SF2520 Exercise 1

Numerical Solution of Initial Value Problem

Oscar Johansson Elling 950101-6837 oelling@kth.se
Oscar Bergqvist 950214-2855 obergq@kth.se
Group SF2520-CE1 49

16/09/2018

1 Problem description

The exercise consists of three problems with associated sub-problems. In this section the goals of each problem is described.

1.1 Accuracy of a Runge-Kutta method

The goal is to numerically find the order of accuracy of the following Runge-Kutta method. u_k is the numerical approximation of u at time t_k, f is the time derivative of u, which can be derived from the implemented differential equation.

$$\begin{aligned}
&\vec{k}_1 = \vec{f}(t_{k-1}, \vec{u}_{k-1}) \\
&\vec{k}_2 = \vec{f}(t_{k-1} + h, \vec{u}_{k-1} + h\vec{k}_1) \\
&\vec{k}_3 = \vec{f}(t_{k-1} + h/2, \vec{u}_{k-1} + h\vec{k}_1/4 + h\vec{k}_2/4) \\
&\vec{u}_k = u_{k-1} + \frac{h}{6}(k_1 + k_2 + 4k_3),\ t_k = t_{k-1} + h,\ k = 1, 2, ..., N.
\end{aligned} \tag{1}$$

This is done by plotting the error denoted by

$$e_N = y_N - y(1) \tag{2}$$

for number of steps $N = 10, 20, 40, 80, 160, 320$ in a log-log plot. Here y_N is the Runge-Kutta estimation of y using number of steps N at $t = 1$ and $y(1) = \frac{du_1}{dt}(1)$ is the real value of y_N at $t = 1$. $y(1)$ is approximated by y_{320}. The implementation is done using pol's equation

$$\begin{aligned}
&\frac{d^2y}{dt^2} + (y^2 - 1)\frac{dy}{dt} + y = 0 \\
&y(0) = 1 \\
&\frac{dy}{dt}(0) = 0.
\end{aligned} \tag{3}$$

1.2 Stability investigation of a Runge-Kutta method

In this problem the stability of the Runge-Kutta method is examined for constant and adaptive step size.

1.2.1 Constant step size

The problem is to find the smallest out of $N = 125, 250, 500, 1000, 2000$ number of steps needed for stability of the Runge-Kutta method 1 implemented on Robertson's problem

$$\begin{aligned}
\frac{dx_1}{dt} &= -k_1x_1 + k_2x_2x_3 \\
\frac{dx_2}{dt} &= k_1x_1 - k_2x_2x_3 - k_3x_2^2 \\
\frac{dx_3}{dt} &= k_3x_2^2 \\
x_1(0) &= 1 \\
x_2(0) &= 0 \\
x_3(0) &= 0.
\end{aligned} \tag{4}$$

Here $x_i(t)$ is the concentration of a substance i at time t, and k_i are some coefficients. The solution trajectory is plotted for the solution computed with the smallest step.

1.2.2 Adaptive step size using MATLAB built in functions

In this problem the step size is plotted against the relative tolerance used with the built in functions ode23 (for $t \in [0, 1]$) and ode23s (for $t \in [0, 1000]$). The relative tolerances are taken as $RelTol = 10^3, 10^4, 10^5$ and 10^6.

1.3 Parameter study of solutions of an ODE-system

In these problems solution trajectories are studied.

1.3.1 Particle flow past a cylinder

The purpose of the problem is to plot the flow of particles past a cylinder in the time interval $t = [0, 10]$. The position of a flow particle at time t is $(x(t), x(t))$ and obeys the equation

$$\begin{aligned}
\frac{dx}{dt} &= 1 - \frac{R^2(x^2 - y^2)}{(x^2 + y^2)^2} \\
\frac{dy}{dt} &= -\frac{2xyR^2}{(x^2 + y^2)^2}
\end{aligned} \tag{5}$$

where $R = 2$ is the radius of the cylinder. The flow trajectories are plotted for four particles with the initial (x, y) positions $(-4, 0.2)$, $(-4, 0.6)$, $(-4, 1.0)$ and $(-4, 1.6)$.

