Numerical Differentiation

Sachin Shanbhag

Department of Scientific Computing
Florida State University,
Tallahassee, FL 32306.

References

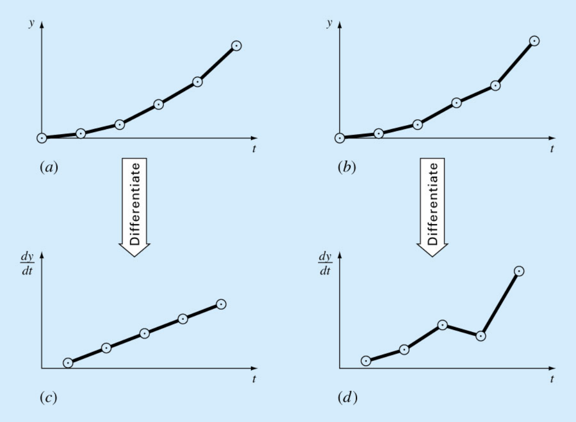
- S. Chapra and R. Canale, Numerical Methods for Engineers
- Gordon Erlebacher, Class Lectures, Fall 2009
- Harvey Stein, Risky Measures of Risk: Error Analysis of Numerical Dierentiation
- Bengt Fornberg, "Calculation of Weights in Finite Difference Formulas"
- Scholarpedia: Finite Difference Method

Why differentiate?

- Root finding in Newton's method
- Minimization of functions
- Solutions to ODEs/PDEs
- Example: Say, $f(x) = \sin x$, and we want to compute the derivative at $x = \pi/4$ using the numerical differentiation rule:
  $$f'(x) = \frac{f(x+h) - f(x)}{h}$$
  The true value is $f'(x = \pi/4) = \cos(\pi/4) = 0.7071$.
  If $h = 0.01$, then the numerical value is
  $$\frac{\sin(\pi/4 + 0.01) - \sin(\pi/4)}{0.01} = 0.7036$$

Differentiation

- Inherently noisy
- Opposite of integration (inherently smoothing)
- Inverse operations, so not surprising



from Chapra and Canale

Contents

- Error analysis

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

  It is more complicated than it looks
- Truncation and round-off errors
- How to optimize errors and improve accuracy?

Naively

- Consider the 1st order forward difference formula

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- May be numerically approximated by

$$f'(x) \approx f'_r(x) = \frac{f(x+h) - f(x)}{h}$$

  Like integration, small h should give us good approximation
- Other approximations: centered first ordered

$$f'_c(x) = \frac{f(x+h) - f(x-h)}{2h}$$

Let's look at an example

So lets look at an example where $f(x)$ is the sine function,

$$f(x) = \sin x$$

and consider centered difference formula,

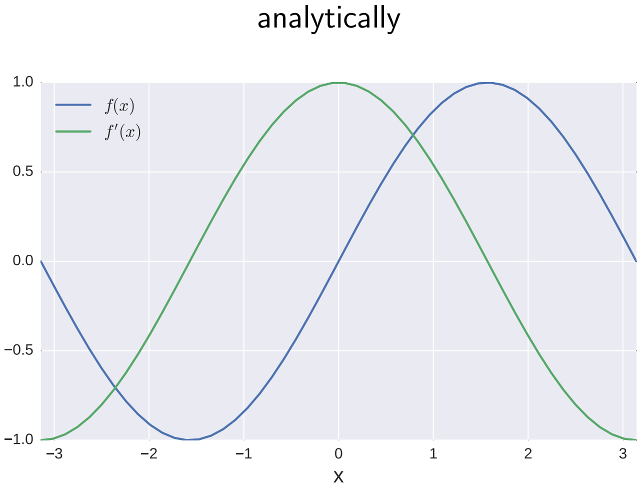
$$f_c'(x) = \frac{f(x+h) - f(x-h)}{2h}$$

and look at what happens as h is varied

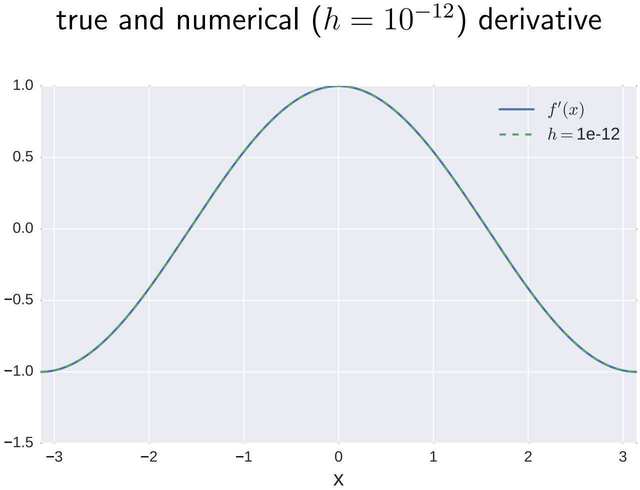
The "true" (symbolic) derivative of $f(x)$ is the normal distribution

