Technology: IBL, inverse questions, and control

Matthias Kawski
School of Mathematical and Statistical Sciences
Arizona State University
Tempe, AZ 85044, USA
https://math.asu.edu/∼kawski
kawski@asu.edu

Abstract

New computational tools become available at every faster rates. A fundamental question asks how such tools can help achieve clearly defined learning objectives. This article argues, in a sequence of examples, that computing technologies can much support the implementation of modern pedagogy. The focus is on enhancing a learner centered environment. Of critical importance is that the learner takes the key role of asking further questions, to take ownership of the discovery experience. We highlight the special role of asking inverse questions with examples ranging from math circles to vector calculus.

1 Introduction

The invention and development of computing tools for mathematical tasks goes back well over 4000 years to precursors of the abacus. What has dramatically changed more recently is the pace at which new tools appear. Some 300 years ago the slide rule was invented. Hand-held calculators with arcane interfaces and syntaxes, and personal computers with huge packages such as MATLAB, now have to compete with ever more intuitive *apps* that are ubiquitous on smart-phones. Some programs and applications still have steep learning curves. But the latest generation of students has very different attitudes towards, and experiences with, installing new *apps* just as needed, quickly learning how to use them to address their problems. Well designed *apps* can go viral in an instant, bad ones are quickly forgotten. Many teachers lag far behind – just think of why not allowing smart phones during final exams?

Our focus here is not on specific computational tools. Indeed our examples largely use spreadsheets, and basic computer algebra systems, just because they were most readily available. It should be easy to adapt any of these to similar platforms. Instead, this article is concerned with what we can and want to use technology for when learning mathematics. The central tenet is to exploit computing technologies to foster student-centered learning environments, in some adaptations of inquiry based learning pedagogies. Before jumping into the fray, briefly recall some of many other uses of technology for mathematics learning that we will not address here. Flash-cards for rote learning of definitions (vocabulary), and similar, have long transitioned from cardboard to electronic form. Online homework and tests have long become the standard for drill exercises for both practice and assessment. It used to be that online collaboration in mathematics was much hampered by the difficulty to transmit and convey mathematical symbolic notation and diagrams. Rapidly evolving communication technologies much facilitate this, too. At this place we like to share a little javascript that much enhances communication in class, by randomly selecting and calling on students, so that everyone gets a fair share, rather than a class dominated by the same few students. In our highly interactive classrooms, this script has a black black-ground so that projectors need not be turned down when using the board for other purposes.

```
<HTML><BODY BGCOLOR=BLACK>
<button onclick="randName()">Select a student</button>
<FONT COLOR=YELLOW SIZE=+2><B><P ID="demo"></P></B></FONT>
<SCRIPT>
function randName() {
var students = Array("=Ann","=Bob","=Charmayne","=Yee-Yang");
    var n = Math.floor(Math.random()*students.length);
    document.getElementById("demo").innerHTML = [n+1 + students[n]];}
</SCRIPT>
</BODY></HTML>
```

Back to our central theme: How to best use computing technology to enhance a student-centered learning environment? A key objective is that the students take ever more of a leading role as being the ones who ask the questions. Starting with one common problem, the plan is that this is just a seed for a string of questions that delve ever deeper. Questions lead to experiments and conjectures, further tests, proof where appropriate, and new questions. Some problem solving strategies may become formalized algorithms. In this context computing technology is first used for quickly checking many examples, generating large data sets which then generate and test conjectures, and open new questions.

The following sections discuss in more detail the concept of inquiry based learning, and the role of inverse questions, and how computing technology can much enhance such learning environment. The examples and settings range from a math circle for elementary teachers to vector calculus.

2 Inquiry based learning using technology

There is no universally accepted definition of inquiry based learning. Yet most users agree that a key component is that students are to take the lead role in asking (research) questions in the process. For a working definition or description we refer to the Academy of Inquiry Based Learning [1]. For longstanding guidelines calling for such student experiences see [8].

The following example is taken from a math teachers circle that intended to demonstrate inquiry based learning to elementary teachers. Math circles have become very popular in recent years as means to provide school age students (and teachers!) with extra-curricular math activities. For more details visit the National Association of Math Circles [7]. Math Circles are also well suited for doing teaching experiments that instructors may not be able to try in their regular classrooms.

