

December 2021

Volume 12 Issue 3

The background features a dark blue gradient with glowing yellow and white circuit-like patterns. Faint mathematical formulas are scattered throughout, including $\frac{dN(t)}{dt}$, $N(t) = N(0)e^{-\lambda t}$, $\int_0^{\infty} f(x) dx$, $\text{return}(\text{double})(U);$, (double) , $(\text{int} n) f$, $(\text{int} n-1) +$, and (t) .

JOCSE

Journal Of Computational Science Education

**Promoting the Use of
Computational Science
Through Education**

ISSN 2153-4136 (online)

JOCSE

Journal Of Computational Science Education

<i>Editor:</i>	Steven Gordon
<i>Associate Editors:</i>	Thomas Hacker, Holly Hirst, David Joiner, Ashok Krishnamurthy, Robert Panoff, Helen Piontkivska, Susan Ragan, Shawn Sendlinger, D.E. Stevenson, Mayya Tokman, Theresa Windus
<i>Technical Editor:</i>	Aaron Weeden
<i>Web Development:</i>	Jennifer Houchins, Valerie Gartland, Aaron Weeden, Claire Thananopavarn
<i>Graphics:</i>	Steven Behun, Heather Marvin

The Journal Of Computational Science Education (JOCSE), ISSN 2153-4136, published in online form, is a supported publication of the Shodor Education Foundation Incorporated. Materials accepted by JOCSE will be hosted on the JOCSE website and will be catalogued by the Computational Science Education Reference Desk (CSERD) for inclusion in the National Science Digital Library (NSDL).

Subscription: JOCSE is a freely available online peer-reviewed publication which can be accessed at <http://jocse.org>.

Copyright ©JOCSE 2021 by the Journal Of Computational Science Education, a supported publication of the Shodor Education Foundation Incorporated.

Contents

Introduction to Volume 12 Issue 3 <i>Steven I. Gordon, Editor</i>	1
Student Simulations of Local Wildfires in a Liberal Arts Geography Course <i>Ted Wetherbee and Elizabeth Jones</i>	2
Expanding HLRS Academic HPC Simulation Training Programs to More Target Groups <i>Tibor Döpper, Bärbel Große-Wöhrmann, Doris Lindner, Darko Milakovic, Jutta Oexle, Michael M. Resch, Oliver Scheel, Sven Slotosch, and Leon Widmaier</i>	13
Infusing Fundamental Competencies of Computational Science to the General Undergraduate Curriculum <i>Ana C. González-Ríos</i>	27

Introduction to Volume 12 Issue 3

Steven I. Gordon
Editor
The Ohio State University
Columbus, OH
gordon.1@osu.edu

FOREWORD

This issue begins with an article by Wetherbee and Jones describing the use of fire simulation codes to introduce students to simulation of a real-world problem impacting their local environment. Students examined a number of the parameters that impact the spread of wildfires and visualized the resulting impacts.

Döpfer et al. describe the content of two major HPC computer simulation projects at the German National Supercomputing Center HLRS aimed at middle and high school students and professionals, respectively. The Simulated Worlds project involves teachers and students in a variety of topics introducing

mathematical modeling and computer simulation. The Supercomputing Academy offers professionals experiences with HPC simulations of interest to industry. The article summarizes the approach to both programs and provides examples of the project contents.

Finally, González-Ríos provides a description of an introductory undergraduate course in modeling and simulation at the University of Puerto Rico - Mayagüez. The course aims to infuse the fundamental competencies in computational science into the undergraduate curriculum. The course uses Python to introduce students from a variety of disciplines to both programming and concepts of modeling and simulation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

© 2021 Journal of Computational Science Education

Student Simulations of Local Wildfires in a Liberal Arts Geography Course

Ted Wetherbee
Fond du Lac Tribal & Community College
ted@fdltcc.edu

Elizabeth Jones
Fond du Lac Tribal & Community College
ejones@fdltcc.edu

ABSTRACT

Wildfire simulations are developed for interactive use in online geography classes under the course titled Disasters. Development of local capability to design and offer computational activities in courses at a small, rural college is a long-term activity based on integrated scientific research and education efforts.

Keywords

Wildfires, Simulations, Computational science partnerships

1. INTRODUCTION

A wildfire is one of the events studied in the Fond du Lac Tribal & Community College (FDLTCC) geography course titled Disasters. This course has been offered online for several years, so spring 2020 and planned spring 2021 delivery have not been affected by COVID-19. It has minimal prerequisites and serves to satisfy a liberal education requirement for associate degrees. The material is often newsworthy enough to command headlines, but history and dramatic video footage are not interactive. We sought to add a “hands-on” component through computer simulation exercises so that students could directly explore some of these events. This is also an opportunity to add special value to an online course; we know every student has a capable computer in hand with a good connection to the Internet.

Rather than study wildfires as a general topic, we wanted students to experiment with specific wildfire variables using local and familiar geographic areas, say surrounding our campus forest or at the Big Lake meeting grounds. Simulations cannot have too fine a scale where individual trees are resolved, but at a reasonable scale, we can model fire behavior in ways interesting to students (they recognize the land) and illustrating key drivers of wildfire growth that they can control by selection: fuel, wind, topography, and moisture. Our needs for this particular class defined the modeling application.

Wildfire propagation is difficult to predict because of the complexity of fuel, terrain, induced weather, and other variables; yet modeling can still be helpful, and several kinds of models are used. WRF-fire and Sfire [5] were developed to study wildfires within the Weather Research Forecast (WRF [6]) code framework. WRF-fire and Sfire are designed for large-scale fires, say within a 50 km x 50 km area or larger, and these codes require considerable

wall-clock time to evolve as grid resolution increases. At this scale, the size of an individual cell might contain our entire campus forest. These are 2D fire-line propagation models driven by external conditions provided by WRF.

The grid size of WRF can be refined arbitrarily in horizontal directions, and the vertical resolution can also be refined by creating more pressure/eta levels. Sfire and WRF-fire run on a separate finer grid that is coupled to the finest WRF grid. It is a 2D model with inputs from the WRF grid and outputs to WRF grid variables. Sfire is used for large-scale wildfire simulations which may cover an entire mountain-side and burn for days and even weeks. Sfire scales using the same WRF mechanisms for use on multiple processes so that simulations can be run on large clusters.

The Sfire group created configuration, runtime, post-run, and display tools so that wildfire simulation results are more easily used by researchers and viewed by the public. In fact, the Sfire group offered to conduct simulations specifically for our Disasters class (given a location and date of a past fire in their catalog) so that students could examine results through the web interface.

We wanted students to set up and run simulations in some interactive fashion, and we could not run Sfire fast enough on our local machines. Our target for a real-time simulation was 15 minutes maximum. This figure can be achieved on modest workstations with a sufficiently coarse grid, and we did just this for classes in previous years using WRF-fire instead by directly modifying the stock ideal wildfire problem provided. A straight-line plume expanding in the wind direction was the consistent result, and visualization quality was poor because of the coarse grid. This is not surprising, because WRF is a meso-scale model designed to predict Earth weather.

We are more interested in small-scale fires that evolve, say, within a 1 km x 1 km area. This scale is suitable for educational uses. Our technical goals are several.

