Numerical Derivatives

Fabio Cannizzo

Use the Definition

- We want to estimate numerically the derivative of a function
- It is natural to think about the definition of derivative

$$\frac{df(x_0)}{dx} = \lim_{h \to 0^+} \frac{f(x_0+h) - f(x_0)}{h} \quad \Rightarrow \quad \frac{df(x_0)}{dx} \approx \frac{f(x_0+h) - f(x_0)}{h} \text{ (h small)}$$

- so we just take a very small value for *h*, and this gives us an approximation of the derivative
- we already know that we cannot take $h < x_0 \varepsilon_m$

Sources of Error

- What are the sources of error of my method:
  - truncation error
    - The definition hold when h ➔ 0, but in reality we use a finite value for *h*
  - round-off error:
    - Computer represent numbers up to a certain precision (machine precision)
    - Every operation results in further rounding
    - Rounding error will increase when h ➔ 0

Trade Off

- So when we decrease h we have a trade off between reduction of truncation error and growth or rounding error
- What is the order of convergence of my method (how quickly the truncation error reduces)?
- How small can I make *h*?
- Given we cannot reduce arbitrarily *h*, what else can we do to improve the approximation?

Order of Convergence

- Let's recall Taylor expansion

$$f(x_0+h) = f(x_0) + \frac{df(x_0)}{dx}h + \frac{1}{2}\frac{d^2 f(\xi)}{dx^2}h^2 \qquad \xi \in [x_0, x_0+h]$$

$$\frac{df(x_0)}{dx} = \frac{f(x_0+h) - f(x_0)}{h} - \frac{1}{2}\frac{d^2 f(\xi)}{dx^2}h$$

$$\frac{df(x_0)}{dx} = \frac{f(x_0+h) - f(x_0)}{h} + O(h)$$

- The error (i.e. the order of convergence) is of order O(*h*)
- This means that if we cut *h* by half, we can expect truncation error to also reduce by half

Round-Off Error

- The function $f(x)$ is computed in approximate form by the computer, e.g. it contains rounding errors

$$\frac{df(x)}{dx} = \frac{\left(f(x+h+\varepsilon_x+\varepsilon_h+\varepsilon_{xh})+\varepsilon_f\right)-\left(f(x+\varepsilon_x)+\varepsilon_f\right)+\varepsilon_d}{h+\varepsilon_h}$$

$$= \frac{\left(f(x(1+\varepsilon_m)+h(1+\varepsilon_m)+\varepsilon_{xh})+\varepsilon_{f^+}\right)-\left(f(x(1+\varepsilon_m))+\varepsilon_f\right)+\varepsilon_d}{h(1+\varepsilon_m)}$$

$$= \frac{f(x(1+\varepsilon_m)+h(1+\varepsilon_m)+\varepsilon_{xh})-f(x(1+\varepsilon_m))}{h(1+\varepsilon_m)} + \frac{\varepsilon_{f^+}-\varepsilon_f+\varepsilon_d}{h(1+\varepsilon_m)}$$

- When *h* becomes small, the second term tend to infinite
- It increases as 1/h

- Eventually, for very small *h* (when $|h/x|<\varepsilon_m$), the computer no longer distinguishes between *x* and *x*+*h*, therefore the numerator drops to zero

A Simple Improvement

- We can deal with the fact that *h* is not an exactly representable number via this trick:

  let $x' = x(1+\varepsilon_m)$, i.e. the machine representation of *x* (we cannot do better!)

