Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation
2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

MCS 471 Lecture 24
Numerical Analysis
Jan Verschelde, 18 October 2021

Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation
2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

the limit definition and Taylor expansion

Give a function $f(x)$, its derivative is defined as

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

For some $x = a$ and $h > 0$, consider the approximation

$$f'(a) \approx \frac{f(a+h) - f(a)}{h}.$$

The above formula is called a *forward difference formula*.

An alternative derivation follows the Taylor expansion of f at $x = a$:

$$f(a+h) = f(a) + hf'(a) + O(h^2) \quad \Rightarrow \quad f'(a) \approx \frac{f(a+h) - f(a)}{h}.$$

forward, backward, and central difference formulas

Given a function $f(x)$, we can approximate f' at $x = a$ with

1. a forward difference formula:

$$f'(a) \approx \frac{f(a+h) - f(a)}{h}$$

2. a backward difference formula:

$$f'(a) \approx \frac{f(a) - f(a-h)}{h}$$

3. a central difference formula:

$$f'(a) \approx \frac{f(a+h/2) - f(a-h/2)}{h}$$

first and second order formulas

- The error of forward and backward difference formulas is $O(h)$. Because of $O(h^1)$, the error is of first order.
- The error of the central difference formula is $O(h^2)$. which is of second order.

Numerically, if $h = 10^{-2}$, then first order gives an error of about 10^{-2}, whereas the error of a second order formula will be about 10^{-4}.

Exercise 1: Consider $f(x) = \exp(x)$, $a = 1$, and $h = 0.01$.

1. Evaluate the forward, backward, and central difference formulas to approximate $f'(1)$.
2. Compute the error for three approximations, using $f'(1) = \exp(1)$. Which formula gives the most accurate result? In your explanation, refer to the order of each formula.

a numerical experiment

What is the right value for h?

$$f'(a) \approx \frac{f(a+h) - f(a)}{h}$$

Consider the following:

- h too large: $O(h^2)$ term matters,
- h too small: dividing by h causes roundoff to magnify.

Setup for a numerical experiment:

1. Take $f(x) = e^x$, which has an easy $f'(x) = e^x$.
2. Start with $h = 1$ and divide h by 10 in each step.
3. Compare $\left| e^x - \frac{e^{a+h} - e^a}{h} \right|$ to h.

a Julia program – the setup

```
fwdexp(x, h) = (exp(x+h) - exp(x))/h

print("Give a value for x : ")
line = readline(stdin)
xval = parse(Float64, line)
print("Give an upper bound for h : ")
line = readline(stdin)
hval = parse(Float64, line)
print("Give the number of approximations : ")
line = readline(stdin)
nbr = parse(Float64, line)
```

