目录

第一部分　创作中

第一部分

创作中

第一章

创作中

开普勒三定律

预备知识 椭圆[21]

开普勒定律描述了行星围绕固定中心天体的运动的规律．第一定律给出了行星运动轨道的几何形状，第二定律描述了行星在轨道不同位置的相对速度，第三定律给出周期与轨道长轴的关系．开普勒三定律最早是开普勒根据对太阳系中行星的观测数据总结而来的，但牛顿运用他的第二定律[??] 和万有引力定律[??]，将开普勒三定律精确地推导了出来．

第一定律：每一个行星都沿各自的椭圆轨道环绕中心天体，中心天体则处在椭圆的一个焦点上．

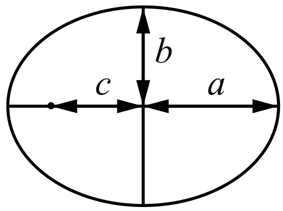

图 1: 开普勒第一定律

第二定律：相等时间内，中心天体与行星的连线所扫过的面积是相等的．

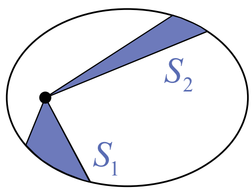

图 2: 开普勒第二定律

第三定律：椭圆轨道半长轴（a）的立方与周期（公转一圈所用的时间 T）的平方成正比．

数值模拟及证明

以下的推导中，需要忽略行星与恒星的大小，行星之间的相互引力以及中心天体的运动．在进行解析推导之前，不妨先看看更为直观的数值模拟．

- 天体运动的简单数值计算[43]
- 开普勒第一定律的证明[16]
- 开普勒第二定律的证明[19]
- 开普勒第三定律的证明[20]

常微分方程（组）的数值解

预备知识 弹簧振子受迫运动的简单数值计算[41]，天体运动的简单数值计算[43]

数值解微分方程或微分方程组时，一般需要先把微分方程（组）化成一阶微分方程组．例如在“弹簧振子受迫运动的简单数值计算[41]”中列出的微分方程为

$$my'' = \alpha y' - ky + f(t) \tag{1}$$

新增变量，令 $v = y'$，则可变为一阶微分方程组

$$\begin{cases} v' = [-\alpha u - ky + f(t)]/m \\ y' = v \end{cases} \tag{2}$$

方程组中，t 是自变量，y 和 u 是因变量．给出某时刻的 $y(t), u(t), t$ 就可以通过方程组求出 $u'(t)$ 和 $y'(t)$ 的数值．

又例如在“天体运动的简单数值计算[43]”中，列出的二阶微分方程组为

$$\begin{cases} x'' = -\dfrac{GM}{(x^2+y^2)^{3/2}}x \\ y'' = -\dfrac{GM}{(x^2+y^2)^{3/2}}y \end{cases} \tag{3}$$

新增变量，令 $v_x = x'$，$v_y = y'$，上式也可变为一阶微分方程组

$$\begin{cases} x' = v_x \\ v_x' = -GMx/(x^2+y^2)^{3/2} \\ y' = v_y \\ v_y' = -GMy/(x^2+y^2)^{3/2} \end{cases} \tag{4}$$

该式中同样 t 是自变量，其他都是因变量，给出某时刻的 x, y, v_x, v_y, t 就可以由该方程组求出因变量的一阶导数．

在以上两个词条中，我们使用了一种较为原始的方法（微分近似）．这种方法相当于把某时刻 t 的各因变量代入一阶微分方程组，得到 t 时刻各因变量

的一阶导数，再通过微分近似由这些一阶导数来计算其 $[t, t+\Delta t]$ 时间内的增量，得到 $t+\Delta t$ 时刻的各因变量，再代入一阶方程组得到 $t+\Delta t$ 时刻各因变量的一阶导数，如此一直循环，得到各因变量每隔 Δt 时间的值．这种方法叫做**欧拉法**．对一阶常微分方程 $y'(t) = f(y, t)$，令 $h = \Delta t$，欧拉法可以表示为

$$y_{n+1} = y_n + hf(y_n, t_n) \tag{5}$$

以下介绍三种更精确的算法，在步长（Δt）相同时它们的精确度递增．注意虽然大多数问题中自变量是时间，但这些方法适用于大多数单个自变量的微分方程（组）．

- 中点法[7]
- 龙格库塔法
- Matlab 中的 Ode45 算法

中点法解常微分方程（组）

预备知识 常微分方程（组）的数值解[5]

我们先来尝试用欧拉法解一阶微分方程

$$y'(t) = y \tag{1}$$

令初始条件为 $y(0) = 1$．令步长为 $h = 0.1$，步数为 5，结果如图 1 所示（代码见词条最后）.

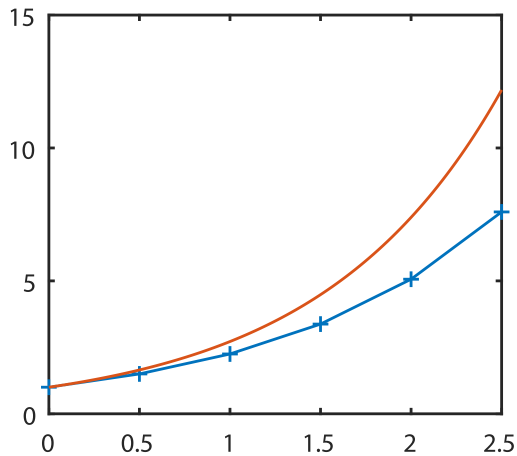

图 1: 欧拉法数值解（蓝）和解析解（红）

由 “” 我们知道该方程的解析解为 $y = \mathrm{e}^t$．对比数值解和解析解，不难分析出误差产生的原因：我们仅用每段步长区间左端的导数预测整个区间的曲线增量．如果我们能利用每个区间的中点的导数计算计算整个区间的增量，这个预测将会比欧拉法更精确.

考虑微分方程 $y'(t) = f(y, t)$ 在区间 $[t_n, t_{n+1}]$ 的曲线，若我们已知区间左端的函数值为 y_n，我们可以先用微分近似估计曲线中点的函数值为

$$y\left(t_n + \frac{h}{2}\right) = y_n + \frac{h}{2} f(y_n, t_n) \tag{2}$$

然后再求出这个近似中点的导数为

$$y'\left(t_n + \frac{h}{2}\right) = f\left[y_n + \frac{h}{2} f(t_n, y_n), t_n + \frac{h}{2}\right] \tag{3}$$

最后我们利用这个导数估算该区间的曲线增量

$$y_{n+1} = y_n + hy'\left(t_n + \frac{h}{2}\right) = y_n + hf\left[y_n + \frac{h}{2}f(t_n, y_n), t_n + \frac{h}{2}\right] \tag{4}$$

这就是解常微分方程的**中点法**.

我们再来用中点法取同样的步长计算式 1，结果如图 2 所示.

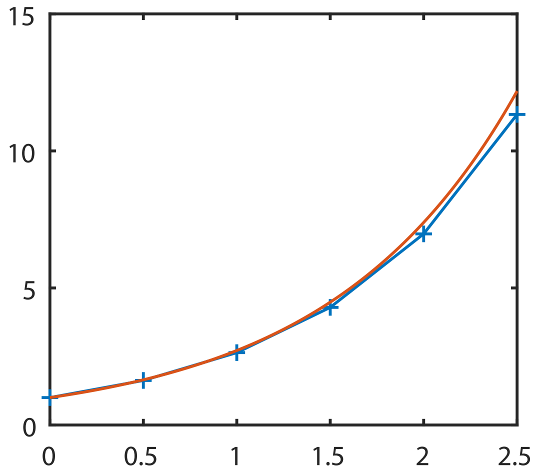

图 2: 欧拉法数值解（蓝）和解析解（红）

可见虽然中点法每一步的计算过程比欧拉法稍微复杂一些，但精度却大大地提高了.

中点法同样适用于微分方程组，例如对于常微分方程组

$$\begin{cases} x'(t) = f(x, y, t) \\ y'(t) = g(x, y, t) \end{cases} \tag{5}$$

首先计算近似中点为

$$\begin{cases} x_{n+1/2} = x_n + hf(x_n, y_n, t_n) \\ y_{n+1/2} = y_n + hg(x_n, y_n, t_n) \end{cases} \tag{6}$$

然后有

$$\begin{cases} x_{n+1} = x_n + hf\left(x_{n+1/2}, y_{n+1/2}, \dfrac{h}{2}\right) \\ y_{n+1} = y_n + hg\left(x_{n+1/2}, y_{n+1/2}, \dfrac{h}{2}\right) \end{cases} \tag{7}$$

例程

```
% 设置参数
N = 6;
h = 0.5;
t = linspace(0, (N-1)*h, N); % 自变量
t0 = linspace(0, (N-1)*h, 100); % 用于画图

% 欧拉法
y = zeros(1,N); % 预赋值
y(1) = 1; % 初值
for ii = 1:N-1
    y(ii+1) = y(ii) + h*y(ii);
end

% 画图
figure;
plot(x,y,'+-');
hold on;
plot(t0, exp(t0));

%中点法
y = zeros(1,N);
y(1) = 1;
for ii = 1:N-1
    y(ii+1) = y(ii) + h*(y(ii) + 0.5*h*y(ii));
end

% 画图
figure;
plot(x,y,'+-');
```

```
hold on;
plot(x0, exp(x0));
```

