Teaching function approximation via structured discovery and the ApproxTool applet

Michelle L. Ghrist
michelle.ghrist@usafa.edu
United States Air Force Academy
USAF Academy, CO 80840

James S. Rolf
jim.rolf@yale.edu
Yale University
New Haven, CT 06520

Abstract

We have developed a Java applet and accompanying exercises to provide a discovery-based experience to help students better understand function approximation. We have successfully used this combination in an upper-level numerical analysis course for several years. In this paper, we introduce the applet and structured exercises and offer observations of student engagement via use of these tools.

1. Introduction

In 2005, we set out to develop technological and pedagogical tools to help students better understand function approximation. Both of us have backgrounds in numerical analysis as well as the scholarship of teaching and learning mathematics (see, e.g., [5][6][9][10]). Jim has considerable experience creating educational Java applets [5][9], and Michelle was preparing to teach the introductory numerical analysis course at the Air Force Academy for the third time. The result of this collaboration was a Java applet, *ApproxTool*[1], and accompanying exercises. This combination of applet and structured exercises is designed to give students a discovery experience of function approximation that can occur inside and/or outside the classroom.

When we started this project, we recognized the utility of other technological tools; we use MATLAB and Mathematica extensively throughout this course. For this activity, we wanted a framework that would support the kind of engaged exploration necessary for a discovery experience. In particular, we wanted a simple interface requiring little syntax knowledge in order to avoid distractions and cognitive overload that might disrupt engagement [1]. Interactive visualizations with minimal guidance are known to have learning benefits [3][8], and we believed that intuitive controls such as mouse drags, clicks, etc. would best elicit this engagement. Therefore we believed that a Java applet such as *ApproxTool* would be more likely to promote the kind of engaged exploration that would drive intellectual curiosity.

Three instructors have now used the *ApproxTool* applet and accompanying exercises a total of at least six times in the introductory numerical analysis course at the Air Force Academy. This paper

[1] Freely available at http://www.jimrolf.com/java/approxTool/approxTool.html

will focus on the observations done by Michelle Ghrist during four semesters of utilizing the applet and discovery exercises. In a typical semester, approximately fifteen upper-level undergraduate students take this course. The course emphasizes conceptual reasoning, analysis of numerical methods, programming, and applications; the course assessments typically include written and oral exams, homework, reading exercises, and one or two writing assignments and/or projects.

The applet and accompanying exercises have evolved through the years as we refine them and experiment with different methods of implementation in the classroom. We have found that the combination of the applet and the guided discovery exercises is valuable in helping students develop a well-rounded understanding of function approximation. In this paper, we discuss the *ApproxTool* applet, our accompanying guided discovery assignments, and our perceived impact on student learning.

2. Our Goals

We had several goals when we developed the applet and exercises. First, we wanted to provide a hands-on laboratory-type experience that requires few mathematical calculations and little (or no) programming. In particular, we wanted to create an environment where students can quickly change the underlying original function and various parameters and then quickly visualize the various approximations and their errors. We expected that the simplicity of the user interface combined with some meaningful questions would help students discover rich conceptual ideas and encourage them to continue exploring.

Second, we expected that the visual nature of these explorations will help students better understand the analysis and error formulas (and associated theorems) in the textbook, resulting in a deeper understanding of the material (e.g., error, polynomial interpolation, splines, and least squares approximations). We also anticipated that these tools will encourage students to explore other ideas which were only briefly discussed in class, thus extending the learning experience outside of the classroom. Some of this additional material includes function norms, extrapolation, minimax approximation, and the spacing of the roots of Chebyshev and Legendre polynomials.

Lastly, we wanted to provide students with an opportunity to practice and improve their written technical communication skills. While this is a desired professional skill for all students in today's technology-heavy world, we also believe that students who work to clearly communicate technical ideas develop a better understanding of that material and better retain that material [7].

3. The *ApproxTool* Applet

The software that we developed, *ApproxTool*, is a Java applet which is designed to help students quickly visualize various function approximation techniques in a user-friendly way. In its current form, the applet can show graphical representations of various interpolants (polynomials and cubic spline) and approximations (least squares and Taylor series) of a user-defined function.

The applet allows a user to enter a function, an interval on which to approximate, and the number of points to use for approximation. For example, Figure 1 shows a graph of the default function (in red) on the interval $[-3, 3]$. Five equally spaced nodes on $[-2, 2]$ are shown as red dots on the graph. The user can change each of these parameters via the boxes in the upper right-hand corner and the labels on the graphs.

In order to approximate the underlying function on the interval, the user can select any subset of the following approximation techniques:

- Polynomial interpolation, shown in yellow in Figure 1 (uses the nodes)
- Cubic splines interpolation (uses the nodes)
- Least squares polynomial approximation via regression (uses the nodes)
- Taylor series approximation (uses a centering point and order *n* instead of the nodes)
- Chebyshev approximation (uses the approximation interval and order *n* instead of the nodes), with two different implementations (to be further discussed)
- Legendre approximation (uses the interval and order *n* instead of the nodes), with two different implementations (to be further discussed)

