IRSTI 29.17.15 https://doi.org/10.26577/RCPh202594310

M. Zhumagaliyev[1], K. Boshkayev[2*], A. Urazalina[2]
[1]King's College, The British School of Alicante, Gta. del Reino Unido, Alicante, Spain
[2]Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: kuantay@mail.ru

NUMERICAL SIMULATION OF THE KEPLER PROBLEM IN WOLFRAM MATHEMATICA

In this article, we consider the classical Kepler problem, emphasizing its fundamental features, implications, and broad applications in celestial mechanics and astrophysics. The Kepler problem, which describes the motion of bodies under the influence of Newton's law of universal gravitation, remains one of the cornerstones of classical mechanics and continues to play a crucial role in modern space science. For pedagogical and illustrative purposes, we employ Wolfram Mathematica to visualize a variety of examples of planetary motion within the Solar System. This computational approach allows us to highlight the interplay between theory and numerical simulations, offering students and researchers an intuitive way to explore orbital dynamics. The Kepler problem is formulated and solved numerically, and we demonstrate how variations in initial conditions - such as position and velocity - lead to distinct orbital trajectories. As representative cases, we analyze the motion of the Earth and selected artificial satellites under the gravitational field of the Sun, stressing the relevance of these simulations for practical applications, such as mission design and orbital prediction. In addition, we examine elliptical orbits in detail and numerically confirm the conservation of orbital angular momentum and total mechanical energy, thereby reinforcing key principles of mechanics. We also validate all three of Kepler's laws within our numerical framework, demonstrating their consistency with the trajectories derived from Newtonian gravity. The results obtained in this work are versatile and can be effectively applied at multiple educational levels. They are suitable for advanced high school programs in physics and astronomy, as well as for undergraduate laboratory courses in physics, astronomy, mathematics, and engineering. Beyond their pedagogical value, the presented methodology offers a simple yet powerful framework for initiating research-oriented activities. Overall, this study provides both educational and scientific insights, serving as a bridge between classical theory, computational methods, and modern applications in space science.

Keywords: physics education methods, gravity, solar system, Kepler problem, Kepler laws.

М. Жұмағалиев[1], Қ. Бошқаев[2*], А. Уразалина[2]
[1]Кинг колледжі, Аликанте Британ мектебі, Аликанте, Испания
[2]Әл-Фараби атындағы Қазақ ұлттық университеті, Әл-Фараби даңғылы, Алматы, Қазақстан
*e-mail: kuantay@mail.ru

Wolfram Mathematica бағдарламасында Кеплер есебін сандық модельдеуі

Бұл жұмыста біз классикалық Кеплер есебін қарастырып, оның іргелі ерекшеліктерін, салдарын және аспан механикасы мен астрофизикадағы кең қолданылуын атап өтеміз. Ньютонның бүкіләлемдік тартылыс заңы негізінде денелердің қозғалысын сипаттайтын Кеплер есебі классикалық механиканың негіздерінің бірі болып қала береді және қазіргі ғарыш ғылымында маңызды рөл атқарады. Педагогикалық және иллюстрациялық мақсаттарда біз Күн жүйесіндегі планеталардың әртүрлі қозғалыс траекторияларын бейнелеу үшін Wolfram Mathematica бағдарламасын қолданамыз. Мұндай есептеу тәсілі теория мен сандық модельдеудің байланысын ашып көрсетіп, студенттер мен зерттеушілерге орбиталық динамиканы зерделеудің көрнекі құралын ұсынады. Кеплер есебі тұжырымдалып, сандық түрде шешіледі және бастапқы шарттардың - мысалы, сынақ

денелердің орны мен жылдамдығын өзгерту - әртүрлі траекторияларға әкелетінін көрсетеміз. Көрнекті мысал ретінде Жердің және жасанды серіктердің Күннің гравитациялық өрісіндегі қозғалысы қарастырылады. Бұл модельдер ғарыштық миссияларды жобалау мен орбиталарды болжаудағы практикалық маңыздылығын дәлелдейді. Сонымен қатар эллипстік орбиталар егжей-тегжейлі талданып, орбиталық импульс моментінің және толық механикалық энергияның сақталуы сандық түрде расталады. Кеплердің үш заңы да сандық модельдеу аясында тексеріліп, олардың Ньютонның тартылыс теориясынан шығатын траекторияларымен сәйкестігі көрсетіледі. Алынған нәтижелер білім берудің әртүрлі деңгейлерінде қолданылуы мүмкін. Олар физика мен астрономияны тереңдетіп оқытатын мектеп бағдарламалары үшін де, сондай-ақ университеттердің физика, астрономия, математика және инженерлік пәндер бойынша курстары мен зертханалық сабақтары үшін де пайдалы. Педагогикалық маңызынан бөлек, ұсынылған әдістеме ғылыми-зерттеу қызметін бастаудың қарапайым, бірақ тиімді құралы бола алады. Жалпы, бұл зерттеу классикалық теорияны, есептеу әдістерін және қазіргі заманғы ғарыш ғылымындағы қолданбаларды біріктіреді.

Түйін сөздер: физиканы оқытудың әдістері, гравитация, Күн жүйесі, Кеплер есебі, Кеплер заңдары.

М. Жумагалиев[1], К. Бошкаев[2*], А. Уразалина[2]
[1]Кинг колледж, Британская школа Аликанте, Аликанте, Испания
[2]Казахский национальный университет имени аль-Фараби, Алматы, Казахстан
*e-mail: kuantay@mail.ru,

Численное моделирование задачи Кеплера в Wolfram Mathematica

В данной работе мы рассматриваем классическую задачу Кеплера, подчеркивая её фундаментальные особенности, следствия и широкие приложения в небесной механике и астрофизике. Задача Кеплера, описывающая движение тел под действием закона всемирного тяготения Ньютона, остаётся краеугольным камнем классической механики и продолжает играть важную роль в современной космической науке. В педагогических и иллюстративных целях мы используем Wolfram Mathematica для визуализации различных примеров движения планет в Солнечной системе. Такой вычислительный подход позволяет подчеркнуть связь между теорией и численным моделированием, предоставляя учащимся и исследователям наглядный инструмент для изучения орбитальной динамики. Задача Кеплера формулируется и решается численно, и мы показываем, как изменения начальных условий - таких как положение и скорость пробных тел - приводят к различным траекториям движения. В качестве наглядных примеров рассматривается движение Земли и отдельных искусственных спутников в гравитационном поле Солнца, что демонстрирует практическую значимость этих моделей для проектирования миссий и предсказания орбит. Кроме того, подробно анализируются эллиптические орбиты и численно подтверждается сохранение орбитального момента и полной механической энергии. Мы также проверяем все три закона Кеплера в рамках численного моделирования, показывая их согласованность с траекториями, вытекающими из ньютоновской теории тяготения. Полученные результаты могут применяться на разных уровнях образования. Они полезны как для углубленных школьных программ по физике и астрономии, так и для университетских курсов и лабораторных занятий по физике, астрономии, математике и инженерным дисциплинам. Помимо педагогической ценности, представленная методология служит простым, но эффективным инструментом для начала исследовательской деятельности. В целом данное исследование объединяет классическую теорию, вычислительные методы и современные приложения в космической науке.