第二章

修改审阅中

计算物理导航

在当今时代，数值计算已经成为理科生不可或缺的技能．利用计算机程序计算理论模型，处理实验数据，绘制各种图表都是学习和研究中都扮演非常重要的角色．虽然一些简单的计算用计算器就可以完成，但许多时候我们需要数以万计的计算量，用计算器就显得不实际了．

计算机语言

计算机的指令等信息在其数字电路中是以二进制 0 和 1（对应低电压和高电压）传播的，然而这样的语言显然不适合人类阅读和编写．为了让计算机按照指定的规则进行运算，我们可以用某种计算机语言（由英文单词，符号和数字组成）写下代码，计算机会把这种代码转换为二进制指令进行相应的运算．

科学计算中，目前比较流行的高级语言有 Matlab，Python，Mathematica 等[1]．这些语言除了可以进行传统的数值计算外，还有符号计算的功能，例如可以对解析表达式进行不定积分和求导等运算．一般来说，Mathematica 主要用于符号计算，Matlab 和 Python 主要用于数值计算．

本章的数值计算以讲解算法（Algorithm）为主，例程使用 Matlab 代码，且尽量使用简单的语法，这样即使读者使用的编程语言不是 Matlab，也能轻易地改写出相应的代码．本章使用的所有 Matlab 语法都会在词条“Matlab 语法基础”中介绍，不要求读者有任何编程基．Matlab 软件的使用见

本章的符号计算使用 Mathematica 软件和 Wolfram Alpha 网站[2]．见“Mathematica 基础”．符号计算主要用于计算解析问题（如复杂的不定积分）或者获得高精度的数值结果（例如计算 π 的一万位）．其使用频率远没有数值计算多．

[1] 高级语言可以理解为自动化程度较高的语言，优点是编程和调试等较简单，易学易用，缺点是效率较低．比上述语言更低级的语言有 Fortran，C++，C 等，更适合做高性能数值计算．一些大计算量的物理问题可能运行一次需要数小时乃至数星期的时间，这时运行效率是首要考虑因素，而编程和调试的难易反而相对次要．

[2] Wolfram Alpha 相当于 Matheematica 的简单网页版，优势在于无需安装软件，用有网络浏览器的设备（包括手机和平板）可随时访问．网址为“https://www.wolframalpha.com/”．

微分方程数值解

微分方程在物理中无处不在，力学中最基本的牛顿第二定律就是一个常微分方程．虽然我们都希望能得到方程的精确解析解，然而往往只有最简单的一些问题才存在．对于没有精确解析解的问题，我们要么加入一些近似条件得到近似的解析解，要么求助于数值计算，也可以是二者结合[3]．本章中我们会学习如何计算常微分方程（组）的数值解．先用一阶微分近似的方法介绍一般的思路，再介绍更精确的高斯法和四阶龙格—库塔法

[3]对于一些复杂的问题（尤其在量子力学中）有时即使用数值计算也不能在合理的时间内解出微分方程，这时就要先使用近似化简问题．

开普勒问题

预备知识 椭圆的三种定义[21]，双曲线的三种定义 [23]，机械能守恒（单个质点）[??]，角动量守恒（单个质点）[??]，万有引力[??]，开普勒三定律[3]

结论

若已知中心天体和行星的质量，假设中心天体不动，能量和角动量可以唯一地确定行星轨道．当能量小于零时轨道为椭圆（$e<1$）

$$E=-\frac{GMm}{2a}\qquad L=bm\sqrt{\frac{GM}{a}}\tag{1}$$

能量等于零时轨道为抛物线（$e=1$），焦距与角动量的关系为（p 为焦距）

$$L=m\sqrt{2GMp}\tag{2}$$

能量大于零时轨道为双曲线（$e>1$）．

$$E=\frac{GMm}{2a}\qquad L=b\sqrt{2E/m}\tag{3}$$

根据圆锥曲线的性质，对椭圆有 $a^2=b^2+c^2$，双曲线有 $c^2=a^2+b^2$ 现在我们假设已知轨道是椭圆，抛物线和双曲线的一种（见开普勒第一定律的证明[16]），证明以上关系

椭圆轨道

令椭圆轨道距离焦点的最近和最远距离分别为 r_1 和 r_2，总能量（动能加势能）守恒

$$\frac{1}{2}mv_1^2-\frac{GMm}{r_1}=\frac{1}{2}mv_2^2-\frac{GMm}{r_2}\tag{4}$$

总角动量守恒

$$mv_1r_1=mv_2r_2\tag{5}$$

把式 5 中的 v_2 代入式 4，可得

$$v_1^2=\frac{2GM}{r_1+r_2}\frac{r_2}{r_1}\tag{6}$$

再代入式 4 的左边，并使用 $r_1+r_2=2a$ 得到总能量 $E=-GMm/2a$. 把式 6 代入式 5 的左边，并使用 $r_1r_2=(a+c)(a-c)=b^2$ 得总角动量 $L=bm\sqrt{GM/a}$.

双曲线

注意双曲线轨道只可能是双曲线离中心天体所在焦点较近的一支，另一支这里没有物理意义（当引力变为斥力，只能取双曲线的另一支）

总能量守恒

$$\frac{1}{2}mv_0^2=\frac{1}{2}mv_1^2-\frac{GMm}{r_1} \tag{7}$$

总角动量守恒

$$mv_0b=mv_1r_1 \tag{8}$$

抛物线

抛物线轨道离焦点的最近距离为焦距 p，

开普勒第一定律的证明

预备知识 圆锥曲线的极坐标方程[??]，极坐标加速度[??]，牛顿第二定律[??]，二阶常系数非齐次微分方程的通解[??]

数学模型

由于太阳质量远大于其他行星，近似认为太阳不动．由于太阳和行星相对于行星轨道来说大小可以忽略，把它们当做质点（另见球体的平方反比力）．以太阳为原点建立平面极坐标系，行星在该平面上运动，且仅受万有引力一个外力．现证明行星的运动轨迹是椭圆，且焦点在原点．

结论

行星轨道是以中心天体为焦点的任意圆锥曲线[4]．极坐标中，圆锥曲线的方程[??] 为

$$r = \frac{p}{1 - e \cdot \cos\theta} \tag{1}$$

令太阳中心天体在坐标原点，则行星沿该轨道运行．

证明

极坐标中径向和角向加速度公式分别为

$$a_r = \frac{\mathrm{d}^2 r}{\mathrm{d}t^2} - r\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)^2 \tag{2}$$

$$a_\theta = \frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}t}\left(r^2\frac{\mathrm{d}\theta}{\mathrm{d}t}\right) \tag{3}$$

根据牛顿第二定律和万有引力定律，由于行星只受到沿径向的万有引力，则有

$$ma_r = -G\frac{Mm}{r^2} \tag{4}$$

[4]所以行星轨道不一定是椭圆，也可以是抛物线或者双曲线，但是抛物线或双曲线轨道是从无穷远来到无穷远去的轨道，不会绕太阳旋转. 所以开普勒定律作为行星运动的经验公式，只描述了椭圆.

$$ma_\theta = 0 \tag{5}$$

在式 4 和式 5 中同除 m，代入式 2 和式 3 得

$$\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} - r\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)^2 = -\frac{GM}{r^2} \tag{6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(r^2\frac{\mathrm{d}\theta}{\mathrm{d}t}\right) = 0 \tag{7}$$

现在用式 6，式 7 消去 t．式 7 括号内部不随时间变化，令其等于常数 h

$$r^2\frac{\mathrm{d}\theta}{\mathrm{d}t} = h \tag{8}$$

其中 h 为任意常数．我们想得到 r 关于 θ 的微分方程，就要把所有的时间导数消去．首先可以把 r 看做复合函数 $r[\theta(t)]$，再用链式法则[??] 把式 6 的第一项用 $\mathrm{d}\theta/\,\mathrm{d}t$ 表示

$$\begin{aligned}\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\mathrm{d}r}{\mathrm{d}t}\right) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\mathrm{d}r}{\mathrm{d}\theta}\frac{\mathrm{d}\theta}{\mathrm{d}t}\right) = \frac{\mathrm{d}}{\mathrm{d}\theta}\left(\frac{\mathrm{d}r}{\mathrm{d}\theta}\right)\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)^2 + \frac{\mathrm{d}r}{\mathrm{d}\theta}\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2}\\ &= \frac{\mathrm{d}^2 r}{\mathrm{d}\theta^2}\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)^2 + \frac{\mathrm{d}r}{\mathrm{d}\theta}\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\frac{\mathrm{d}\theta}{\mathrm{d}t}\right)\frac{\mathrm{d}\theta}{\mathrm{d}t}\end{aligned} \tag{9}$$

然后把式 8 两边同除 r^2 并代入式 6 消去所有时间导数，得到 r 关于 θ 的微分方程

$$\frac{\mathrm{d}^2 r}{\mathrm{d}\theta^2}\left(\frac{h}{r^2}\right)^2 + \frac{\mathrm{d}r}{\mathrm{d}\theta}\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\frac{h}{r^2}\right)\frac{h}{r^2} - r\left(\frac{h}{r^2}\right)^2 = -\frac{GM}{r^2} \tag{10}$$

化简得

$$\frac{h^2}{r^4}\left[\frac{\mathrm{d}^2 r}{\mathrm{d}\theta^2} + r^2\frac{\mathrm{d}r}{\mathrm{d}\theta}\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\frac{1}{r^2}\right) - r\right] = -\frac{GM}{r^2} \tag{11}$$

这就是 r 关于 θ 的微分方程

将式 1 代入，可验证式 1 是该方程的解．当然，在事先不知道轨道方程的情况下，也可以直接解该方程．令

$$u \equiv \frac{1}{r} \tag{12}$$

代入式 11，得到 u 关于 θ 的微分方程

$$h^2\left(\frac{\mathrm{d}^2 u}{\mathrm{d}\theta^2} + u\right) = GM \tag{13}$$

这就是**比耐公式**，是一个二阶常系数非齐次微分方程，通解[??] 为

$$u=\frac{1}{p}\left[1-e\cos(\theta+\varphi)\right] \tag{14}$$

其中

$$p=\frac{h^2}{GM} \tag{15}$$

e,φ,h 为任意常数．写成关于 r 的函数，得到圆锥曲线

$$r=\frac{p}{1-e\cos\theta} \tag{16}$$

证毕．

拓展阅读 比耐公式[??]