The setting was a math festival which featured an rectangular array of *stations* (tables) in a large gymnasium, each station offering different math activities. A simple question was asked:

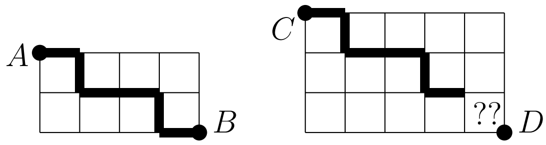


Figure 1: How many different paths from A to B? From C to D?

"In how many different (shortest) ways (walking between the tables) can one get from one corner of the gym to the diametrically opposite one?", as illustrated in figure 2. The first technology employed consisted of paper and pencil, teachers drawing many possible routes, checking that none were missing, eventually labeling them e.g. $RRRRDD$ or $RDRRDR$ (R for right and D for down). It took them substantial time to propose a recursive way to systematically find all routes. The key is to embed the specific 2×4 or 3×5 problem in the family of problems of $m \times n$ grids of all possible sizes. First tabulating results by hand (with the key change of switching vertices and faces in the grid), once the recursion formula $f(m,n) = f(m-1,n) + f(m,n-1)$ was understood, it was natural to use computing technology to get more numbers.

B3 =B2+A3

	A	B	C	D	E	F	G	H	I	J
1										
2		1	1	1	1	1	1	1	1	1
3		1	2	3	4	5	6	7	8	9
4		1	3	6	10	15	21	28	36	45
5		1	4	10	20	35	56	84	120	165
6		1	5	15	35	70	126	210	330	495
7		1	6	21	56	126	252	462	792	1287
8		1	7	28	84	210	462	924	1716	3003
9		1	8	36	120	330	792	1716	3432	6435
10		1	9	45	165	495	1287	3003	6435	####

B3 =A2+A3

	A	B	C	D	E	F	G	H	I	J
2		1	1	1	1	1	1	1	1	1
3		0	1	2	3	4	5	6	7	8
4		0	0	1	3	6	10	15	21	28
5		0	0	0	1	4	10	20	35	56
6		0	0	0	0	1	5	15	35	70
7		0	0	0	0	0	1	6	21	56
8		0	0	0	0	0	0	1	7	28
9		0	0	0	0	0	0	0	1	8
10		0	0	0	0	0	0	0	0	1
12		1	2	4	8	16	32	64	128	256

Figure 2: How many different paths on $m \times n$ grid?

A spreadsheet is ideally suited to implement such recursion, e.g. click the formula $B2 = A2 + B1$, and fill down and right. Then enter a single 1 in the cell $B2$, see the left table in figure 2. Once we had numbers it was time to look for patterns. The original defining relation clearly related to the binomial expansion $(R+D)^{m+n}$, but discussion arose whether $RRRRDD$ was equal or not $RDRRDR$, paths versus *real variables.*

Summing the columns (in usual Pascal triangle, the diagonals) was just the start (left table in figure 2). To better see more fun patterns, we slightly rearranged the table, by copying the formula $B2 = A1 + B1$ down and right. Summing all entries, or every other entry of the columns yielded the expected result. But the biggest surprise was summing along the other SW-NE diagonals (right table in figure 2). Looking for patterns, and trying to understand them – our teachers were doing real math, see the right table in figure 2. Some noticed patterns of even and odd numbers – we first colored these on the projector image on the white board (after raising the screen). Again, the desire was to better see the pattern. We needed more numbers.

In the spreadsheet we changed the formula to $B2 = MOD(B1 + A2, 2)$ and used conditional formatting to see the patterns in color, see figure 2. Now we were truly in the running, trying to understand and predict the sizes of the triangles in the fractal patterns.

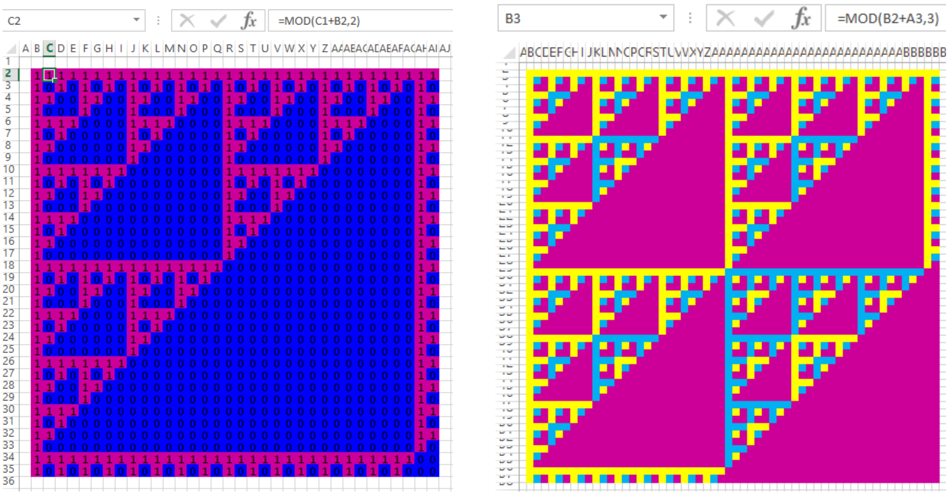


Figure 3: Start to play. More and more patterns and questions.