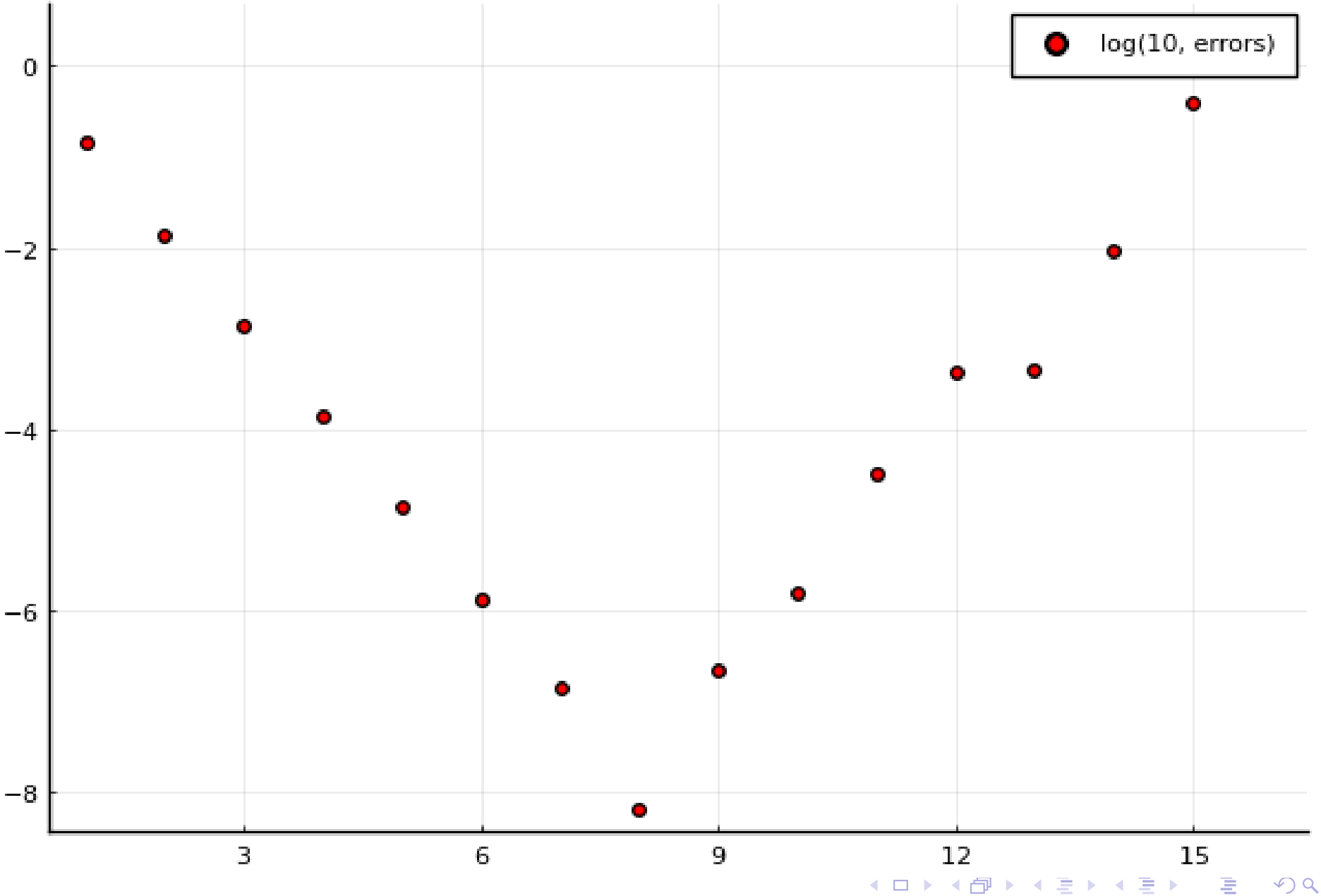
a Julia program – the loop

```
exact = exp(xval)
strexact = @sprintf("%.16e", exact)
println("Approximating $strexact :")
for i=1:nbr
    approx = fwdexp(xval, hval)
    err = abs(exact - approx)
    strhval = @sprintf("%.3e", hval)
    strapprox = @sprintf("%.16e", approx)
    strexact = @sprintf("%.16e", exact)
    strerr = @sprintf("%.3e", err)
    println("  $strhval   $strapprox   $strerr")
    hval = hval/10.0
end
```

running the program at the command prompt

```
$ julia difforward.jl
Give a value for x : 1
Give an upper bound for h : 0.1
Give the number of approximations : 16
Approximating 2.7182818284590451e+00 :
  1.000e-01   2.8588419548738830e+00   1.406e-01
  1.000e-02   2.7319186557871245e+00   1.364e-02
  1.000e-03   2.7196414225332255e+00   1.360e-03
  1.000e-04   2.7184177470829241e+00   1.359e-04
  1.000e-05   2.7182954199567173e+00   1.359e-05
  1.000e-06   2.7182831874306141e+00   1.359e-06
  1.000e-07   2.7182819684057331e+00   1.399e-07
  1.000e-08   2.7182818218562939e+00   6.603e-09
  1.000e-09   2.7182820439008983e+00   2.154e-07
  1.000e-10   2.7182833761685279e+00   1.548e-06
  1.000e-11   2.7183144624132174e+00   3.263e-05
  1.000e-12   2.7187141427020829e+00   4.323e-04
  1.000e-13   2.7178259642823828e+00   4.559e-04
  1.000e-14   2.7089441800853815e+00   9.338e-03
  1.000e-15   3.1086244689504379e+00   3.903e-01
  1.000e-16   0.0000000000000000e+00   2.718e+00
$
```

a log error plot



errors of finite differences

We observe the most accurate approximation at $h = 10^{-8}$:

- for $h > 10^{-8}$, the $O(h^2)$ dominates,
- for $h < 10^{-8}$, the roundoff on $f(\cdot)$ dominates.

