

Report on Chapter 4—Numerical Differentiation and Integration

1 Introduction to Algorithms

Those methods are used to calculate differentiation or integration of a given function numerically, i.e. without using the analytic expressions of them. The basic idea is using the interpolation polynomials to approximate the functions.

2 Codes and Numerical Experiments

2.1 Numerical Differentiation

In this section we use 6 methods, two-point forward&backward, three-point endpoint&midpoint, five-point endpoint&midpoint, to calculate the numerical derivative of 8 functions at $x = 1$. We also compare these value with the exact value see how exact these methods are. We package these 8 functions as one function, the code is in Appendix A.1:

We also package 6 numerical differentiation method as a function, the code is in Appendix A.2:

Now we use the function above to calculate the result, the code is in Appendix A.3:

The value is listed in Table 1 and the error is listed in Table 2 (The 4 number in each square is got by taking the length of step $h = 0.1, 0.01, 0.2, 0.02$).

From Table 2, we see that the two-point methods are not so accurate as the other 4 methods. We also see that, for $n = 2, 3, 5$, points, more points we use, more exact the value will be. What's more, for $h = 0.1, 0.01, 0.2, 0.02$, the smaller h is, more exact the value will be.

Table 1: Value of Methods

value $f(x)$ method	$\frac{1}{1+x^2}$	$\sin(x)$	$\tan(x)$	$x \log(x)$	xe^x	$e^x \sin(x)$	$\sqrt{x^2+1}$	$\sinh(x)$
two-point	-0.497500124	0.536085981	3.479829956	1.196888873	5.477519671	3.770708495	0.708865742	1.548982408
forward	-0.475113122	0.497363753	4.073519326	1.229838986	5.863007979	3.899795842	0.723933124	1.604462765
	-0.49500098	0.531851857	3.536134378	1.200614054	5.518841539	3.785310881	0.710607224	1.554935913
	-0.450819672	0.452840506	5.073719487	1.265008259	6.329292394	4.035617274	0.739181864	1.671300809
two-point	-0.502499874	0.544500621	3.373098364	1.189388821	5.395970083	3.741335379	0.705330142	1.537230298
backward	-0.524861878	0.581440752	2.972495071	1.15478683	5.046390284	3.606819832	0.688511577	1.486844679
	-0.50499898	0.548680716	3.322472929	1.185613637	5.355735568	3.726569223	0.703535626	1.531431105
	-0.548780488	0.62057447	2.638845838	1.114589243	4.689245428	3.454249733	0.667943574	1.435476057
three-point	-0.499999267	0.540320105	3.423525535	1.193163692	5.436197802	3.756106109	0.707124261	1.543028903
endpoint	-0.499406572	0.541886999	3.073319165	1.194669713	5.396723564	3.76397441	0.708684383	1.537624721
	-0.499994274	0.540372653	3.417096566	1.193212618	5.435086417	3.756286118	0.707175914	1.542872511
	-0.496233939	0.545734149	-0.453751002	1.198743889	5.261089475	3.799133485	0.712735	1.519850848
three-point	-0.499999999	0.540293301	3.42646416	1.193138847	5.436744877	3.756021937	0.707097942	1.543106353
midpoint	-0.4999875	0.539402252	3.523007198	1.192312908	5.454699131	3.753307837	0.70622235	1.545653722
	-0.49999998	0.540266286	3.429303653	1.193113846	5.437288554	3.755940052	0.707071425	1.543183509
	-0.49980008	0.536707488	3.856282663	1.189798751	5.509268911	3.744933503	0.703562719	1.553388433
five-point	-0.500000029	0.540302305	3.425510512	1.193147183	5.436563624	3.756049258	0.707106783	1.543080632
endpoint	-0.500166225	0.540294454	2.513166713	1.193162817	5.436166764	3.756386017	0.707124373	1.543045352
	-0.500000432	0.540302289	3.425358001	1.193147214	5.436563114	3.75604972	0.707106805	1.543080584
	-0.501331734	0.540227493	-274.6072782	1.193328425	5.428810194	3.761864458	0.707365602	1.542427314
five-point	-0.500000005	0.540302306	3.425517662	1.193147181	5.436563651	3.756049232	0.707106781	1.543080634
midpoint	-0.500049974	0.540300507	3.411915377	1.193150961	5.436509205	3.756099282	0.707108894	1.543075485
	-0.50000008	0.540302303	3.425500199	1.193147187	5.43656357	3.756049307	0.707106785	1.543080627
	-0.500793323	0.540273627	3.010982339	1.193209146	5.435688268	3.756848819	0.707135845	1.542997944
exact value	-0.5	0.540302306	3.425518821	1.193147181	5.436563657	3.756049227	0.707106781	1.543080635

