

Lecture 12: Numerical Differentiation

Outline

- 1) relationship to quadrature and differential equations
- 2) Basic Finite Difference approximations and errors (Taylor)
 - A) First order differences
 - B) 2nd order and 2nd derivatives
- 3) Interpolation and Finite Difference "Stencils"
 - A) 2nd order stencils
 - B) higher order and Chebyshev polynomials
- 4) Partial Differentials
- 5) Intro to PDE's (and the pitfalls of simple schemes)

Numerical Differentiation: Basic ideas

given a function $f(x)$, find an approximation for its derivative(s) $df/dx(x)$ as a weighted linear combination of function evaluations.

Relationship to Quadrature:

Caution: Numerical Quadrature is stable (maps R^n to R) and smoothing...(the only issues are accuracy and efficiency).
Numerical Differentiation can amplify noise.

Utility: useful for solving differential equations such as
 $-f'' + p(x)f = g(x)$

Finite Difference Approximations to $f'(x)$

Fundamental Definition of the derivative df/dx : given continuous function $f(x)$

Finite Difference approximation to df/dx (let h be finite)

Question: How bad an approximation is it?

Taylor series derivation of First derivative df/dx and Errors

Forward Difference:

Backwards Difference:

Centered Difference:

Finite Difference approximation to the second derivative d^2f/dx^2

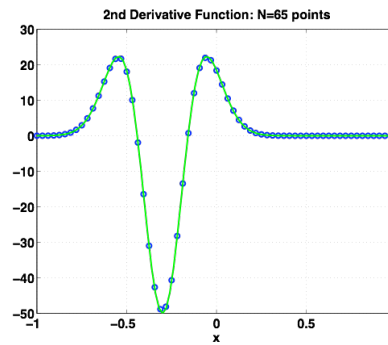
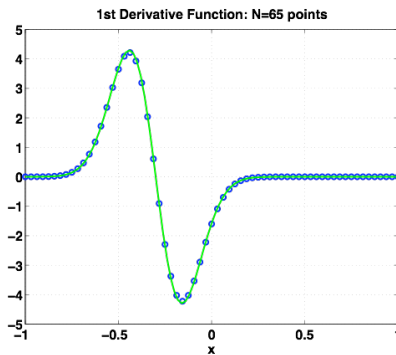
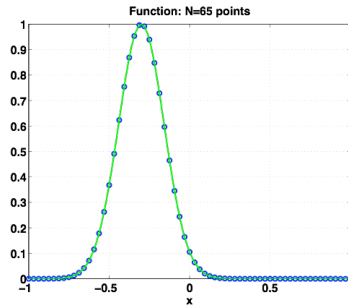
Finite Difference Stencils

Using Interpolation to derive finite difference stencils

Finite Difference Operators as sparse matrices

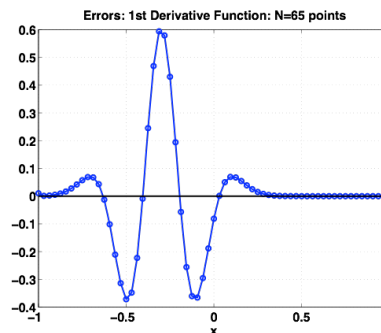
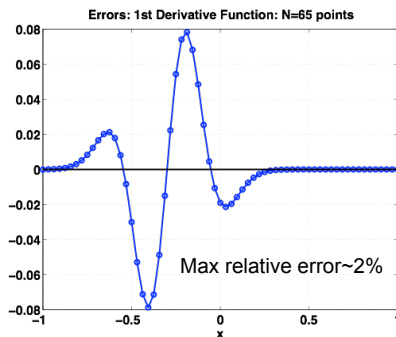
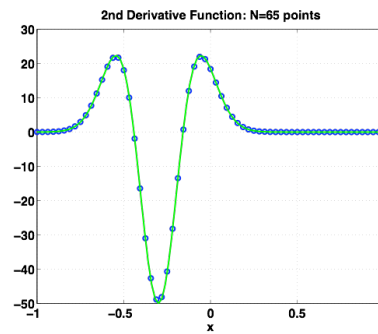
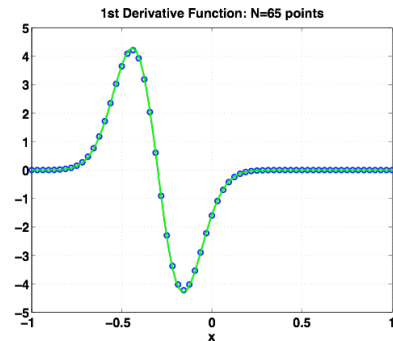
Finite Difference Operators as sparse matrices

Example: $f(x) = \exp(-[(x-x_0)/\sigma]^2)$ on $[-1, 1]$
 $[D1, D2, x] = \text{centerDifference}(65, [-1, 1])$, $\text{dfdx} = D1 * f(x)$, $\text{d2fdx2} = D2 * f(x)$;



Finite Difference Operators as sparse matrices

Example: $f(x) = \exp(-[(x-x_0)/\sigma]^2)$ on $[-1, 1]$



Higher Accuracy: h-refinement vs p-refinement

Two ways to improve accuracy:

h-refinement: add more points

p-refinement: use higher order interpolating polynomial

Danger Will Robinson! Higher order isn't necessarily higher accuracy...