Ключевые слова: методы физического образования, гравитация, солнечная система, задача Кеплера, законы Кеплера.

Introduction

The two-body problem is a classical and well-known problem in mechanics, describing the mutual gravitational motion of two celestial bodies of comparable masses (m1∼m2), both revolving around their common center of mass [1]. A special limiting case of this problem is the Kepler problem**,** where one body is assumed to be much more massive than the other ($m_1 = M \gg m_2 = m$) [2, 3]. In this case, the massive body acts as a fixed gravitational source whose motion is negligibly affected by the lighter body. The lighter one - often referred to as a *test body*, *test particle*, or simply *orbiting object* - moves under the influence of the central gravitational field. The Kepler problem provides an elegant and practical framework for describing the motion of planets, natural and artificial satellites, comets, and asteroids within the Solar System. It plays a fundamental role in celestial mechanics**,** astronomy, and space research [3].

Historically, the problem was formulated and solved by Isaac Newton (1643–1727) using his laws of mechanics, dynamics, and universal gravitation, building on the observational discoveries and empirical laws of Johannes Kepler (1571–1630). Through careful observation of planetary motions and extensive data analysis, Kepler derived his three fundamental laws of planetary motion, which remain cornerstones in physics education and are widely presented in high school and university textbooks [2, 3]. While Kepler's laws were strongly supported by observations, they lacked a deeper theoretical foundation until Newton provided a rigorous mathematical explanation through his laws of motion and the law of universal gravitation.

Despite their importance, visualizing the Kepler problem during university lectures can be challenging. Showing how planetary trajectories evolve when initial conditions such as position and velocity are varied is difficult using only schematic diagrams or pre-made videos. Such traditional approaches often fail to fully engage students, who must rely largely on their imagination without the possibility of conducting real experiments.

To address this pedagogical gap, this article focuses on the numerical solution of the Kepler problem and the visualization of planetary motion within the Solar System. We provide a concrete example implemented in Wolfram Mathematica, which can be directly used in practical or laboratory classes on Newton's law of universal gravitation and the Kepler problem. Through this computational approach, students and educators can simulate planetary orbits, analyze different initial conditions, and verify fundamental properties such as the conservation of orbital angular momentum and total mechanical energy. Moreover, the code can be easily adapted to study the motion of natural and artificial satellites, simulate cometary trajectories, or explore other space research scenarios.

Mathematical Formulation and Methods

Within the framework of Newtonian gravity, the Kepler problem can be formulated step by step as follows:

1. Newton's Second Law of Motion

This law relates the net force **F** acting on a body to its mass m and acceleration **a**:

$$\mathbf{F} = m\,\mathbf{a} \tag{1}$$

Since acceleration is the time derivative of velocity $\mathbf{a} = d\mathbf{v}/dt$, and velocity is the time derivative of the position vector $\mathbf{v} = d\mathbf{r}/dt$, the equation naturally links the applied force to the trajectory of the particle. Note that we use boldface notation for all vector quantities throughout this article.

2. Law of Universal Gravitation

Newton's law of gravitation gives the attractive force between two point masses M and m, separated by a distance $r = |\mathbf{r}|$:

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

where G is the universal gravitational constant and r, on the other hand is the magnitude of the position vector, which represents the radial distance between the two bodies [4, 5]. This force is directly determined by the position vector of the test particle with respect to the massive body. The negative sign in the force expression appears because the position vector **r** is directed outward from the source (located at the coordinate origin) toward the particle, whereas the gravitational force acting on the test particle is directed inward, toward the source.

3. Equation of Motion in a Central Gravitational Field

By equating the two expressions for the force, one obtains the equation of motion for a test particle in the gravitational field of a central body of mass M:

$$m\,d^2\mathbf{r}/dt^2 = -\,G\,m\,M\,\mathbf{r}/r^3, \tag{3}$$

After canceling the test particle's mass m, this simplifies to

$$d^2\mathbf{r}/dt^2 = -G\,M\,\mathbf{r}/r^3 \quad (4)$$

which is a second-order differential equation describing the relationship between acceleration, velocity, and position of the orbiting body.

Thus, the Kepler problem reduces to solving this vector differential equation for given initial conditions (initial position and initial velocity). Its solutions describe all possible orbital trajectories: circular, elliptical, parabolic, or hyperbolic, depending on the total energy of the system. As can be seen, Eq. (4) constitutes a system of second-order differential equations for the Cartesian components x, *y*, and *z*. For simplicity, we restrict the motion to the equatorial plane by setting $z = 0$. In this case, the position vector and its magnitude are expressed as

$$\mathbf{r} = \mathbf{i}\,x + \mathbf{j}\,y, \quad (5)$$

$$r = |\mathbf{r}| = (x^2+y^2)^{1/2} \quad (6)$$

where **i** and ***j*** are the unit vectors along the x and y axes, respectively. Taking into account the definition of the position vector and its magnitude, Eq. (4) can be decomposed into two ordinary differential equations forming a coupled system

$$d^2x/dt^2 = -G\,M\,x/(x^2+y^2)^{3/2} \quad (7)$$

$$d^2y/dt^2 = -G\,M\,y/(x^2+y^2)^{3/2} \quad (8)$$

Consequently, we obtain a system of equations that must be solved numerically. It is worth noting that an analytical solution for this system exists in polar coordinates, as presented in various physics textbooks, where the conservation of energy and orbital angular momentum is explicitly taken into account [1, 2, 3]. However, the analytical treatment is beyond the scope of the present article.

To solve the system of Eqs. (7) and (8), one must first specify a particular planet - for example, the Earth orbiting the Sun, any other planet within the Solar System, or even an exoplanetary system. Naturally, basic orbital parameters, such as the average orbital speed and the mean distance from the Sun, should be obtained from observational data [2, 3]. For more specific or precise problems, additional parameters like the orbital eccentricity, the lengths of the semi-major and semi-minor axes, and the inclination angle relative to the equatorial plane would also be required. However, for purely illustrative purposes, knowing only the average orbital speed and distance is sufficient.

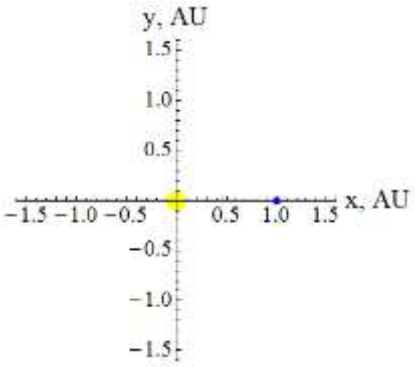


Figure 1 – Schematic illustration of the Sun (yellow disk) and Earth (blue dot) located in the z=0 plane.

