Demo 1: Introduction to Nonlinear Systems and a Brief Scilab Tutorial

In this demo you will be introduced to some basic concepts and terminology that are commonly encountered in the analysis of nonlinear systems. You will also learn how to use Scilab, with particular emphasis on commands related to solving differential equations and plotting the solutions.

Some Basic Concepts

Nonlinear systems are typically described using one or more differential equations, whose variables are referred to as *states*. What these variables actually represent varies from one system to another. In electric circuits, the states could be the capacitor voltages and inductor currents; in mechanical models, it could be positions and velocities, and so on. The important thing to remember is that in all these cases, the state variables constitute the *minimal* amount of information that is needed to fully describe the system.

The following pair of differential equations describes a typical nonlinear dynamic system

$$\dot{x}_1 = -3x_1^2 + 2x_1x_2 + x_2^2$$
$$\dot{x}_2 = \sin x_2 + x_1x_2$$

in which x_1 and x_2 represent the states, and

$$\dot{x}_1 = \frac{dx_1}{dt} \quad \text{and} \quad \dot{x}_2 = \frac{dx_2}{dt}$$

denote their derivatives over time. This system is *nonlinear* because the functions on the right hand side involve squares and products of the state variables, as well as a sinusoidal function of x_2. An example of a *linear* system would be

$$\dot{x}_1 = -3x_1 + 2x_2$$
$$\dot{x}_2 = x_1 - 4x_2$$

where all the terms on the right hand side involve only x_1 and x_2 (possibly multiplied by constants).

In analyzing dynamic systems (both linear and nonlinear), we are particularly interested in finding *constant* solutions to the equations that describe them. Such solutions are known as the *system equilibria*, and they can be computed by solving a system of algebraic equations which are obtained by setting all the derivatives to zero (since the derivative of a constant solution necessarily equals 0). In the example shown above, the equilibria could be found by solving the system

$$0 = -3x_1^2 + 2x_1x_2 + x_2^2$$
$$0 = \sin x_2 + x_1x_2$$

To illustrate some of the key properties of system equilibria, in the following we will focus our attention on the simple first order model

$$\dot{x} = x^2 + x + p$$

where p represents a *parameter*. This sort of description is quite common in practice, since many physical systems have parameters that can take on a range of different values. The equilibria of this system can be found by solving the quadratic equation

$$0 = x^2 + x + p$$

In this case we actually have an analytic solution of the form

$$x_{1,2} = \frac{-1 \pm \sqrt{1 - 4p}}{2}$$

but this a rather unusual situation. In most cases, the algebraic equations need to be solved numerically, and this can often be quite challenging.

The system that we are currently considering has two important features that are typical of many nonlinear models:

1. It has *multiple* equilibria.

2. For some values of parameter p, there is *no equilibrium at all* (this is the case when $p > 1/4$, since equilibria cannot be complex numbers).

Contrast this to the linear system shown below

$$\dot{x}_1 = -x_1 + 2x_2$$
$$\dot{x}_2 = 3x_1 - 4x_2 - p$$

whose equilibria can be found by solving

$$0 = -x_1 + 2x_2$$
$$0 = 3x_1 - 4x_2 - p$$

or equivalently

$$\begin{bmatrix} -1 & 2 \\ 3 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ p \end{bmatrix}$$

It is not difficult to see that such a system has a *unique* solution

$$x^e = \begin{bmatrix} p \\ p/2 \end{bmatrix}$$

for any choice of p. Note that this solution is *always*

$$x^e = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

when parameter p is set to zero (in such cases, we say that the system has an equilibrium at the *origin*).

An important characteristic of any system equilibrium is its *stability*. A stable equilibrium acts like a sort of "magnet" which attracts a range of solutions that start from different initial conditions. For that reason, it is commonly referred to as an *attractor*. An unstable equilibrium, on the other hand, *repels* solutions that originate in its vicinity (and is therefore referred to as a *repeller*). There are a number of analytic techniques for determining whether an equilibrium is stable or unstable. We will not examine these techniques in any detail, and will instead draw our conclusions from numerical simulations (which will be performed using Scilab).

