基于解析法的行星运动轨道及相关参数计算

摘要

行星运动轨道的及其相关参数的确定是天文学中一个重要的议题。本文基于开普勒第一定律及二次曲线理论，结合空间直角坐标系中的观测数据，研究了小行星绕日轨道的确定与参数计算问题。

本文先从二维情况入手。通过分析题中所给的小行星观测数据，构建并求解小行星轨道的椭圆方程；下一步，利用矩阵特征值法分析推导椭圆的标准形式，计算长半轴、短半轴等参数；进一步结合几何分析，求解近日点、远日点坐标及轨道周长；最后，基于二次曲线不变量理论，详细推导了椭圆方程标准化过程。

接下来，本文将情景扩展到三维空间中，进一步画出太阳系七大行星的运动图。首先通过美国国家航天局 NASA 的 app：HORIZEN 查询到太阳系行星的运动数据，再根据空间解析几何的一般研究方法，将三维空间的数据转化到二维平面中；然后通过平面二次曲线的研究方法进一步求解运动轨道并画出运动动图。

研究结果为小行星轨道动力学分析提供了完整的数学模型与参数支持。

关键字　解析法 运动轨道 三维空间

1 问题背景及重述

1.1 问题背景

天文学家通过观测，得到小行星在直角空间中的运动数据（单位：AU）。从解析法入手，求解小行星的运动轨道和轨道的相关参数，从而得到小行星的运动轨道和相关参数，得到进一步研究空间中行星轨道和运动的基础。从二维和三维角度入手，分成两个部分问题求解。此研究结果为小行星轨道动力学分析提供了完整的数学模型与参数支持。

1.2 问题重述

对于二维情况，可分为以下任务：

1. 通过观测数据求得小行星运动轨道的再直角空间中的系数，并绘制其轨道图；
2. 基于二次曲线理论计算椭圆轨道的相关参数（长半轴、短半轴、半焦距、近日点距、远日点距）；
3. 计算小行星近日点、远日点坐标及轨道周长；
4. 通过解析几何中的不变量理论，推导二次曲线变换化简过程。

对于三维空间中的轨道问题，可分成以下任务：

1. 通过美国国家航天局 NASA 的 app：HORIZEN 查询到的太阳系中各行星的历史运动数据；
2. 从单个行星数据入手，基于空间解析几何理论，将三维数据转化成二维数据；
3. 通过第一部分的理论，得到二维数据所对应的椭圆方程，进一步将平面方程变转化成空间方程；
4. 将方程参数化，得到关于时间的运动数据，进一步得到行星运动动图。

2 模型假设

1. 行星观测数据不考虑广义相对论影响；
2. 各行星运动相对独立，不考虑星系中其他行星对其的影响；
3. 太阳系中行星运动中心在各行星的几个中心，忽略其卫星的影响

3 符号说明

表 1: 符号说明

意义	符号	单位
天文单位	AU	1.4959787×10^{11} m
方程的各项参数	$a_i(i = 1, 2, 3, 4, 5)$	1
椭圆标准型参数 (半长、短轴和半焦距)	a、b、c	1
矩阵（向量）	A, B	1
近（远）日点距	$d_m(d_M)$	AU

4 问题分析

天文学中的开普勒第一定律指出：小行星在空间中的运动轨迹是一个椭圆。而椭圆是二次曲线中的一种，故可以从而二次曲线的角度入手，通过解析法解决第一部分的一众问题：

1. 第一小题。由于假设小行星满足开普勒第一定律，可以知道其运动轨道是椭圆。通过解析几何中二次曲线一般形式，可以得到关于方程系数的线性方程组，通过 matlab 相关命令，不难得到方程系数；
2. 第二小题要求椭圆轨道的重要参数（长半轴、短半轴、半焦距、近日点距、远日点距），且题目中已经给出了问题的解决思路。依赖题目条件，此题就是一道纯计算题；
3. 第三小题要求使用恰当的方法解出小行星椭圆轨道的近日点坐标、远日点坐标及椭圆轨道周长。对于近日点和远日点坐标的寻找，可以将其化为在以椭圆的轨道方程约束下，方程上离原点（太阳）的最大最小值的条件极值问题，只用拉格朗日乘数法即可求解。
4. 第四小题要求从二次曲线不变量的相关知识，给出椭圆变换为最简方程形式的推导过程。可以从旋转变换和平移变换入手，分别证明变换矩阵是不变量，也就是单位变换；再从解析几何的角度，推导椭圆变换为最简方程的证明过程。

接下来要求太阳系中行星运动的轨道和运动动图，情形在三维空间中。但是行星运动轨道方程仍然是椭圆，所以依然可以从二次曲线和旋转变换的角度入手：

1. 首先说明空间中的所以曲线都可以转化成单变量方程形式，方便后续将曲线参数化；
2. 由于空间中的所有向量都可以通过两次绕轴的单位旋转得到与坐标轴平行的向量，因此，先确定这两次绕轴变换在三维空间中的矩阵。再将的三维数据通过绕轴变换在到 xOy 平面上；
3. 通过第一部分的第四小问，得到符合化简后的椭圆轨道的方程。再引入参数变换，将关于空间的双变量方程转化成关于时间的单变量方程。最后通过逆变换，即可得到空间中的关于时间的椭圆轨道方程；
4. 使用 set 函数和 drawdown 命令，即可画出动图。