  $xp = x' + h$

  $h' = xp - x'$

$$\frac{df(x)}{dx} = \frac{(f(x'+h') + \varepsilon_f) - (f(x') + \varepsilon_f) + \varepsilon_d}{h'}$$

$$= \frac{(f(x'+h') + \varepsilon_{f^+}) - (f(x') + \varepsilon_f) + \varepsilon_d}{h'}$$

$$= \frac{f(x'+h') - f(x')}{h'} + \frac{\varepsilon_{f^+} - \varepsilon_f + \varepsilon_d}{h'}$$

The two function arguments now differ by exactly *h*', which is not exactly *h*, but it does not really matter, as then we divide by exactly *h*'

- Note: this makes *h* a representable machine number, but it won't help us if $h/x < \varepsilon_m$, which would lead to h'=0

Total Error Profile

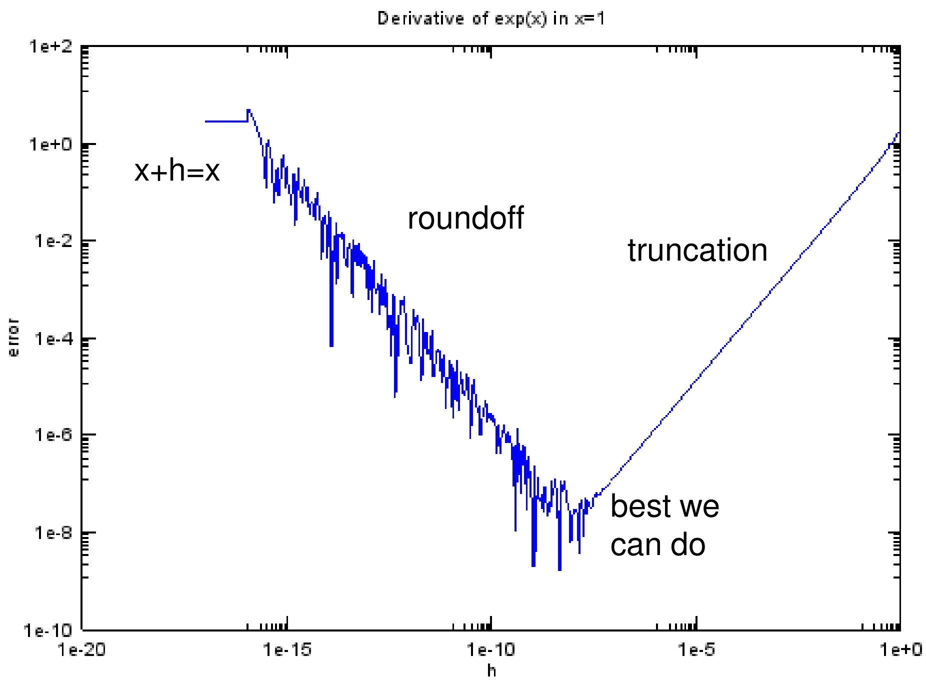
- How can we expect the error if we progressively reduce the size of the step?
  - The round-off error is roughly proportional to the function evaluation error ε_f, and grows as $O(h^{-1})$
  - The truncation error decrease steadily as $O(h)$
  - There is an optimal value of h, which achieve the best possible accuracy

Stability Regions

- The demo program *firstDeriv1.m* evaluates the derivative of the function '*exp*(*x*)' in *x*=1, using bump size ranging from *hMin* to *hMax*. Then plots the error in logarithmic space
  >> firstDeriv1(@exp,@exp,"exp(x)",1,1e-18,1)

- If h becomes too small, rounding error will start to dominate and the approximation will worsen instead of improving

- Note: we are plotting the total error, which is the sum of truncation error (decrease steadily) and round-off error (increase wiggly). Because the chart is in log-scale, the larger of the two errors completely hides the smaller one



- We observe three regions
  - Truncation error dominates: reducing h accuracy increase. Stable.
  - Round-off error dominates: reducing h accuracy decrease. Unstable.
  - *h*/*x* beyond machine precision, hence *x*+*h*=*x*. That is "game over".