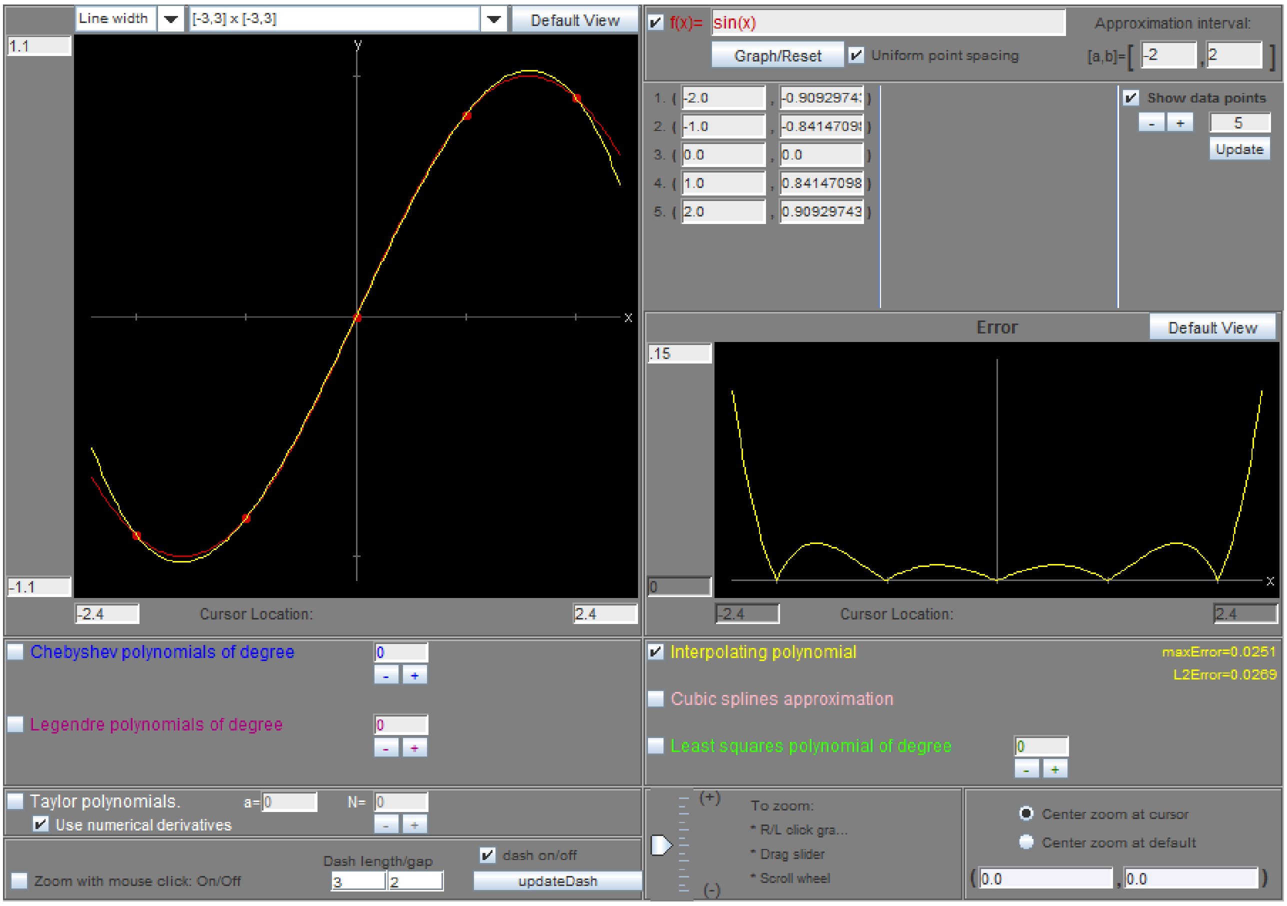

Figure 1: *ApproxTool applet. The left graph shows the user-defined function in red, any approximations (in various colors), and the approximation points used (red dots). The right graph shows the error(s) of the approximation(s) (in the same color as the approximations). The max norm and Euclidean norm of the error of each active approximation are also given.*