In Fig. 1 we present a schematic representation of the position of the Sun and Earth position in a Cartesian coordinate system. The Sun is placed at the coordinate origin (0,0), while the Earth is initially positioned at coordinates (1,0) in astronomical units (AU). In principle, the Earth could be placed at any location; however, for simplicity and numerical convenience, we adopt these initial coordinates.

Let us assume that at initial time t=0, the position of the Earth is given as shown in Fig. 1.

Thus, $x(0) = 1$ AU and $y(0) = 0$ AU. The time derivative of the coordinates, $x' = dx/dt = v_x$ and $y' = dy/dt = v_y$, represent the components of the velocity along the x and y axes, respectively. According to Fig. 1, we choose the initial velocity components of the Earth as $x'(0)=0$, indicating that the Earth does not move along the x axis at this instant, and $y'(0)= 29.8$ km/s, meaning that at the point (1, 0) AU the Earth moves perpendicular to the x axis i.e. tangentially to its orbit.

The motion is taken to be counterclockwise. With these initial conditions specified, one can run the Wolfram Mathematica [6-8] code to obtain numerical the results.

Mathematica and its alternatives

Before presenting the coding part, it is important to emphasize that Mathematica is a comprehensive computational software environment designed to perform a wide spectrum of mathematical and scientific tasks. At the simplest level, it can function as a standard calculator for engineers and applied scientists. At the same time, its more advanced capabilities allow it to serve as a powerful platform for symbolic and analytic computations. Specifically, Mathematica can solve linear, nonlinear, and transcendental equations, as well as ordinary and partial differential equations, both analytically and

numerically. It is also capable of performing definite and indefinite integrations, Taylor and other series expansions, limits, summations, and root-finding, along with convergence checks for series. Visualization is another strong feature: the software can generate two- and three-dimensional plots, contour and density plots, parametric plots, and even dynamic animations of curves, surfaces, and volumes. In addition, Mathematica incorporates advanced functionalities such as tensor calculus, variational principles, optimization routines, and symbolic algebra, making it especially suitable for research in physics, mathematics, and engineering. Many experts also employ Mathematica as a high-level programming language, applying it to tasks ranging from data analysis to complex physical simulations. Thus, Mathematica may reasonably be regarded as an effective and versatile tool for physicists and other scientists engaged in analytical and computational investigations.

Despite these strengths, Mathematica is not unique in the field of computational software. Its closest competitors include Maple [9], Mathcad [10], and Matlab [11]. In terms of symbolic manipulation and pure mathematical rigor, Maple is often considered more transparent and efficient, particularly for algebraic computations. Mathcad, on the other hand, provides an interface that is highly intuitive and document-oriented, which can be advantageous for engineering applications requiring interactive calculations. Matlab is widely regarded as the most robust option for numerical linear algebra, signal processing, and large-scale numerical simulations, though its symbolic capabilities are more limited and typically rely on additional toolboxes. By contrast, Mathematica distinguishes itself by combining symbolic and numerical methods in a single integrated environment, while also offering strong visualization and programming support.

Beyond these direct rivals, there exist indirect competitors such as Python [12] and GeoGebra [13] etc. Python, with its extensive ecosystem of open-source libraries, has become increasingly popular for both education and research due to its flexibility and accessibility. GeoGebra, while simpler, offers an interactive environment well-suited for teaching and introductory-level visualization of mathematical problems. Although these alternatives overlap with Mathematica in certain applications, they often target specific niches rather than attempting to provide a single unified framework.

Overall, the choice between Mathematica and its alternatives depends on the scope of the problem, available resources, and the user's background. However, its unique integration of symbolic manipulation, numerical analysis, visualization, and programming continues to make Mathematica a valuable and versatile tool in both education and scientific research.

Results and Discussion

We begin by entering the position vector along with its first and second derivatives with respect to time

```
In[1]:= r = {x[t], y[t]}
        v = D[r, t]
        a = D[r, {t, 2}]
Out[1]= {x[t], y[t]}
Out[2]= {x'[t], y'[t]}
Out[3]= {x''[t], y''[t]}
```

where "In[]" indicates the user's input, while "Out[]" represents the corresponding output generated by Wolfram Mathematica \cite{6–8}. Moreover, "Out[]" also signifies that all quantities in the output are stored in Mathematica's active memory. The components of the position vector are enclosed within curly brackets "{}" without explicitly writing unit vectors. To compute the first and second derivatives, we use the command "D"; for more details, see [6–8]. The dot (scalar) product of the position vector with itself is calculated using the "." operator. For example:

```
In[4]:= r.r
Out[4]= x[t]^2 + y[t]^2
```

this input returns the square of the position vector, and taking its square root yields the magnitude of the position vector, i.e., the radial distance. Based on this, we can now express the forces according to Newton's second law of dynamics and the law of universal gravitation as follows:

$$\text{In[5]:= } \mathbf{FN = m\,a}$$

$$\mathbf{FG = -\frac{G\,m\,M\,r}{(r.r)^{3/2}}}$$

$$\text{Out[5]= } \{m\,x''[t],\ m\,y''[t]\}$$

$$\text{Out[6]= } \left\{-\frac{G\,m\,M\,x[t]}{\left(x[t]^2+y[t]^2\right)^{3/2}},\ -\frac{G\,m\,M\,y[t]}{\left(x[t]^2+y[t]^2\right)^{3/2}}\right\}$$

where the outputs display the *x* and *y* components of the corresponding forces. The equations of motion are then obtained by equating these two forces and canceling the mass of the Earth, as follows:

```
In[7]:= eq = FN/m == FG/m // Thread
Out[7]= {x''[t] == -(G M x[t])/(x[t]^2 + y[t]^2)^(3/2), y''[t] == -(G M y[t])/(x[t]^2 + y[t]^2)^(3/2)}
```

where we use the command "Thread" to equate the corresponding components of the forces. To solve the resulting system of equations, we must specify the initial conditions for the Earth:

```
In[8]:= ic = {x[0] == AU, y[0] == 0, x'[0] == 0, y'[0] == v0}
Out[8]= {x[0] == AU, y[0] == 0, x'[0] == 0, y'[0] == v0}
```

In addition, it is necessary to define the physical constants and the parameters of the Earth's orbit:

```
In[9]:= G = 6.6743 * 10^-11 (*gravitational constant in SI units or m^3/(kg s^2)*)
        M = 1.98892 * 10^30 (*mass of the Sun in kilograms (kg)*)
        v0 = 29.8 * 1000 (*average orbital speed of Earth in meters per second (m/s)*)
        AU = 149597870700 (*astronomical unit in meters*)
        t0 = 365.25 * 24 * 60 * 60 (*one year in seconds*)
Out[9]= 6.6743 × 10^-11
Out[10]= 1.98892 × 10^30
Out[11]= 29800.
Out[12]= 149597870700
Out[13]= 3.15576 × 10^7
```

Using all the data above, one can proceed with the solution:

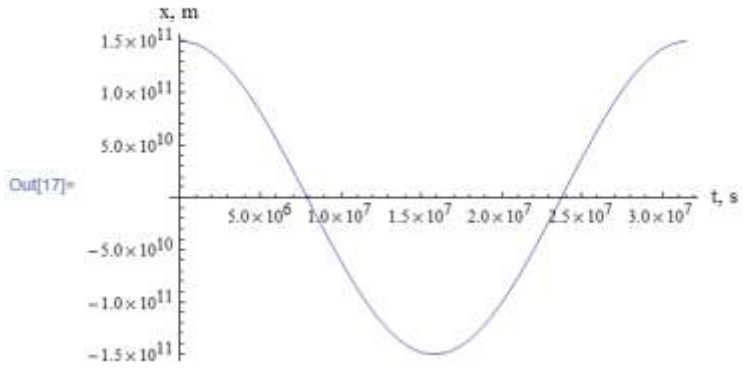
```
In[14]:= sol = NDSolve[Join[eq, ic], {x[t], y[t]}, {t, 0, t0}]
Out[14]= {{x[t] → InterpolatingFunction[{{0., 3.15576 × 10^7}}, <>][t],
          y[t] → InterpolatingFunction[{{0., 3.15576 × 10^7}}, <>][t]}}
```

where "sol" is simply the name assigned to the numerical solution. The command "NDSolve" is used to numerically solve the differential equations, while "Join" combines the equations and initial conditions in a single system. It is necessary to explicitly specify the functions we seek - "*x*[*t*]" and "*y*[*t*]" - along with the range of the independent variable, "{*t*, 0, *t*0}". The solution is provided as interpolation functions, which do not have closed-form analytic representations. However, if needed, one can obtain approximate analytic expressions using a fitting function. Next, one can assign the numerical solutions to new functions using the following procedure:

```
In[15]:= xs[t_] = x[t] /. sol[[1]]
         ys[t_] = y[t] /. sol[[1]]
Out[15]= InterpolatingFunction[{{0., 3.15576 × 10^7}}, <>][t]
Out[16]= InterpolatingFunction[{{0., 3.15576 × 10^7}}, <>][t]
```

This step is necessary to reduce the number of subsequent inputs. Using this newly defined functions, one can plot their time dependence. Let us first consider the function $x(t)$:

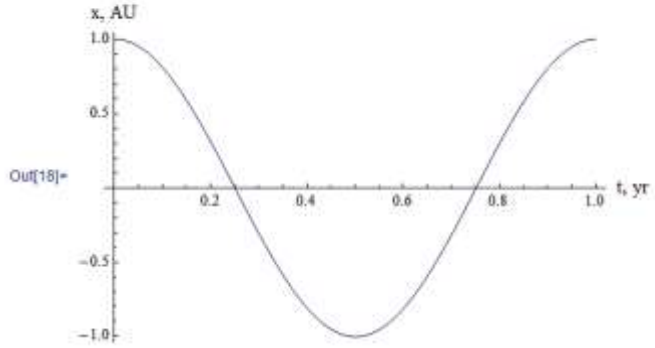
```
In[17]:= Plot[xs[t], {t, 0, t0}, AxesLabel → {Style["t, s", Medium], Style["x, m", Medium]},
         BaseStyle → {FontFamily → "Times", FontSize → 10}]
```



Here, the "Plot" command is used to generate this graph. The option "AxesLabel" specifies the labels for the x- and y-axes, while "BaseStyle" controls the font family and size used for these labels. As one can see, the function (coordinate) x is expressed in meters and the time t in seconds, which are not always convenient units.

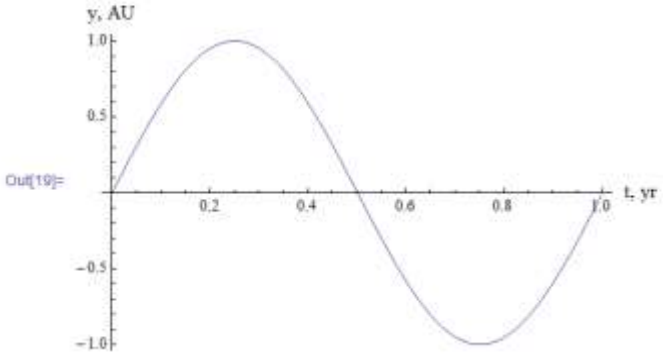
Therefore, to express x in AU and t in years we normalize x by AU and redefine its argument accordingly:

```
In[18]:= Plot[xs[t t0] / AU, {t, 0, 1}, AxesLabel → {Style["t, yr", Medium], Style["x, AU", Medium]},
         BaseStyle → {FontFamily → "Times", FontSize → 10}]
```



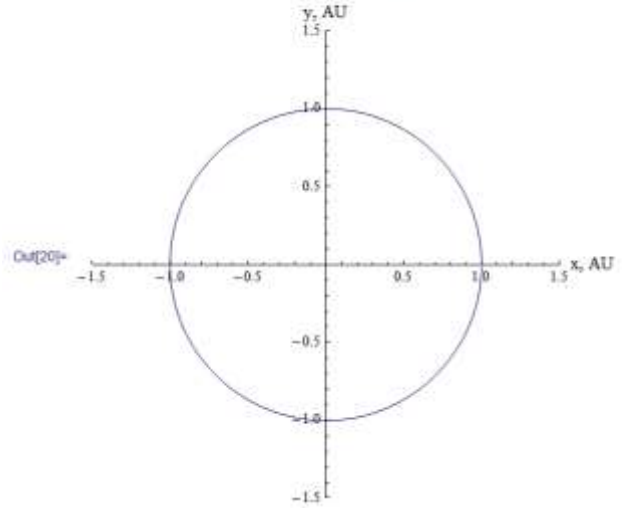
Similarly, one can plot the graph for the $y(t)$ coordinate:

```
In[19]:= Plot[ys[t t0] / AU, {t, 0, 1}, AxesLabel → {Style["t, yr", Medium], Style["y, AU", Medium]},
         BaseStyle → {FontFamily → "Times", FontSize → 10}]
```



Evidently, the function $x(t)$ resembles a cosine function, while $y(t)$ resemble a sine function, indicating that the Earth trajectory is circular. The meaning of these curves becomes evident upon examining Fig. 1, where the coordinates x and y for circular orbits vary periodically, taking both positive and negative values. To explicitly demonstrate that the trajectory is indeed circular, one should plot y as a function of x. This can be easily achieved in Mathematica using the "ParametricPlot" command:

```
In[20]:= ParametricPlot[{xs[t t0] / AU, ys[t t0] / AU}, {t, 0, 1}, PlotRange → {-1.5, 1.5},
          AxesLabel → {Style["x, AU", Medium], Style["y, AU", Medium]}]
```



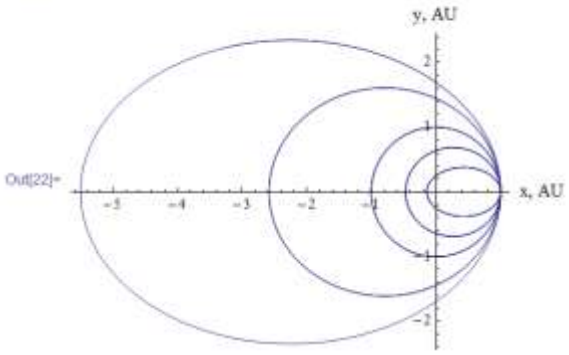
This plot represents the trajectory of the Earth around the Sun over the course of one year. In practice, one can choose any planet in the Solar System and analyze its trajectory. Most planets follow nearly circular orbits, with only small deviations from circularity, which are not noticeable at the scale and time interval considered here. In this figure, the "PlotRange" option of the "ParametericPlot" command specifies the ranges of the *y*-axis.