Exercise 2:
Set up a numerical experiment to approximate the derivative of $\cos(x)$ at $x = 0$, with *central* difference formulas.

Try values $h = 10^{-p}$ for p ranging from 1 to 16.

For which value of p do you observe the most accurate approximation?

Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation
2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

differentiation by interpolation

The condition of the differentiation problem is often stated as very poor because small random noise on the input function values gives huge differences in the values of the derivatives.

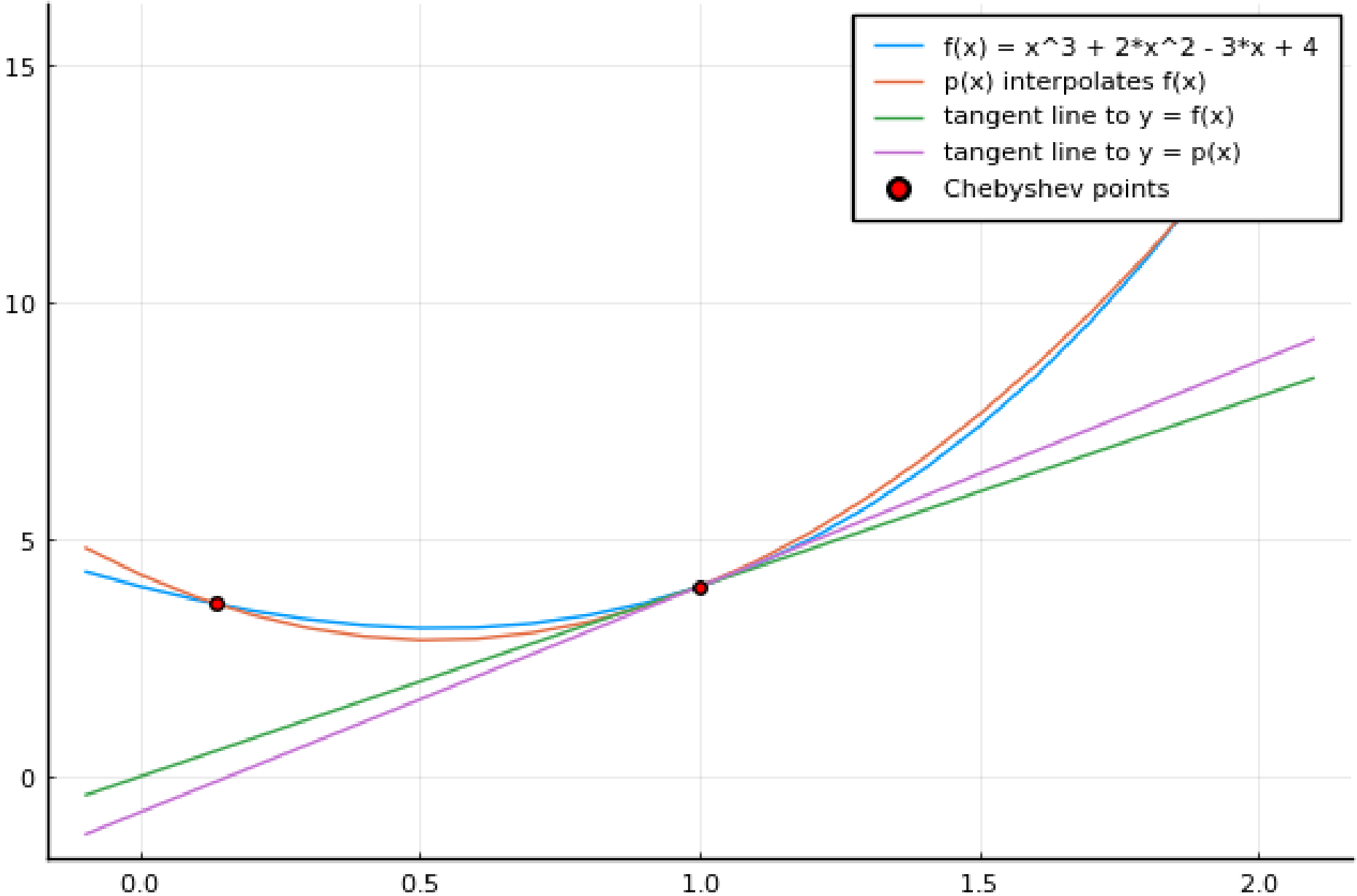
We can take an interpolating polynomial for $f(x)$ and approximate $f'(x)$ by $p'(x)$.