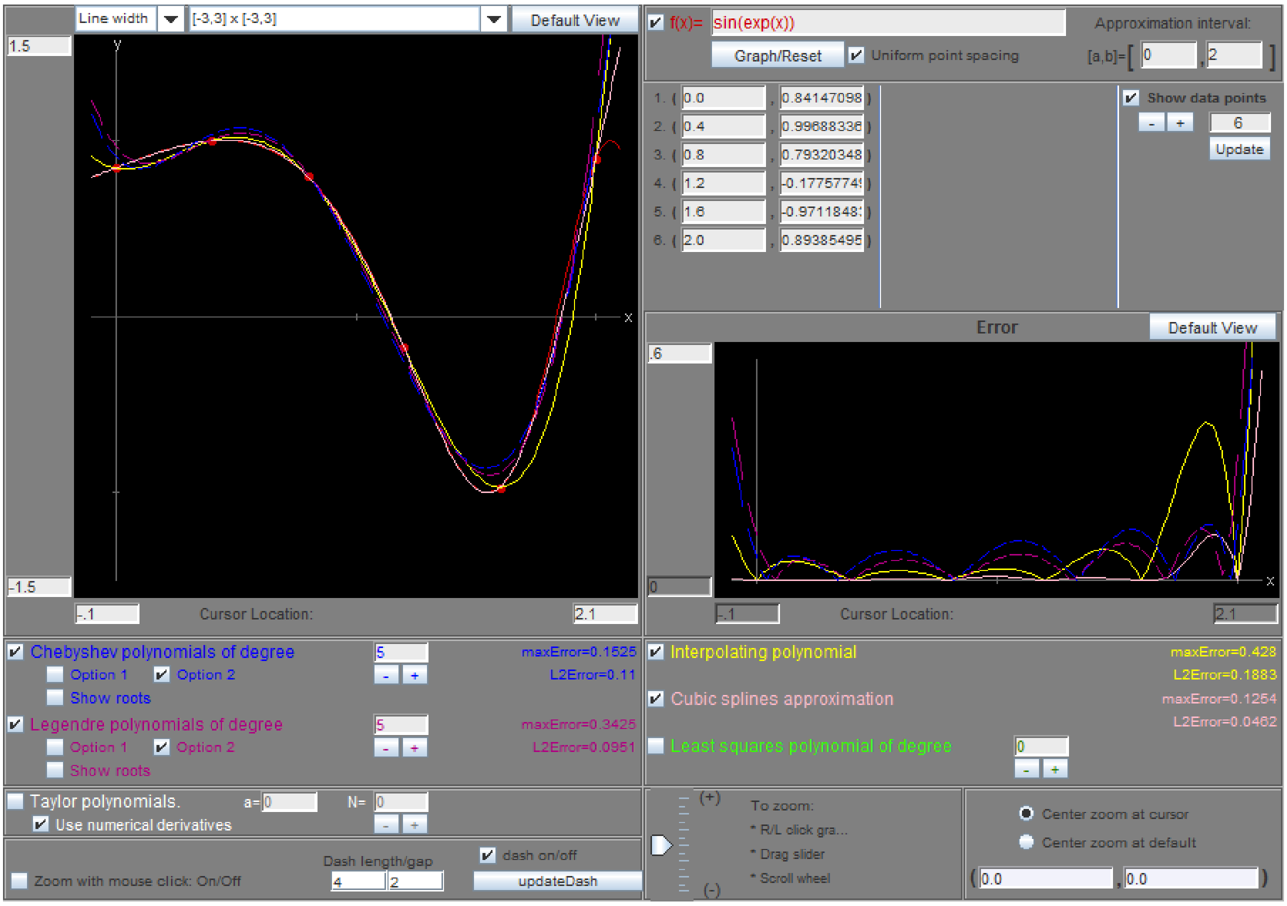

Figure **2**: *Illustrating the use of the ApproxTool applet. Four different approximations (including three 5th order polynomial approximations) and their corresponding errors are shown. We expect that, in this case, students would observe that equispaced polynomial interpolation is the worst and splines are the best (this result is dependent on the function and interval used). We would also like them to notice that among the polynomial interpolants, the Chebyshev approximation minimizes the L_∞ (max) norm and the Legendre approximation minimizes the L_2 norm of the error [2][4].*

We also included several other pedagogically useful features:

- *Estimates of error.* A graph of the absolute value of the pointwise error of each approximation is provided along with two global estimates of each error: the L_∞ (max) and L_2 (Euclidean) norms of the error of each approximation. This feature is instructive for comparing the various approximations, especially comparing equispaced polynomial interpolation to Chebyshev and Legendre approximation (see Figure 2) [4]. It also allows for exploration of the concept of function norms.

- *Changing the number of nodes*. Additional data points can be added to the graph by changing the number of points desired in the upper right-hand corner or by clicking anywhere on the graph. The user can require that the nodes be equispaced or relax that condition via the check box. Alternatively, one can change the node values directly by typing specific values into the list in the upper right-hand corner; the applet automatically updates the *y*-values using the original function (which the user can subsequently overwrite if desired). These features are useful for helping students explore the effects of adding more nodes and changing the spacing of the nodes. The applet allows between 2 and 16 data points to be used. The user has the option to not display the nodes (if, for example, the user is only using Legendre, Chebyshev, and/or Taylor approximation).

- *Changing nodes*. After graphing, the user may then drag any of the data points around the graph (either landing on the function or not) and immediately see the resulting changes in the function approximations and errors. Alternatively, the user can directly change a *y*-value in the list of nodes. This feature lends itself well to studying the effect of data value error on the various approximations and to exploring whether introducing data error has more of a local or a global effect on the approximation.

- *Chebyshev and Legendre approximations.* Chebyshev interpolation of order *n* can be done in two different ways:
    - Orthogonal function expansion using the Chebyshev polynomials of the first kind up to order *n* as a basis with the appropriate weight function[2]
    - Polynomial interpolation using the roots of the $(n+1)^{st}$-order Chebyshev polynomial as nodes.

  In general, these do not give the same approximations; see Chapters 10 and 12 of [2] or Section 5.4 of [4], for example. One can also perform Legendre approximation similarly using either approach (with the weight function $w(x) = 1$). Previously, the *ApproxTool* applet implemented the first option for Chebyshev approximation and the second option for Legendre approximation; one of our more challenging exercises for students was to deduce which method was actually implemented for Chebyshev and Legendre approximation. A recent improvement to *ApproxTool* allows the user to employ either (or both) options for both Chebyshev and Legendre approximation, but we have purposely left it as a mystery for the user to deduce which is which; this can be used as a pedagogical tool by the instructor (see question 6 in Section I of the exercises provided in the Appendix).