To explore further, let us consider a thought (gedanken) experiment. Suppose that, similar to the Earth, we place artificial bodies at the same initial coordinates but assign them different initial velocities. What would happen to their trajectories if their speeds were 0.5, 0.8, 1.0, 1.2, and 1.3 times the Earth's orbital speed (29.8 km/s)?

To display different trajectories on the same plot, we use the "Table" and "Show" commands as follows:

```
In[21]:= Table[
          {ic = {x[0] == AU, y[0] = 0, x'[0] = 0, y'[0] = i * v0};
           sol = NDSolve[Join[eq, ic], {x[t], y[t]}, {t, 0, 7 t0}];
           xs[t_] = x[t] /. sol[[1]];
           ys[t_] = y[t] /. sol[[1]];
           ParametricPlot[{xs[t t0] / AU, ys[t t0] / AU}, {t, 0, 7},
            AxesLabel → {Style["x, AU", Medium], Style["y, AU", Medium]}]}, {i, {0.5, 0.8, 1, 1.2, 1.3}}];

In[22]:= Show[%, PlotRange → All]
```



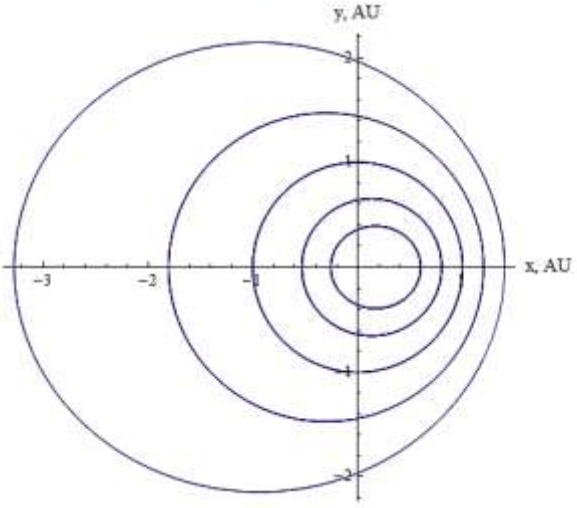
where the index "*i*" in front of v0 corresponds to the velocity scaling factor, running through 0.5, 0.8, 1.0, 1.2, and 1.3. We kept the initial position of the Earth fixed, but varied its initial speed as described. The resulting trajectories explicitly show that for smaller speeds, the initial position of the artificial bodies corresponds to the aphelion, while for larger speeds, the initial position corresponds to the perihelion. In all cases, the Sun remains fixed and located at one of the foci of the resulting ellipses. Practically, the "Table" command makes it possible to generate multiple individual plots simultaneously in a table, depending on the range of the index "i", while the "Show" command superimposes all these plots into a single figure.

Next, let us now keep initial speed constant and instead vary initial position to 0.6, 0.8, 1, 1.2 and 1.4 AU for the artificial objects.

```
In[23]:= Table[
          {ic = {x[0] = i * AU, y[0] = 0, x'[0] = 0, y'[0] = v0};
           sol = NDSolve[Join[eq, ic], {x[t], y[t]}, {t, 0, 7 t0}];
           xs[t_] = x[t] /. sol[[1]];
           ys[t_] = y[t] /. sol[[1]];
           ParametricPlot[{xs[t t0] / AU, ys[t t0] / AU}, {t, 0, 7},
            AxesLabel → {Style["x, AU", Medium], Style["y, AU", Medium]}]}, {i, {0.6, 0.8, 1, 1.2, 1.4}}];

In[24]:= Show[%, PlotRange → All]
```

Out[24]=



where the index "*i*", now in front of AU, runs through 0.6, 0.8, 1.0, 1.2, and 1.4. We observe that even with a fixed initial speed, changing the initial position can significantly alter the resulting trajectories. For orbits lying inside (outside) the Earth's trajectory, the initial position corresponds to the aphelion (perihelion).

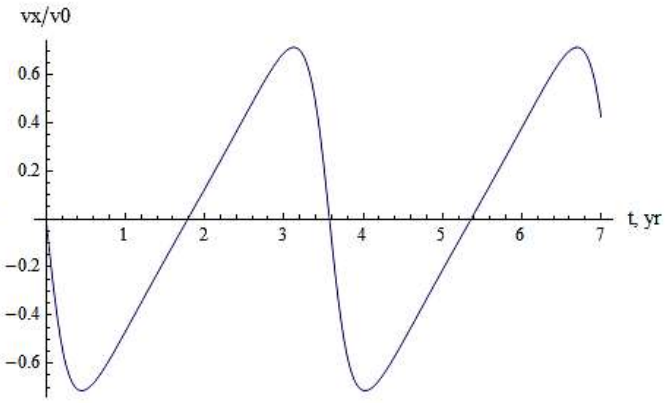
These two plots together illustrate Kepler's First Law, which states: *"The orbit of a planet is an ellipse with the Sun at one of the two foci."*

Next, we select the largest orbit – specifically, the outmost one with $v0$=29.8 km/s and initial coordinates (1.4, 0) AU from "Out [24]".

The x component of the velocity $v_x = x'(t)$ is plotted in the following graph. This velocity component takes both positive and negative values, indicating changes in the direction of motion with respect to x axis.

```
In[25]:= Plot[xs'[t t0] / v0, {t, 0, 7}, AxesLabel → {Style["t, yr", Medium], Style["vx/v0", Medium]},
          BaseStyle → {FontFamily → "Times", FontSize → 10}]
```

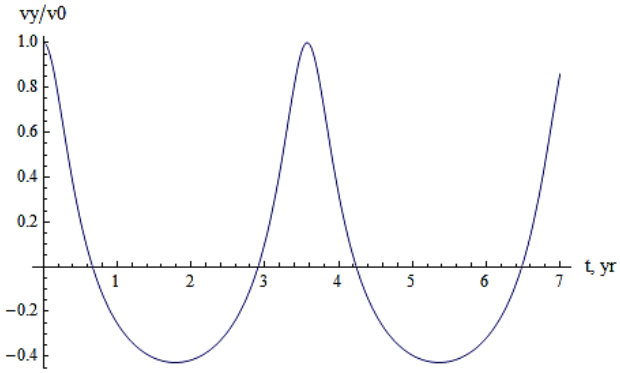
Out[25]=



In the graph below, we plot the y component of the velocity $v_y = y'(t)$. As seen, the maximum speed is attained at the perihelion (which coincides with the planet's initial position), while the minimum speed is reached at the aphelion. The positive and negative values indicate the direction of motion with respect to the y axis.

```
In[26]:= Plot[ys'[t t0] / v0, {t, 0, 7}, AxesLabel → {Style["t, yr", Medium], Style["vy/v0", Medium]},
  BaseStyle → {FontFamily → "Times", FontSize → 10}]
```

