一、问题引入

（1）推导求解常微分方程初值问题 $\frac{dy}{dt}=f(t,y), y(a)=y_0, a\le t\le b$ 的 Heun 方法；

（2）编程实现用 Heun 方法求 I. V. P. $\boldsymbol{y'=3y+3t, y(0)=1, 0\le t\le 2}$ 的数值解，分别取 h=0.1 和 h=0.05;

（3）做出（2）中问题的数值解和精确解的图形.

二、问题求解

2.1.1 求 I.V.P 的 Heun 方法推导[1][2][3]

问题的关键是要找到一个数列 (t_k, y_k) ，用 y_k 来与 $y(t_k)$ 近似，为了求出 (t_k, y_k) ，我们先要求出 (t_1, y_1) ，我们已知：

$$y'(t)=f(t,y) \qquad t_0=a, \mathrm{y}(a)=y_0 \;\; a\le t\le b \tag{2.1.1.1}$$

利用微积分基本定理，在 $[t_0,t_1]$ 上对 $y'(t)$ 积分得：

$$\int_{t_0}^{t_1} f(t,y(t))dt=\int_{t_0}^{t_1} y'(t)dt=y(t_1)-y(t_0) \tag{2.1.1.2}$$

其中，$y'(t)$ 的不定积分为待求函数 $y(t)$，对 $y(t_1)$ 求解方程 2.1.1.2 ，结果为：

$$y(t_1)=y(t_0)+\int_{t_0}^{t_1} f(t,y(t))dt \tag{2.1.1.3}$$

然后，可以利用数值积分的方法逼近 2.1.1.3 中的定积分，如果采用步长为 $h=t_1-t_0$ 的梯形公式，则结果为：

$$y(t_1)\approx y(t_0)+\frac{h}{2}\left(f(t_0,y(t_0))+f(t_1,y(t_1))\right) \tag{2.1.1.4}$$

注意到公式 2.1.1.4 的右端包含了特定值 $y(t_1)$。如果要继续求解，需要 $y(t_1)$ 的一个估计值，而欧拉方法的解

$$y_1=y_0+hf(t_0,y_0) \tag{2.1.1.5}$$

恰好能够满足这一目的，于是，将 2.1.1.5 代入 2.1.1.4 后，得到求解 (t_1, y_1) 的公式：

$$y_1=y(t_0)+\frac{h}{2}\left(f(t_0,y_0)+f(t_1,y_0+hf(t_0,y_o))\right) \tag{2.1.1.6}$$

之后，我们对 2.1.1.2~ 2.1.1.6 的过程，便会得到 (t_k, y_k) ，在每一步中都使用欧拉方法作为估计值，然后使用梯形公式进行校正，得到最终的值。

因此，Heun 方法的一般步骤为：

$$p_{k+1} = y_k + hf(t_k, y_k), \qquad t_{k+1} = t_k + h$$
$$y_{k+1} = y_k + \frac{h}{2}\left(f(t_k, y_k) + f(t_{k+1}, p_{k+1})\right) \qquad 2.1.1.7$$

至此，求解常微分方程初值问题 $\frac{dy}{dt} = f(t,y), y(a) = y_0, a \le t \le b$ 的 Heun 方法即已得到推导。(见附录 程序四 heun.m)

2.1.2 误差分析

2.1.1 中已经说到 Heun 方法是使用 y_k 与 $y(t_k)$ 近似，既然是近似计算，那么就一定会存在误差。

I） 累计误差

从 2.1.1 的公式 2.1.1.3 到 2.1.1.4 的过程中，使用了梯形公式来逼近数值积分，则每次产生的误差项为：

$$-y^{(2)}(c_k)\frac{h^3}{12} \qquad 2.1.2.1$$

如果每步中的误差都是 2.1.2.1 ，则在 M 步后 Heun 方法的累计误差将是：

$$-\sum_{k=1}^{M} y^{(2)}(c_k)\frac{h^3}{12} \approx \frac{b-a}{12} y^{(2)}(c)h^2 = O(h^2) \qquad 2.1.2.2$$

II） 全局误差

设 $y(t)$ 为初值问题 （1） 的解，如果 $y(t) \in C^3[t_0, b]$ ，并且 $\{(t_k, y_k)\}_{k=0}^{M}$ 是 Heun 方法产生的一个近似值序列，则全局误差为：

$$|e_k| = |y(t_k) - y_k| = O(h^2) \qquad 2.1.2.3$$

其中 $k = 0,1,2\cdots,M$

III）局部误差

局部误差表示从 t_k 到 t_{k+1} 这一步的误差：

$$|\varepsilon_{k+1}| = |y(t_{k+1}) - y_k - h\Phi(t_k, y_k)| = O(h^3) \qquad 2.1.2.4$$