5 模型建立及求解

5.1 第一部分：第一小问

问题指出，小行星的运动轨迹是一个椭圆，且已知以太阳为平面直角坐标系原点的五个观测数据。那么第一阶段此问题完全可以抽象为一个平面解析几何问题，即：已知二次曲线的五点坐标，求此二次曲线的解析方程并进行一系列的分析。下面开始逐步解决问题：

二次曲线的方程为：

$$a_1x^2 + 2a_2xy + a_3y^2 + 2a_4x + 2a_5y + 1 = 0 \tag{1}$$

将数据直接代入曲线方程可以得到一个线性方程组：

$$AX = B \tag{2}$$

其中 A 为已代入数据的系数矩阵且 rank(A)=5，X 为未知数向量，B 为非齐次项。使用 matlab 解出

这个线性方程组立即得到系数向量

$$[a_1 \ \ a_2 \ \ a_3 \ \ a_4 \ \ a_5 \ ] = [-0.3378 \ \ 0.1892 \ \ -0.3818 \ \ 0.4609 \ \ 0.4104] \tag{3}$$

使用 matlab 画出以此系数向量为系数的二次曲线的小行星轨道图。使用函数 fimplicit 即可画出隐函数（方程）图像，再经过一系列标点处理后得到图一

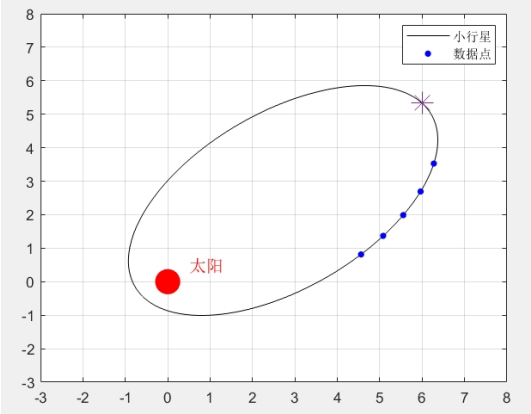

图 1: 小行星轨道相对图

可以发现这个小行星运动轨道是一个椭圆，这符合开普勒第一定律。

5.2 第一部分：第二小问

通过第一小问求得了椭圆轨道的系数向量，很容易就可以得到题目中所需求的两个矩阵

$$C = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \qquad D = \begin{bmatrix} a_1 & a_2 & a_4 \\ a_2 & a_3 & a_5 \\ a_4 & a_5 & 1 \end{bmatrix} \tag{4}$$

使用 matlab 计算矩阵 C 的特征值和 $C,\ D$ 的行列式即可得到椭圆的最简形式

$$\lambda_1 X^2 + \lambda_2 Y^2 + \frac{|D|}{|C|} = 0 \tag{3}$$

现在开始解决问题。可以将（3）式改写成标准形式

$$\frac{X^2}{a_{\lambda 1}} + \frac{X^2}{a_{\lambda 2}} = 1 \tag{4}$$

其中 $a_{\lambda 1} = -\dfrac{\lambda_1 |D|}{|C|},\ a_{\lambda 2} = -\dfrac{\lambda_2 |D|}{|C|}$。从解析几何不难知道，$\sqrt{a_{\lambda 1}}$ 为其长半轴的长度，$\sqrt{a_{\lambda 2}}$ 为其短半轴的长度。那么半焦距可以通过公式

$$c = \sqrt{1 - \frac{\min\{a_{\lambda 1},\ a_{\lambda 2}\}}{\max\{a_{\lambda 1},\ a_{\lambda 2}\}}} \tag{5}$$

求出。进一步地，通过几何直观，不难知道近日点距离 d_m 和远日点距 d_M 可以由以下公式求出。

$$d_m = \sqrt{\max\{a_{\lambda 1},\ a_{\lambda 2}\}} - c$$

$$d_M = \sqrt{\max\{a_{\lambda 1},\ a_{\lambda 2}\}} + c$$

5.3 第一部分：第三小问

本质上，求近日点坐标和求远日点坐标是求解：满足点到轨迹方程上距离地最小值和最大值。建立下面的条件优化问题

$$\begin{cases} \min\ /\ \max\ (x-x_0)^2 + (y-y_0)^2 \\ \text{s.t. } a_1x^2 + 2a_2xy + a_3y^2 + 2a_4x + 2a_5y + 1 = 0 \end{cases} \tag{6}$$

这是一个条件极值问题，构造拉格朗日函数：

$$\mathcal{L} = (x-x_0)^2 + (y-y_0)^2 + \lambda(a_1x^2 + 2a_2xy + a_3y^2 + 2a_4x + 2a_5y + 1) \tag{7}$$

