CODEE Journal

Volume 11 Article 2

11-13-2018

Teaching Differential Equations without Computer Graphics Solutions is a Crime

Beverly H. West
Cornell University, Retired, bhw2@cornell.edu

Teaching Differential Equations without Computer Graphics Solutions is a Crime.

Beverly West
Cornell University

Keywords: Differential Equations, Computer Graphics, Teaching
Manuscript received on June 22, 2017; published on November 13, 2018.

Abstract: In the early 1980s computer graphics revolutionized the teaching of ordinary differential equations (ODEs). Yet the movement to teach and learn the qualitative methods that interactive graphics affords seems to have lost momentum. There still exist college courses, even at big universities, being taught *without* the immense power that computer graphics has brought to differential equations. The vast majority of ODEs that arise in mathematical models are nonlinear, and linearization only approximates solutions sufficiently near an equilibrium. Introductory courses need to include nonlinear DEs. Graphs of phase plane trajectories and time series solutions allow one to see and analyze the crucial behaviors, whether or not analytic solutions exist. Furthermore, interactivity is key to experimenting with parameters in order to modify behaviors. Now, a quarter of a century later, we have far more technology—but many features of the original software have been lost in the rush to the future. We have both educational and software concerns. This is not only an academic issue—scientists at multiple nonacademic agencies (e.g., FDA, NIH, USCGS) immediately took up our software tools in the late 1980s, and increasingly more of our students come from fields that did not traditionally require mathematics background. We should not be depriving today's students of the skills to analyze behaviors of solutions to ODEs.

1 Introduction

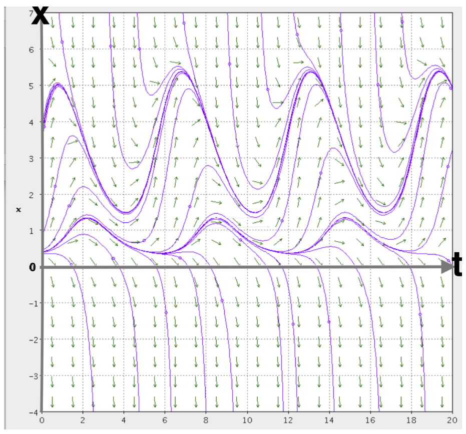
Modeling is essential in today's world—problem solving depends on exploration with changing parameters. But how can you do that without interactive graphics that allows and encourages experimentation?

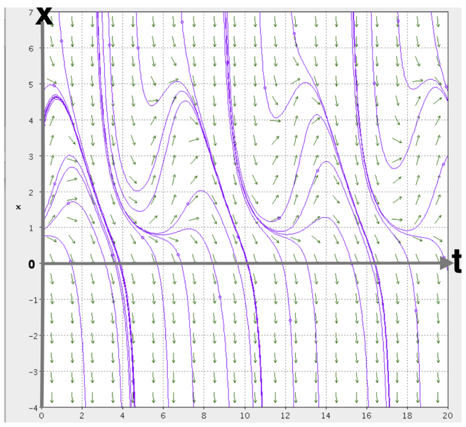
In the early 1990s (a quarter of a century ago!) the revolution of readily accessible interactive computer graphics opened up the ability to do just that sort of problem solving with orderinary differential equations (ODEs). Suddenly we could plot direction fields, and by clicking on any point as initial condition, see the trajectories evolve. In seconds you could fill the plane with trajectories, see the singularities/equilibrium points, and analyze behaviors. Then you could just as quickly see the effects of changing parameters (coefficients and constants).

A few quick examples (two from talk slides, one from a colleague) show how behaviors change as parameters change, and how visualization makes a huge leap in understanding a solution.

Example 1. A refined population model. Consider a population x with a variable fertility rate, subject to crowding and a constant rate of decrease (for example, by hunting licenses). Changing only the parameter C gives vastly different behaviors .

$$x' = \underbrace{(2+\cos t)\,x}_{\boxed{\text{Natural, positive rate of growth, with seasonal fluctuation}}} - \underbrace{0.5x^2}_{\boxed{\text{Population decrease due to crowding}}} - \underbrace{C}_{\boxed{\text{Constant rate of decrease due to hunting}}}$$

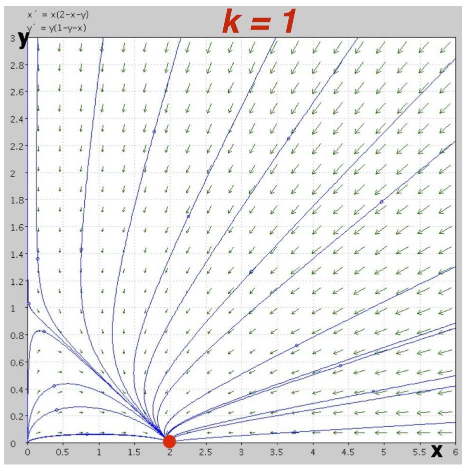


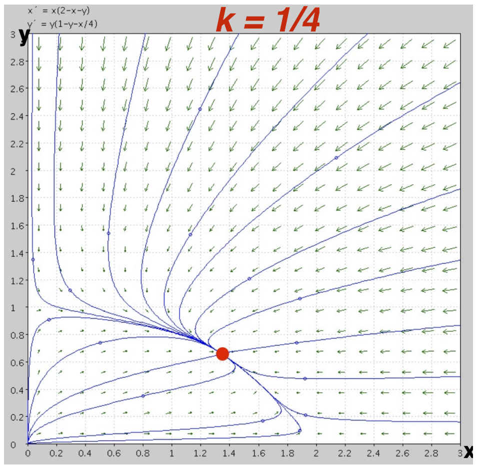


$C = 1$: Many sustainable solutions $\qquad$ $C = 2$: NO sustainable solution

Example 2. Predator/Prey. Standard population equations for two interacting species are based on $P' = P$ (birth rate − crowding term ± interaction term). For x predators and y prey we show 2 cases where only the predation rate (shown in red) has changed. Are you an ecologist or a pest control specialist? Which outcome do you want?

$$x' = x' = x(2 - x - y), \qquad y' = y(1 - y - kx)$$





Example 3. Interpreting Obscure Formula Solutions. Analytic solutions may be given in functions that are otherwise difficult, if not impossible, to interpret. With graphics, one can *see* them!

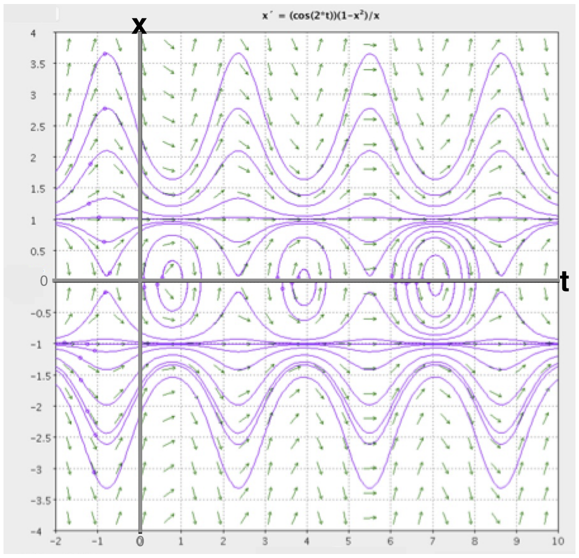
A simple example of this, due to David Tall, from a national mathematics examination paper in the UK [20]:

$$x\frac{dx}{dt}\sec 2t = 1 - x^2$$

It is easily "solved" by separating the variables to get $\frac{x}{1-x^2}dx = \cos 2t dt$, and integrated to give the "general solution"

$$-\ln\left|1-x^2\right| = \sin 2t + C,$$

but what does this *mean*? A graph of solutions on the direction field is invaluable.