其中 $k = 0,1,2\cdots,M-1$ ， $\Phi(t_k, y_k) = y_k + \frac{h}{2}\left(f(t_k, y_k) + f(t_{k+1}, y_k + hf(t_k, y_k))\right)$

IV）最终全局误差

最终全局误差用来研究不同步长的误差特点，它可使我们了解要得到精确的近似所需的计算量。则区间终点处的最终全局误差满足：

$$E(y(b),h)=|y(b)-y_M|=O(h^2) \quad 2.1.2.5$$

如果，我们分别以 h 和 h/2 计算近似值，那么对步长 h 有：

$$E(y(b),h)\approx Ch^2 \quad 2.1.2.6$$

而对步长 h/2 有：

$$E\left(y(b),\frac{h}{2}\right)\approx C\frac{h^2}{4}=\frac{1}{4}Ch^2\approx\frac{1}{4}E(y(b),h) \quad 2.1.2.7$$

从 2.1.2.6 到 2.1.2.7 的过程中，步长减少了 $\frac{1}{2}$ 倍，那么，最终全局误差将至 $\frac{1}{4}$ ，这可以作为 Heun 方法步长对最终全局误差的灵敏度影响。

2.2 求解 I.V.P

2.2.1 求解精确解（见程序一、二 figure.nb 、result.m）

观察题目中给出的方程，发现它的形式满足 3.2.1.1 的形式，其中

$$P(t)=3,Q(t)=3t \quad 2.2.1.1$$

将 2.2.1.1 连同初始值 $y(0)=1$ 代入 3.2.1.4 中，便可得到题目（2）的精确解：

$$y(t)=\frac{4}{3}e^{3t}-t-\frac{1}{3} \quad 2.2.1.2$$

其中 $0\le t\le 2$

同样，我们利用 MATLAB 和 Mathematica 对题目所给的初值问题进行求解：

Figure.nb 给出的结果为： y[t]->1/3 (-1+4 e3t-3 t)

Result.m 给出的结果为： y=(4*exp(3*t))/3 - t - 1/3

2.2.2 编程求解数值解

在 MATLAB 上运行 apps.m (见附录 程序五 apps.m),(请保证附录中所有涉及 MATLAB 的 .m 文件在同一个工作目录下)，依次输入 0 2 1 0.1 ，便可求得在 h = 0.1 时的数值解，具体结果请参看 表一 1、3、4、5 列；

同样，在 MATLAB 上运行 apps.m (见附录 程序五 apps.m)，依次输入 0 2 1 0.05 ，便可求得在 h = 0.05 时的数值解，具体结果请参看 表一 1、6、7、8 列；

2.3 图形分析

（为了减少篇幅，以下图片都进行缩小处理）

图 1 为 h = 0.1 时的近似值的图像（见附录 程序六 image1.m）

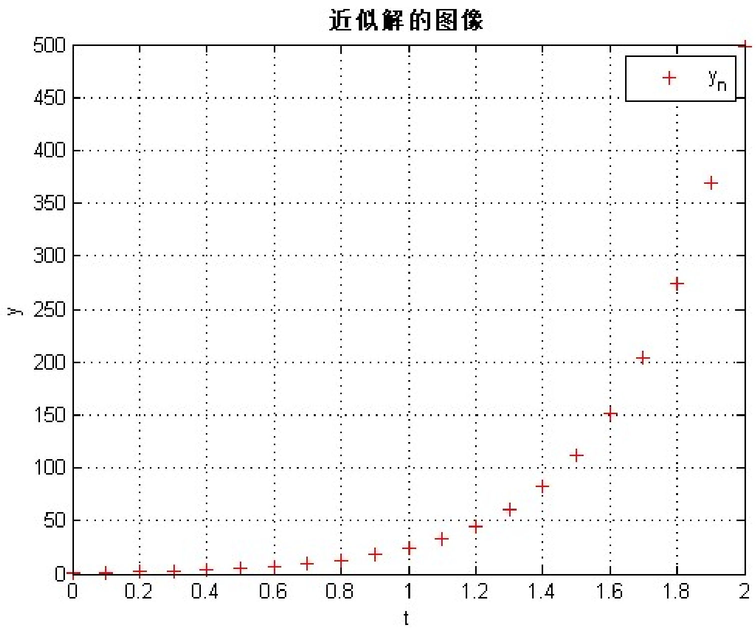


图 1

图 2 为精确解的图像(见附录 程序七 image2.m)(matlab 所作)