- *Visualization of nodes*. Options to show the nodes for the second technique of Chebyshev and/or Legendre approximation are given; these also show the roots of the $(n+1)^{st}$-order Chebyshev (Legendre) polynomial. This can help students better understand the spacing of the roots of the Chebyshev and Legendre polynomials as well as the connection between these approximation and the Chebyshev and Legendre polynomials. It also allows them to compare and explore the resulting errors in these four polynomial approximations.

---

[2] On $[-1, 1]$, this is done via $f(x) \approx \sum_{k=0}^{n} c_k \varphi_k(x)$, where $\varphi_k(x)$ is the Chebyshev polynomial of order k, $c_k = \frac{\int_{-1}^{1} f(x)\varphi_k(x)w(x)dx}{\int_{-1}^{1} \varphi_k{}^2(x)w(x)dx}$, and $w(x) = \frac{1}{\sqrt{1-x^2}}$. A linear change of coordinates is done if the interval is other than [-1,1]. See [2] or [4] for details.

4. Structured Exercises

To guide students in their discovery process, we have created an assignment that is part laboratory experience (exploration and experimentation) and part writing (reflecting and interpreting). Throughout the lab time, we want students to look for patterns and connections and to frame their observations in the context of the bigger picture of function approximation. To encourage students to delve deeper and connect what they observe with the course and textbook material, we encourage them to ask "Why might this be true?" throughout the assignment. To further encourage deeper thought and formation of connections, we request that students write their final observations as more of an essay than a traditional lab report (when a writing assignment is required).
Our discovery exercises (listed in their entirety in the Appendix) have four main parts:

(1) In-depth exploration of polynomial interpolation,
(2) In-depth exploration of cubic spline interpolation,
(3) Comparison of least squares and Taylor series methods with interpolation methods, and
(4) Broader questions designed to encourage students to delve deeper.

Each student is instructed to select two non-polynomial functions to use throughout the assignment, one function with symmetry (even or odd) and one without symmetry.

The first two parts of the assignment consist of various experiments and questions about interpolation via polynomials and cubic splines. We want students to explore the impacts of data error and of adding more nodes, so we ask students to use *ApproxTool* and answer the following:

> *Explore the effect of introducing a small amount of error into your data values at a given node; does moving a data point (i.e., introducing a small amount of error) or changing a node location (with no error) have a global or local effect on the approximation? Explore the effect of adding more nodes on the original function; explore putting these points at different locations, e.g., clustered together, spread out, in several clusters, etc. What do you observe?*

Typically, most students correctly observe that the location and spacing of the nodes make a big difference in the error of the interpolant and that data value errors have only local effects on splines but global effects on polynomial interpolants.

We also want students to explore the Runge phenomenon, where equispaced polynomial interpolants of some functions have oscillations of greater magnitude as one increases the order of the approximation; we typically explore this phenomenon in MATLAB prior to this lab. We ask:

> *If you continue to increase the order of your polynomial approximation, do you observe the Runge phenomenon? If you do, at what point (order-wise) does this become significant? If you don't, why do you suppose not?*

The ability to use *ApproxTool* to quickly change the order of interpolating polynomials makes it easy to ask students to extend the in-class exploration and discussion of the Runge phenomenon. Most students don't observe the Runge phenomenon because they pick "nice" functions like e^x and $\cos(x)$. This result typically leads to a rich class discussion after students submit their assignments as they believe that the Runge phenomenon should be more widespread. This kind of discovery and subsequent discussion makes the concept of stability more tangible to the students in the context of function approximation.

One of the more difficult questions with which students grapple on this assignment is:

> *The cubic spline created by the applet is one of the four we have discussed (cubic Hermite, not-a-knot, clamped, or natural). Use your best detective skills to hunt down which one is used. How can you tell?*

This requires students to understand not only the formal requirements for each spline (and their error formulas) but also how these translate to the graphical representation of the splines; students can also rely on their previous experience constructing splines in MATLAB. The synthesis of these ideas combined with a necessary process of elimination proves to be challenging for most students.

The third part of the assignment asks students to explore least squares and Taylor series approximation and then compare and contrast these approximations with each other and with the interpolation methods already discussed. For least squares approximation, we want students to discover several key concepts: in general (but not always), polynomial interpolation is better than least squares regression (unless there is data error), and when the order of the least squares approximation gets high enough, the least squares approximation is identical to polynomial interpolation. We ask:

> *Compare least squares approximation to polynomial interpolation. When does one perform better than the other (and what does it mean to perform better)? What if there is error in the nodes? When are they reasonably indistinguishable? When are they reasonably indistinguishable?*

Most students are able to ascertain the critical ideas and relationships between these approximations.

Regarding Taylor series approximation, we ask students to utilize *ApproxTool* to answer the following questions:

> *How does Taylor series approximation compare with the other methods of approximation? When is it better (and what does it mean to perform better)? When is it worse? Do you notice any patterns to the error of Taylor polynomial approximation as compared to the errors of the various polynomial interpolations? What difference does changing the center a have on the approximation and error graph?*

Most students are able to determine that the error is typically only zero at one point for Taylor series approximation while it is zero at each node for interpolation; they can usually articulate that the interpolation methods attempt to spread the error throughout the approximation interval while Taylor series focus on making the error much smaller close to the one centering point. This visual exploration provides students some insight into the structure of the analytical error formulas of polynomial interpolation and Taylor approximation; as a consequence, students are typically more willing to grapple with error analysis later in the course.

