SOLVING SOME PHYSICAL PROBLEMS USING THE METHODS OF NUMERICAL MATHEMATICS WITH THE HELP OF SYSTEM MATHEMATICA

Pavel Trojovský, Jaroslav Seibert, Antonín Slabý

ABSTRACT
Every future teacher of mathematics must know symbolic approaches, for example, for finding derivations and integrals, solving algebraic and differential equations very well. But most of the problems from practice are not solvable by a symbolic way and therefore numerical ones must be used. That is way the basic knowledge of some numerical methods for solving problems of the real world is a component of the complex training of teachers of mathematics. Different computer systems for symbolic and numerical mathematics as MatLab, Maple, *Mathematica* or Derive enable their effective use. The posibility to join numerical and symbolic computation in these systems makes them very well suited for expressing numerical algorithms, testing them out and postprocessing the results, for example in a graphical form. Their ability for high-precision or exact calculations is also important. This contribution deals with a concrete illustration of using the system *Mathematica* for solving several typical physical problems by differential equations or their systems.

KEYWORDS
System *Mathematica*, Runge-Kutta method, the simple pendulum, pendulum physlet, movement of projectile, orbits of satelite

INTRODUCTION

Mathematica is a powerful mathematical software system for students, researchers, and someone finding an effective tool for mathematical analysis. As the portable electronic calculator made doing simple arithmetic unnecessary, so powerful and sophisticated symbolic processing programs, such as

Mathematica, do calculus unnecessary by hand. Algebraic manipulation, finding roots of equations, derivatives of functions, solutions of differential equations and very large number of other mathematical operations have been done in a similar way and the solutions appear in symbolic form.

Mathematica also has extensive and powerful graphic capabilities. Such a symbolic processing programs have profound implications for mathematically rich fields (for example physics), where a great deal of time and much accumulated skill goes into master handling about complicated mathematics. With the advent of *Mathematica* educators have endeavored to make use of the mathematical manipulation power of this software system to improve the traditional way of teaching mathematics, physics and engineering.

Mathematica has been incorporated in a number of universities as a productivity tool to help students in analytical performance, numerical and graphical work.

Mathematica provides facilities for doing both of symbolic and numerical mathematics, or mixing the two as required.

Mathematica contains an extensive collection of algorithms for the users, covering almost every field of mathematics, including calculus, differential equations, matrix algebra, etc.

In our contribution we take two types of problems, the problems solvable symbolically using build-in commands in the system *Mathematica* (commands Integrate[], DSolve[] and so on) and the problems whose solution can be found only by numerical way. In this second case we use the fourth order Runge-Kutta method for numerical integration of autonomous systems of ordinary first-order differential equations. For example, we take solving the problem of movement projectile with resistance of air, the motion of satellite of the Earth in its gravitational field and oscillating of a simple pendulum in vacuum and with resistance of environment too. In all these examples we are going to show a mathematical description of the problem, a numerical solution, the graphs of functional dependencies of various quantities and computer animations of the movements.

THE RUNGE-KUTTA METHOD

An autonomous first-order system of differential equations of *n* variables $y_1, y_2, ..., y_n$ with the initial conditions at $y_1(x_0) = y_{10}$, $y_2(x_0) = y_{20}$, ..., $y_n(x_0) = y_{n0}$ in the system

$$y_1' = f_1(x, y_1, y_2, ..., y_n)$$
$$y_2' = f_2(x, y_1, y_2, ..., y_n)$$
.
.
.
$$y_n' = f_n(x, y_1, y_2, ..., y_n),$$

where $y_i', i = 1,2,...,n$ denote derivative of y_i with respect to a variable *x*.

