Comparing Methods of First Derivative Approximation

Forward, Backward and Central Divided Difference

Ana Catalina Torres, Autar Kaw

University of South Florida

United States of America

kaw@eng.usf.edu

Introduction

This worksheet demonstrates the use of Maple to compare the approximation of first order derivatives using three different methods. Each method uses a point h ahead, behind or both of the given value of x at which the first derivative of $f(x)$ is to be found.

Forward Difference Approximation *(FDD)*

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

Backward Difference Approximation *(BDD)*

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

Central Difference Approximation *(CDD)*

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2\,h}$$

Initialization

```
> restart;
  with (plots) :
```

Section 1: Input

The following simulation approximates the first derivative of a function using different methods of approximation *(FDD,BDD,CDD)*. The user inputs are

a) function, $f(x)$
b) point at which the derivative is to be found, xv
c) starting step size, h
d) number of times user wants to halve the step size, n

The outputs include

a) approximate values of the first derivative at the point and initial step size given using different types of approximation
b) exact value
c) absolute relative true error, absolute relative approximate error, and number of at least correct significant digits in the solution as a function of step size.

Function $f(x)$.

```
> f := x→x·exp(2·x);
```

$$f := x \rightarrow x\, e^{2x} \tag{3.1}$$

Value of x at which $f(x)$ is desired, xv

```
> xv := 4.0;
```

$$xv := 4.0 \tag{3.2}$$

Starting step size, h

```
> h := 0.2;
```

$$h := 0.2 \tag{3.3}$$

Number of times step size is halved

```
> n := 6;
```

$$n := 6 \tag{3.4}$$

This is the end of the user section. All the information must be entered before proceeding to the next section. Re-execute the program.

Section 2: Procedure

The following procedure estimates the solution of first order derivate of an equation at a point xv using different methods of approximation.

$f(x)$ = function
xv = value at which the solution is desired
h = step size value
n = number of times step size is halved

```
> Forward Divided Difference Procedure
FDD :=proc(f, xv, h)
  local deriv :
   deriv := (f(xv+h) - f(xv)) / h :
  return (deriv) :
end proc:
```

```
> Backward Divided Difference Procedure
BDD :=proc(f, xv, h)
 local deriv :
  deriv := (f(xv) - f(xv-h)) / h :
 return (deriv) :
end proc:
```

```
> Central Divided Difference Precedure
CDD :=proc(f, xv, h)
 local deriv :
  deriv := (f(xv+h) - f(xv-h)) / (2·h) :
 return (deriv) :
```

```
end proc:
```

Section 3: Calculation

The exact value Ev of the first derivative of the equation:
First, using the *diff* command the solution is found. In a second step, the exact value of the derivative is shown.

```
> y(x) = f(x);
```

$$y(x) = x\,e^{2x} \tag{5.1}$$

```
> Soln := diff(f(x), x);
```

$$Soln := e^{2x} + 2\,x\,e^{2x} \tag{5.2}$$

```
> Ev := evalf(subs(x = xv, Soln));
```

$$Ev := 26828.62188 \tag{5.3}$$

The next loop calculates the following:
Av: Approximate value of the first derivative using various first derivative approximation methods by calling the procedures *"FDD", "BDD",* and *"CDD"*
Ev: Exact value of the second derivative
et: Absolute relative true percentage error
ea: Absolute relative approximate percentage error
Sig: Least number of correct significant digits in an approximation

```
> for i from 0 by 1 to n-1 do
    N[i] := 2^i :
    H[i] := h/N[i] :
    AvFDD[i] := FDD(f, xv, H[i]) :
    AvBDD[i] := BDD(f, xv, H[i]) :
    AvCDD[i] := CDD(f, xv, H[i]) :
    etFDD[i] := abs((Ev - AvFDD[i])/Ev)·100 :
    etBDD[i] := abs((Ev - AvBDD[i])/Ev)·100 :
    etCDD[i] := abs((Ev - AvCDD[i])/Ev)·100 :
    if (i > 0) then
      eaFDD[i] := abs((AvFDD[i] - AvFDD[i-1])/AvFDD[i])·100 :
      eaBDD[i] := abs((AvBDD[i] - AvBDD[i-1])/AvBDD[i])·100 :
      eaCDD[i] := abs((AvCDD[i] - AvCDD[i-1])/AvCDD[i])·100 :
      SigFDD[i] := floor(2 - log10(eaFDD[i]/0.5)) :
```

```
    SigBDD[i] := floor(2 - log10(eaBDD[i] / 0.5)) :
    SigCDD[i] := floor(2 - log10(eaCDD[i] / 0.5)) :
     if SigFDD[i] < 0  then
      SigFDD[i] := 0 :
     end if:
     if SigBDD[i] < 0  then
      SigBDD[i] := 0 :
     end if:
     if SigCDD[i] < 0  then
      SigCDD[i] := 0 :
     end if:
   end if:
  end do:
```

