CODEE Journal

Volume 9 Article 8

11-25-2012

Understanding Differential Equations Using Mathematica and Interactive Demonstrations

Paritosh Mokhasi

James Adduci

Devendra Kapadia

Follow this and additional works at: http://scholarship.claremont.edu/codee

Recommended Citation

Mokhasi, Paritosh; Adduci, James; and Kapadia, Devendra (2012) "Understanding Differential Equations Using Mathematica and Interactive Demonstrations," *CODEE Journal*: Vol. 9, Article 8.
Available at: http://scholarship.claremont.edu/codee/vol9/iss1/8

This Article is brought to you for free and open access by the Journals at Claremont at Scholarship @ Claremont. It has been accepted for inclusion in CODEE Journal by an authorized administrator of Scholarship @ Claremont. For more information, please contact scholarship@cuc.claremont.edu.

Understanding Differential Equations Using Mathematica and Interactive Demonstrations

Paritosh Mokhasi, James Adduci and Devendra Kapadia
Wolfram Research, Inc., Champaign, IL.

Keywords: *Mathematica*, Wolfram Demonstrations Project

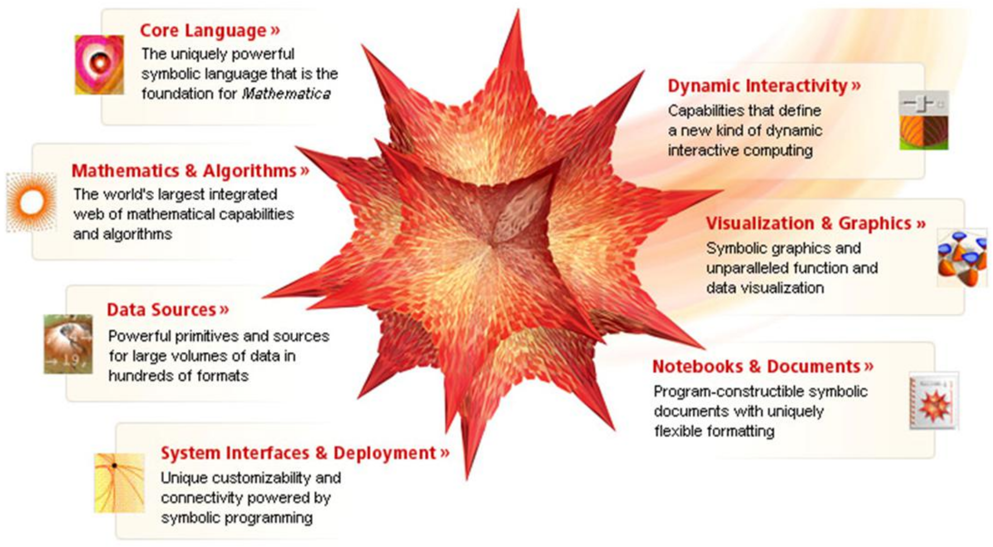
Manuscript received on May 24, 2012; published on November 25, 2012.

Abstract: The solution of differential equations using the software package *Mathematica* is discussed in this paper. We focus on two functions, DSolve and NDSolve, and give various examples of how one can obtain symbolic or numerical results using these functions. An overview of the Wolfram Demonstrations Project is given, along with various novel user-contributed examples in the field of differential equations. The use of these Demonstrations in a classroom setting is elaborated upon to emphasize their significance for education.

1 Introduction

Mathematica[1] is a powerful software package used for all kinds of symbolic and numerical computations. It has been available for around 25 years. *Mathematica* is sometimes viewed as a very sophisticated calculator useful for solving a variety of different problems, including differential equations. However, the use of the term "calculator" is a misnomer in the case of *Mathematica*. As shown in Figure 1, *Mathematica* has its own programming language and has sophisticated graphics and visualization capability which, combined with the use of dynamic interactivity, makes it a valuable tool for many professionals. One can access a large number of data sources from within *Mathematica* through Wolfram servers, ranging from real-time financial data to country data. The *Mathematica* front end allows one to generate professional-quality reports.

[1]*Mathematica* is a registered trademark of Wolfram Research, which holds the copyright to the *Wolfram Mathematica* software system.



Core Language »
The uniquely powerful symbolic language that is the foundation for *Mathematica*

Mathematics & Algorithms »
The world's largest integrated web of mathematical capabilities and algorithms

Data Sources »
Powerful primitives and sources for large volumes of data in hundreds of formats

System Interfaces & Deployment »
Unique customizability and connectivity powered by symbolic programming

Dynamic Interactivity »
Capabilities that define a new kind of dynamic interactive computing

Visualization & Graphics »
Symbolic graphics and unparalleled function and data visualization

Notebooks & Documents »
Program-constructible symbolic documents with uniquely flexible formatting

Figure 1: *Mathematica* as a single integrated system

The power of performing computations in *Mathematica* comes from the seamless integration between symbolic and numeric computations. Due to the integrated nature of the system, *Mathematica* functions tend to rely on each other to analyze and produce results in an efficient manner. The current version of *Mathematica*, Version 8.0.4, has over 3000 different functions. Figure 2 gives an understanding of the evolution of the number of functions in *Mathematica* since its inception back in the late 1980s.

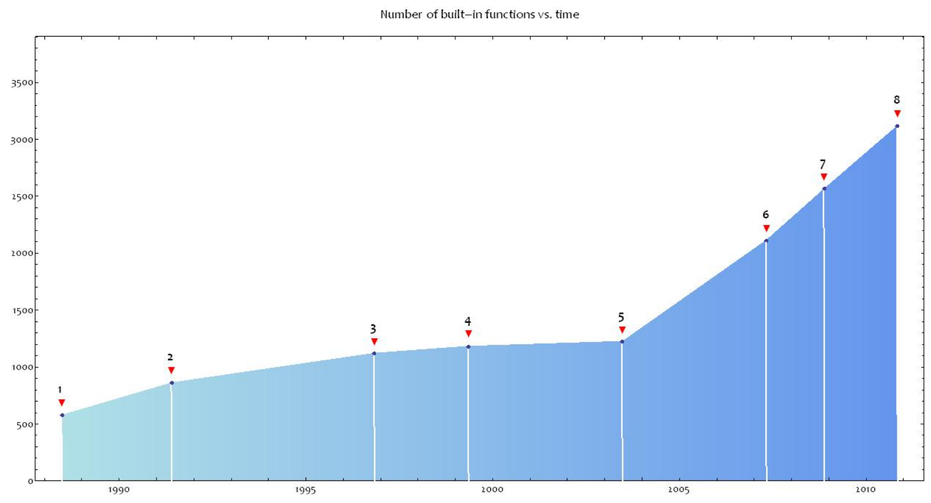


Figure 2: Graph of number of *Mathematica* functions as a function of time

Among the 3000+ functions in *Mathematica*, there are two major *superfunctions* that are used to solve differential equations: DSolve and NDSolve. The term "superfunctions" is used here because these two functions handle a large class of differential equations in a very unified way. The functions preprocess the differential equations, automatically decide what algorithms would be best suited for solving the system, and solve it without any further user interaction. The *Mathematica* function DSolve finds *symbolic* solutions to differential equations. This allows the user to obtain a closed-form solution when possible. In the absence of sufficient constraints, DSolve returns a family of solutions. The function NDSolve, on the other hand, finds *numeric* solutions to differential equations. Details of how DSolve and NDSolve work will be discussed in more detail in Section 2.