求导得方程组：

$$\begin{cases} 2(x-x_0) + \lambda(2a_1x + 2a_2y + 2a_4) & = 0 \\ 2(y-y_0) + \lambda(a_2x + 2a_3y + a_5) & = 0 \\ a_1x^2 + 2a_2xy + a_3y^2 + 2a_4x + 2a_5y + 1 & = 0 \end{cases} \tag{8}$$

这个方程组是一个四元二次方程组，意味着对于对偶 $(x,\ y)$ 会有两个结果。其分别对应着原优化极值中的最大值和最小值，也就是远日点坐标和近日点坐标。

最终求得

$$\text{近日点坐标} = (-0.5496,\ -0.4894)$$

$$\text{远日点坐标} = (-5.9935,\ 5.4469)$$

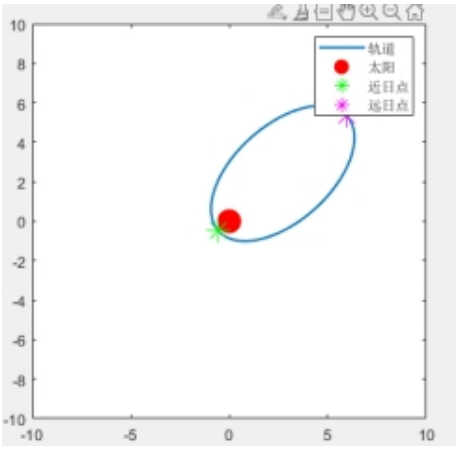

图 2: 参数曲线的图像

接下来考虑求解椭圆轨道的周长，通过 5.2 已经知道了椭圆的长半轴、短半轴和离心率，那么可以通过一定的符号运算，得到椭圆的周期计算公式

$$L = 4a \int_0^{\pi/2} \sqrt{1 - e^2 \sin^2 \theta}\, d\theta = 4aE(e) \tag{9}$$

很明显，这个函数是个椭圆积分，无法使用常规手段得到原函数，进而使用牛顿-莱布尼茨公式算出具体的积分值，这里使用数值方法计算求得椭圆周长为

$$L = 12.6371AU \tag{10}$$

5.4 第一部分：第四小问

二次曲线的平移变换分为平移变换和旋转变换

首先引入旋转变换

$$\begin{cases} x = x' \cos\alpha - y' \sin\alpha \\ y = x' \sin\alpha + y' \cos\alpha \end{cases} \tag{11}$$

其中 α 是将图像顺时针旋转的角度。很明显这个变换对应一个行列式为 1 的矩阵，也就是说

$$\begin{vmatrix} \cos\alpha & -\sin\alpha \\ -\sin\alpha & \cos\alpha \end{vmatrix} = 1 \tag{12}$$

换句话说，这个变换是一个单位变换，是不变量的一种这个这个可以将二次曲线的方程转化成

$$a_1' x'^2 + 2a_2' x' y' + a_3' y'^2 + 2a_4' x' + 2a_5' y' + 1 = 0 \tag{13}$$

转化后方程的常数项分别为

$$\begin{cases} a_1' = a_1 \cos^2\alpha + 2a_2 \sin\alpha\cos\alpha + a_3 \sin^2\alpha \\ a_2' = (a_3 - a_1)\sin\alpha\cos\alpha + a_3(\cos^2\alpha - \sin^2\alpha) \\ a_3' = a_1 \sin^2\alpha - 2a_2 \sin\alpha\cos\alpha + a_3 \cos^2\alpha \\ a_4' = a_4 \cos\alpha + a_5 \sin\alpha \\ a_5' = -a_4 \sin\alpha + a_5 \cos\alpha \end{cases} \tag{14}$$

由于首要目的是消去方程的交叉项，以此我们令 $a_3' = 0$，即

$$a_3' = a_1 \sin^2\alpha - 2a_2 \sin\alpha\cos\alpha + a_3 \cos^2\alpha = 0 \tag{15}$$

经过一系列的推导不难得到下面关于 $\tan\alpha$ 的一元二次方程

$$a_2 \tan\alpha + (a_1 - a_3)\tan\alpha - a_2 = 0 \tag{16}$$

由此得到旋转角度的正切值 $\tan\alpha$ 由于是顺时针转动，故（4）的根取正值。

接下来再根据三角变换公式

$$\sin\alpha=\frac{\tan\alpha}{\sqrt{1+\tan^2\alpha}},\quad \cos\alpha=\frac{1}{\sqrt{1+\tan^2\alpha}} \tag{17}$$