1. Problem evolution is fast enough for immediate feedback. Students—like researchers—want to see results from their experiments in short order.
2. Recognizable local terrain features are clear in visualization.
3. Students set the fire ignition point or line, fuel moisture, and wind conditions in a visual fashion.
4. Running simulations are viewable by everyone, and finished results are archived and viewable by everyone.
5. Controls for setting fire simulations, tracking jobs on the queue, and viewing results are simple and intuitive.
6. Students in 2 sections of 35 students each can make several runs over a period of 1–2 weeks.
7. An ordinary stock web browser front-end to the simulation exercise is sufficient, i.e., there are no machine-specific requirements, and no special web browser plug-ins are needed.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

2. HISTORY

Our problem came first. Wildfires are a local concern for practical and historical reasons. The “Fire of 1918” destroyed Cloquet, Minnesota, and much of the surrounding region, and it is still the deadliest in the United States for loss of life (453 deaths). In fall 1918, when local WWI and flu epidemic casualties were peaking, fire driven by high winds consumed towns and forests within a 1,000 mi² region in a matter of hours.

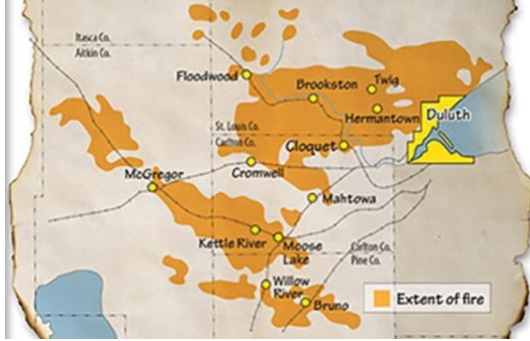


Figure 1. 1918 Fire (Duluth News Tribune, 2018).

102 years later, the region surrounding Cloquet is again mostly forest, and it is now used for a mix of production and recreation.

The FDLTCC campus was built within a planted red pine production “farm” that grew past its ideal thinning and harvesting times. It now looks much like ordinary “wild” forest in surrounding areas, except that the tallest red pines are aligned in neat rows at somewhat uniform height.

This particular stand of trees would have been harvested for telephone poles earlier had it remained in production, and there would have been periodic brushing (removing undergrowth) to reduce fire danger to the canopy. It is more of a fire hazard than necessary, and future aesthetic value of tall red pines (up to 250 feet) is eliminated by over-crowding.

In fact, there was a wildfire in the campus forest within a year after the campus was built in 1993. At that time, the undergrowth was low, so an undergrowth fire burned for several hours during the night without any damage and before notice. Tall red pines can easily survive a surface fire, and the Cloquet fire department quickly put out the fire because ordinary trucks could move between trees over low brush.

3. SIMULATION DETAILS

Earlier, we had developed fluid code for use at FDLTCC in collaboration with The Laboratory for Computational Science and Engineering at The University of Minnesota (LCSE). This was done in order to simulate tracer flow over complex terrain for a different problem: track flow and dispersion of a benzene cloud. This disaster occurred in Duluth-Superior during 1991 from a railroad tanker derailment and rupture off the Nemadji River Bridge.

The code tracks tracers well toward visualization of smoke and gaseous products. For fire propagation, the movement and state of the fluid can be used directly. Heat dries out vegetation, which is then easier to ignite, and hot gas rises for a chimney effect on slopes. We can add radiation from burning vegetation and then add a cellular probability function for cell ignition based on fuel, proximity to burning cells, moisture, and temperature. This allows for a physics-based simulation to a degree practical for the course.

3.1 Fluid Code

The fluid code Piecewise Linear Advection and Boltzmann (PLAB) is a finite volume Godunov method [4] code similar to Piecewise Parabolic Method (PPM [3]) and Piecewise Parabolic Boltzmann (PPB [11]) codes but with linear sub-cell reconstruction and representation, respectively. Linear methods are simpler and sufficient for our relatively low-speed flow fluid problems. Higher-order parabolic methods were used at this time by Woodward’s Blue Waters (BW [1]) Petascale Team to study evolution of a Sakurai’s Object-class star [10]. The star burns a different fuel (H and He) and at a vastly different scale, yet these codes are alike in key principles and implementation details that matter for successful execution.

An interesting aspect of this type of fluid code is that, while details can be complex, it is a straightforward application of conservation and ideal gas laws that students already know. We can explain the essence of how it works in visual fashion to our lower division students. This may seem to be a tall claim that a research code suitable for leading-edge astrophysical simulations on BW is something lower division students in a liberal arts course can follow in some fashion. Some explanation is due.

The Euler equations in one dimension for a compressible fluid are usually used to describe the situation.

$$\begin{aligned}\rho_t + (\rho u)_x &= 0 \\ (\rho u)_t + (\rho u^2 + p)_x &= 0 \\ E_t + (uE + up)_x &= 0\end{aligned}$$

These express the relationship of mass ρ , velocity u , momentum ρu , pressure p , and energy E across space x and time t , and they are linked through an equation of state. They apply, but a more intuitive view of the situation is actually used.

We have cells in a 3D rectangular grid that all affect each other eventually. Yet, over a sufficiently small time increment in which sound waves move less than a cell width, we just have to figure out what crosses each face between every pair of adjoining cells in each x , y , and z direction. Every cell has an average state: pressure p , density ρ , and velocity u in the direction across the cell face.

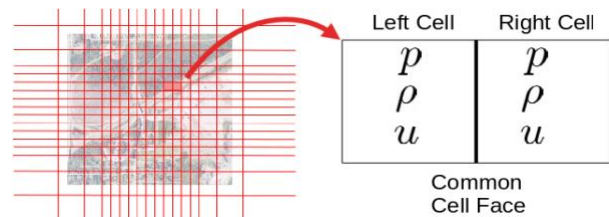


Figure 2. Cell face from a Cartesian grid.

From surrounding cells, we then reconstruct the value of each of these variables against the common cell face—left and right shown in Figure 3 for density ρ —by sub-cell distribution curves.

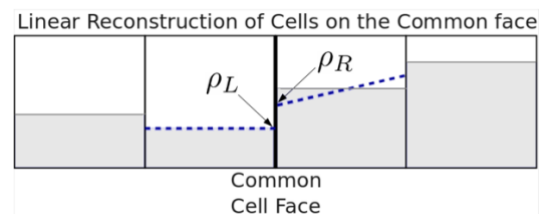


Figure 3. Reconstruction.

These curves model how real fluids behave at a sub-cell level. Our PLAB code uses linear curves sufficient for gentle fluid dynamics, while PPM uses parabolas, which are important for stronger shock waves driven by, say, thermonuclear explosions.

The Riemann solution is the state of the “star region” about the moving interface (between light dashed lines). The interface (heavy dashed line) moves with velocity u^* —left for the case shown in Figure 4. The star region expands at the speed of sound with a common pressure p^* and densities ρ_R^* and ρ_L^* on each side of the moving interface.

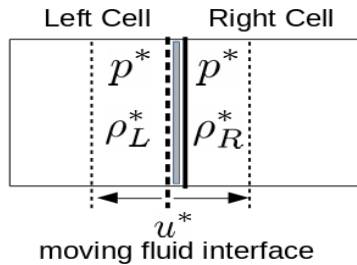


Figure 4. Riemann solution.

Mass, momentum, and energy pass from one cell to another. This is the flux. The little gray piece in Figure 4 that has moved left across the face represents flux, and we calculate this from the star region state. We do this for all cells sequentially in x , y , and z directions, then we repeat to evolve the problem over time. For us, the solution is an illustrative video of the wildfire.

The tracer (smoke and other fire products) carried within the fluid is represented explicitly in each cell, a Boltzmann-style sub-cell distribution. This method accurately tracks tracers, and it doubles the effective resolution in each dimension toward visualization. In fact, multiple tracers can be represented within the fluid, each with its own distribution, but one tracer is enough for this wildfire simulation.