Experimental setup:

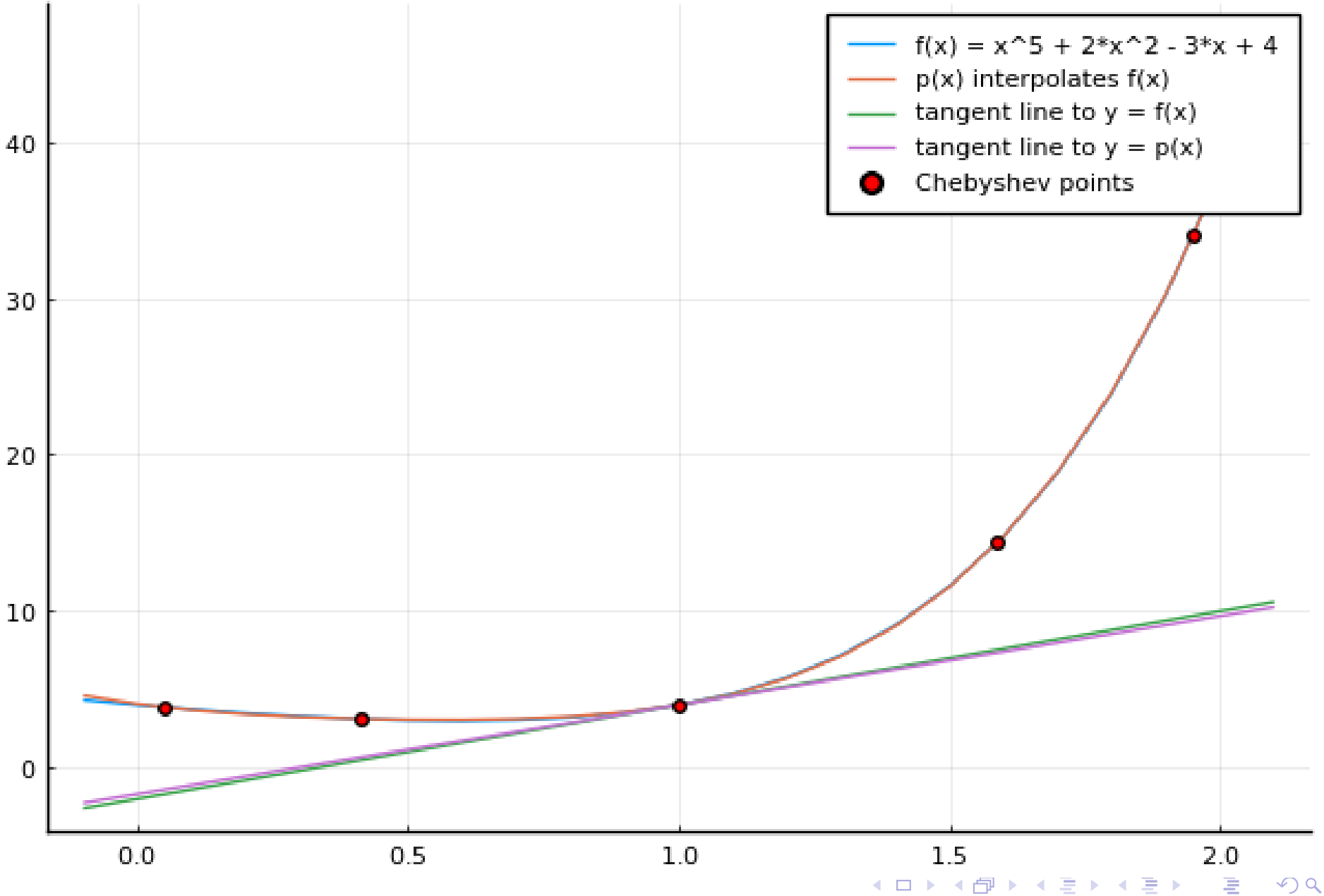
1. Consider $f(x)$ over $[0, 2]$.
2. Interpolate at Chebyshev points $x_k = \cos((2k-1)\pi/(2n))$, $k = 1, 2, \ldots, n$.
3. Compare $p'(x_k)$ with $f'(x_k)$.