3 Inverse questions and parameterized families

The near ubiquitous availability of computing technology, especially with graphing capabilities, has trivialized many traditional practice exercises and assessment items. This includes evaluation of algebraic expressions and functions, solving for specific values (finding roots), or graphing individual functions.

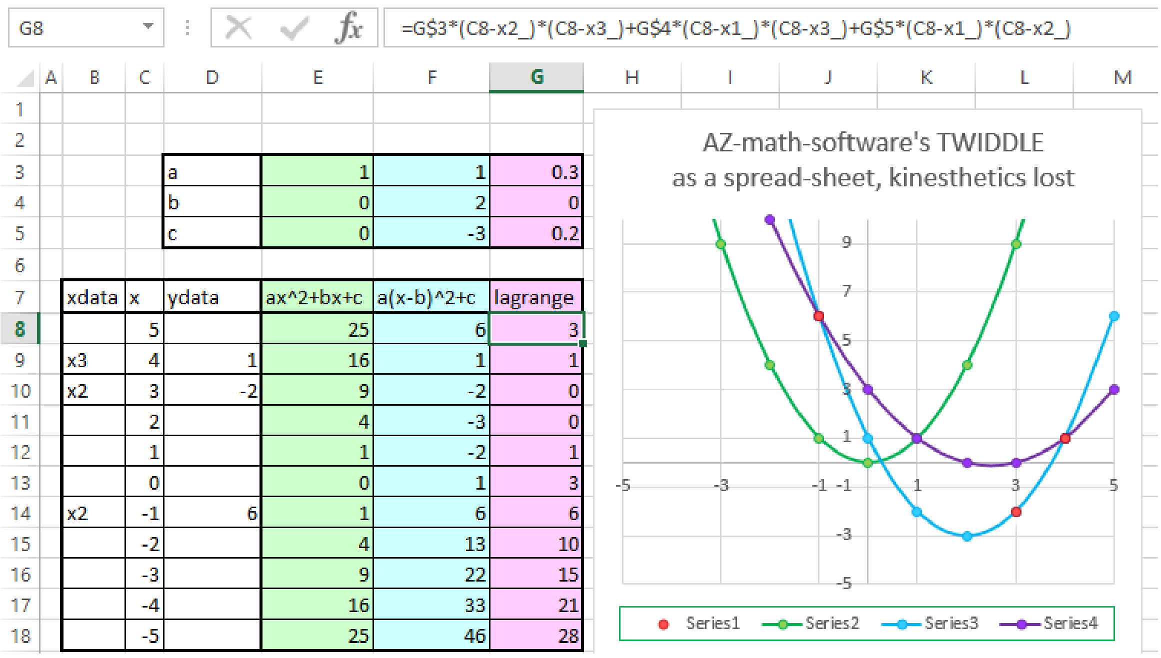
A natural way to address this situation is to ask inverse questions instead. Instead of asking for the graph of a function (given by a symbolic expression), ask for a formula of a function that is presented graphically, verbally, or as a dance choreography, see section 4.

Here a standard mathematical strategy comes into play, namely instead of considering single objects, consider (finitely) parameterized families of objects. Instead of asking for the location of the maximum of a function, find the values of parameters in a cubic function that will place the maximum at a designated place.

Important is that such inverse questions are not seen by the students just as unmotivated mean tricks. The following sections will address this issue – providing well-motivated settings that naturally ask for such inverse questions. For dynamical systems (differential equations) a natural such story involves controls.

The graphing technology can still be used by the learner, but instead of having an almost trivial graphing or zooming problem, now the technology is used totest conjectures, get a feeling for the graphical and geometric roles of the parameters, and to strategize. As a rule of thumb, graphing problems involving one-parameter families of functions generally are straightforward with graphing technology, using simple trial and error, or bisection methods. Three or more parameters are generally hard to solve by random guesswork.