The Wolfram Demonstrations Project (http://demonstrations.wolfram.com) was conceived by Stephen Wolfram as a way to bring computational exploration to the widest possible audience. It is an open-code resource that uses dynamic computation to illuminate concepts in science, technology, mathematics, art, finance, and a remarkable range of other fields. Its daily growing collection of interactive illustrations is created by *Mathematica* users from around the world who participate by contributing innovative Demonstrations. The submissions undergo a review process before being accepted as part of the Demonstration project.

A Demonstration is an interactive visualization of a concept. Demonstrations can be about any topic. As you move a Demonstration's controls, you see a change in its output that helps you understand the concept being shown. At its core, a Demonstration is a small program created with a standard copy of *Mathematica*. Demonstration authors fill out an authoring template and upload it to our site. The Wolfram servers convert the filled-in template to an interactive Computable Document Format (CDF) object (http://www.wolfram.com/cdf). There are in early 2012 over 7500 different Demonstrations that have been submitted by the user community at large. The authors come from various backgrounds, ranging from high school students to research scientists.

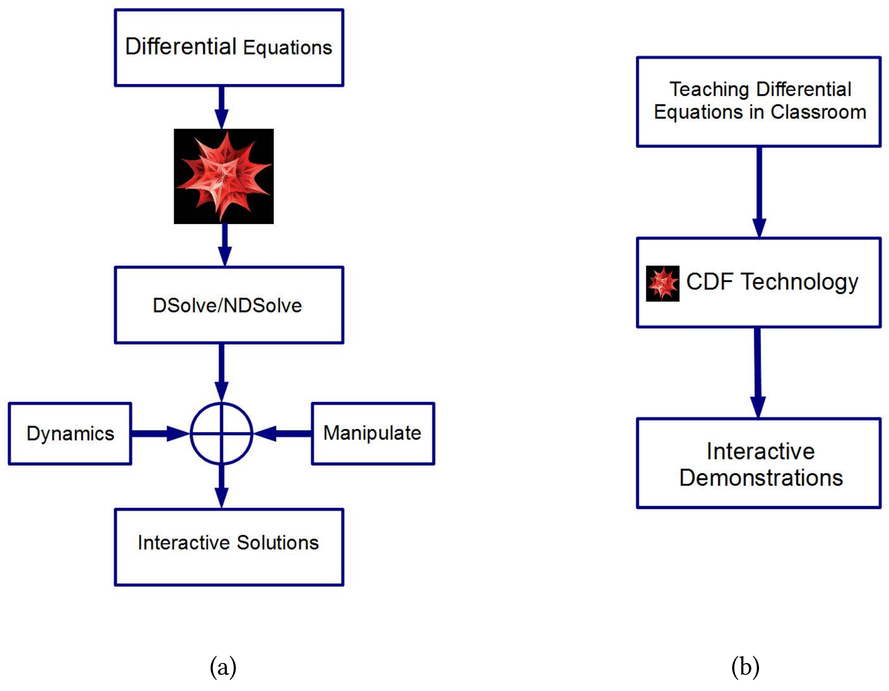


Figure 3: Workflow for understanding and studying differential equations

There are essentially two general paths to understanding and studying differential equations with *Mathematica*. Figure 3 shows these two paths. The first approach is more hands on, using DSolve and NDSolve to solve a differential equation and then using interactive elements of *Mathematica* to obtain a dynamic representation of the solution. This approach is intensive. The second approach is more suited if the objective is to teach, understand, and learn about ODEs in a classroom setting and not spend time coding the example. Then you can use pre-built interactive Demonstrations from the Wolfram Demonstrations site. These do not require a copy of *Mathematica*; you only need to have the Wolfram CDF plugin which can be downloaded from (http://www.wolfram.com/cdf). In what follows, we will describe each approach in more detail.

2 Symbolic and Numerical Solutions to Differential Equations

As noted in introduction, the *Mathematica* function DSolve finds symbolic solutions to differential equations. The *Mathematica* function NDSolve, on the other hand, is a general numerical differential equation solver. DSolve and NDSolve are equipped with a wide variety of techniques for solving single ODEs as well as systems of ODEs. To compare and contrast the syntax of these two solvers, consider the differential equation

$$\begin{aligned} y'(t) &= y(t), \\ y(0) &= 1, \end{aligned} \tag{2.1}$$

whose solution is $y(t) = e^t$. We use DSolve to analytically solve this equation as follows:

```
In[1]:= sol1 = DSolve[{y'[t] == y[t], y[0] == 1}, y, t][[1]]
Out[1]= {y -> Function[{t}, e^t]}.
```

It is important for the reader to note that if you type the above equation verbatim in *Mathematica*, one should not include the $In[1] :=$ statement. This is automatically generated by *Mathematica*. The solution is returned as a *Mathematica replacement rule*. This rule can then be used with expressions involving the solution y as follows:

```
In[2]:= y[t] /. sol1
Out[2]= e^t

In[3]:= y[1] /. sol1
Out[3]= e
```

The replacement rule can be applied in conjunction with other *Mathematica* operations:

```
In[5]:= Sin[y[t]] /. sol1
Out[5]= Sin[e^t]

In[6]:= D[Sin[y[t]], t] /. sol1
Out[6]= e^t Cos[e^t]
```

```
In[7]:= Integrate[y[t] /.sol, {t, 0, 1}]
Out[7]= -1 + e
```

While the NDSolve syntax resembles that of DSolve, it returns a numerical "InterpolatingFunction" rather than an exact symbolic solution:

```
In[8]:= sol2 = NDSolve[{y'[t] == y[t], y[0] == 1}, y, {t, 0, 1}][[1]]
Out[8]= {y -> InterpolatingFunction[{{0.,1.}},<>]}.
```

The interpolating function is an internal object within *Mathematica* that contains the numerical solution data. The function can be used as a "black-box" function which can be used for further mathematical operations like taking derivatives, integrating, etc. in a unified manner.

This time, the replacement rule can be used to evaluate y at different numerical points. Note that the value for $y(1)$ determined by NDSolve is a decimal approximation of exp, the exact value for $y(1)$ which was returned by DSolve:

```
In[11]:= y[1] /. sol2
Out[11]=2.718281861392969
```

Having given a brief introduction to the *Mathematica* syntax, we now look at a number of different problems that can be solved using DSolve/NDSolve.

2.1 Forced Vibrations

Let us begin by considering an example of forced vibrations [6] with initial conditions as given below:

$$\begin{aligned} y''(t) + y(t) &= e^{-t/5}\cos(10t), \\ y(0) &= 0 \\ y'(0) &= 0. \end{aligned} \tag{2.2}$$

For this problem, DSolve returns a solution in terms of exponential and trigonometric functions. The solution can be expressed in a more compact form using the Simplify function, as shown in Figure 4.