开普勒第二定律的证明

预备知识 开普勒第二定律[3]，角动量守恒（单个质点）[??]

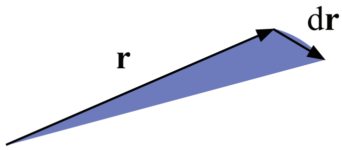

图 1: 微小时间 $\mathrm{d}t$ 扫过的面积

令行星的位矢为 $\mathbf{r}$，在很小一段时间 $\mathrm{d}t$ 内移动了 $\mathrm{d}\mathbf{r}$，于是扫过的面积就是以 $\mathbf{r}$ 和 $\mathrm{d}\mathbf{r}$ 为两条边的三角形的面积 $\mathrm{d}S = 1/2\,|\mathbf{r}|\cdot|\,\mathrm{d}\mathbf{r}|\sin\theta$，其中 θ 为两条矢量的夹角．若把面积看成矢量，方向垂直于三角形所在的平面，则根据叉乘的定义有 $\mathrm{d}\mathbf{S} = \mathbf{r}\times\mathrm{d}\mathbf{r}/2$．两边除以 $\mathrm{d}t$，得扫过面积的速率为

$$\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t} = \frac{1}{2}\mathbf{r}\times\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \frac{1}{2}\mathbf{r}\times\mathbf{v} \tag{1}$$

由“角动量守恒[??]”，行星角动量不变．$\mathbf{r}\times(m\mathbf{v}) = \mathbf{L}$（常矢量，垂直运动平面）所以 $\mathrm{d}\mathbf{S}/\,\mathrm{d}t = \mathbf{L}/2m$ 也是常矢量．写成标量形式，即

$$\frac{\mathrm{d}S}{\mathrm{d}t} = \frac{L}{2m} \tag{2}$$

所以从任意时刻 t_0 开始，在一段时间 Δt 内行星扫过的面积为

$$\Delta S = \frac{L}{2m}\Delta t \tag{3}$$

只和时间间隔有关，而与起始位置无关．

开普勒第三定律的证明

预备知识 开普勒第一定律证明[16]，开普勒第二定律证明[19]，

在开普勒第一定律证明[16] 中，我们得出行星轨道的极坐标方程为

$$r = \frac{p}{1 - e \cdot \cos\theta} \tag{1}$$

其中 $p = h^2/GM$，h 是行星的角动量与质量之比，G 是引力常数，M 是中心天体的质量．e 是圆锥曲线的离心率，当 $0 \leq e < 1$ 时，轨道是椭圆，取等号时，轨道是圆的．椭圆的半长轴为

$$a = \frac{1}{2}\left(r\left(0\right) + r\left(\pi\right)\right) = \frac{1}{1 - e^2}\frac{h^2}{GM} \tag{2}$$

椭圆的半短轴为

$$b = a\sqrt{1 - e^2} = \frac{1}{\sqrt{1 - e^2}}\frac{h^2}{GM} \tag{3}$$

椭圆的面积为

$$S = \pi ab = \frac{\pi}{\sqrt{\left(1 - e^2\right)^3}}\frac{h^4}{\left(GM\right)^2} \tag{4}$$

由“开普勒第二定律证明[19]”中的结论，单位时间扫过的面积为

$$\frac{\mathrm{d}S}{\mathrm{d}t} = \frac{L}{2m} = \frac{h}{2} \tag{5}$$

其中 L 是角动量，m 是行星的质量．所以行星的周期为

$$T = \frac{S}{\mathrm{d}S/\mathrm{d}t} = \frac{2\pi}{\sqrt{\left(1 - e^2\right)^3}}\frac{h^3}{\left(GM\right)^2} \tag{6}$$

所以由式 2 与式 6 得 $T^2/a^3 = 4\pi^2/GM$ 是一个常数，但注意这个常数与中心天体的质量有关，所以不同中心天体的比例常数不同．这个比例系数不用记忆，需要用时，只需用高中的知识计算天体圆周运动时的特例即可．圆作为特殊的椭圆，其半长轴就是半径．

椭圆的三种定义

预备知识 圆锥曲线的极坐标方程[??]

第二种定义

我们前面已经见过椭圆在极坐标中的定义，但从椭圆的极坐标公式难以看出椭圆的对称性，这里用相同的定义推导直角坐标的表达式．我们不妨先以一个焦点为原点定义直角坐标系，且令 x 轴指向另一个焦点，则**??**[??] 变为

$$\frac{\sqrt{x^2+y^2}}{x+h}=e \tag{1}$$

其中 h 为 y 轴到准线的距离．两边平方并整理得

$$(1-e^2)\left(x-\frac{e^2h}{1-e^2}\right)^2+y^2=\frac{e^2h^2}{1-e^2} \tag{2}$$

由此可见，如果我们把椭圆左移 $e^2h/(1-e^2)$，椭圆将具有

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1 \tag{3}$$

的形式．其中 a 为**半长轴**，b 为**半短轴**．这就是椭圆的第二种定义，即把单位圆沿两个垂直方向分别均匀拉长 a 和 b．下面来看系数的关系．首先定义椭圆的焦距为焦点到椭圆中心的距离（即以上左移的距离）为

$$c=\frac{e^2h}{1-e^2} \tag{4}$$

式 2 和式 3对比系数得

$$a=\frac{eh}{1-e^2}\qquad b=\frac{eh}{\sqrt{1-e^2}} \tag{5}$$

不难证明

$$a^2=b^2+c^2 \tag{6}$$

以及

$$e=\frac{c}{a}\qquad h=\frac{b^2}{c} \tag{7}$$

第三种定义

椭圆的第三种定义是，椭圆上任意一点到两焦点的距离之和等于 $2a$. 现在我们来证明前两种定义下的椭圆满足这个条件. 由直角坐标方程可知对称性，可在椭圆的两边做两条准线，令椭圆上任意一点到两焦点的距离分别为 r_1, r_2，到两准线的距离分别为 d_1, d_2，则有

$$e = \frac{r_1}{d_1} = \frac{r_2}{d_2} = \frac{r_1 + r_2}{d_1 + d_2} \tag{8}$$

所以

$$r_1 + r_2 = e(d_1 + d_2) = 2e(c + h) = 2\frac{c}{a}\left(c + \frac{b^2}{c}\right) = 2a \tag{9}$$

证毕.

双曲线的三种定义

预备知识 圆锥曲线的极坐标方程[??]

第二种定义

双曲线的极坐标方程为（$e>1$）

$$r=\frac{p}{1-e\cos\theta} \tag{1}$$

以与极坐标系相同的原点建立直角坐标系，要把以上方程变到直角坐标系中，将 $r=\sqrt{x^2+y^2}$，$\cos\theta=x/\sqrt{x^2+y^2}$ 代入得

$$\sqrt{x^2+y^2}=p+ex \tag{2}$$

两边平方且化简得

$$\frac{(e^2-1)^2}{p^2}\left(x+\frac{ep}{e^2-1}\right)^2-\frac{e^2-1}{p^2}y^2=1 \tag{3}$$

把双曲线沿 x 轴正方向移动 c，可得以下形式

$$\frac{x^2}{a^2}-\frac{y^2}{b^2}=1 \tag{4}$$

这就是双曲线的第二种定义，从上式容易看出，双曲线的两支是左右对称的．以上两式对比系数得

$$a=\frac{p}{e^2-1}\qquad b=\frac{p}{\sqrt{e^2-1}}\qquad c=\frac{ep}{e^2-1} \tag{5}$$

$$c^2=a^2+b^2 \tag{6}$$

用 a,b,c 表示 e,p 有

$$e=\frac{c}{a}\qquad p=\frac{b^2}{a} \tag{7}$$

由离心率的定义，双曲线的焦点到准线的距离为 $p/e=b^2/c$，准线的坐标为 $c-p/e=a^2/c$．由对称性，双曲线有两个焦点和两条准线，任意一个焦点到双曲线两支的任意一点比上该点到焦点同侧准线的距离都等于离心率．

第三种定义

双曲线的另一种定义是，曲线上任意一点到两个焦点距离之差等于 $2a$．这里证明前两种定义满足该性质．由对称性，我们不妨只考虑右支上的某点，令其到右焦点和右准线的距离分别为 r_1 和 d_1，到左焦点和左准线的距离分别为 r_2 和 d_2．由离心率的定义，有

$$e = \frac{r_1}{d_1} = \frac{r_2}{d_2} = \frac{r_2 - r_1}{d_2 - d_1} \tag{8}$$

由于两准线之间的距离恒为 $2a^2/c$，上式变为

$$r_2 - r_1 = e(d_2 - d_1) = 2a \tag{9}$$

证毕．

渐近线

当 $x, y \to \infty$ 时，式 4 中的 1 可以忽略不计，有 $y/x = \pm b/a$，渐近线与 x 轴夹角为

$$\theta_0 = \arctan(b/a) \tag{10}$$

两条渐近线到两个焦点的距离都为

$$c \sin \theta_0 = c \cdot b/c = b \tag{11}$$

Matlab 编程基础

Matlab（Matrix Laboratory）的中文名叫矩阵实验室，是一款著名的科学计算软件，也指这个软件中使用的编程语言．这里仅介绍最基本的 Matlab 功能和语法，且仅介绍本书使用到的功能．如果只需要了解本书的算法而不想使用 Matlab，可略过界面相关的介绍．