Table 2: Error of Methods

value $f(x)$ method	$\frac{1}{1+x^2}$	$\sin(x)$	$\tan(x)$	$x \log(x)$	xe^x	$e^x \sin(x)$	$\sqrt{x^2+1}$	$\sinh(x)$
two-point	0.002499876	-0.004216325	0.054311136	0.003741693	0.040956014	0.014659268	0.001758961	0.005901773
forward	0.024886878	-0.042938553	0.648000505	0.036691806	0.426444322	0.143746615	0.016826342	0.06138213
	0.00499902	-0.008450449	0.110615557	0.007466874	0.082277882	0.029261654	0.003500443	0.011855278
	0.049180328	-0.0874618	1.648200667	0.071861079	0.892728737	0.279568047	0.032075083	0.128220174
two-point	-0.002499874	0.004198315	-0.052420457	-0.003758359	-0.040593574	-0.014713848	-0.001776639	-0.005850337
backward	-0.024861878	0.041138446	-0.45302375	-0.038360351	-0.390173373	-0.149229395	-0.018595205	-0.056235955
	-0.00499898	0.00837841	-0.103045892	-0.007533543	-0.080828089	-0.029480004	-0.003571155	-0.01164953
	-0.048780488	0.080272164	-0.786672983	-0.078557937	-0.747318229	-0.301799494	-0.039163207	-0.107604578
three-point	7.32687E-07	1.77991E-05	-0.001993286	1.65117E-05	-0.000365855	5.68824E-05	1.74796E-05	-5.17316E-05
endpoint	0.000593428	0.001584693	-0.352199656	0.001522533	-0.039840093	0.007925183	0.001577602	-0.005455914
	5.726E-06	7.03474E-05	-0.008422255	6.54373E-05	-0.00147724	0.000236891	6.91324E-05	-0.000208123
	0.003766061	0.005431843	-3.879269823	0.005596709	-0.175474181	0.043084258	0.005628219	-0.023229787
three-point	1.24999E-09	-9.00499E-06	0.000945339	-8.33343E-06	0.00018122	-2.729E-05	-8.83889E-06	2.57181E-05
midpoint	1.24997E-05	-0.000900054	0.097488378	-0.000834272	0.018135475	-0.00274139	-0.000884431	0.002573087
	0.00000002	-3.60194E-05	0.003784833	-3.33348E-05	0.000724897	-0.000109175	-3.53562E-05	0.000102874
	0.00019992	-0.003594818	0.430763842	-0.003348429	0.072705254	-0.011115724	-0.003544062	0.010307798
five-point	-2.85025E-08	-1.05242E-09	-8.3088E-06	2.16477E-09	-3.32607E-08	3.04377E-08	1.41797E-09	-3.12573E-09
endpoint	-0.000166225	-7.852E-06	-0.912352107	1.56362E-05	-0.000396893	0.00033679	1.75922E-05	-3.52829E-05
	-4.32305E-07	-1.63816E-08	-0.00016082	3.33421E-08	-5.42644E-07	4.93168E-07	2.39402E-08	-5.0663E-08
	-0.001331734	-7.48131E-05	-278.0327971	0.000181245	-0.007753463	0.005815231	0.000258821	-0.000653321
five-point	-0.000000005	-1.80097E-10	-1.15851E-06	3.75025E-10	-5.43667E-09	5.008E-09	2.20886E-10	-5.14369E-10
midpoint	-4.99738E-05	-1.79886E-06	-0.013603444	3.77999E-06	-5.4452E-05	5.00545E-05	2.11256E-06	-5.14973E-06
	-0.00000008	-2.88147E-09	-1.86218E-05	6.00191E-09	-8.69906E-08	8.01274E-08	3.52935E-09	-8.23016E-09
	-0.000793323	-2.86792E-05	-0.414536482	6.1965E-05	-0.000875389	0.000799592	2.90634E-05	-8.26904E-05

2.2 Numerical Integration

In this section we use 4 method, composite Simpson's method, Romberg's method, composite G-L method with 2&3 nodes to calculate integral

$$I = \int_1^{100} x e^{-x^2}$$

And we compare the efficiency and error of these methods to see which is better.

We package these methods and integrand $f(x) = x e^{-x^2}$ as functions, the codes are in Appendix B.1 B.2 B.3 B.4.

Now we use these functions above to calculate the numerical integral and compare the value with the exact value got from Wolframalpha, the results are listed in Table 3 4 5 6 and the code is in Appendix B.5.