2.2 Logistic Equation

Next, we consider the logistic equation that occurs in population dynamics:

$$\begin{aligned} y'(x) &= y(x)(1 - by(x)), \\ y(0) &= a \end{aligned} \tag{2.3}$$

We can solve this equation and plot the solution for different values of the parameters as follows. The *Mathematica* program and result are given in Figure 5.

```
ForcedVibrations = {y''[t] + y[t] == Cos[10 * t] / E^(t / 5),
  y[0] == 0, y'[0] == 0};

sol = Simplify[DSolve[ForcedVibrations, y[t], t]]
```

$$\left\{\left\{y[t] \to \frac{5\, e^{-t/5}\left(6185\, e^{t/5}\,\mathrm{Cos}[t] - 6185\,\mathrm{Cos}[10\,t] + 1263\, e^{t/5}\,\mathrm{Sin}[t] - 250\,\mathrm{Sin}[10\,t]\right)}{3\,065\,338}\right\}\right\}$$

```
Plot[y[t] /. sol, {t, 0, 25}, Frame -> True, FrameLabel -> {t, y[t]},
 FrameStyle -> 14, Axes -> None]
```

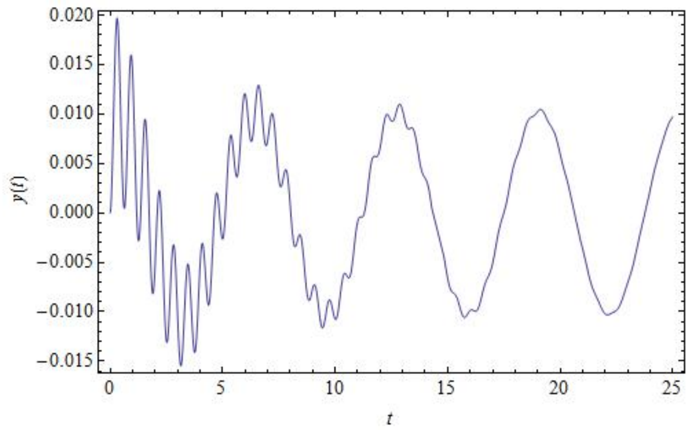


Figure 4: Solution to the forced vibration problem.

```
LogisticEquation = {y'[x] == y[x] * (1 - b * y[x]),
  y[0] == a};

sol = Quiet[DSolve[LogisticEquation, y, x]]
```

$$\left\{\left\{y \to \mathrm{Function}\left[\{x\},\ \frac{a\, e^{x}}{1 - a\,b + a\,b\, e^{x}}\right]\right\}\right\}$$

```
Plot[Evaluate[y[x] /. sol /. {b -> 1 / 27} /. {{a -> 1 / 13}, {a -> 1 / 2}, {a -> 1 / 6},
  {a -> 1 / 28}}], {x, 0, 18}, Frame -> True, FrameLabel -> {t, y[t]},
 FrameStyle -> 14, Axes -> None]
```

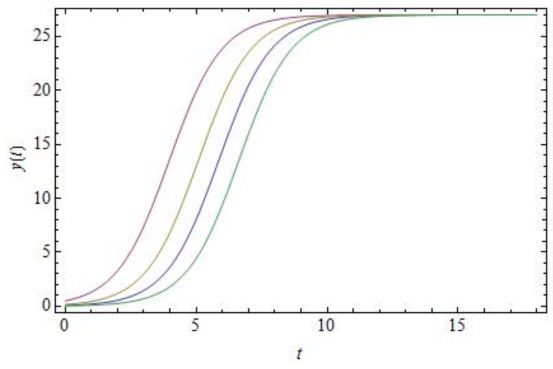


Figure 5: Solution of the logistic equation

2.3 Cornu Spiral Problem

As a final example for DSolve, we consider the following equations, which define a Cornu spiral:

$$\begin{array}{lll} x'(s) = \cos(t(s)) & y'(s) = \sin(t(s)) & t'(s) = s \\ x(0) = 0 & y(0) = 0 & t(0) = 0. \end{array} \tag{2.4}$$

This is a plane curve through the origin whose curvature is equal to the parameter value s at every point. The solution of the ODEs contains Fresnel functions. The Fresnel functions [1, 2] are defined as

$$\begin{aligned} \textbf{FresnelC}(z) &= \int_0^z \cos\frac{\pi t^2}{2} dt, \\ \textbf{FresnelS}(z) &= \int_0^z \sin\frac{\pi t^2}{2} dt \end{aligned} \tag{2.5}$$

and are widely used in the field of optics [4]. A graph of the curve in Figure 6 clearly shows that the curvature becomes large as s approaches infinity in the positive or negative direction.

```
CornuSpiral = {x'[s] == Cos[t[s]], y'[s] == Sin[t[s]],
   t'[s] == s, x[0] == 0, y[0] == 0, t[0] == 0};

sol = DSolve[CornuSpiral, {x, y, t}, s]
```

$$\left\{\left\{t \to \text{Function}\left[\{s\}, \frac{s^2}{2}\right], x \to \text{Function}\left[\{s\}, \sqrt{\pi}\,\text{FresnelC}\left[\frac{s}{\sqrt{\pi}}\right]\right], y \to \text{Function}\left[\{s\}, \sqrt{\pi}\,\text{FresnelS}\left[\frac{s}{\sqrt{\pi}}\right]\right]\right\}\right\}$$

```
ParametricPlot[Evaluate[{x[s], y[s]} /. sol[[1]]], {s, -20, 20},
 Frame -> True, FrameLabel -> {t, y[t]}, FrameStyle -> 14,
 ImageSize -> 300]
```

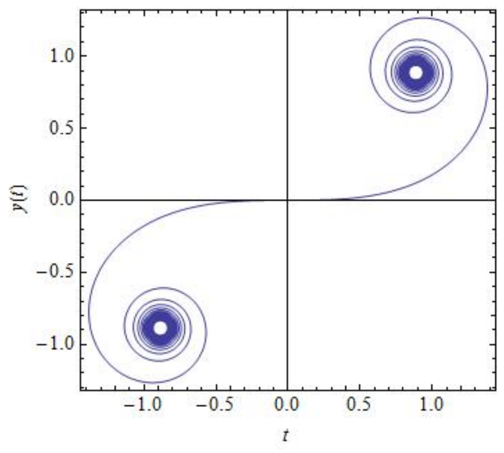


Figure 6: Solution of the Cornu spiral problem

A brief overview of the functionality for solving ODEs using DSolve is given in [13]. The tutorial [14] gives further details for finding symbolic solutions of differential equations and includes a user guide with helpful tips for working with DSolve.

We will now present two examples of ODEs for which symbolic solutions are not available. These problems can be handled by NDSolve, which also has a wide variety of options for precision and step control, choice of solution method, etc.

