Предисловие

Анализ опыта преподавания физики в высших учебных заведениях показывает, что до настоящего времени преподавание большинства курсов физических дисциплин остается «классическим». Оно базируется на «трех китах»: теоретическом курсе, излагаемом в виде лекций; семинарских занятиях, на которых проводится решение задач без использования ПК; лабораторных занятиях. Как следствие, не удается сформировать у обучаемых целый ряд умений, в том числе: умения дать физически правильную формулировку поставленной задачи; умения обосновать выбор математического аппарата и создать математическую модель изучаемого процесса; умения разработки и отладки компьютерной реализации математической модели; умения проводить анализ и оценивать адекватность получаемых результатов.

Поскольку в настоящее время большинство высших учебных заведений оснащены ПК, возможности которых позволяют решать задачи, требующие большого объема вычислений, назрела необходимость пересмотра подходов к использованию ПК в преподавании физики.

Автор считает, что выход из создавшегося положения состоит в дополнении «трех китов», на которых основана методика преподавания физики, соответствующими курсами лабораторных работ по компьютерному моделированию. Это, во-первых, позволит внедрить принципы компьютерного мышления в изучение физики. Во-вторых, потребует от студента более глубокого проникновения в суть изучаемой проблемы, будет способствовать закреплению физического материала и развитию физической интуиции. Первые шаги в развитии именно такого подхода сделаны в [1, 2]. Здесь удалось отойти от традиционных для книг, посвященных численному моделированию, подходов и на примерах решения широкого класса физических задач показать возможности использования ПК как экспериментальной установки для проведения физических опытов.

Не ставя под сомнение высокий научно-методической уровень этих книг и их актуальность, необходимо отметить, что они ориентированы на возможности ПК десятилетней давности (в первую очередь это относится к программному обеспечению). Учитывая современное состояние аппаратных и программных средств ПК, по нашему мнению, более разумно ориентироваться на специализированные пакеты для математических вычислений, одним из которых является пакет MATLAB, что позволяет экономить время, необходимое на разработку и отладку программы, в пользу анализа физического содержания решаемой задачи.

Наш выбор из достаточно большого количества специализированных пакетов для математических вычислений в качестве базового программного средст-

ва MATLAB обусловлен тем, что:

- в пакет интегрирован мощный математический аппарат, позволяющий решать сложные задачи без вызова внешних процедур, который позволяет находить решения:
  - линейных и нелинейных алгебраических уравнений и систем;
  - задачи Коши и краевой задачи для дифференциальных уравнений;
  - дифференциальных уравнений в частных производных;
  - задач статистической обработки данных (вычисление статистических параметров, интерполяция, аппроксимация, сглаживание и т. д.);
  - задач линейной алгебры (операции с матрицами и векторами);
  - задач поиска экстремумов функциональных зависимостей;
- пакет имеет мощные средства графического представления информации (функции, зависящие от одной переменной, полярные графики, графики поверхностей, карты линий уровня, векторные поля и т. д.);
- система снабжена средствами анимации, что позволяет рассматривать временную эволюцию математических моделей в динамике и т. д.;
- в пакет интегрирован математический аппарат, реализующий символьные вычисления.

Примеры физических систем и процессов, рассмотренных в этой книге, требуют, с одной стороны, привлечения минимального набора численных методов и представляют несомненный физический интерес, с другой. Приоритет физического содержания задачи определил структуру изложения материала книги: физическая постановка задачи, обсуждение численных методов и алгоритмов, необходимых для ее решения, описание функций и последовательностей команд, позволяющих реализовать данные методы в MATLAB, задачи для самостоятельной работы. Отметим, что книга не является учебником по курсу «Численные методы», поэтому мы не даем в ней строгое с математической точки зрения обоснование применяемым численным методам, но ограничиваемся обсуждением вопросов их практического использования.

При написании этой книги, равно как и любой другой, в которой описывается использование какого-либо математического пакета для решения конкретных задач, перед автором стояла достаточно сложная проблема, связанная с необходимостью одновременного изложения конкретных математических методов и соответствующих приемов работы с математическим пакетом. Здесь был использован подход «от задачи», в котором автор ограничился краткими описаниями основных приемов работы с MATLAB, помещенными в приложения, а в ходе решения конкретной задачи сообщает необходимые дополнительные сведения по работе с пакетом. Данный подход, как показывает личный опыт автора, оказывается более эффективным в отличие от подхода, основанном на изучении различных справочников без привязки к конкретной задаче.