Seeing the solutions shows much simpler behavior than we might expect from the formula; it gives something we can wrap our heads around, and offers the question of how it relates to the formula.

"The symbolic solution in this case is of little value without a graphical interpretation of its meaning, whilst the graphical interpretation alone lacks the precision of the symbolism." [20]

2 History

The tremendous excitement caused by this graphics revolution completely changed the ODE curriculum, and gave immense power to students, as well as to researchers within and without the mathematical and academic communities.

How so?

The Federal Drug Administration (FDA), National Institutes of Health (NIH), U.S. Coast and Geodetic Survey (USCGS), and other entities were quick to start using our first interactive graphics software *MacMath* [28], which emphasized its usefulness. My colleague John Hubbard gained some renown when he testified before a U.S. Congressional Committee re NSF funding and stated that “differential equations are *the* interface of mathematics with the real world!” While this may be an exaggeration, certainly DEs are a *major* interface, and at that Congressional Hearing, the presenters for chemistry and physics agreed that the models of which they had spoken were exactly differential equations models. Hubbard liked to tell how he could *feel* the status of mathematics rising in that room full of politicians!

Throughout the United States new textbooks and supporting software packages were created, based on the tremendous power of the graphics approach. Those by Blanchard, Devaney & Hall; Borrelli & Coleman; Farlow; Harvey Mudd; Hohn; Hubbard & West; and M.I.T. are listed in Sections 7 and 8. Other authors that quickly joined the new standard included Boyce & diPrima; Danby; Edwards & Penney; Goode; Hale & Skidmore; Kohler & Johnson; Lomen & Lovelock; Nagle, Saff, & Snider; Noonburg; Polking; Strang; Strogatz; and Zill.

In 1991 CODEE (an NSF-funded **Consortium for ODEs with Experiments**) [5] came into being to inform and help our colleagues teaching ODEs how to take advantage of the sudden amazing new capabilities—we were a consortium of seven diverse institutions: Harvey Mudd College, Cornell University, Rensselaer Polytechnic Institute, Washington State University, St. Olaf College, West Valley Community College, Stetson University.

Over the next 4 summers, consortium members at each CODEE institution gave a weeklong summer workshop at their school for 30 faculty from all over the U.S. These were wildly successful, and participants went on to give dozens of talks and workshops at department colloquia, regional meetings, and conferences. For many years we had contributed paper sessions at every JMM (Joint Mathematics Meetings) and International Conference for Technology in Collegiate Mathematics, which engendered much excitement and progress. CODEE produced a newsletter, and then in a second grant, a powerful ODE software package called *ODE Architect* [33]. Finally in 2007 under yet another NSF grant, we reorganized, using the same acronym and web address, as the **Community of ODE Educators**, creating a website with a refereed electronic *CODEE Journal* and many other features to support the teaching of ODEs. See the Article *Evolution of the Modern ODE Course* in the *CODEE Journal* [5]. We felt that everyone was getting on board.

However we have now encountered two problems:

Problem 1: Everyone is not on board. Our initial enthusiasm and success was misleading—it seems we ended up preaching to the choir! For many recent years prior to 2016 there were *no* ODE sessions in the MAA part of JMM, or at ICTCM (which is much smaller now). Other “fashions” in other fields have taken over the attention of many instructors. We have been stunned and saddened to find that there are *still* many students taking ODEs in the traditional fashion, with *no* computer component, and *no* attention to nonlinear systems (beyond teaching linearization, which is useful only if you can see how far in a given direction that linearization might give a decent approximation). Some of the reasons that so many students are losing out are:

- **institutional inertia,**
- **holding on to traditional texts** that authors have (painstakingly) created,

- **instructors who resist computers** (fear of distraction, time commitment)
- **large course loads** where mathematicians with specialties outside ODEs might be forgiven for taking the easy way out and just teaching the extra course in DEs in the cookbook manner in which *they* were taught.

I assert that it is a *crime* to deprive students of the power enabled by the computer graphics revolution. They will be at a total disadvantage compared to students who can easily attack and analyze behaviors for any nonlinear system.

There are many ways to overcome these objections.

- **single sessions, workshops, or minicourses** can bring faculty on board *quickly*;
- **computer work on homework**—today's students can handle it, with simple, straightforward software dedicated to ODEs; no extra class time required;
- **timesaving in lecturing**—e.g., a quick showing (either in lecture or homework) of a linear classification animation tool such as *Interactive Differential Equations* (IDE) [27] or *MIT Mathlets* [31][1] fosters faster and better understanding of concepts;
- **timesaving in syllabus** See [8], Preface section by Jean Marie McDill re how computer work assigned as homework *before* lecture eliminates confusion and misconceptions, thus saving valuable syllabus time;
- **timesaving in grading**—wrong graphs immediately point to errors.

We do *not* need to throw out all the traditional material, but we can make teaching and learning much *easier*, for instructors and students alike.

We must also keep in mind that students taking ODEs are majoring increasingly in "nontraditional" fields (e.g., biology, ecology, meteorology, finance, economics), not just the "usual" mathematics, physics, and engineering.

Problem 2: Software crisis. Our "old" dedicated graphic DE solvers (such as *MacMath* [26, 27]) became outdated as operating systems evolved and left them behind. It is difficult from academia to find funding that will keep software updated.

Java has been a solution for some years (for programs such as *ODE Toolkit* [26], Polking's *Dfield* and *Pplane* [34], Tutsch's *OdeFactory* [35], or Blanchard/Devaney/Hall's *Differential Equations* [25] software package that accompanies their text), but now these run into problems with newer security policies, or the lack of "Apple Certification."

Worse, Java will not run on mobile devices such as iPads or Google ChromeBooks. These newer devices, which are becoming ever more ubiquitous, require JavaScript, (which is entirely different from Java), or some other newer language; a simple translation is not possible.

True, we have much larger computer algebra systems (CAS) such as *MatLab*, *Maple*, and *Mathematica* that do many other things, as well as much more specialized (but

[1]Both *IDE* and the *MIT Mathlets* were originally developed by Hubert Hohn, a master programmer at the Massachusetts College of Art. They were then reprogrammed in Java, and now, in the latter case, in JavaScript for mobile devices.

powerful) packages like *Scicos* and *XPP* for dynamical systems. But these require much syntax and computer smarts, so although they are appropriate for research, they are too overwhelming for introductory courses (unless one is operating in a department that already uses such software for other courses, in which case many colleagues have made good use of them). Problems with these powerful multi-purpose programs at schools where they are not entrenched include

- Cumbersome slow processes to enter equations and graph a number of trajectories, (See Example 4 for a most disappointing attempt to use *Wolfram Alpha*.)