2.4 Lorenz Attractor

First, we consider the Lorenz attractor from chaotic dynamics [23], which can be obtained by solving the associated system of nonlinear ODEs using NDSolve. The equations are

$$\begin{aligned} x'(t) &= -3(x(t) - y(t)) \\ y'(t) &= -x(t)z(t) + 26.5x(t) - y(t) \\ z'(t) &= x(t)y(t) - z(t) \\ x(0) &= z(0) = 0 \\ y(0) &= 1. \end{aligned} \tag{2.6}$$

Figure 7 shows the program and a phase-space portrait of the solution. One can clearly see the classical butterfly pattern describing the chaotic nature of the solution.

```
LorenzAttractor = {x'[t] == -3*(x[t] - y[t]),
  y'[t] == x[t]*(-z[t]) + 26.5*x[t] - y[t],
  z'[t] == x[t]*y[t] - z[t], x[0] == z[0] == 0,
  y[0] == 1};

sol = NDSolve[LorenzAttractor, {x, y, z}, {t, 0, 100}];

ParametricPlot3D[Evaluate[{x[t], y[t], z[t]} /. sol], {t, 0, 100},
 ColorFunction -> (Hue[#4] &), BoxRatios -> 1,
 AxesLabel -> {x[t], y[t], z[t]}, AxesStyle -> 14, ImageSize -> 300]
```

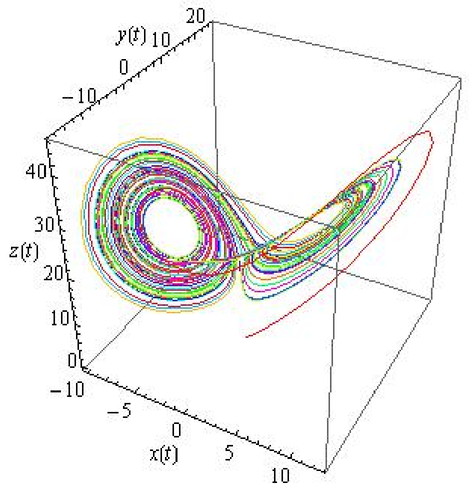


Figure 7: Phase-space portrait of the 3D Lorenz model

2.5 More Complicated Spiral

As a second example, we consider the differential equations for a spiral with curvature function $t'(s) = s\sin(s)$. the curvature function measures the rate of turning for the tangent vector along the spiral curve.

$$\begin{aligned} x'(s) &= \cos(t(s)) \\ y'(s) &= \sin(t(s)) \\ t'(s) &= s\sin(t(s)) \\ x(0) &= y(0) = t(0) = 0 \end{aligned} \tag{2.7}$$

In this case, as opposed to the Cornu spiral, there is no closed form available, but the problem can be solved efficiently using NDSolve and is visualized in Figure 8.

```
AnotherSpiral = {x'[s] == Cos[t[s]],
  y'[s] == Sin[t[s]], t'[s] == s*Sin[s],
   x[0] == 0, y[0] == 0, t[0] == 0};

sol = NDSolve[AnotherSpiral, {x, y, t}, {s, -20, 20}];

ParametricPlot[Evaluate[{x[s], y[s]} /. sol[[1]]], {s, -20, 20},
 Frame -> True, FrameLabel -> {x[s], y[s]}, FrameStyle -> 14,
 ImageSize -> 200]
```

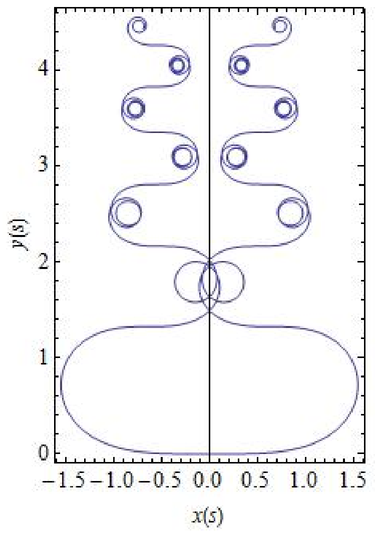


Figure 8: Parametric plot for a spiral with curvature function $s\sin(s)$

The tutorial [21] gives further details on numerical differential equation solving using NDSolve, including the method plugin framework for users who wish to extend the built-in algorithms.

All of the examples given in this section involve numerical parameters (e.g. the logistic equation) or functional parameters (e.g. the spirals of curvature). Clearly, it would be very useful to have a function that allows us to visualize the solutions of ODEs obtained by varying the parameters that occur in them. This can be accomplished by using the Manipulate function [3], which is a part of the system for incorporating dynamic objects in *Mathematica*. The interested reader can also visit the website [18] for some interesting examples of using the Manipulate function.

While differential equations can be solved easily and efficiently with DSolve and NDSolve, the use of dynamic interactivity to explore standard themes in undergraduate-level differential equations and dynamical systems courses is a major focus of the Wolfram Demonstrations Project website [5]. The Wolfram Demonstrations Project includes a broad collection of examples involving linear and nonlinear ODEs that have been contributed by an international community of educators and experts in technical fields. The next section discusses these Demonstrations in detail, in order to make it easy for ODE educators who wish to use the examples in a classroom setting.

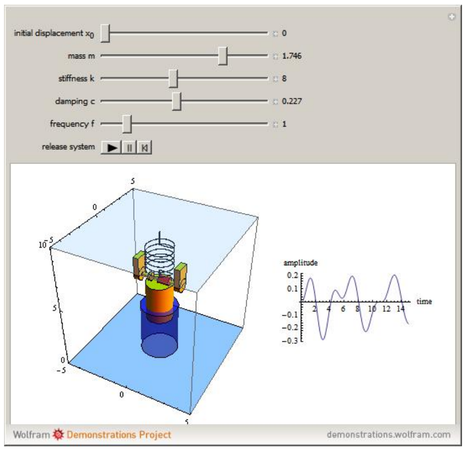
3 ODE Demonstrations

The Wolfram Demonstrations site contains a vast collection of Demonstrations related to differential equations. New Demonstrations are being added all the time. The site can be navigated either by topic or through a convenient search feature. The technical sophistication of the Demonstrations varies from introductory concepts to research topics. A search for the term "oscillation," for example, currently yields 224 Demonstrations. These range from visualizations of simple spring mass systems to more advanced nonlinear systems and even PDEs from mathematical physics. Some snapshots of the popular Demonstrations can be seen in Figure 9. All Demonstrations are highly interactive, so that even those dealing with advanced mathematical concepts could be amusing for beginning students. Most Demonstrations provide sliders and buttons that allow the user to visualize the solutions or other properties of the system while varying parameters.

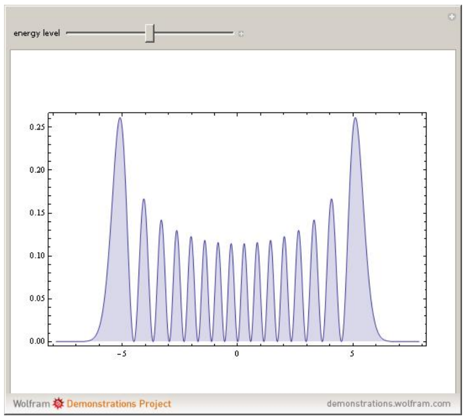
Probably the easiest way for educators to incorporate the Demonstrations site into their curriculum is to recommend that students explore it independently. To an educator who has struggled to encourage students to keep up with the "required" textbook reading, this may seem a bit like wishful thinking. However, the Wolfram Demonstration site is not a textbook and does not replace a textbook. The Demonstrations on the site are interesting and visually fascinating in their own right. As we will show in the examples below, the site facilitates an interactive approach to mathematics. Also, a student visiting the site for a differential equations class may be delighted to find fascinating Demonstrations of optical illusions [12], trebuchets [16], and tsunamis [8], as seen in Figure 10, in addition to differential equation Demonstrations.