界面介绍

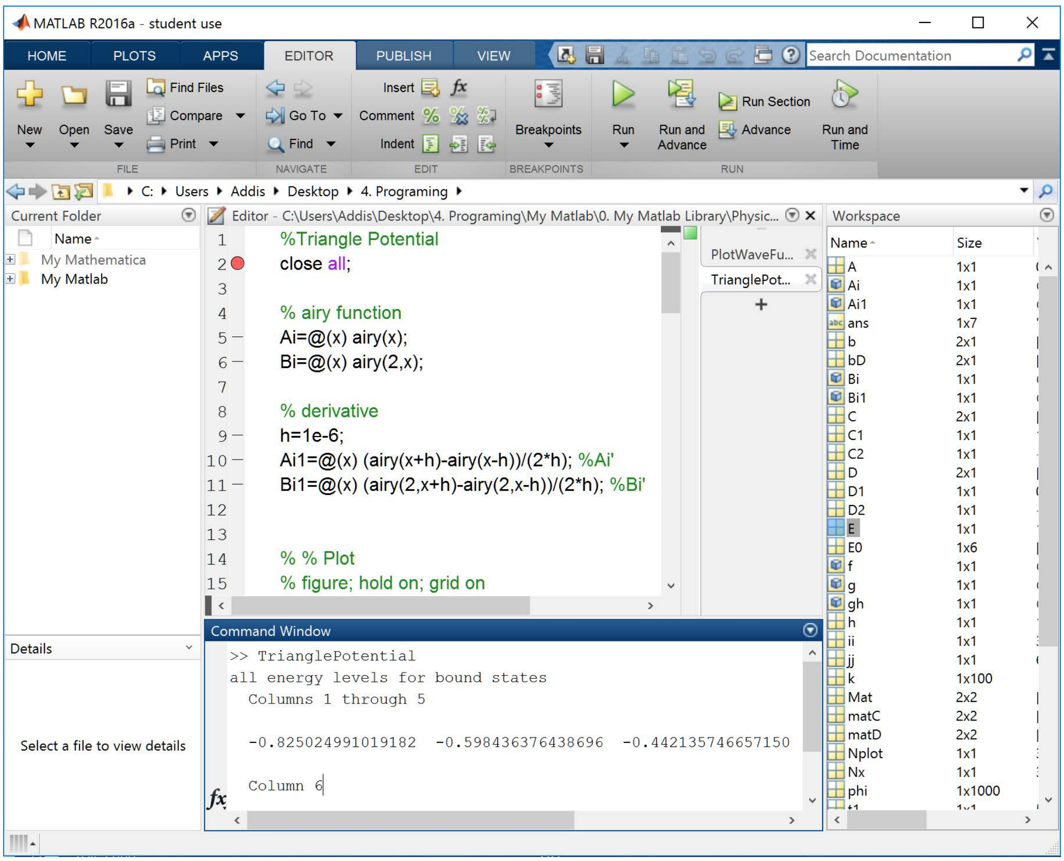

图 1: Matlab 的 IDE 界面

Matlab 的编程界面（图 1）属于**集成开发环境（IDE/Integrated Develop Environment）**，简而言之就是一切与 Matlab 编程有关的工作都可以在该界面

完成[5]. 以下介绍界面中常用的窗口. 要选择显示的窗口，可在 Home 菜单中点击 Layout 按钮，并在 Show 下面勾选需要的窗口.

Editor 用于编辑代码，同时具有自动检测语法错误，代码调试等功能. Matlab 的代码文件分为脚本文件和函数文件两种形式，后缀名都为 m，用图 1 中 Editor 菜单栏的 Save 按钮可保存代码文件. Matlab 作为一种**解释语言（Interpreted Language）**可以直接在 Editor 中运行源代码，无需传统的编译过程. 为了让 Matlab 能运行代码文件，需要把文件所在的目录[6]添加到 Matlab 的搜索路径下.

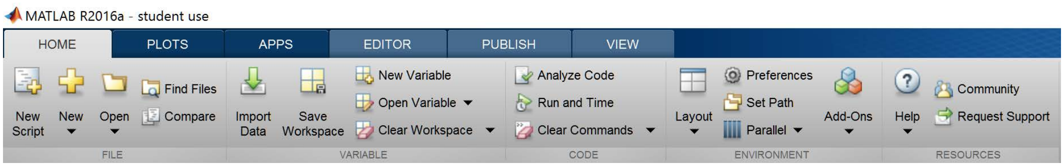

图 2: Home 菜单

如图 2，Home 菜单中的 Set Path 按钮可以设置 Matlab 的搜索路径. 点开后用 Add Folder 按钮可以添加单个文件夹（不包含子文件夹），用 Add with Subfolders 添加文件夹（包含子文件夹）. 用 Remove 删除已添加的路径，用 Default 还原初始设置，用 Save 保存修改，用 Close 关闭窗口. 若要运行程序，回到 Editor 菜单点击 Run 按钮即可.

Command Window 主要用于输入临时指令或者调试程序，可输入除了函数定义外的任意指令. Command Window 只能按输入顺序执行，不方便修改和编辑，如果指令较长或有多个指令，应该使用 Editor. 在 Command Window 中按回车执行输入的指令，按上箭头可重复已输入的指令.

Workspace 用于查看 Matlab 当前的所有变量的列表. Matlab 的所有变量都可以理解为矩阵，单个值可理解为 1×1 的矩阵. 列表中 Name 是变量名，Size 是矩阵维度，Value 是变量值，右上角的下拉菜单中的 Choose Columns 中还可设置显示更多属性，例如 Bytes 是占用字节数，Class 是变量类型，Min 是最小值，Max 是最大值，Mean 是平均值，Median 是中位数，Std 是标准差等. 双击 Workspace 中的变量可显示变量值.

[5]界面语言默认与操作系统语言相同，本书使用英文界面.

[6]在英文界面下 Matlab 不能识别中文目录，建议用英文命名文件夹.

计算器

下面我们仅用 Command Window 来熟悉 Matlab 的基本语法．我们先看如何把 Matlab 当做普通的科学计算器使用．

```
>> 1.2/3.4 + (5.6+7.8)*9 -1
ans = 119.9529
>> 1/exp(1)
ans = 0.3679
>> exp(-1i*pi)+1
ans = 0
```

常用的运算符号有

+，-，*，/，^（指数）

常用的数学函数有

sqrt（开方），exp，sin，cos，tan，cot，asin，acos，atan，acot，real（实部），imag（虚部），conj（共轭）

等（三角函数前面加 a 代表其反函数）．运算的优先顺序与数学上的习惯一样．注意这些函数的自变量都可以是复数．为了区分虚数单位 i 和变量 i，好习惯是在 i 前面加数字（上面的第三条命令）．

mod(N,n) 是求余运算，计算 N 被 n 整除后的余数．注意这个函数有两个变量，用逗号隔开．要注意在 Matlab 中，这种有输入和输出的命令都是广义的**函数（function）**，不仅是数学函数．

用大写 E 或小写 e 表示科学计数法（不允许有空格），如 2.997×10^8 表示为 2.997e8 或 2.997e+8．用小写 pi 表示圆周率，用 exp(1) 表示自然对数底，用 1+2i 或 1+2j 等表示复数，注意 i 和 j 前面不能有空格．

如果需要多位小数，可使用 format long 命令使结果显示为**双精度**（约 16 位有效数字），用 format short 命令恢复默认格式．

```
>> format long; pi
ans = 3.141592653589793
>> exp(1)
ans = 2.718281828459046
```

变量与矩阵

变量（variable) 可用于传递数据．变量名可以用含有多个字母，数字和下划线，注意变量名区分大小写，且首字符只能是字母．用等号对变量赋值，变量必须在等号左边，等号右边的运算结果会储存在变量中，直到再次被赋值．

```
>> a = 1.2/3.4 + (5.6+7.8)*9 -1
a = 119.9529
>> b = atan(a + 1)
b = 1.5625
```

如果新的变量第一次被赋值,它会自动出现在 Workspace 窗口中.注意 Workspace 中的一个特殊的变量 `ans`，如果命令的输出结果没有赋值给变量，就会自动赋值给 `ans`．注意一般不要对 `ans` 赋值．

另外，如果在命令后面加分号（semicolon）"；"，则命令执行后不输出结果．也可以用分号把多个命令写到一行．在以后用 Editor 编写程序时，每个命令后面都需要加分号，用 diplay 函数在 Command Window 显示结果．

```
>> 1 + 1; a = ans^2
a = 4
```

用 `clear` 命令清空 Workspace 中的所有变量，用 `clear <var1>,<var2> ...` 清除指定的变量（`<var1>,<var2>` 是变量名）．用 `clc` 清空 Command Window（上箭头仍然可以查看历史命令）．