Expressing this sub-cell distribution in usable form amounts to calculating the volume fraction and moments with respect to x , y , and z of the tracer in a $[-\frac{1}{2}, \frac{1}{2}]^3$ cell then limiting the slope so that our distribution function returns a value in $[0,1]$ for each point in the cell.

$$T(x, y, z) = T_0 + T_x x + T_y y + T_z z$$

The constants T_x , T_y , and T_z are each 12 times the moment in the x , y , and z axis direction, respectively. Tracer is just advected with density in our wildfire code, and we set the mass T_0 for the air cell at or immediately above a burning cell. After each advection step, the new tracer distribution is calculated for each cell.

Work with this function employs standard techniques from our Calculus 1 & 2 courses, a nice applied example used for these classes. It is easier to illustrate the idea first in 2D so that the density z is a function of the position (x,y) in a $[-\frac{1}{2}, \frac{1}{2}]^2$ square, i.e., we have a plane in space to view:

$$z = T(x, y) = T_0 + T_x x + T_y y.$$

Students can see that it really works, too: crisp, realistic tracer flow.

We feel that these methods play well to intuition and experience held by our liberal arts students. We can explain how this wildfire simulation works, more or less, depending on what students want to know.

Our PLAB code also uses the Simple Line Interface Calculation (SLIC [7]) method for solid boundaries, so we can embed the solid surface terrain, buildings, and partial solid items like tree canopy within the fluid code. Thus, dynamical features of wildfires such as eddies and chimney-like effects of slopes are simulated directly.

PLAB is not an adaptive mesh refinement (AMR) code, nor is it nestable like WRF, so we stretched the lateral and top boundaries outward from the focus of the simulation within a fine grid center. Stretching provides dampening of high frequencies generated within the fine grid, and it also provides a way to maintain outer fluid boundaries during relatively brief student simulations. Smoke does “pile up” within the boundaries in our visualizations because of these; this is a visual flaw we currently accept.

3.2 Visualization Code

Visualization code in various forms was built into earlier educational applications at FDLTCC. A ray casting volume rendering code Srend [9] was developed for use within BW Sakurai Object simulation code, and the same visualization code was used in new educational applications at FDLTCC, including this wildfire simulation. Running simulations wrote rendered images to website directories, then these images were drawn as available by web applications to show imagery and also turn sequences of frames into movies.

Visualization is a special challenge when the number of processes rises. Traditional strategies of dumping data to disk for post-processing do work, and BW is designed for this. BW storage can be written to by simulation XE nodes; then a separate job on XC nodes (the visualization cluster) can read the data, process it to imagery, then write results to storage. However, data can be condensed significantly through rendering in-place and deliver useful imagery during running simulations. This is a worthwhile strategy if in-core volume rendering is fast enough and if pre-defined rendering parameters are sufficient to explore simulation results.

Our BW team had investigated the idea that an in-core method for volume rendering could simplify aspects of imagery generation, handling, and delivery on BW as well as other machines toward exascale performance. Other significant benefits from embedding in-core visualization within simulation code are elimination of dependencies and delay associated with visualization codes and post-processing. This is a high-value convenience for research using petascale clusters, but it is an essential feature for our educational applications.

Of note—which may be surprising—is that scalable fluid and visualization code capable of employing every node of the largest clusters also runs perfectly on laptop and desktop machines, just slower or on smaller problems.

A rendering of stellar fusion fuel was done using 13,824 MPI ranks on BW compute nodes, and “smoke” was rendered by 32 MPI ranks on a local machine: same code, different numbers.



Figure 5. Stellar fuel and wildfire “smoke” visualization.

We can scale our wildfire application on machines we have available to suit the class, i.e., adjust the grid size and evolution time to get results delivered to students within acceptable limits.

3.3 Wildfire Simulation

We integrated the PLAB fluid code with Srend for runtime visualization, then we added variables and physics for wildfires. The tracer variable T for the simulation injected in a cell was defined to be mass of fuel (wood, grass, ...) consumed by the fire. This tracer includes smoke particles, CO_2 , water, and other gases.

Students define ignition points by drawing an ignition line using a mouse on a web image of a pre-defined, 1 km area surrounding the FDLTCC campus. They can switch between an image, an elevation map, and a fuel category map of the same area when they set the ignition line. The wind speed and direction can be set by clicking on a compass rose. Fuel moisture percentage is set using a slider.

Fire evolution itself is generated by a combination of physics and cellular methods. The fire grid is a 2D array at the same resolution as the fine horizontal inner grid of the fluid code. Each fire grid cell has variables:

- 1) Remaining fuel: in kg, initialized by the fuel category value by cell position.
- 2) Moisture: in percent, initialized by the student using the slider, where kg of water = kg of fuel * percent of initial moisture.
- 3) Ignition status: burning (1) or not burning (0), initialized to 1 if on the student ignition line and 0 if not.
- 4) Energy in from burning step: in J, initialized at 0.

For each fire evolution step, there is a burning step followed by an ignition step. In the burning step for each cell with burning = 1, a fraction of remaining fuel is burned, then:

- 1) The mass burned is injected within the same fluid cell as a tracer.
- 2) The energy production (1.4 MJ/kg burned fuel) from burning is radiated. Surface cells in the line of sight can absorb energy.
- 3) Each cell transfers energy in to evaporation of moisture, and this energy is added to the fluid.

The ignition step is a cellular model but driven by physics. As the moisture content approaches zero and more touching cells are burning, the cell's probability of ignition rises. The temperature of the fluid (air) also changes the probability of ignition. Wind direction and relative elevation of cells are not used directly; the fluid code advects fluid in the direction of the wind and allows hot gas to rise. A random number is generated then input to the probability function to determine ignition.

This fire evolution model was tweaked through trials and guided by appearance of real wildfires. For limitations, we do not formally validate our models against other wildfire models or real fires but focus on educational uses; we make no claims about predicting real fires. On the other hand, we observe fire-line propagation in our models similar to that of real fires and research-grade wildfire models.

3.4 Delivery to Students

Our original server for this exercise was a 4-core Linux machine running an Apache webserver. There is JavaScript within the student web page so that students can navigate between views and define parameters visually, but form and CGI methods are mostly unchanged from the mid-1990s. The CGI script `fire2.pl` checks the

key and form variables then loads a job within the queue by writing variables to a file in the queue `q2/` directory. Another CGI script `queue2.pl` shows the queue to students and displays results from any running simulation whenever the page is reloaded. The queue itself is processed by a perpetually running script `runfire.pl` that sequentially runs jobs (files in the queue directory) ordered by file creation time. The fire simulation code `plab_fire` is launched by the `runfire.pl` script after modifying the `namelist` file with the run parameters, then the simulation code runs the simulation according to the `namelist` file.

Completed runs—with parameters, a selection of images, and a movie generated from all images—are written to the `html/runs2/` directory by ID for public viewing.

3.5 Directions

The tutorial for the wildfire exercise comes after material and class discussion. A sequence of slides from the tutorial follow in the Appendix.

4. RESULTS

The wildfire activity has been used in several classes with very similar results.

4.1 Student Commentary

Students completed a post-simulation survey (spring 2017), and we include questions and responses from 4–7. Their comments are especially valuable; something can be done about specific complaints. We include their words complete with some minor punctuation corrections for clarity.

Q4: Did you find the wildfire simulation and visualization activity engaging?