1.3.2 Motion of a particle

In this problem the the motion of a particle thrown trough the air is modeled. This is described by the equations

$$\begin{aligned} \frac{d^2x}{dt^2} &= -k\frac{dx}{dt}\sqrt{(\frac{dx}{dt})^2 + (\frac{dy}{dt})^2} \\ \frac{d^2y}{dt^2} &= -9.82 - k\frac{dy}{dt}\sqrt{(\frac{dx}{dt})^2 + (\frac{dy}{dt})^2} \end{aligned} \tag{6}$$

where k models the air resistance and has the value of 0.065 and 0.020. The initial values are specified by Equation 7, where H takes the value of one and α takes the values of 30 , 45 and 60 degrees. The model is only valid until the particle hits the ground. The task is to simulate this . This is done by using the Runge-Kutta method described in 1.

$$\begin{aligned} \begin{bmatrix} x(0) & y(0) \end{bmatrix} &= \begin{bmatrix} 0 & H \end{bmatrix} \\ \begin{bmatrix} \frac{dx}{dt}(0) & \frac{dy}{dt}(0) \end{bmatrix} &= v_0 \cdot \begin{bmatrix} cos(\alpha) & sin(\alpha) \end{bmatrix} \end{aligned} \tag{7}$$

2 Analytic derivations

2.1 Accuracy of a Runge-Kutta method

In order to find a numeric solution to the differential equation 3. We need to find the appropriate system of differential equations.

$$\frac{d\vec{u}}{dt} = \vec{f}(\vec{u}, t) \tag{8}$$

To do this we introduce the following variables $u_1 = \frac{dy}{dt}$, $u_2 = y$. we create the vector $[u_1 u_2]$ and recreates Pol's differential equation.

$$\frac{du_1}{dt} = (1 - u_2^2) \cdot u_1 - u_2$$

$$\frac{du_2}{dt} = u_1$$

$$\frac{d}{dt}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} (1 - u_2^2) \cdot u_1 \\ u_1 \end{bmatrix} \tag{9}$$

The right hand sign of equation 9 is the $\vec{f}$ we sought at the beginning of this subsection.

2.2 Parameter study of solutions of an ODE-system

To get an numeric solution to equation 6 it is essential to find the corresponding $\vec{f}$. This is done in the same manner as in section 2.1. The following variables are introduced and defined $u_1 = \frac{dx}{dt}$, $u_2 = x$, $u_3 = \frac{dy}{dt}$ and $u_4 = y$

$$\frac{du_1}{dt} = -k \cdot u_1\sqrt{(u_1)^2 + (u_3)^2}$$

$$\frac{du_3}{dt} = -9.82 - k \cdot u_3\sqrt{(u_1)^2 + (u_3)^2}$$

This enables us to rewrite equation 6 in the following way

$$\frac{d}{dt}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} -k \cdot u_1\sqrt{(u_1)^2 + (u_3)^2} \\ u_1 \\ -9.82 - k \cdot u_3\sqrt{(u_1)^2 + (u_3)^2} \\ u_3 \end{bmatrix} \tag{10}$$

The right hand side of equation 10 is the desired $\vec{f}$.

3 Results

Figure 1 was linear and had the incline of 3. The linear behavior in a loglog plot is expected from the theoretical error dependence $c|h|^p$.

The step size taken by method ode23 and ode23s differed greatly (see Figure 5 and 6). The smallest number of steps that insured numerical stability for the Runge-Kutta method applied to Robertson's problem are 1000.

4 Discussion

Figure 1 implies that the accuracy of the Runge-Kutta method 1 is 3. The acurracy is indentified as p in

$$log(e_N) = log(c) + p \cdot log(h). \tag{11}$$

which is the slope in Figure 1. From Figure 5 and Figure 6 we see that the stiff ODE solver ode23s is more suited to solve Robertson's problem, enabling much bigger step sizes. This indicates that Robertson's problem is stiff.