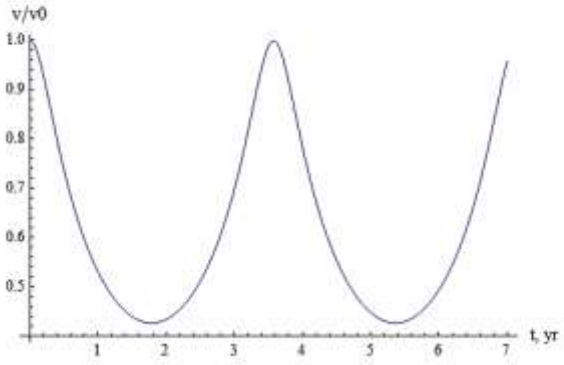
Out[26]=



The total speed is calculate as $v = (v_x^2 + v_y^2)^{1/2}$, as shown in "Out[27]". Naturally, the magnitude of the speed is always non-negative and never zero, with its maximum value at the perihelion and its minimum at the aphelion, as illustrated below

```
In[27]:= Plot[√((xs'[t t0] / v0)^2 + (ys'[t t0] / v0)^2), {t, 0, 7}, AxesLabel → {Style["t, yr", Medium], Style["v/v0", Medium]},
  BaseStyle → {FontFamily → "Times", FontSize → 10}, AxesOrigin → {0, 0.4}]
```

Out[27]=



In this context, one can numerically verify the conservation of orbital angular momentum. To this purpose, we use the standard definition of angular momentum [1-5]

$$\mathbf{L} = [\mathbf{r} \times \mathbf{p}] = m[\mathbf{r} \times \mathbf{v}] = \text{constant} \qquad (9)$$

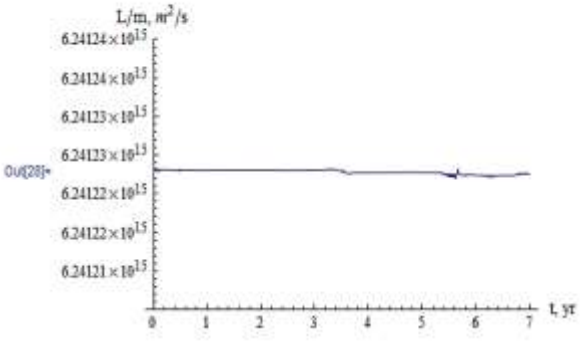
where **p** denotes the linear momentum of the test body. In the equatorial plane ($z = 0$), the magnitude of the orbital angular momentum is given by

$$L = |\mathbf{L}| = m\,(x\,v_y - y\,v_x) = \text{constant}. \qquad (10)$$

If we express L/m using our numerical results, we obtain

```
In[28]:= Plot[xs[t t0] * ys'[t t0] - ys[t t0] * xs'[t t0], {t, 0, 7}, AxesLabel → {Style["t, yr", Medium], Style["L/m, m^2/s", Medium]},
  BaseStyle → {FontFamily → "Times", FontSize → 10}, PlotRange → {6.241205 * 10^15, 6.24124 * 10^15}]
```

Out[28]=



From the plot, it is evident that the orbital angular momentum normalized by the mass of the test body remains constant – a conserved physical quantity, also referred to as an integral of motion. The

small deviations observed are merely numerical artifacts and do not affect the physical interpretation of the problem.

The constancy of the planet's orbital angular momentum directly confirms Kepler's second law [5]. To further illustrate this, let us recall the definition of the areal (or sectorial) velocity [9, 10]

$$d\mathbf{A}/dt = [\mathbf{r} \times \mathbf{v}]/2 = \mathbf{L}/(2m). \quad (11)$$

For a conserved orbital angular momentum, and a given time interval Δt, this expression can be written as

$$\Delta A = L/(2m)\ \Delta t \quad (12)$$

where ΔA denotes the area of the sector swept by the planet during the time interval Δt. Therefore, Kepler's second law - stating: "A line segment joining a planet and the Sun sweeps out equal areas during equal intervals of time" is numerically validated through the constancy of the orbital angular momentum.

Furthermore, one can also numerically verify the conservation of the total energy of the test body (planet/particle). To do so, we use the standard definition of the total energy [11]

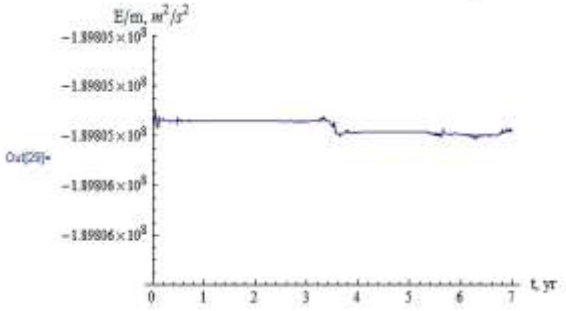
$$E = m\,v^2/2 - G\,m\,M/r. \quad (13)$$

The total energy, normalized with respect to the mass of the test body, can be expressed in terms of the velocities and coordinates

$$E/m = (v_x^2 + v_y^2)/2 - G\,M/(x^2+y^2)^{1/2}. \quad (14)$$

and using our numerical results we obtain

```
In[29]:= Plot[{(xs'[t t0])^2 + (ys'[t t0])^2}/2 - G M/Sqrt[(xs[t t0])^2 + (ys[t t0])^2], {t, 0, 7},
  AxesLabel -> {Style["t, yr", Medium], Style["E/m, m^2/s^2", Medium]}, BaseStyle -> {FontFamily -> "Times", FontSize -> 10},
  PlotRange -> {-1.89805 * 10^8, -1.89806 * 10^8}]
```

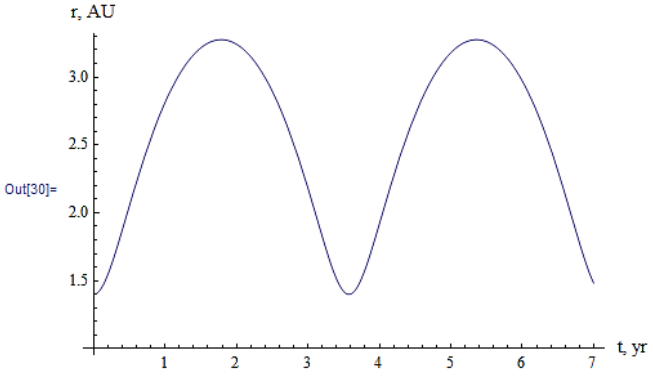


The resulting plot confirms the constancy of the total energy. Notably, the energy remains negative, indicating bound orbits are elliptical. The small oscillations visible in the plot arise from numerical artifacts and do not affect the physical interpretation.

