Experiential Learning Applied to a MATLAB-based Undergraduate Numerical Weather Prediction Course

Thomas A. Guinn*
Embry-Riddle Aeronautical University, Daytona Beach, FL

1. Introduction

Strong computer-based problem solving skills are frequently discussed as being highly desired for both graduate school and industry. To help develop these skills, Embry-Riddle Aeronautical University, Daytona Beach (ERAU-DB) created a new undergraduate numerical weather prediction (NWP) course, which is required for all meteorology majors. The motivation for developing the course was to enhance and build student competency and confidence in scientific computing and problem solving, as applied to NWP, using experiential learning concepts. In addition, the course was designed to provide students a deeper yet more qualitative insight into the complexities of modern operational NWP models.

Variations of the course have been offered three separate times as an independent studies with MATLAB serving as the programming language of choice. In fall 2015, the course was formally offered as a traditional course.

The purpose of this paper is to discuss how some of the experiential learning were applied as well as some of the limitations and lessons learned during the first formal course offering.

2. Experiential Learning Basics

Kolb's (1984) model for experiential learning focuses on a cyclical active experimentation process where students are given opportunities for hands-on practice (concrete experiences) allowing them to immediately observe what works and what doesn't (reflective observation) followed by thinking of ways to improve the results (abstract conceptualization) and further testing. Figure 1 shows this cyclic process as presented by McCleod (2013) but adapted to the NWP course.

This educational model is especially well suited for an undergraduate numerical modeling class where students can experiment with different modeling techniques to determine what works and what doesn't, then apply theoretical concepts to determine the best action to take to correct the problems. Key to experiential learning is immediate feedback, which MATLAB provides.

MATLAB was chosen as the programming language for the course both because of its benefits for experiential learning as well as other more pragmatic reasons. First, all meteorology majors at ERAU-DB are required to take an introductory course in scientific computing taught entirely in MATLAB. This gives the students the skills necessary to be immediately productive in the NWP class with minimal review. Second, MATLAB is a relatively easy language to learn with substantial on-line help so students can spend more time focusing on experimenting with the application of theoretical concepts and less time on learning to code. Third, MATLAB allows students to plot and view their results quickly and easily, which increases the efficiency of the active experimentation process. For example, when examining the numerical stability of different finite-difference schemes, students can immediately see the impact of violating CFL criteria. They can also experiment with the sensitivity of the scheme to CFL criteria by varying the time step or advective speed and noting the impact on the success of the numerical solution. Lastly, MATLAB's fast Fourier transform (FFT) routines are simple to use, allowing the introduction of spectral and pseudo-spectral methods at the undergraduate level.

**Corresponding author address*: Thomas A. Guinn, Embry-Riddle Aeronautical University, Dept. of Applied Aviation Sciences, Daytona Beach, FL 32114-3900; email: guinnt@erau.edu

3. Course Details

The NWP course is offered at the senior level and requires students to have completed their introductory computing course plus their two-semester atmospheric dynamics sequence and their differential equations and matrix methods course (a single combined four-credit course) as well. This ensures they have adequate familiarity with the equations used in NWP as well as the basic math concepts for manipulating the equations. Additional math concepts are taught within the course as needed, such as Fourier series and discrete Fourier transforms, which are necessary for the discussing spectral modeling. MATLAB is used exclusively for the quantitative concepts taught within the course such as: finite differencing, stability analysis, spectral and pseudo-spectral methods, explicit schemes, and time-integration schemes.

Because of the undergraduate nature of the course and the time constraints, some important operational NWP topics are only taught at a more descriptive level with little use of MATLAB. These topics include: parameterization of sub-grid scale processes, data assimilation, vertical coordinates, an overview of current operational numerical models, and spectral methods extended to the sphere.

The textbook chosen for the course was Decaria and van Knowe (2014). This text worked well because it was specifically designed as an introductory atmospheric modeling book geared directly towards undergraduate students. To the author's knowledge, this is only NWP textbook written specifically towards undergraduates. Theory is provided along with coding examples to aid in experiential learning. In addition, Lackmann (2011) is used for some of the more qualitative topics because of thorough overview and impressive graphics. Since the Lackmann text is also used for another course, it required no extra expenditure by the students.