The loop halves the value of the step size n times. Each time, the approximate values of the first derivative are calculated and saved in different vectors depending on the method of approximation. The approximate error is calculated after at least two approximate values of the first derivative have been saved. The number of significant digits is calculated and written as the lowest real number for each method. If the number of significant digits calculated is less than zero, then is shown as zero.

Section 4: Spreadsheet

The next table shows the step size value, exact value, approximate value, and the least number of correct significant digits in an approximation as a function of the step size value for each method of approximation.

```
> with(Spread) :
   tableoutput := CreateSpreadsheet("FDD, BDD and CDD Comparison") :
   SetCellFormula(tableoutput, 1, 2, "Step Size") :
   SetCellFormula(tableoutput, 1, 3, "Exact Value") :
   SetCellFormula(tableoutput, 1, 4, "Approx Value FDD") :
   SetCellFormula(tableoutput, 1, 5, "Abs Rel True Error FDD") :
   SetCellFormula(tableoutput, 1, 6, "Approx Value BDD") :
   SetCellFormula(tableoutput, 1, 7, "Abs Rel True Error BDD") :
   SetCellFormula(tableoutput, 1, 8, "Approx Value CDD") :
   SetCellFormula(tableoutput, 1, 9, "Abs Rel True Error CDD");
  for i from 0 by 1 to n-1 do
     SetCellFormula(tableoutput, i+2, 1, i+1) :
     SetCellFormula(tableoutput, i+2, 2, evalf(H[i])) :
     SetCellFormula(tableoutput, i+2, 3, evalf(Ev)) :
     SetCellFormula(tableoutput, i+2, 4, evalf(AvFDD[i])) :
     SetCellFormula(tableoutput, i+2, 5, evalf(etFDD[i])) :
     SetCellFormula(tableoutput, i+2, 6, evalf(AvBDD[i])) :
     SetCellFormula(tableoutput, i+2, 7, evalf(etBDD[i])) :
```

```
    SetCellFormula(tableoutput, i+2, 8, evalf(AvCDD[i])) :
    SetCellFormula(tableoutput, i+2, 9, evalf(etCDD[i]));
  end do:
  EvaluateSpreadsheet(tableoutput) :
```

FDD, BDD and CDD Comparison

	...	B	C	D	E	F	G	H	I
1		"Step Size"	"Exact Value"	"Approx Value FDD"	"Abs Rel True Error FDD"	"Approx Value BDD"	"Abs Rel True Error BDD"	"Approx Value CDD"	"Abs Rel True Error CDD"
2	1	0.2	26828.62188	33769.24195	25.87020720	21653.43774	19.28978746	27711.33985	3.290209888
3	2	0.1000000000	26828.62188	30040.64310	11.97236755	24054.84236	10.33888186	27047.74273	0.8167428464
4	3	0.05000000000	26828.62188	28375.27500	5.764936891	25391.33500	5.357289265	26883.30500	0.2038238127
5	4	0.02500000000	26828.62188	27587.71240	2.829405563	26096.86080	2.727538832	26842.28660	0.05093336535
6	5	0.01250000000	26828.62188	27204.68080	1.401707929	26459.39520	1.376241693	26832.03800	0.01273311770
7	6	0.006250000000	26828.62188	27015.78880	0.6976389650	26643.16320	0.6912717352	26829.47600	0.003183614886
8									
9									
10									
11									

(6.1)