得到旋转变换中所需要的三角函数值。

现在我们已经使得交叉项为零，且由于二次曲线的长轴是与 x 轴对齐的，所以方程（4）中关于 y 的项一定为 0。那么，我们就得到化简后的（4）：

$$a_1'x'^2+a_3'y'^2+2a_4'x'+1=0 \tag{18}$$

进一步考虑平移。将（18）进行配方得到

$$a_1'(x'+\frac{a_4'}{a_1'})^2+a_3'y'^2=\frac{a_4'^2}{a_1'}-1 \tag{19}$$

取平移变换

$$\begin{cases}x''=x'+\dfrac{a_4'}{a_1'}\\ y''=y'\end{cases} \tag{20}$$

得到只关于二次项的曲线方程

$$a_1'x''^2+a_3'y''^2=\frac{a_4'^2}{a_1'}-1 \tag{21}$$

接下来只需要简单的归一化就可以将方程化为标准方程

$$a_1''x''^2+a_3''y''^2=1 \tag{22}$$

其中 $a_1''=\dfrac{a_1'}{\dfrac{a_4'^2}{a_1'}-1},\quad a_3''=\dfrac{a_3'}{\dfrac{a_4'^2}{a_1'}-1}$ 现在再使用参数变换即得

$$\begin{cases}x'=\sqrt{\dfrac{\dfrac{a_4'^2}{a_1'}-1}{a_1'}}\cos t-\dfrac{a_4'}{a_1'}\\ y'=\sqrt{\dfrac{\dfrac{a_4'^2}{a_1'}-1}{a_3'}}\sin t\end{cases}\qquad t\in[0,\ 2\pi] \tag{23}$$

现在，再使用旋转公式就可以得到原二次曲线方程的参数形式，也就是

$$F:\begin{cases}x=x'\cos\alpha-y'\sin\alpha\\ y=x'\sin\alpha+y'\cos\alpha\end{cases} \tag{24}$$

式（24）中的三角项由式（17）决定，变换项由式（23）决定。至此，我们已经把问题所需要的理论依据阐述完毕，现在开始具体的算法实现。

结合第一问中得到的参数向量 X，将其中的第 1，2，3 项代入式子（17）中得到旋转角度的正

切值

$$\tan\alpha = 0.8905 \tag{25}$$

再根据三角变换得，旋转所需要的三角函数值

$$\sin\alpha = 0.6651, \quad \cos\alpha = 0.7468 \tag{26}$$

现在根据旋转公式得到变换后的常数项：

$$X' = [-0.1694 \;\; 0 \;\; -0.5502 \;\; 0.6171 \;\; 0] \tag{27}$$

接下来根据 2.3.1 的固定流程不难得到关于参数 t 的参数方程

$$F: \begin{cases} x = x'\cos\alpha - y'\sin\alpha \\ y = x'\sin\alpha + y'\cos\alpha \end{cases} \tag{28}$$

这就是 set 函数所需要的两个向量.

另外，为了检验是否正确，使用 plot 函数以分别一上面两个向量组为横坐标和纵坐标，得到图像

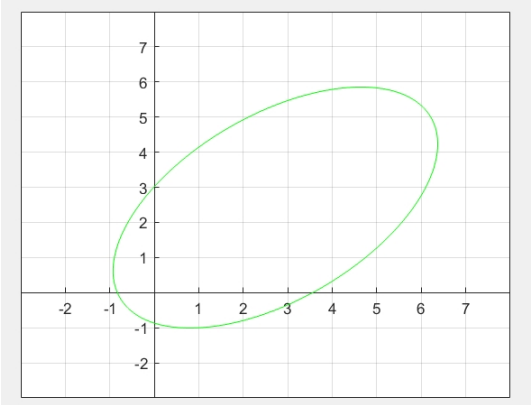

图 3: 参数曲线的图像

发现和其图 2 完全一致，这证明上述思路正确，同时也再次证明了二次曲线确实可以用单参数表示。

现在已经完全满足了做出动图的条件，使用 for 循环和 drawdnow 命令即可画出动态图（见文件夹附件 point_animation.gif）

5.5 第二部分：理论支撑

对于此部分需要引入两个三维欧氏空间的定理，以支撑后续运算和证明。

定理 1 在三维欧氏空间中去定笛卡尔直角坐标系后，空间中的一条曲线可以表示为三个一元函数：

$$x = x(t), \quad y = y(t), \quad z = z(t) \tag{29}$$

定理 2 三维欧氏空间中存在等距变换，可以将一个笛卡尔坐标系变成另一个笛卡尔坐标系。

定理 1 指出，空间中的曲线可以被三个一元函数。也就是对于三维坐标系，每一个维度都可以转化成一个一元函数来表示。换句话说，太阳系中行星绕椭圆轨道的三元运动可以转化成以时间为变量的单变量运动，这为使用 set 函数来标点提供了理论支撑。

定理 2 指出，空间中存在等距变换，可以将处于一个坐标系的向量变换成另一个坐标系的向量。也就是说，空间中的三维数据可以转化成平面上的二维数据。放到此题上，就是将以 $o-xyz$ 的三维数据转化成了在 xoy 的平面数据（换句话说，将数据所在平面变换到坐标平面 xoy 上）。后者可以使用第一部分：第四小题的理论将轨道简化，进而参数化。