- It was interesting. Better graphics and more variables would be nice.
- Yes, I enjoyed seeing the different effects that can cause fire to act differently.
- It was confusing and not very interesting at first, but then once I understood it better and was able to see the simulation and how my decisions affected the end result, it was much more engaging.
- Yes, I found it very interesting. I thought it was very well set up.
- Yes, after I got the simulations going and could see what they were all about, I really enjoyed it.
- I found it confusing at first, but once I got the hang of it, I did think it was engaging.
- Not really. I was kind of confused how to run it, even after directions. I did not really understand what I was trying to see or figure out. I wish it were explained to me better.
- It was kind of boring until I understood what I was doing.
- Yes, it was engaging.
- Definitely, I enjoy doing visualization activity. I get more out of doing things like that and hands on.
- Yes, but it could be a little faster and maybe updated. It did do its job, though.
- Yes, I found the simulation very engaging and enjoyable. It was very easy to navigate.
- Absolutely!
- The simulation was very neat in the fact that when you predict the way the wildfire would be, the wildfire does

something else. The fires did many different things that were unpredictable, but it showed how different each one was with the alternate factors.

- Yes, I did. It was interesting watching how things could develop though this activity.
- Yes, once I got the hang of it, it was very intriguing to see what my simulation would end up being like, being that the results are not given right away.

Q5: What was the most effective part of this learning module?

- How the change of different elements can affect the force of the wildfire.
- It gave me an idea of how wildfire actually works. It was nice to try new things and to be able to watch all of my runs.
- After doing this, you are able to actually visualize how different changes can affect wildfires.
- Seeing the results. Just wish it didn't take so long.
- I figured out that different spots strike up different fires. Maybe big ones or small ones.
- I would say the video were the most effective aspect of the module.
- Seeing how fast and hot fires can be and what fuel and wind factor into it.
- Actually getting to create the simulations ourselves rather than just watching ones that were already created.
- Everything about the module was effective because it shows us the dangers that could happen and gives us an opportunity on what to look out for.
- The most effective part of the module was seeing the fires counteract the predictions. The many factors that make each fire different makes them unique and spread differently. I learned that fires can spread just about anywhere, even if it's in an empty field or over a concrete highway.
- See how it all played out in the video.
- The most effective part were the different views that could be accessed by the clicking of a few simple buttons.
- That I was able to see the way the fire was going.
- Watching the fires, and how the winds and fire line play a role together.
- Having to write a summary about the simulations reinforced them.

Q6: Were there barriers to completing the activity? If so, please discuss.

- It was very confusing at first and lacked direction. You just kind of had to play with it and figure everything out for yourself.
- No.
- Just waiting for my simulations to run.
- I would say the only barrier would be the fact that the simulations run one at a time so it can get to be time consuming to get your simulations done. With proper planning this minor issue can be curbed.
- Yeah like sometimes it would not zoom in for me. That got super frustrating.
- I thought it was a little confusing how to find my way around the website after clicking on submit job.

- Yes, if you typed in the wrong code. You would have to re-enter everything. Also sometimes it would take a long time for the simulations to run.
- Yes, one I was sure if it was land cover in dirt over a very dry area. When trying to produce a fire, it was unsuccessful.
- Time was the major barrier. It took a lot of time from one run to another.
- I didn't encounter any issues in particular.
- Just too many people trying to complete runs at the same time.
- The barriers of completing this activity was that the video simulations did not work on my computer at home. I had to go to the school and play the video modules so I could summarize each one accurately.
- No.
- Following the directions closely. It gives you enough structure to do it.
- It was a little difficult to understand how the pass worked at first. I know it took me more than one try to really understand what I was doing even though there was a tutorial provided.

Q7: What are your suggestions for improvement, if any?

- Graphics and more variables.
- Maybe a better system to find your code within everyone else's code.
- I just suggest that the tutorial be supplied right away because it was very stressful trying to figure it out without it. Other than that, it was all right.
- More direction on how to use the website.
- I would try to explain the simulation better. Like inform the students what they are doing and explain to them what will happen. Make it more user friendly.
- I have no other suggestions.
- None.
- Make the process faster. This took me 5 hours to complete because of waiting time.
- Nothing really.
- Offer ways to configure computers at home like HP and Mac to play the videos from changing the settings.
- Longer video.
- I would try and maybe have weather conditions, such as light rain, rain, snow covered, just to add another seasonal variable.
- Update the system so it is less time between runs and better graphics.
- Maybe have different modules with different agriculture.
- The only thing I could suggest would be to have the simulations run faster.
- It would be interesting to pick completely different locations all around the country.

4.2 Notes on Student Comments

4.2.1 Engagement

Some student yeas for "engaging" were qualified with something like, "after I figured out what I was doing." This is not a terrible sign. A bit of initial mystery to solve is a wonderful spice. Our thoughts on the ideal instructions and user-interface are formed

about compromise between an interesting, educational puzzle and a sure-fire recipe. We definitely do shoot for pizzazz! Student opinions on the matter are there to guide us in this aspect.

4.2.2 *Waiting in the Queue*

Many students complained that simulations took too much time to run. This is a familiar complaint, and this was mostly about their job waiting in the queue. Everyone appreciates speedup when it is about solving their own problem. It brings up memories of researchers complaining, say about “those chemists hogging the machine.” We agree with almost all student complaints, and this is one that can be addressed in a straightforward fashion because this is a scalable simulation code. There are practical issues to think about, however, on the idea of migrating a scalable code to a faster but remote machine.

Setting up and using a remote machine is more complex than using a local machine, and queue waiting can make optimization efforts pointless. On the other hand, it can work well. For a previous project, we used SDSC Trestles, TACC Lonestar, and LONI Queenbee successfully for remote runs on demand handled by a local intermediate portal server, and these particular machines consistently had short queue wait times for small jobs—a feature carefully sought. Currently, the XSEDE project features SDSC machines aimed squarely at this kind of on-demand service [8].

For an interesting cluster management paradigm, we also formerly used LCSE clusters for educational applications from FDLTCC. The principle for LCSE clusters was that educational usage trumps production jobs, and interactive usage trumps all. LCSE cluster users were expected to write production codes with restart dumps so that they could be killed any time then gracefully restarted. Production is important, but a 336-hour production job can be delayed a bit to accommodate educational and interactive usage.

A newer machine reduced the simulation run time from 15 minutes to 4 minutes, so we partly addressed the complaint about delay.

Longer simulation runs would be nice to do. Obviously, students wanted to see what else would have happened when their simulation ended.

4.2.3 *Add Simulation Variables and Places*

Students want more variables in the simulation, namely arbitrary locations and weather conditions. We do, too. There are difficulties for some.

Real-time weather conditions are easiest to obtain because point conditions are sufficient in these fine-scale simulations. It is possible to obtain historical and contemporary forecast data from online sources then interpolate to desired position and date. A realistic method to incorporate rain or snow is an unknown to us other than by ramping up the moisture level in fuel. We might do well for this class exercise to have students simply look up current conditions or create their own. The current GUI allows wind speed and direction input as this is highly significant for wildfire evolution.

Fine-scale terrain and imagery data is obtainable online at various refinements. Fine-scale land cover data was not available at the fine scale of our simulations. We processed fine-scale imagery to 4 categories (grass, forest, buildings, and non-burnable) using Gimp raster features then edited 4-color raster images by hand over the simulation area using our knowledge of the land. Rather than try to store terrain, land cover, and imagery data for a large region (say Minnesota), or download and process data from a larger region (say continental US), we could practically prepare a selection of smaller

yet representative regions of interest and allow students to select one. These could be somewhat larger regions than our 1 km square region currently used.