- A tendency to simply put up *finished* phase portraits—students don?t have the grip they get when they see solutions evolve; they must do a lot more "figuring out" to comprehend what they are seeing. (This objection could be ameliorated by introducing concepts with interactive illustrations in class or on homework, such as can be provided by *IDE* [27] or *MIT Mathlets* [31].)

Amazingly (to us), with all the applications that abound on the internet, by the end of 2015 we had found a number of specialized DE examples, but *nothing* that did what our old *MacMath* programs did in the 1980s — the ability to enter any DE, see a vector field, and point/click to start solutions. The simple ideas simply have not carried forward, or have been obscured by adding too many other options. Newer software like *GeoGebra* has potential, but we did not find a general DE application that would do what we want. *TI-Nspire* by Texas Instruments is multipurpose mathematics software that *does* run on iPads, and *does* handle ODEs, but I have only seen heavy use of this with colleagues in Denmark, not the U.S.

At the JMM 2016 CODEE sessions, I acknowledged that I was preaching to the choir, but I think we *all* have to *work* to make access, and *use*, of simple graphic DE solvers the norm in an introductory DE classroom. Note: Since I made this speech, we have been alerted to two new possibilities—see the **Good News** at the end of Section 5.

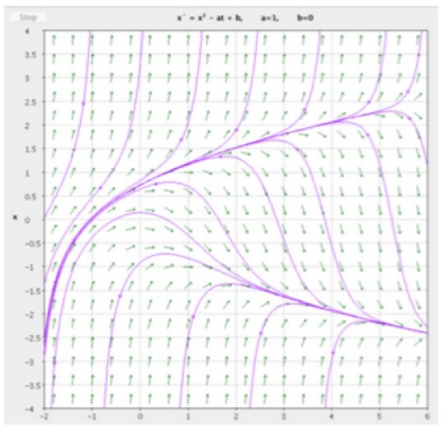
The revolution of the late 1980s vastly empowered students and practitioners of differential equations (and of iterative or difference equations). The majority of real-world differential equations are *nonlinear*, and without formula solutions. Students should *not* have to wait until a later course to encounter these in a global pictorial form.

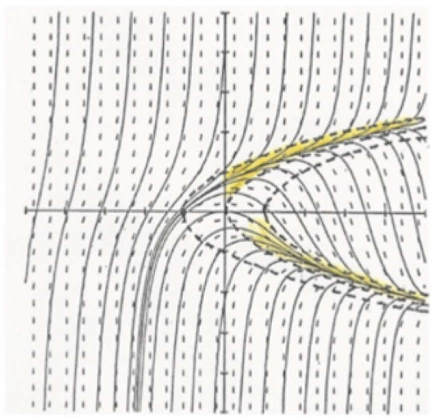
With graphic solutions to differential equations, one can now attack (the majority of) DEs that do not *have* analytic solutions by the traditional cookbook methods. Now that we can *see* previously invisible solutions to DEs, we can concentrate on the global *behaviors*, and how to analyze them.

3 Motivating Examples

Example 4. Necessity of a Qualitative Approach. Our favorite example, $dx/dt = x^2 - t$, *looks* very simple, but this differential equation is not separable, linear, or exact. In fact, by a continuous version of Galois theory, there exists *no* solution expressible in terms of elementary functions [11].

But, that does *not* mean there exist no solutions! We can *see* them!



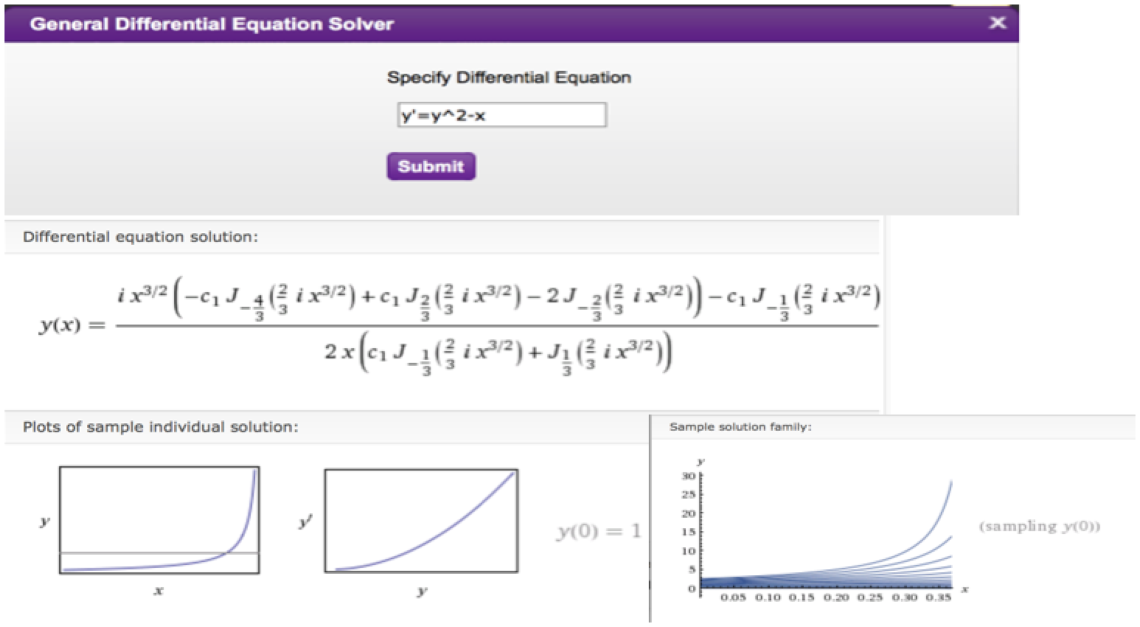


$x^2 - t = 1$
$x^2 - t = 0$
$x^2 - t = -1$

It is now easy to observe the overall *behavior* of solutions. In fact, we can observe, and prove, many statements, such as

- Solutions either fall into a "funnel" and are attracted asymptotically to the lower half of the parabola $x^2 - t = -1$, or they fly off to infinity.
- The upper half of the parabola $x^2 - t = 1$ bounds a separatrix that divides these two behaviors, as you might conjecture.
- Solutions at the top and bottom of the graph (in forward and/or backward time) have vertical asymptotes.

Compare these insights with how, on the next page, Wolfram Alpha responds to an innocent student request for solution to the same simple equation, $dx/dt = x^2 - t$:



General Differential Equation Solver

Specify Differential Equation

y'=y^2-x

Submit

Differential equation solution:

$$y(x) = \frac{i\,x^{3/2}\left(-c_1 J_{-\frac{4}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right) + c_1 J_{\frac{2}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right) - 2 J_{-\frac{2}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right)\right) - c_1 J_{-\frac{1}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right)}{2\,x\left(c_1 J_{-\frac{1}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right) + J_{\frac{1}{3}}\left(\frac{2}{3}\,i\,x^{3/2}\right)\right)}$$

Plots of sample individual solution:

$y(0) = 1$

Sample solution family:

(sampling $y(0)$)

Take a closer look. This formula solution from *Wolfram Alpha* is incomprehensible (and as promised by Hubbard's result [10], is *not* in terms of *elementary* functions)[2].