The fourth part of the assignment gives students a chance to delve deeper in areas of their choice. Students are required to pick at least three out of eight questions to explore and discuss at length. These questions include addressing the effects of symmetry of the original function, comparing extrapolation to interpolation for the various approximants, exploring the concept of minimax approximation, addressing the minimization of certain error norms for polynomial interpolation (see Figure 2), and examining the spacing of the roots of the Legendre and Chebyshev polynomials.

An example that illustrates our goals for this assignment stems from the following questions:

> *What is the general shape of the error graph? Why does it look that way? Compare the errors of the approximations for extrapolation. Which method(s) seems to do the "best" for extrapolation? Is this related to how well they perform for interpolation?*

Students would first graph their chosen function in *ApproxTool* and observe something similar to the function, interpolant, and yellow error graph shown in Figure 1. Several features should stand out to the student: (1) the error is zero at five locations (which appear to be at the nodes), (2) the error is smallest towards the middle of the approximation interval, (3) the error humps are somewhat distorted parabolas, and (4) the error grows quickly outside the approximation interval. Students should already be familiar with the standard error formula for polynomial interpolation

$$E(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n} (x - x_i),$$

where $x_0, x_1, \ldots, x_n$ are the nodes and $\xi(x)$ is some point in the smallest interval containing the nodes and *x*. This is a daunting formula to many students. However, the visual cues provided by *ApproxTool* give some insight to the structure of this error formula; for example, an error function that is equal to zero at the nodes can be modeled by the product of *n* linear expressions. And the reverse is true as well—the structure of the error formula helps students answer why the error seems to be smallest towards the middle of the interval, why the error humps are not parabolic, and why extrapolation outside the interval can lead to large errors. In general, we find that this combination of visual discovery and subsequent mathematical analysis leads to synthesis and deeper understanding, especially when utilized by an experienced instructor who can help guide students towards these discoveries.

5. Observations

On the day that students submit their writing assignments, the in-class discussion proves to be quite enlightening. (Note: during semesters when a formal writing assignment is not required, the instructor facilitates these discussions during or after the lab time.) The instructor is able to capture various insights from the students and use this to fill in gaps in student knowledge. For example, most students do not observe the Runge phenomenon (for various reasons) while others do not consider the impact of symmetry of the function. Some students are typically still confused concerning the impact of data noise on local versus global error. Because of the time that students spend exploring, writing, and thinking deeply, the in-class discussion provides an ideal environment for students to reconcile their various observations and for the instructor to highlight the key ideas. In the absence of this kind of assignment, these discussions might be closer to a monologue by the instructor with students taking notes with the distant hope that they might understand the nuances of the material at some later time.

When using the applet, the instructor has observed that students are much more willing to spend time experimenting. Changing functions and values for parameters and properties is simple and requires little knowledge of syntax which would be required by a computer-algebra system (CAS) such as Mathematica. Thus students are willing to consider more possibilities than would have been reasonable when using a syntax-laden computer algebra system or a programming environment. (Note: we did also teach students to leverage the function approximation tools in MATLAB independently, including use of the Curve Fitting and Spline Toolboxes.)

The assignment was rather time-intensive. During the first two offerings (before this course had an established separate lab time), about one hour was devoted to experimentation during class time; the students who performed well on the writing assignment typically invested another 2-4 hours in experiment. For the two latter offerings, about 3 hours was devoted during lab time. During semesters in which a formal writing assignment was required, most students spent about three hours writing, although some spent considerably more. The instructor noted that the students who invested time in exploring and learning displayed a deeper knowledge of function approximations on the relevant questions on the subsequent oral exam. We find that oral exams allow the instructor to quickly probe the limits of student knowledge and conceptual understanding, and we believe that the writing assignment stressed a somewhat different approach to learning and understanding than most other assessment tools used in the course.

To more formally examine evidence of these observations, we studied student data from three different groups, each containing two years of data from 2004-2011. Each group had the same instructor and similar assessments.

- Group 1 (35 students) did not use the applet or exercises; they learned function approximation primarily through their textbook and interactive lecture. They completed a project instead of the writing assignment which was more applied and less conceptual.
- Group 2 (23 students) used the applet and exercises (in addition to the textbook and lecture); they also submitted a writing assignment concerning their discoveries.
- Group 3 (21 students) used the applet and exercises as part of two in-class labs (in addition to the textbook and lecture) but were not required to complete a writing assignment; they completed an applied project instead of the writing assignment.

Based on an analysis of grades in previous math classes, Group 1 and 2 had fairly similar skills entering this course, while Group 3 was expected to perform about 5% higher than Groups 1 and 2.

The means on various assessments in this course are given in Table 1; we see that the overall performance of Group 2 students relative to Groups 1 and 3 was about what was expected overall, but the Group 2 students performed relatively worse on written exams and better on assessments for which they could prepare such as homework and class preparation.

Table 1: *Means of student scores on course assessments (%) by group.*

	Oral exams	Homework	Class Prep	Written Exams	Project (1,3) or Writing Assignment (2)	Final Exam	Final Score
Group 1	87.91	83.49	82.23	77.16	90.05	75.59	81.43
Group 2	87.52	86.44	85.15	74.26	86.46	69.06	81.03
Group 3	92.83	90.57	90.85	85.48	92.12	88.82	90.70

Table 2 shows a correlation matrix for the grades of students in Group 2. For this group, the writing assignment shows weaker correlations with the other assessments than other assignments. The writing assignment seems to be more highly correlated with the oral exams (which students can prepare for) and class preparation (i.e., reading questions); this aligns with our informal observations that one of most important factors on the writing assignment was effort and diligence.