Math Review

Theorem

- If $f(x)$ is a continuous function in the closed interval $[a,b]$, let $m=\min\{f(a),f(b)\}$ and $M=\max\{f(a),f(b)\}$, for any value y^* such that $m<y^*<M$ there exist at least one number x^* in $[a,b]$ such that $f(x^*)=y^*$

- Note that this is a special case of the more general “**Intermediate Value Theorem**”, but it is enough for our needs

- This obviously applies to the mean value $y^*=[f(a)+f(b)]/2$, which is a point in the interval $[m,M]$

Central Differences

- If we can afford two function valuations, one at *x*+*h*/2 and one at *x*-*h*/2

$$f\left(x+\frac{h}{2}\right)=f(x)+\frac{df(x)}{dx}\frac{h}{2}+\frac{1}{2}\frac{d^2f(x)}{dx^2}\left(\frac{h}{2}\right)^2+\frac{1}{6}\frac{d^3f(\xi)}{dx^3}\left(\frac{h}{2}\right)^3 \quad \xi\in\left[x,x+\frac{h}{2}\right]$$

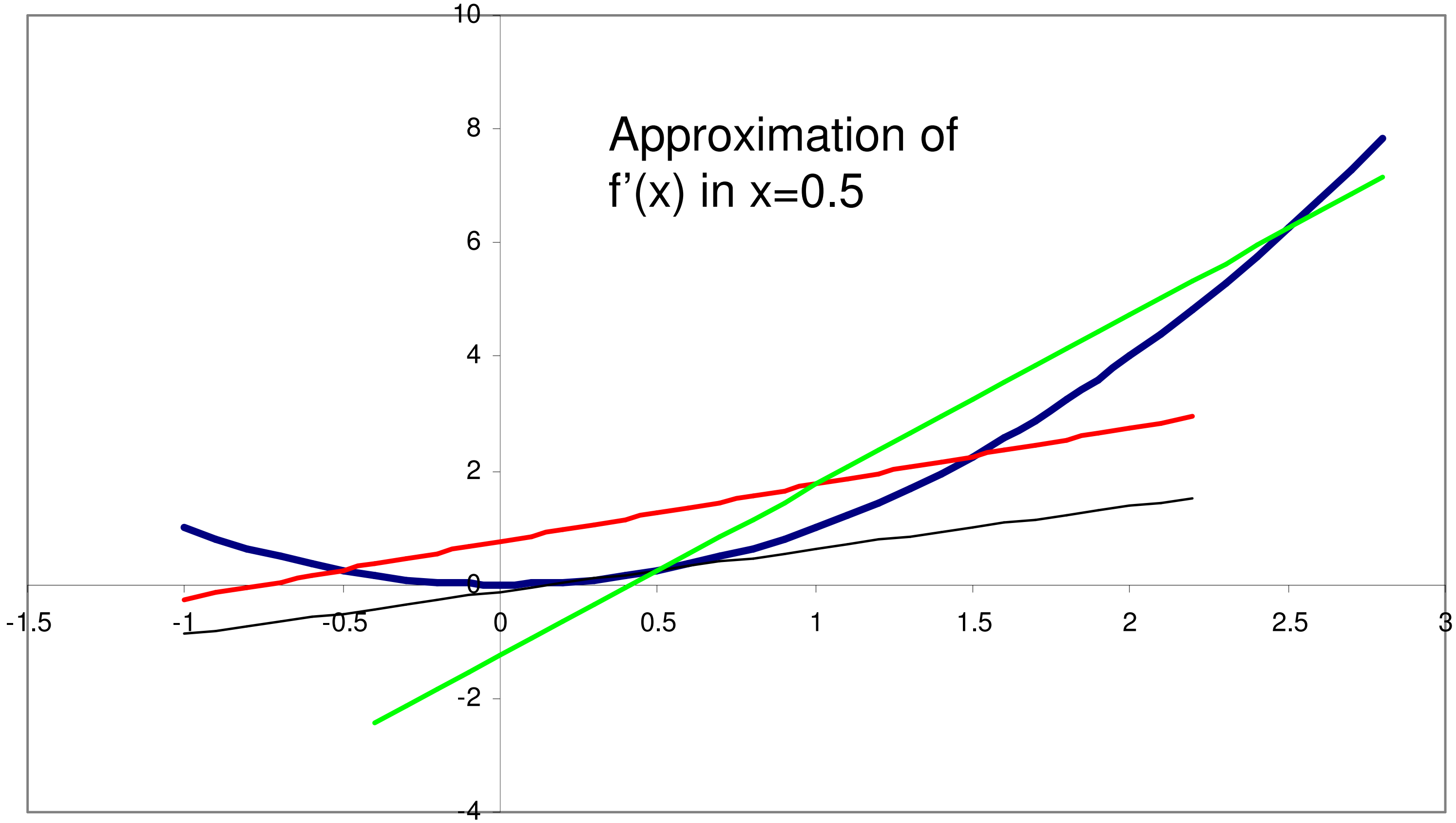
$$f\left(x-\frac{h}{2}\right)=f(x)-\frac{df(x)}{dx}\frac{h}{2}+\frac{1}{2}\frac{d^2f(x)}{dx^2}\left(\frac{h}{2}\right)^2-\frac{1}{6}\frac{d^3f(\vartheta)}{dx^3}\left(\frac{h}{2}\right)^3 \quad \vartheta\in\left[x-\frac{h}{2},x\right]$$