A Brief Guide to Scilab

Our principal objective in the following will be to learn how to solve a system of differential equations numerically. For that purpose, we will initially focus on the system

$$\dot{x} = x^2 + x + p$$

which we already considered in the previous section. The Scilab differential equation solver requires that you describe the right hand side of the equation in a separate file. In order to do that, click on the second icon from the left, which will open a new file in SciNotes. Type

```
function  y = dem1(t, x, p)
    y = x^2+x+p;
endfunction
```

Then click on Execute, and choose Save and Execute from the drop down menu. This will save your file as dem1.sci, and will also make it available to Scilab for the purpose of solving the differential equation.

The function that you created returns

$$y = x^2 + x + p$$

given specific values for t, x and p (which represent the *input variables*). Although time t doesn't appear explicitly in this expression, we included it because Scilab's general format is designed to accommodate time varying equations as well. An example of such an equation would be

$$\dot{x} = x^3 + x^2 + p + \cos 2t$$

where time shows up in the cosine term.

In order to solve our differential equation, we need to specify an initial condition x_0, an initial time t_0 and a value for parameter p. We also need to indicate the points in which you want the solution computed. If we choose $x_0 = -1$, $t_0 = 0$ and $p = -2$, and decide to solve the equation on the interval $0 \leq t \leq 5$ with points that are separated by $\Delta t = 0.01$, you should type the following four commands in the workspace:

```
x0 = -1;
t0 = 0;
p = -2;
t = 0:0.01:5;
```

Placing the semicolon after each command is a good practice in general, since it prevents the program from displaying unnecessary intermediate results on the screen.

You are now ready to use Scilab's differential equation solver. You can do that by typing the command

y=ode(x0,t0,t,list(dem1,p));

The term dem1 in this expression indicates the name of the .sci file that describes the right hand side of your equation, and p refers to the parameter that appears in this description. You can plot the solution that you obtained using the command

plot(t, y)

which opens a new graphics window and displays the curve that was computed numerically.

Plotting Options

To see what plotting options are available in Scilab, click on Edit in the graphics window, and then select Axes Properties from the pull down menu. This will open the Axes Editor, which allows you to modify the way your plot is displayed. Suppose, for example, that you would like a more detailed view of your solution on the interval $0 \leq t \leq 1$. To do that, change the data bounds in the Axis Options from [0 5] to [0 1]. When you click on your original plot, this will produce the graph shown in Figure 1.

You can label the x-axis by adding the appropriate text in the Label Options (the text should be inserted between the two quotation marks). You can also create a grid, by changing the grid color in the Axis Options from -1 to 1. Note that these kinds of changes can be applied to the y-axis as well - all you need to do is select Y from the Object Properties menu, and repeat the steps described above.

Plots created in Scilab can be exported in a number of ways. To create a pdf or encapsulated post script (.eps) copy, click on the File option in the graphics window, select Vectorial Export, and then choose the appropriate file format. If you prefer to create a jpeg version of the file, select Export To, and then pick this format. The resolution of the graphics is somewhat reduced when you convert the plot to a jpeg file, but this may be necessary in cases when the number of data point is large.

Equilibria and Stability

In this section, we will be interested in analyzing how system

$$\dot{x} = x^2 + x + p$$

behaves when it starts from different initial conditions (we will continue to use $p = -2$ in these calculations). In order to do that, we will consider $x_0 = 0.99$, $x_0 = 0$, $x_0 = -1$ and $x_0 = -4$, and type in the following set of commands:

```
x0 = 0.99;
y1 = ode(x0,t0,t,list(dem1,p));
x0=0;
y2=ode(x0,t0,t,list(dem1,p));
x0=-1;
y3=ode(x0,t0,t,list(dem1,p));
```

```
x0=-4;
y4=ode(x0,t0,t,list(dem1,p));
```

In order to plot all four functions on the same graph, type

```
plot(t,y1,t,y2,t,y3,t,y4)
```

You will obtain a graph like the one shown in Figure 2, which suggests that equilibrium $x^e = -2$ is an *attractor*. To get a numerical value for this equilibrium without having to read it from the graph, type in

```
size(y1)
```

This will give you the number of rows and columns in y1 (in this case, y1 has 1 row and 501 columns). The number of columns tells us that y1 has 501 elements, which correspond to values of $x(t)$ evaluated at $t = 0, 0.01, 0.02, \ldots, 4.99, 5.00$. The last of these values represents a good approximation for the attractor to which this solution has converged. You can verify this by typing in

```
y1(501)
```

which produces -1.9999995. This value is obviously very close to the exact equilibrium value (which is $x^e = -2$).