Table 3: Composite Simpson's Method

step \ Value or error	$\frac{100}{10}$	$\frac{100}{20}$	$\frac{100}{100}$	$\frac{100}{1000}$	$\frac{100}{10000}$
value	1.214002156	0.607001078	0.122393163	0.101470417	0.101476482
error	-1.112525673	-0.505524595	-0.020916681	6.06587E-06	5.89142E-10

Table 4: Romberg's Method

step \ Value or error	$\frac{100}{5}$	$\frac{100}{10}$	$\frac{100}{15}$	$\frac{100}{20}$
value	0.694194116	0.101606423	0.101476483	0.101476483
error	-0.592717633	-0.000129941	-1.38778E-17	2.498E-16

Table 5: Composite Gauss-Legendre Method with 2 Nodes

step \ Value or error	$\frac{100}{10}$	$\frac{100}{20}$	$\frac{100}{100}$	$\frac{100}{1000}$	$\frac{100}{10000}$
value	2.21439E-12	0.000965015	0.105283673	0.10147673	0.101476483
error	0.101476483	0.100511468	-0.003807191	-2.47303E-07	-2.45424E-11

Table 6: Composite Gauss-Legendre Method with 3 Nodes

Value or error \ step	$\frac{100}{10}$	$\frac{100}{20}$	$\frac{100}{100}$	$\frac{100}{1000}$	$\frac{100}{10000}$
value	0.000448374	0.048843459	0.10142398	0.101476482	0.101476483
error	0.101028109	0.052633024	5.2503E-05	1.28445E-10	1.52656E-16

From these table,clearly,Romberg's method is the best method among them.When the step $h = \frac{100}{15}$,the accuracy is of 10^{-17} ,while other methods never give such accuracy.And the composite G-L method,no matter with 2 or 3 nodes,behaves better than composite Simpson's method.Also,the composite G-L method with 3 nodes behaves better than the one with 2 nodes.

3 Discussions and Conclusions

3.1 Numerical Differentiation

We get from our experiments that when the number of points rising or the length of step being smaller, the numerical methods become more accurate. This meets the theoretical calculation. But the theoretical calculation also tells us that the number of points can't be too big and the length of step can't be too small because these algorithms are instable.

3.2 Numerical Integration

Romberg's method is the best method among them, and the composite G-L method, no matter with 2 or 3 nodes, behaves better than composite Simpson's method. Also, the composite G-L method with 3 nodes behaves better than the one with 2 nodes. These all meet our theoretical calculation.

A Codes for Numerical Differentiation

A.1

```
1  function f=f(x,n)
2  switch n
3      case 1
4          f=1./(1+x.^2);
5      case 2
6          f=sin(x);
7      case 3
8          f=tan(x);
9      case 4
10         f=x.*log(1+x);
11     case 5
12         f=x.*exp(x);
13     case 6
14         f=exp(x).*sin(x);
15     case 7
16         f=(x.^2+1).^0.5;
17     case 8
18         f=sinh(x);
19 end
20 end
```

A.2

```

1  function ndif=ndif(f,x,h,n,m)
2  switch n
3      case 1
4          switch m
5              case 1
6                  ndif=(f(x+h)-f(x))./h;%two points forward
7              case 2
8                  ndif=(f(x)-f(x-h))./h;%two points backward
9          end
10     case 2
11         switch m
12             case 1
13                 ndif=(-3.*f(x)+4.*f(x+h)-f(x+2.*h))./(2.*h);
14                 %three points endpoint
15             case 2
16                 ndif=(f(x+h)-f(x-h))./(2.*h);
17                 %three points midpoint
18         end
19     case 3
20         switch m
21             case 1
22                 ndif=(-25.*f(x)+48.*f(x+h)-36.*f(x+2.*h)+...
23     16.*f(x+3.*h)-3.*f(x+4.*h))./(12.*h);%five point endpoint

```



```

24         case 2

25         ndif=(f(x-2.*h)-8.*f(x-h)+8.*f(x+h)...

26 -f(x+2.*h))./(12.*h);%five points midpoint

27     end

28 end

29 end

```

A.3

```

1  clear

2  clc

3  h=[0.01,0.1,0.02,0.2];%

4  x=1;

5  a=zeros(8,25);

6  e=zeros(8,24);

7  for n=1:8

8      a(n,25)=df(x,n);%exact value

9      for k=1:3

10         for m=1:2

11             for t=1:4

12                 a(n,4*(2*(k-1)+m-1)+t)=ndif(@(x)f(x,n),x,h(t),k,m);

13                 %numerical value

14                 e(n,4*(2*(k-1)+m-1)+t)=a(n,4*(2*(k-1)+m-1)+t)-a(n,25);

15                 %error

16             end

```

```

17         end
18     end
19 end
20 xlswrite("C:\Users\rixinner\Desktop\...
21 \\Ch_4\Numerical differentiation\data1.xls",a', 'A1:H25')
22 xlswrite("C:\Users\rixinner\Desktop\...
23 \\Ch_4\Numerical differentiation\data2.xls",e', 'A1:H24')