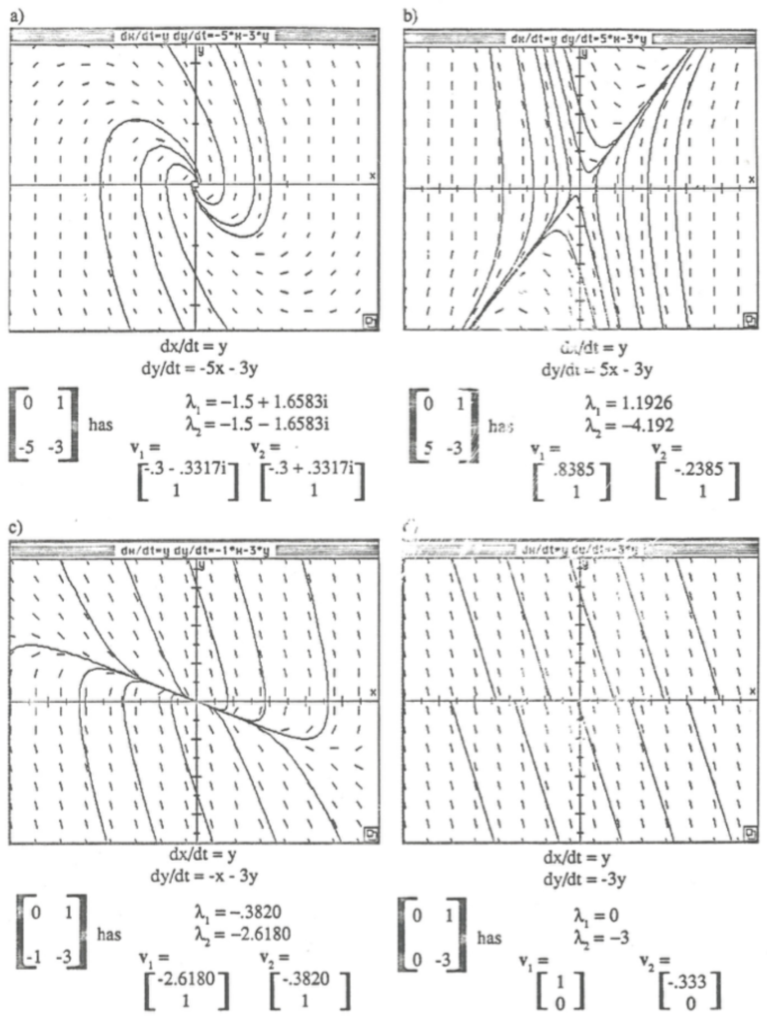
[2]We *are* aware that solutions of this sort *can* be found by recognizing the original equation as a special case of a Ricatti equation, and using a lot of manipulation with change of variables. See Blanchard, Devaney and Hall [4], pp 731-732, for a particularly nice treatment.

Worse, this presentation gives no clue of the overall *behaviors*! The “sample” solution (who decides that?) cannot be called “typical”—there is no suspicion of the funnel behavior for an initial condition with $t_0 > 0$. The individual solution axes have no numerical labels; the sample family solution *does* have numerical labels, but they are of such different scales as to be incomparable with the plots given on p.7.

Example 4 is a sober reminder of how badly we need to give students the power to handle such a “rogue” DE. We can so easily prepare them. We only need software that does not overreach, that keeps within bounds students can comprehend.

At the same time that we are begging for qualitative treatment of behaviors of solutions to differential equations, we must emphasize that we are not throwing out the traditional quantitative approaches. We are simply stretching their capabilities in very important ways. Let me paraphrase John Hubbard in [10]:

> An explicit solution is a wonderful thing, especially if the formula is analyzed and graphed to emphasize the *behavior* of the solution. Solutions of differential equations are *functions*, and we seek to know where they are *increasing or decreasing*, are *stable or unstable*, have *maxima or minima*, have *vertical asymptotes*, or show *oscillatory behavior*.



Example 5. Linear Systems. One can *see* eigenvectors. The worksheet on the previous page is from a lab asking students to compute eigenvalues and eigenvectors of the matrices, then make phase portraits showing the eigenvectors and direction arrows.

They could *see* that

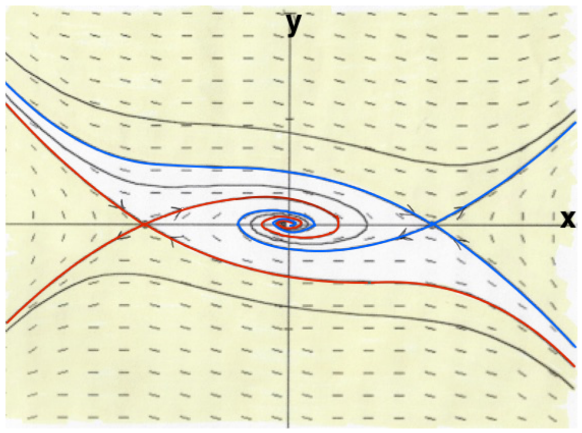
- for *positive* eigenvalue λ, trajectories point *away from* the origin;
- for *negative* eigenvalue λ, trajectories point *toward* the origin;
- for *zero* eigenvalue λ, trajectories still cannot cross each other;
- for *complex* eigenvalues λ, trajectories *spiral*, toward or away from the node depending on whether the real part of λ is negative or positive, respectively.

That is, for linear differential equations with constant coefficients, the behavior of solutions is controlled by the *signs of the real parts of eigenvalues.*

This use of linear algebra really comes to life when studying nonlinear DEs and linearizing them near their zeroes.

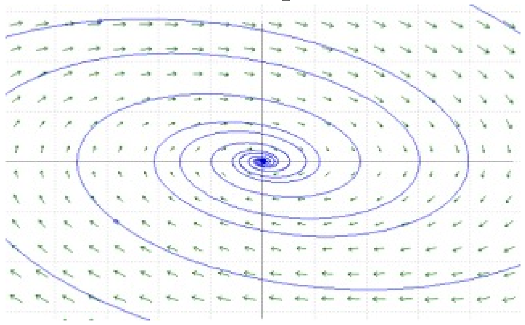
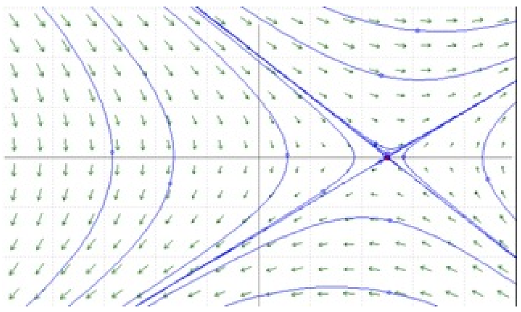
Example 6. Linearizing a Nonlinear System. Consider the nonlinear system [3]

$$x' = y, \quad y' = -0.25x + x^3 - 0.2y$$



The nonlinear analog of the eigenvectors are the *separatrices*, which separate regions of different behavior. Understanding separatrices (also called stable and unstable manifolds) is the most powerful tool we have to study nonlinear equations. In Exammple 6 The separatrices of the left hand saddle are shown in red, those of the right hand saddle in blue. Note the warped basin of the stable equilibrium that results. Linearizing about an equilibrium shows the familiar patterns seen in linear equations, but we see that these approximations are good only in an asymmetric region of the appropriate equilibrium.