It is interesting to note that the graph in Fig. 2 tells us nothing at all about the other equilibrium - we know that there is one at $x^e = 1$, but this value was computed *analytically*. To see why this equilibrium is "invisible", let us solve our system for initial conditions $x_0 = 0.99$, $x_0 = 1$, and $x_0 = 1.00001$ on the interval $0 \leq t \leq 4$. The following sequence of commands yield the three solutions that we are interested in (we will denote them z_1, z_2 and z_3, in order to distinguish them from our previous solutions):

```
x0=0.99;
z1=ode(x0,t0,t,list(dem1,p));
x0=1;
z2=ode(x0,t0,t,list(dem1,p));
x0=1.00001;
z3=ode(x0,t0,t,list(dem1,p));
```

The command

```
plot(t,z1,t,z2,t,z3)
```

produces the graph shown in Fig. 3, which indicates that $x^e = 1$ is an *unstable* equilibrium. We can obtain this solution *only* if we set $x_0 = 1$ - any deviation from this value will lead to a different attractor, or to divergence. This is easily verified by observing that the solution originating at $x_0 = 0.99$ is attracted to $x^e = -2$, while the other one appears to increase unboundedly. To get a better idea of what happens to the solution that corresponds to $x_0 = 1.00001$, it is helpful to expand our simulation interval from $0 \leq t \leq 4$ to $0 \leq t \leq 4.2$. The graph in Fig. 4 shows that this solution "blows up" very rapidly.

What can we conclude from this analysis? Perhaps the most important insight is that numerical simulation can reveal the *stable* equilibria of the system, but not the *unstable* ones. It can also give us an idea of the range of initial conditions that converge to a particular stable

equilibrium (the so-called *region of attraction*). In our example, it would seem that solutions with $x_0 < 1$ are attracted to $x^e = -2$, while those with $x_0 > 1$ diverge to infinity. This, however, is only a rough estimate of the region of attraction - a more precise evaluation would require solving the equation for a much larger set of initial conditions.