Puting $\vec{y} = (y_1, y_2, ..., y_n)$ and $\vec{f} = (f_1(x, \vec{y}), f_2(x, \vec{y}), ..., f_n(x, \vec{y}))$ we can rewrite the equations and the initial conditions simply as $\vec{y}' = \vec{f}(x, \vec{y})$, $\vec{y}(x_0) = \vec{y}^{(0)}$. A numerical method for solving such a system starts from the initial conditions $\vec{y}^{(0)}$. The given value $\vec{y}^{(0)} = \vec{y}(x_0)$ leads to find the value $\vec{y}^{(1)}$ at the point $x_1 = x_0 + h$, then this value leads to find the value $\vec{y}^{(2)}$ at $x_1 = x_0 + 2h$ and so on. The constant $h > 0$ is called the step size.

There are many different formulaes for this purpose. We take the fourth order Runge-Kutta method in solutions of the following problems.

This method computes $\vec{y}^{(i+1)}$ from $\vec{y}^{(i)}$ in the following way:

$$\vec{k}_1 = \vec{f}(x_i, \vec{y}^{(i)})$$
$$\vec{k}_2 = \vec{f}(x_i + \frac{h}{2}, \frac{h}{2}\vec{k}_1)$$
$$\vec{k}_3 = \vec{f}(x_i + \frac{h}{2}, \frac{h}{2}\vec{k}_2)$$

$$\vec{k}_4 = \vec{f}(x_i + h, h\vec{k}_3)$$

$$\vec{y}^{(i+1)} = \vec{y}^{(i)} + \frac{h}{6}(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4).$$

This Runge-Kutta method can be easily programmed in *Mathematica* because it contains a special type of object for vectors and many functions for operations with them. If we use built-in functions SetAttributes, ClearAttributes, Flatten, Append, While and Module we get this *Mathematica*'s function for implementation of Runge-Kutta method:

```
RungeKuttaMethod[f_,{x_,x0_},{y_,y0_},b_,h_]:=
    Module[{k1,k2,k3,k4,
            xk=x0, yk=y0, res={ Flatten[{x0,y0}] } },
            SetAttributes[Rule, Listable];
          While[xk<b,
                 k1=f/.x->xk      /. y->yk;
                 k2=f/.x->xk+h/2 /. y->yk+h/2 k1;
                 k3=f/.x->xk+h/2 /. y->yk+h/2 k2;
                 k4=f/.x->xk+h   /. y->yk+h   k3;
                 yk=yk+h/6 (k1+2(k2+k3)+k4);
                 xk=xk+h;
                 res=Append[res, Flatten[{xk,yk}] ]
               ];
          ClearAttributes[Rule, Listable];
          res
          ]
```

THE SIMPLE PENDULUM

The pendulum should be thought of as a weight hung on a rigid rod of negligible mass from a pivot without friction in a medium which offers no resistance to things moving through it. The state of the motion at any time is defined completely by the position and velocity at that time. If we know those values we can calculate its position and velocity for any time in the future or past.

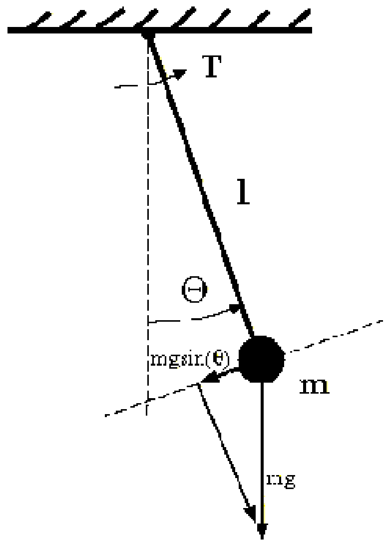

Figure 1

Using the second Newton law the differencial equation for a description of the movement of the pendulum is

$$ml\frac{d^2\Theta}{dt^2} = -mg\sin\Theta$$

where Θ is the angle of the pendulum from the vertical, m is the mass, l is the length, and g is the acceleration due to gravity. Introducing the angular velocity $\omega = \frac{d\Theta}{dt}$ we can rewrite this equation as a system of two first order ordinary differential equations,

$$\frac{d\Theta}{dt} = \omega$$
$$\frac{d\omega}{dt} = -\frac{g}{l}\sin\Theta \, .$$

Analytically the motion of a pendulum is given in an implicit form by elliptic integrals. We will solve the system of equations numerically using system *Mathematica*. The computed displacement of the value Θ of the pendulum versus time t is shown in the following Figure 2:

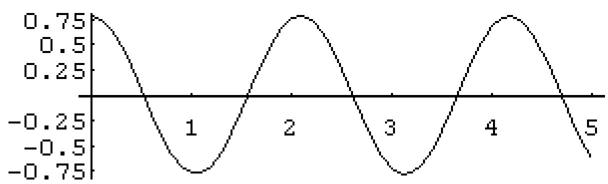

Figure 2

Now we relax some of our unrealistic requirements for our system and we allow an air consistence. We get the similar secondary-order equation:

$$lm\frac{d^2\Theta}{dt^2} = -mg\sin\Theta - c\frac{d\Theta}{dt}.$$

We can transform it into two first order equations with the variables $y_1 = \Theta$, $y_2 = \frac{d\Theta}{dt}$.

The equations become:

$$y_1' = y_2$$
$$y_2' = -\frac{g}{l}\sin y_1 - \frac{c}{ml}y_2 \, .$$

The displacement of the pendulum is in the Figure 3

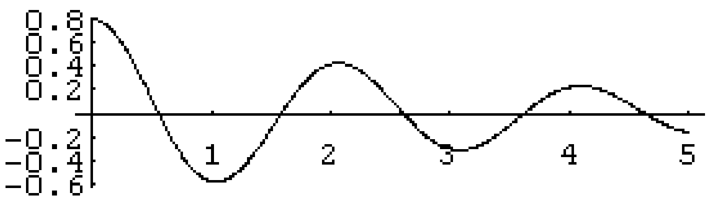

Figure 3

MOVEMENT OF PROJECTILE

Now we will study elements of a projectile motion with the resistence of air if it is shot with a initial velocity $\vec{v}_0$ and the mass is *m*. Using the second Newton law we again obtain the differential equations for *x* and *y* coordinates positions of the projectile:

$$m\frac{d^2x}{dt^2} = -c_1 v_x - c_2 v_x \sqrt{v_x^2 + v_y^2}$$

$$m\frac{d^2y}{dt^2} = -mg - c_1 v_y - c_2 v_y \sqrt{v_x^2 + v_y^2} \ .$$

The system can be rewritten to the four first-order equations with variables $y_1 = x$, $y_2 = \frac{dx}{dt}$, $y_3 = y$, $y_4 = \frac{dy}{dt}$.

We obtain

$$y_1' = y_2$$
$$y_2' = -\frac{c_1}{m} y_2 - \frac{c_2}{m} y_2 \sqrt{y_2^2 + y_4^2}$$
$$y_3' = y_4$$
$$y_4' = -g - \frac{c_1}{m} y_4 - \frac{c_2}{m} y_4 \sqrt{y_2^2 + y_4^2}$$

with the initial conditions $y_1(0) = 0$, $y_2(0) = v_{0x}$, $y_3(0) = 0$, $y_4(0) = v_{0y}$.

Fig. 4 shows the displacement of the projectile and Fig. 5 the graph of the velocity versus time ($v_{0x} = 100, v_{0y} = 80$).

Figure 4

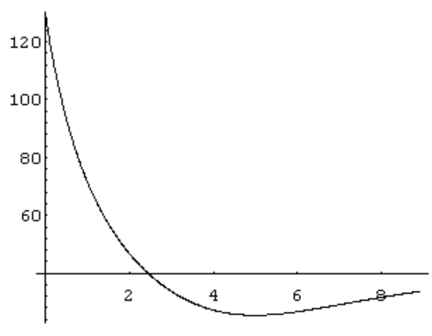

Figure 5

MOVEMENT OF A SATELITE OF PLANETS

In this part we find satelite orbits for the case of one planet and then for the case of a couple of planets. Using the second Newton law and Newton's classical law of gravity we get an equation for the description of orbits of satelite