The original intention of the course was to culminate in students running and experimenting with a spectral version of a barotropic vorticity equation (BVE) model written in MATLAB. However, the course pace didn't allow students to progress enough to make this possible. Spectral methods were therefore limited to the one dimensional case with only a qualitative expansion to two dimensions as well as the sphere. The course instead finished with students applying FD techniques to the two-dimensional advection equation with constant winds

4. Experiential Learning Techniques Employed

Homework and lab experiences were developed to be experiential in nature. Several different techniques were provided depending on the learning outcomes of the section. These techniques included the following.

The first technique involved drawing parallels between existing code and required new code. They were also provided a graphic of the final expected result. For example, to learn how to compute spatial derivatives, students were given an analytic field of geopotential heights (Holton, 2014) and asked to compute the geostrophic winds and vorticity (Fig. 2). For this exercise students were given portions of code, such as the x-derivatives of certain variables, and they then had to complete the remaining derivatives, both at the interior grid points as well as at the boundaries. Providing portions of the code helped instill confidence in their coding as well as some level of standardization for grading. In addition, since they knew the desired results, students could immediately test their code to determine if their changes worked (reflect). If not, they could reexamine the lecture material (conceptualize) to make proper adjustments (test).

A second technique was to provide working code for one time integration scheme and have the students re-write the code to yield a different time integration scheme. For this exercise, student had to know in advance which schemes were absolutely unstable and which would work. If not, they would not know if poor results were caused by the stability of the scheme or coding errors. Again, students were able to use active experimentation blended with theory (conceptualization) to find the desired solution. This method was used to examine the FD schemes applied to the one-dimensional advection equation as shown in Fig. 3.

Yet a third technique employed was to give students “broken” code; that is, code with intentional syntax or logic errors inserted. When correcting the syntax errors, students had to examine the output to determine the most likely cause of the logic error. Once completed students were required to compare results from the different schemes as shown in Fig. 4.

The final method was to give students partially developed code but with a time-step that violated CFL criteria. Students were not told the specifics of the error but had to arrive at the conclusion after experimentation. Students also had to recognize the difference between an unstable scheme and a scheme that was simply noisy due to spatial discontinuities in the initial conditions. An example of this type of exercise is shown in Fig. 5.

5. Summary, Challenges and Limitations

In summary, the NWP course provided experiential learning with the goal of improving both competency and confidence in scientific computing at the undergraduate level. MATLAB provided an excellent platform to allow students to actively experiment with different atmospheric modeling concepts, immediately view the results, and apply theory to improve the outcome. However, throughout the course, several challenges were noted.

Perhaps the biggest challenge observed during the course occurred during the reflection and conceptualization stages of experiential learning. Students had a tendency to simply try random coding changes in an effort to correct perceived errors rather than closely examining the properties of the error and comparing with theory. This could potentially stem from the nearly instantaneous speed at which MATLAB provides feedback. That is, if simulations took significantly longer to run, the students may have reflected on the errors more carefully to arrive at more logical coding changes. Therefore, one of the primary advantages of MATLAB in experiential learning could also be viewed as a potential disadvantage.

Another challenge involved going from theory to coding. Student feedback from the course indicated difficulty in translating the mathematical concepts into MATLAB code. To correct this, more examples directly linking the mathematics to the code will be created in the next course offering.

It should be noted that a significant limitation of using an interpretive language, such as MATLAB or Python, was the absence of instruction in working with a complied language (*e.g.,* FORTRAN) Gaining the skills to work with a compiled language would be most useful for students planning to attend graduate school. This is especially true for students with a desire to conduct more complicated simulations requiring the computational efficiency of a compiled language. Currently, these skills are only offered through computational mathematics courses or through independent study.

6. References

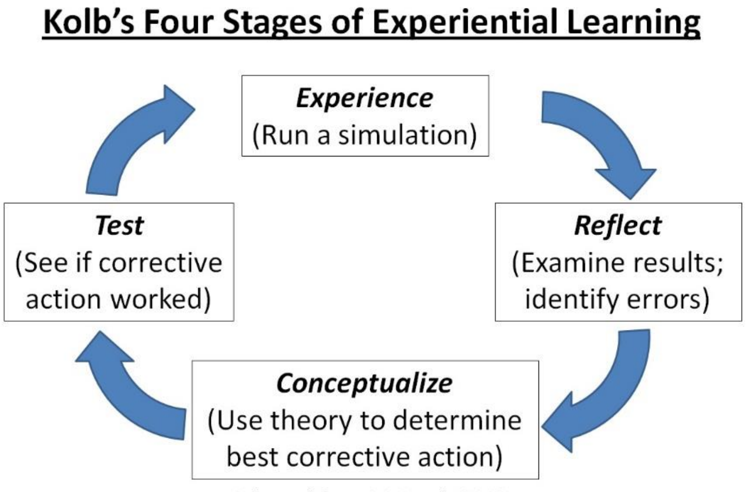
DeCaria, A.J, and G.E. Van Knowe, 2014: *A First Course in Atmospheric Numerical Modeling*. Sundog Publishing, 320pp.

