



**数值分析实验题**

**大作业**

姓 名：胡 宇 航

学 号：M201773466

班 级：土木1701

专 业：结构工程

指导老师：柴 振 华

土木工程与力学学院

2017.12

# 0. 说明

所选择的三个实验为：实验1.2、3.1和5.1，采用 Matlab2014b 作为计算软件。

# 1. 误差传播与算法稳定性（实验1.2）

## 1.1 Matlab code

1. function test1\_2
2. promps={'请选择递推关系式，选择E.1.6则输入1，选择E.1.7则输入2：'};
3. result=inputdlg(promps,'test1\_2',1,{'1'});
4. nb=str2num(**char**(result));
5. **if** nb~=1 && nb~=2
6. errordlg('输入错误，重新输入！');
7. **return**;
8. end
9. result=inputdlg({'请输入递推步数n:'},'test1\_2',1,{'10'});
10. steps=str2num(**char**(result));
11. **if** steps<1
12. errordlg('输入错误，重新输入！');
13. **return**;
14. end
15. result=inputdlg({'请输入有效数字位数:'},'test1\_2',1,{'5'});
16. sd=str2num(**char**(result));
17. format **long**
18. result=zeros(1,steps);
19. err=result;
20. func=result;
21. %用库函数quadl计算积分的近似值
22. **for** n=1:steps
23. fun=@(x)x.^n.\*exp(x-1);
24. func(n)=quadl(fun,0,1);
25. end
26. **if** nb==1
27. %E.1.6
28. digits(sd);
29. result(1)=subs(vpa(1/exp(1)));
30. **for** n=2:steps
31. result(n)=subs(vpa(1-n\*result(n-1)));
32. end
33. err=abs(result-func);
34. elseif nb==2
35. %E.1.7
36. digits(sd);
37. result(steps)=0;
38. **for** n=steps:-1:2
39. result(n-1)=subs(vpa((1-result(n))/n));
40. end
41. err=abs(result-func);
42. end
43. clf;
44. disp('递推值:');
45. fprintf('%e  \n',result);
46. disp('误差:')
47. fprintf('%e  \n',err);
48. plot([1:steps],result,'-');
49. grid on
50. hold on;
51. plot([1:steps],err,'r--');
52. xlabel('n');
53. ylabel('en-and err n--');
54. text(2,err(2),'\uparrow err(n)');
55. text(4,result(4),'\downarrow en');

## 1.2 Result analysis

表1.1 使用两种算法（E.1.6和E.1.7）的递推结果

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 有效  数字 | 5位 | | 6位 | | 7位 | |
| 算法 | E.1.6 | E.1.7 | E.1.6 | E.1.7 | E.1.6 | E.1.7 |
| 结果 | 3.678800e-01 | 3.678800e-01 | 3.678790e-01 | 3.678790e-01 | 3.678794e-01 | 3.678795e-01 |
| 2.642400e-01 | 2.642400e-01 | 2.642420e-01 | 2.642420e-01 | 2.642412e-01 | 2.642411e-01 |
| 2.072800e-01 | 2.072800e-01 | 2.072740e-01 | 2.072740e-01 | 2.072764e-01 | 2.072768e-01 |
| 1.708800e-01 | 1.708900e-01 | 1.709040e-01 | 1.709040e-01 | 1.708944e-01 | 1.708929e-01 |
| 1.456000e-01 | 1.455300e-01 | 1.454800e-01 | 1.454800e-01 | 1.455280e-01 | 1.455357e-01 |
| 1.264000e-01 | 1.267900e-01 | 1.271200e-01 | 1.271200e-01 | 1.268320e-01 | 1.267857e-01 |
| 1.152000e-01 | 1.125000e-01 | 1.101600e-01 | 1.101600e-01 | 1.121760e-01 | 1.125000e-01 |
| 7.840000e-02 | 1.000000e-01 | 1.187200e-01 | 1.187200e-01 | 1.025920e-01 | 1.000000e-01 |
| 2.944000e-01 | 1.000000e-01 | -6.848000e-02 | -6.848000e-02 | 7.667200e-02 | 1.000000e-01 |
| 1.944000e+00 | 0.000000e+00 | 1.684800e+00 | 1.684800e+00 | 2.332800e-01 | 0.000000e+00 |

（1）对比表1.1中不同有效数字位数下的递推结果可知，算法E.1.7有更好的精确性。

（2）误差的理论结果

算法E.1.6误差：



算法E.1.7误差：



（3）根据上式可知，随着步数n的增大，算法E.1.6误差逐渐增大，也就是说产生误差后其影响是扩张的；而算法E.1.7误差随着步数n的增大而减小，也就是说产生误差后其影响是衰减的。

（4）结合上述理论分析和计算实验，算法E.1.7的结果随步数n的增大变化更小，故其稳定性更好。

# 2. 函数逼近与曲线拟合（实验3.1）

## 2.1 Matlab code

1. function test3\_1
2. clear;clc;close all
3. x0=-1:0.5:2;
4. y0=[-4.447 -0.452 0.551 0.048 -0.447 0.549 4.552];
5. n=3; %3次多项式
6. alph=polyfit(x0,y0,n);
7. y=polyval(alph,x0);
8. r=(y0-y)\*(y0-y)';
9. x=-1:0.01:2;
10. y=polyval(alph,x);
11. plot(x,y,'r--');
12. xlabel('x');
13. ylabel('y');
14. hold on;
15. plot(x0,y0,'x');
16. grid on;
17. legend('Polynomial fitting','Discrete data')
18. savename = 'test3\_1.eps';
19. print('-depsc','-r300',savename)
20. disp(['Square error:',sprintf('%g',r)]);
21. disp(['parameter alph:',sprintf('%g\t',alph)])