4.2.4 *More and Better Graphics*

These are the easiest to address. Our visualization software (Srend) can deliver several sets of imagery using different views simultaneously, and volume rendering does not take much time compared to problem evolution. We used FFmpeg to convert sequences of images to video, and we can create alternative formats that work in more browsers. MP4 for video seems to be favored today.

5. ACKNOWLEDGMENTS

For some things which LCSE partners did not do, writing code, developing local techniques, and defining specific FDLTCC activities were done at FDLTCC. LCSE researchers could have just done these things, surely a lot easier and faster. However, encouragement and support in the form of technical advice was otherwise available, within limits. (This activity was not funded directly.) LCSE researchers usually have answers to problems with fluid dynamics, HPC techniques, visualization, and N-body codes, and they helped make resources available for development, testing, and running educational applications. It is impossible to overstate the value of research and education partners willing to help cultivate institutional capacity.

Allocations from TeraGrid [2] and the San Diego Supercomputing Center for development and testing were available and ready to use within 24 hours of request. Similar XSEDE response (swift and certain) is available, and there are specific services available for applications of this sort, i.e., web access to applications for use in classes running on remote machines.

NASA’s project Center for Applied Atmospheric Research and Education (*NASA-CAARE NNX15AQ02A*) supported refinement of wildfire modeling activities, and the current FDLTCC KNL node was purchased with NASA CAARE funds. A NASA funded project at FDLTCC (*Elizabeth Jones, “Environmental Modeling And Research Experience”, NNX11AQ96G*) provided startup funding for tracer flow modeling.

FDLTCC and Minnesota State College and Universities funds sabbaticals for community college faculty with the proviso that efforts contribute to the mission of teaching and learning. Faculty research activities are viewed as important components, and internal FDLTCC support is also available. We have enjoyed strong, continuous internal administrative support for these specific efforts described.

6. FUTURE PLANS

The current stretched grid wildfire model still works so we are using it for the spring 2021 classes, but we are working on an AMR code as a replacement for similar simulations which involve fluid flow over complex terrain. Incorporating external data is easier with a fixed 2:1 grid refinement ratio vs. stretched grids. Also, visualization of AMR data is an Srend capability to exploit. Stretching the grid was a solution to get this problem running in time for the Disasters class. The AMR code would not evolve this wildfire problem faster using the same number of finest grid cells, but it would be easier to scale and would apply to a broader set of problems.

7. REFERENCES

- [1] Brett Bode, Michelle Butler, Thom Dunning, Torsten Hoefler, William Kramer, William Gropp, and Wen-mei Hwu. 2013. The Blue Waters Super-System for Super-Science. *Contemporary High Performance Computing: From Petascale toward Exascale* (1st. ed.). Chapman and Hall/CRC, Boca Raton, FL.
- [2] Charlie Catlett. 2002. The Philosophy of TeraGrid: Building an Open, Extensible, Distributed TeraScale Facility. In *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, May 21–24, 2002, Berlin, Germany. IEEE Inc., Piscataway, NJ, 8. DOI: <https://doi.org/10.1109/CCGRID.2002.1017101>
- [3] Phillip Colella and Paul R. Woodward. 1984. The piecewise parabolic method for gas dynamical simulations. *J. Comput. Phys.* 54, 1 (April 1984), 174–201. DOI: [https://doi.org/10.1016/0021-9991\(84\)90143-8](https://doi.org/10.1016/0021-9991(84)90143-8)
- [4] Sergei Godunov, I. Bohachevsky. 1959. Finite Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Fluid Dynamics. *Matematicheskii Sbornik*, 47(89), 3 (1959), 271–306. hal-01620642
- [5] J. Mandel., J. D. Beezley, and A. K. Kochanski. 2011. Coupled atmosphere-wildland fire modeling with WRF 3.3 and SFIRE. *Geosci. Model Dev.* 4, 3 (July 2011), 591–610. DOI: <https://doi.org/10.5194/gmd-4-591-2011>
- [6] J. Michalakes, S. Chen, J. Dudhia, L. Hart, J. Klemp, J. Middlecoff, and W. Skamarock. 2001. Development of a Next Generation Regional Weather Research and Forecast Model. In *Developments in Teracomputing: Proceedings of the Ninth ECMWF Workshop on the Use of High Performance Computing in Meteorology*, November 13–17, 2000, Reading, UK. World Scientific, Singapore. 269–276. DOI: https://doi.org/10.1142/9789812799685_0024
- [7] William F. Noh and Paul Woodward. 1976. SLIC (Simple Line Interface Calculation). In *Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics* June 28 – July 2, 1976, Enschede, The Netherlands. Lecture Notes in Physics, Vol 59. Springer, Berlin/Heidelberg, Germany, 330–340. DOI: https://doi.org/10.1007/3-540-08004-X_336
- [8] John Towns, Timothy Cockerill, Maytal Dahan, Ian Foster, Kelly Gaiher, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D. Peterson, Ralph Roskies, J. Ray Scott, and Nancy Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Comput. Sci. Eng.* 16, 5 (Sept.–Oct. 2014), 62–74. DOI: <https://doi.org/10.1109/MCSE.2014.80>
- [9] Ted Wetherbee, Elizabeth Jones, Michael Knox, Stou Sandalski, and Paul Woodward. 2015. In-core volume rendering for Cartesian grid fluid dynamics simulations. In *XSEDE '15: Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, July 26–30, 2015, St. Louis, MO. ACM, New York, NY, 1–8. DOI: <https://doi.org/10.1145/2792745.2792780>
- [10] Paul R. Woodward, Falk Herwig, and Ted Wetherbee. 2018. Simulating Stellar Hydrodynamics at Extreme Scale. *Comput. Sci. Eng.* 20, 5 (Sep./Oct. 2018), 8–17. DOI: <https://doi.org/10.1109/MCSE.2018.05329811>
- [11] Paul R. Woodward, David Porter, William Dai, Tyler Fuchs, Tony Nowatzki, Michael Knox, Guy Dimonte, Falk Herwig, and Chris Fryer. 2010. The Piecewise-Parabolic Boltzmann Advection Scheme (PPB) Applied to Multifluid Hydrodynamics. In *Proceedings of the International Conference on Computational Science (ICCS 2010)*, May 31 – June 2, 2010, Amsterdam, The Netherlands. Elsevier, Amsterdam, The Netherlands, 10 pages.

APPENDIX: SLIDES


Wildfire Simulation and Visualization

Disasters – GEOG 2010
Spring 2017

Slide 1

The Fire Triangle

From Chapter 16 – Wildfires, we know that fires require three elements: fuel, heat, oxygen



Slide 2

Additional Factors

Fire growth and spread is influenced by a number of factors:

- Topography – presence of slopes, elevation
- Fuel moisture – humidity, soil moisture, presence of lush vegetation (or lack of)
- Wind conditions – wind speed, wind direction, often seasonally influenced
- Land cover – buildings and concrete verses forest or shrub