A simple example at the pre-calculus level serves as an illustration, see figure 3. Given three points (x_i, y_i) in the plane with pairwise different first coordinates, the task is to determine a quadratic function that interpolates these points. Serving as a counterpoint to an algebraic, symbolic solution this task is about exploring the graphical role of the parameters. The example compares three different parameterizations $Ax^2 + Bx + C$, $a(x-b)^2 + c$, and similar to the Lagrangian interpolation formula $\alpha(x-x_2)(x-x_3) + \beta(x-x_2)(x-x_3) + \gamma(x-x_2)(x-x_3)$. It is very easy to write a short app, or little program in most any software to facilitate this. The



G8 =G$3*(C8-x2_)*(C8-x3_)+G$4*(C8-x1_)*(C8-x3_)+G$5*(C8-x1_)*(C8-x2_)

	B	C	D	E	F	G
1						
2						
3			a	1	1	0.3
4			b	0	2	0
5			c	0	-3	0.2
6						
7	xdata	x	ydata	ax^2+bx+c	a(x-b)^2+c	lagrange
8		5		25	6	3
9	x3	4	1	16	1	1
10	x2	3	-2	9	-2	0
11		2		4	-3	0
12		1		1	-2	1
13		0		0	1	3
14	x2	-1	6	1	6	6
15		-2		4	13	10
16		-3		9	22	15
17		-4		16	33	21
18		-5		25	46	28

Figure 4: Curve-fitting: *TWIDDLE* adaptation to a spreadsheet.

screen images show a simple implementation as an EXCEL spreadsheet.

One of the earliest outstanding implementations was in the program *TWIDDLE* of the University of Arizona Software [2], also `http://math.arizona.edu/~dsl/software/software.htm`, running of a 360kB floppy disk in DOS. One of its best features was its kinesthetic component: the values of the parameters were increased or decreased (by adjustable steps, powers of 10) by repeatedly tapping on the c or *SHIFT C* keys, respectively, and the program then showed overlaid families of curves. Some more recent *apps* replace such controls by sliders – but this in some sense defeats the purpose of connecting the shape to numbers, numerical values of the parameters. The EXCEL implementation here asks that the user type the parameter values. Of course, the key observation is that the most immediate linear parameterization is much harder to work with since changing the value of b moves the vertex of the parabola along another parabola. The nonlinear second parameterization much more nicely identifies the graphical meanings and roles of the parameters. The third, and fanciest, parameterization produces the nicest *pinched families* of curves, with each of the parameters not changing the function value at two of the given data points x_i.

A very successful extended project that we have used many times in calculus classes asks students to *write* their names as parameterized (broken) curves. We had many students make amazing efforts to write their names in cursive, using very creative parameterized families of functions. In its easiest implementation, one may simply ask to spell a four letter word that includes at least one of each of the following: horizontal, slanted, and vertical line segments, and some curved arc, see figure 3. Motivated by a pen-plotter or welding robot, the time intervals for the different parts may not overlap. The welding robot story may suggest that pieces are reparameterized by arc-length. Of course, the computer graphing technology is utilized for many adjustments of the parameters to catch mistakes, and to make the curves even prettier. Grading is trivial, as the perfect picture is evidence for a correct formula that works! Which student would want her or his name look ugly? Pride and beauty are very powerful motivators

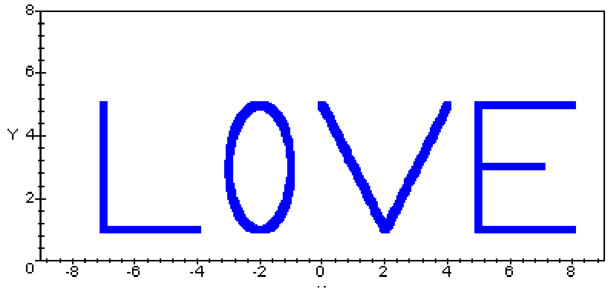
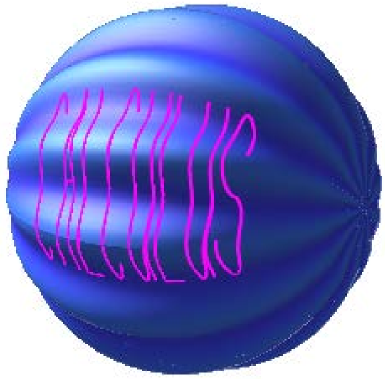
 

Figure 5: From curves to formulas: Spell your name.

for outstanding efforts. Our project usually asks to also plot the speed, components of the acceleration, and curvature along the broken curves. As an extension, students are invited to compose the curve with a parameterized surface, e.g. resulting in some message advertised on a blimp, as shown in the sample picture in figure 3.