$$\frac{df(x)}{dx}=\frac{f\left(x+\frac{h}{2}\right)-f\left(x-\frac{h}{2}\right)}{h}-\frac{1}{6}\left[\frac{d^3f(\vartheta)}{dx^3}+\frac{d^3f(\xi)}{dx^3}\right]\frac{1}{2}\left(\frac{h}{2}\right)^2$$

$$\frac{df(x)}{dx}=\frac{f\left(x+\frac{h}{2}\right)-f\left(x-\frac{h}{2}\right)}{h}-\frac{1}{6}\frac{d^3f(\psi)}{dx^3}\left(\frac{h}{2}\right)^2 \quad \psi\in[\vartheta,\xi]\subset\left[x-\frac{h}{2},x+\frac{h}{2}\right]$$

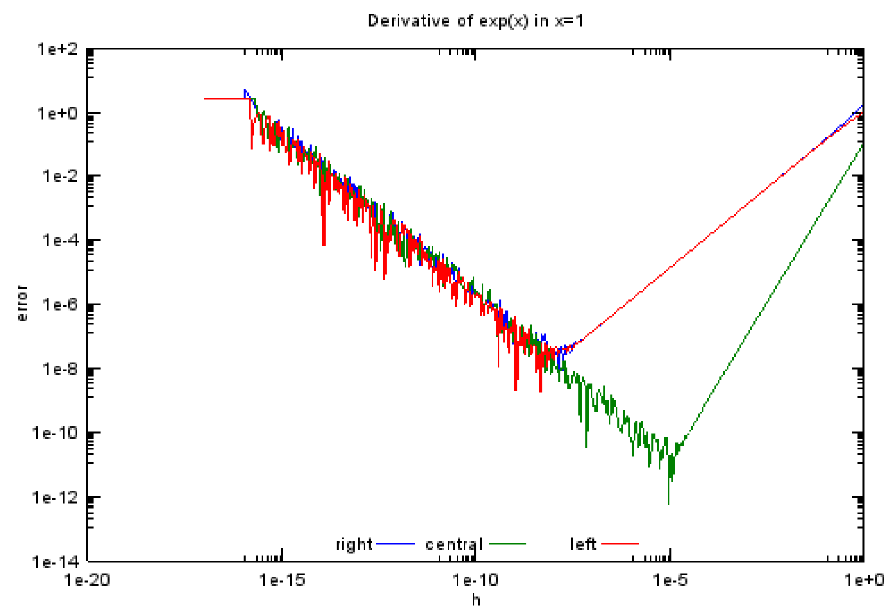
$$\frac{df(x)}{dx}=\frac{f\left(x+\frac{h}{2}\right)-f\left(x-\frac{h}{2}\right)}{h}+O\left(h^2\right)$$