$$f'(x) = \cos x$$

Example



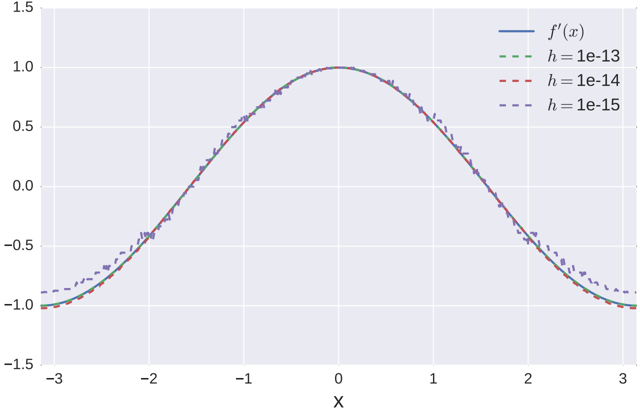
Example



true and numerical $(h = 10^{-12})$ derivative
1.0
0.5
0.0
−0.5
−1.0
−1.5
−3
−2
−1
0
1
2
3
x
$f'(x)$
$h=$1e-12

Example

$$h = 10^{-13}, 10^{-14}, 10^{-15}$$



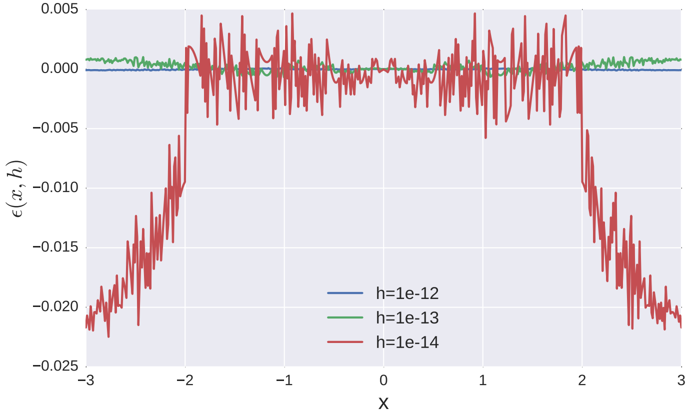
What just happened?

- Short answer: We hit machine precision
- IEEE double standard (64 bit) has 52 bit mantissa $(+1$ for sign)
- We can represent upto $2^{-52} \sim 10^{-16}$ or only 16 decimal digits
- As we approach $h = 10^{-16}$, we hit this limit relentlessly. So small is not necessarily good.
- In fact, there is more bizarre stuff!
- Since we know the derivative of this function analytically, we can look at the true error

$$\epsilon(x, h) = f'(x) - f_c(x, h)$$

Example

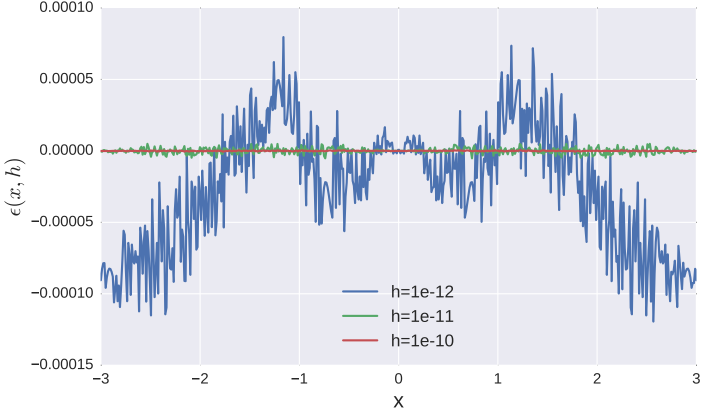
Looking at previous results through this lens



Error seems to increase as h is *decreased*.