In problem 2.1 we can conclude that the smallest number for steps required for stability is 1000, which is quite small. This also indicated that Robertson's problem is stiff which is in agreement with the previous conclusion.

The numerical results in problem 3 is in accordance with the expected behaviour of particles subject to the imposed conditions.

5 Conclusion

1. The accuracy of the Runge-Kutta method, Equation 1, is 3.

2. In problem 2.1 the smallest number of steps required for stability is 1000.

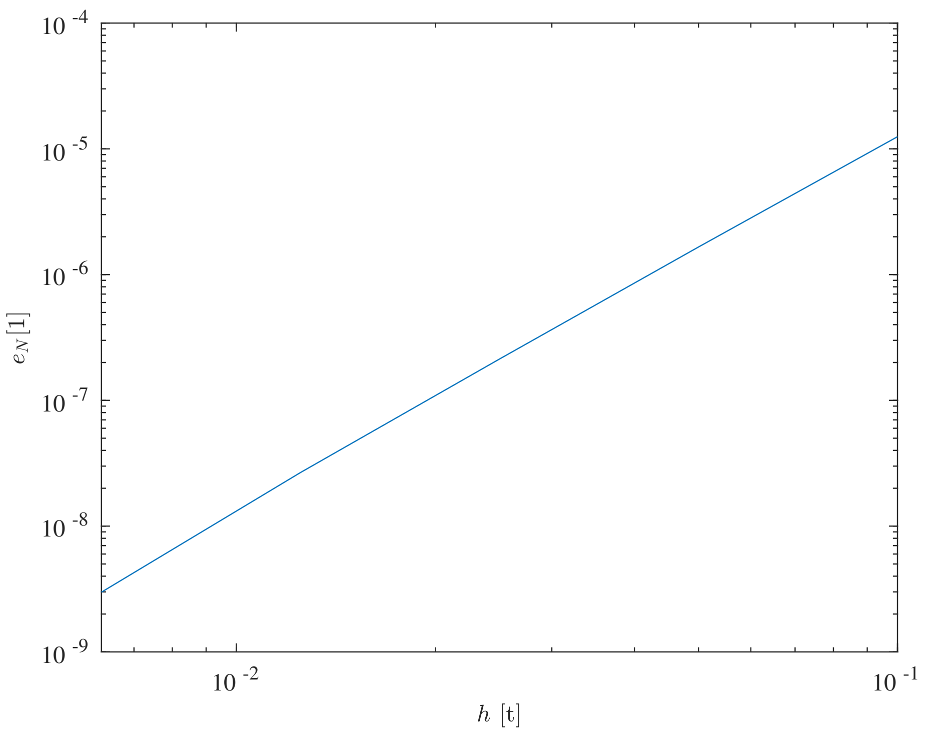

Figure 1: A log log plot of the error of our simulation of Pol's differential equation at time 1 as a function of step length h

3. The built in Matlab solver ode23s is able to use bigger step sizes then the function ode23. Thus it is more useful for stiff problems such as Robertson's problem.