Encouraging students to visit the Demonstrations site could be accomplished by augmenting homework sets with links to relevant Demonstrations. The following is a typical exercise for a first year calculus student [22]:

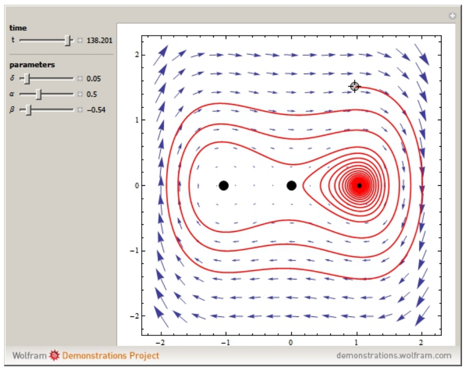
> The population of the world was about 5 billion in 1986. Using the exponential model for population growth (...) with the recently observed rate of increase of population of 2% per year in 1986, find an expression for the population of the world in the year t.


initial displacement x_0 0
mass m 1.746
stiffness k 8
damping c 0.227
frequency f 1
release system

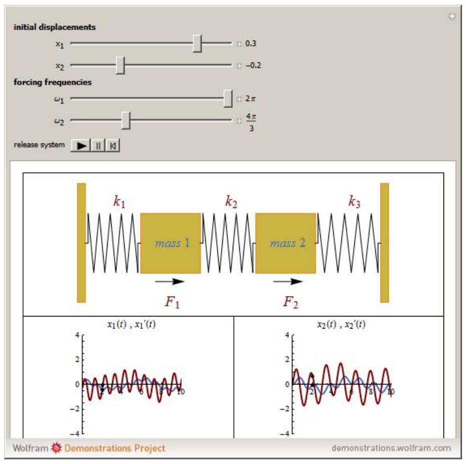
(a)


energy level

(b)

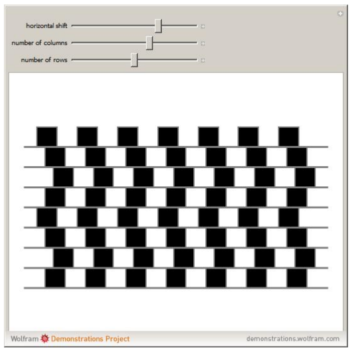

time
t 138.201
parameters
δ 0.05
α 0.5
β −0.54

(c)

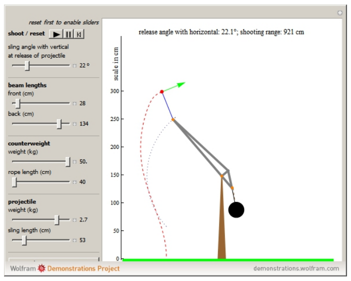

initial displacements
x_1 0.3
x_2 −0.2
forcing frequencies
ω_1 2π
ω_2 $\frac{4\pi}{3}$
release system

(d)

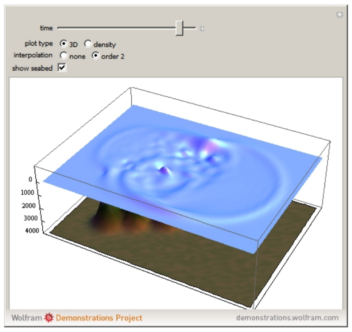
Figure 9: Some Demonstrations returned by a search for the term "oscillation": (a)forced oscillations [30], (b) harmonic oscillator [24], (c) Duffing oscillator [9], (d) oscillator with two masses [28]


horizontal shift
number of columns
number of rows

(a)


release angle with horizontal: 22.1°; shooting range: 921 cm
reset first to enable sliders
shoot / reset
sling angle with vertical at release of projectile 22°
beam lengths
front (cm) 28
back (cm) 134
counterweight
weight (kg) 50.
rope length (cm) 40
projectile
weight (kg) 2.7
sling length (cm) 53

(b)


time
plot type [✓ 3D] density
interpolation none [✓ order 2]
show seabed ✓

(c)

Figure 10: Some amusing Demonstrations: (a) optical illusions [12], (b) trebuchets [16], (c) tsunamis [8]

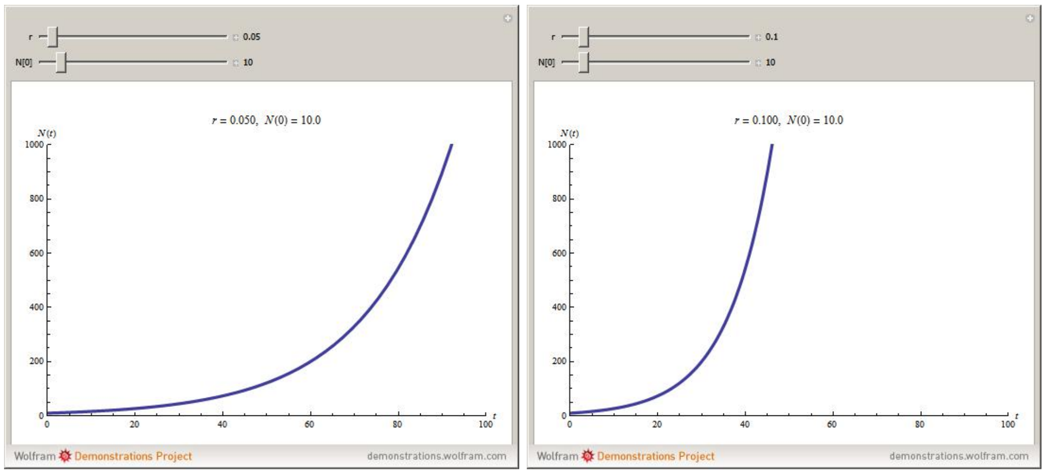

r
0.05
N[0]
10
$r = 0.050,\ N(0) = 10.0$
$N(t)$
1000
800
600
400
200
0
20
40
60
80
100
t
Wolfram Demonstrations Project
demonstrations.wolfram.com
r
0.1
N[0]
10
$r = 0.100,\ N(0) = 10.0$
$N(t)$
1000
800
600
400
200
0
20
40
60
80
100
t
Wolfram Demonstrations Project
demonstrations.wolfram.com

Figure 11: Demonstration of exponential growth [19]. Moving the sliders for r or N shows how the curve reacts.

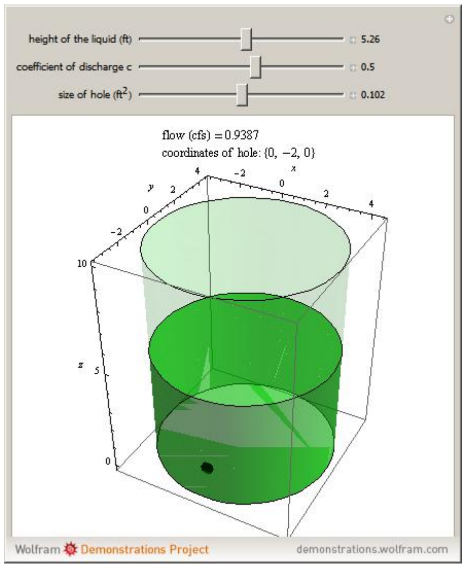
A search for the term "population growth" presently returns 36 results. The first result ("Continuous Exponential Growth") is of direct relevance to this problem (see Figure 11). It is important to note that while the Demonstration does not solve the exercise for the student, it does provide a means by which the student can check his/her intuition. Other results ("State Population Growth" [7], "Predator-Prey Equations" [25], and "Lily Pond" [20]), while not directly related to the exercise, are tangentially related. It has been our experience that typical textbook applications of calculus and differential equations problems have very good coverage on the Demonstrations site. This enables students to augment their homework sets with interactive visualizations of the problems they are solving or have solved.