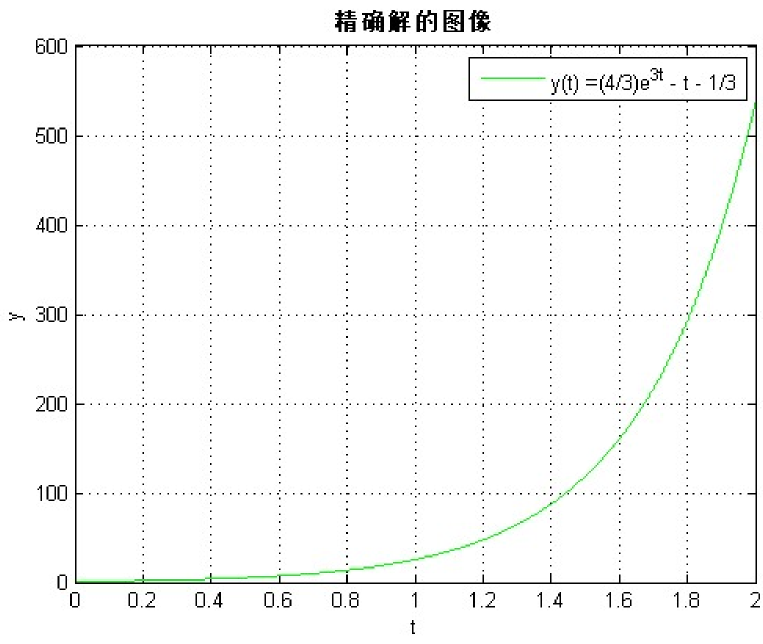


图 2

图 3 为精确解的图像(见附录 程序 1 figure.nb)(mathematica 所作)，mathematica 一般为精确计算，而 MATLAB 为近似计算，从下图我们也可以看出来，图 3 相对于 图 2 更为光滑（支持 SVG 向量图）。