Table 2: *Correlation matrix for grades of students in Group 2. These students utilized the applet and exercises and then completed a formal writing assignment.*

	Oral exams	Writing assignment	Homework	Class Prep	Written exams	Final exam
Oral exams	1.0000	0.3566	0.4545	0.5742	0.5907	0.6049
Writing assignment	0.3566	1.0000	0.2460	0.3463	0.2536	0.0970
Homework	0.4545	0.2460	1.0000	0.6804	0.8055	0.6863
Class Prep	0.5742	0.3463	0.6804	1.0000	0.6023	0.4145
Written Exams	0.5907	0.2536	0.8055	0.6023	1.0000	0.7978
Final exam	0.6049	0.0970	0.6863	0.4145	0.7978	1.0000

When we examine the correlation matrices for Groups 1 and 3 (see Tables 3 and 4), overall we see that most correlations are higher than the writing assignment was for Group 2. Thus, we believe that the writing assignment (as implemented for Group 2) did test different skills than the other assessments.

Table 3: *Correlation matrix for grades of students in Group 1. These students did not use the applet or exercises.*

	Oral exams	Project	Homework	Class Prep	Written exams	Final exam
Oral exams	1.0000	0.5867	0.4594	0.5759	0.4147	0.4779
Project	0.5867	1.0000	0.5036	0.7579	0.2948	0.4194
Homework	0.4594	0.5036	1.0000	0.5234	0.5778	0.5702
Class Prep	0.5759	0.7579	0.5234	1.0000	0.1397	0.3298
Written Exams	0.4147	0.2948	0.5778	0.1397	1.0000	0.6382
Final exam	0.4779	0.4194	0.5702	0.3298	0.6382	1.0000

Table 4: *Correlation matrix for grades of students in Group 3. These students used the applet and exercises but did not complete a formal writing assignment.*

	Oral exams	Project	Homework	Class Prep	Written exams	Final exam
Oral exams	1.0000	0.7598	0.8918	0.8094	0.8076	0.8225
Project	0.7598	1.0000	0.7007	0.6078	0.7039	0.7525
Homework	0.8918	0.7007	1.0000	0.7257	0.8150	0.8758
Class Prep	0.8094	0.6078	0.7257	1.0000	0.6437	0.6607
Written Exams	0.8076	0.7039	0.8150	0.6437	1.0000	0.9325
Final exam	0.8225	0.7525	0.8758	0.6607	0.9325	1.0000

When examining the correlations between oral exams and the final exam, we see that Groups 2 and 3 have significantly higher correlations than Group 1 (see orange highlighting in Tables 2-4), which may be evidence that the conceptual skills learned from using the *ApproxTool* applet and exercises helped these students better learn how to engage numerical analysis on a deeper level and retain the course material. In general, we observe some similarities across the groups regarding higher correlations of the project (or writing assignment) with assignments which require more preparation and/or effort (e.g., oral exams and class preparation) than with written exams (see yellow highlighting).

Figure 3 shows a scatter plot of Group 2 students' scores on the writing assignment and subsequent oral exam; about half of the oral exam material tested their conceptual skills in function approximation. The distribution is quite interesting; although the data has a general positive trend, there appear to be two distinct groups of students with larger and smaller positive slopes. We conjecture that this difference is also related to the amount of effort put forth by the students on the writing assignment.

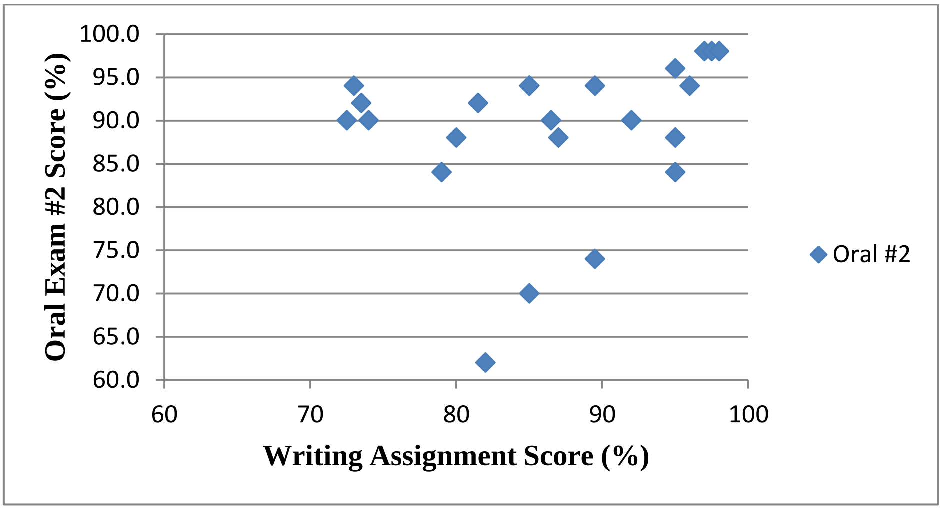

Figure **3**: *Scatter plot of student scores on the second oral exam versus their score on the writing assignment for Group 2. These students used the applet and exercises and then completed a formal writing assignment.*

Informally, several students in Group 2 either took time in class or approached the instructor later to communicate that while the assignment had taken a great deal of time, they had a much deeper level of understanding as a result of the assignment. What was once just a bunch of formulas in the textbook came alive to them in a very real way, and they started to see the need for having many different methods of function approximation.