4. The numerical predictions of the systems in problem three, behaved as expected.

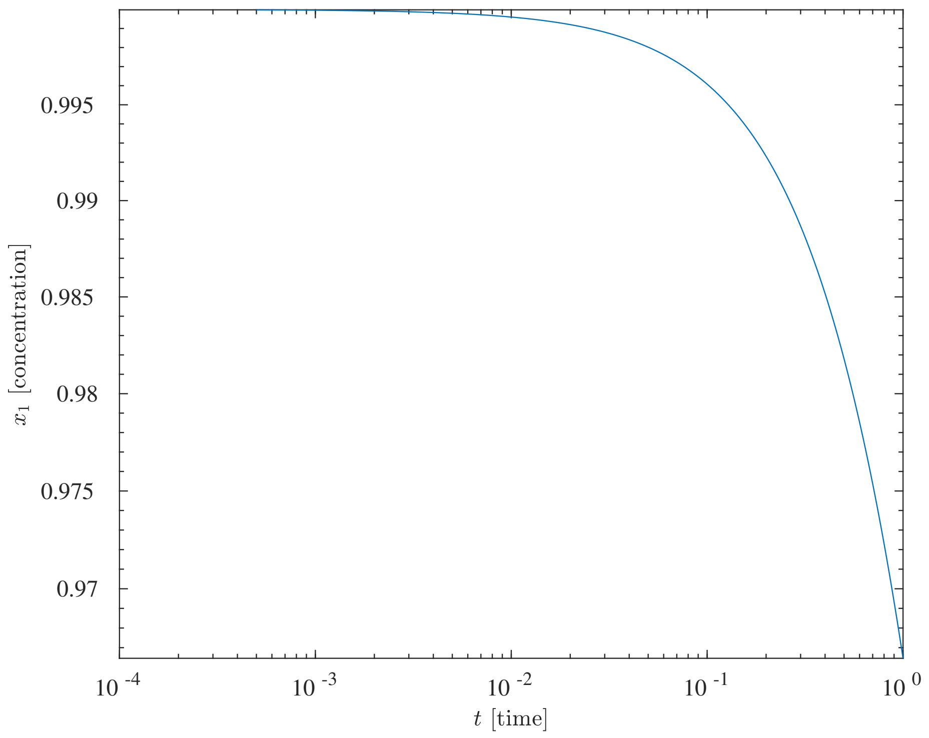

Figure 2: Robertson's problem simulated with 1000 steps between time 0 and 1

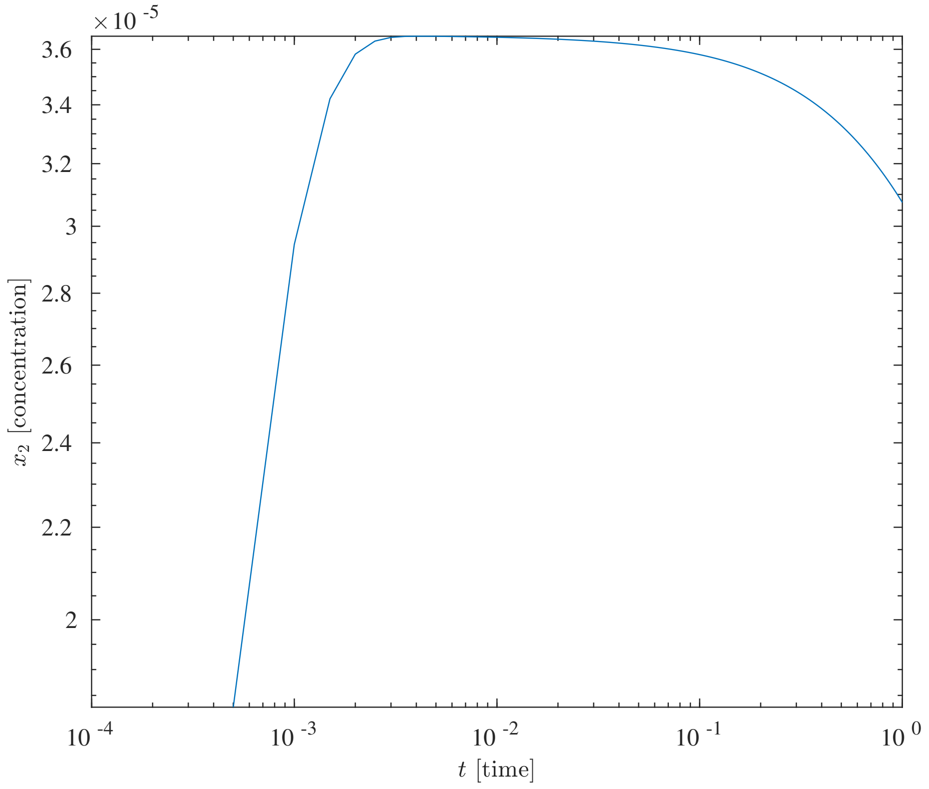