本书只涉及到 3 种**变量类型（class）：双精度（double），字符（char）**和**逻辑（logical)．**

双精度变量用于储存数值，有效数字约为 16 位（如果是复数，实部和虚部各 16 位），取值范围约为 10^{-308} 到 10^{308}．如无变量类型声明，所有命令中出现的常数及储存数值的变量都为 double．

Matlab 中的所有变量都可以理解为**矩阵**，单值变量（标量，scalar）可以理解为 1×1 的矩阵，只有一行或一列的矩阵叫做**行矩阵（row vector）**和**列矩阵（column vector）**．一些简单的矩阵操作如下

```
>> a = [1,2,3]
a = 1   2   3
```

用方括号创建矩阵，用逗号分隔每行的矩阵元，行矢量中逗号可省略

```
>> a = [1 2 3]
a = 1   2   3
```

用分号分隔行

```
>> b = [1;2;3]
b =
    1
    2
    3
>> c = [1 2 3; 2 3 4; 3 4 5]
c =
    1   2   3
    2   3   4
    3   4   5
```

方括号还可以用来合并矩阵（注意矩阵尺寸必须合适）

```
>> d = [a;a]
d =
    1   2   3
    1   2   3
>> e = [a a]
e =
    1   2   3   1   2   3
```

用 `size` 函数获取矩阵尺寸，用 `numel` 函数获取矩阵元个数

```
>> size(d)
ans = 2   3
>> size(d,1)
ans = 2
>> numel(d)
ans = 6
```

用 `zeros` 函数生成全零矩阵

```
>> zeros(2,3)
```

```
ans =
    0  0  0
    0  0  0
```

用 zeros([2,3]) 和 zeros(size(d)) 结果也相同．用 `ones` 可以生成全 1 矩阵，也可以乘以任意常数

```
>> ones(2,3)*5
ans =
    5  5  5
    5  5  5
```

用 `eye(N)` 生成 $N \times N$ 的单位矩阵，用 `rand(M,N)` 生成随机矩阵，矩阵元从 0 到 1 均匀分布．用 `M:step:N` 生成等差数列（行矢量），例如

```
>> 1:2:10
ans = 1  3  5  7  9
>> 0:pi/3:pi*2
ans = 0  1.0472  2.0944  3.1416  4.1888  5.2360  6.2832
>> 10:-2:1
ans = 10  8  6  4  2
```

用 `linspace(x1,x2,Nx)` 生成指定首项尾项和项数的等差数列（行矢量）

```
>> linspace(0,pi,4)
ans = 0  1.0472  1.2566  2.0944  3.1416
```

下面介绍矩阵运算．同规格的尺寸可以进行 `+` 和 `-` 运算，矩阵和标量也可以；矩阵乘法 `*` 既可以把常数与矩阵相乘，也可以进行数学上的矩阵乘法；矩阵的幂“`^`”相当于矩阵与自己多次相乘；“`/`”可以把矩阵除以一个常数．

```
>> a = [1 2; 3 4]; b = [1 -1; 2 -2];
>> a + b
ans =
    2  1
    5  2
>> a * b
ans =
    5  -5
    11  -11
```

若要两个尺寸相同，可进行**逐个元素运算**，如

```
>> a .* b
ans =
    1  -2
    6  -8
>> a.^2
ans =
    1  4
    9  16
>> a ./ b
ans =
    1.0000  -2.0000
    1.5000  -2.0000
```

单引号“'”可以使实数矩阵转置，或使复矩阵取厄米共轭．若需要对复矩阵转置，用“.'”即可．

```
>> c = a + 1i*b
c =
    1.0000 + 1.0000i  2.0000 - 1.0000i
    3.0000 + 2.0000i  4.0000 - 2.0000i
>> c'
ans =
    1.0000 - 1.0000i  3.0000 - 2.0000i
    2.0000 + 1.0000i  4.0000 + 2.0000i
>> c.'
ans =
    1.0000 + 1.0000i  3.0000 + 2.0000i
    2.0000 - 1.0000i  4.0000 - 2.0000i
```

Matlab 自带的数学函数一般支持矩阵自变量，结果是该函数对每个矩阵元分别运算

```
>> cos(0:pi/4:pi)
ans =
    1.0000  0.7071  0.0000  -0.7071  -1.0000
```

用矢量运算可以使代码简短易懂，且提高计算效率．

字符型变量一般用于控制行输出结果或对生成的图片进行标注．把 N 个字符放在一对单引号内，可生成 1 × N 的字符类型数组．

```
>> str1 = '这是一个字符串'; str2 = 'this is a string'
>> [str1, ',', str2]
ans =
    这是一个字符串, this is a string
>> numel(str)
ans = 24
```

把双精度变类型变为字符串用 num2str 函数（注意“2”的读音与“to”相同，“num”代表“number”，“str”代表字符串“string”），通常用于与其他字符矩阵合并，如

```
>> number = 3; str = ['The number is ', num2str(number), '.']
str =
    The number is 3.
```

若要在字符串中加入英文单引号，可用两个英文单引号表示.

逻辑变量只能具有 0 或 1 两个值，分别代表**假**（**false**）和**真**（**true**）. 最常用的地方是判断语句 if 的后面以及获取矩阵元. 逻辑算符有

>, >=（大于等于）, <, <=, ==（等于）

可用于比较双精度数组，返回逻辑型数组

```
>> L = 1 + 1 > 3
L = 0
```

逻辑“与”,“或”,“非”算符分别为（仅用于逻辑标量）&&, ||, ~. 例如

```
>> 1 > 0 && 2 > 1
ans = 1
>> 1 > 0 || 2 < 1
ans = 1 >>    (1 > 0)
```

只有两边都为真时，与运算才能为真. 至少有一边为真，或运算就为真. 非运算把真假互换，注意必须要加括号.

矩阵索引用于表示矩阵部分矩阵元，例如

```
>> a = [1 2 3; 4 5 6; 7 8 9];
ans =
    1  2  3
```

```
    4  5  6
    7  8  9
>> a(1,2)
ans = 2
>> a(2:3,1)
ans =
    4
    7
>> a(2:3,1:2)
ans =
    4  5
    7  8
>> a([2,3],[1,2])
ans =
    4  5
    7  8
>> a(:,2)
ans =
    2
    5
    8
>> a(1:end-1,2:3)
ans =
    2  3
    5  6
```

其中 end 表示某维度的最大索引（仅在索引中有效）. 以上的格式可以归结为"在小括号中用逗号把行矢量隔开". 注意索引不仅可以用来取值，还可以放在等号左边赋值.

```
>> b = a; b(1:3) = a(2:4)
b =
    4  2  3
    7  5  6
    2  8  9
```

要求左边的矩阵元个数等于右边. 唯一的例外是当右边为标量

```
b(1:3) = 0
```

```
b =
    0   2   3
    0   5   6
    0   8   9
```

我们还可以用单个索引

```
>> a(2:5)
ans = 4   7   2   5
>> a(7:end)
ans = 3   6   9
```

注意单个索引的顺序是先增加第一个维度（行标），再增加第二个维度（列标）.

我们还可以用相同大小逻辑数组索引

```
>> mark = logical([1 0 0; 0 0 1; 1 0 0]); a(mark)
ans =
    1
    7
    6
```

逻辑索引常见的例子如

```
>> a(a-3 <= 1)
ans =
    1
    4
    2
    3
```

注意逻辑索引中不能使用双精度类型代替逻辑类型.

脚本文件

在讲解更复杂的程序结构前，我们先来看脚本文件. **脚本（script）文件**是包含若干个指令的文件，文件后缀名为 m. 脚本文件可以单独执行，也在其他文件或 Command Window 中被调用（记得设置搜索路径）. 后者相当于把被调用脚本的代码直接插入到调用指令处，调用指令就是脚本文件的文件名. 脚本中的每条命令后面应该加分号以隐藏输出结果，若需要输出，用 `disp` 函数.

该函数的自变量可以是字符串或矩阵.

在脚本文件中，可以在行首或命令后用百分号 % 进行**注释**（**comment**）. 注释是程序的说明，使程序更易读，在执行程序时会被忽略（图 1）.

```
>> disp('good'); a = 3; disp(['a = ',num2str(a)])
good
a = 3
```

注意即使使用了分号，disp 仍然会显示结果.

判断结构

现在来看一段代码（脚本文件）

```
a = rand(1,1); b = 0.5;
if a > b
    disp('a is larger');
else
    disp('b is larger');
end
```

不难猜测出这里的 if 用于判断，如果条件满足，则只执行 if 和 else 之间的指令. 如果条件不满足，则只执行 else 到 end 的指令. 以上程序随机生成从 0 到 1 的数，如果随机数大则输出第一段文字，否则输出第二段文字.

循环结构

用 for 表示循环

```
for ii = 1:3
    disp(['ii^2 = ' num2str(ii^2)]);
end
```

运行结果为

```
ii^2 = 1
ii^2 = 4
ii^2 = 9
```

容易看出这段代码被执行了 3 次，**循环变量** `ii` 按顺序取 1:3 中的一个矩阵元. 注意选取 `ii` 作为变量名是为了与虚数单位区分，也可以选择其他变量名. 再来看一个稍复杂的循环

```
Nx = 5;
x = zeros(1,Nx);
x(1) = 2;
for ii = 2:numel(x)
    x(ii) = x(ii-1)^2;
end
disp(['x = ' num2str(x)])
```

在循环开始前 `x(1)` 被赋值为 2，在循环中，第 `ii` 个矩阵元依次被赋值为第 `ii-1` 个矩阵元的平方. 运行结果为

```
x = 2   4   16   256   65536
```

注意在循环前用 `zero` 对矩阵进行了**预赋值（preallocation）**. 预赋值不是必须的，但如果不进行预赋值，每次循环矩阵的尺寸都要改变，会导致程序运行变慢. 另外注意循环中不允许给循环变量赋值.