Отметим, что высказанная точка зрения ни в коей мере не ставит под сомнение необходимость издания соответствующей справочной и учебной литературы по MATLAB. Однако при ее выборе читателю следует проявлять определенную осторожность, так как в ряде случаев включенный в них материал

оказывается плохо отредактированным, а в некоторых местах неотредактированным, а потому лишенным всякого смысла вариантом электронного перевода на русский язык англоязычной документации MATLAB. Примером подобной литературы является книга [3], по мнению ее авторов являющаяся «одновременно научной монографией по математической системе MATLAB и руководством пользователя» [3, с. 23], которая, однако, никоим образом не ответствует требованиям, предъявляемым к научным монографиям. Кроме того, практически все примеры использования функций и программ, приводимые авторами [3], заимствованы (при этом некоторые из них с ошибками) из фирменной документации MATLAB. В этой связи представляется уместным посоветовать пользователям математическим пакетам при выборе соответствующей литературы помнить бессмертный афоризм Козьмы Пруткова: «Бди!».

При написании данной книги автор, разумеется, опирался на содержимое учебного пособия «Компьютерное моделирование физических процессов в пакете MATLAB» [4, 5], которое выдержало два издания (в 2003 и 2011 годах).

Приближение юбилейной даты (10 лет со времени последнего издания) всегда является поводом для подведения некоторых итогов. В этой связи автор провел анализ динамики цитирований учебного пособия в системе учета индекса научного цитирования РИНЦ. Его результаты показали, что количество работ, авторы которых использовали и сослались на первое или второе издание, с течением времени не уменьшается. Таким образом, жизнь и первого, и второго изданий учебного пособия продолжается. Они по-прежнему представляют интерес для своего читателя. С этой точки зрения можно считать его судьбу достаточно успешной.

В то же время понятно, что охватить в одной книге все множество известных компьютерных моделей физических процессов и систем невозможно (здесь уместно напомнить еще один афоризм Козьмы Пруткова: «Никто не обнимет необъятного!»). В течение нескольких последних лет автор размышлял о целесообразности дальнейшей работы над данным пособием с целью включения в нее моделей тех физических процессов и систем, которые не были рассмотрены в предыдущих изданиях, но перейти от умозрительных рассуждений о целесообразности доработки учебного пособия к практическим действиям у автора физически не хватало необходимого для этого времени. Однако в условиях самоизоляции во время эпидемии COVID-19 возможность реализовать задуманное неожиданно появилась, чем автор и воспользовался.

В процессе доработки учебного пособия автор, во-первых, провел ревизию второго издания учебного пособия «Компьютерное моделирование физических процессов в пакете MATLAB» и исправил досадные опечатки, которые, в прочем, не влияли на работоспособность программного кода, представленного в книге. Во-вторых, автор включил в пособие ряд новых задач, связанных с описанием относительных движений и движения яхты в направлении, противоположном направлению ветра, а также ряд моделей физических систем, в том числе:

- модель, описывающая динамику свободных колебаний линейной системы, состоящей из связанных друг с другом линейными и нелинейными пружинками шариков (модель Паста–Ферми–Улама);
- модели, описывающие динамику вынужденных колебаний линейной системы, состоящей из связанных друг с другом линейными и нелинейными пружинками шариков, и привел поучительный пример, демонстрирующий, что возбуждение распределенной колебательной системы внешней периодической силой, частота которой совпадает с одной из собственных частот колебательной системы, отнюдь не означает нахождения в резонансе (линейного увеличения размаха колебаний) одновременно всех осцилляторов, а также анализом особенностей вынужденных колебаний цепочки связанных нелинейных осцилляторов;
- модель, описывающую движение солитонов (солитон — уединенное устойчивое во времени возмещение, распространяющееся, например, на поверхности канала с водой), присутствие которой в учебном пособии после рассмотрения нелинейных колебаний цепочек связанных осцилляторов и особенностей движения волновых пакетов в средах с дисперсией — основных механизмов, обеспечивающих существование солитона, представляется вполне логичным;
- модели случайных блужданий в одномерной ограниченной с одной стороны области рассеяния и одномерной ограниченной с двух сторон области рассеяния;
- модели множества Жюлюа и множества Мандельброта, являющихся самодобными объектами.