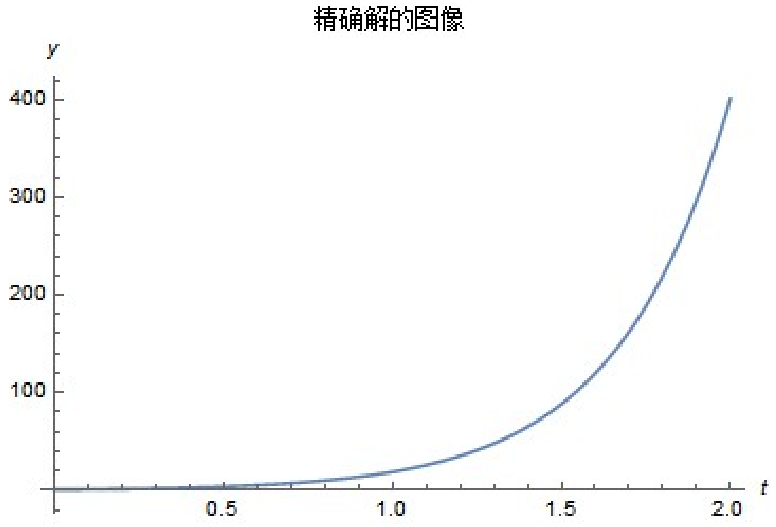


图 3

图 4 为 h =0.1 的近似解与精确解的对比图

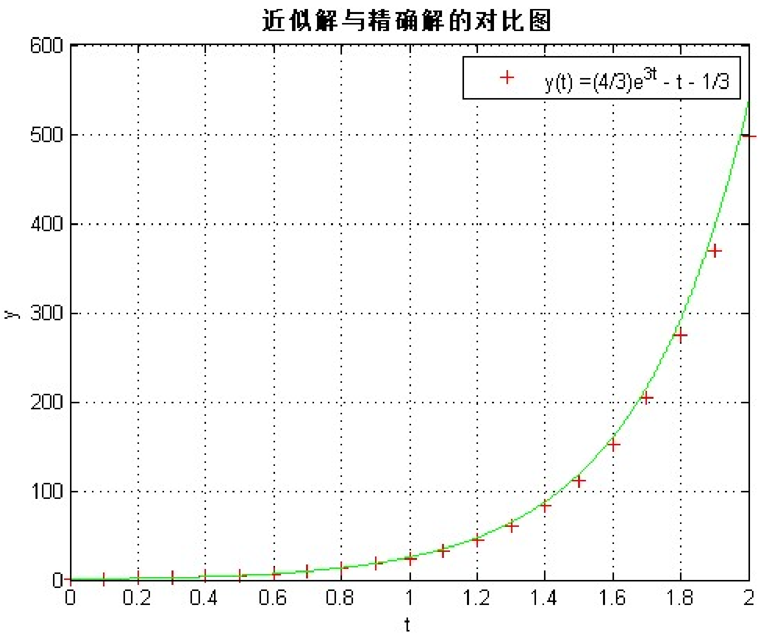


图 4

图 5 为 h = 0.05 时的近似值的图像

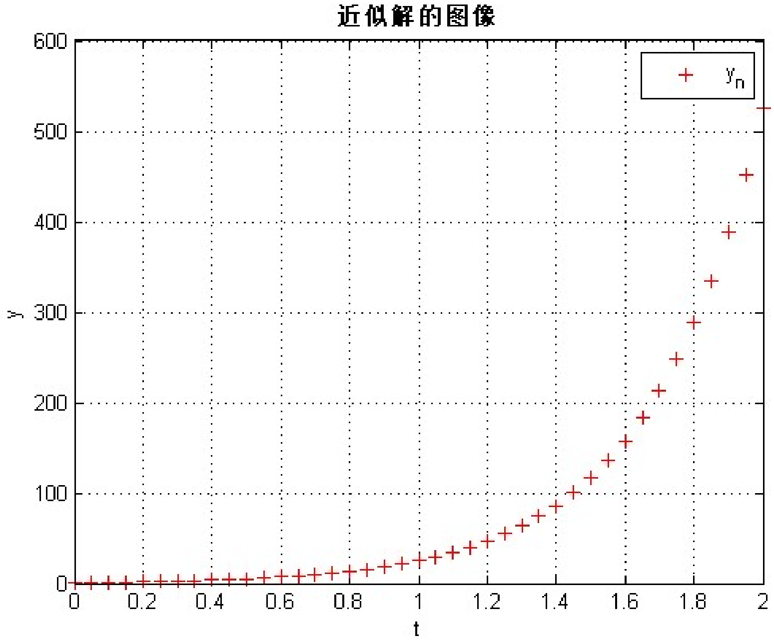


图 5

图 6 为 h =0.05 的近似解与精确解的对比图

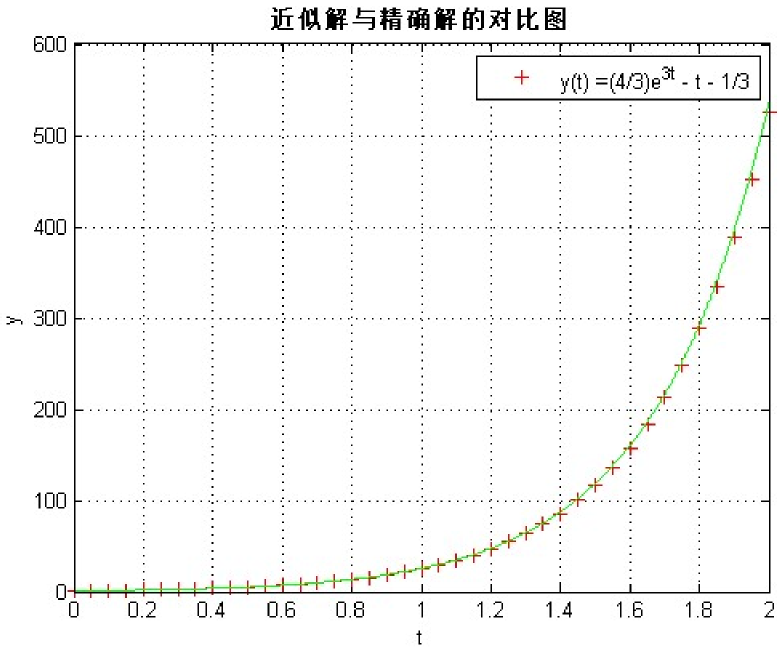


图 6

图 7 为 h = 0.1　h =0.05 的近似解与精确解的对比图

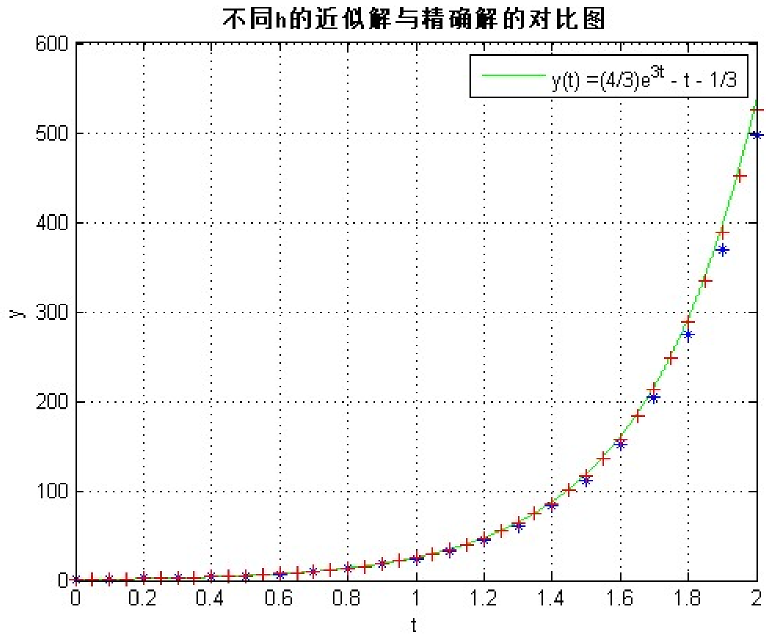


图 7

在图 4、6、7 中，我们会发现，近似解的图像要低于真实值的图像，那是因为，我们在做近似解时，总是假设上一步所得的结果为准确的，而且，我们舍去了真实解的无穷小量，因此会导致，近似解的图像在真实解的图像下方的情况。从图 7 中，我们可以轻易的发现，h = 0.05 的近似结果要优于 h = 0.1 的近似结果。

三、求一阶微分方程精确解[4]

3.1 变量分离

3.1.1 变量分离方程

形如

$$\frac{dy}{dx} = f(x)\varphi(y) \qquad 3.1.1.1$$

的方程称为 变量分离方程 ，其中 $f(x),\varphi(y)$ 分别为 x ,y 的连续函数，如果 $\varphi(y) \neq 0$，我们可以将 3.1.1.1 改写成：

$$\frac{dy}{\varphi(y)} = f(x)dx \qquad 3.1.1.2$$

对 3.1.1.2 两边积分，即可得到