$$m\vec{a} = -\kappa \frac{mM}{r^3}\vec{r},$$

where $\vec{a} = (\frac{d^2x}{dt^2}, \frac{d^2y}{dt^2})$ is the acceleration of the satelite, *m* the mass of it, *M* the mass of the planet, κ the universal constant of gravity and $\vec{r} = (x, y)$ is a position vector of the satelite. We get

$$\frac{d^2x}{dt^2} = -\kappa M \frac{x}{(x^2+y^2)^{3/2}}$$
$$\frac{d^2y}{dt^2} = -\kappa M \frac{y}{(x^2+y^2)^{3/2}},$$

which can be rewritten to the four first-order equations with variables $y_1 = x$, $y_2 = \frac{dx}{dt}$, $y_3 = y$, $y_4 = \frac{dy}{dt}$:

$$y_1' = y_2$$
$$y_2' = -\kappa M \frac{y_1}{({y_1}^2 + {y_3}^2)^{3/2}}$$
$$y_3' = y_4$$
$$y_4' = -\kappa M \frac{y_3}{({y_1}^2 + {y_3}^2)^{3/2}}$$

with $y_1(0) = 0$, $y_2(0) = v_{0x}$, $y_3(0) = H$, $y_4(0) = 0$.

Figure 6 contains the orbits of the satelite of one planet for two different initial velocities of the satelite (circular resp. parabolical orbits):

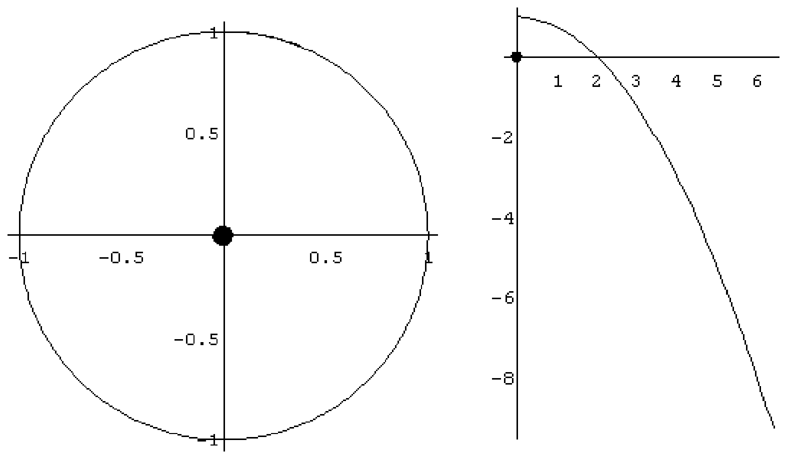

Figure 6

Similarly we get the equation for the orbit of the satelite of a couple of planets, which are in the distance D:

$$m\vec{a} = -\kappa \frac{mM_1}{r_1^3}\vec{r}_1 - \kappa \frac{mM_2}{r_2^3}\vec{r}_2,$$

where $\vec{r}_1 = (x - D, y)$ and $\vec{r}_2 = (x + D, y)$ and we can again rewrite it to four first-order differential equations. The computed orbits of the satelite of the couple of planets for different velocities of the satelite are in Figure 7.