[3]This system is an example in Blanchard/Devaney/Hall [2]—they give a nice (crude) earthquake model on the structure of a building, with very lopsided neighborhoods of the equilibria, emphasizing the fact that one needs to know *where* a linearization might give a *good* approximation.

Linearization about spiral node at $(0, 0)$	Linearization about right hand saddle at $(0.5, 0)$
 $\frac{dx}{dt} = y$ $\frac{dy}{dt} = -0.25x - 0.2y$	 $\frac{dx}{dt} = y$ $\frac{dy}{dt} = 0.5x - 0.2y - 0.25$

For the majority of differential equations in real world applications, solutions may not be expressible in elementary terms, or even as integrals of elementary functions. Any course in differential equations can and should provide tools to describe the behaviors of solutions without finding them explicitly.

Moreover, we must be careful to assess our *numerical methods* that create the graphical solutions—to know that the solutions we are airily describing actually exist. We can do this by proving that an effective approximation procedure converges. For example, see [4], [12], [13].

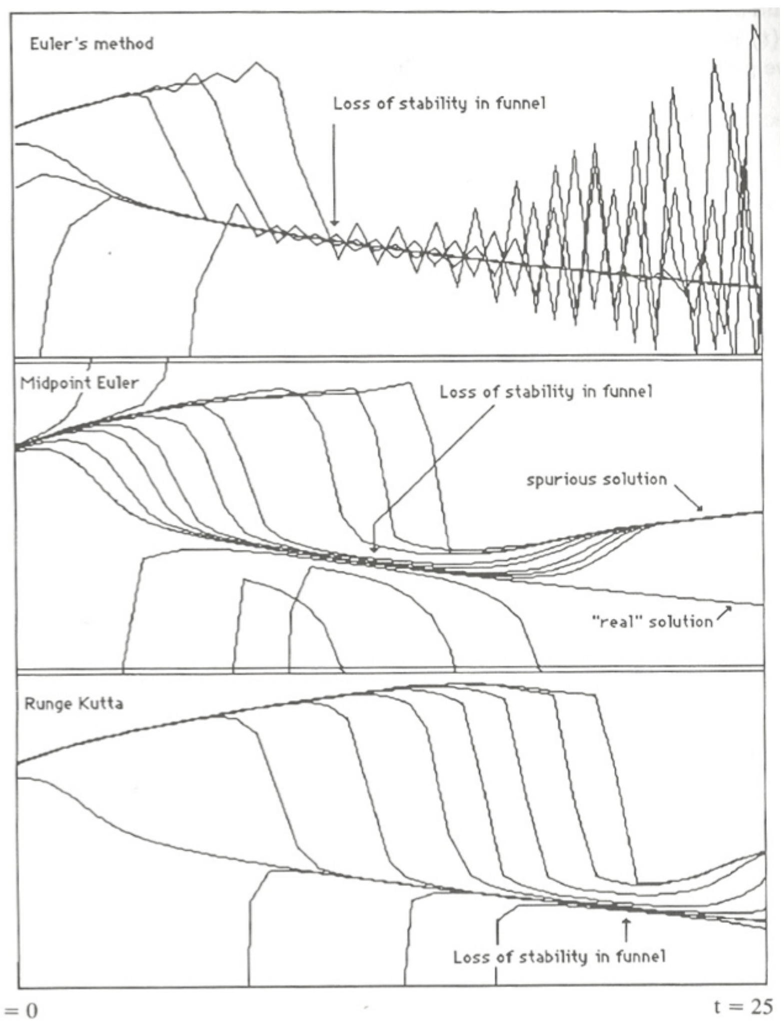
Example 7. Comparing Numerical Methods. In the text by Hubbard and West [12] we use Euler, midpoint Euler, and 4th order Runge Kutta methods. Software is essential to see how these methods work in practice, and what sorts of convergence one finds. The *MacMath* [28] program *Numerical Methods* was essential to convey what the various methods do, how they can behave and misbehave. An unforgettable impression from [12] illustrates the fact that eventually *any* numerical method breaks down, emphasizing the need for theory that pays attention to bounding errors, as shown on the next page.

4 Software Questions

What simple ODE software do we want?

The focus must be just on *simple* operations, made most transparent, to:

- Enter any differential equation or system, especially nonlinear.
- Put up a direction field, or vector field in the phase plane for a system.
- Click on the direction field to choose an initial condition.
- See solutions *evolve*, one by one (quickly, but not too quickly).



Iterating $x' = x^2 - t$ by different methods, for fixed stepsize $h = 0.4$.

This gives the student control that *interprets* what he or she is seeing. My previous examples depend on these features.

The next level, also made *simple*, is to allow:

- Choosing numerical method and stepsize (important for analysis).
- Numerical entry of initial conditions when it is critical.

With a little linear algebra behind the scenes, we can also

- Locate, and identify equilibria.
- Find eigenvectors at equilibria.
- Draw separatrices of a saddle (as initial conditions, take tiny steps in the eigenvector directions on either side of the saddle, for a total of four initial conditions).

These are all *simple* requests. Additional nice features included on many of the simple original software packages were the abilities to

- Add (in a different color or line style) a parametric curve (e.g., isoclines, nullclines).