$$y = \int \frac{dy}{\varphi(y)} = \int f(x)dx + c \qquad 3.1.1.3$$

另外，考虑到 $\varphi(y)=0$，即可得到 y 的精确解。

3.1.2 可化为变量分离方程的类型

形如以下几种形式的方程可以利用 变量替换 的方法化为 3.1.1 的变量分离方程，通过 3.1.1.3 式即可得到 y 的精确解。

<1> $$\frac{dy}{dx}=g\left(\frac{y}{x}\right) \qquad 3.1.2.1$$

<2> $$\frac{dy}{dx}=\frac{a_1x+b_1y+c_1}{a_2x+b_2y+c_2} \qquad 3.1.2.2$$

3.2 常数变易法

3.2.1 一般形式

形如

$$\frac{dy}{dx}=P(x)y+Q(x) \qquad 3.2.1.1$$

其中，$P(x),Q(x)$ 为所在区间上关于 x 的连续函数，假设 $Q(x)=0$，则利用 3.1.1.3 可以求得它的通解：

$$y=ce^{\int P(x)dx} \qquad 3.2.1.2$$

这里 c 为任意常数。

将常数 c 变易为 x 的待定函数 $c(x)$ ，令

$$y=c(x)e^{\int P(x)dx} \qquad 3.2.1.3$$

将 3.2.1.3 微分之后连同 3.2.1.3 一起代入 3.2.1.1 后与 3.2.1.1 比较即可得到 $c(x)$，将其代入 3.2.1.2 的 c 中即可得到方程的解。

综合以上的步骤后得到解 3.2.1.1 的一般公式：

$$y=e^{\int P(x)dx}\left(\int Q(x)e^{-\int P(x)dx}dx+\tilde{c}\right) \qquad 3.2.1.4$$