Holton, J.R., 2004: *An Introduction to Dynamic Meteorology, 4th Ed.*, Elsevier Academic Press, 535 pp.

Kolb, D.A., 1984*: Experiential learning: Experience as the source of learning and development (Vol. 1)*. Englewood Cliffs, NJ: Prentice-Hall.

Lackmann, G.M., 2011: *Mid-Latitude Synoptic Meteorology, Dynamics, Analysis and Forecasting*. Amer. Meteor. Soc., 345pp.

McLeod, S. A. (2013). Kolb - Learning Styles. Retrieved from www.simplypsychology.org/learning-kolb.html



Adapted from McLeod, 2013

Figure 1. Kolb's Four Stages of Learning as applied to the NWP Course. Adapted from McLeod (2013).

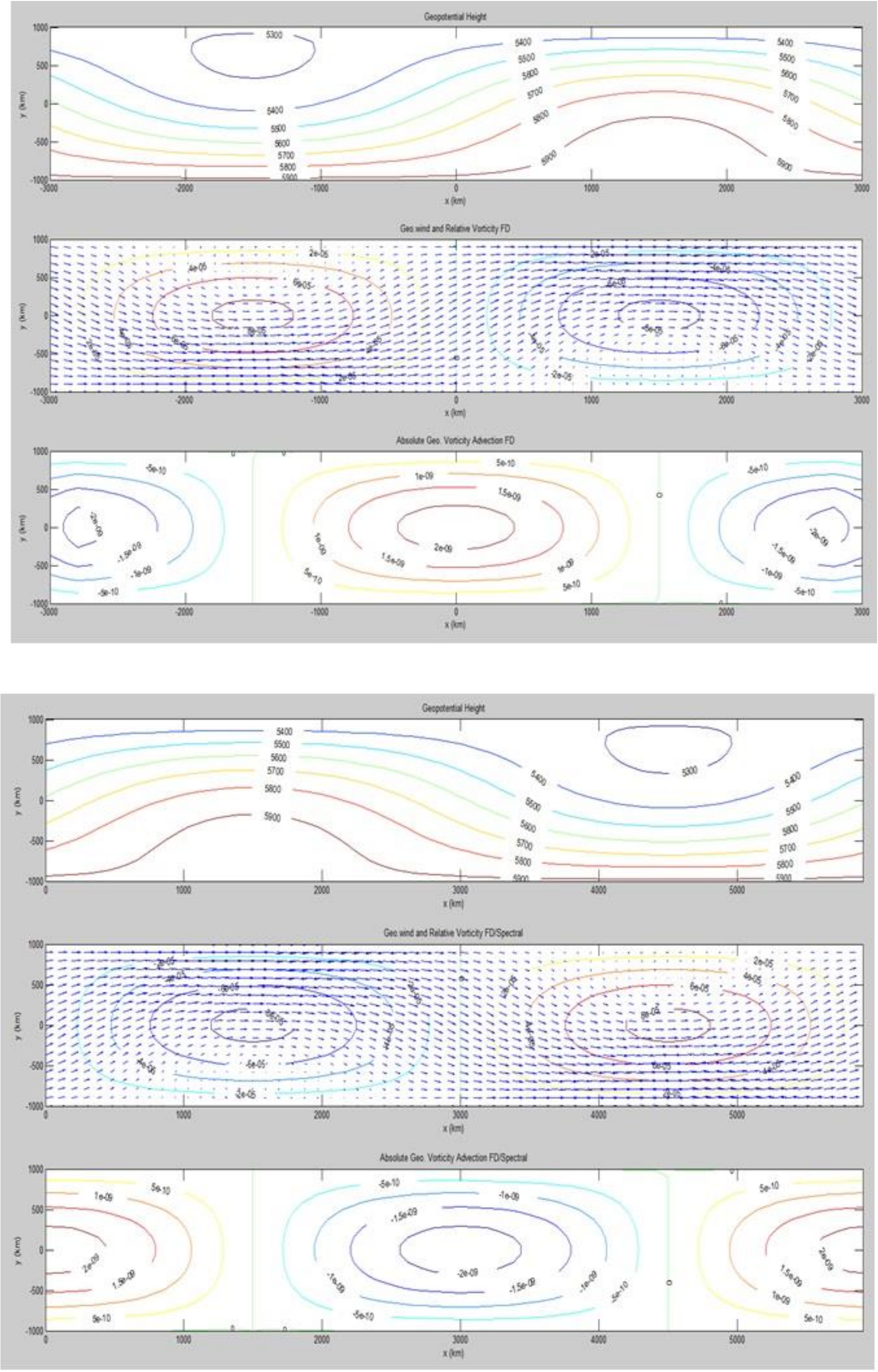