Figure 3: Robertson's problem simulated with 1000 steps between time 0 and 1

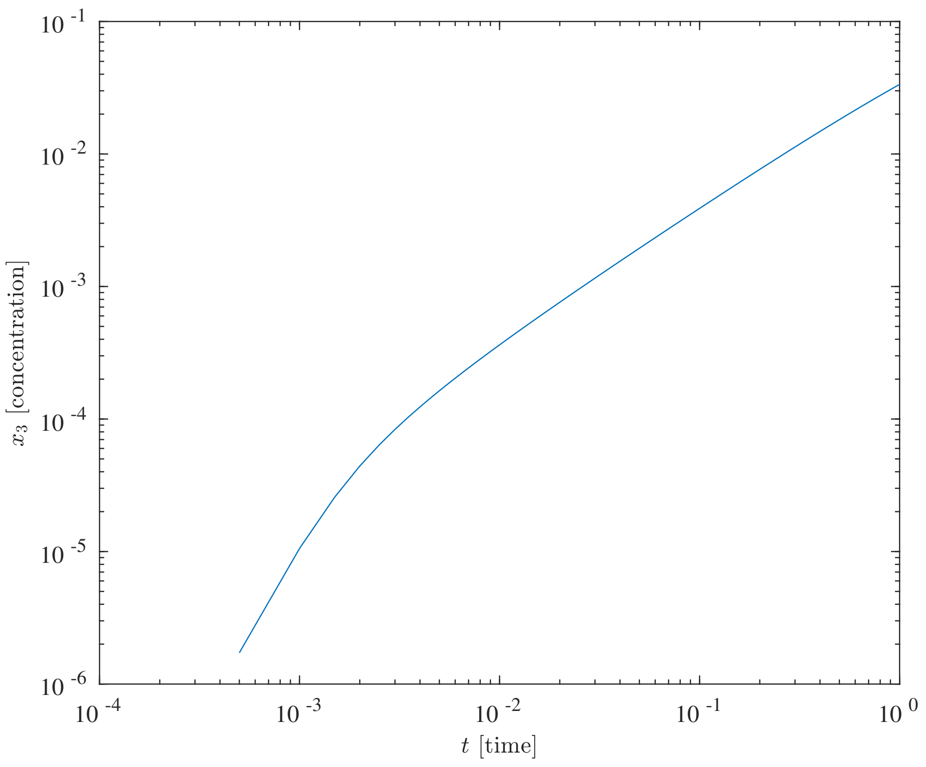

Figure 4: Robertson's problem simulated with 1000 steps between time 0 and 1

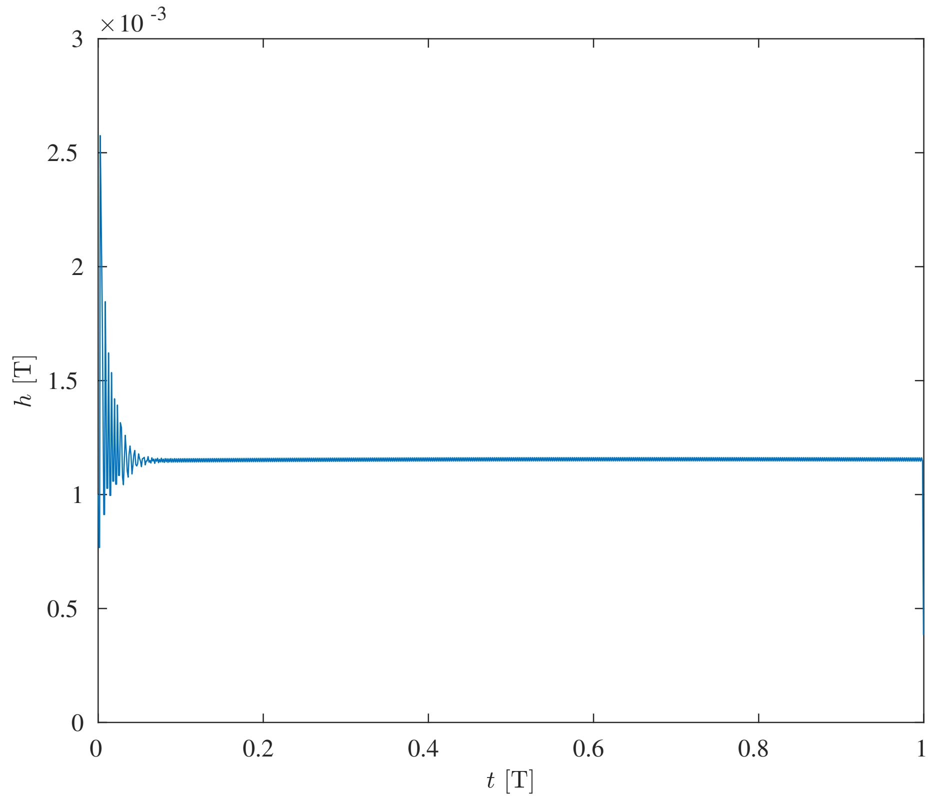