```

B Codes for Numerical Integration

B.1

```

1 function f1=f1(x)
2 f1=x.*exp(-x.^3);
3 end

```

B.2

```

1 function CS=CS(f,a,b,n)%composite Simpson's method
2 %f is the function & a,b is the interval
3 %(b-a)/n is the step
4 h=(b-a)./n;
5 S0=f(a)+f(b);
6 S1=0;
7 S2=0;

```

```

8  for i=1:n-1
9      X=a+i.*h;
10     if mod(i,2)==0
11         S2=S2+f(X);
12     else
13         S1=S1+f(X);
14     end
15 end
16 CS=h.*(S0+2.*S2+4.*S1)./3;
17 end

```

B.3

```

1  function Rbg=Rbg(f,a,b,n)%Romberg's method
2  %f is the function & a,b is the interval
3  %(b-a)/n is the step
4  h=b-a;
5  R(1,1)=h./2.*(f(a)+f(b));
6  for i=2:n
7      R(2,1)=1/2.*(R(1,1)+h.*sum(f(a+((1:2^(i-2))-0.5).*h)));
8      for j=2:i
9          R(2,j)=(4^(j-1).*R(2,j-1)-R(1,j-1))./(4^(j-1)-1);
10     end
11     h=h/2;
12     for j=1:i

```

```

13         R(1,j)=R(2,j);
14     end
15 end
16 Rbg=R(1,n);
17 end

```

B.4

```

1  function CGL=CGL(f,a,b,n,m)%composite G-L method
2  %f is the function & a,b is the interval
3  %(b-a)/n is the step & m is number of node
4  switch m
5      case 2%2-node
6          h=(b-a)./n;
7          CGL=h./2.*sum(f(a+((1:n)-1/2).*h...
8      -h./(2*sqrt(3)))+f(a+((1:n)-1/2).*h+h./(2*sqrt(3))));
9      case 3%3-node
10         h=(b-a)./(2*n);
11         CGL=h./9.*sum(5.*f(a+(2.*(1:n)-1).*h-h.*sqrt(3/5))...
12     +5.*f(a+(2.*(1:n)-1).*h+h.*sqrt(3/5))+8.*f(a+(2.*(1:n)-1).*h));
13     end
14 end

```

B.5

```

1  clear

2  clc

3  ev=0.1014764825947195668201483098935834921538325145...

4      055243911271631087040604166222922998554307931949939979;

5      %exact value

6

7  %calculate the quadrature using these methods with different step

8  cs(1,1)=CS(@f1,1,100,10);

9  cs(1,2)=CS(@f1,1,100,20);

10 cs(1,3)=CS(@f1,1,100,100);

11 cs(1,4)=CS(@f1,1,100,1000);

12 cs(1,5)=CS(@f1,1,100,10000);

13

14 rbg(1,1)=Rbg(@f1,1,100,5);

15 rbg(1,2)=Rbg(@f1,1,100,10);

16 rbg(1,3)=Rbg(@f1,1,100,15);

17 rbg(1,4)=Rbg(@f1,1,100,20);

18

19 cgl2(1,1)=CGL(@f1,1,100,10,2);

20 cgl2(1,2)=CGL(@f1,1,100,20,2);

21 cgl2(1,3)=CGL(@f1,1,100,100,2);

22 cgl2(1,4)=CGL(@f1,1,100,1000,2);

23 cgl2(1,5)=CGL(@f1,1,100,10000,2);

24

25 cgl3(1,1)=CGL(@f1,1,100,10,3);

```

```

26  cgl3(1,2)=CGL(@f1,1,100,20,3);
27  cgl3(1,3)=CGL(@f1,1,100,100,3);
28  cgl3(1,4)=CGL(@f1,1,100,1000,3);
29  cgl3(1,5)=CGL(@f1,1,100,10000,3);
30
31  eocs=ev-cs;
32  eorbg=ev-rbg;
33  eocgl2=ev-cgl2;
34  eocgl3=ev-cgl3;
35
36  %get the results
37  xlswrite("C:\Users\rixinner\Desktop    \Ch_4...
38  \Numerical integration\cs.xls",[cs;eocs],'A1:E2')
39  xlswrite("C:\Users\rixinner\Desktop    \Ch_4...
40  \Numerical integration\rbg.xls",[rbg;eorbg],'A1:D2')
41  xlswrite("C:\Users\rixinner\Desktop    \Ch_4...
42  \Numerical integration\cgl2.xls",[cgl2;eocgl2],'A1:E2')
43  xlswrite("C:\Users\rixinner\Desktop    \Ch_4...
44  \Numerical integration\cgl3.xls",[cgl3;eocgl3],'A1:E2')

```

References

- [1] Richard L. Burden, J. Douglas Faires, Annette M. Burden, Numerical Analysis (Tenth Edition), Cengage Learning, 2014.