Figure 2. Sample output from an exercise where students were required to apply the concept of finite differences (top) and the spectral method (bottom) to calculate the necessary spatial derivatives to create the geostrophic winds and relative vorticity fields given an analytic geopotential height field. (Adapted from Holton, 2004).

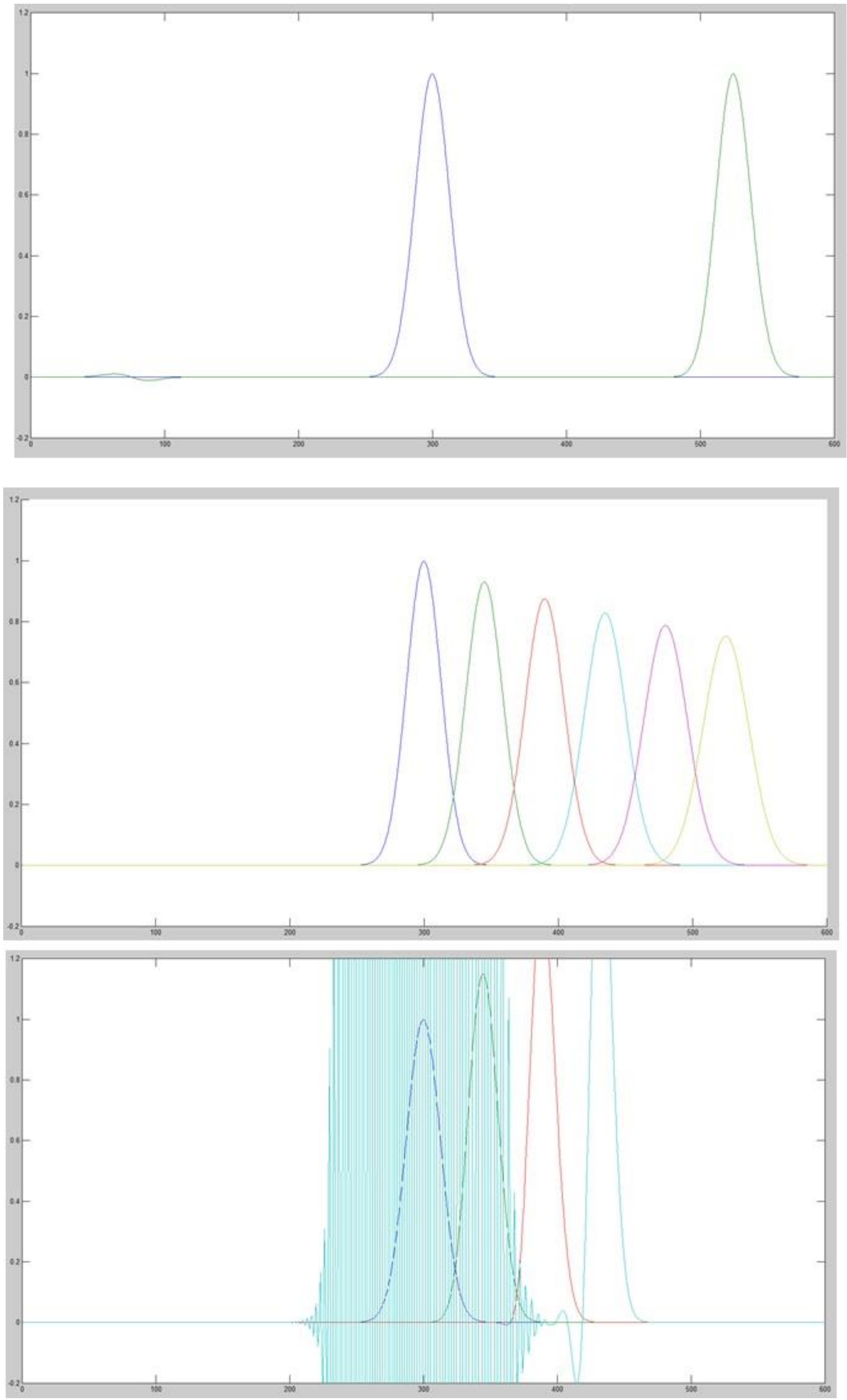


Figure 3. Sample output from a student exercise requiring students to modify a forward-time, centered space scheme (top) to create a centered-time, backwards-space scheme and a forward-time, centered space scheme to compare conditionally stable and absolutely unstable schemes. .