Slide 3

Wildfire Simulation

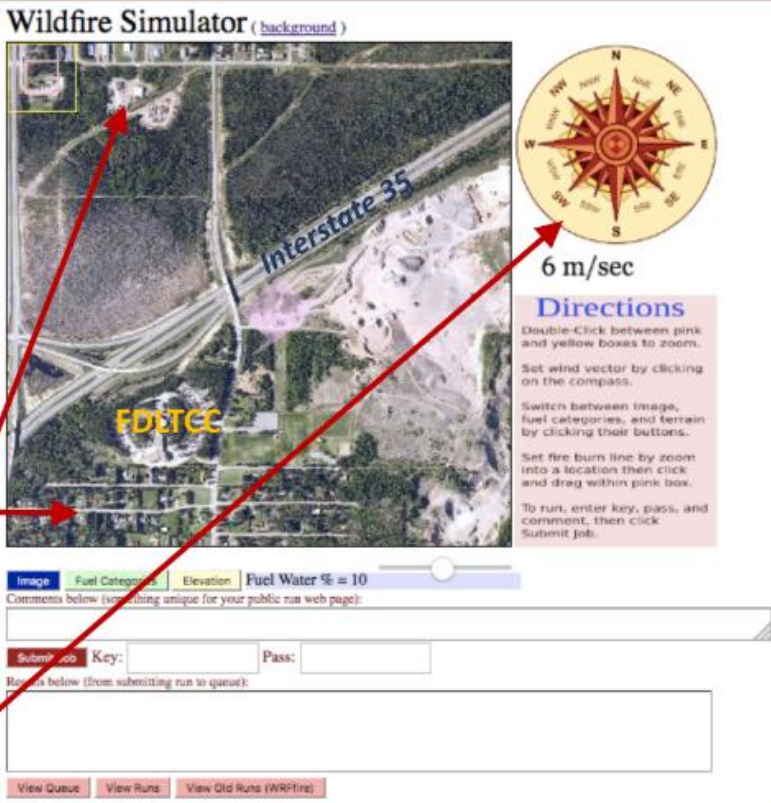
Let's apply these factors to the wildfire simulation interface, and look at the process of setting up a successful simulation.

Slide 4

This is what you'll see when you access the Wildfire Simulator.

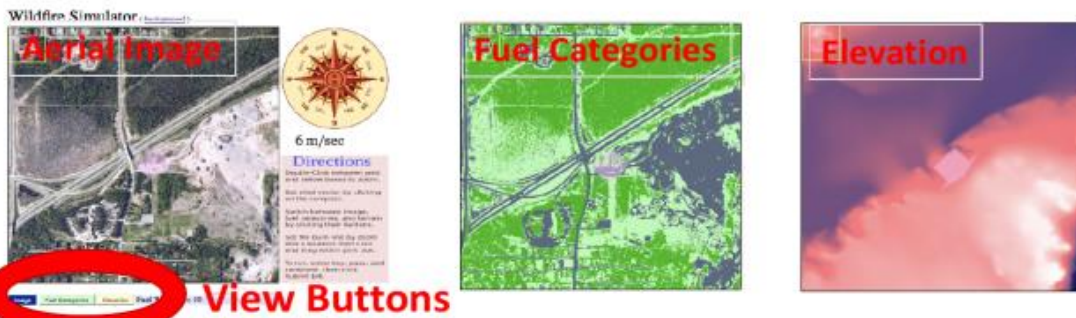
You'll notice that the image shows the campus in the lower left, Interstate 35 running diagonally from the lower left to the upper right, and residential areas to the south of the campus and in the upper portion of the image.

The compass rose provides a sense of cardinal direction.



Slide 5

Three Views of the Area of Interest



1. Image: The initial view you have of the area that you'll be setting up simulation runs within is an aerial image, taken most likely from an airplane. We are most familiar with this view, and it provides us with visual reference of our area.
2. Fuel Categories: Clicking the Fuel Categories button will show you the type of fuels available. Gray and purplish are likely impervious surfaces such as concrete, and barren land such as the gravel pit across the street from the college.
3. Elevation: Clicking the Elevation button will show you a gradient surface of elevation for the area. The darker areas are lower elevation, the lighter are higher elevation. You notice the sharp contrast between the purple and pinkish colors – this represents a rapid change in elevation as can be seen when walking toward the parking lot to the north of the dorms and looking out toward Interstate 35.

Slide 6

Setting up a Simulation Run



First steps...

1. Move your intended burn area square to the desired location by clicking in the yellow square and dragging it. To zoom in, double-click between the yellow and pink boxes. To zoom back out, double-click again between the yellow and pink boxes.
2. Click on the compass to set your wind direction and speed. The farther out from the center of the compass you click, the higher the wind speed will be. An arrow will appear in light pink on your image showing the direction and intensity of the wind.
3. Select your Fuel Water (fuel moisture) by clicking on the slider beneath the image. The default is 10%, which is a good estimate for the area shown.

During the set-up, feel free to click around and experiment with your parameters; look at the different images available; play with the interface. It will help you become more familiar with the capabilities of the simulation.

Slide 7

Setting up a Simulation Run

Wildfire Simulator ([background](#))



24.96 m/sec

Directions

Double-Click between pink and yellow boxes to zoom.

Set wind vector by clicking on the compass.

Switch between Image, fuel categories, and terrain by clicking their buttons.

Set fire burn line by zoom into a location then click and drag within pink box.

To run, enter key, pass, and comment, then click Submit job.

First steps...

4. To create your fire burn line, which is the line from which your fire begins, you will need to be zoomed in on your intended burn area. Click on a point inside the pink box and drag to create the line, staying within the pink box.

In the image above, you see the high wind speed of 24.96 meters per second, blowing toward the east south east. What do you think these wind conditions will result in?

Slide 8

Setting up a Simulation Run

Wildfire Simulator ([background](#))



24.96 m/sec

Directions

Double-Click between pink and yellow boxes to zoom.

Set wind vector by clicking on the compass.

Switch between Image, fuel categories, and terrain by clicking their buttons.

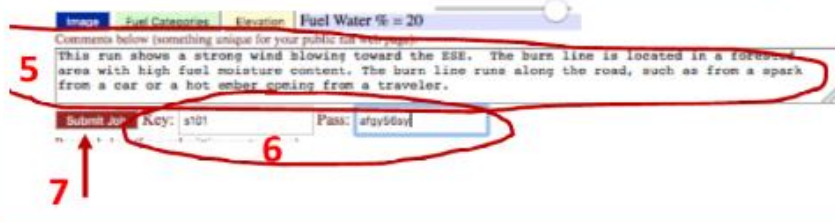
Set fire burn line by zoom into a location then click and drag within pink box.

To run, enter key, pass, and comment, then click Submit job.

First steps...

5. Enter in comments about the parameters you've selected for your run. Be specific about your conditions and your thoughts. The run will take 15-30 minutes once it is first in the queue, so you may have to come back to it later if there are other jobs waiting in line to run before yours. Your comments will help you remember what conditions you selected, and why!

6. Enter one of the Key and Pass codes that you received. The Key begins with an "s" and is followed by 3 numbers. The Pass is an 8-character code provided after each Key code. Each Key and Pass combination is unique and can only be used one time.



7. Click Submit Job! Slide 9

Running a Simulation Run

Once you click Submit Job, a report message will appear with code that contains the conditions you've selected. Unless this results box reports an invalid Key or Pass code, you can move on!

Submit Job Key: s630 Pass: db26hc6b

Results below (from submitting run to queue):

```
REPORT=:pass=db26hc6b:inpass=db26hc6b: time= 1493658079 qfile= /var/www/q2/1493658079_s630
water=20 key=s630 pass=db26hc6b utmx=195 utmz=89 ux= 22.5 uz=10.8 sx=2 sz=21 ex=2 ez=7
phrase=:This run shows a strong wind blowing toward the ESE. The burn line is located in a forested area
with high fuel moisture content. The burn line runs along the road, such as from a spark from a car or a hot
```

View Queue View Runs View Old Runs (WRFFire)

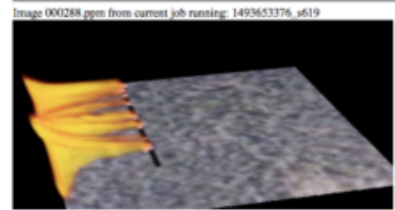
8 ↑

Checking on your run (next steps)...
 8. Once you have submitted your job, you can view where you are in the queue by clicking View Queue. All jobs that have been submitted will appear in the order by which they were submitted.