Graphical Intuition



Stability Regions

- The convergence of order O(h^2) can be clearly observed in the loglog chart (*firstDeriv2.m*)
- We achieve very fast the "best we can do", at which point the size of the truncation error becomes comparable with the size of the round-off error (increasing on the same wiggly line as before). Then we see that further reductions in truncation error are negligible compared to the deterioration of the round-off error.
- It is generally difficult to predict how small can *h* be.



Source Code: *firstDeriv2.m*

```
function firstDeriv2 ( f, fp, fn, x, hMin, hMax )
    y  = f(x);  # exact value (note the semicolon at the end)

    # h vector: equally spaced points in log space
    hs = exp( log(hMin):(log(hMax)-log(hMin))/500:log(hMax) );
    lx = ratio( f, x, hs, -1,   0   );           # left estimator
    rx = ratio( f, x, hs,  0,   1   );           # right estimator
    cx = ratio( f, x, hs, -0.5, 0.5 );           # central estimator
    yp = fp(x) * ones(size(hs));              # exact value

    # plot errors in log space
    loglog( hs, abs(rx-yp), hs, abs(cx-yp), hs, abs(lx-yp) )
    xlabel("h");
    ylabel("error");
    title(["Derivative of ", fn, " in x=", num2str(x) ])  # note the string operations
    legend( "right", "central", "left", "location", ‘west')
endfunction

function yp = ratio( f, x, h, hLeftFact, hRightFact )
   xm = x + h * hLeftFact;
   xp = x + h * hRightFact;
   hs = xp-xm;
   yp = (f(xp)-f(xm)) ./ hs;
   yp(isnan(yp)) = 0.0;
endfunction
```

Is Symmetry Necessary?

- To achieve *O*(*h*²) we chose two symmetric points, *x+h* and *x-h*. Is that necessary?

$$\alpha\left[f(x+h_1) = f(x) + \frac{df(x)}{dx}h_1 + \frac{1}{2}\frac{d^2 f(x)}{dx^2}h_1^{\,2} + \frac{1}{6}\frac{d^3 f(\xi)}{dx^3}h_1^{\,3} \right] \quad \xi \in \left[\min(x, x+h_1), \max(x, x+h_1)\right]$$

$$\beta\left[f(x+h_2) = f(x) + \frac{df(x)}{dx}h_2 + \frac{1}{2}\frac{d^2 f(x)}{dx^2}h_2^{\,2} + \frac{1}{6}\frac{d^3 f(\vartheta)}{dx^3}h_2^{\,3} \right] \quad \vartheta \in \left[\min(x, x+h_2), \max(x, x+h_2)\right]$$

- We would like to choose α and β so that, adding the two equations the terms with power 0 and 2 disappear, and the term with power 1 has coefficient 1
- It is a system of 3 equations in 2 unknowns, hence we cannot solve it: we need a 3rd point!
- Symmetry makes the last equation linearly dependent on the first, which is why we make it with just 2 points

$$\begin{cases} \alpha + \beta = 0 \\ \alpha\, h_1 + \beta\, h_2 = 1 \\ \alpha\, h_1^{\,2} + \beta\, h_2^{\,2} = 0 \end{cases}$$

Is Higher Order Always Better?

- Is higher order of convergence guarantee of a smaller error? No. It usually yields smaller error, but not always
- Consider the following C^1 function :

$$f(x)=\begin{cases} 12x-16 & x \le 2 \\ x^3 & x > 2 \end{cases}$$

f'(2)=12

[f(2)-f(1)] / 1 = 12 (left differences is exact!)