## 2.2 Result analysis

表2.1离散数据

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | -1.0 | -0.5 | 0.0 | 0.5 | 1.0 | 1.5 | 2.0 |
|  | -4.447 | -0.452 | 0.551 | 0.048 | -0.447 | 0.549 | 4.552 |



图2.1离散数据拟合结果

离散数据的拟合函数的图形见图2.1。

拟合曲线参数 = {1.99911，-2.99767，-3.96825e-05，0.549119}，平方误差 = 2.17619e-05。

# 3. 函数逼近与曲线拟合（实验5.1）

## 3.1 Matlab code

3.1.1 Code for question (1)

1. function test5\_1\_1
2. clear;clc;close all;
3. clf;
4. alpha=[-2,-0.3,0.3,2];
5. **for** i = 1:4
6. a = alpha(i);
7. h=0.01;
8. x=0:h:1;
9. y=x;
10. N=length(x);
11. y(1)=1;
12. func=**inline**('1+(y-x).\*a');
13. **for** n=1:N-1
14. k1=func(a,x(n),y(n));
15. k2=func(a,x(n)+h/2,y(n)+k1\*h/2);
16. k3=func(a,x(n)+h/2,y(n)+k2\*h/2);
17. k4=func(a,x(n)+h,y(n)+k3\*h);
18. y(n+1)=y(n)+h\*(k1+2\*k2+2\*k3+k4)/6;
19. end
20. fprintf('%e  ',y);
21. y0=exp(alpha(i)\*x)+x;
22. %     Linespec1 =['r+','r+','r+','r+'];
23. plot(x,y0,'r+');
24. hold on;
25. plot(x,y,'b');
26. xlabel('x');
27. ylabel('y');
28. end
29. legend('exact solution, \alpha=-2', ...
30. 'approximate solution, \alpha=-2', ...
31. 'exact solution, \alpha=-0.3', ...
32. 'approximate solution, \alpha=-0.3', ...
33. 'exact solution, \alpha=0.3', ...
34. 'approximate solution, \alpha=0.3', ...
35. 'exact solution, \alpha=2', ...
36. 'approximate solution, \alpha=2', ...
37. 'Location','NorthWest')
38. set(gca,'Fontsize',10);
39. savename = 'test5\_1\_1.eps';
40. print('-depsc','-r300',savename)

3.1.1 Code for question (2)

1. function test5\_1\_2
2. clear;clc;close all;
3. clf;
4. alpha=-10;
5. % hn=[0.1,0.3];
6. **for** h=[0.1,0.3]
7. a = alpha;
8. x=0:h:1;
9. y=x;
10. N=length(x);
11. y(1)=1;
12. func=**inline**('1+(y-x).\*a');
13. **for** n=1:N-1
14. k1=func(a,x(n),y(n));
15. k2=func(a,x(n)+h/2,y(n)+k1\*h/2);
16. k3=func(a,x(n)+h/2,y(n)+k2\*h/2);
17. k4=func(a,x(n)+h,y(n)+k3\*h);
18. y(n+1)=y(n)+h\*(k1+2\*k2+2\*k3+k4)/6;
19. end
20. fprintf('%e  ',y);
21. y0=exp(a\*x)+x;
22. plot(x,y0,'r+');
23. hold on;
24. plot(x,y,'b');
25. xlabel('x');
26. ylabel('y');
27. end
28. legend('exact solution, h=0.1', ...
29. 'approximate solution, h=0.1', ...
30. 'exact solution, h=0.3', ...
31. 'approximate solution, h=0.3', ...
32. 'Location','NorthWest')
33. set(gca,'Fontsize',10);
34. savename = 'test5\_1\_2.eps';
35. print('-depsc','-r300',savename)

## 3.2 Result analysis



图3.1 四组α的计算结果



图3.2 两组h的计算结果

（1）程序代码见3.1.1，取*h*=0.01，=[-2, -0.3, 0.3, 2]。由图3.1可知，4组值的结果稳定，位于稳定域内。

（2）经典R-K法的稳定域为：，取=-10，取=0.1（位于稳定域内）与=0.3（位于稳定域外），结果图像见图3.2。取全域等距的10个点上的计算值，见表3.1。

表3.1 两组h的计算结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| xn | 解析解 | h=0.1 | | h=0.3 |
| 解 | Δ | 解 |
| 0.1 | 0.467879441 | 0.475000000 | 0.007120559 |  |
| 0.2 | 0.335335283 | 0.340625000 | 0.005289717 |  |
| 0.3 | 0.349787068 | 0.352734375 | 0.002947307 | 1.67500000000000 |
| 0.4 | 0.418315639 | 0.419775391 | 0.001459752 |  |
| 0.5 | 0.506737947 | 0.507415771 | 0.000677824 |  |
| 0.6 | 0.602478752 | 0.602780914 | 0.000302162 | 2.49062500000000 |
| 0.7 | 0.700911882 | 0.701042843 | 0.000130961 |  |
| 0.8 | 0.800335463 | 0.800391066 | 5.5603E-05 |  |
| 0.9 | 0.900123410 | 0.900146650 | 2.324E-05 | 3.49960937500000 |

由表中结果可知，当h=0.1时，位于稳定域内，与解析结吻合较好。而当h=0.3时，位于稳定域外，与解析解相差很大。