Во-третьих, автор включил в книгу новый материал, учитывающий, с его точки зрения, современное состояние науки, а также позволяющий ознакомить читателя с рядом новых результатов, полученных в компьютерном моделировании физических процессов и систем в последние годы, в том числе:

- результаты компьютерного моделирования распределения простых чисел и простых чисел-близнецов на скатерти Улама, визуализированной в полярной системе координат, ранее публиковавшиеся им только в соответствующих научных статьях;
- результаты компьютерного моделирования фрактального броуновского движения случайных блужданий в одномерной неограниченной области рассеяния, анализа алгоритмов генерации фрактального броуновского движения и методов оценки показателя Херста.

Анализ новой версии учебного пособия показал, что его объем в сравнении с предыдущей версией увеличился более чем на 30 %, что позволяет рассматривать его как новую книгу. Отмеченное обстоятельство также определило необходимость корректировки названия учебного пособия с целью более точного соответствия его содержанию, которое теперь называется «Компьютерное моделирование физических процессов и систем в пакете MATLAB».

Компьютерные реализации всех обсуждавшихся в книге m-файлов и m-функций можно скачать с сайта издательства http://www.techbook.ru.

В заключение необходимо отметить, что организация нетривиального использования компьютеров в процессе обучения физике далеко не простая задача, для решения которой потребуется не один год. Автор надеется, что настоящая книга будет определенным вкладом в решение этой задачи, и с благодарностью примет замечания, предложения и советы, которые можно направить по E-mail: sergey_porshnev@mail.ru.

1. Гулд Х., Тобочник Я. Компьютерное моделирование в физике: В 2-х частях. М.: Мир. 1990. Ч. 1.: 349 с., Ч. 2. 400 с.

2. Кунин С. Вычислительная физика. М.: Мир, 1992.

3. Дьяконов В.П., Абраменкова И.В., Круглов В.В. MATLAB с пакетами расширений. М.: Нолидж, 2001.

4. Поршнев С.В. Компьютерное моделирование физических процессов в пакете MATLAB. М.: Горячая линия — Телеком, 2003. 592 с.

5. Поршнев С.В. Компьютерное моделирование физических процессов в пакете MATLAB. 2-е издание. М.: Лань, 2011. 736 с.

1 Моделирование относительных движений в классической механике

В классической механике для описания кинематики движения материальных точек (тел, размерами которых можно пренебречь в сравнении с длиной их перемещений) в качестве первого шага следует выбрать соответствующую систему отсчета (СО). В зависимости от выбранной СО законы движения будут иметь, вообще говоря, различный вид, поэтому при использовании произвольной СО траектория движения материальной точки, вообще говоря, может выглядеть достаточно сложно. В этой связи при решении кинематических задач представляется целесообразным использовать персональный компьютер.

Для нахождения траекторий относительных движений в классической механике используется предположение об абсолютности времени во всех СО (как инерциальных, так и неинерциальных). Используя данное предположение, рассмотрим движение одной и той же материальной точки в двух различных СО K и K' считая, что СО K' движется относительно СО K с произвольной скоростью $\vec{V}(t) = \frac{d}{dt}\vec{R}(t)$ ($\vec{R}(t)$ — радиус-вектор, описывающий положение точки начала системы координат K' относительно СО K). Будем описывать движение точки в СО K' радиусом-вектором $\frac{d}{dt}\vec{r}(t)$ направленным из начала координат СО K' в текущее положение точки. Тогда движение рассматриваемой точки относительно СО K описывается радиусом-вектором

$$\vec{r}(t) = \vec{r}'(t) + \vec{R}(t), \tag{1.1}$$

а относительная скорость

$$\vec{v}(t) = \frac{d}{dt}\vec{r}(t) = \frac{d}{dt}\vec{r}'(t) + \frac{d}{dt}\vec{R}(t), \tag{1.2}$$

где $\frac{d}{dt}\vec{r}'(t)$ — скорость точки относительно СО K'; $\frac{d}{dt}\vec{R}(t)$ — скорость движения СО K' относительно СО K (рис. 1.1).

Таким образом, для нахождения закона движения точки в произвольной СО K необходимо:

- задать закон движения точки относительно СО K' (функцию $\vec{r}'(t)$);
- задать закон движения СО K' относительно СО K (функцию $\vec{R}(t)$);
- определить закон движения точки относительно СО K в соответствии с (1.1).