[f(2.5)-f(1.5)] / 1 = 13.625 (central differences)

Richardson Extrapolation

- Using more points we can obtain approximations which converge with even higher order.
- Assuming f(x) is smooth:

$$f(x+h)=\sum_{k=0}^{\infty}\frac{1}{k!}f^{(k)}(x)\,h^k$$

$$f(x-h)=\sum_{k=0}^{\infty}\frac{1}{k!}f^{(k)}(x)\,(-h)^k$$

$$f(x+h)-f(x-h)=2h\;f^{(1)}(x)+2\sum_{k=1}^{\infty}\underbrace{\frac{1}{(2k+1)!}f^{(2k+1)}(x)}_{a_{2k}}\,h^{2k+1}$$

$$f^{(1)}(x)=\underbrace{\frac{f(x+h)-f(x-h)}{2h}}_{g(h)}-\sum_{k=1}^{\infty}a_{2k}\;h^{2k}$$

$$f^{(1)}(x)=g(h)+\sum_{k=1}^{\infty}a_{2k}\;h^{2k}$$

Richardson Extrapolation

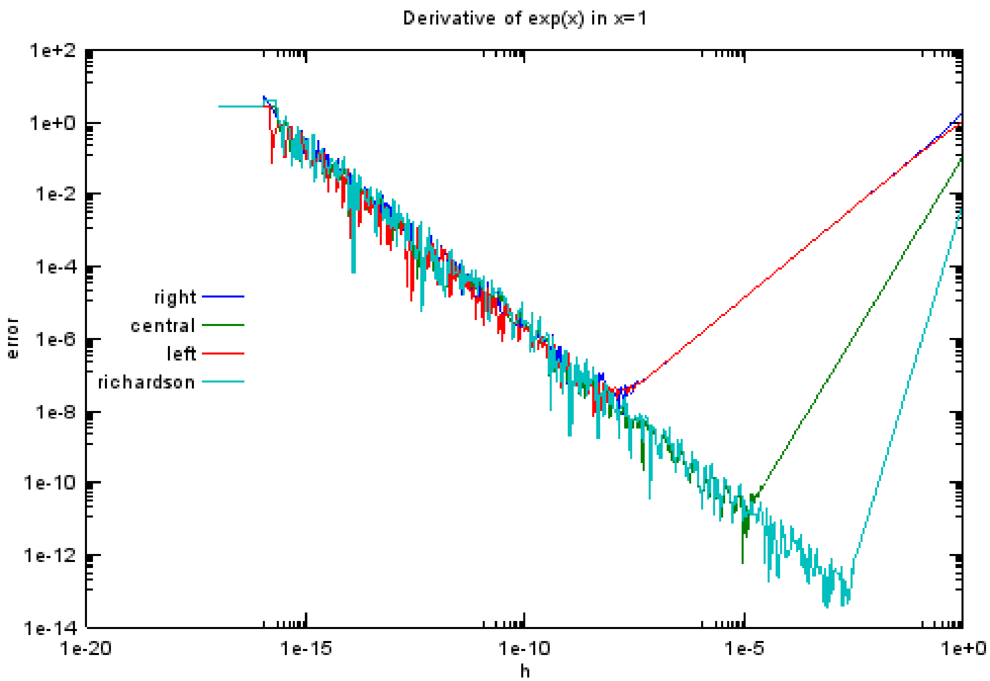
- Let's compute 4 points: f(x-h), f(x-h/2), f(x+h/2), f(x+h), we obtain the system of equations:

$$\begin{cases} f^{(1)}(x) = g(h) + \sum_{k=1}^{\infty} a_{2k}\, h^{2k} \\ f^{(1)}(x) = g\left(\frac{h}{2}\right) + \sum_{k=1}^{\infty} a_{2k} \left(\frac{h}{2}\right)^{2k} \end{cases} \Rightarrow \begin{cases} f^{(1)}(x) = g(h) + a_2\, h^2 + O\left(h^4\right) \\ f^{(1)}(x) = g\left(\frac{h}{2}\right) + a_2 \left(\frac{h}{2}\right)^2 + O\left(h^4\right) \end{cases}$$

- Subtracting the 2nd equation 4 times from the first one:

$$f^{(1)}(x) = \frac{4}{3}\, g\left(\frac{h}{2}\right) - \frac{1}{3}\, g(h) + O\left(h^4\right)$$

Stability Regions



- The convergence of order O(h^4) can be clearly observed in the loglog chart (*firstDeriv3.m*)
- We reach the "best-we-can-do" point sooner, i.e. we intercept the round-off error line at a larger h.