Fire Run Queue Status

There are currently 6 jobs on the queue to run. Jobs from first to last on the queue (green ranning):

```
1: 1493653376_s619
2: 1493653666_s629
3: 1493653799_s622
4: 1493658079_s630
5: 1493658651_s630
6: 1493658875_s655
results= im=000288 ppm
```



Click your browser reload button to refresh the queue and current image.

Slide 10

Running a Simulation Run

Submit Job Key: s630 Pass: db26hc6b

Results below (from submitting run to queue):

```
REPORT=:pass=db26hc6b:inpass=db26hc6b: time= 1493658079 qfile= /var/www/w
water=20 key=s630 pass=db26hc6b utmx=195 utmz=89 ux= 22.5 uz=10.8 sx=2 s
phrase=:This run shows a strong wind blowing toward the ESE. The burn line is l
with high fuel moisture content. The burn line runs along the road, such as from a
```

View Queue View Runs View Old Runs (WRFFire)

9 ↑

Checking on your run (next steps)...
 9. When your run is complete, you can see it when you click View Runs. It appears as a folder in the directory. Clicking on your Key will bring you to snap shots of the run, a summary of the conditions and run time, your comments, and a link to a short video of your run.

Index of /runs2

Name	Last modified	Size	Description
Parent Directory	-	-	-
current.jpg	2017-05-01 12:20	19K	
s002/	2017-03-13 12:16	-	
s003/	2017-03-13 12:48	-	
s004/	2017-03-13 16:20	-	
s005/	2017-03-13 16:55	-	
s006/	2017-04-25 07:39	-	
s007/	2017-04-26 07:08	-	
s140/	2017-04-24 23:22	-	
s616/	2017-05-01 11:16	-	
s619/	2017-05-01 12:21	-	
s621/	2017-05-01 11:54	-	

Fire run s140 : Mon Apr 24 22:55:24 2017



10. Once you have completed 5-10 runs, compose a summary as described in the simulation directions.

Slide 11

Expanding HLRS Academic HPC Simulation Training Programs to More Target Groups

Tibor Döpfer¹
doepfer@hlrs.de

Bärbel Große-Wöhrmann¹
grosse-woehrmann@hlrs.de

Doris Lindner¹
lindner@hlrs.de

Darko Milakovic¹
milakovic@hlrs.de

Jutta Oexle¹
oexle@hlrs.de

Michael M. Resch¹
resch@hlrs.de

Oliver Scheel¹
scheel@hlrs.de

Sven Slotosch²
sven.slotosch@rz.uni-freiburg.de

Leon Widmaier²
leon.widmaier@rz.uni-freiburg.de

ABSTRACT

For a long time, high-performance computers and simulations were of interest only at universities and research institutes. In recent years, however, their application and relevance in a wider field has grown; not only do industry and small and medium-sized businesses benefit from these technologies, but their social and political impacts are also increasing significantly. Therefore, there is an increasing need for experts in this field as well as better understanding of the importance of high-performance computing (HPC) and simulations among the general public. For this reason, the German National Supercomputing Center HLRS has broadened its academic training program to include courses for students and teachers as well as for professionals. Specifically, this expansion involves two projects: “Simulated Worlds,” which offers a variety of educational programs for middle and high school students, and the “MoeWE” project with its “Supercomputing Academy” for professionals. These projects complement the center’s academic educational focus by addressing the special needs of these new target groups who have otherwise not been able to benefit from HLRS’ academic training program. In this paper, we present background concepts, programmatic offerings, and exemplary content of the two projects; discuss the experiences involved in their development and implementation; and provide insights that may be useful for improving education and training in this area.

Keywords

Supercomputer, HPC, Simulation, Numerical methods, Machine learning, Demand oriented school events, Blended learning, High school students, Professionals

1. INTRODUCTION

Computer simulation has become an indispensable tool in most fields of science and technology, while simulation results increasingly influence decision making in business, politics, and

society. Taking advantage of the opportunities that simulation offers requires not just numerous experts in simulation techniques and high-performance computing (HPC) but also a broad awareness of the nature of simulation in the general public.

While simulations and HPC have been used primarily for academic research in past decades, their impact in industry is now growing as well. In addition to traditional HPC using supercomputers, new topics such as high performance data analytics (HPDA), artificial intelligence and machine learning, and the comprehensive digitalization of industrial production have also been attracting enormous interest. For this reason, there is an increasing need, especially in small and medium-sized enterprises (SMEs), for targeted training of employees.

As a consequence, demand for trainees, students, and young professionals with experience in computing, simulation, and mathematical modeling is also growing. In order to bring young people into contact with the subject as early as possible, the topics of simulation and mathematical modeling are included in German school curricula although for several reasons learning materials and teaching methods have only recently become available. In this context, the teaching content on these topics should complement school experiments in the classroom with the help of computers to convey scientific insight in a vivid way.

In Europe, many HPC centers offer education and training. PRACE’s³ European training courses were designed for the classroom and are open to participants from industry but are primarily geared to the needs of academic participants. The training courses of the 15 European Centers of Excellence for HPC applications⁴ and the National Competence Centers (NCCs)⁵ in HPC include both face-to-face and online courses and are generally aimed equally at academic participants and participants from industry but often cover specialized topics. The NCCs, which are being established by the EuroHPC Joint Undertaking within the framework of the EuroCC and CASTIEL⁶ initiatives, focus on science, public administration, and both large industry and small and medium-sized enterprises.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

© 2021 Journal of Computational Science Education
DOI: <https://doi.org/10.22369/issn.2153-4136/12/3/2>

¹ HLRS, University of Stuttgart, Stuttgart, Germany

² Department for E-Learning, University of Freiburg, Freiburg, Germany

³ <https://training.prace-ri.eu/>

⁴ <https://www.hpccoe.eu/eu-hpc-centers-of-excellence2/>

⁵ <https://www.eurocc-access.eu/>

⁶ <https://www.eurocc-access.eu/the-projects/>

There are also numerous providers of online education and training for academics in the IT sector, but only a few specialize in HPC and data science. For example, the University of Edinburgh⁷ and the Johns Hopkins Whiting School of Engineering in Baltimore⁸ offer master's degree programs, and the University of Waikato in New Zealand⁹ specializes in massive open online courses (MOOCs).

As commercial online education providers in HPC and data science, Udemy¹⁰ and Udacity¹¹ offer training videos, the number of participants is unlimited, and the courses are not supervised. NAFEMS¹² organizes web seminars, mainly on simulation.

In Germany, the School of Advanced Professional Studies (SAPS)¹³ at the University of Ulm and at the Ulm University of Technology has built a tiered online continuing education program for SMEs as part of its “Data Literacy and Data Science” program, and the Hasso Plattner Institute offers free MOOCs and discussion forums on openHPI¹⁴.

One of the leading HPC centers in Europe, the High Performance Computing Center Stuttgart (HLRS) at the University of Stuttgart, has been operating supercomputers and conducting research in simulation and HPC since 1986. It became Germany's first national supercomputing center in 1996 and is one of the three members of the Gauss Centre for Supercomputing (GCS)¹⁵. GCS has played a leading role in the establishment of the German National Competence Centers of HPC.