Возможна другая постановка данной задачи, в которой по известному закону движения материальной точки $\vec{r}(t)$ относительно некоторой СО K и по известному закону движения $\vec{R}(t)$ СО K' относительно СО K требуется найти

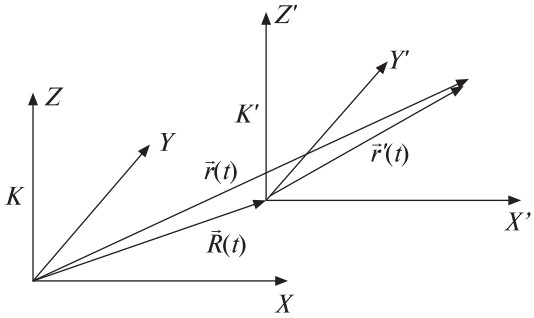

Рис. 1.1. К постановке задачи об описании относительных движений

закон движения $\vec{r}'(t)$ материальной точки в СО K'. Очевидно, что решение задачи в данной постановке дается следующими формулами:

$$\vec{r}'(t) = \vec{r}(t) - \vec{R}(t); \tag{1.3}$$

$$\vec{v}'(t) = \frac{d}{dt}\vec{r}(t) - \frac{d}{dt}\vec{R}(t). \tag{1.4}$$

В физике преобразования (1.1)–(1.4) называются преобразованием Галилея. Несмотря на то что решение задачи о нахождении траектории относительного движения исчерпывается формулами (1.1)–(1.4), построение конкретных траекторий движения без использования ПК может вызвать определенные трудности, а в ряде случаев привести к неправильным результатам. Ниже мы рассмотрим две поучительные, на наш взгляд, задачи, соответствующие обеим постановкам, и продемонстрируем в ходе их решения возможности, заложенные в MATLAB для построения графиков функций, зависящих от одной переменной.

1.1. Построение орбиты Луны в гелиоцентрической системе отсчета

С точки зрения кинематического подхода эта задача соответствует первой постановке задаче об относительном движении. В гелиоцентрической СО (K) Земля движется по окружности радиуса $R_1 = 1{,}496 \cdot 10^8$ км (период обращения $T_1 = 3{,}156 \cdot 10^7$ с). Луна в свою очередь движется вокруг Земли (K') по окружности радиуса $R_2 = 3{,}844 \cdot 10^5$ км (период обращения $T_2 = 2{,}36 \cdot 10^6$ с). Как известно [1, 2], при движении материальной точки по окружности радиуса R с постоянной угловой скоростью ω координаты радиуса-вектора, проведенного из начала координат к текущему положению точки, меняются по закону

$$\vec{R}(t) = \begin{pmatrix} R\cos(\omega t + \varphi_0) \\ R\sin(\omega t + \varphi_0) \end{pmatrix} = \begin{pmatrix} R\cos\left(\dfrac{2\pi}{T}t + \varphi_0\right) \\ R\sin\left(\dfrac{2\pi}{T}t + \varphi_0\right) \end{pmatrix}, \tag{1.5}$$

где φ_0 — начальная фаза, характеризующая положение частицы в момент времени $t = 0$, которую в дальнейшем мы будем полагать равной нулю.

Заменив в (1.5) R на R_1, R_2 и подставив в (1.1), получаем зависимость

радиуса-вектора Луны в гелиоцентрической системе координат от времени:

$$\vec{r}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} R_2 \cos\left(\dfrac{2\pi}{T_2}t\right) + R_1 \cos\left(\frac{2\pi}{T_1}t\right) \\ R_2 \sin\left(\dfrac{2\pi}{T_2}t\right) + R_1 \sin\left(\frac{2\pi}{T_1}t\right) \end{pmatrix}. \qquad (1.6)$$

Выражение (1.6) задает орбиту Луны $y = y(x(t))$ в параметрической форме, где параметром является время. Предваряя дальнейшее чтение раздела, где орбита Луны построена с использованием MATLAB, рекомендуем читателю самостоятельно качественно оценить ее вид.