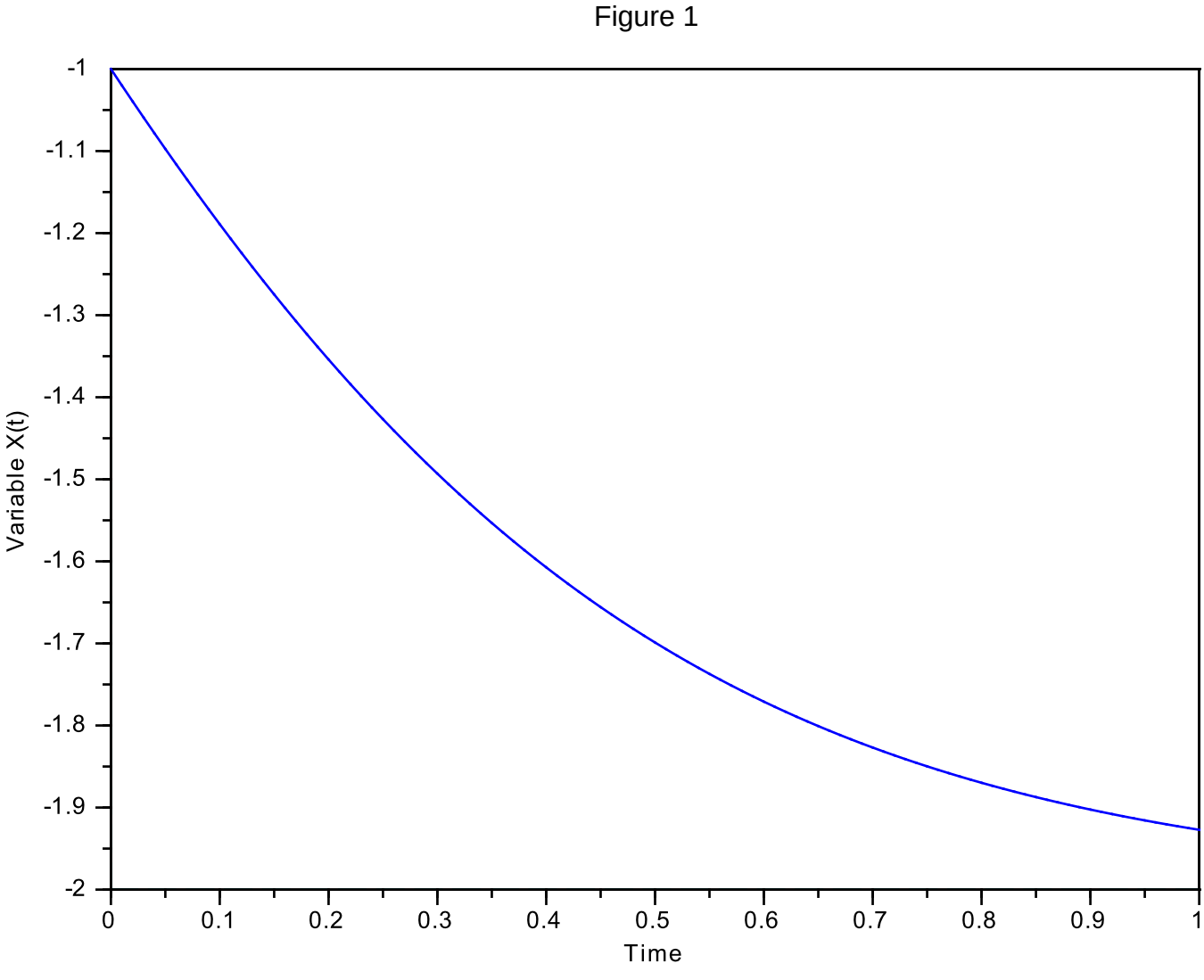


Figure 1
-1
-1.1
-1.2
-1.3
-1.4
-1.5
-1.6
-1.7
-1.8
-1.9
-2
Variable X(t)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time