4 Inverse questions and control: Falling cat

In first year calculus the parameters, that inverse questions typically ask for, are (finite sets of) static numbers. In vector calculus and differential equations, often the natural parameters are functions. As we said before, it is important to provide students with reasonable scenarios that motivate such questions. Otherwise they might just be regarded as mean questions by an instructor who makes things unreasonably harder.

A typical setting involves line integrals: With common computing technology it is rather trivial to evaluate an integral when given the integrand and the curve. It is difficult to assess a student's understanding from such assignments. Our preferred twist is to give the integrand and desired value of the integral. In general, such problem has infinitely many solutions, and one may add additional constraints to the task. But the students will demand a nice story for this. This is what this example is about.

Mathematical models for falling cats have a long history, dating back at least to Stokes and Maxwell in the 1800s. As can be seen in many popular videos on-line, cats have an uncanny ability to quickly turn around when falling so that they land on their feet. Of course, a key constraint is that if a cat starts with zero angular momentum, then this zero angular momentum is preserved throughout the fall. How then does the cat turn around? The answer, of course, must rely on the cat not being a rigid body. It is not enough to model the cat as two connected rigid bodies, like front and back, that can twist relative to each other: as one end rotates one way, the other rotates the other way so that at all times the angular momenta cancel. But, of course, the cat does not land twisted. Several related models and mechanisms are discussed in the collection [3] which deals with various geometric aspects of nonholonomic mechanical systems. An advanced model is analyzed in [6]. But here we will analyze one of the simplest systems that explains the underlying geometry, called a *planar skater* in [5], and, by others, *planar robot.*

This study is suitable as an extended project in a multi-variable and vector calculus class that has access to standard computing technology. Its character as an inherently inverse question with many possible solutions makes it particularly suitable for student driven explorations, while teaching line integrals (and controlled dynamical systems).

As basic model we consider a chain of three planar rigid bodies (*"links"*) that are connected by actuated rotary joints. Think of the ted Swiss-army-knife-like cartoon in figure 4. Or, in more practical terms, think of sitting on a barstool, feet off the ground, desiring to rotate without touching anything. The central red rigid body represents the torso (with head and legs that are not moving with respect to each other). Connected to this at the shoulders are the arms that represent the outside rigid bodies (yellow and blue). Due to conservation of total angular momentum, when one arm is moved, the rest of the assembly will move in the opposite direction. The goal is to design rotary (sideways, forward, backwards) motions at the shoulders (assumed to move the arms in a horizontal plane) that come back to the initial configuration, yet leave the whole assembly (body) rotated.

One of the first tasks is to make dancing choreographies into curves in the (θ_1, θ_2)-plane (or on the torus) of the angles at the shoulders (joints). E.f student were asked to stand up and move their arms according to the curve tracing out the edges of the square the square $(0,0) \longrightarrow (\frac{\pi}{2}, 0) \longrightarrow (\frac{\pi}{2}, \frac{\pi}{2}) \longrightarrow (0, \frac{\pi}{2}) \longrightarrow (0,0)$. This is a very different motion than the one defined by the diamond $(0,0) \longrightarrow (\frac{\pi}{4}, \frac{\pi}{4}) \longrightarrow (0, \frac{\pi}{2}) \longrightarrow (-\frac{\pi}{4}, \frac{\pi}{4}) \longrightarrow (0,0)$. Conversely, some student performs a dance and the others draw the curve in the (θ_1, θ_2)-plane, and then describe is in algebraic symbols as a piecewise continuous curve. We want to think of this curve (or motion) as the *control input* which shall reorient the whole assembly in a desired way.