Для построения искомой орбиты в MATLAB в режиме непосредственного вычисления необходимо передать пакету следующую последовательность команд, сохраненную в файле `Glava1_1.m`. (В MATLAB часть строки, расположенная после знака %, является комментарием и при самостоятельном вводе команд может быть опущена.)

```
% листинг файла Glava1_1.m
clear all
close all
R1=1.496*10^8;     % задание численного значения радиуса орбиты Земли
T1=3.156*10^7;     % задание численного значения периода
                   % обращения Земли вокруг Солнца
R2=3.844*10^5;     % задание численного значения радиуса орбиты Луны
T2=2.360*10^6;     % задание численного значения периода обращения Земли
t=0:T1/1000:T1;       % задание дискретной переменной,
                      % изменяющейся от 0 до T1 с шагом T1/1000
Xz=R1*cos(2*pi*t/T1); % вычисление x-й координаты радиуса-вектора Земли
Yz=R1*sin(2*pi*t/T1); % вычисление y-й координаты радиуса-вектора Земли
Xm=R2*cos(2*pi*t/T2); % вычисление x-й координаты радиуса-вектора Луны
                      % в системе координат, связанной с Землей
Ym=R2*sin(2*pi*t/T2); % вычисление y-й координаты радиуса-вектора Луны
                      % в СО, связанной с Землей
Xotn=Xz+Xm;           % вычисление x-й координаты радиуса-вектора Луны
                      % в гелиоцентрической СО
Yotn=Yz+Ym;           % вычисление y-й координаты радиуса-вектора Луны
                      % в гелиоцентрической СО
plot(Xotn,Yotn);      % визуализация орбиты Луны в гелиоцентрической СО
h = gca;              % загрузка дискриптора графического окна
 % задание режима отображения пропорциональных отрезков по обеим
 % координатным осям
set(h,'DataAspectRatioMode','manual');
```

Для отображения на одном графике орбиты Земли и орбиты Луны в гелиоцентрической СО следует вместо команды

```
plot(Xotn,Yont)
```

выполнить команду

```
plot(Xotn,Yont,Xz,Yz)
```

Фрагмент траекторий Земли и Луны в гелиоцентрической СО представлен на рис. 1.3.

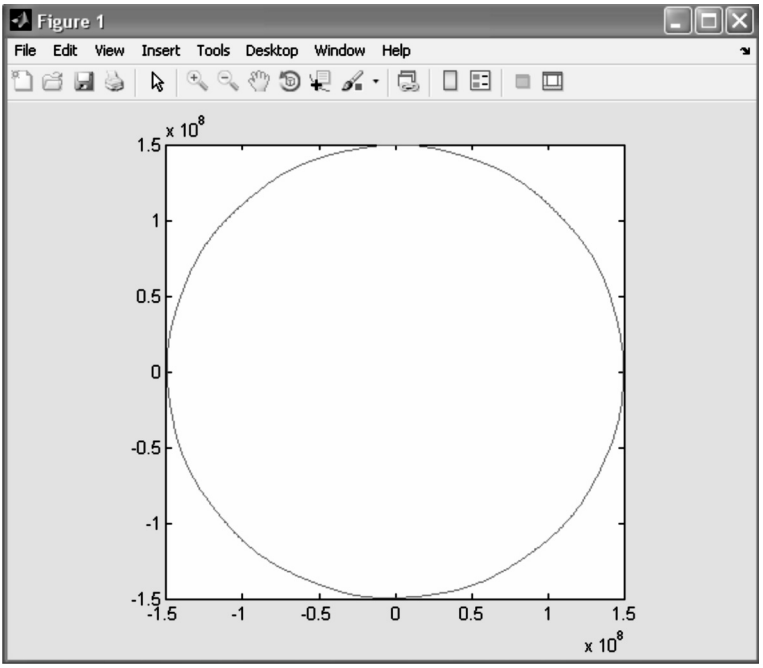

Рис. 1.2. Орбита Луны в гелиоцентрической СО. (Подробное описание интерфейса окна **Figure № 1** приведено в Приложении С)

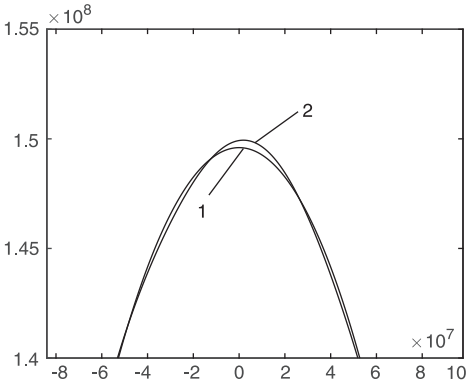

Рис. 1.3. Фрагмент траекторий движения Луны (1) и Земли (2) в гелиоцентрической СО

```
>> whos
  Name      Size          Bytes  Class          Attributes

  R1        1x1               8  double
  R2        1x1               8  double
  T1        1x1               8  double
  T2        1x1               8  double
  Xm        1x1001         8008  double
  Xotn      1x1001         8008  double
  Xz        1x1001         8008  double
  Ym        1x1001         8008  double
  Yotn      1x1001         8008  double
  Yz        1x1001         8008  double
  h         1x1               8  matlab.graphics.axis.Axes
  t         1x1001         8008  double
>>
```