Another typical class of problems for introductory differential equations deals with the flow of water between (or out of) tanks. Searches for "tank," "Torricelli's law," "two tank problem," etc. all yield a rich collection of visualizations related to this class of problem. A number of these Demonstrations even directly reference problems in standard differential equations textbooks. A small sample of these Demonstrations appears in Figure 12. Thus, it is often possible for students with no programming background to quickly and easily find engaging visualizations related to assigned exercises. This could go a long way toward addressing the common concern among mathematics educators that students do not take the necessary time to "digest" a problem once they have found the solution.

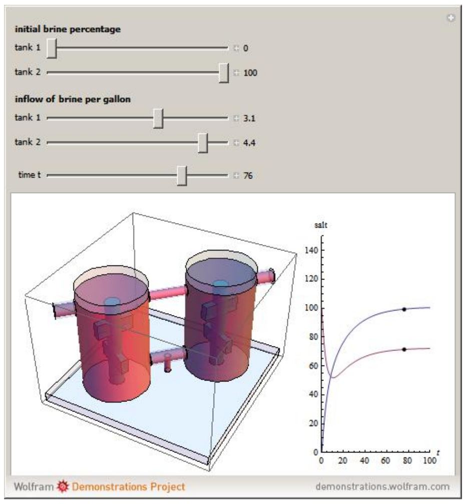
The Demonstrations Project is also an effective tool for in-class activities. Even educators without computing experience can use the site to incorporate graphical Demonstrations into their lessons. Suppose, for example, an instructor has derived the equations of motion of a pendulum:

$$\theta''(t) + \frac{g}{l}\sin(\theta(t)) = 0$$

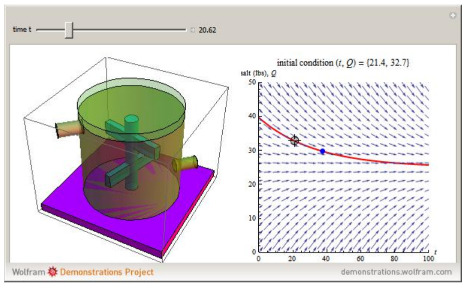
Typically, the instructor would next explain to the students how the values of the parameters of the system affect the dynamics. A quick search for "pendulum" on the Demonstrations site yields 76 Demonstrations at this time. Several of these are immediately relevant to this lesson. For example, "Animated Pendulum" [10] shows the relationship


height of the liquid (ft) 5.26
coefficient of discharge c 0.5
size of hole (ft^2) 0.102
flow (cfs) = 0.9387
coordinates of hole: {0, −2, 0}

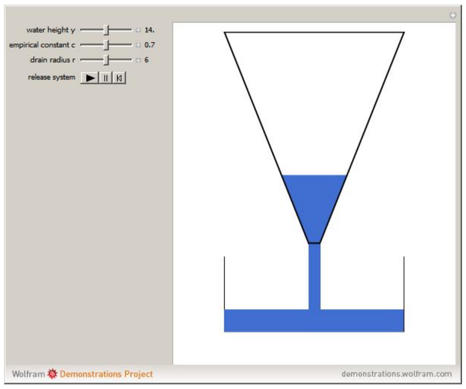
(a)


initial brine percentage
tank 1 0
tank 2 100
inflow of brine per gallon
tank 1 3.1
tank 2 4.4
time t 76

(b)


time t 20.62
initial condition (t, Q) = {21.4, 32.7}

(c)


water height y 14.
empirical constant c 0.7
drain radius r 6
release system

(d)

Figure 12: Some Demonstrations related to flow in a tank: (a) flow of fluid through a hole [11], (b) mixing in two tanks [27], (c) mixing salt in water [26], (d) Torricelli's law [29]

between length, maximum angle, mass, and gravity in an animated pendulum/bar chart (see Figure 13).

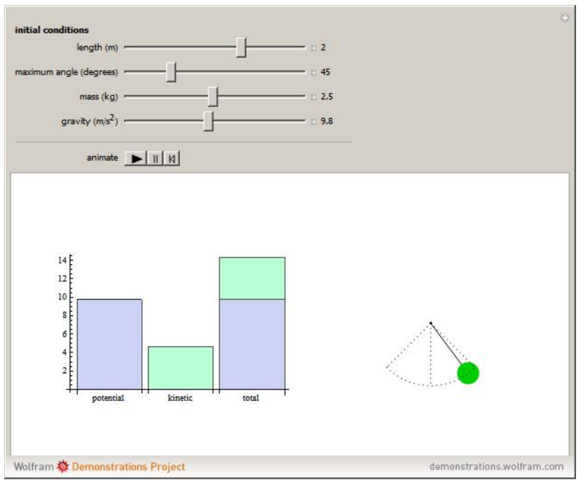


Figure 13: Animated pendulum with graphs [10]

This could be followed by a series of related Demonstrations. For example, "Pendulum with Varying Length or How to Improve Your Next Swing Ride" [17] is a Demonstration that simulates a person sitting on a swing bouncing vertically to try to swing higher (see Figure 14). The problem parameters (amplitude and frequency of the bouncing) can be varied to determine the optimal experience (i.e. swing as high as possible).

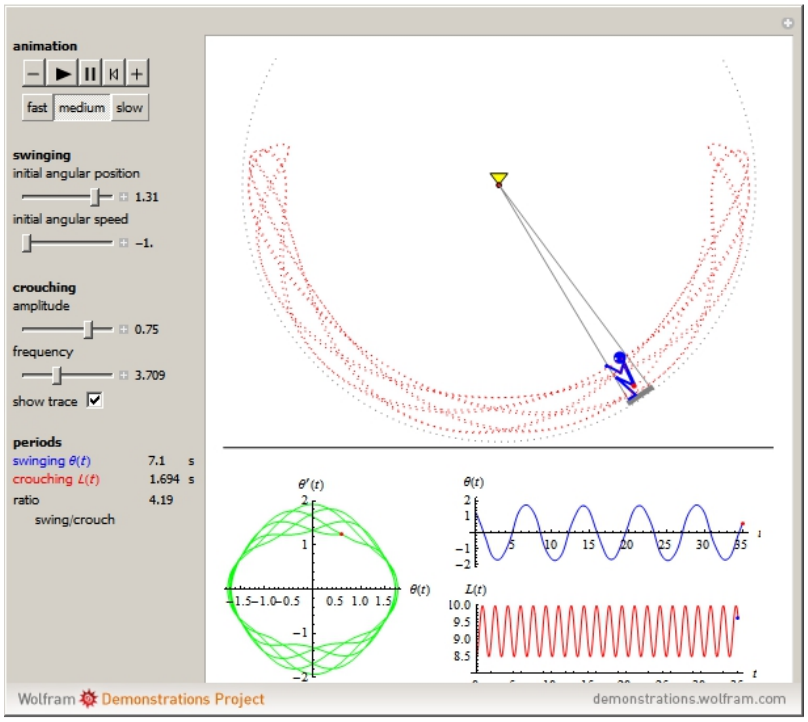
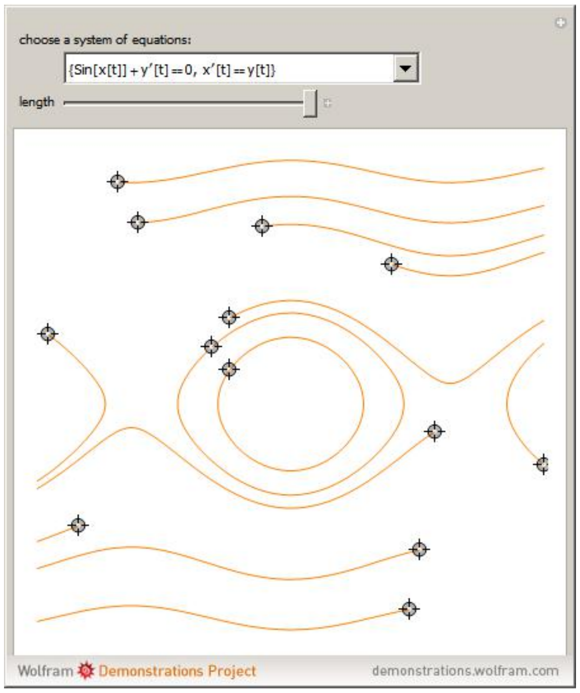