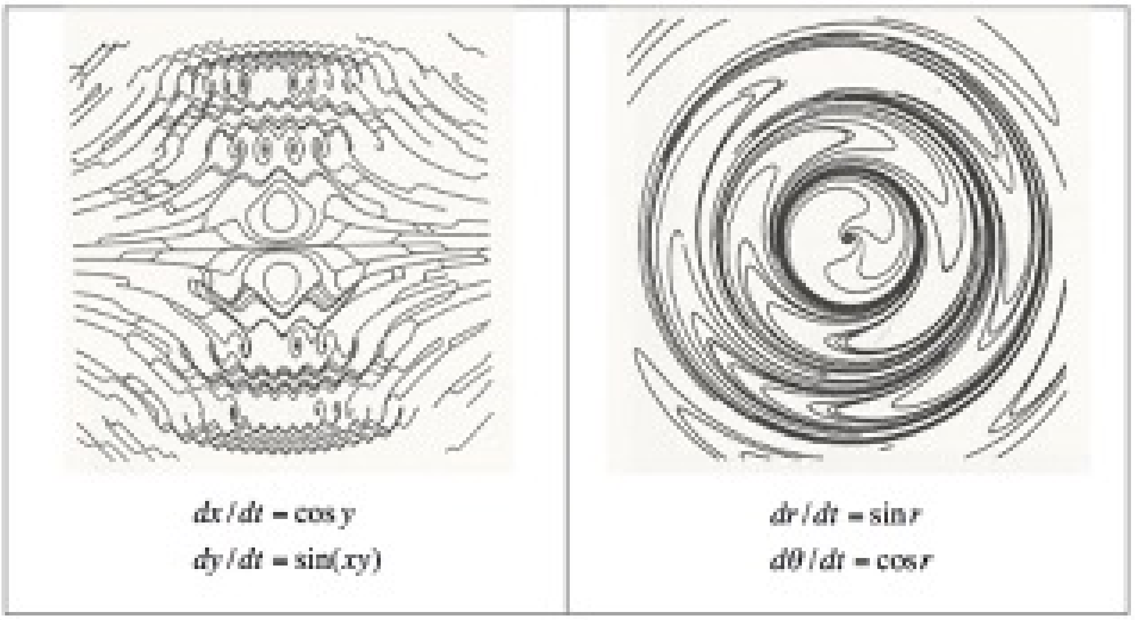
- Clear trajectories (all, or just a selected trajectory).
- Color or thicken any particular curve on screen for highlighting.
- Flow a *region* of initial conditions.

Yes, this is a downright plea for *dedicated* DE software (apps?) that can do these things for any DE system that one can enter (in normal mathematical syntax)!

5 More Examples and Questions

Look at what was done 30 years ago, with only the most basic technology:

Example 8. Brief Code on Cash Register Tape. By 1983 Michelle Artigue and Veronique Gautheron in Paris [1] were doing all of this, with high resolution results on a plotter, by short programs *written on a couple of inches of cash register tape*!

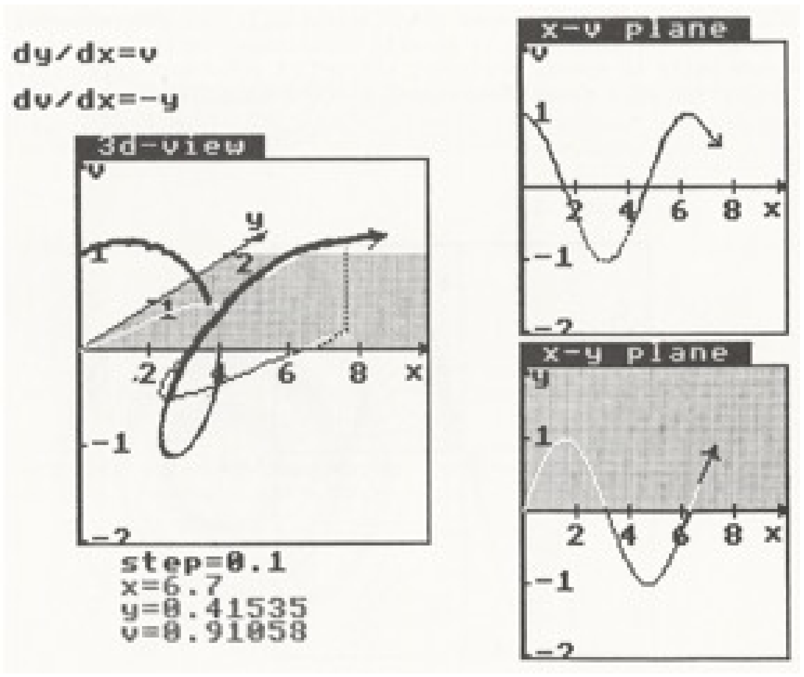


Example 9. System Pictures in just 7 Kilobytes. By 1985 David Tall at University of Warwick in England had also done these things, even for second degree differential equations, on a BBC computer *with just seven kilobytes of memory* (and in color, too). [34] (See next page.)

Why do we want this simple ODE software?

- For student understanding:

"Lets you feel it in your bones" (Rod Smart, University of Wisconsin)



- For the ability to explore and experiment:

  What happens if . . . ?

  What can we change (coefficients, or entire model), by how much, to get a desired result?

  Predator/Prey or Competition models ? e.g., we can change the food supply coefficients, import more predators, include effects of hunting, etc.

  Creative assignments—e.g., Borrelli and Coleman at Harvey Mudd [2, 3] gave a problem to “Draw a Cat”—i.e., to find a system of differential equations that would give a phase portrait resembling some kind of cat. This assignment gave lots of different results, fun teamwork, and an understanding of what equations are needed to produce equilibria in desired spots, with spirals or nodes.

  Serious problem solving.

- To give students more immediate access to real world applications.

  See Example 10 is for a pharmacokinetic model.

A list of Software Resources that allow these simple graphing techniques appears on pages 20-21.

The Good News

Since this was given as a talk at the JMM 2016, we have been alerted to two software efforts of simple straightforward packages that extend to mobile devices! Darryl Nester [32] of

Bluffton University has created *SlopeFields*, a wonderful app in JavaScript that runs on both computers *and* mobile devices. One enters a DE in one or two independent variables, the vector field appears, and with a point/click entry of initial conditions, solutions evolve. This app is freely available on the web, which should be most helpful in our campaign to get the power of computer graphics into every DE course. At JMM 2017, Tim Lucas of Pepperdine University introduced *Slopes* [30], which includes additional capabilities. Currently available for the iPad and iPhone, adaptations of *Slopes* for other platforms are in the works.

Let me conclude my arguments with two final examples of the power enabled by interactive computer graphics—the sort of facility that our introductory DE courses can, and should, give to *all* students. Those in nontraditional fields are particularly in need of simple and clear visualizations. These graphical techniques are especially valuable and easy to use with systems of *nonlinear* differential equations, which are the dominant type in real world modeling.

Example 10. Pharmacokinetics. Ed Spitznagel presented in an early CODEE newsletter [19] an analysis of a simple two-compartment model that gives a good approximation of how drugs move through the gastro-intestinal tract and then the bloodstream (the two compartments, quantified by subscripts G and B respectively).

Half lives determine decay rates $k = \frac{\ln 2}{\text{half life}}$.

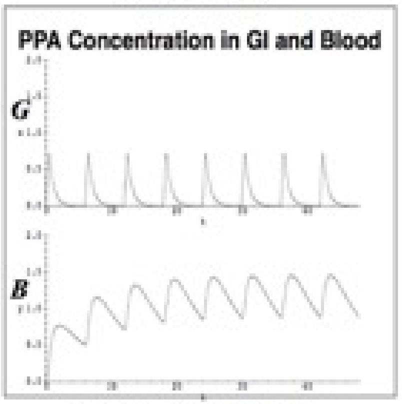
An input function $f(t)$ gives a pulse (width $\frac{1}{2}$, height 2) every 6 hours.