Finally, we have found that our collaboration has resulted in much better final products for both the applet and guided discovery assignment. The construction of the guided discovery exercises and *ApproxTool* has been (and continues to be) an iterative process; some questions in the structured exercises were generated as a result of features in *ApproxTool*, and some features in *ApproxTool* were added due to the desire to ask certain questions. While some features in *ApproxTool* did not initially work as advertised, our collaborative process allowed us to identify and correct most of these bugs before students did. We believe that our work together has allowed us to make the applet more user-friendly and to include features with lasting educational impact.

6. Conclusion

The *ApproxTool* applet and accompanying activities can help create an exceptional structured environment to help students explore function approximation in a lab-like setting, especially when used in conjunction with other programs such as MATLAB or *Mathematica*. We believe that this experience helped our students develop a deeper conceptual understanding of the topics, reinforced critical ideas, illuminated mathematical analysis, and drove rich conversations in the classroom. Most students enjoyed the discovery-based learning and were willing to invest more time than for a standard assignment. While the assignment required a significant time investment on the part of the students and instructors, we had excellent returns on this investment. We believe that the use of the *ApproxTool* applet and guided discovery exercises in a numerical analysis course allows students to develop a richer concept map of interpolation and function approximation, especially if the students are willing to invest the necessary time and effort.

7. References

[1] Adams, W.K., Reid, S., LeMaster, R., McKagan, S.B., Perkins, K.K. Dubson, M., and Weiman, C.E. (2008). A study of educational simulations: part I - engagement and learning. *Journal of Interactive Learning Research*, 19(3), 397-419.

[2] Ascher, U. and Greif, C. *A First Course in Numerical Methods*, SIAM, 2011.

[3] Bodemer, D., Ploetzner, R., Feuerlein, I., and Spada, H. (2004). The active integration of information during learning with dynamic and interactive visualizations, *Learning and Instruction*, 14(3), 325-341.

[4] Bradie, B. *A Friendly Introduction to Numerical Analysis*, Pearson Prentice Hall, 2006.

[5] Brilleslyper, M.A., Dorff, M.J., McDougall, J.M., Rolf, J.S., Schaubroeck, L.E., Stankewitz, R.L., and Stephenson, K. *Explorations in Complex Analysis*, Mathematical Association of America, 2012.

[6] Brilleslyper, M., Ghrist, M., Holcomb, T., Schaubroeck, B., Warner, B., and Williams, S. (2012) What's the point? The benefits of grading without points, *PRIMUS*, 22(5), 411-427.

[7] Gent, I., Johnston, B., and Prosser, P. (1999). Thinking on your feet in undergraduate computer science: a constructivist approach to developing and assessing critical thinking. *Teaching in Higher Education*, 4(4), 511-522.

[8] Podolefsky, N.S., Perkins, K.K., and Adams, W.K. (2010). Factors promoting engaged exploration with computer simulations, *Physical Review Special Topics: Physics Education Research*, 6(2), 020117:1-11.

[9] Rolf, J.S. and Schaubroeck, L. (2009). Interactive complex analysis: supplementing the traditional course with computer visualization, *The Electronic Journal of Mathematics and Technology*, 3(2), 176-185.

[10] Scharff, L., Rolf, J., Novotny, S. and Lee, R. (2011). Factors impacting completion of pre-class assignments (JiTT) in Physics, Math, and Behavioural Sciences. In Rust, C. (Ed.), *Improving Student Learning: Improving Student Learning Global Theories and Local Practices: Institutional, Disciplinary and Cultural Variations*, Oxford Brookes University, United Kingdom.

Appendix: Assignment Activities

In this appendix, we give the actual discovery questions and assignment given to students. The first paragraph was only used during the semesters in which we assigned a writing assignment.

Concerning your final report, keep in mind that this is a writing assignment; your final product should read much more like an essay than a lab report. You should write a coherent paper that frames what you have done in light of the bigger picture of function approximation. Your paper should reflect what you have learned through this exercise. In general, you should look for patterns and try to connect your observations with what you have learned in class and from the textbook.

You have two main goals:

1. To explore the effect of changing various parameters on the errors of both polynomial interpolation and cubic spline interpolation, and
2. To compare the error in function approximation via polynomial interpolation and cubic spline approximation to the error from other types of function approximation.

You should pick two different non-polynomial functions to approximate:

(a) one function that is either even or odd
(b) one function that is neither even nor odd.

These functions should be chosen independently of other students in the course. You should perform all tasks for both functions, comparing your results.

I. Interpolating Polynomials

Perform the following tasks for polynomial interpolation, keeping an eye on the error graph and error norms:

1. What is the general shape of the error graph? Why does it look that way?

2. Explore the effect of introducing a small amount of error into your data values at a given node; do this via changing a *y*-value directly in one of the data values boxes. (You can also move a node away from the original function via depressing the mouse on a red data point and dragging this point to the desired location.) Does moving a data point (i.e., introducing a small amount of error) or changing a node location (with no error) have a global or local effect on the approximation, i.e., does the entire approximation change significantly, or does only a piece of the approximation change?