Figure 14: Pendulum with varying length [17]

For educators who wish to delve a little deeper into the world of Wolfram Demonstrations and *Mathematica* as a whole, it is also possible to customize existing Demonstrations or to write and contribute completely new ones. All Demonstrations on the site include the *Mathematica* program that generated them. These programs are typically relatively short and have been checked for quality and content before being put on the site. Consider the "EquationTrekker" Demonstration [15], which visualizes the solutions of a differential equation, beginning at various user-determined starting points (see Figure 15).

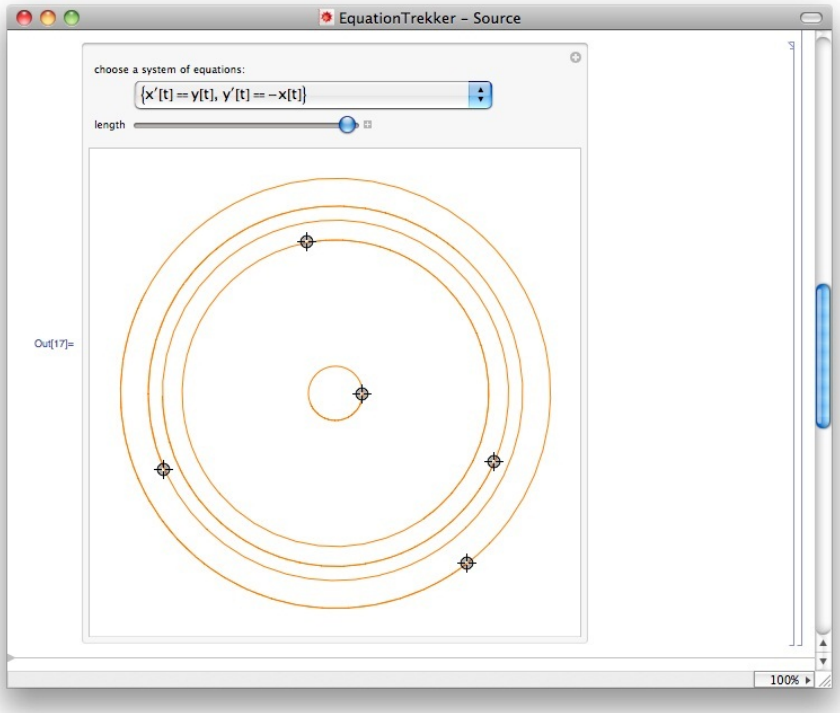

choose a system of equations: {Sin[x[t]] + y′[t] == 0, x′[t] == y[t]}

length

Figure 15: The EquationTrekker Demonstration [15]

This Demonstration only works for ten predefined differential equations. Suppose an educator wishes to visualize the solutions to an ODE not on the list, for example, $x'(t) = y(t), y'(t) = -x(t)$. This is easily accomplished by downloading the source code. In this notebook the equation list appears as:

```
  equationList = {
{y’[t] == -Sin[x[t]], x’[t] == y[t]},
{y’[t] == -Cos[x[t]], x’[t] == y[t]},
{y’[t] == -Tan[x[t]], x’[t] == y[t]},
{y’[t] == y[t] - x[t], x’[t] == y[t]},
{y’[t] == Sin[y[t]] - x[t], x’[t] == y[t]},
{y’[t] == Cos[y[t]] - x[t], x’[t] == y[t]},
{y’[t] == 4 x[t] + y[t], x’[t] == x[t] + y[t]},
{y’[t] == Sqrt[2] x[t] - 2 y[t], x’[t] == -3 x[t] + Sqrt[2] y[t]},
{y’[t] == -2 x[t] - y[t], x’[t] == -x[t] + 2 y[t]},
{y’[t] == x[t] + 3 y[t], x’[t] == x[t] - y[t]}};
```



choose a system of equations:
{x′[t] == y[t], y′[t] == −x[t]}
length

Figure 16: Solution in EquationTrekker using modified equations

It is then simply a matter of adding the new equation:

```
{y'[t]==x[t], x'[t]==-y[t]}
```

to the list and running the notebook (see Figure 16).

One can often build on the ideas obtained from the demonstrations to develop and write custom code for use in the classroom. As an example, we can improve on the idea of "EquationTrekker" and develop a function that plots the vector fields associated with the differential equations while interactively moving the initial point/condition. The *Mathematica* code is given below:

```
EquationVisualizer[eqns_, vars_, {t_, tmin_, tmax_}] :=
  Module[{splot, vt, vo, neweqns},
      {vt, vo} = Map[Through[vars[#]] &, {t, tmin}];
      neweqns = First[Solve[eqns, D[vt, t]]] /. Rule :> Equal ;
      splot = StreamPlot[ neweqns[[All, 2]] /. Thread[vt -> vars],
          {x, -4, 4}, {y, -3, 3}, StreamScale -> Large,
          StreamColorFunction -> "Rainbow"];
      Manipulate[
          Show[splot,
              ParametricPlot[
                  Evaluate[First[vt /. NDSolve[{eqns,
                  Thread[vo == point]},vars, {t, tmin, tmax}]]],
                  {t, tmin, T},PlotStyle -> {Red, Thick}
                  ]
              ],
          {{T, tmax/10}, 1,tmax}, {{point, {1, 0}}, Locator},
          SaveDefinitions -> True]
]
```

Figure 17 shows the use of the above custom function and the resulting output.

```
EquationVisualizer[{x'[t] == y[t], y'[t] == -x[t] + (1 - x[t]^2) y[t]},
 {x, y}, {t, 0, 100}]
```