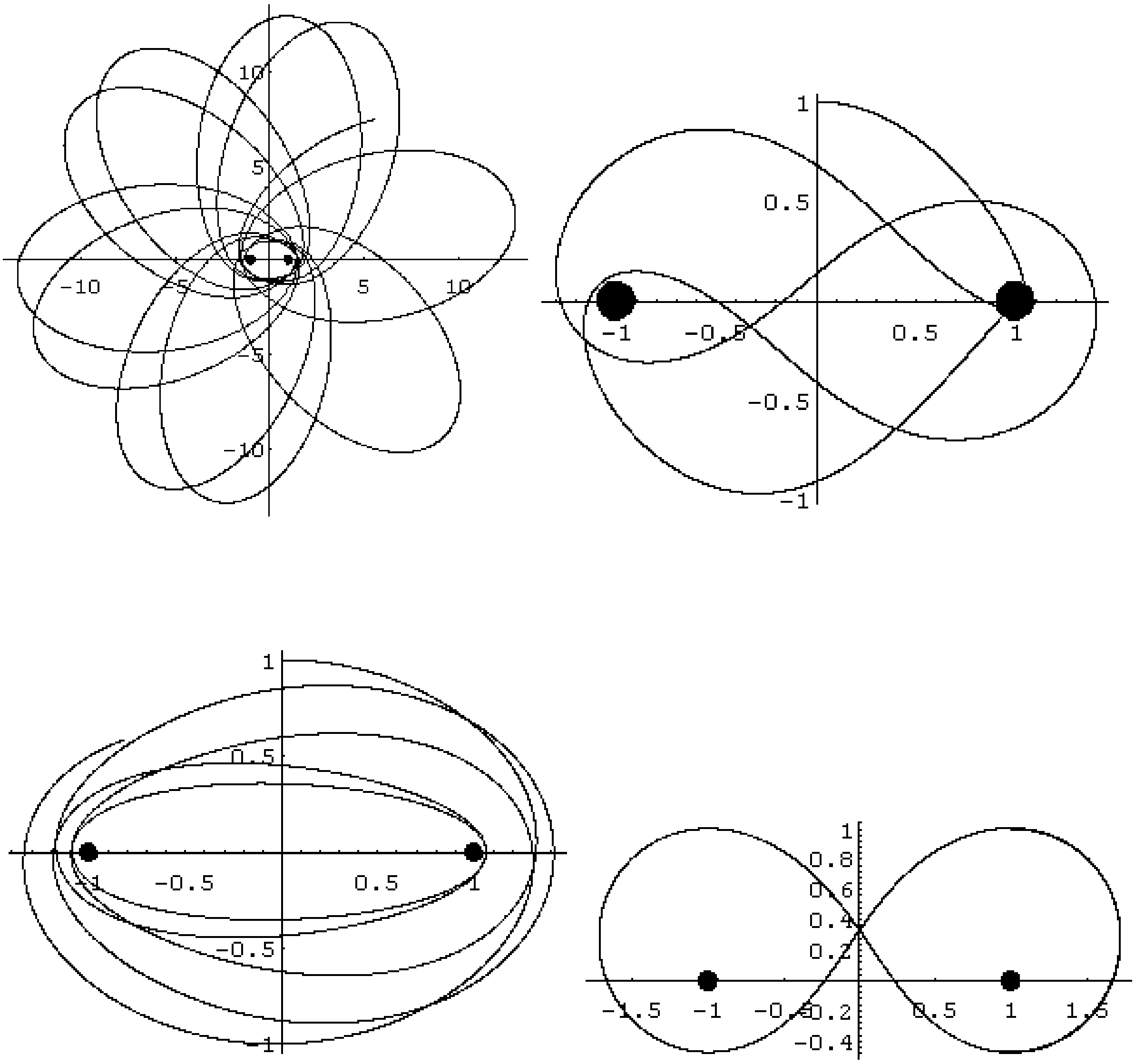

Figure 7

CONCLUSION

We showed an approach of finding solutions of several physical problems of using function `RungeKuttaMethod[]`, which we created. The system *Mathematica* contains from version 2.0 built-in command `NDSolve[]`, which can be used to solve out majority of the solved problems. Practical use of the theory of numerical methods is on this way more important for our students.

REFERENCES

Maeder R., Programming in Mathematica, Adison-Wesley Publishing Company, 1991

Wolfram S., Mathematica A system for doing mathematics by computer, Adison-Wesley Publishing, 1992

Pavel Trojovský
University of Hradec Králové
Department of Mathematics
Víta Nejedlého 573
500 03 Hradec Králové
Czech Republic
Email : pavel.trojovsky@uhk.cz

Jaroslav Seibert
University of Hradec Králové
Department of Mathematics
Víta Nejedlého 573
500 03 Hradec Králové
Czech Republic
Email : jaroslav.seibert@uhk.cz

Antonín Slabý
University of Hradec Králové
Department of Informatics
Víta Nejedlého 573
500 03 Hradec Králové
Czech Republic
Email : antonin.slaby@uhk.cz