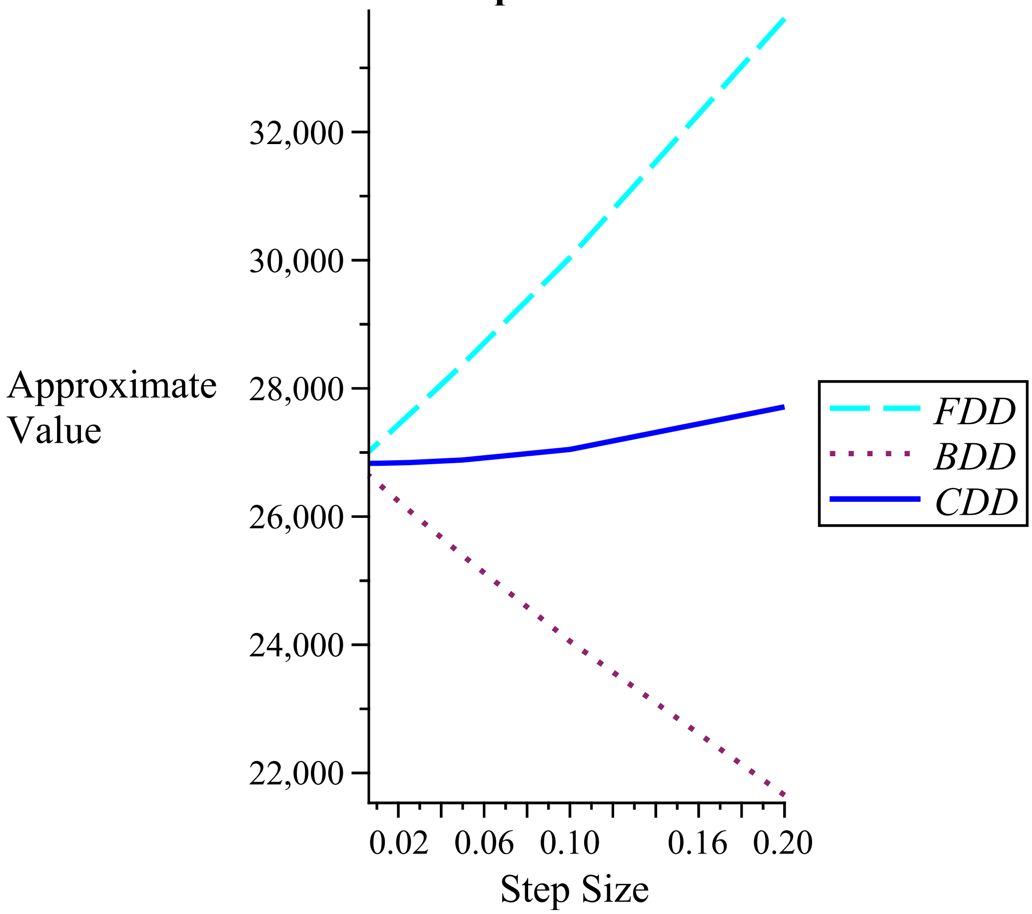
Section 5: Graphs

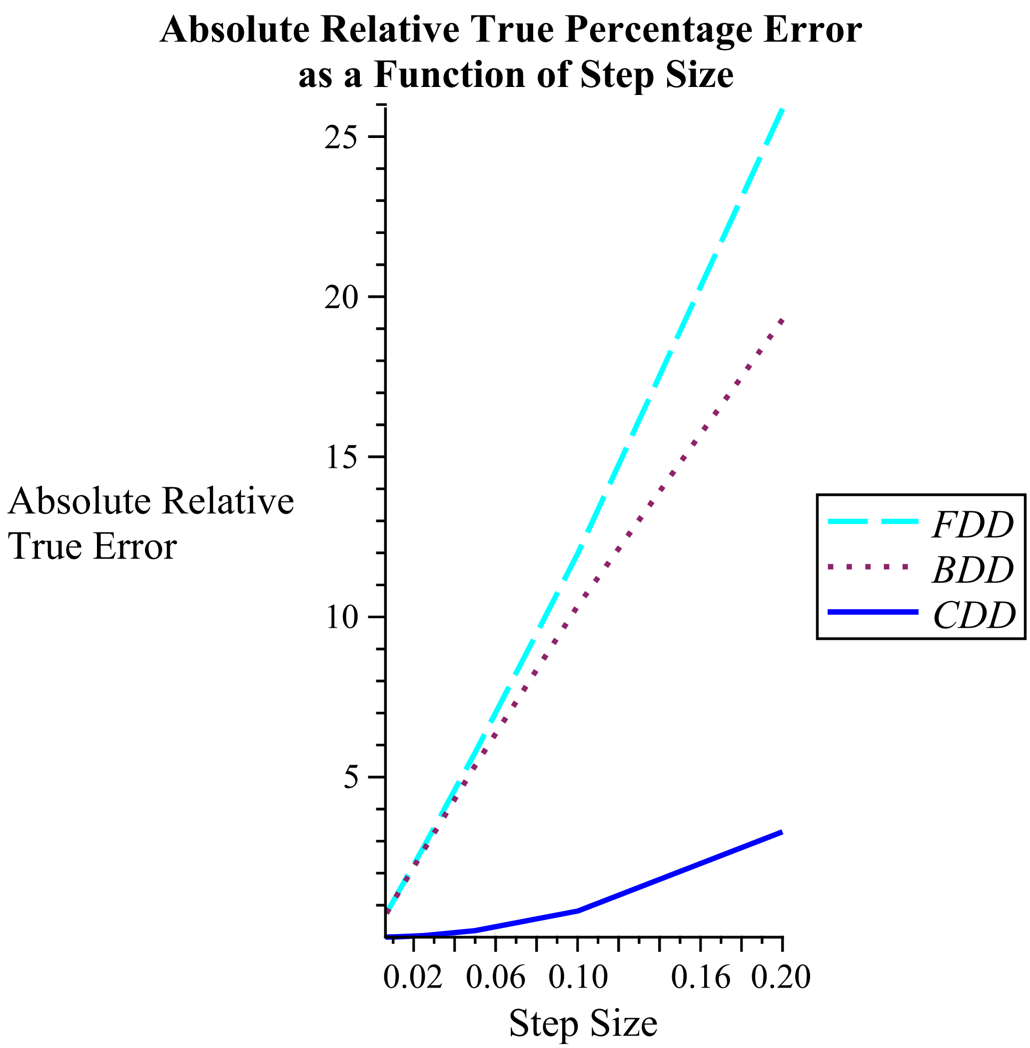
The following graphs show the approximate solution, absolute relative true error, absolute relative approximate error and least number of significant digits as a function of step size. Each graph displays the resaults of each of the methods of approximation.

```
> data := [seq([H[i], AvFDD[i]], i = 0 ..n-1)] :
  data1 := [seq([H[i], AvBDD[i]], i = 0 ..n-1)] :
  data2 := [seq([H[i], AvCDD[i]], i = 0 ..n-1)] :
  plot([data, data1, data2], x = H[0] ..H[n-1], color = [cyan,
    maroon, blue], thickness = 2, title
    = "Approximate Solution of the First Derivative of a
    Function using different Methods of Approximation as a
    Function of Step Size", linestyle = [3, 2, 1], legend = [FDD,
    BDD, CDD], legendstyle = [location = right], labels
    = ["Step Size", "Approximate
  Value"], titlefont = [TIMES, BOLD, 12], labelfont = [TIMES, ROMAN,
    12]);
  data := [seq([H[i], etFDD[i]], i = 0 ..n-1)] :
  data1 := [seq([H[i], etBDD[i]], i = 0 ..n-1)] :
  data2 := [seq([H[i], etCDD[i]], i = 0 ..n-1)] :
  plot([data, data1, data2], x = H[0] ..H[n-1], color = [cyan,
    maroon, blue], thickness = 2, title = "Absolute Relative True
    Percentage Error
  as a Function of Step Size", labels = ["Step Size", "Absolute
    Relative
  True Error"], linestyle = [3, 2, 1], legend = [FDD, BDD, CDD],
     legendstyle = [location = right], titlefont = [TIMES, BOLD,
    12], labelfont = [TIMES, ROMAN, 12]);
  data := [seq([H[i], eaFDD[i]], i = 0 ..n-1)] :
```