3. Explore the effect of adding more nodes on the original function. Do this in two ways:
    a. First, explore adding more equispaced nodes by changing the number of points used under “Show data points” and hitting the “Graph/Reset” button.
    b. Second, add more (not equispaced) nodes by typing in *x*-values directly in the data values boxes (after deselecting the “Uniform grid spacing” box). Explore putting these points at different locations, e.g., clustered together, spread out, in several clusters, etc. What do you observe?
    c. What happens if you try to put two nodes on top of each other? Why?

4. If you continue to increase the order of your polynomial approximation, do you observe the Runge phenomenon? If you do, at what point (order-wise) does this become significant? If you don't, why do you suppose not?

5. There are two different (commonly used) ways to do an *n*th-order Chebyshev (or Legendre) approximation on $[-1, 1]$.

   - Polynomial interpolation is done using the roots of the (n+1)st-order Chebyshev (or Legendre) polynomial as the nodes.
   - Create the *n*th-order orthogonal polynomial approximation: $f(x) \approx \sum_{k=0}^{n} c_k \varphi_k(x)$, where $\varphi_k(x)$ is the Chebyshev (or Legendre) polynomial of order k, $c_k = \frac{\int_{-1}^{1} f(x)\varphi_k(x)w(x)dx}{\int_{-1}^{1} \varphi_k{}^2(x)w(x)dx}$, and $w(x) = \frac{1}{\sqrt{1-x^2}}$ ($w(x) = 1$ for Legendre polynomials). A linear change of coordinates is done if the interval is other than [-1, 1].

   For both Legendre and Chebyshev approximation, the applet has Option 1 and Option 2, each of which employs one of these approaches. Figure out which option corresponds to each of the above approaches. (Hint: for most functions, the two Chebyshev (or Legendre) approximations are quite similar. However, for two certain categories of functions, they are not.)

6. Now, consider the Chebyshev and Legendre approximations. Assuming that you use the same order polynomial approximation, compare the errors of the following five approximations.
   a) Interpolating polynomial with equispaced nodes
   b) Chebyshev polynomial approximation (both options)
   c) Legendre polynomial approximation (both options)

   You may need to consider these in sets of threes (e.g., (a) and (c), (a) and (b), (a) and one options each of (b) and (c)). For each of these polynomials, compare and contrast error graphs and error norms for a given order. Are your conclusions affected by changing the order of the polynomials?

II. <u>Cubic Splines</u>

Next, explore approximation via cubic spline approximation:

1. Repeat tasks #1 and #2 from the polynomial interpolation section above. Is there a Runge phenomenon for cubic splines that occurs as more nodes are added? Speculate on why or why not.

2. The cubic spline created by the applet is one of the four we have discussed (cubic Hermite, not-a-knot, clamped, or natural). Use your best detective skills to hunt down which one is used. How can you tell?

3. Does adding a small amount of error have a global or local effect on the approximation? Does moving a node location (with no error) have a global or local effect on the approximation?

III. Least Squares and Taylor Series

Finally, compare polynomial interpolation and cubic splines approximation with the remaining two methods (least squares regression and Taylor series approximation):

1. Compare approximations of the same order. What similarities and differences do you notice, both in the error graph and the error norms? Test your hypotheses via checking several different orders.

2. Which of the six approximation methods used by the applet are actually interpolation methods? Why are the other(s) not considered interpolation methods? Explain.

3. Does the applet use discrete or continuous least squares approximation? How can you tell?

4. Compare least squares approximation to polynomial interpolation. When does one perform better than the other (and what does it mean to perform better)? What if there is error in the nodes? When are they reasonably indistinguishable?

5. How does Taylor series approximation compare with the other methods of approximation? When is it better (and what does it mean to perform better)? When is it worse? Do you notice any patterns to the error of Taylor polynomial approximation as compared to the error of the various polynomial interpolations? What difference does changing the center *a* have on the approximation and error graph?

IV. Other Interesting Questions to Consider *(pick a minimum of three to address)*

- Compare the errors of the approximations for extrapolation. Which method(s) seems to do the "best" for extrapolation? Is this related to how well they perform for interpolation?

- What effect does the symmetry of the function (or lack thereof) have on the various approximation methods and the associated error graphs?

- What effect does the chosen interval(s) of approximation have on the approximations? What happens if you try different intervals?

- Why might we want to have multiple measurements of the error? What relationships can you deduce between the two different norms? What kind of error graph would lead to one error norm being much larger than the other (and vice versa)?

- Which error norm does the Chebyshev polynomial approximation minimize (for a given order)? What about for the Legendre polynomial approximation? Did you observe this?

- With just the nodes showing (not the approximations themselves), repeatedly increase the order of the Chebyshev approximation. What patterns do you notice concerning the locations and spacing of the nodes, when compared to equispaced nodes? Repeat for the Legendre approximation. Compare and contrast the three node locations.

- Explain the word "minimax" as it connects to the Chebyshev approximation (hint: see your text). Explain how this property appears in the error graphs for the Chebyshev approximation. To which option of the Chebyshev approximations does this term apply?

- For the least squares approximation, is it ever "better" to use a lower-order approximation rather than a higher-order one? Explain.