Figure 2

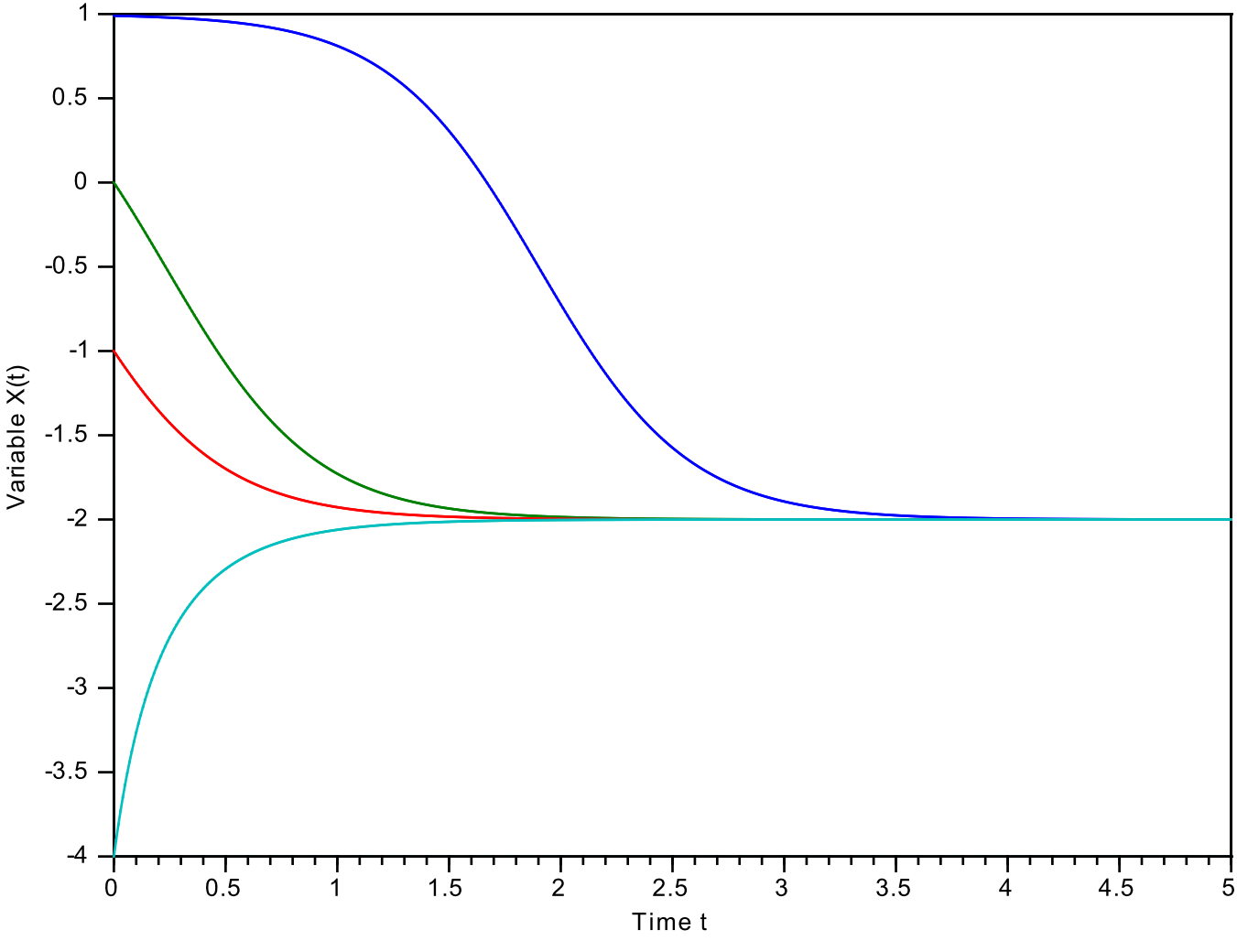


Figure 3

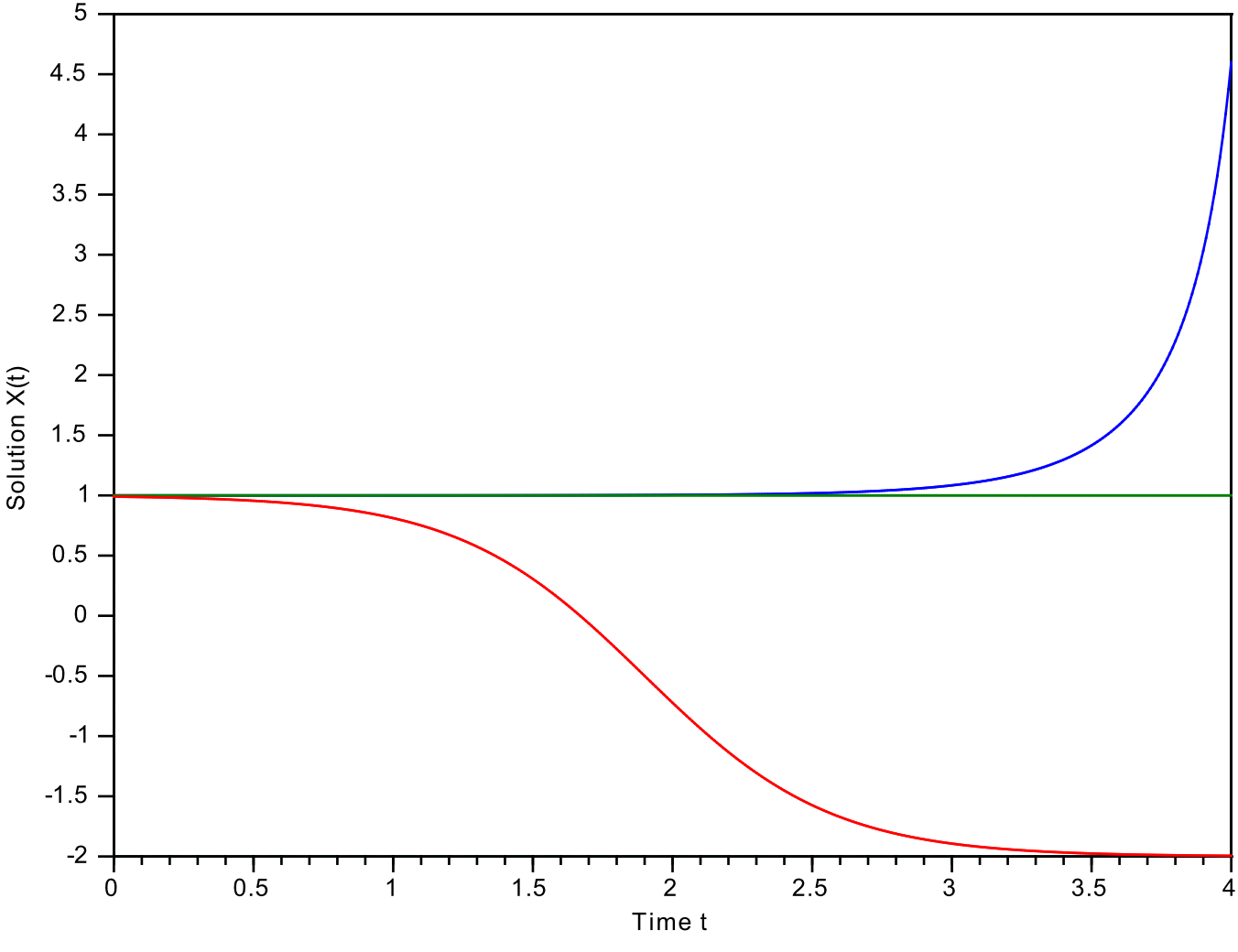
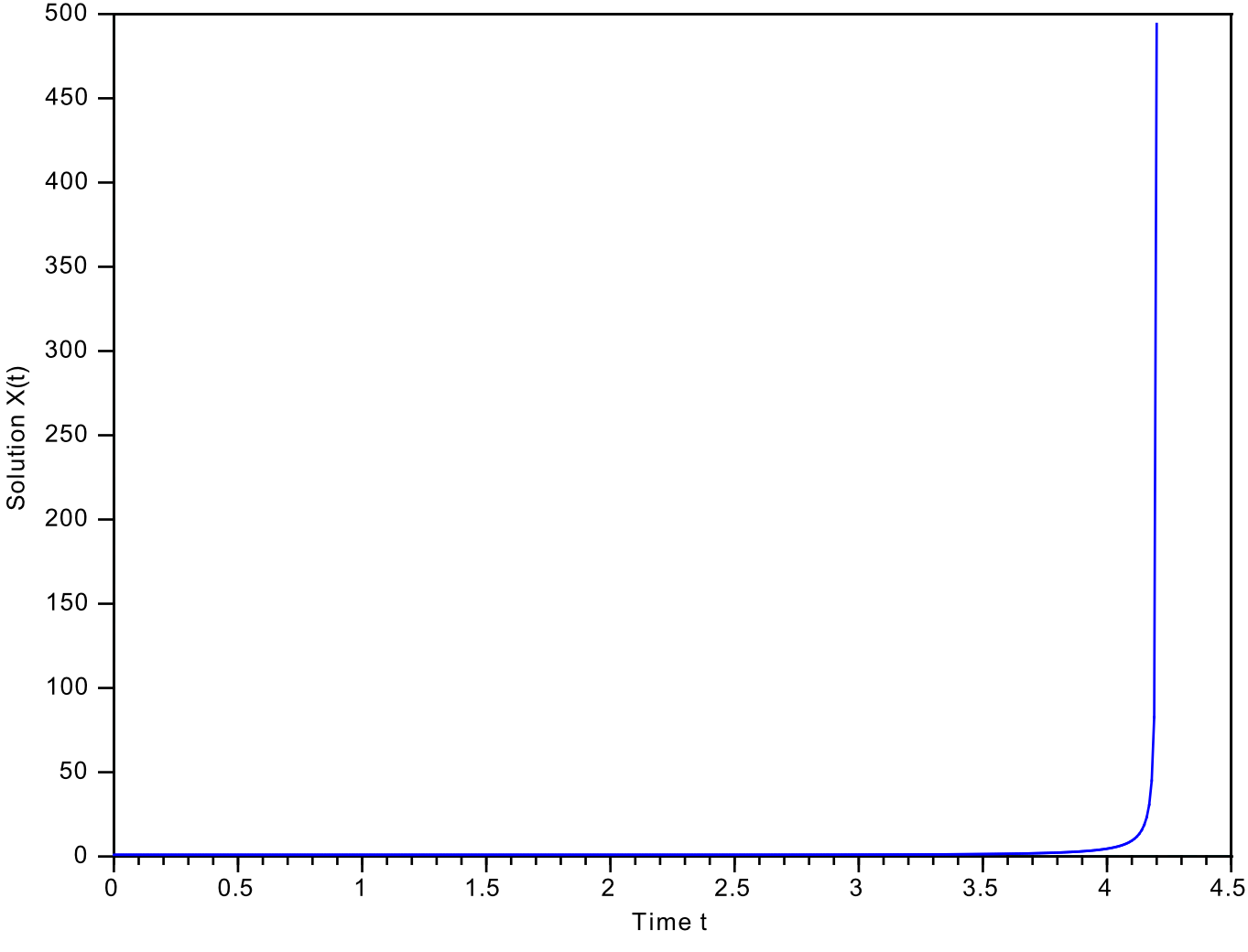

5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
Solution X(t)
0
0.5
1
1.5
2
2.5
3
3.5
4
Time t

Figure 4


500
450
400
350
300
250
200
150
100
50
0
Solution X(t)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Time t