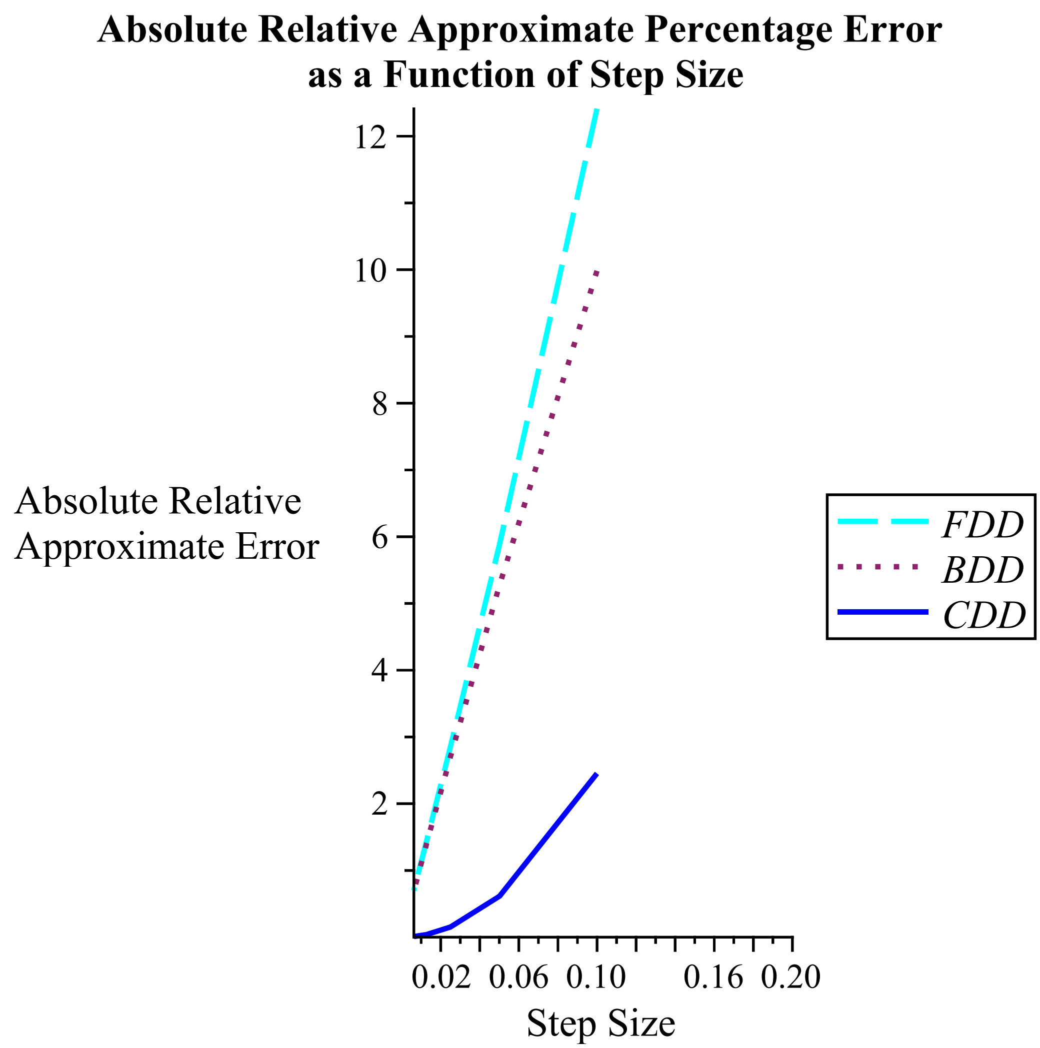
```
data1 := [seq([H[i], eaBDD[i]], i = 0 ..n − 1)] :
data2 := [seq([H[i], eaCDD[i]], i = 0 ..n − 1)] :
plot([data, data1, data2], x = H[0] ..H[n − 1], color = [cyan,
   maroon, blue], thickness = 2, title = "Absolute Relative
   Approximate Percentage Error
as a Function of Step Size", labels = ["Step Size", "Absolute
   Relative
Approximate Error "], linestyle = [3, 2, 1], legend = [FDD, BDD,
   CDD], legendstyle = [location = right], titlefont = [TIMES,
   BOLD, 12], labelfont = [TIMES, ROMAN, 12]);
data := [seq([H[i], SigFDD[i]], i = 0 ..n − 1)] :
data1 := [seq([H[i], SigBDD[i]], i = 0 ..n − 1)] :
data2 := [seq([H[i], SigCDD[i]], i = 0 ..n − 1)] :
plot([data, data1, data2], x = H[0] ..H[n − 1], color = [cyan,
   maroon, blue], thickness = 2, title = "Least Correct
   Significant Digits
as a Function of Step Size", labels = ["Step size", "Least
   number of
significant digits"], linestyle = [3, 2, 1], legend = [FDD, BDD,
   CDD], legendstyle = [location = right], titlefont = [TIMES,
   BOLD, 12], labelfont = [TIMES, ROMAN, 12]);
```