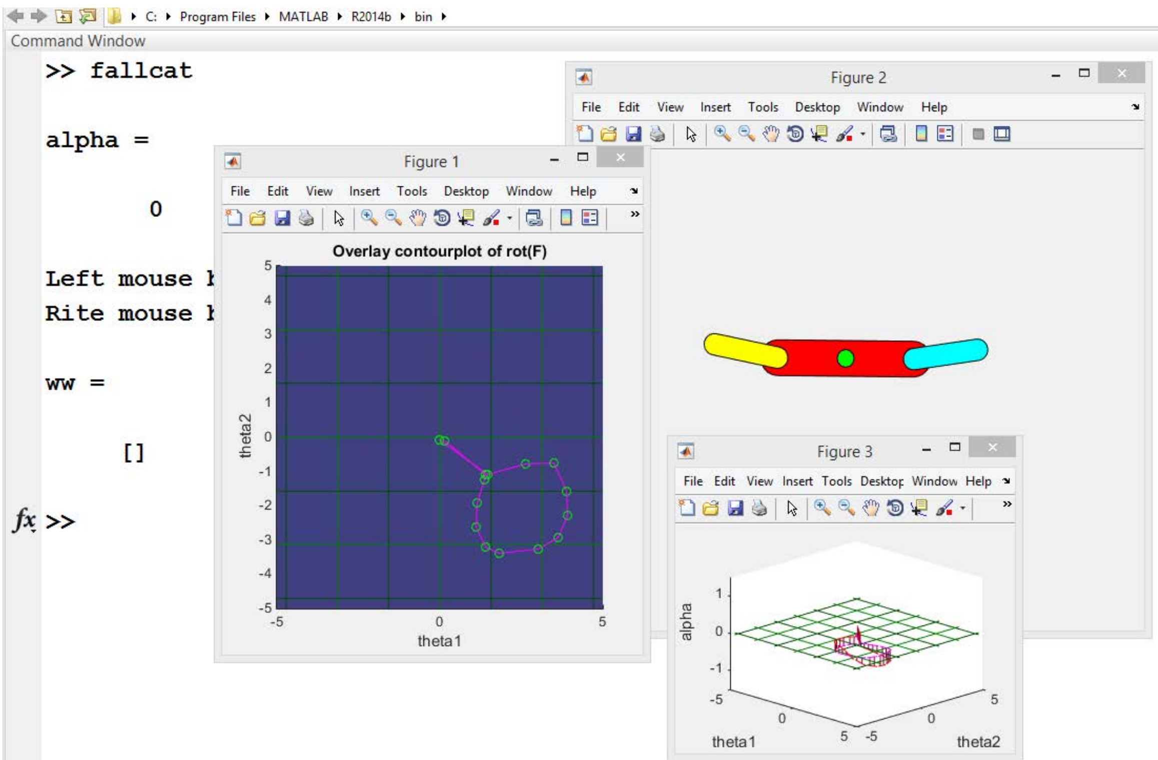


Figure 6: Interactive MATLAB animation of a planar robot.

We here focus on the model where the center of the middle link is at a fixed location (like a barstool), and the assembly can freely rotate about this pivot. More resembling the original cat problem is an analogous model that floats freely in the plane, and one considers the rotation about its center of mass which, of course, continuously moves as the shape (angles at the shoulders) changes.

