Vector Fields and Solutions to Ordinary Differential Equations using `MATLAB`/*Octave*

Andreas Stahel

15th December 2017

Contents

In these notes you find the methods to generate vector fields with the help of `MATLAB`/*Octave*. In addition one of the methods to solve ordinary differential equations is illustrated. The notes assume that you have a recent version of *Octave* or `MATLAB` installed.

1 Vector field for the logistic equation

A vector field in the plane $\mathbb{R}^2$ is determined by a vector function

$$\vec{F}(\vec{x}) = \left(\begin{array}{c} F_1(x_1, x_2) \\ F_2(x_1, x_2) \end{array} \right)$$

At each point $\vec{x} \in \mathbb{R}^2$ the vector $\vec{F}(\vec{x})$ is attached. One of the possible applications of vectors fields is the visualization of solution of ordinary differential equations.

The vector field for the logistic differential equation

$$\frac{d}{dt} x(t) = 2\, x(t) - x^2(t)$$

is given by

$$\vec{F}(t, x) = \left(\begin{array}{c} 1 \\ 2\, x - x^2 \end{array} \right)$$

To visualize this field we have to

- choose the domain to be used

- define the functions for the vector field

1

- choose how many vectors to draw

- compute the vectors at the chosen points

- generate the graphic on screen

- save the result in a file, either as PNG or PDF

The above tasks can be performed with *Octave* or MATLAB, as shown in the code below with the resulting Figure 1(a).

```
t = 0:0.5:5; %% choose the time values
x = 0:0.2:3; %% choose the x-values

%% define the function
logistic = @(t,x)2*x-x.^2;

%% create the matrices for the vector field
nt = length(t); nx = length(x);
V1 = ones(nx,nt); V2 = zeros(nx,nt);

%% compute the vector field
for j=1:nx
  V2(j,:) = logistic(0,x(j));
end%for

%% display the vector field
figure(1);
quiver(t,x,V1,V2);
axis([0,5,0,3]);
xlabel('time t'); ylabel('population p')
grid on
%% print -dpng Logistic.png
%% print -dpdfwrite Logistic.pdf
```
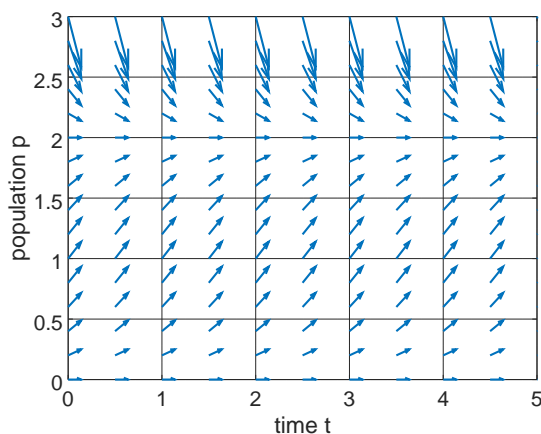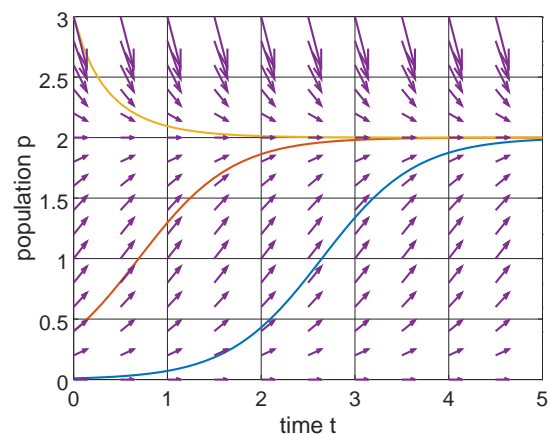

(a) vector field

(b) vector field and solutions

Figure 1: Vector field and solutions for the logistic equation

2   Solutions of ordinary differential equations with `ode45()`

Octave and `MATLAB` provide a selection of commands to solve ordinary differential equations, including systems. In this section we use `ode45()` exclusively to generate solutions for the differential equations arising from the above vector fields.

To solve the initial value problem

$$\frac{d}{dt}\, x(t) = 2\, x(t) - x^2(t) \quad \text{with} \quad x(0) = 0.2$$

we proceed as follows.

- Choose the values of time t for which the solution is to be computed.

- Compute the solution with the command `ode45()`.

- We may compute more solution by using different initial conditions, e.g. $x(0) = 0.4$ and $x(0) = 2.0$.

- Plot the solutions.

- The solutions and the vector field can be shown in one graphic.

Find the result in Figure 1(b).

```
%%% solving the ODE
T = 0:0.1:5;

[T1,X1] = ode45(logistic,T,0.01);
[T2,X2] = ode45(logistic,T,0.4);
[T3,X3] = ode45(logistic,T,3.0);

figure(2); clf
plot(T1,X1,T2,X2,T3,X3);
grid on
axis([0,5,0,3])
xlabel('time t'); ylabel('population p')
hold on
quiver(t,x,V1,V2)
hold off
```

3   Vector field for the equation of a damped pendulum

For autonomous differential equations the first component of the vector field equals 1. For general problems this is not the case. As an example we convert the differential equation for a damped pendulum

$$\ddot{y}(t) = -k\, y(t) - \alpha\, \dot{y}(t)$$

into a system of differential equations of order one

$$\frac{d}{dt} \begin{pmatrix} y(t) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ -k\, y(t) - \alpha\, v(t) \end{pmatrix}$$

and examine the resulting vector field. With the numerical values $k = 1$ and $\alpha = 0.1$ we have to examine the vector field

$$\vec{F}(y, v) = \begin{pmatrix} v \\ -y - 0.1\, v \end{pmatrix}$$

The code to be used is very similar to the logistic equations and thus shown without any further explanation.

```
y = −1:0.1:1; %% choose the y values
v = −1:0.1:1; %% choose the v−values
[y,v] = meshgrid(y,v);

%% compute the vector field
k = 1;   alpha = 0.1;
V1 = v; V2 = −k∗y − alpha∗v;

figure(1);
quiver(y,v,V1, V2)
grid on; axis([−1 1 −1 1])
xlabel('position x'); ylabel('speed v')
```

The only additional feature is the explicit choice of the domain shown in the graphic by the command
`axis()`.