For a common over-the-counter cold drug having two components, with x = the amount of decongestant and y = the amount of antihistamine, the resulting ODE system is

$$\frac{dx}{dt} = f(t) - k_G x,$$
$$\frac{dy}{dt} = k_G x - k_B x.$$

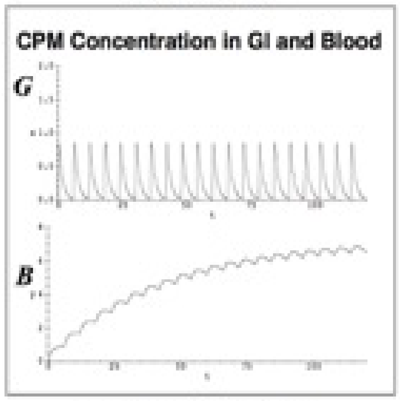
Resulting graphs for each component, and both compartments, are easily comprehended.

Decongestant



half lives: ½ hr, 5 hrs

Antihistamine



half lives: 1 hr, 30 hrs

Many simple questions are ripe for exploration with this system:

- What are the effects of skipping a dose? Doubling a dose?
- Would a larger initial “loading” dose be effective?
- How would it work with sustained release tablets (small beads in resins of various dissolution rates)?

Notes:

- The FDA requires the same graphical results for dispersal of drugs through a multi-compartment model in order to certify a generic version of the combination drug.
- Two sophomore pharmacology students got so excited with this project that they created a (fictitious) “designer drug” and went so far as to prove that the prescribed dosages would not exceed a certain limit!

Example 11. Extensions of Predator/Prey Problems. Tony Danby [6,7] noted that a third species, having population z , can be introduced into a Volterra predator/prey model in many ways. For instance, it might prey on both of the others, but be preyed upon by neither. An example is

$$\frac{dx}{dt} = x - xy - 0.2xz$$
$$\frac{dy}{dt} = -y + xy - 0.1yz$$
$$\frac{dz}{dt} = -z + 0.8xz + 0.1yz$$

A good exercise is to interpret each of the graphs (on the next page) in terms of the populations in the model. Then one could suggest how changing coefficients would alter the results.

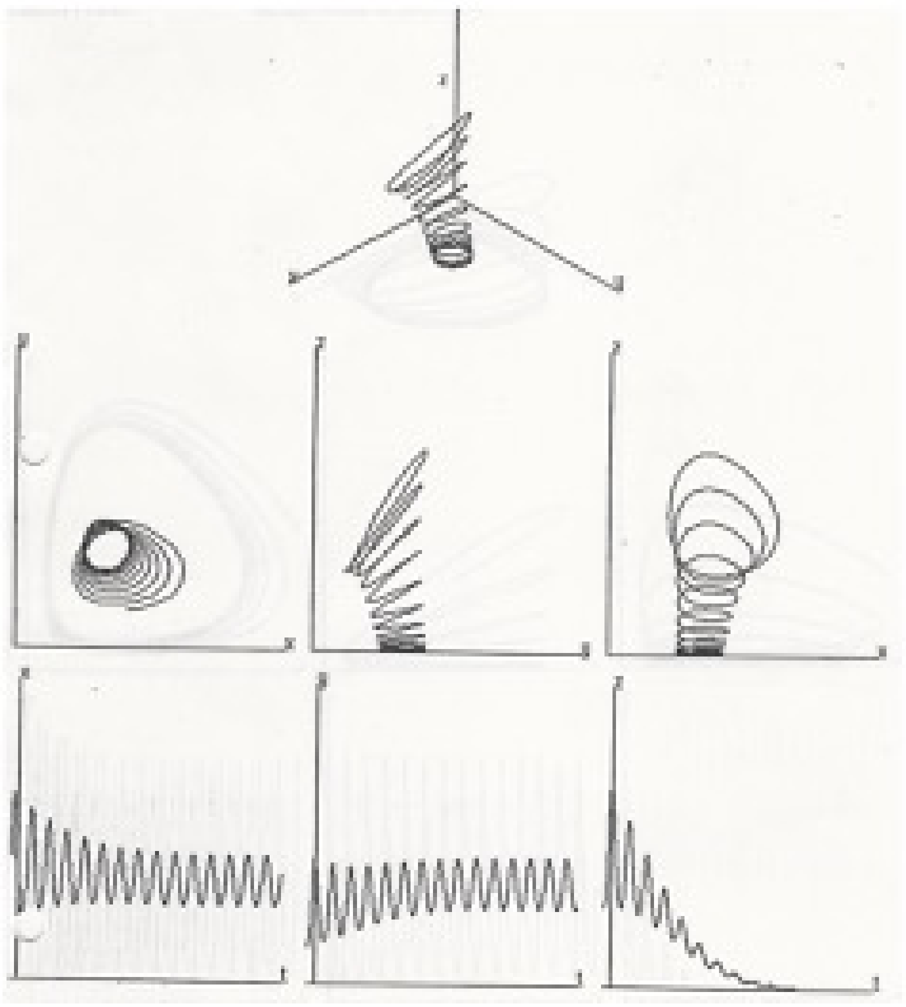
Software *dedicated* to differential equations (as in this example, by *MacMath* [28]) can produce a collection of graphs like this instantly; then playing with coefficients to explore modifications is *easy*.

6 Call To Action

Please encourage and help your colleagues with efforts to visualize the *behaviors* of solutions to differential equations. Do anything you can to bring about appropriate simple software, especially for mobile devices. Nester’s *Slope Fields* App [32] and Lucas’s *Slopes* [30] give a good start.

Please carry on the fervor for qualitative analysis of solution behaviors. This is especially essential for *nonlinear* differential equations, which should be covered in *all* introductory differential equations courses. For an article on this point in particular, see [24], as well as [10] in the collection, *Dynamical mathematical software and visualization in the learning of mathematics*, edited by Samer Habre [9].

Sample resources are listed below in two sections, first for text and then for software.



Resources (text):

[1] M. Artigue and V. Gautheron. *Systèmes Differentiels: Ètude Graphique*, CEDEC, Paris, 1983.

[2] P. Blanchard, R.L. Devaney, and G.R. Hall. *Differential Equations, 4th edition*, Brooks Cole. CEngage, 2011.

[3] R. Borrelli and C. Coleman. *Mathematical Methods, Models and Applications for Engineers, Mathematicians and Scientists*, 1977. The "monster" text, archived on line at URL `http://ccdl.libraries.claremont.edu/col/mmm`.

[4] R. Borrelli and C. Coleman. *Differential Equations, A Modeling Perspective* Wiley, 2004.

[5] CODEE (Community of Ordinary Differential Equations Educators). URL `http://www.codee.org`, 2007. Website with *CODEE Journal* for articles, projects, and reviews. Includes newsletters from 1994, from original NSF Consortium for Ordinary

Differential Equations Experiments, and other archives. See the article *Evolution of the Modern ODE Course in the CODEE Journal.*

[6] J.M.A. Danby. *Computing Applications to Differential Equations: Modeling in the Physical and Social Sciences*, Reston Publishing Co, 1985. A classic—the best source I have found of many models, with variations on each.

[7] J.M.A. Danby. *Computer Modeling: From Sports to Spaceflight . . . From Order to Chaos*, Willman Bell, Inc., 1997. An update of the above classic; includes CD-ROM, with IBM-PC Software.