Figure 5: The step size used by Matlab's method ode23 (applied to Robertson's problem) as a function of time

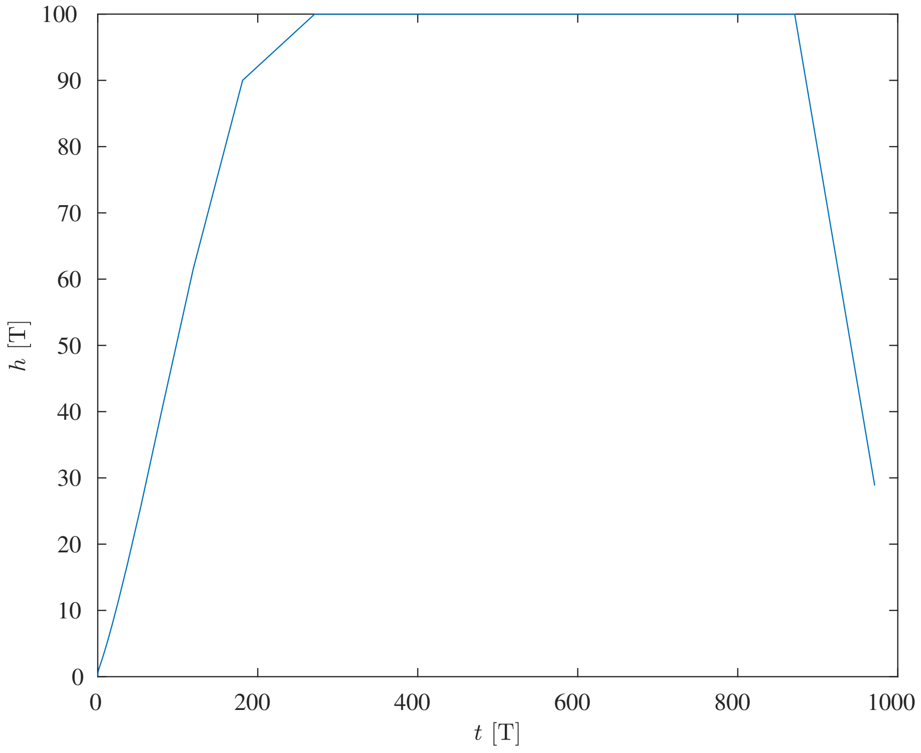

Figure 6: The step size used by Matlab's method ode23s (also applied to Robertson's problem) as a function of time