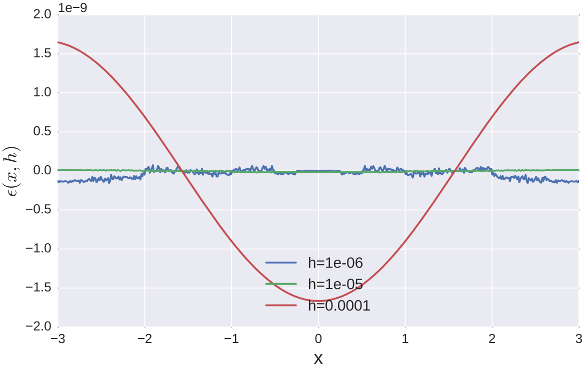
Example

The story continues. Note that y-axis is stretched.



Example

Until finally, "commonsense" prevails



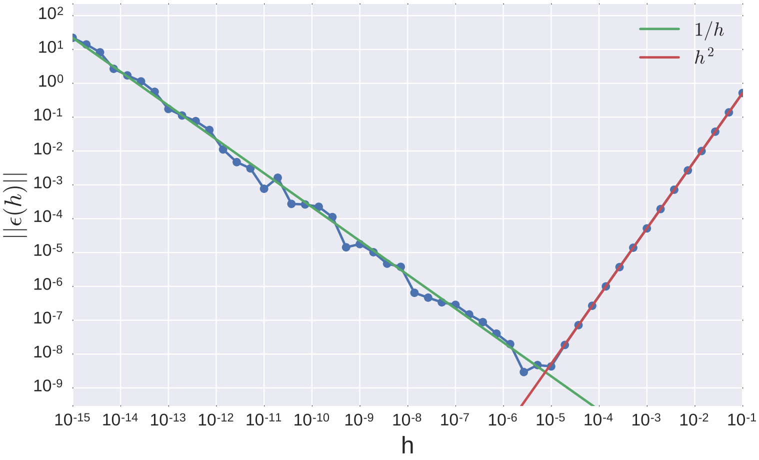
$h = 10^{-5}$ is the best choice? Who would have thought?

Recap

- Why is optimal $h \sim 10^{-5}$ so far from $\epsilon_{\text{mach}} \sim 10^{-16}$?
- Looked at a particular $f(x)$; qualitatively same for other functions
- Two important sources of error
  Truncation Error: increases with increasing h
  Roundoff error: increases with decreasing h
- Define the "error" as, say, the 1-norm over the domain on "x" of the function $\epsilon(x,h) = f'(x) - f_c(x,h)$, i.e.,

$$\text{error} = \frac{1}{n}\sum_{i=0}^{n} |\epsilon(x_i,h)|$$

Recap



$1/h$
h^2
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
$||\epsilon(h)||$
10^{-15} 10^{-14} 10^{-13} 10^{-12} 10^{-11} 10^{-10} 10^{-9} 10^{-8} 10^{-7} 10^{-6} 10^{-5} 10^{-4} 10^{-3} 10^{-2} 10^{-1}
h

Convexity or Truncation Error

Taylor Series

$$
\begin{aligned}
f(x+h) &= f(x) + hf'(x) + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \ldots \\
f(x-h) &= f(x) - hf'(x) + \frac{f''(x)}{2!}h^2 - \frac{f'''(x)}{3!}h^3 + \ldots
\end{aligned}
$$

Subtracting the two:

$$f(x+h) - f(x-h) = 2hf'(x) + 2\frac{f'''(x)}{3!}h^3 + \ldots$$

Yields

$$\frac{f(x+h) - f(x-h)}{2h} = f'(x) + \frac{f'''(x)}{3!}h^2 + \ldots$$

Convexity or Truncation Error

The leading term of the truncation error is

$$\epsilon_{\text{trunc}} = \frac{f'''(x)}{3!} h^2$$

Error increases with h! (exclamation mark = "wow", not factorial)

Now we turn our attention to round-off error.

Recall that floating point numbers are written in terms of a mantissa and an exponent.

$$\pm 0.b_1 b_2 \ldots b_m \times 2^t$$

double precision numbers

- 1 bit for sign
- $m = 52$ bits for mantissa
- 11 bits for exponent t

Cancellation or Round-off Error

- The fractional error (or "machine precision") arises from the mantissa:

$$\epsilon_{\mathsf{mach}} = \frac{1}{2^m} = \frac{1}{2^{52}} \approx 10^{-16}$$

- Thus,

$$\underbrace{\bar{f}(x)}_{\text{finite precision}} = \underbrace{f(x)}_{\text{true}} + \underbrace{\epsilon_{\mathsf{fp}}}_{\text{error}},$$

  where ϵ_{fp} is the actual error in $f(x)$ given by:

$$\epsilon_{\mathsf{fp}} = \epsilon_{\mathsf{mach}} f(x)$$

- We are interested in the cancellation error of $f(x+h) - f(x-h)$

Cancellation Error

$f(x+h)$ and $f(x-h)$ are approximately equal for small h

Thus, the round-off error *in the difference* $\approx C\epsilon_{\textsf{mach}} f(x)$, where C is a constant of order unity.