5.6 第二部分：等距变换

现在找出符合定理 2 的变换，并找出其矩阵形式。首先考虑平面方向向量

$$A = [\eta_1, \eta_2,\ \eta_3] \tag{30}$$

不难知道此向量的经纬度所对应的正切值：

$$\tan\theta = \frac{\eta_1}{\eta_2} \tag{31}$$

$$\tan\phi = \frac{\eta_3}{\sqrt{\eta_1^2 + \eta_2^2}} \tag{32}$$

其中的 θ 和 ϕ 分别是平面角和天顶角。

现在再次点明变换的目的：将向量变换到 xoy 平面上，这意味着要使向量的平面角 θ 的余弦值为零，这样就消去了 x 方向的值。向量变成了在 yoz 平面上的向量，接下来只需要将向量旋转至天顶角 ϕ 为零，这样变换得到的方向向量对应的平面就应该是和 xoy 平行的。放在行星运动轨道上来开，此平面应该是过原点的，也就是说 xoy 应该与其平行

接下来考虑向量在平面 xoy 上的旋转，也就是绕 z 轴的旋转。将向量投影到 xoy 平面得到投影向量

$$A' = [\eta_1',\ \eta_2',\ 0] \tag{33}$$

其平面角的正切值仍然是

$$\tan\theta = \frac{\eta_1}{\eta_2} \tag{34}$$

正好，旋转变换已经在 5.4 部分指出并且已经证明了是单位变换，也就是等距变换。直接写出变换所对应的矩阵

$$\mathcal{A}: \begin{bmatrix} \cos\theta & -\sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{35}$$

回到三维情况，现在遇到问题：如何将上面的旋转变换延拓到三维情况并且不干扰 z 轴坐标且还是等距变换？事实上，只要保证 z 坐标不为零即可，也就是说，变换后各坐标应该满足下面的方程组

$$\begin{cases} x' = x\cos\theta - y\sin\theta \\ y' = x\sin\theta + y\cos\theta \\ z' = z \end{cases} \tag{36}$$

此方程对应的矩阵是

$$\mathcal{A}\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{37}$$

不难知道上面矩阵的行列式也是 1，这就意味着变换 $\mathcal{A}$ 不仅是保 z 的旋转变换，也是等距变换。这样我们就成功完成了第一步：消去了 x 方向的值；

同理，不难知道如果要得到绕 x 轴的旋转变换以消去 y，可以得到其变换所对应的矩阵

$$\mathcal{B}:\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\phi & -\sin\phi\\ 0 & -\sin\phi & \cos\phi\end{bmatrix} \tag{38}$$

这个矩阵的行列式也是 1，同样意味着这是等距变换。如此一来，我们就成功得到了将任意向量的旋转变换：将平面旋转至与 xoy 平面平行或相等。将上面两个变换复合即可得到变换

$$\mathcal{T}=\mathcal{B}\cdot\mathcal{A} \tag{39}$$

由于参与复合的变换所对应的矩阵行列式都是 1，且 Rank($\mathcal{T}$)=3，所以变换 $\mathcal{T}$ 的行列式也是 1，其也是等距变换。

现在不妨再次说明为什么要找出等距变换。我们知道，行星是在椭圆轨道上运动的，这就意味着其轨道的挠度为 0，换句话说就是其运动数据应该都在一个平面上的（尽管有垂直误差，但不影响讨论），这个平面对应着一个方向向量。等距变换可以将这个向量变换至其与 xoy 平面垂直，也就是使这个平面与 xoy 平行了，这样，原三维数据就可以移动到平面上来，可以进一步使用 5.4 的内容简化并参数化方程了。再加上原数据所在的平面应该是过原点的，这样就省去了平移变换了。

5.7 第二部分：数据处理

通过美国国家航天局 NASA 的 app：HORIZEN 查询到太阳系中各行星的运动数据，记录在文件 data.xlsx 中。由于原数据已经经过天文单位处理，这里不再对其进行放缩。前面说到过，这些数据应该是在一个平面上的，现在着手使用这些数据得到最佳符合这些数据的平面。

考虑到数据点和平面可能存在垂直误差，因此，这里使用最小二乘法求得最佳拟合平面。首先从原数据中选取相聚较远的 N 个点，使其平面坐标满足

$$Ax+By+Cz+D=0 \tag{40}$$

很明显，这里需要满足的应该是对于 z 坐标有最小的残差值，不妨将上面的方程进行一定的移项和变换得到

$$z=a_1x+a_2y+a_3 \tag{41}$$

其中 $a_1=-\dfrac{A}{C},\ a_2=-\dfrac{B}{C},\ a_3=\dfrac{D}{C}$，其所对应的方向向量应该是

$$D=[a_0,\ a_2,\ -1] \tag{42}$$

将选出的 N 个点的平面坐标代入上面方程，同时要使残差值最小，这样就可以得到下面的方程

$$E(z)=\sum_{i=1}^{N}(a_1x_i+a_2y_i+a_3-z_i)^2 \tag{43}$$

这是一个二元函数，要求其最小值。将各项求偏导后得到方程组

$$\begin{cases} a_1\sum\limits_{i=1}^{N}x_i^2+a_2\sum\limits_{i=1}^{N}x_iy_i+a_3\sum\limits_{i=1}^{N}x_i=\sum\limits_{i=1}^{N}x_iz_i \\ a_1\sum\limits_{i=1}^{N}x_iy_i+a_2\sum\limits_{i=1}^{N}y_i^2+a_3\sum\limits_{i=1}^{N}y_i=\sum\limits_{i=1}^{N}x_iy_i \\ a_1\sum\limits_{i=1}^{N}x_i+a_2\sum\limits_{i=1}^{N}y_i+Na_3=\sum\limits_{i=1}^{N}z_i \end{cases} \tag{44}$$