[8] J. Farlow, J. Hall, J. McDill and B.H. West. *Differential Equations and Linear Algebra,* Prentice Hall (2002), second edition Pearson, 2007. Incorporates IDE [28].

[9] S.S. Habre (Editor). *Enhancing Mathematics Understanding through Visualization: The Role of Dynamic Software.* IGI Global, 2013. Twelve chapters by fifteen international specialists provide valuable insights on this phenomenon.

[10] J.H. Hubbard. Technology and Differential Equations, *Enhancing Mathematics Understanding through Visualization: The Role of Dynamic Software.* IGI Global, 2013. pp. 1-11.

[11] J.H. Hubbard and B.E. Lundell. A First Look at Differential Algebra *American Mathematical Monthly*, March 2011. pp. 245-261.

[12] J.H. Hubbard and B.H. West. *Differential Equations, A Dynamical Systems Approach, Part I: The One-Dimensional Theory.* Texts in Applied Mathematics #5, Springer Verlag, 1991.

[13] J.H. Hubbard and B.H. West. *Differential Equations, A Dynamical Systems Approach, Part II: Higher Dimensional Systems.* Texts in Applied Mathematics #18, Springer Verlag, 1995.

[14] J.H. Hubbard and B.H. West, *Èquations Différentielles et Systèmes Dynamiques.* Cassini, Paris, 1999. A translation and adaptation by V. Gautheron combining books [13] and [14].

[15] H.R. Miller and D.S. Upton. Computer manipulatives in an ordinary differential equations course: development, implementation, and assessment, *Journal of Science Education and Technology* #17, 2008, pp. 124-137. Available at URL `http://math.mit.edu/mathlets/wp-content/uploads/cet-published`.

[16] *MIT Open Courseware.* 2001-2018. URL `http://ocw.mit.edu`.

[17] *MIT OCW Differential Equations Course.* 2010. URL `http://ocw.mit.edu/courses/mathematics/18-03-differential-equations-spring-2010/`.

[18] SIMIODE (Systematic Initiative for Modeling Investigations and Opportunities with Differential Equations). URL http://simiode.org. 2013-2018. An active open community of teachers and learners using modeling first for differential equations in an original way. Website includes valuable resources including course materials, newsletters, blog, and information about ongoing SCUDEM competitions.

[19] E. Spitznagel. Two-Compartment Pharmacokinetic Models. *CODEE Newsletter* Fall 1992, pp. 2-4.

[20] D. Tall. Recent developments in the use of the computer to visualize and symbolize calculus concepts The Laboratory Approach to Teaching Calculus, M.A.A. Notes #20, 1991. pp. 15-25.

[21] D. Tall and B.H. West. Graphic Insight into Calculus and Differential Equations, *The Influence of Computers and Informatics on Mathematics and its Teaching* (ICMI Study Series, Strasbourg). Cambridge University Press, 1986. pp. 107-119.

[22] B.H. West (Editor). *Special Issue on Teaching Differential Equations*, College Mathematics Journal Vol 25 No.16. (1994)

[23] B.H. West. Technology in differential equations courses: my experiences, student reactions, *Revolutions in Differential Equations*, MAA Notes 50. Math. Assoc. America, Washington, DC., 1999. pp. 79-89.

[24] B.H. West. Nonlinear is essential, Linearization is not enough, Visualization is absolutely necessary, *Enhancing Mathematics Understanding through Visualization: The Role of Dynamic Software.* IGI Global, 2013. pp 89-112.

Resources (software):

[25] P. Blanchard, R.L. Devaney, and G.R. Hall. *Differential Equations, 4th edition*, Brooks Cole, 2011. CEngage Software available to those who purchase this text includes both interactive illustrations (originated by Hubert Hohn) and simple graphic solvers.

[26] *Harvey Mudd College ODE Toolkit.* 2011. URL http://odetoolkit.hmc.edu/. An update of the original program from the late 1980s.

[27] H. Hohn, with B.H. West, S.H. Strogatz, J.M. McDill, and J. Cantwell. *Interactive Differential Equations.* Addison Wesley, 2000. Available at URL https://media.pearsoncmg.com/aw/ide/index.html. Interactive illustrations with labs; not a "solver", but incredibly useful in conveying understanding of concepts. Incorporated in text by Farlow et al [8], and bundled with two Addison Wesley differential equations texts (by Nagel, Saff & Snider, and by Kohler & Johnson).

[28] J.H. Hubbard and B.H. West. *MacMath 9.2.* New York, N.Y. Springer Verlag, 1992.

[29] J.H. Hubbard and B.H. West. *MacMath 10.0*. unpublished, 1999. It only runs on the Mac Classic operating system. But the programs for 2D differential equations and 2D maps are available at URL `http://www.math.cornell.edu/~dynamics/BenHinkle/index.html.`

[30] T. Lucas, J. Haug, et al. *Slopes: A Differential Equations Graphing Environment*. 2017-2018. There are five parts—Slope Fields, Phase Planes, Systems, Waves, and Methods—that create an interactive "playground" for ODEs. You can use it for homework, in-class activities or a new research project. *Slopes* is currently available for the iPad and iPhone via the App Store; the authors hope to expand to other platforms in the future. See full description at URL `https://sites.google.com/a/pepperdine.edu/slopes/home,`.

[31] MIT *Mathlets*. 2009-2018. Also originated by Hubert Hohn, with Haynes Miller. These are available for free at URL `http://math.mit.edu/mathlets/`.

[32] D. Nester. *Slope Fields*. 2016. Gives slope or vector fields for any user-entered 1D or 2D differential equation, with multiple solutions drawn by point/click on these graphs. Includes choice of numerical method and step size. Works on all mobile devices, and Google ChromeBooks. Available for free on the web at URL `http://www.bluffton.edu/homepages/facstaff/nesterd/java/slopefields.html`.

[33] *ODE Architect*. by CODEE (NSF Consortium for Ordinary Differential Equations Educators). John Wiley & Sons, 1998. Dynamic software package built around very powerful graphic solvers. URL `http://www.math.hmc.edu/resources/odes/odearchitect`. *ODE Architect (ODEA)* won the distinction of being named by Forbes Magazine as one of the "nine best digital projects on the planet" in December 1998. This was not just a list of math programs—*ODEA* was the only math program so honored. There had been 1,080 entries in 40 categories submitted to the New Media INVISION 98 Awards and *ODEA* was one of only nine given an Award of Excellence. *ODE Architect* also won a gold medal in the category of Higher Education.

[34] J.C. Polking. *Dfield*, and *Pplane*. 2005. Available at URL `http://math.rice.edu/~dfield/dfpp.html`. Written in Java, so cannot run on mobile devices.

[35] D. Tall. *Graphic Calculus for the BBC Computer*. Glentop Publishers, London, 1985.

[36] J.H. Tutsch. *Ode Factory*. 2011. Available at URL `http://odefactory.com/`. Another graphic solver program, in Java, with many additional features under construction, for both differential and iterative equations.