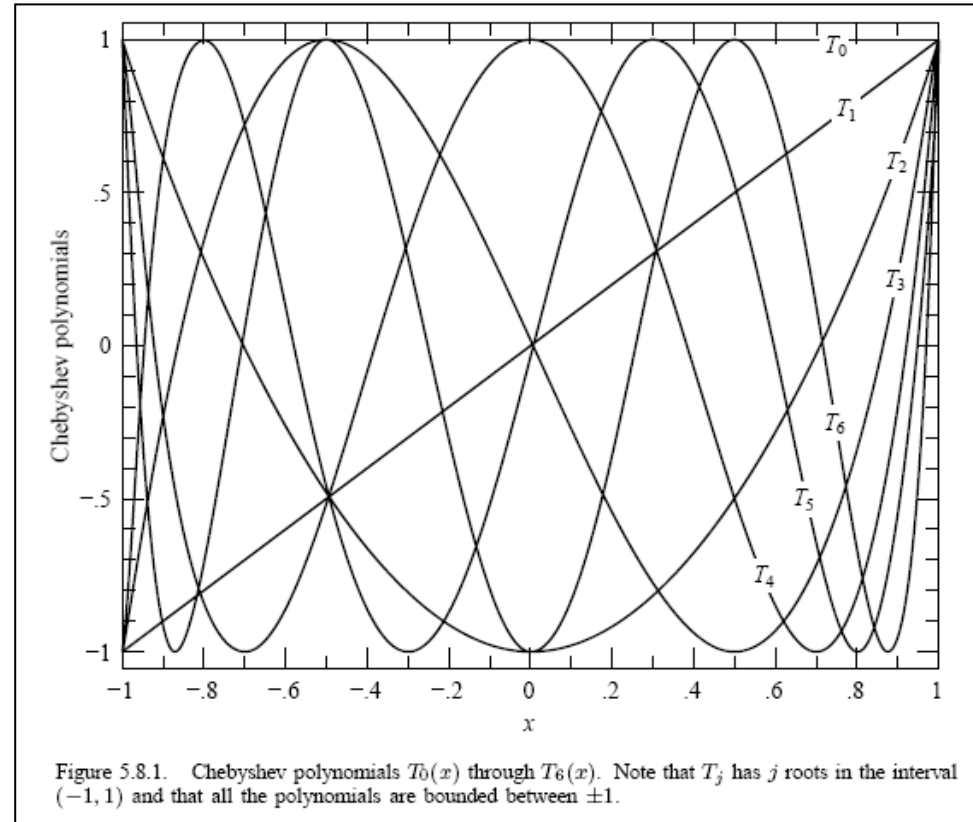
Interpolation

- Take care, do not use **too high orders!**
 - 25th order polynomial going through all grid values: contains factors of x^{25} , \gg billion.
 - At the boundaries, the polynomial runs through the grid points, but the values in-between deviate $\gg 10^5$



Function calculation

- Chebyshev: very powerful technique.
 - Integrals and differentials included
 - Clenshaw recurrence
 - Close to minimax polynomial that gives optimal description.
 - Always have higher order than zero crossings in the function you want to approximate!



Root finding

- Can be difficult, for close-by roots and for small first-order derivatives
- Bisection: linear convergence, fail-safe.
- Newton-Rhapson: quadratic convergence, doubling the number of significant digits per step!
 - May shoot off to infinity, may end in a loop
 - Use only when a reliable manner to calculate the derivative is available.
- Optimal general strategy: Van Wijngaarden-Dekker-Brent (fail-safe bracketing included, quadratic extrapolation)
- Polynomials: use Laguerre algorithm