函数文件

我们已经学了一些函数，现在来看如何自定义函数. 自定义函数需要一个单独的**函数文件**. 的第一个命令必须是 function，用于定义主函数. 文件名必须与主函数同名. 文件中其他函数都是子函数. 主函数可以调用子函数，子函数可以调用同文件中的其他子函数，但不能调用主函数，主函数和子函数都可以调用 Matlab 的内部函数或搜索路径下其他函数文件中的主函数. 若函数文件在搜索路径下，其他 m 文件或 Command Window 中可以直接调用它的主函数. 注意函数文件中的子函数不能从文件外被调用. 函数的 workspace 是独立的，即 < 定义函数的指令 > 在执行的过程中，不能获取除 < 变量 > 外的变量，除非定义了全局变量（见 global）. 注意函数只能通过函数文件定义，不能在脚本或控制行中定义.

函数句柄（function handle）

函数句柄是一种特殊的变量类型，可用于定义一个临时的函数，也可传递到其他函数中．首先，对于已经存在的函数（包括函数文件定义的），可直接在函数名前面加 @ 生成函数句柄

```
>> f = @sin
>> f(pi/2)
ans = 1;
```

若函数由若干个运算组合而成，或包含若干其他变量，要在 @ 后面指定函数句柄的变量

```
>> A = 3; w = 5; phi = pi/2;
>> f = @(x) A*sin(w*x+phi) + (2*x./pi).^2;
>> f([0,pi/2])
ans =
    3.0000   1.0000;
f = @(x,phi) A*sin(w*x+phi) + (2*x./pi).^2;
>> f([0,pi/2],pi/2)
ans =
    3.0000   1.0000;
```

注意这里用了逐个元素算符，使函数支持矩阵输入．

自定义函数（function）

格式为

[< 输出 1>,< 输出 2>,...] = function < 函数名 >(< 变量 1>,< 变量 2>...)

< 定义函数的指令 >

end

其中 < 函数名 > 是字母，数字和下划线的组合，例如 MyFun_123，第一个字符不能是数字或下划线．若函数无变量，则小括号可省略．若函数无输出，则等号及方括号可省略．

函数的调用格式为

[< 输出 1>,< 输出 2>,...] = < 函数名 >(< 变量 1>,< 变量 2>...)

程序调试

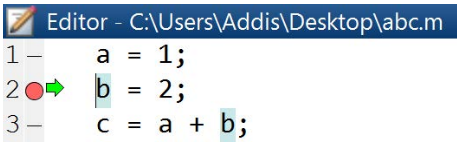

图 3: 在行首设置 Breakpoint

若要调试程序，可选择一行代码并单击该行前面的横线，这时会出现红色圆点 Breakpoint （图 3），程序运行到 Breakpoint 会暂停.

此时要查看变量情况，可通过 Workspace 查看各个变量的情况，也可用光标悬停在某个变量上．还可以用 Command Window 改变某些变量的值，或画图等．在这种调试状态下，也可以通过 Edit 菜单中的一些按钮控制接下来程序如何运行（图 4）．其中“Continue”（快捷键 F5）是继续运行直到下一个

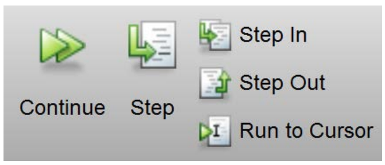

图 4: Step 菜单

Breakpoint 或结束．“Step”（F10）是运行到下一行，“Step In”（F11）是进入子程序并暂停，“Step Out”是运行完当前子程序并回到子程序被调用的地方．“Run to Cursor”是运行到光标所在处.

分节

在行首用两个百分号“%%”可以对代码进行分节（图 5）．这样做一是可以使代码结构更清晰，二是可以单独选择某一节运行（Edit 菜单中的“Run Section”按钮）.

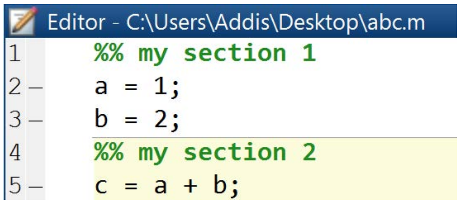

图 5: 代码分节

工具箱（Toolbox）

在购买和安装 Matlab 软件时，可以选择各种各样的工具箱，常用的工具箱有曲线拟合（Curve Fitting，从离散的数据点得到一条曲线），图像处理（Image Processing，图像变换，增强，降噪，二值化等），图像获取（Image Acquisition，从相机获取图像），Matlab 编译器（MATLAB Compiler，编译代码，提高运行速度）．注意使用了工具箱功能的代码在没有工具箱的 Matlab 软件上将无法运行．

Matlab Online

具有 Matlab 的基本功能，和类似于软件的界面，需要购买了正版 Matlab 的 Matlab 账号登录（学生账号也可以）．若账号购买了工具箱（Toolbox），也可以使用对应的工具箱．本书官网提供免费的 Matlab 账号供读者试用和体验 Matlab Online，网址见本书封面或前言．

二项式定理（非整数）的数值验证

预备知识 二项式定理（非整数）[??]，Matlab 编程基础[25]

这里介绍一个简单的 Matlab 程序用于计算**??**[??]．第 7-10 行的循环中，每个循环通过 `x^(ii-1)` 项的系数计算 `x^ii` 项的系数，并将 `x^ii` 项累加到求和结果 `result` 上．

```matlab
1 % 验证二项式定理(非整数幂)
2 u = -3.5; % 幂
3 x = 0.6; % |x|<1 使级数收敛
4 N = 100; % 求和项数
5 Coeff = 1; % x^ii 项前面的系数
6 result = 1; % 求和结果
7 for ii = 1:N
8     % 由 x^(ii-1) 项系数计算 x^ii 项系数
9     Coeff = Coeff*(u-ii+1) / ii;
10    % 将 x^(ii-1) 项累加到求和结果上
11    result = result + Coeff * x^ii;
12 end
13 disp('求和结果为')
14 disp(result)
15 format short % 恢复默认显示
16 disp('精确结果为')
17 format long % 显示全部小数位
18 disp((1+x)^u)
```

弹簧振子受迫运动的简单数值计算

预备知识 简谐振子受迫运动[??]，微分近似[??]，Matlab 编程基础[25]

以下以弹簧振子的受迫运动为例，介绍一种解 n 阶微分方程的简单方法，以便了解数值解微分方程的基本思想. 但是这种方法误差较大，需要大量计算才能获得较精确的数值解. 在实际运用中已有更复杂更成熟的算法（参考 MATLAB 常微分方程（组）数值解简介）.

在弹簧振子受迫运动[??] 中，列出的二阶微分方程为

$$m\ddot{y} = \alpha\dot{y} - ky + f(t) \tag{1}$$

若已知初值条件（可代入任意具体数值）$\dot{y}(0) = v_0$，$y(0) = y_0$，且已知驱动力 $f(t)$，此时可以把初值条件代入式 1 求出 $t = 0$ 时的加速度.

$$\ddot{y}(0) = [-\alpha\dot{y}(0) - ky(0) + f(0)]/m \tag{2}$$

接下来的一小段微小的时间 Δt 内（Δt 称为**步长**，步长越小误差越小），根据微分近似，可以算出 $t = \Delta t$ 时刻的状态.

$$y(\Delta t) = y(0) + \Delta y \approx y(0) + \dot{y}(0)\Delta t \tag{3}$$

$$\dot{y}(\Delta t) = \dot{y}(0) + \Delta\dot{y} \approx \dot{y}(0) + \ddot{y}(0)\Delta t \tag{4}$$

微分近似在这里的物理意义是在 Δt 内速度和加速度都近似为常数. 把 $y(\Delta t)$ 和 $\dot{y}(\Delta t)$ 再次代入式 1

$$\ddot{y}(\Delta t) = [-\alpha\dot{y}(\Delta t) - ky(\Delta t) + f(\Delta t)]/m \tag{5}$$

再次使用微分近似有

$$y(2\Delta t) = y(\Delta t) + \Delta y \approx y(\Delta t) + \dot{y}(\Delta t)\Delta t \tag{6}$$

$$\dot{y}(2\Delta t) = \dot{y}(\Delta t) + \Delta\dot{y} \approx \dot{y}(\Delta t) + \ddot{y}(\Delta t)\Delta t \tag{7}$$

重复以上各步骤，就可以继续得到 $y(3\Delta t)$，$y(4\Delta t)$ 等的近似值. 在 y-t 图中把这些散点连接起来，就得到了 $y(t)$ 的函数图.

例程 SHOFN.m

```
% 参数设定
m = 0.1; % 质量
k = 1; % 劲度系数
a = 0.03; % 阻尼系数
T = 20; % 停止时间
Nstep = 10000; % 步数
A = 2; w = 3; % f(t)=A*sin(w*t);

dt = T/Nstep; % 计算步长
y2 = zeros(step,1); y1 = y2; y = y2; % 矩阵预赋值
y(1) = 0; y1(1) = 0; % 初值, y1 是 y 的一阶导数

% 迭代循环
for ii = 2:step
    y2(ii) = (-a*y1(ii)-k*y(ii)+2*sin(w*(ii*dt)))/m; % 代入微分方程求出 y''.
    y(ii) = y(ii-1) + y1(ii-1)*dt; % y 的微分近似
    y1(ii) = y1(ii-1) + y2(ii-1)*dt; % y' 的微分近似
end

% 画图
t=(0:step-1)*dt;
plot(t,y);
```

运行结果如图 1，可见开始时驱动力不断给弹簧振子补充能量，振幅变大. 当补充的功率等于消耗的功率时，弹簧做稳定振动.