Therefore,

$$\begin{aligned} \bar{f}(x+h) - \bar{f}(x-h) &= f(x+h) - f(x-h) + \\ &\quad C\epsilon_{\textsf{mach}} f(x), \end{aligned}$$

or,

$$\begin{aligned} \epsilon_{\textsf{roundoff}} &= \frac{\bar{f}(x+h) - \bar{f}(x-h)}{2h} - \frac{f(x+h) - f(x-h)}{2h} \\ &\approx \frac{C\epsilon_{\textsf{mach}} f(x)}{2h} \end{aligned}$$

This sets up the optimization problem

Optimal h

We want to minimize convexity and cancellation error

$$\begin{aligned}\frac{f(x+h)-f(x-h)}{2h} &\approx f'(x)+\epsilon_{\text{trunc}}+\epsilon_{\text{roundoff}} \\ &\approx f'(x)+\frac{f'''(x)}{3!}h^2+\frac{C\epsilon_{\text{mach}}f(x)}{2h}\end{aligned}$$

- We want to minimize the blue part
- Setting

$$\frac{d}{dh}\left[\frac{f'''(x)}{3!}h^2+\frac{C\epsilon_{\text{mach}}f(x)}{2h}\right]=0$$

- Yields

$$h^*=\left(\frac{3C\epsilon_{\text{mach}}f}{2f'''}\right)^{1/3}$$

Optimal h

Note that the optimal h

$$h^* \sim \epsilon_{\mathsf{mach}}^{1/3}.$$

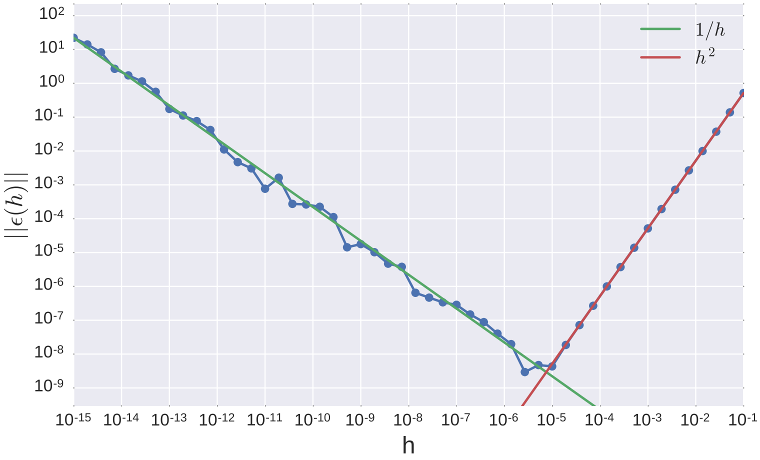
Explains our little numerical experiment,

$$\epsilon_{\mathsf{mach}}^{1/3} = \left(10^{-16}\right)^{1/3} \approx 10^{-5}$$

Thus,

Note that for different ϵ_{mach} and different differentiation formulae, the optimal h will be different.

Recap



$1/h$
h^2
$||\epsilon(h)||$
h
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-15} 10^{-14} 10^{-13} 10^{-12} 10^{-11} 10^{-10} 10^{-9} 10^{-8} 10^{-7} 10^{-6} 10^{-5} 10^{-4} 10^{-3} 10^{-2} 10^{-1}

Exercise: Finite Differences for Second Derivatives

- Write a Taylor expansion for $f(x+h)$ and $f(x-h)$.
- Evaluate and simplify the combination $f(x+h) - 2f(x) + f(x-h)$.
- Write down an finite difference expression for the second derivative.
- What is the truncation error?
- What is the round-off error?
- Estimate the optimal step size h.

Epilog

- If you want to find derivatives of noisy data, it is better to approximate the data using a smooth function before attempting numerical differentiation
- Alternative methods like complex step differentiation rephrase the problem to reduce its susceptibility to round-off error
- Automatic differentiation is an algorithmic technique to find the derivative of a function specified by computer code. Implementations of AD (as it is frequently called) are readily available for most computational languages and platforms.