As can be seen in Figure 2(a) the vectors at the origin have a small length and thus it is difficult to read of the direction. This can be improved by normalizing all the vectors to have length 1 and then rescale them to have a good length for visualization. This is done in the code below with the resulting Figure 2(b).
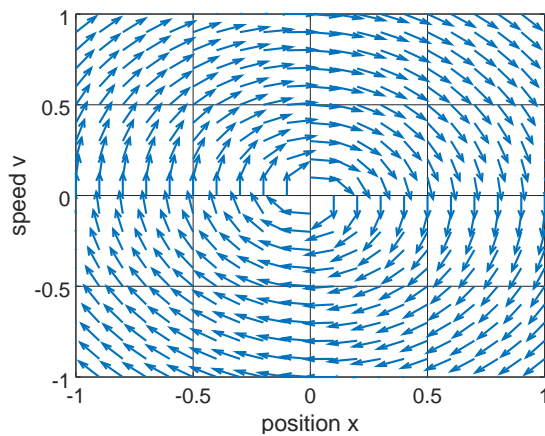
```
LengthVF = sqrt(V1.^2 + V2.^2);
figure(2);
quiver(y,v,V1./LengthVF, V2./LengthVF)
grid on; axis([−1 1 −1 1])
xlabel('position x'); ylabel('speed v')
```
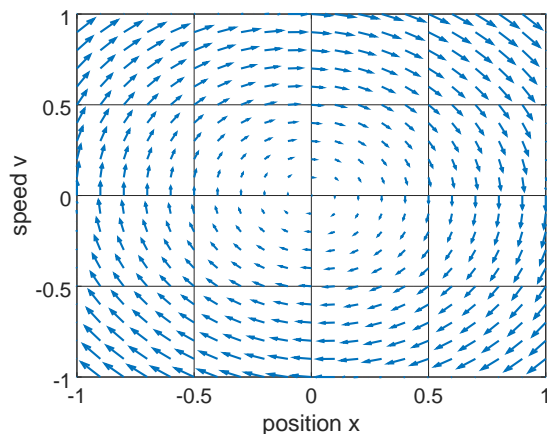

(a) regular vector field   (b) normalized vector field

Figure 2: Vector field for the pendulum equation

4   Solution to the pendulum equation

To determine solutions of the system of differential equations

$$\frac{d}{dt}\begin{pmatrix} y(y) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ -y(t) - 0.1\,v(t) \end{pmatrix} \quad \text{with} \quad \begin{pmatrix} y(0) \\ v(0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0.9 \end{pmatrix}$$

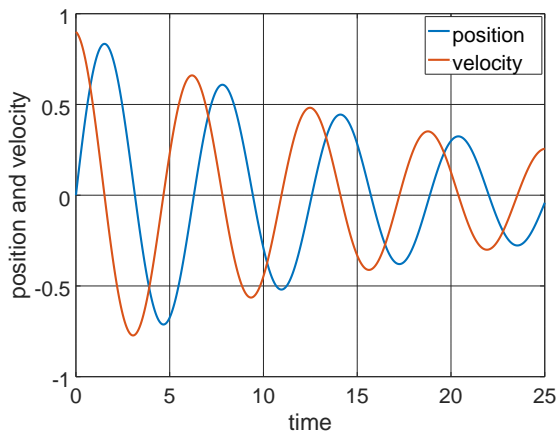we have to rewrite the function to be used by `ode45()`.

```
T = 0:0.05:25;
k = 1;  alpha = 0.1;
ODEPend = @(t,y)[y(2); -k*y(1)-alpha*y(2)];

[t,YV] = ode45(ODEPend,T,[0;0.9]);

figure(3);
plot(T,YV(:,1),T,YV(:,2));
legend('position','velocity')
grid on
xlabel('time'); ylabel('position and velocity');

figure(4); clf
quiver(y,v,V1,V2)
grid on
axis([-1 1 -1 1])
hold on
plot(YV(:,1),YV(:,2), 'r')
xlabel('position'); ylabel('velocity');
hold off
```
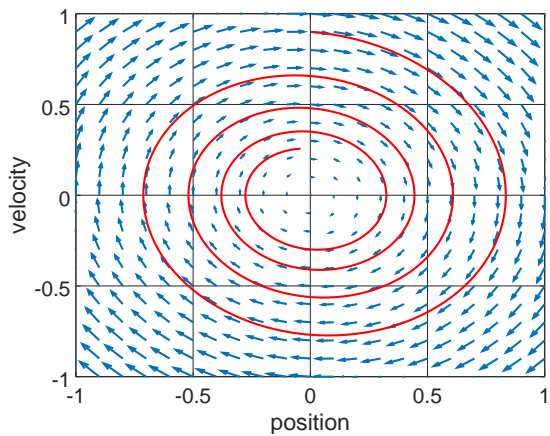

(a) position and velocity

(b) phase portrait

Figure 3: One solution and the vector field for the pendulum equation