使用 matlab 和相关算法可以求出 $a_0,\ a_1$，进而求出最佳拟合平面的方向向量 $D=[a_0,\ a_2,\ -1]$。

现在可以通过等距变换 $\mathcal{T}$ 将上面的方向向量，外加相关的数据点变换到 xoy 平面，这里以火星为例，使用 data.xlsx 中的数据可以得到拟合后的方向向量

$$D=[-0.024564896919102,\ 0.020967417504512,\ -1.000000000000000] \tag{45}$$

通过旋转变换后得到方向向量

$$D'=[0,\ 0,\ 1.000521397451076] \tag{46}$$

相关数据点。变换前数据矩阵为

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}=\begin{bmatrix} 0.274174 & 1.116318 & 1.374819 & 0.938631 & 0.078716 \\ -1.401874 & -0.802101 & 0.210571 & 1.126938 & 1.559157 \\ -0.036192 & -0.044262 & -0.029361 & 0.000561 & 0.030731 \end{bmatrix} \tag{47}$$

变换后数据向量为

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}=\begin{bmatrix} -0.888273 & 0.114649 & 1.052715 & 1.466528 & 1.237005 \\ 1.119242 & 1.370528 & 0.909461 & -0.017707 & -0.952852 \\ 0.000063 & 0.000022 & 0.000004 & 0.000010 & 0.000026 \end{bmatrix} \tag{48}$$

可以看到变换后的 Z 轴数据基本可以忽略（均值小于 10^{-3}）。因此，这里直接使用前变换后的 X' 和 Y' 来做椭圆轨道的参数化处理，逆变化时使 Z' 为零向量即可。

通过上面的流程，使用 data.xlsx 中的数据，考虑到地球和土星的距离相差过大，会使得图形扭曲，这里先画出得太阳系前五个行星的轨道图：

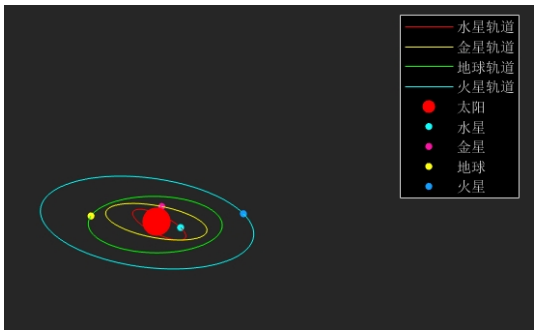

图 4: 参数曲线的图像

5.8 第二部分：做出动图

通过 for 循环和 set 函数以及 drawdown 命令，可以画出太阳系行星运动轨道动图。考虑到开普勒第三定律：行星的运动周期和轨道成正相关。但是根据计算，行星彼此间的运动之和太阳的质量又关。这里比较行星的运动轨道的长短，选出运动周期最大的行星，以此为计算的基础，得到每个行星的运动程度：也就是周期最大的行星运动一周，其里面行星运动多少圈符合以下关系

$$\text{偏内的行星运动圈数} = \left[\frac{\text{最外圈行星的运动周期}}{\text{此行星的运动周期}}\right] - 1 \tag{49}$$

其中分号的外部方框为取整函数。用上面的关系式，可以粗略地画出行星运动（见 point_animation.gif）。

6 模型优点

1. 拥有理论支撑，保证计算过程正确；
2. 计算快速，能够在短时间内得到行星运行结果；
3. 所需要的数据少，只需要五个相隔较远的点即可计算出结果。

7 模型缺点

1. 没有考虑同星系中其他行星对此行星的影响，没有考虑行星自带的卫星对行星轨道的作用；
2. 画动图时没有进一步考虑开普勒第二定律。

8 附录

使用软件：matlab 2023a；texworks

相关文件：data.xlsx，point_animation_3d.gif，point_animation_2d.gif

运行代码：

calculate.m

```
syms x y;

    A = -0.3378; B = 0.3784; C = -0.3818; D = 0.9218; E = 0.8208; F = 1;
    eq = @(x,y) A*x.^2 + B*x.*y + C*y.^2 + D*x + E*y + F;

    figure;
    h_plot = fimplicit(eq, [-10 10 -10 10], 'LineWidth', 1.5);
    axis equal;
    hold on;
    title('');

   M = [2*A, B; B, 2*C] \ -[D; E];
   h = M(1); k = M(2);

   theta = 0.5 * atan2(B, A - C);

   C_matrix = [A, B/2; B/2, C];
   lambda = eig(C_matrix);
   [lambda_sorted, ~] = sort(lambda, 'descend');
   K = (A*h^2 + C*k^2 + B*h*k + D*h + E*k + F);
   a = sqrt(-K / lambda_sorted(1));
   b = sqrt(-K / lambda_sorted(2));

   if a >= b
   c = sqrt(a^2 - b^2);
   f_rot = [c, 0; -c, 0];
   else
   c = sqrt(b^2 - a^2);
   f_rot = [0, c; 0, -c];
   end

   R = [cos(theta), -sin(theta); sin(theta), cos(theta)];
   f_original = R * f_rot' + [h; k];
   f1 = f_original(:,1)';
   f2 = f_original(:,2)';

   sun_position = f2;
   plot(sun_position(1), sun_position(2), 'ro', 'MarkerSize', 15, 'MarkerFaceColor'
       , 'r');
   text(sun_position(1), sun_position(2), ' ', 'FontSize', 12, 'Color', 'r');
```