Рис. 1.4. Получение информации об активных переменных, находящихся в памяти компьютера после выполнения приведенной последовательности команд

Вопрос: Соответствует ли вид орбиты, представленной на рис. 1.2, орбите, ранее построенной Вами ранее качественно?

Для получения информации о всех активных переменных, находящихся в памяти компьютера (рабочей области) в данный момент времени, используется команда `whos`. На рис. 1.4 показана информация, выведенная данной командой на монитор компьютера после выполнения приведенной выше последовательности команд.

Отметим, что постоянные в MATLAB трактуются как матрицы размерности 1×1. Для сохранения значений переменных, находящихся в рабочей области используется команда

```
>> save имя_файла
```

или функция

```
>> save(имя_файла)
```

Команда save имеет несколько различных форм:

`save имя_файла` — записываются все переменные рабочей области в файле «имя_файла» и расширением .m;

`save имя_файла X` — записывается только значение переменной X;

`save имя_файла X Y Z` — записываются только значения переменных X, Y и Z.

После записи команды save также можно указывать ключи, уточняющие формат записи файлов:

`-mat` — двоичный MAT-формат, используемый по умолчанию;

`-ascii` — ASCII формат единичной точности (8 цифр);

`-ascii -double` — ASCII формат двойной точности (16 цифр);

`-ascii -double -tabs` — ASCII формат двойной точности с разделителем и метками табуляции;

`V4` — запись MAT-файла в стандарте версии MATLAB 4.0;

`-append` — добавление в существующий MAT-файл.

Для загрузки ранее проведенного сеанса работы с MATLAB используется команда

```
load(имя_файла)
```

или функция

```
load(имя_файла)
```

Приведенный выше протокол команд можно сохранить в виде файла на диске для последующего анализа решения или использования как основы программы сценария, используемой при решении подобных задач, командой

```
>> diary имя_файла
```

или функцией

```
>> diary(имя_файла)
```

Например, для записи в файл `Moon.m` протокола приведенных выше команд необходимо ввести следующую команду:

```
>> diary Moon.m % открытие файла для сохранения протокола
```

Далее каждая последовательно выполненная команда будет заноситься в файл Moon.m. Для приостановки записи выполняемых команд в файл используется команда

```
>> diary off
```

Отметим, что данная команда также записывается в файл Moon.m. Данный набор команд, начиная с версии MATLAB 6.0 и выше, можно выполнить

автоматически, набрав в командной строке имя файла Moon и нажав клавишу <Enter>.

Созданный нами документ позволяет вычислять траекторий движения Луны в гелиоцентрической СО для различных значений радиуса орбиты Луны и периода ее обращения вокруг Земли и проводить их анализ.

Например, на рис. 1.5 представлена возможная орбита Луны в гелиоцентрической системе координат при $r = 3{,}844 \cdot 10^7$ км.

Сравнивая орбиты Луны, представленные на рис. 1.2 и 1.5, обнаруживаем их существенные отличия. Для объяснения причины этих отличий сравним линейные скорости движения Луны относительно Солнца в первом и во втором случае с линейной скоростью движения Земли относительно Солнца. Так как направление линейной скорости движения Земли относительно Солнца и направление линейной скорости движения Луны относительно Земли меняются во времени, оставаясь постоянными по величине, в качестве количественной характеристики соотношения линейных скоростей движения Луны в Земли в гелиоцентрической системе координат выберем разность между модулем линейной скорости движения Земли и проекцией линейной скорости Луны на направление вектора линейной скорости Земли:

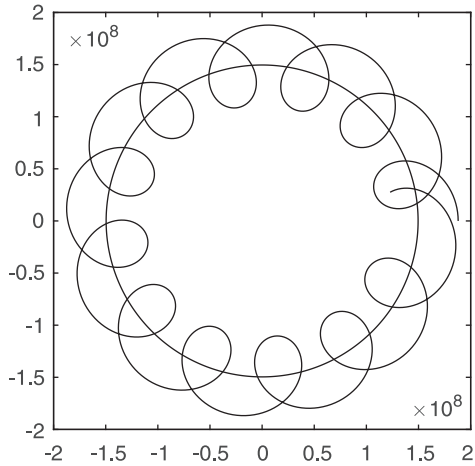

Рис. 1.5. Орбита Луны в гелиоцентрической СО при $R_2 = 3{,}844 \cdot 10^7$ км

$$v_{\text{отн}}(t) = |\vec{V}(t)| - \frac{(\vec{V}(t)\vec{v}(t))}{|\vec{V}(t)|}, \tag{1.7}$$

где $\vec{V}(t) = \begin{pmatrix} dX(t)/dt \\ dY(t)/dt \end{pmatrix}$ — вектор скорости движения Земли относительно Солнца; $\vec{v}(t) = \begin{pmatrix} dx(t)/dt \\ dy(t)/dt \end{pmatrix}$ — вектор скорости Луны относительно Земли.

Для визуализации искомой зависимости необходимо выполнить следующую последовательность команд, сохраненную в файле `Glava1_2.m`:

```
% листинг файла Glava1_2.m
clear all
close all
R1=1.496*10^8; % задание численного значения радиуса орбиты Земли
T1=3.156*10^7; % задание численного значения периода обращения Земли
               % вокруг Солнца
R2=3.844*10^5; % задание численного значения радиуса орбиты Луны
T2=2.360*10^6; % задание численного значения периода обращения Земли
dt=T1/2000;    % задание шага по времени T1/2000
t=0:dt:T1;     % задание дискретной переменной, изменяющейся от 0 до T1
```

```
Xz=R1*cos(2*pi*t/T1); % вычисление x-й координаты радиуса-вектора Земли
Yz=R1*sin(2*pi*t/T1); % вычисление y-й координаты радиуса-вектора Земли
Xm=R2*cos(2*pi*t/T2); % вычисление x-й координаты радиуса-вектора
                      % Луны в СО, связанной с Землей
Ym=R2*sin(2*pi*t/T2); % вычисление y-й координаты радиуса-вектора
                      % Луны в СО, связанной с Землей
Xotn=Xz+Xm;           % вычисление x-й координаты радиуса-вектора Луны
                      % в гелиоцентрической СО
Yotn=Yz+Ym;           % вычисление y-й координаты радиуса-вектора Луны
                      % в гелиоцентрической СО
Vx=diff(Xz)/dt; % вычисление значений проекции скорости движения
Vy=diff(Yz)/dt; % вычисление значений проекции скорости движения Земли
                % на ось oY в гелиоцентрической СО
vx=diff(Xm)/dt; % вычисление значений проекций скорости движения Луны
                % на ось oX в СО, связанной с Землей
vy=diff(Ym)/dt; % вычисление значений проекций скорости движения Луны
                % на ось oY в СО, связанной с Землей
% вычисление значений функции, задаваемой выражением (1.7)
V=(Vx.^2+Vy.^2).^0.5-(Vx.*vx+Vy.*vy)./(Vx.^2+Vy.^2).^0.5;
t1=0:dt:T1-dt;
% построение траектории движения Луны и траектории движения Земли
% в гелиоцентрической СО
subplot(2,1,1); plot(Xz,Yz,Xotn,Yotn);
% построение графика функции, задаваемой выражением (1.7)
subplot(2,1,2); plot(t1,V);
```

На рис. 1.6 ($R_2 = 3{,}844 \cdot 10^7$ км) и рис. 1.7 ($R_2 = 3{,}844 \cdot 10^5$ км) представлены результаты выполнения приведенной выше последовательности команд.

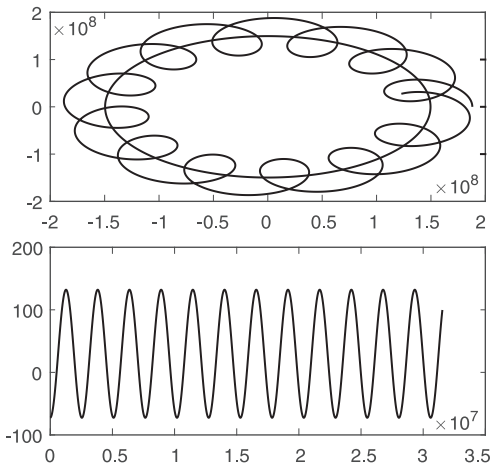

Рис. 1.6. Траектория движения Луны в гелиоцентрической СО при $R_2 = 3{,}844 \times$ $\times 10^7$ км (вверху); $v_{\text{отн}}(t)$ — зависимость мгновенных значений разностей между модулем скорости Земли и проекцией скорости движения Луны на направление скорости движения Земли (внизу)