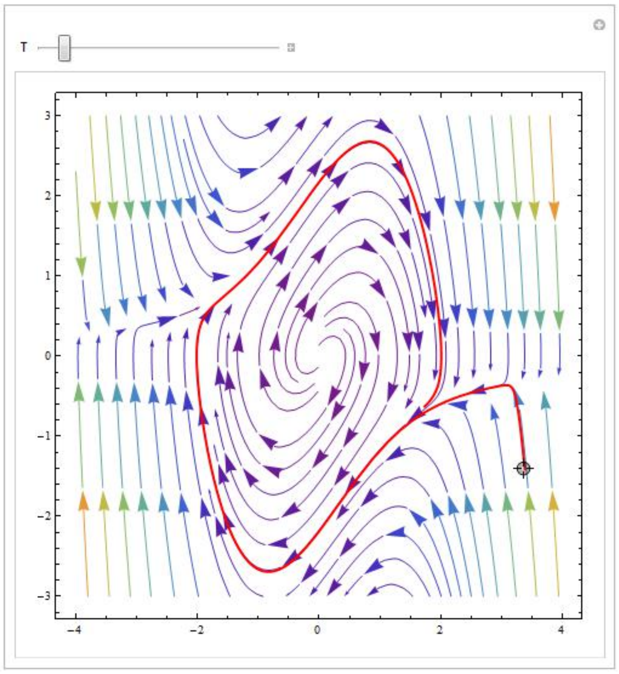


Figure 17: Use and output for the function "EquationVisualizer". The interactive point (indicated by circle with cross hairs) represents the initial condition. Solution is updated when point is moved.

4 Conclusion

In this paper we have explored various ways that researchers and educators can use *Mathematica* and the Wolfram Demonstrations Project to study and solve differential equations. Particular attention was focused on the functions DSolve and NDSolve for solving differential equations. To summarize, DSolve and NDSolve are versatile functions with a vast array of powerful methods for solving a broad range of linear and nonlinear differential equations. Once a solution has been obtained, it can be plotted or used in later computations. To this end, the examples that were presented in Section 2 give a general overview of how these functions are to be used and how one can analyze the results. It should be noted that the user often does not have to make any drastic change to syntax when moving from one problem to the other. The unified framework of the function call makes using these functions very easy.

It is also very convenient to have ready-made examples such as those provided by the Demonstrations Project, which typically suppress computational details in favor of appealing graphics, dynamic interactivity, and visualization of the actual physical models (e.g. the vibrations of a spring) and have the potential to improve the quality of learning in a typical undergraduate differential equations course.

The focus of Section 3 was to elaborate on the various pre-built Demonstrations that one can use from the Wolfram Demonstrations Project site. The interactive nature of these Demonstrations makes them a very valuable tool when trying to understand different properties of differential equations. We focused on practical examples that one encounters often. A detailed explanation of how you can take these Demonstrations and modify them to a specific setting was discussed. This is very important if you want to avoid building a Demonstration from scratch.

A combination of both the above paradigms appears to be the most fruitful approach for using *Mathematica* in an educational setting.

References

[1] FresnelC function page. *Wolfram Research, Inc.*, . URL `http://reference.wolfram.com/mathematica/ref/FresnelC.html`.

[2] FresnelS function page. *Wolfram Research, Inc.*, . URL `http://reference.wolfram.com/mathematica/ref/FresnelS.html`.

[3] Introduction to Manipulate. *Wolfram Research, Inc.* URL, `http://reference.wolfram.com/mathematica/tutorial/IntroductionToManipulate.html`.

[4] Fresnel integrals. URL `http://en.wikipedia.org/wiki/Fresnel_integral`.

[5] Wolfram Demonstrations Project. URL `http://demonstrations.wolfram.com`.

[6] W.E. Boyce and R.C. DiPrima. *Elementary Differential Equations and Boundary Value Problems.* New York: John Wiley and Sons Inc., 1997.

[7] S.J. Chandler. State population growth. *The Wolfram Demonstrations Project.* URL `http://demonstrations.wolfram.com/StatePopulationGrowth/`.

[8] Y.S. Chang, R. Knapp, and R. Germundsson. Mathematics of tsunamis. *The Wolfram Demonstrations Project.* URL `http://demonstrations.wolfram.com/MathematicsOfTsunamis/`.

[9] G. Craven. Phase plane trajectories of the unforced Duffing oscillator. *The Wolfram Demonstrations Project.* URL `http://demonstrations.wolfram.com/PhasePlaneTrajectoriesOfTheUnforcedDuffingOscillator/`.

[10] J. Frederico and R. Morris. Animated pendulum. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/AnimatedPendulum/.

[11] J. Gordji and S. Gordji. Flow of fluid through a hole in a tank. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/FlowOfFluidThroughAHoleInATank/.

[12] T. Gray. Parallel lines optical illusion. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/ParallelLinesOpticalIllusion/.

[13] D.A. Kapadia. Exploring differential equations with Mathematica. *Proceedings of the ICTCM, Boston*, 2007. URL http://archives.math.utk.edu/ICTCM/i/19/S024.html.

[14] D.A. Kapadia. Differential equation solving with DSolve. *Wolfram Research Inc.* URL http://reference.wolfram.com/mathematica/tutorial/DSolveOverview.html.

[15] R. Knapp and T. Gray. EquationTrekker. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/EquationTrekker/.

[16] E. Mahieu. Optimizing the counterweight trebuchet. *The Wolfram Demonstrations Project*, . URL http://demonstrations.wolfram.com/OptimizingTheCounterweightTrebuchet/.

[17] E. Mahieu. Pendulum with varying length or how to improve your next swing ride. *The Wolfram Demonstrations Project*, . URL http://demonstrations.wolfram.com/PendulumWithVaryingLengthOrHowToImproveYourNextSwingRide.

[18] M. McGuff. *Mathematica* material for differential equations. *Austin Community College*, 2009. URL http://www.austincc.edu/mmcguff/mathematica/diffeq/.

[19] G. Russell. Continuous exponential growth. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/ContinuousExponentialGrowth/.

[20] M. Schreiber. Lily pond. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/LilyPond/.

[21] M. Sofroniou and R. Knapp. Advanced numerical differential equation solving in *Mathematica*. *Wolfram Research Inc.* URL http://reference.wolfram.com/mathematica/tutorial/NDSolveOverview.html.

[22] James Stewart. *Calculus.* Pacific Grove, CA: Brooks-Cole, 3rd edition, 1995.

[23] S.H. Strogatz. *Nonlinear Dynamics and Chaos.* Cambridge, MA: Westview, 1994.

[24] M. Trott. Harmonic oscillator eigenfunctions. *The Wolfram Demonstrations Project.* URL http://demonstrations.wolfram.com/HarmonicOscillatorEigenfunctions/.

[25] E.W. Weisstein. Predator-prey equations. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/PredatorPreyEquations/`.

[26] S. Wilkerson. Mixing salt in water in one tank. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/MixingSaltInWaterInOneTank/`.

[27] S. Wilkerson. Mixing in two connected tanks. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/MixingInTwoConnectedTanks/`.

[28] S. Wilkerson. Two masses with forcing functions oscillating between three springs. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/TwoMassesWithForcingFunctionsOscillatingBetweenThreeSprings/`.

[29] S. Wilkerson and M. Evans. Torricelli's law for tank draining. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/TorricellisLawForTankDraining/`.

[30] S. Wilkerson and M. Evans. Forced oscillations. *The Wolfram Demonstrations Project*. URL `http://demonstrations.wolfram.com/ForcedOscillations/`.