PDE: Lab Report 1

Kieran Molloy

December 11, 2023

Abstract

Investigating fourth order Taylor series expansions using undetermined coefficients and MatLab

1 Introduction

The study was undertaken as part of MMU's Partial Differential Equations unit, under supervision of Jon Shiach with the aim to better understand fourth-order Taylor Series expanions and the methods surrounding them.

2 Numerical model

The fourth-order Taylor series expansion of the function $f(x)$ can be written as (Shiach and Rattenbury, 2016)

$$f(x+h) = f(x) + hf_x(x) + \frac{h^2}{2}f_{xx}(x) + \frac{h^3}{6}f_{xxx}(x) + \frac{h^4}{24}f_{xxxx}(x) + O(h^5),$$

$$f(x+h) = f(x) - hf_x(x) + \frac{h^2}{2}f_{xx}(x) - \frac{h^3}{6}f_{xxx}(x) + \frac{h^4}{24}f_{xxxx}(x) + O(h^5),$$

$$f(x+2h) = f(x) + 2hf_x(x) + 2h^2f_{xx}(x) + \frac{4h^3}{3}f_{xxx}(x) + \frac{2h^4}{3}f_{xxxx}(x) + O(h^5),$$

$$f(x-2h) = f(x) - 2hf_x(x) + 2h^2f_{xx}(x) - \frac{4h^3}{3}f_{xxx}(x) + \frac{2h^4}{3}f_{xxxx}(x) + O(h^5),$$

where h is the step length and $f_x(x), f_{xx}(x), \ldots$ denotes the partial derivatives of $f(x)$ with respect to x.

2.1 Finite-difference approximations

The fourth-order central difference approximation of $f_x(x)$ is given in Eq. (1)

$$f_x(x) = \frac{f_{i-2} - 8f_{i-1} + 8f_{i+1} - f_{i+2}}{12h} + O(h^5). \tag{1}$$

$$f_x(x) = \frac{f(x-2h) - f(x-h) + f(x+h) - f(x+2h)}{12h} + O(h^5). \tag{2}$$

3 Results

The approximate values of the foward, backward, second and fourth order finite-difference for $\frac{\partial}{\partial x}cos(x)$ at $f_x(\pi/6)$ for different step lengths are shown in Table 1.

Table 1: The approximations for the forward, backward and central difference approximations of $\frac{\partial}{\partial x}cos(x)$ at $f_x(\pi/6)$.

h	forward	backward	second	fourth
0.5000	0.707972	0.952807	0.830389	0.864274
0.2500	0.794857	0.919208	0.857032	0.865913
0.1250	0.832563	0.894981	0.863772	0.866018
0.0625	0.849842	0.881082	0.865462	0.866025
0.0313	0.858073	0.873696	0.865884	0.866025

The absolute errors between the finite-difference approximations of $\frac{\partial}{\partial x}cos(x)$ at $f_x(\pi/6)$ for different step lengths are shown in Table 2 and plotted on a loglog scale in Fig. 1.

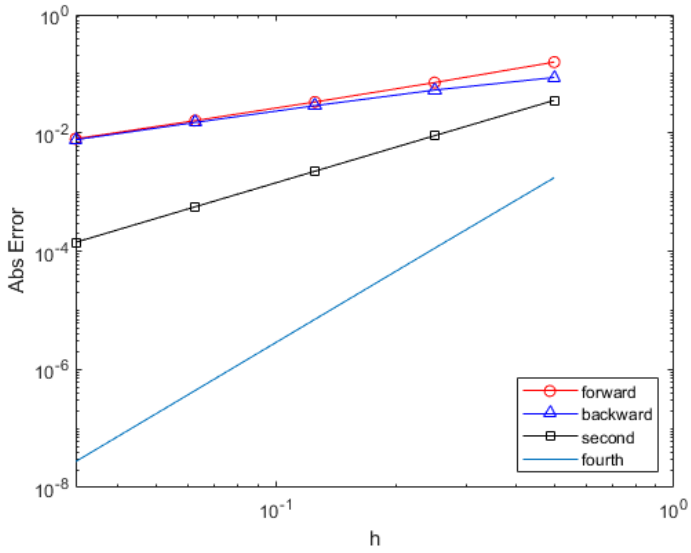

Figure 1: Loglog plot of the absolute errors between the finite-difference approximations using different values of h and the exact derivative of $\frac{\partial}{\partial x}e^x$ at $x = 1$.