3.2.2 伯努利方程

形如

$$\frac{dy}{dx}=P(x)y+Q(x)y^n \qquad 3.2.2.1$$

的方程称之为伯努利微分方程，引入 $z=y^{1-n}$ 的变量代换，即可将 3.2.2.1 化为 3.2.1.1

的形式，利用 3.2.1.4 即可得方程的解。

除此之外，还有恰微分方程，构造积分因子等其他方法。

四、结果及误差分析

t_n	$y(t_n)$	h = 0.1			h = 0.05		
		y_n	全局误差	F.G.E	y_n	全局误差	F.G.E
0	1.0000	1.0000	0	37.4277	1.0000	0	10.7143
0.0500	1.1658				1.1650	0.0008	
0.1000	1.3665	1.3600	0.0065		1.3647	0.0018	
0.1500	1.6077				1.6046	0.0032	
0.2000	1.8962	1.8787	0.0175		1.8913	0.0049	
0.2500	2.2393				2.2322	0.0071	
0.3000	2.6461	2.6109	0.0353		2.6363	0.0099	
0.3500	3.1269				3.1135	0.0134	
0.4000	3.6935	3.6301	0.0634		3.6757	0.0178	
0.4500	4.3599				4.3367	0.0232	
0.5000	5.1423	5.0355	0.1068		5.1123	0.0300	
0.5500	6.0593				6.0310	0.0383	
0.6000	7.1329	6.9602	0.1726		7.0843	0.0485	
0.6500	8.3883				8.3272	0.0611	
0.7000	9.8549	9.5835	0.2714		9.7785	0.0764	
0.7500	11.5670				11.4719	0.0951	
0.8000	13.5642	13.1463	0.4179		13.4464	0.1178	
0.8500	15.8928				15.7474	0.1454	
0.9000	18.6063	17.9728	0.6335		18.4275	0.1788	
0.9500	21.7670				21.5478	0.2192	
1.0000	25.4474	24.4989	0.9485		25.1793	0.2681	
1.0500	29.7314				29.4045	0.3269	
1.1000	34.7169	33.3110	1.4059		34.3190	0.3978	

1.1500	40.5172				40.0341	0.4831	
1.2000	47.2643	45.1978	2.0665		46.6788	0.5855	
1.2500	55.1114				54.4030	0.7085	
1.3000	64.2366	61.2200	3.0166		63.3808	0.8558	
1.3500	74.8466				73.8143	1.0323	
1.4000	87.1818	82.8044	4.3774		85.9383	1.2435	
1.4500	101.5213				100.0253	1.4959	
1.5000	118.1895	111.8699	6.3196		116.3920	1.7975	
1.5500	137.5633				135.4058	2.1575	
1.6000	160.0806	150.9976	9.0830		157.4937	2.5868	
1.6500	186.2500				183.1513	3.0986	
1.7000	216.6625	203.6587	13.0038		212.9543	3.7083	
1.7500	252.0050				247.5710	4.4340	
1.8000	293.0752	274.5225	18.5527		287.7778	5.2974	
1.8500	340.8001				334.4760	6.3241	
1.9000	396.2565	369.8688	26.3878		388.7123	7.5442	
1.9500	460.6958				451.7023	8.9936	
2.0000	535.5717	498.1440	37.4277		524.8575	10.7143	

表一 数据及误差表

由以上的数据表，我们可以发现：当 h 减为原来的 $\frac{1}{2}$ 时，最终误差的精度将会大约提升为原来的 4 倍，这样论证了 2.1.2.7 式所得结论，当然，我们完全有理由相信：当 h = 0.025 时，最终误差可以 达到 2.7 左右，经过 apps.m 的程序计算，得出最终误差为 2.8530 ，因此，我们完全有理由按照这个思路推理下去，以获得我们想要的精度。

五、问题思考

Heun 方法其实是改进的欧拉方法，我们知道欧拉方法的全局误差精度为 $O(h)$，局部离散误差的精度为 $O(h^2)$ ，而 Heun 方法相对于欧拉方法提升了一个精度，其全

局误差为 $O(h^2)$ 局部离散误差为 $O(h^3)$，这也就是说， Heun 方法的误差变为了更高阶的无穷小量，那么，其所得的近似值也相应的提升了一阶精确度。

此外，在求 Heun 的递推公式的过程中，我们使用了步长为 $t_{k+1}-t_k(k\geq 0)$ 的梯形公式进行对积分的近似，在步长较大的情况下采用梯形公式处理上凸或下凸的函数都不是特别理想；同时，在处理时，我们还采用了欧拉方法来近似计算，因此，Heun 方法是在欧拉方法的基础上建立起来的。

六、总结与体会

经过这次的课程设计，使得我对欧拉方法以及 Heun 方法的原理和几何意义更加深刻，同时，也锻炼了我利用所学原理处理实际问题的能力，另一方面也锻炼了我对 MATLAB 的熟悉程度，使我加深了对 MATLAB 等科学计算工具的喜爱，培养了我的编程思维。在今后处理类似于求解 I.V.P 近似解的近似问题时，可以尝试考虑使用科学计算工具进行编程求解，这样既节省了时间，又锻炼了自己的编程能力。