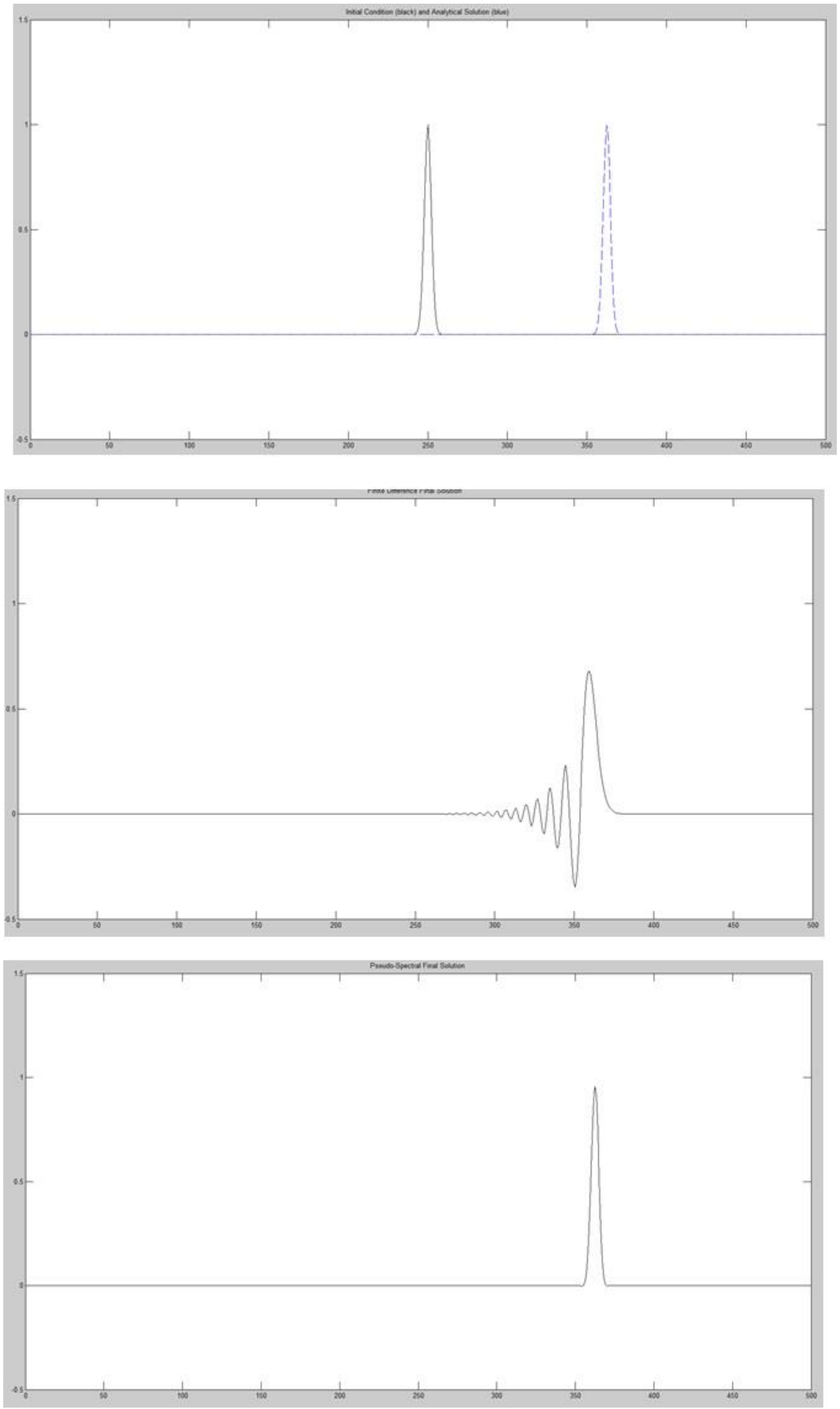


Figure 4. Student exercise asking students to correct coding errors then compare an Adams-Bashform scheme applied to a narrow Gaussian pulse using both a finite difference scheme (middle) and pseudo-spectral scheme (top) for spatial derivatives. Comparison is made to the analytical solution (top). Students are asked to discuss the computational mode as well as the accuracy of the spatial derivatives.