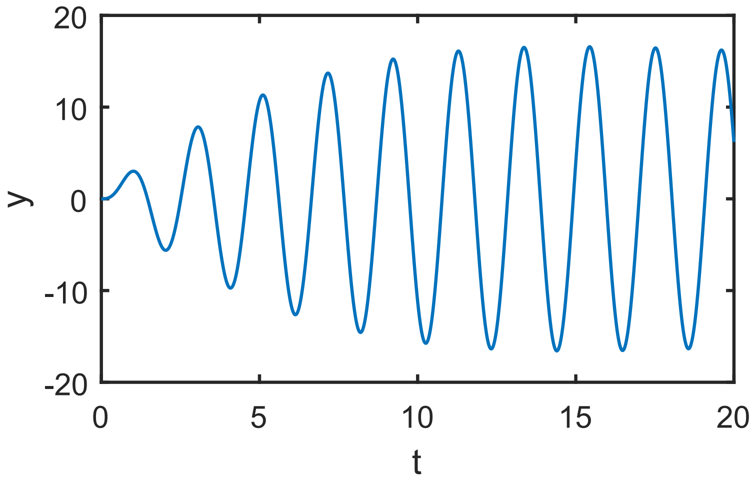

图 1: 运行结果

天体运动的简单数值计算

预备知识 万有引力[??]，弹簧振子受迫运动的简单数值计算[41]

直角坐标系中，设中心天体质量为 M，固定在原点不动. 根据牛顿万有引力定律，质量为 m 的行星受到中心天体的力为

$$\mathbf{F} = -G\frac{Mm}{r^2}\mathbf{r} = -G\frac{Mm}{r^2}\mathbf{r} \tag{1}$$

其中 $\mathbf{r}$ 为行星的位矢（设行星在 xy 平面上运动）. 根据牛顿第二定律[??]，加速度为

$$\mathbf{a} = \frac{\mathbf{F}}{m} = -G\frac{M}{r^3}\mathbf{r} \tag{2}$$

用 $r = \sqrt{x^2+y^2}$ 以及 $\mathbf{r} = x\hat{\mathbf{x}}+y\hat{\mathbf{y}}$，其中 x, y 看成 t 的函数. 考虑到 $a_x = \mathrm{d}^2x/\mathrm{d}t^2$，$a_y = \mathrm{d}^2y/\mathrm{d}t^2$，可以列出二阶微分方程组

$$\left\{\begin{aligned} \frac{\mathrm{d}^2x}{\mathrm{d}t^2} &= -\frac{GM}{(x^2+y^2)^{3/2}}x \\ \frac{\mathrm{d}^2y}{\mathrm{d}t^2} &= -\frac{GM}{(x^2+y^2)^{3/2}}y \end{aligned}\right. \tag{3}$$

假设已知初值条件 $x(0) = x_0$，$y(0) = y_0$，$\dot{x}(0) = v_{x0}$，$\dot{y}(0) = v_{y0}$. 下面用“簧振子受迫振动的简单数值计算[41]”中类似的方法求接下来行星的运动轨迹.

1．将初始条件代入式 3，得到初始加速度

$$\begin{cases} \ddot{x}(0) = -\dfrac{GM}{(x_0^2+y_0^2)^{3/2}}x_0 \\ \ddot{y}(0) = -\dfrac{GM}{(x_0^2+y_0^2)^{3/2}}y_0 \end{cases} \tag{4}$$

2．设经过一段极微小的时间步长 Δt（例如 0.0001，数值越小误差越小），根据微分近似（微分近似在这里的物理意义是在 Δt 内速度和加速度都近似为常数）

$$\begin{cases} \dot{x}(\Delta t) \approx \ddot{x}(0)\Delta t + x(0) \\ \dot{y}(\Delta t) \approx \ddot{y}(0)\Delta t + \dot{y}(0) \end{cases} \qquad \begin{cases} x(\Delta t) \approx \dot{x}(0)\Delta t + x(0) \\ y(\Delta t) \approx \dot{y}(0)\Delta t + y(0) \end{cases} \tag{5}$$

3．把 $x(\Delta t), y(\Delta t)$ 再次代入式 3，得到 $x''(\Delta t), y''(\Delta t)$，再次利用微分近似求出 $x(2\Delta t), y(2\Delta t), x'(2\Delta t), y'(2\Delta t)\dots$ 如此循环下去就可以得到每隔 Δt 的数值解.

MATLAB 例程

```
% 参数设定
GM = 1; % 万有引力常数乘以中心天体质量
x0 = 1; y0 = 0; % 初始位置
vx0 = 0; vy0 = 0.7; % 初始速度
T = 4; Nstep = 4000; % 总时间和步数
dt = T/Nstep; % 步长

% 矩阵预赋值
x = nan(Nstep,1); y = x;
x1 = x;  y1 = x;
x2 = x; y2 = x;

% 初始位置，速度，加速度
x(1) = x0; y(1) = y0; % 初位置
x1(1) = vx0; y1(1) = vy0; % 初速度
```

```
x2(1) = -GM*x(1)/(x(1)^2+y(1)^2)^(3/2); % 代入方程得到 x''(0)
y2(1) = -GM*y(1)/(x(1)^2+y(1)^2)^(3/2); % 代入方程得到 y''(0)

% 迭代循环
for ii = 2:Nstep
    x(ii) = x(ii-1)+x1(ii-1)*dt; % x的微分
    y(ii) = y(ii-1)+y1(ii-1)*dt; % y的微分

    x1(ii) = x1(ii-1)+x2(ii-1)*dt; % x' 的微分
    y1(ii) = y1(ii-1)+y2(ii-1)*dt; % y' 的微分

    x2(ii) = -GM*x(ii)/(x(ii)^2+y(ii)^2)^(3/2); % 代入微分方程求出 x''
    y2(ii) = -GM*y(ii)/(x(ii)^2+y(ii)^2)^(3/2); % 代入微分方程求出 y''
end

% 画图
plot(x,y); % 画行星轨道
axis equal; % xy坐标长度一致
hold on; % 继续画图
scatter(0,0); % 标出中心天体
```

程序运行结果如图 1 所示. 注意行星轨道并不是一个闭合的椭圆，这是由于这种算法误差较大，为了减小误差，可以增加程序中 Nstep 的值.

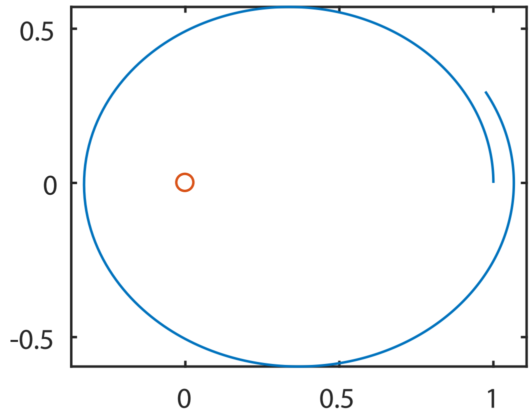

图 1: 运行结果

本书编写规范

预备知识 本书编写规范[46]

软件使用规范

本书使用 TeXLive2016 软件中的 XeLaTeX 进行编译．如果 Windows 中编译卡在 eu1lmr.fd 上的时间较长，说明 font config 有问题，在 Windows 的控制行运行“fc-cache -fv”，重启 TeXLive，多试几次即可．TeXWork 编辑器中 Ctrl+T 编译，Ctrl+ 单击跳转到对应的 pdf 或代码，在 pdf 中 Alt+ 左箭头返回上一个位置．代码中\beq+Tab 生成公式环境，\sub+Tab 生成 subsection．Ctrl+F 进行查找，Ctrl+G 查找下一个．菜单中的 Edit>Preference 设置默认字体为 Microsoft YaHei UI（11pt），默认编译器为 XeLaTeX，编码选择 UTF-8．

搜索文件夹内所有文档的内容用 FileSeek 软件，搜索空格用“\空格”，搜索“$”用“\$”，以此类推．对比两个文档或文件夹用 WinMerge 软件．

画图用 Adobe Illustrator 和 Autodesk Graphics，用 MathType 在图中添加公式，希腊字母粗体正体矢量用从 Symbol 字体中插入，更简单的方法是，先输入希腊字母，选中，然后在 Style 里面选 Vector-Matrix．

文件版本管理

使用 GitHub Desktop，用 PhysWiki repository 管理所有文件，每次 commit 需要做的事情如下

- 用 FileSeek 替换所有文档中的空心句号.
- 确保所有文档可以顺利编译.
- 解决编译产生的 warning.
- 清除编译产生的中间文件，包括 content 文件夹中的非 tex 文件.
- 把 ManicTime 记录的写作时间记录到“计时.txt”.
- 检查变化的内容.

每次 commit 的标题必须是下列之一

- 常规更新：包括完善词条，新词条等.
- 词条统计：统计文件夹，对照表，和书中的词条，查看不一致或缺失.
- 模板更新：模板有更新.

词条统计的方法：首先把 contents 文件夹中的所有文件名按顺序排列，复制到表格中，然后把词条对照表中的所有标签在表格中找到对应项，做标记，并把对照表中的词条名粘贴到表格中．最后到 PhysWiki.tex 中逐个把标签在表格中找到对应项，做标记，对照词条名，并对照词条文件中第一行的词条名.