学习就是一个不断思考，不断锻炼，不断实践的过程，只有通过类似于课程设计这样的课程研究或者活动，才能充分的提高知识素养与水平，同时，也能提高学习动手能力。

七、参考文献

[1]数值分析（第四版），周璐、陈瑜、钱方等译，电子工业出版社

[2]Numerical Methods Using MATLAB(Fourth Edition)，黄仿伦改编，电子工业出版社

[3]数值分析与科学计算，薛毅，科学出版社

[4]常微分方程（第三版），王高雄，周之铭，朱思铭等，高等教育出版社

[5]MATLAB programing for Engineers(Second Edition),Stephen J.Chapman,科学出版社

[6]Wolfram Documentation Center(Wolfram Mathematica 10.0)

八、附录[5][6]

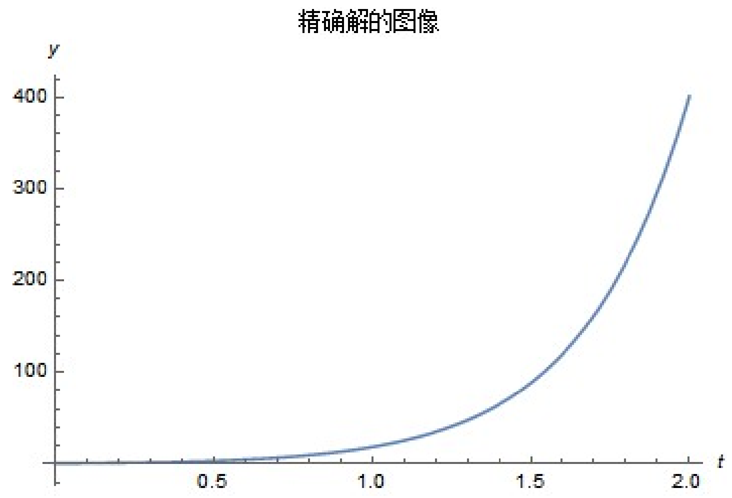
8.1 程序一 figure.nb

```
In[1] =  DSolve[{y'[t]==3*y[t]+3*t,y[0]==1},y[t],{t,0,2}]

Out[1]=  {{y[t]->1/3 (-1+4 e^3t-3 t)}}
```

In[2] = Plot[-1/3-t+(4/3)*Exp[3*t],{t,0,2}]

Out[2]=



8.2 程序二 result.m

```
%求解微分方程
y = dsolve('Dy=3*t+3*y','y(0)=1')
```

8.3 程序三 fun.m （函数）

```
% 定义将要求解的微分方程
function f = fun(t,y)
f = (3*t+3*y);
end
```

8.4 程序四 heun.m（函数）

```
% Heun 算法的程序
function H = heun(f,a,b,ya,M)
% 调用方法:
%         H = heun(f,a,b,ya,M)
%
% 输入:
%     -f 是 y'(即 y 的导数) ，其形式为 字符串
%     -a 和 b 是左右边界
%     -ya 是初始值
%     -M 是需要迭代的次数
% 输出:
%     -H = [T' Y']，T 和 Y 分别为 tn 和 yn 构成的迭代序列
%
% 定义迭代间隔
h = (b-a)/M;
% 构造 1*(M+1) 的全 0 矩阵
% 相当于开辟空间，用于存放 tn 和 yn
T = zeros(1,M+1);
Y = zeros(1,M+1);
```

```
% 使用 h 计算 tn
T = a:h:b;
% 赋初值
Y(1) = ya;
% 使用循环结构，计算 yn
for j = 1:M
    k1 = feval(f,T(j),Y(j));
    k2 = feval(f,T(j+1),Y(j)+h*k1);
    Y(j+1) = Y(j)+(h/2)*(k1+k2);
end
% 获得最终结果，并组合为新的 (M+1)*2 矩阵
H = [T' Y'];
end
```

8.5 程序五 apps.m

```
% 计算数值解的运行程序 apps.m
clear all;
% 数据的初始化
a  = input('Enter the left bound: ');
b  = input('Enter the right bound: ');
y0 = input('Enter the initial value: ');
h  = input('Enter the increment: ');
M  = (b-a)/h;
fprintf('\nThe approximate result is as bellow:\n');
% 求 tn  yn
H  = heun(@fun,a,b,y0,M);
% 真实值
t  = a:h:b;
y  = (4/3)*exp(3*t)-t-1/3;
% 给 H 增加一列，用于存储真实值
H(:,3) = y';
% 求全局误差
for k=1:(M+1)
    e(k) = abs(y(k)-H(k,2));
end
% 给 H 增加一列，用于存储全局误差
    H(:,4) = e';
% 输出 tn yn 真实值 全局误差
fprintf('\t\ttn\t\tyn\t\ty(tn)\t\terror\n');
% 输出 H
H
```