Table 2: The absolute errors for the forward, backward and central difference approximations of $\frac{\partial}{\partial x}e^x$ at $x = 1$.

h	forward	backward	second	fourth
0.5000	0.158053	0.086781	0.035636	0.001751
0.2500	0.071168	0.053182	0.008993	0.000112
0.1250	0.033463	0.028956	0.002254	0.000007
0.0625	0.016184	0.015056	0.000564	0.000000
0.0313	0.007953	0.007671	0.000141	0.000000

The results can be checked with $\frac{log(\alpha)-log(\beta)}{log(h_1)-log(h_5)}$ where α is error at the first step length, and β is error at the fifth step length, the results of this calculation are shown in Table 3.

Table 3: The approximation of N iterations of Taylor

forward	backward	second	fourth
1.0782	0.8750	1.9955	3.9893

4 Conclusions

The study shows that error lowers significantly using higher order methods, In table 2 $h = 0.0625$ at fourth order gives an error below 6 decimal places compared to 0.15 at $h = 0.5$ using first order. Table 3 gives an indication of what order each result is, with forward and backward close to 1, second close to 2, and fourth close to 4. This accurately represents which order each data set represents.

References

Shiach, J. and Rattenbury, N. (2016). *Runge-Kutta Methods and Computational Linear Algebra*. Lecture Notes. Manchester Metropolitan University.

A Appendix section

A.1 Matlab Code

```
% clear stuff
clear; clc;
% Initial X Value
x=pi/6;
% Step Sizes
h=[0.5 0.25 0.125 0.0625 0.03125];
% f function
f = @(x) sin(x);
% Exact F Value
fexact = @(x) cos(x);

%FD Approx
fw = (f(x+h)-f(x)) ./ h; %Forward
bw = (f(x)-f(x-h)) ./ h; %Backward
so = (f(x+h)-f(x-h)) ./ (2*h); % Second
fo = (f(x-2*h)-8*f(x-h)+8*f(x+h)-f(x+2*h)) ./ (12*h); % Fourth
% Table 1: Approximations
fprintf('  h  forward backward second fourth\n')
for i=1:5
    fprintf('%1.4f %1.6f %1.6f %1.6f %1.6f\n',h(i),fw(i),bw(i),so(i),fo(i))
end
% Error Calculations
fw_err = abs(fexact(x)-fw);
bw_err = abs(fexact(x)-bw);
so_err = abs(fexact(x)-so);
fo_err = abs(fexact(x)-fo);
% Table 2: Error
fprintf('  h  forward backward second\n')
for i=1:5
    fprintf('%1.4f %1.6f %1.6f %1.6f %1.6f\n',h(i),fw_err(i),bw_err(i),so_err(i),fo_err
    (i))
end
% Figure 1: Error
loglog(h,fw_err,'ro-',h,bw_err,'b^-',h,so_err,'ks-',h,fo_err,'')
xlabel('h')
ylabel('Abs Error')
legend('forward','backward','second','fourth','location','southeast')

% Table 3: N Iterations
format short
fw_n = (log(fw_err(1))-log(fw_err(5))) / (log(h(1)) - log(h(5)))
bw_n = (log(bw_err(1))-log(bw_err(5))) / (log(h(1)) - log(h(5)))
so_n = (log(so_err(1))-log(so_err(5))) / (log(h(1)) - log(h(5)))
fo_n = (log(fo_err(1))-log(fo_err(5))) / (log(h(1)) - log(h(5)))
```