data_mix.m

```
function [x,y,z,vector,I0,T]=data_mix(X,Y,Z,tau1,tau2)

A_1=curve_fit(X,Y,Z);
vector=[A_1(1),A_1(2),-1];

```

```
cos_theta=A_1(2)/sqrt(A_1(1)^2+A_1(2)^2);
sin_theta=A_1(1)/sqrt(A_1(1)^2+A_1(2)^2);
T_z=[cos_theta -sin_theta 0;sin_theta cos_theta 0; 0 0 1];
I=T_z*vector.';
cos_phi=I(3)/sqrt(vector*vector.');
sin_phi=I(2)/sqrt(vector*vector.');
T_x=[1 0 0; 0 cos_phi -sin_phi; 0 sin_phi cos_phi];
T=T_x*T_z;
I0=T*vector.';

ALL=[X;Y;Z];
ALL_T=T*ALL;
X=ALL_T(1,1:5);Y=ALL_T(2,1:5);
%处 理
A=asteroid(X,Y);

b=A(2);a=A(1);c=A(3);
co=[b,a-c,-b];
r=roots(co);t=r(2);
sin1= t/sqrt(1 +t^2);
cos1= 1/sqrt(1 +t^2);
A2=[A(1)*cos1^2+2*A(2)*sin1*cos1+A(3)*sin1^2
    0
    A(1)*sin1^2-2*A(2)*sin1*cos1+A(3)*cos1^2
    A(4)*cos1+A(5)*sin1
    -A(4)*sin1+A(5)*cos1];

t=0:0.03:2*pi*(tau2/tau1);
x1=cos(t)*sqrt((A2(4)^2/A2(1)+A2(5)^2/A2(3)-1)/A2(1))-A2(4)/A2(1);
y1=sin(t)*sqrt((A2(4)^2/A2(1)+A2(5)^2/A2(3)-1)/A2(3))-A2(5)/A2(3);

x=cos1*x1-y1*sin1;
y=sin1*x1+y1*cos1;
All1=inv(T)*[x;y;zeros(1,length(x1))];
x=All1(1,:);y=All1(2,:);z=All1(3,:);
```

asteroid_3d.m

```
function A=asteroid(X,Y,Z)
f=@(x,y,z)[x^2,2*x*y,y^2,2*x,2*y];
C=zeros(5,5);
for i=1:5
    C(i,:)=f(X(i),Y(i));
end
B=-ones(1,5);
A=B/C.';
end
```

problem.m

```
clc
format long

data_mercury = readtable('data.xlsx', 'Sheet', 'Sheet1', 'Range', 'A2:C16');
data_venus = readtable('data.xlsx', 'Sheet', 'Sheet1', 'Range', 'D2:F16');
data_earth = readtable('data.xlsx', 'Sheet', 'Sheet1', 'Range', 'G2:I16');
data_mars = readtable('data.xlsx', 'Sheet', 'Sheet1', 'Range', 'J2:L16');
data_jupiter = readtable('data.xlsx', 'Sheet', 'Sheet1', 'Range', 'M2:O16');


X_mercury=data_mercury.Var1(1:9).';Y_mercury=data_mercury.Var2(1:9).';
Z_mercury=data_mercury.Var3(1:9).';
X_venus=data_venus.Var1(1:9).';Y_venus=data_venus.Var2(1:9).';
Z_venus=data_venus.Var3(1:9).';
X_earth=data_earth.Var1(1:9).';Y_earth=data_earth.Var2(1:9).';
Z_earth=data_earth.Var3(1:9).';
X_mars=data_mars.Var1(1:9).';Y_mars=data_mars.Var2(1:9).';
Z_mars=data_mars.Var3(1:9).';
X_jupiter=data_jupiter.Var1(1:9).';Y_jupiter=data_jupiter.Var2(1:9).';
Z_jupiter=data_jupiter.Var3(1:9).';

T_max=686.98;
[x1,y1,z1]=data_mix(X_mercury,Y_mercury,Z_mercury,87.7,T_max);
[x2,y2,z2]=data_mix(X_venus,Y_venus,Z_venus,224.701,T_max);
[x3,y3,z3]=data_mix(X_earth,Y_earth,Z_earth,365,T_max);
[x4,y4,z4,vector,I0,T]=data_mix(X_mars,Y_mars,Z_mars,686.98,T_max);
[x5,y5,z5]=data_mix(X_jupiter,Y_jupiter,Z_jupiter,87.7,T_max);
x_cell = {x1, x2, x3, x4};
step=zeros(1,4);
for i=1:4
    step(i)=floor(length(x_cell{i})/length(x_cell{4}));
end

figure
colordef black
plot3(x1,y1,z1,"r")
hold on
plot3(x2,y2,z2,'y')
plot3(x3,y3,z3,'g')
plot3(x4,y4,z4,'c')
%plot3(x5,y5,z5,'m')

plot3(0,0,0,'.', 'MarkerSize', 60,Color="red");
legend({'水星轨道','金星轨道','地球轨道','火星轨道'})
xlabel('X(au)');ylabel('Y(au)');zlabel('Z(au)');
zlim([-0.5,0.5])
axis off