词条编写规范

词条标签必须限制在 6 个字符内，必须在 PhysWiki.tex，词条标签对照表和词条文件名中一致．词条的中文名必须在 PhysWiki.tex，词条标签对照表和词条文件的第一行注释中一致．如果不是超纲词条，在主文件中用 \entry 命令，否则用 \Entry 命令．非超纲词条放在 contents 文件夹，超纲词条放在 contents1 文件夹．引用词条用 \upref 命令，“预备知识”用 \pentry 命令，“应用实例”用 \eentry 命令，拓展阅读用 \rentry 命令．总文件 PhysWiki.tex 编译较慢，可以先

使用 debug.tex 编译，然后再把 \entry 或 \Entry 指令复制到 PhysWiki.tex 中．注意 PhysWiki.tex 中 \entry 命令的后面可以用 \newpage 命令强制换页，但为了排版紧凑不推荐这么做．

黑色的小标题

正文必须使用中文的括号，逗号，引号，冒号，分号，问号，感叹号，以及全角实心的句号．所有的标点符号前面不能有空格，后面要有空格．行内公式用单个美元符号，且两边要有空格，例如 $a^2+b^2=c^2$，后面有标点符号的除外．方便的办法是先全部使用中文标点，最后再把所有空心句号替换成全角实心句号．

公式的 label 必须要按照“词条标签 _eq 编号”的格式，只有需要引用的公式才加标签，编号尽量与显示的编号一致，但原则上不重复即可．图表的标签分别把 eq 改成 fig 和 tab 即可，例题用 ex．但凡是有 \caption 命令的，\label 需要紧接其后．

$$(a+b)^n = \sum_{i=0}^n C_n^i a^i b^{n-i} \quad (n\text{为整数}) \tag{1}$$

引用公式和图表都统一使用 \autoref 命令，注意前面不加空格后面要加空格，例如式 1．如果要引用其他词条中的公式，可以引用“其他词条[46]”的式 1 也可以用“式 1[46]”，为了方便在纸质书上使用，词条页码是不能忽略的．

错别字替换

正文中常见的错别字如“他们”（它们），“一下”（以下），可以时常搜索替换．

公式规范

公式中的空格从小到大如 $a\,b\:c\quad d\qquad e$，微分符号如 $\mathrm{d}x$，自然对数底如 e，双重极限如

$$\lim_{\substack{\Delta x_i\to 0\\ \Delta y_i\to 0}} \sum_{i,j} f(x_i, y_i)\Delta x_i \Delta y_j \tag{2}$$

导数和偏导尽量用 Physics 宏包里面的

$$\frac{\mathrm{d}}{\mathrm{d}x} \quad \frac{\mathrm{d}f}{\mathrm{d}x} \quad \frac{\mathrm{d}^2 f}{\mathrm{d}x^2} \quad \mathrm{d}^2 f/\mathrm{d}x^2 \quad \frac{\partial}{\partial x} \quad \frac{\partial f}{\partial x} \quad \frac{\partial^2 f}{\partial x^2} \quad \partial^2 f/\partial x^2 \tag{3}$$

复数如 $u+\mathrm{i}v$，复共轭如 a^*，行内公式如 a/b，不允许行内用立体分式. 公式中的绝对值如 $|a|$，矢量如 $\mathbf{a}$，手写矢量如 $\vec{a}$，单位矢量如 $\hat{\mathbf{a}}$，矢量点乘如 $\mathbf{A}\cdot\mathbf{B}$（不可省略），矢量叉乘如 $\mathbf{A}\times\mathbf{B}$. 量子力学算符如 $\hat{a}$，狄拉克符号如 $\langle a|\,,|b\rangle\,,\langle a|b\rangle$. 梯度散度旋度拉普拉斯如 $\boldsymbol{\nabla} V$，$\boldsymbol{\nabla}\cdot\mathbf{A}$，$\boldsymbol{\nabla}\times\mathbf{A}$，$\nabla^2 V$，但最好用 $\boldsymbol{\nabla} V$，$\boldsymbol{\nabla}\cdot\mathbf{A}$，$\boldsymbol{\nabla}\times\mathbf{A}$，$\nabla^2 V$. 单独一个粗体的 ∇ 用 $\boldsymbol{\nabla}$. 行列式，矩阵 $\mathbf{A}$，转置 $\mathbf{A}^{\mathrm{T}}$，厄米共轭 $\mathbf{A}^\dagger$ 如

$$\begin{vmatrix}1 & 2\\3 & 4\end{vmatrix}\qquad\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}\qquad\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}^{\mathrm{T}}\qquad\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}^{\dagger}\tag{4}$$

行内的列矢量用行矢量的转置表示，如 $(1,2,3)^{\mathrm{T}}$.

行间公式换行及对齐用 aligned 环境，或用自定义的 \ali 命令

$$\begin{aligned}(a-b)^2 &= a^2+b^2-2ab\\&= a^2+b^2+2ab-4ab\\&= (a+b)^2-4ab\end{aligned}\tag{5}$$

$$\begin{gathered}d+e+f=g\\a+b=c\end{gathered}\tag{6}$$

左大括号用自定义的 \leftgroup 命令，里面相当于 aligned 环境

$$\left\{\begin{aligned}d+e+f&=g\\a+b&=c\end{aligned}\right.\tag{7}$$

可变化尺寸的斜分数线如下

$$\frac{\mathrm{d}^2X}{\mathrm{d}x^2}\bigg/X+\frac{\mathrm{d}^2Y}{\mathrm{d}y^2}\bigg/Y+\frac{\mathrm{d}^2Z}{\mathrm{d}z^2}\bigg/Z=\frac{1}{c^2}\frac{\mathrm{d}^2T}{\mathrm{d}t^2}\bigg/T\tag{8}$$

希腊字母如下

$$\begin{aligned}&\alpha(a),\beta(b),\chi(c),\delta(d),\epsilon/\varepsilon(e),\phi(f),\gamma(g),\eta(h),\iota(i),\varphi(j),\kappa(k),\lambda(l),\mu(m),\\&\nu(n),o(o),\pi(p),\theta(q),\rho(r),\sigma(s),\tau(t),\upsilon(u),\varpi(v),\omega(w),\xi(x),\psi(y),\zeta(z)\end{aligned}\tag{9}$$

以下是 script 字母，只有大写有效. 所谓大写 ϵ 其实是花体的 E.

$$\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{E},\mathcal{F},\mathcal{G},\mathcal{H},\mathcal{I},\mathcal{J},\mathcal{K},\mathcal{L},\mathcal{M},\mathcal{N},\mathcal{O},\mathcal{P},\mathcal{Q},\mathcal{R},\mathcal{S},\mathcal{T},\mathcal{U},\mathcal{V},\mathcal{W},\mathcal{X},\mathcal{Y},\mathcal{Z}\tag{10}$$

另外，电介质常数一律用 ϵ 而不是 ε.

写单位，用 \si，如 $a=100\ \mathrm{m/s^2}$.

图表

现在来引用一张图片，图片必须以 eps 以及 pdf 两种格式放在 figures 文件夹中，代码中使用 pdf 图片．图片宽度一律用 cm 为单位．在图 1 中，label 只

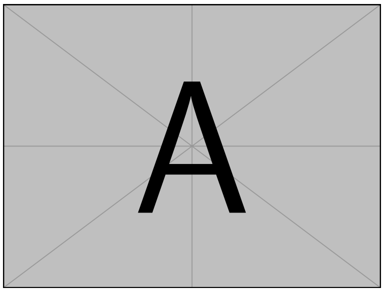

图 1: 例图

能放在 caption 的后面，否则编号会出错．由于图片是浮动的，避免使用“上图”，“下图”等词．

再来看一个表格，如表 1．注意标签要放在 caption 后面，使用 tb 命名．

表 1: 极限 e 数值验证

x	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
$(1+x)^{1/x}$	2.59374	2.70481	2.71692	2.71815	2.71827	2.71828

下面我们举一个例子并引用

例 1　名称

在例子中，我们的字体可以自定义，包括公式的字号会保持与内容一致．

$$(a+b)^n=\sum_{i=0}^{n}C_n^i a^i b^{n-i}(n\text{为整数}) \tag{11}$$

引用例子同样使用 \autoref，如例 1．

以下给出一段 Matlab 代码，代码必须有“.m”和“.tex”两个版本，放在 codes 文件夹中．

Matlab 代码

显示 Command Window 中的代码

```
>> 1.2/3.4 + (5.6+7.8)*9 -1
ans = 119.9529
>> 1/exp(1)
ans = 0.3679
>> exp(-1i*pi)+1
ans = 0
```

显示 m 文件中的代码

代码必须以.tex 文件格式放在 code 文件夹中的中，并用 \input 命令导入正文．.tex 代码文件的命名与图片命名相同．禁止在正文中直接写代码．

```
% 验证二项式定理(非整数幂)
u = -3.5;
x = 0.6; % |x|<1 使级数收敛
N = 100; % 求和项数
Coeff = 1; % x^ii 项前面的系数
result = 1; % 求和结果
for ii = 1:N
    Coeff = Coeff*(u-ii+1) / ii;
    result = result + Coeff * x^(ii);
end
disp('直接计算结果为')
format long % 显示全部小数位
disp((1+x)^u)
disp('求和结果为')
disp(result)
```

```
format short % 恢复默认显示
```

应用举例 本书格式规范[46]

拓展阅读 本书格式规范[46]