Richardson Extrapolation

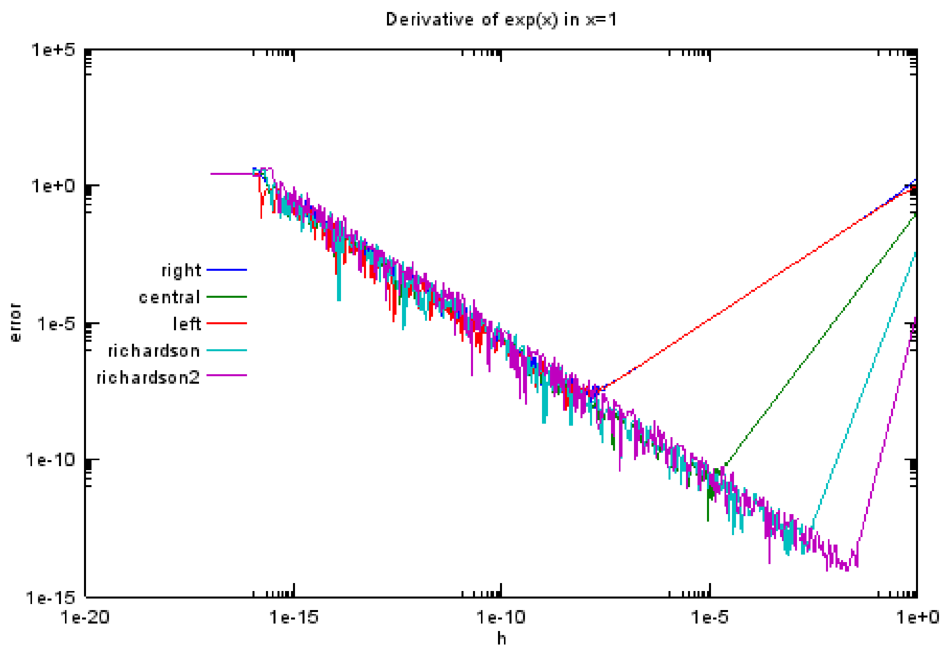
$$\begin{cases} \alpha\left[f^{(1)}(x) = g(h) + a_2\, h^2 + a_4\, h^4 + O\left(h^6\right)\right] \\ \beta\left[f^{(1)}(x) = g\left(\frac{h}{3}\right) + a_2\left(\frac{h}{3}\right)^2 + a_4\left(\frac{h}{3}\right)^4 + O\left(h^6\right)\right] \\ \gamma\left[f^{(1)}(x) = g\left(\frac{2h}{3}\right) + a_2\left(\frac{2h}{3}\right)^2 + a_4\left(\frac{2h}{3}\right)^4 + O\left(h^6\right)\right] \end{cases}$$

$$\begin{cases} \alpha + \beta + \gamma = 1 \\ \alpha + \frac{1}{9}\beta + \frac{4}{9}\gamma = 0 \\ \alpha + \frac{1}{81}\beta + \frac{16}{81}\gamma = 0 \end{cases} \Rightarrow \begin{cases} \alpha = \frac{1}{10} \\ \beta = \frac{3}{2} \\ \gamma = -\frac{3}{5} \end{cases}$$

$$f^{(1)}(x) = \frac{3}{2} g\left(\frac{h}{3}\right) - \frac{3}{5} g\left(\frac{2h}{3}\right) + \frac{1}{10} g(h) + O\left(h^6\right)$$