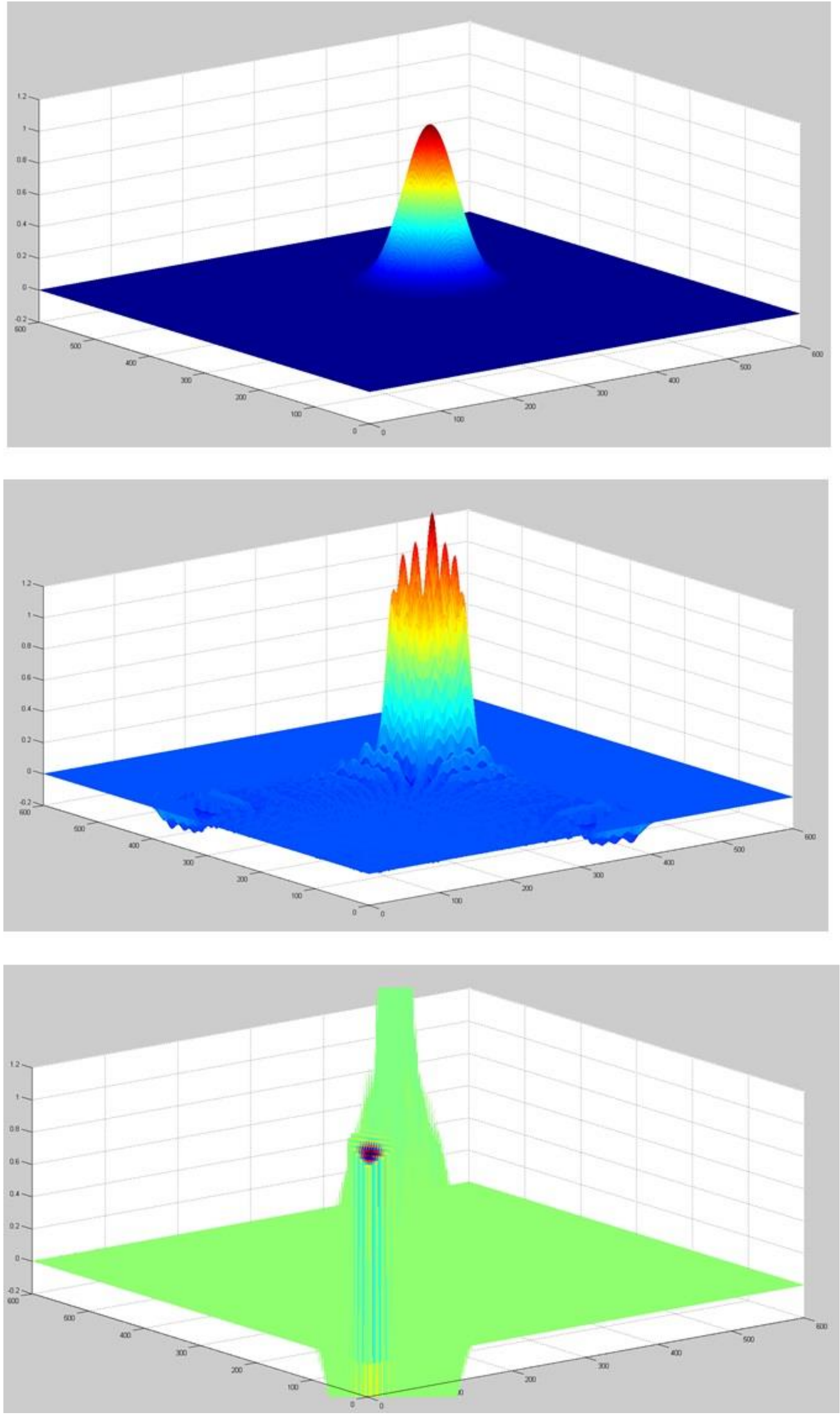


Figure 5. Sample output from a student exercise asking them to compare results from the 2D advection equation applied to a Gaussian pulse (top) and a discontinuous, block shape (middle). The bottom pane shows a Gaussian shape using a time-step that violates CFL criteria. Students were expected to use the effective grid spacing and wind magnitude to calculate the maximum time step that did not violate the CFL criteria. In addition, they had to recognize noise versus instability (middle vs bottom).