In the literature, the conservation of both the orbital angular momentum and the total energy of the test body is commonly used to derive and confirm Kepler's third law [12-15]. However, to validate the third law, we will adopt a slightly different approach based on our numerical results.

First, by using Eq. (6) and the output from "Out [24]" we compute the magnitude of the position vector in AU

```
In[30]:= Plot[Sqrt[xs[t t0]^2 + ys[t t0]^2]/AU, {t, 0, 7}, AxesOrigin -> {0, 1.0},
  AxesLabel -> {Style["t, yr", Medium], Style["r, AU", Medium]},
  BaseStyle -> {FontFamily -> "Times", FontSize -> 10}]
```



We observe that the radial distance varies over time, exhibiting both maxima and minima, which can be determined using the Mathematica commands: "FindMaximum" and "FindMinimum"

```
In[31]:= rt1 = FindMaximum[Sqrt[xs[t t0]^2 + ys[t t0]^2] / AU, {t, 2}]
Out[31]= {3.27508, {t → 1.78674}}

In[32]:= rmax = rt1[[1]]
Out[32]= 3.27508

In[33]:= rt2 = FindMinimum[Sqrt[xs[t t0]^2 + ys[t t0]^2] / AU, {t, 3.6}]
Out[33]= {1.4, {t → 3.57348}}

In[34]:= rmin = rt2[[1]]
Out[34]= 1.4
```

where rt1 and rt2 are simple notations representing the maximum and minimum values of the radial distance r (rmax and rmin) along with the corresponding time interval t (see the previous graph). The notation "[[]]" is used to select a specific element from a given list. The length of the semi-major axis "a" is defined in AU as

```
In[35]:= a = (rmax + rmin) / 2
Out[35]= 2.33754
```

and the orbital period is obtained in years directly from the graph, and it is contained within the result of rt2

```
In[36]:= period = rt2[[2, 1, 2]]
Out[36]= 3.57348
```

Thus, the ratio of the square of the orbital period to the cube of the semi-major axis is equal to

```
In[37]:= period^2 / a^3
Out[37]= 0.999781
```

It is important to note that this computation is performed for the initial conditions "$x(0) = i$*AU, $y(0) = 0$, $x'(0) = 0$, $y'(0) = v0$", where "i" was set to 1.4, corresponding to the outermost orbit from "Out[24]".

Now, let us consider the case when "$i = 0.6$". In order to optimize the computation, we employ the "Table" command. Note that "$i = 0.6$" corresponds to the innermost orbit from "Out[24]".

Examining Out[24], we observe that the Sun remains at the coordinate origin, but its position relative to foci differs for the cases "$i = 0.6$" and "$i = 1.4$" cases. This implies, as before, that for the innermost (outermost) orbit the initial coordinates correspond to the aphelion (perihelion). Therefore, special care must be taken when calculating "rt1" and "rt2", and orbital period (see "In[38]").

In "Out[38]" we obtain two graphs: the first one shows the trajectory of the test body, while the second displays the radial distance r in AU as a function of time t in years. The final value in the list of "Out[38]" gives the ratio of the square of the orbital period to the cube of the semi-major axis, which equals 0.999782, very close to the result obtained in "Out[37]".

The slight discrepancy is due to numerical integration errors. Thus, we have numerically validated Kepler's third law, which states: "The square of a planet's orbital period is proportional to the cube of the length of the semi-major axis of its orbit" [13].

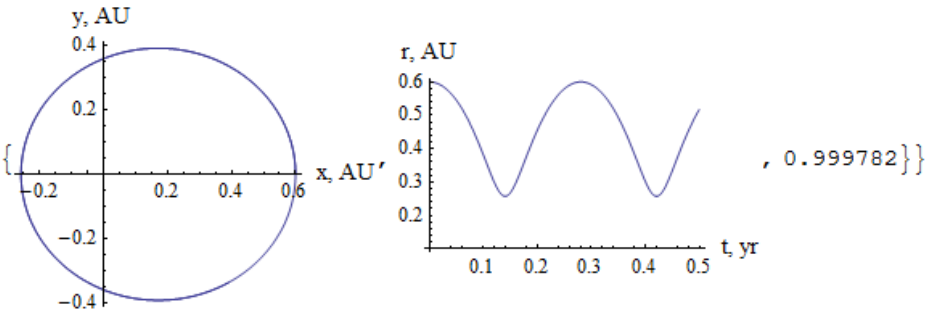
```
In[38]:= Table[
  {ic = {x[0] == i * AU, y[0] == 0, x'[0] == 0, y'[0] == v0};
   sol = NDSolve[Join[eq, ic], {x[t], y[t]}, {t, 0, 7 t0}];
   xs[t_] = x[t] /. sol[[1]];
   ys[t_] = y[t] /. sol[[1]];

   ParametricPlot[{xs[t t0] / AU, ys[t t0] / AU}, {t, 0, 0.5},
    AxesLabel → {Style["x, AU", Medium], Style["y, AU", Medium]}],

   Plot[Sqrt[xs[t t0]^2 + ys[t t0]^2] / AU, {t, 0, 0.5}, AxesOrigin → {0, 0.1},
    AxesLabel → {Style["t, yr", Medium], Style["r, AU", Medium]},
    BaseStyle → {FontFamily → "Times", FontSize → 10}],

   rt1 = FindMaximum[Sqrt[xs[t t0]^2 + ys[t t0]^2] / AU, {t, 0.25, 0.35}];
   rmax = rt1[[1]];
   rt2 = FindMinimum[Sqrt[xs[t t0]^2 + ys[t t0]^2] / AU, {t, 0.1, 0.2}];
   rmin = rt2[[1]];
   a = (rmax + rmin) / 2;
   period = rt1[[2, 1, 2]];
   period^2 / a^3 }, {i, {0.6}}]
```

Out[38]=



$$\{\{\ldots, 0.999782\}\}$$

Conclusion

In this article, we revisited the classical Kepler problem, discussed its fundamental characteristics, and presented its mathematical formulation. We demonstrated how it can be solved numerically using Wolfram Mathematica software. The provided code illustrates the motion of the Earth and artificial objects in the gravitational field of the Sun. However, it can be easily adapted to simulate a wide range of problems, such as the motion of artificial satellites in Earth's gravitational field, the dynamics of Jupiter's or Saturn's moons, or even the study of exoplanetary systems.

The code presented here was developed in Mathematica 7.0, but it is fully compatible with more recent versions. A detailed description of the commands and their usage can be found in textbooks, online resources, or within the software's built-in Help system.

Through the analyses in this work, we numerically verified the conservation laws of orbital angular momentum and total energy for test bodies. Furthermore, we validated Kepler's three laws of planetary motion - results often omitted during standard lectures and practical classes.