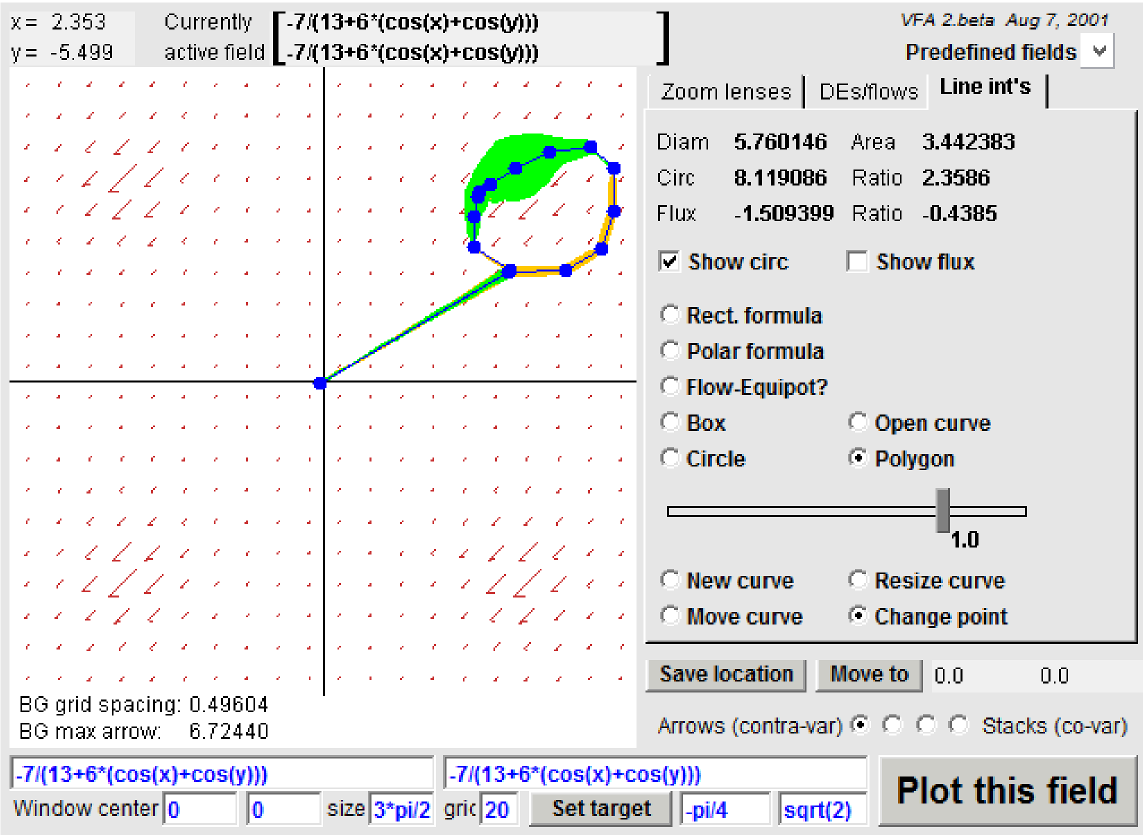


Figure 7: Optimizing the polygonal contour one point at a time

To make a formal mathematical model, let m_i and r_i denote the masses and half lengths of the links ($i = 0$ the central link, and $i = 1, 2$ the outside links). The shape and configuration are determined by the angles θ_i, $i = 1, 2$ at the shoulders, (with $\theta_i = 0$ corresponding to fully outward extended arms), and the angle α of the central link with respect to a reference direction. The angular momenta of the outside links rotating about the hinges (shoulders) are $M_i = \beta_i m_i (2r_i)^2 \frac{d\theta_i}{dt}$ where e.g. $\beta = \frac{4}{3}$ for the case of a thin rod. To obtain the total angular momentum, add to the sum of these the angular momenta of the middle link and the outside links as they rotate about the center of the middle link. These momenta are $M_0 = \beta_0 m_0 r_0^2 \frac{d\alpha}{dt}$ with $\beta_0 = \frac{1}{3}$ for the case of a thin rod, and $M_i = \beta_i m_i r_{i0}^2 \frac{d\alpha}{dt}$ where the distances r_{i0} of the center of mass of the outside links from the pivot depend on the shoulder angles $r_{i0}^2 = r_0^2 + r_i^2 + 2r_0 r_i cos\theta_i$. The sum of all these 5 expressions is constant. We may solve this constraint for $\frac{d\alpha}{dt}$ as a linear combination of the shoulder angular velocities $\frac{d\theta_i}{dt}$

$$\frac{d\alpha}{dt} = g_1(\theta_1, \theta_2)\frac{d\theta_1}{dt} + g_2(\theta_1, \theta_2)\frac{d\theta_1}{dt}$$

with

$$g_i(\theta_1, \theta_2) = -\frac{1 + \beta_i m_i r_i^2}{\beta_0 m_0 r_0^2 + m_1(r_0^2 + r_1^2 + 2r_0 r_1 \cos\theta_1) + m_2(r_0^2 + r_2^2 + 2r_0 r_2 \cos\theta_2)}$$

Now let $\theta\colon [0,T] \mapsto (\theta_1(t), \theta_2(t))$ be a piecewise smooth curve modeling the angles at the shoulders as functions of time. Then the corresponding change in overall orientation $\Delta\alpha = \alpha(T) - \alpha(0)$ is the line integral (which is independent of the chosen time parameterization)

$$\Delta\alpha = \int_C \underbrace{g_1(\theta_1,\theta_2)d\theta_1 + g_2(\theta_1,\theta_2)d\theta_2}_{=\omega}.$$

Evaluating the exterior derivative (or scalar curl) (a computer algebra system comes in handy)

$$d\omega = \left(\frac{\partial g_2(\theta_1,\theta_2)}{\partial\theta_1} - \frac{\partial g_1(\theta_1,\theta_2)}{\partial\theta_2}\right) d\theta_1\, d\theta_2$$

establishes that $d\omega \neq 0$ and therefore ω is not exact. Consequently, there is no *potential function* $\alpha(\theta_1,\theta_2)$ that is invariant under the described motions. This is the reason why the cat can rotate, or the planar robot, or the student on a bar-stool can rotate into any desired position (while returning to the initial shoulder angles).