```

```
h1= plot3(x1(1), y1(1),z1(1), '.', 'MarkerSize', 15, 'MarkerFaceColor', 'r');
h2= plot3(x2(1), y2(1),z2(1), '.', 'MarkerSize', 15, 'MarkerFaceColor', 'r');
h3= plot3(x3(1), y3(1),z3(1), '.', 'MarkerSize', 15, 'MarkerFaceColor', 'r');
h4= plot3(x4(1), y4(1),z4(1), '.', 'MarkerSize', 15, 'MarkerFaceColor', 'r');
legend({'水星轨道','金星轨道','地球轨道','火星轨道','太阳','水星','金星','地球','火
    星'})
filename = 'point_animation.gif';
for k = 1:length(x4)
    set(h1, 'XData', x1(k*step(1)), 'YData', y1(k*step(1)),'ZData',z1(k*step(1)));
    set(h2, 'XData', x2(k*step(2)), 'YData', y2(k*step(2)),'ZData',z2(k*step(2)));
    set(h3, 'XData', x3(k*step(3)), 'YData', y3(k*step(3)),'ZData',z3(k*step(3)));
    set(h4, 'XData', x4(k), 'YData', y4(k),'ZData',z4(k));
    drawnow;
    pause(0.0001);

    frame = getframe(gcf);
    im = frame2im(frame);
    [imind, cm] = rgb2ind(im, 256);
    if k == 1
       imwrite(imind, cm, filename, 'gif', 'Loopcount', inf, 'DelayTime', 0.1);
    else
       imwrite(imind, cm, filename, 'gif', 'WriteMode', 'append', 'DelayTime', 0.1)
           ;
    end
end
```

asteroid.m

```
function A=asteroid(X,Y)
f=@(x,y,z)[x^2,2*x*y,y^2,2*x,2*y];
C=zeros(5,5);
for i=1:5
    C(i,:)=f(X(i),Y(i));
end
B=-ones(1,5);
A=B/C.';
end
```

promblem_2d.m

```
clc
X=[4.5596 5.0816 5.5546 5.9636 6.2756];
Y=[0.8145 1.3685 1.9895 2.6925 3.5265];
Initial=[0 0];
A=asteroid(X,Y);

f=@(x,y)A(1)*x^2+2*A(2)*x*y+A(3)*y^2+2*A(4)*x+A(5)*2*y+1;

```

```
b=A(2);
a=A(1);
c=A(3);
co=[b,a-c,-b];
r=roots(co);
t=r(2);
sin1= t/sqrt(1 +t^2);
cos1= 1/sqrt(1 +t^2);
A2=[A(1)*cos1^2+2*A(2)*sin1*cos1+A(3)*sin1^2
    0
    A(1)*sin1^2-2*A(2)*sin1*cos1+A(3)*cos1^2
    A(4)*cos1+A(5)*sin1
    -A(4)*sin1+A(5)*cos1];

t=0:0.01:2*pi;
x1=cos(t)*sqrt((A2(4)^2/A2(1)+A2(5)^2/A2(3)-1)/A2(1))-A2(4)/A2(1);
y1=sin(t)*sqrt((A2(4)^2/A2(1)+A2(5)^2/A2(3)-1)/A2(3))-A2(5)/A2(3);

x=cos1*x1-y1*sin1;
y=sin1*x1+y1*cos1;

figure;
fimplicit(f, [-3, 8, -3, 8], 'black-',LineWidth=0.2);
grid on
hold on
plot(X,Y,'.', 'MarkerSize', 15,Color="blue")
p=plot(0,0,'.', 'MarkerSize', 60,Color="red");
text(0.5,0.5,'太阳','FontSize',12,'Color',"red")
h= plot(x(1), y(1), '*', 'MarkerSize', 15, 'MarkerFaceColor', '#77AC30');
legend("小行星",'数据点')

filename = 'point_animation.gif';
for k = 1:length(t)
    set(h, 'XData', x(k), 'YData', y(k));
    drawnow;
    pause(0.0001);

    %frame = getframe(gcf);
    %im = frame2im(frame);
    %[imind, cm] = rgb2ind(im, 256);
    %if k == 1
    %    imwrite(imind, cm, filename, 'gif', 'Loopcount', inf, 'DelayTime', 0.1);
    %else
    %    imwrite(imind, cm, filename, 'gif', 'WriteMode', 'append', 'DelayTime',
        0.1);
    %end
end
```