Finally, the material and methodology presented in this paper can be effectively used in practical or laboratory sessions to reinforce theoretical knowledge acquired during lectures and to perform numerical experiments. Moreover, these procedures can serve as an entry point for exploring other fundamental problems in celestial mechanics and can inspire further research and academic activities for students.

We anticipate that the analyses and simulations presented here, when used alongside traditional lectures and practical classes, will foster a deeper and

broader understanding of the underlying physics and mathematical aspects of the problem under consideration. Supplementary materials in the form of codes provide students with greater flexibility, enabling them to explore a wider range of scenarios by varying not only the initial conditions but also by selecting different systems, such as extrasolar planetary systems on elliptical orbits or individual objects on parabolic and hyperbolic trajectories. We therefore expect that such simulations can have high pedagogical effectiveness because they help students visualize not obvious and straightforward concepts, increase engagement, and allow experimentation with parameters that would be impossible in a traditional classroom setting.

Finally, it would be of interest to investigate the motion of various objects in the gravitational field of deformed central bodies, in analogy with [21-23]. Furthermore, it would be instructive to analyze the dynamics of charged particles in both uniform and non-uniform electromagnetic fields, following the approaches of [24-24]. These problems, however, fall beyond the scope of the present study and are left for future works.

Acknowledgments

KB and AU acknowledge the support from the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan Grants No. AP19680128 and BR21881941.

References

1 L.D. Landau & E.M. Lifshitz, Mechanics, Vol. 1 of Course of Theoretical Physics, 3rd ed. (Butterworth-Heinemann, 1993).

2 R.A. Serway & J.W. Jewett, Physics for Scientists and Engineers with Modern Physics, 9th ed. (Cengage Learning, 2014).

3 R.A. Freedman, H.D. Young & A.L. Ford, University Physics. 15th ed. (Pearson, 2018).

4 E. Poisson & C.M. Will, Gravity: Newtonian, Post-Newtonian, Relativistic, (Cambridge University Press, 2014).

5 H.C. Ohanian & R. Ruffini, Gravitation and Spacetime, 3rd ed., (Cambridge University Press, 2013).

6 Wolfram Research. (n.d.). Mathematica Online. Retrieved from https://www.wolfram.com/mathematica/online/

7 S. Hassani, Mathematical Methods Using Mathematica: For Students of Physics and Related Fields. (Springer, 2003).

8 R.L. Zimmerman & F.I. Olness, Mathematica for Physics, 2nd ed., (Addison-Wesley. 2002).

9 https://www.maplesoft.com/products/maple/

10 https://www.mathcad.com/en

11 https://www.mathworks.com/products/matlab.html

12 https://www.python.org/

13 https://www.geogebra.org/

14 J. Casey, Areal velocity and angular momentum for non-planar problems in particle mechanics, American Journal of Physics **75**(8), 677–685 (2007). https://doi.org/10.1119/1.2735630

15 J.B. Brackenridge, The Key to Newton's Dynamics: The Kepler Problem and the Principia, (University of California Press. ISBN 978-0-520-20217-7, 1995).

16 G.K. Vijaya, Original form of Kepler's third law and its misapplication in Propositions XXXII–XXXVII in Newton's Principia (Book I), Heliyon **5**(2), e01274 (2019). https://doi.org/10.1016/j.heliyon.2019.e01274

17 A. Tan & W.L. Chameides, Kepler's third law, American Journal of Physics **49**(7), 691–692 (1981).

18 R.A. Aziz, Kepler's third law, American Journal of Physics **34**, 538 (1966).

19 D. Kleppner & R. Kolenkow, Central force motion. In An Introduction to Mechanic, 2nd ed., (Cambridge University Press, 2014), pp. 391–392.

20 H.D. Young & R.A. Freedman, Chapter 13: Gravitation. In University Physics with Modern Physics, 13th ed., (Addison-Wesley, 2012), 416 p.

21 K. Boshkayev, K. Baiseitov, Z. Brisheva A. & Tlemisov, Investigation of the motion of test particles in the gravitational field of axially symmetric central body in classical physics, Recent Contributions to Physics **3**(66), 84–98 (2018). https://bph.kaznu.kz/index.php/zhuzhu/article/view/787 (in Russ).

22 B.A. Mukushev, Computational experiments on the study of the motion of celestial bodies, Recent Contributions to Physics **2**(85), 49-58 (2023). https://doi.org/10.26577/RCPh.2023.v85.i2.08 (in Russ).

23 B.A. Mukushev, Energy picture of the gravitational field of the Solar system, Recent Contributions to Physics **4**(83), 59–66 (2022). https://doi.org/10.26577/RCPh.2022.v83.i4.07 (in Russ).

24 K. Mukashev, & K. Kusmanuly, Towards a methodology for teaching the motion of a charged particle in an electric and magnetic field in a physics course, Recent Contributions to Physics **1**(25), 169–175 (2008). https://bph.kaznu.kz/index.php/zhuzhu/article/view/310 (in Kaz).

25 B. Beisenov, K. Boshkayev, Z. Brisheva, B. Zhami, Z. Kalymova, E. Kuanyshbayuly, & A. Urazalina, Methods for calculating the magnetic field of rotating charge distribution with spherical symmetry, Recent Contributions to Physics **3**(70), 82–91 (2019). https://doi.org/10.26577/RCPh-2019-i3-10 (in Russ).

Мақала тарихы:
Түсті – 23.07.2025
Түзетілген түрде түсті – 29.09.2025
Қабылданды – 10.10.2025

Article history:
Received 23 July 2025
Received in revised form 29 September 2025
Accepted 10 October 2025

Авторлар туралы мәлімет:

1. **Мирас Жұмағалиев** – King's College студенті, British School of Alicante, Аликанте, Испания, e-mail: miraszhumagaliyev@gmail.com
2. **Қуантай Бошқаев** (хат-хабарларға жауапты автор), PhD, әл-Фараби атындағы Қазақ ұлттық университетінің Теориялық және ядролық физика кафедрасының профессоры, Алматы, Қазақстан, e-mail: kuantay@mail.ru
3. **Айнұр Уразалина**, PhD, әл-Фараби атындағы Қазақ ұлттық университетінің Теориялық және ядролық физика кафедрасының доценті, Алматы, Қазақстан, e-mail: y.a.a.707@mail.ru

Information about authors:

1. **Miras Zhumagaliyev** – King's College student, The British School of Alicante, Alicante, Spain, e-mail: miraszhumagaliyev@gmail.com
2. **Kuantay Boshkayev** (corresponding author), PhD, Professor of the Department of Theoretical and Nuclear Physics, Al-Farabi Kazakh National University, Almaty, Kazakhstan, e-mail: kuantay@mail.ru
3. **Ainur Urazalina**, PhD, Associate Professor of the Department of Theoretical and Nuclear Physics, Al-Farabi Kazakh National University, Almaty, Kazakhstan, e-mail: y.a.a.707@mail.ru