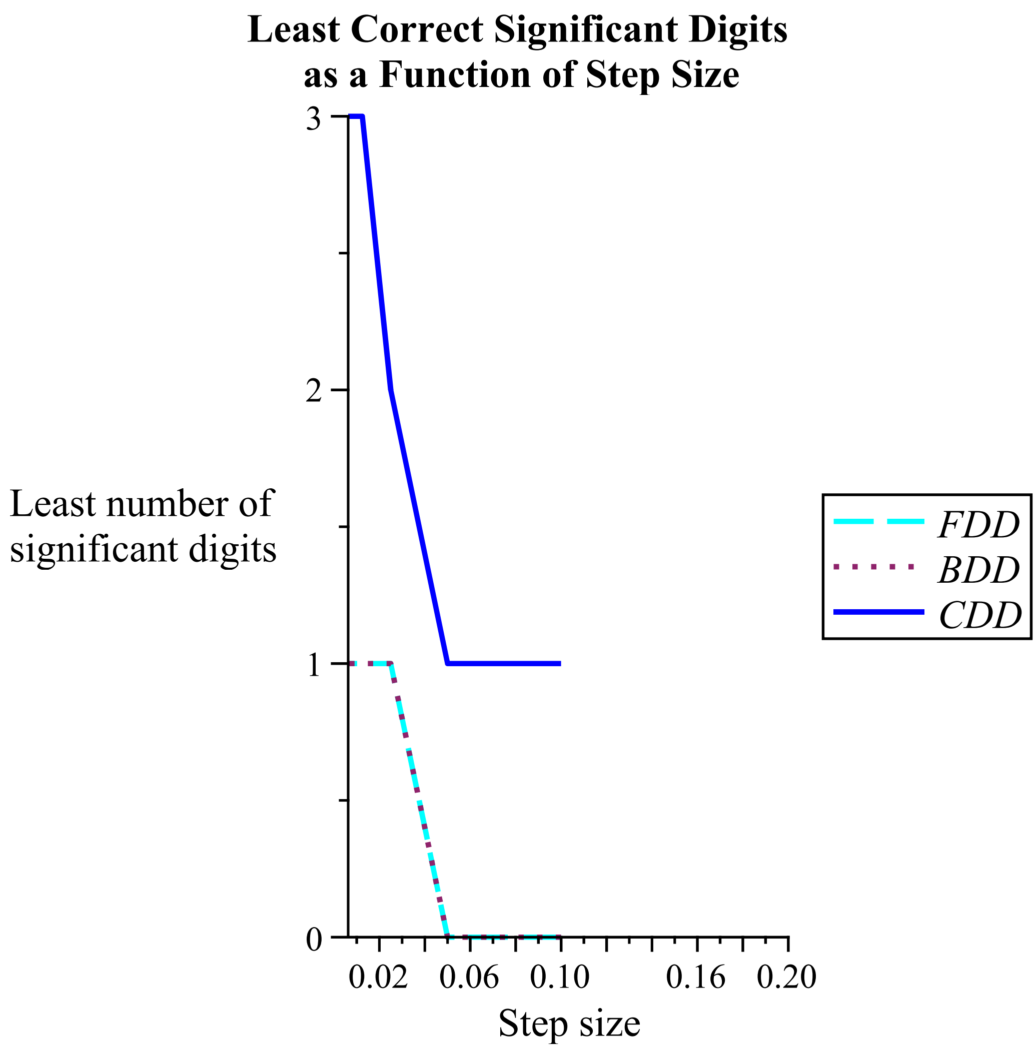
Approximate Solution of the First Derivative of a Function using different Methods of Approximation as a Function of Step Size





Absolute Relative True Percentage Error
as a Function of Step Size
Absolute Relative
True Error
25
20
15
10
5
0.02
0.06
0.10
0.16
0.20
Step Size
FDD
BDD
CDD


Absolute Relative Approximate Percentage Error
as a Function of Step Size
Absolute Relative
Approximate Error
12
10
8
6
4
2
0.02
0.06
0.10
0.16
0.20
Step Size
FDD
BDD
CDD



>

References

Numerical Differentiation of Continuous Functions.
See http://numericalmethods.eng.usf.edu/mws/gen/02def/

Questions

1 The velocity of a rocket is given by

$$v(t) = 2000 \cdot \ln \frac{140000}{140000 - 2100\,t} - 9.8\,t$$

Use three different methods with a step size of 0.25 to find the acceleration at *t=5s*. Compare with the exact answer and study the effect of the step size.

2 Look at the true error vs. step size data for problem 1. Do you see a relationship between the value of the true error and step size ? Is this concidential? Is it similar for Forward and Backward Divided Difference? Is it different for Central Divided Difference method?

Conclusions

The worksheet shows the nature of accuracy of the three different methods of finding the first derivative of a continuous function. Forward and Backward Divided Difference methods exhibit similar accuraciees as they are first order accurate, while central divided difference shows more accuracy as it is second order accurate.