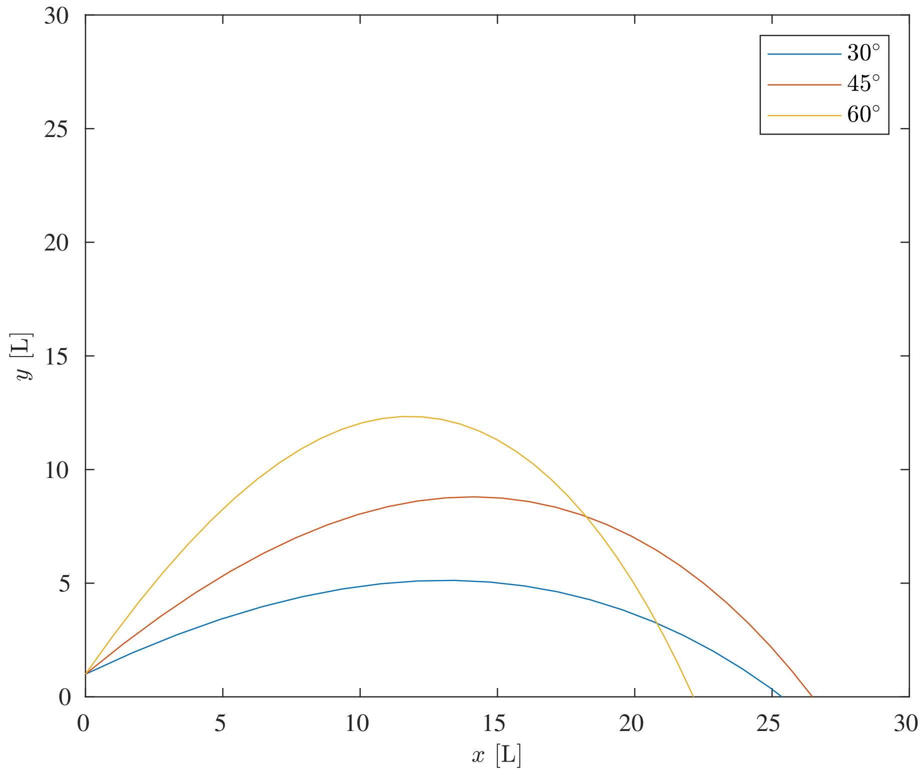

Figure 7: Particles thrown with drag coefficient 0.020

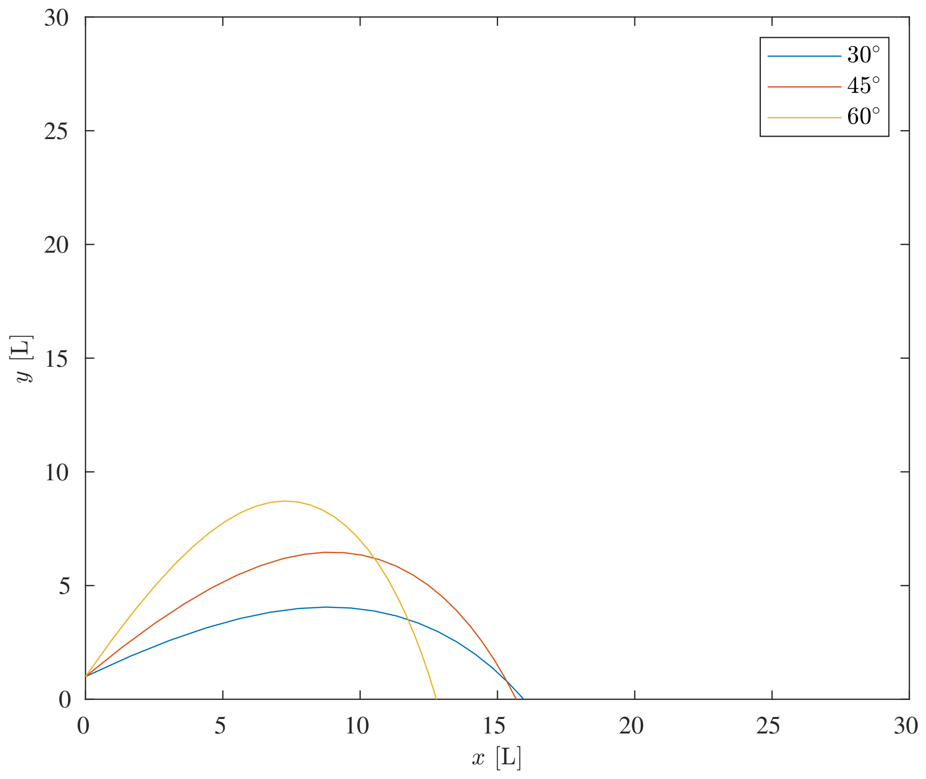

Figure 8: Particles thrown with drag coefficient 0.065