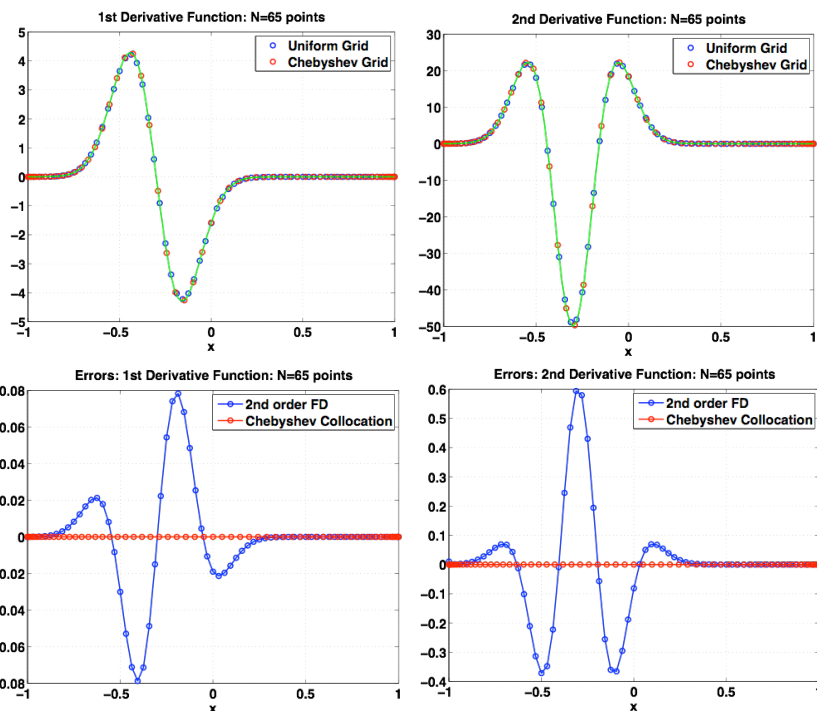
Issues with high-order polynomials:?

The usual fix:

Higher Accuracy: h-refinement vs p-refinement

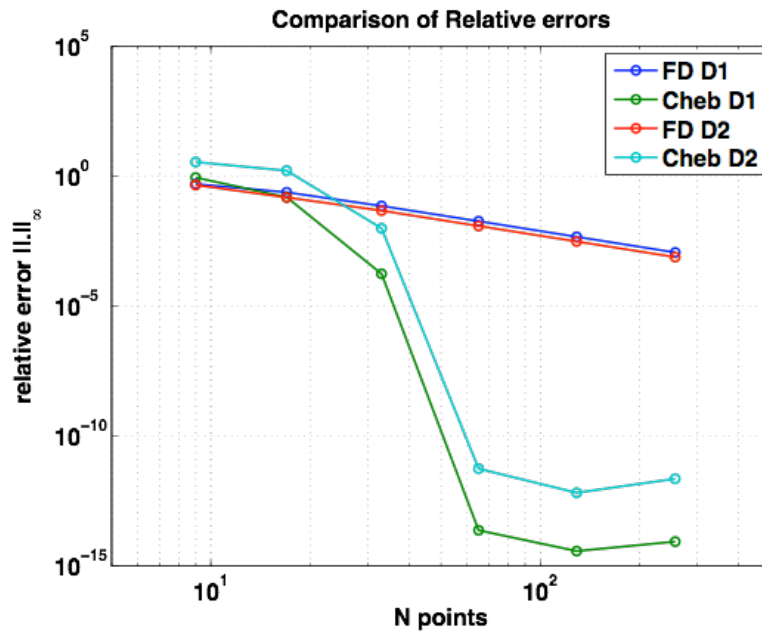
Same Problem, but using Chebyshev basis (and collocation points)

$[D, x] = \text{cheb}(64)$, $\text{dfdx} = D * f(x)$, $\text{d2fdx2} = D * (D * f(x))$;



Higher Accuracy: h-refinement vs p-refinement

Same Problem, but using Chebyshev basis (and collocation points)



Numerical Differentiation for Partial Derivatives

(Sneak-Peek for PDE's)

Numerical Differentiation for Partial Derivatives

Example: FTCS finite difference approximation to the simplest 1-D linear wave equation (not actually a good idea)