A natural next try is to draw various curves using suitable computing technology and evaluate the corresponding line integrals. Our JAVA Vector Field Analyzer II [4] (still running well in Firefox) is well suited for this task, see figure 4. A screenshot of an interactive MATLAB inplementation is shown in figure 4. Not unexpected, for most closed test curves (without loss of generality starting and ending at $(\theta_1,\theta_2) = (0,0)$), the resulting change $|\Delta\alpha|$ is disappointingly small. When we worked this as an extended project, the students were asked to provide an algorithm to quickly generate a suitable closed curve when handed initial and final angles.

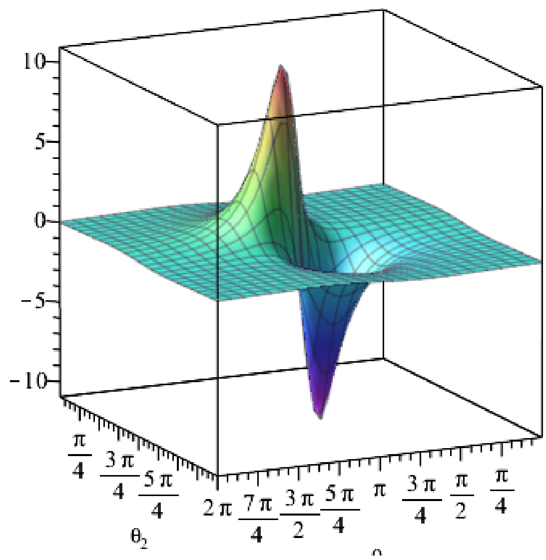


Figure 8: Optimal loop goes around the peak but misses the pit.

The key to finding highly effective curves is to use Green's theorem which states that $\Delta\alpha = \iint_R d\omega$ where R is the region *"inside"* the closed curve C. A computer generated plot of the scalar curl $d\alpha$ reveals the mystery: To *catch* the most volume, the curve should loop around the peak, but miss the deep pit next to it. Some of the creative solutions proposed by our students included one parameter families of curves that look like lollipops, with the stick starting and ending at e.g. $(\theta_1, \theta_2) = (0, 0)$ and not contributing anything (since it is traversed twice in opposite directions), similar to figure 4. The circles looping around the peak but staying clear of the pit, are parameterized by their radii. Given any $\Delta\alpha$, it is very fast to numerically find a matching value for the radius.

5 Summary and conclusion

We have argued that some of the most effective uses of computing technology for learning mathematics involves inverse questions, which in turn nicely match a desired inquiry based learning environment. Rather than requiring a specific platform or software, we found it not too hard to basically use whatever is readily available, what the students already know, or find in the play store. As this generation of students routinely finds and installs all sorts of new *apps* on their smartphones we expect that in the future it will be common to just find and use ad-hoc apps that suit different purposes – as illustrated in this article.

The spreadsheets, computer algebra worksheets, MATLAB code, the JAVA Vector Field Analyzer II, as presented in the conference talk, and used for the screen shots in this article, will be maintained at and kept freely available at the author's website `https://math.asu.edu/~kawski`.

References

[1] Academy of Inquiry Based Learning. `http://www.inquirybasedlearning.org/about/`

[2] Cushing, J. M., Gay, D., Grove L., Lomen, D., and Lovelock D., "The Arizona experience: software development & use", *Proc. International Conference on Techonology in Collegiate Mathematics*, 1991.

[3] Enos, M., "Dynamics and control of mechanical systems. The falling cat and related problems." *Fields Institute Communications*, American Mathematical Society, 1993.

[4] Kawski, M., "Vector Field Analyzer II", `https://math.la.asu.edu/~kawski/vfa2/vfa2sample.html`.

[5] Marsden, J., "Geometric Foundations of Motion and Control", in "Motion, Control, and Geometry", Proceedings of a Symposium, National Research Council, National Academy Press Washington, D.C., 1997. `https://www.nap.edu/read/5772/chapter/3#9`

[6] Montgomery, R., "Gauge Theory of the Falling Cat", in "Dynamics and Control of Mechanical Systems", *Fields Institute Communications*, American Mathematical Society, 1993, pp. 193218.

[7] National Association of Math Circles Wiki. `https://www.mathcircles.org/Wiki_WhatIsAMathCircle`

[8] "Shaping The Future: New Expectations for Undergraduate Education in Science, Mathematics, Engineering, and Technology", *National Science Foundation*, 1996. `http://www.nsf.gov/publications/pub_summ.jsp?ods_key=nsf96139`.