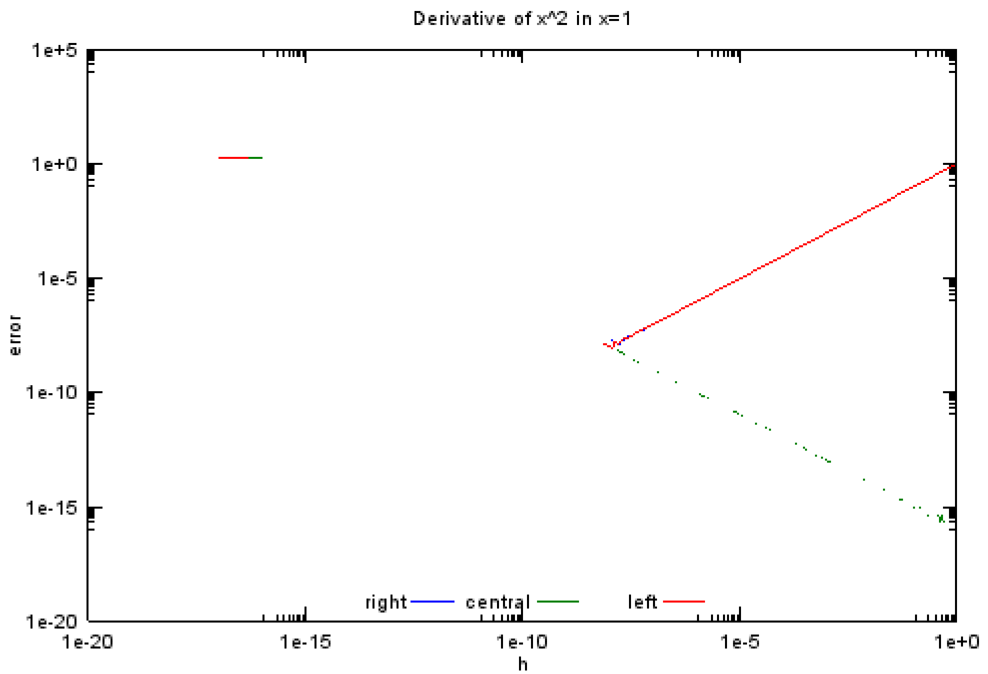
- We could use more points using a arbitrary set of chosen *h*, e.g 6 points located at $\pm h$, $\pm h/3$, $\pm 2h/3$
- Classic Richardson approach keeps halving the increment, i.e. *h*, *h*/2, *h*/4

Stability Regions



- The convergence of order O(h^6) can be clearly observed in the loglog chart (*firstDeriv4.m*)
- The range of *h* where the approximation is stable and accuracy is strictly superior to the previous Richardson scheme is very small
- We are getting very close to machine precision. Doing better becomes harder and harder and certainly we cannot exceed machine precision.

A Simple Polynomial




```
>> function d=g(x) d=x.*x;
    endfunction
>> function d=gp(x) d=2*x;
    endfunction
>> firstDeriv2(@g,@gp,1,1e-17,1)
```

- Central differences starts from an accuracy comparable with machine precision. How do we explain that?
- A 2nd order polynomial has only Taylor terms up to 2nd order, therefore our approximation is exact
- In other words, central differences is equivalent to construct a 2nd order polynomial passing by the points f(*x*-*h*), f(*x*), f(*x*+*h*), and then computing the derivative analytically at *x*

In general, any N-points finite difference schemes, consists of constructing a polynomial which pass through the N points, then take its derivative at x

Further Readings

- Online lecture Notes from Prof Binegar
  http://www.math.okstate.edu/~binegar/4513-F98/4513-l18.pdf

- Richardson Extrapolation
  http://en.wikipedia.org/wiki/Richardson_extrapolation