Experiment with quadric: $p'(x_1) = 4.75$, $f'(x_1) = 4$.
Experiment with quartic: $p'(x_2) = 5.69$, $f'(x_2) = 6$.

the derivative of an interpolating quadric



the derivative of an interpolating quartic



interpolation at higher degrees

Did our experiment now work?

While taking more points in the interpolation improves,
our problem is a local problem, we want more accuracy
nearby the point where we want to compute the derivative.

We can obtain highly accurate results by

1. computing more approximations for smaller values of h, and
2. extrapolating on those approximations,

as will be explained next.

For numerical differentiation, extrapolation works, not interpolation.

Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation

2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

Taylor expansions

Consider the Taylor expansion of $f(x)$ at $x = a$:

$$f(a+h) = f(a) + hf'(a) + h^2 \frac{f''(a)}{2} + O(h^3).$$

We want to compute $f'(a)$ and rewrite accordingly:

$$\begin{aligned} f(a+h) - f(a) &= hf'(a) + h^2 \frac{f''(a)}{2} + O(h^3), \\ \frac{f(a+h) - f(a)}{h} &= f'(a) + h\frac{f''(a)}{2} + O(h^2). \end{aligned}$$

In the last formula, replace h by $h/2$:

$$\frac{f\left(a + \frac{h}{2}\right) - f(a)}{\frac{h}{2}} = f'(a) + \frac{h}{2}\frac{f''(a)}{2} + O(h^2).$$

eliminating error terms

With $D(f,a,h) = \dfrac{f(a+h)-f(a)}{h}$, the last two formulas simplify:

$$\begin{aligned} D(f,a,h) &= f'(a) + h\frac{f''(a)}{2} + O(h^2), \\ D\left(f,a,\frac{h}{2}\right) &= f'(a) + \frac{h}{2}\frac{f''(a)}{2} + O(h^2). \end{aligned}$$

Now we can eliminate the $f''(a)$ term:

$$2D\left(f,a,\frac{h}{2}\right) - D(f,a,h) = (2-1)f'(a) + O(h^2).$$

Then we have an $O(h^2)$ approximation for $f'(a)$:

$$f'(a) = 2D\left(f,a,\frac{h}{2}\right) - D(f,a,h) + O(h^2).$$

Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation

2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

more terms in the Taylor series

$$f(a+h) = f(a) + hf'(a) + h^2\frac{f^{(2)}(a)}{2} + h^3\frac{f^{(3)}(a)}{3!} + h^4\frac{f^{(4)}(a)}{4!} + O(h^5).$$

With $D(f,a,h) = \dfrac{f(a+h)-f(a)}{h}$, we rearrange:

$$D(f,a,h) = f'(a) + h\frac{f^{(2)}(a)}{2} + h^2\frac{f^{(3)}(a)}{3!} + h^3\frac{f^{(4)}(a)}{4!} + O(h^4).$$

We drop the $O(h^4)$ term and introduce coefficients $c_k = \dfrac{f^{(k)}(a)}{k!}$:

$$\begin{aligned}
D(f,a,h) &\approx f'(a) + c_1 h + c_2 h^2 + c_3 h^3 \\
D(f,a,h/2) &\approx f'(a) + c_1 h/2 + c_2 h^2/4 + c_3 h^3/8 \\
D(f,a,h/4) &\approx f'(a) + c_1 h/4 + c_2 h^2/16 + c_3 h^3/64 \\
D(f,a,h/8) &\approx f'(a) + c_1 h/8 + c_2 h^2/64 + c_3 h^3/512
\end{aligned}$$

By linear combinations we obtain an $O(h^4)$ approximation for $f'(a)$.

eliminating more error terms

$$
\begin{aligned}
D(f,a,h) &\approx f'(a) + c_1 h + c_2 h^2 + c_3 h^3 \\
D(f,a,h/2) &\approx f'(a) + c_1 h/2 + c_2 h^2/4 + c_3 h^3/8 \\
D(f,a,h/4) &\approx f'(a) + c_1 h/4 + c_2 h^2/16 + c_3 h^3/64 \\
D(f,a,h/8) &\approx f'(a) + c_1 h/8 + c_2 h^2/64 + c_3 h^3/512
\end{aligned}
$$