Minimization

- Precision of minimum: at best half the significant digits

$$f(x + \varepsilon) \approx f(x) + \varepsilon^2 f''(x) / 2$$

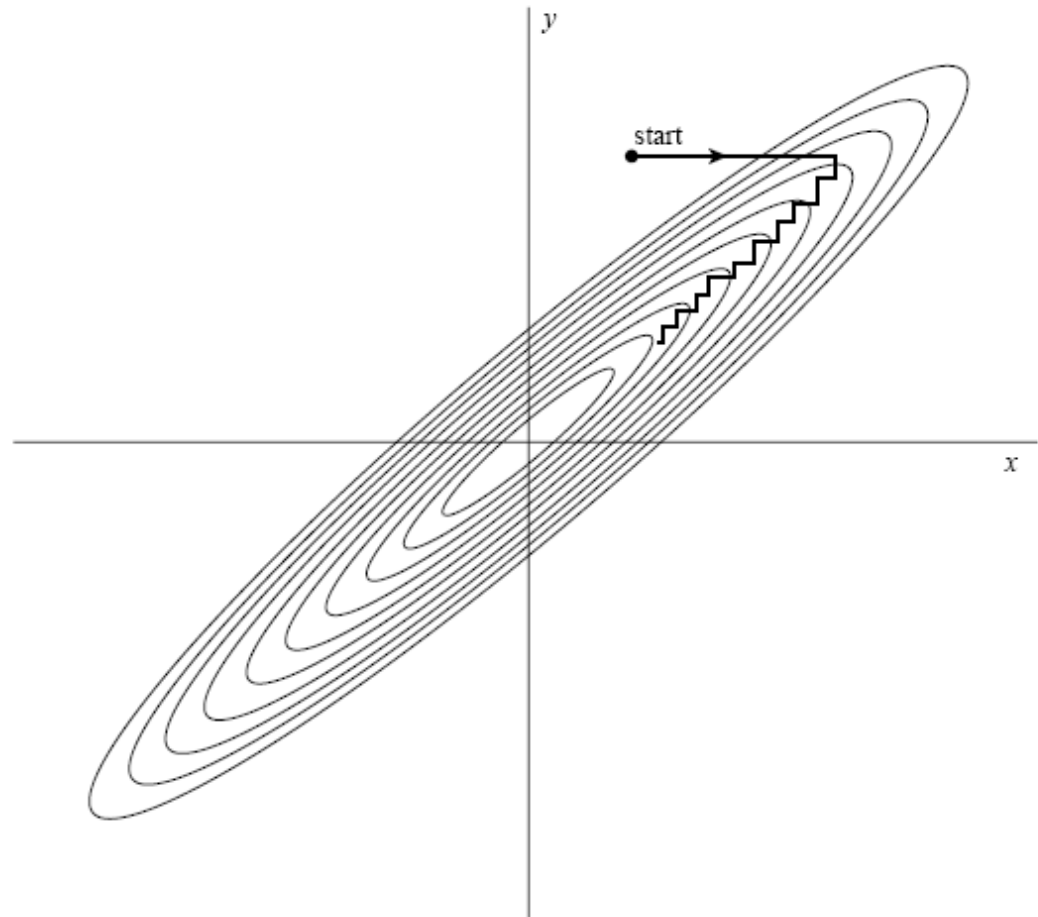
- Bracketing: with 3 function values
 - Golden-section steps are optimal.
 - Van Wijngaarden-Dekker-Brent quadratic step
 - Derivatives: use when reliable calculations exist.
- Simulated annealing
 - For large number of function values, traveling salesmen problem
 - Experiment with scheme

Minimization

- Multi-dimensional:
hard to find minima!

Downhill-simplex method

split off conjugate directions
(Powells method,
Fletcher-Reeves method)



integration

- Open-ended interval: 1 order lower accuracy (also with thousands of steps)
- Romberg integration: cancels higher-order derivatives per doubling of points (Euler-MacLaurin summation)
- Gaussian quadrature:
 - Very efficient for smooth functions
 - Abscissa chosen to be the zeros of a polynomial.
- Always have more grid points than zero crossings!
- Periodic functions: use Fast-Fourier transforms
- Chebyshev approximation also gives integral.
- Summation: usually a billion times less precise for the same amount of steps than e.g. Gaussian quadrature.

Fourier techniques

- Linear operation
 - used in many algorithms.
 - Information conserved in both time and frequency domain.
- Convolution and correlation – via FFT
 - Zero padding or windowing may be needed: FFTs are assumed to be periodic
- Higher frequency resolution: longer observation time
- Higher frequency range: higher sampling rate
 - Nyquist frequency: half the sampling frequency

Matrix operations

- Most operations (eigenfunctions, the inverse) scale with N^3 (for $N \times N$ matrix).
- Inverse, Gauss-Jordan elimination: only stable with pivoting:
 - Use existing code, don't try to do matrix operations by hand
- Nearly-singular matrices: try singular-value decomposition

Differential equations

- Always in terms of coupled sets of ordinary first-order diff. equations
- Adaptive stepsize : odeint integration package
 - Bulirsh-Stoer: uses Euler-MacLaurin summation
 - Runge-Kutta : fifth and sixth-order approximation in 6 steps.
- If unstable: use implicit differencing
- Two-point boundary conditions:
 - Shooting – good for linear problems
 - Relaxation – good for complicated boundaries, like in Finite-element analysis.
- Very important to test results : change precision/number of steps
 - or try to integrate from final result back to initial result (if possible)

Exam

- Last exercise due Dec. 23
- I'll put the results on the web around Dec. 27
- I'll mail you the final grade plus feedback when for the exercises that are below the maximum mark, around Jan. 7
 - I am not in the Netherlands next week, nor between christmas and Jan. 5