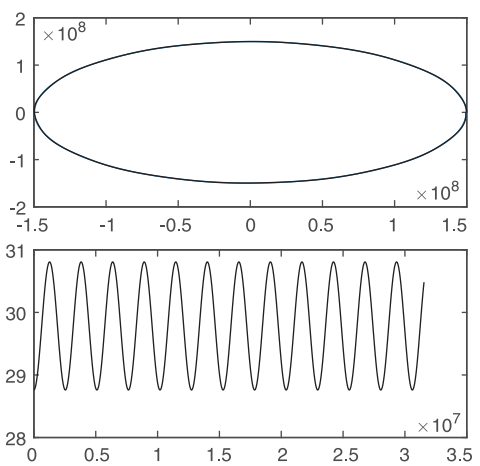

Рис. 1.7. Траектория движения Луны в гелиоцентрической СО при $R_2 = 3{,}844 \times$ $\times 10^5$ км (вверху); $v_{\text{отн}}(t)$ — зависимость мгновенных значений разностей между модулем скорости Земли и проекцией скорости движения Луны на направление скорости движения Земли (внизу)

Анализ зависимостей $v_{\text{отн}}(t)$, представленных на рис. 1.6 и 1.7, позволяет объяснить причину отличий орбит. Область изменения функции $v_{\text{отн}}(t)$ при $R_2 = 3{,}844 \cdot 10^5$ км всегда положительна. Это означает, что Луна всегда движется в направлении движения Земли, а потому петли отсутствуют. При $R_2 = 3{,}844 \cdot 10^7$ км функция $v_{\text{отн}}(t)$ принимает как положительные, так и отрицательные значения. Это означает, что существуют моменты времени, в которые Луна движется в направлении противоположном направлению движения Земли, а потому орбита имеет петли.

В заключение сделаем ряд замечаний по поводу некоторых операторов и функций, использованных в данном разделе. Обратите внимание на способ задания дискретных переменных (векторов) `t`, `t1`. Здесь мы использовали знак :, являющийся одним из важнейших синтаксических знаков MATLAB. Этот знак, будучи расположенным между двумя числами, задает вектор, компоненты которого принимают значения от меньшего числа до большего с шагом 1. Например, оператор `x = 0:9` задает целочисленный вектор `x =[0,1,2,3,4,5,6,7,8,9]`. Отметим, что в MATLAB также допускается явное задание вектора с помощью квадратных скобок. Если шаг изменения дискретной переменной отличается от 1, то при ее определении следует указывать значение шага аналогично тому, как это было сделано выше в программах, приведенных в настоящем разделе.

Так как MATLAB является пакетом, изначально ориентированным на матричные вычисления, основные объекты, с которыми оперирует пакет, являются векторами и матрицами. При обращении к функции, например `y=cos(x)`, где `x` — вектор, значение которого определено выше, MATLAB проводит вычисления для каждого элемента вектора аргумента и присваивает их соответствующим компонентам вновь создаваемого вектора.

Для построения двух графиков на одном чертеже были использованы команды `subplot(2,1,1)`, `subplot(2,1,2)`, позволившие разбить графическое окно на две отдельные части. В общем случае обращение к данной команде имеет следующий вид: `subplot(m,n,p)` или `subplot(m n p)`. Здесь значение `m` указывает, на сколько частей разбивается окно по вертикали, `n` указывает, на сколько частей окно разбивается по горизонтали, `p` — порядковый номер подокна, считая слева направо и сверху вниз. Команда `subplot( )` используется как для создания нового подокна, так и для перехода от одного подокна к другому. После вызова данной команды команда `plot( )` строит график и/или графики в соответствующем подокне.

Технология создания интерфейса пользователя, позволяющего упростить процедуру вычислений, описана в Приложении С.

Задача 1.1

1. Определите предельное значение радиуса орбиты Луны, при котором не происходит появление петель. Как выглядит орбита Луны в гелиоцентрической системе координат в этом случае?

2. Зафиксируйте радиус орбиты Луны и постройте орбиту Луны при различных значениях периода обращения. Что можно сказать о размере петель в этом случае?