Observe the pattern:

1. to eliminate h : $(\ 2 \times (\text{ more accurate }) - \text{previous }\ )/(2-1)$,
2. to eliminate h^2: $(\ 4 \times (\text{ more accurate }) - \text{previous }\ )/(4-1)$,
3. to eliminate h^3: $(\ 8 \times (\text{ more accurate }) - \text{previous }\ )/(8-1)$.

a triangular table

Define the table $R[i,j]$, for $i = 1, 2, \ldots, n$, and $j = 1, 2, \ldots, i$.

The first column contains forward differences:

$$R[i,1] = D(f, a, h/2^{i-1}), \quad i = 1, 2, \ldots, n.$$

For $j > 1$, we apply the formula:

$$R[i,j] = \frac{2^{j-1} R[i, j-1] - R[i-1, j-1]}{2^{j-1} - 1}, \quad i = j, j+1, \ldots, n.$$

The algorithm is called *Richardson extrapolation*.

If the error of $R[n,1]$ is $O(h)$, then the error of $R[n,2]$ is $O(h^2)$, and then the error of $R[n,3]$ is $O(h^3)$, ..., the error of $R[n,k]$ is $O(h^k)$, for column k in the table.

Numerical Differentiation

1. Finite Difference Formulas
   - the limit definition and Taylor expansion
   - differentiation by interpolation

2. Richardson Extrapolation
   - eliminating error terms
   - higher order approximations
   - a Julia function

a Julia function

```
"""
    richardson(f::Function,z::Float64,h::Float64,n::Int)

returns the triangular table of numerical approximations of
the derivative of f at z.
"""
function richardson(f::Function,z::Float64,h::Float64,n::Int)
    R = zeros(n, n)
    w = h
    for i=1:n
        R[i, 1] = (f(z+w) - f(z))/w
        w = w/2
    end
    for j=2:n
        for i=j:n
            w = 2^(j-1) - 1
            R[i, j] = (2^(j-1)*R[i, j-1] - R[i-1, j-1])/w
        end
    end
    return R
end
```

a test on the exponential function

$f(x) = \exp(x)$, $a = 1.0$, $h = 0.1$, $n = 4$:

```
$ julia richardson.jl
Richardson extrapolation on exp(x) :
4×4 Array{Float64,2}:
 2.85884195e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 2.78738579e+00  2.71592963e+00  0.00000000e+00  0.00000000e+00
 2.75254528e+00  2.71770478e+00  2.71829649e+00  0.00000000e+00
 2.73534210e+00  2.71813892e+00  2.71828363e+00  2.71828179e+00

The table with errors :
4×4 Array{Float64,2}:
 1.40560126e-01  0.00000000e+00  0.00000000e+00  0.00000000e+00
 6.91039636e-02  2.35219917e-03  0.00000000e+00  0.00000000e+00
 3.42634558e-02  5.77051997e-04  1.46637268e-05  0.00000000e+00
 1.70602718e-02  1.42912242e-04  1.80100989e-06  3.65210884e-08
$
```

Observe the magnitude of the errors in the columns. We have a 4-th order approximation. With $h = 0.1/8 = 0.0125$, we obtain an error 3.7×10^{-8}.

one last exercise

Exercise 3:

Use extrapolation on forward differences to compute a third order approximation for the derivative of $\cos(x)$ at $\frac{\pi}{4}$.
Start with $h = 0.1$.

1. Compute the error of the approximation.
2. Verify that the approximation is indeed of third order.

concluding remarks on Richardson extrapolation

Two remarks:

- For Richardson extrapolation to be effective, all terms in the Taylor series should appear with nonzero coefficients.

  Some functions are odd or even, for example:

  - expanding $\sin(x)$ at $x = 0$ gives $x - x^3/6 + x^5/120 + O(x^6)$,
  - expanding $\cos(x)$ at $x = 0$ gives $1 - x^2/2 + x^4/24 + O(x^6)$.

- Richardson extrapolation starting with central difference formulas will lead to more accurate results.