8.6 程序六 image1.m

```
% 绘制 数值解 的图像的程序 image1.m
% 图形绘制依赖于之前计算过的数据，在使用此程序前请勿 clear all
```

```
% 或者 clear H
plot(H(:,1),H(:,2),'r+');
% 标签设置
% LaTeX 转义，显示粗体
title('\bf 近似解的图像');
xlabel('t');
ylabel('y');
% 使用 LaTeX 转义，使得出现下标
legend('y_{n}')
grid on;
```

8.7 程序七 image2.m

```
% 绘制 精确解 的图像的程序 image2.m
% 经 常数变易法 计算得到精确解为
%       y(t) = (4/3)exp(3*t)-t-1/3
%
t = 0:.00001:2;
y = (4/3)*exp(3*t)-t-1/3;
plot(t,y,'g-');
% 标签设置
% LaTeX 转义，显示粗体
title('\bf 精确解的图像');
xlabel('t');
ylabel('y');
% 使用 LaTeX 转义
legend('y(t) =(4/3)e^{3t} - t - 1/3')
grid on;
```

8.8 程序八 image3.m

```
% 绘制数值解和精确解的复合图像的程序 image3.m
% 图形绘制依赖于之前计算过的数据，在使用此程序前请勿 clear all
% 或者 clear H
% image1 之前已经做过定义
image1;
hold on;
% image2 之前已经做过定义
image2;
% 重新定义标题文字
title('\bf 近似解与精确解的对比图');
hold off;
```

8.9 程序九 image4.m

```
% 绘制不同 h 下的数值解和精确解的复合图像的程序 image4.m
% 使用两次计算出的 tn 和 yn 作图
% 精确的图像
image2;
```

```
hold on;
clear all;
% h=0.1 的序列
H=heun(@fun,0,2,1,20);
T1=H(:,1)';
Y1=H(:,2)';
% h=0.1 的图像
plot(T1,Y1,'b*');
hold on;
clear all;
% h=0.05 的序列
H=heun(@fun,0,2,1,40);
T2=H(:,1)';
Y2=H(:,2)';
% h=0.05 的图像
plot(T2,Y2,'r+');
% 重新定义标题文字
title('\bf 不同 h 的近似解与精确解的对比图');
hold off;
fprintf('蓝色 * 表示 h=0.1\n 红色 + 表示 h=0.05\n');
```

课程设计评分表

评阅点	评分标准	分值	得分
选题（0-10）	与课程的关联度密切，选题适中	9-10	
	与课程相关，选题内容过大或偏小	5-8	
	与课程基本无关	0-4	
摘要（0-20）	基本写出论文大意且语言简练、文字组织合理	16-20	
	基本写出论文大意且语言简练	11-15	
	基本写出论文大意	6-10	
	套话、虚话较多或字数不够或文不对题	0-5	
正文（0-20）	论证严谨、思路清晰、逻辑性强、有较强说服力，引文准确	16-20	
	论证较严谨、思路较清晰、符合逻辑、有一定说服力，引文准确	11-15	
	思路较清晰、引文较恰当	6-10	
	有一定的说服力但论文紊乱、自相矛盾、大段抄袭他人章	0-5	
结构（0-20）	结构严谨、逻辑严密、层次清晰	16-20	
	结构合理、符合逻辑、层次分明	11-15	
	结构基本合理、层次比较清楚、文理通顺	6-10	
	有不合理部分，逻辑性不强	0-5	
深广度（0-20）	见解独特，对问题分析透彻，且非常全面	16-20	
	有自主的见解，对问题的分析比较深入全面	11-15	
	能提出自己的见解，分析的深度、广度一般	6-10	
	对问题的分析既无深度，又无广度	0-5	
规范性（0-10）	格式完全符合规范，字数符合要求 ，参考文献大于5篇，文献相关性密切	10	
	格式比较规范，字数偏少，参考文献3-5篇，文献相关性较好	8-10	
	格式基本符合规范，但有个别地方不合规，字数、参考文献偏少	4-7	
	格式规范性尚可，但不足之处较多，字数太少	0-3	
		总分	