HLRS offers services and support in the field of supercomputing to users from science, research, and industry. It is one of the world's leading research facilities in the fields of supercomputing, data analysis, and cloud computing; participates in the Cluster of Excellence for Simulation Technologies (SimTech) at the University of Stuttgart; and conducts innovative research and solution-oriented technology development.

To meet the increasing demand for HPC expertise, HLRS has offered a successful education program for more than two decades. The program includes classroom courses and workshops on all HPC-relevant topics and is part of national (GCS) and European training programs (PRACE).

Due to their academic focus, however, most HPC training programs do not reach young people and only occasionally reach working professionals. HLRS has filled these gaps by launching two projects. The first, called Simulated Worlds (“Simulierte Welten”), targets students, teachers, and the interested public. It creates awareness of HPC and simulation concepts by implementing the topics into the State of Baden-Württemberg's science, technology, engineering, and mathematics (STEM) curricula.

The second project, “Modular Continuing Education to become an HPC Expert” (MoeWE), established the Supercomputing Academy (“Supercomputing-Akademie”) for working professionals. Unlike the above-mentioned offerings, Supercomputing Academy courses are designed in a blended learning format that combines attendance

days, self-learning phases, supervised exercises, and web seminars. The Supercomputing Academy has become an important component in providing modular continuing education for industry within the framework of the German National Competence Center.

The expansion of HLRS' training program to address these two additional target groups is outlined in Figure 1. Both Simulated Worlds and the Supercomputing Academy focus on the topics of simulation and HPC and benefit from the expertise of HLRS experts. Project teams also designed appropriate programs following communication with experts from industry and schools about needs and expectations. Existing content from academia could also be used for the new target groups but was adjusted in depth and degree of abstraction to suit their needs and preexisting knowledge. Information is also presented using appropriate methods for each case.

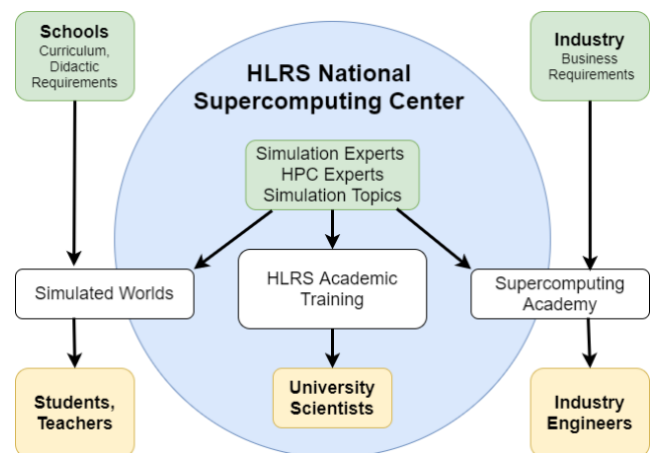


Figure 1. Training activities at the High Performance Computing Center Stuttgart.

In the Section 2, the concepts behind the two programs are presented in detail. Then, in Section 3, the contents of the programs are outlined. In Section 4, the concrete implementation in practice is presented with examples. The conclusion reports on the most important evaluation results and offers an outlook on the further development of both programs.

2. THE CONCEPTS OF THE TWO PROGRAMS

Consideration of the specific needs of the two target groups in the development of methods and content used is crucial for the long-term success of both projects, Simulated Worlds and the Supercomputing Academy (MoeWE).

In Sections 2.1 and 2.2, we present the motivations and the process of concept development for Simulated Worlds and the Supercomputing Academy. The goals and special features of each program are described and then compared in Section 2.3. Both projects are united above all by the goal of bringing the topics of HPC and simulation to society. The different target groups—teachers and students on the one hand and professionals on the other—result in different offerings that nevertheless have commonalities. Table 1 in Section 2.3 provides further information on both programs.

2.1 Simulated Worlds—Events for Schools

Simulated Worlds has been working since 2011 to bring as many students as possible in the German State of Baden-Württemberg

⁷ <https://www.ed.ac.uk/studying/postgraduate/degrees/index.php?r=site/view&edition=2021&id=997>

⁸ <https://engineering.jhu.edu/ams/data-science-masters-program/>

⁹ <https://www.futurelearn.com/courses/advanced-data-mining-with-weka>

¹⁰ <https://www.udemy.com/course/deep-learning-und-ai/>

¹¹ <https://www.udacity.com/course/data-scientist-nanodegree--nd025>

¹² <https://www.nafems.org/training/>

¹³ <https://www.uni-ulm.de/en/einrichtungen/saps/projekte/data-literacy-und-data-science/>

¹⁴ <https://open.hpi.de/courses>

¹⁵ <https://www.gauss-centre.eu/>

into contact with the topic of simulation at least once in their school careers. It also provides teachers in Baden-Württemberg with suitable teaching materials and corresponding training.

When developing a suitable educational enrichment concept, it is important to take characteristics of the relevant school system into consideration. In Germany, after completing four years of common primary education, students go to one of three types of secondary school. The first, called the *Gymnasium*, corresponds to Level 3 “upper secondary education” of UNESCO’s International Classification of Education (ISCED) [15] and aims at preparing students for college and university careers, focusing on logical and analytical skills. For students from grade 6 upwards who attend this type of school, *Simulated Worlds* has offered a variety of courses since 2011 (presented in an overview in Section 3 and deepened with two detailed examples in Section 4).

The second type of school, called the *Realschule*, teaches more practically applicable knowledge for professional practice. *Simulated Worlds* is currently developing content adapted to this curriculum in cooperation with a pedagogical university. The third type of school, the *Hauptschule*, is less suitable for the thematic goals of the project as its teaching of basic educational content is even less oriented toward the abstract concepts typical of computer science.

Originally, *Simulated Worlds* was intended to raise awareness of the science of HPC. The elaboration of this idea was carried out with the help of focus groups involving students and teachers who provided input that was used to concretize program goals and implementation strategies. Focus groups rely on a moderated process in which small groups discuss a particular topic in order to articulate as many facets as possible about the topic [19]. In the *Simulated Worlds* focus groups, participant input resulted in a shift of emphasis from HPC to simulation, a term that would be more descriptive and meaningful for students. Based on the input of the focus groups, the project team expanded and developed flexible, directly applicable and easily adaptable programmatic offerings with a thematic focus on simulation. The project was funded by the Ministry of Science, Research and the Arts of the State of Baden-Württemberg and started in 2011 on a small scale with HPC centers in Stuttgart (HLRS) and Karlsruhe (Steinbuch Center for Computing at the Karlsruhe Institute of Technology). Shortly thereafter, the program launched in cooperation with two model schools. Over the years, they were joined by the HPC Center Ulm (Communication and Information Center of the University of Ulm) and two more model schools. The project team is now composed of interdisciplinary social scientists, mathematics educators, computer scientists, and HPC Ambassadors; full-time teachers in the secondary education sector (*Gymnasium*) who provide expert insights to the project for three to five hours per week, based on their daily teaching experience.

This multidisciplinary approach connects different areas of knowledge; while scientists at the computing centers develop initial ideas and concepts that are suitable for teaching to students, the HPC Ambassadors advise them on the development of new teaching materials and check content for its compatibility with the official school curriculum and the corresponding knowledge level of the students. The HPC Ambassadors also test the materials in the classroom, present them at other schools, and help to keep the program’s offerings up-to-date and thus interesting for students. They are also multipliers of the offerings and help build networks among different teachers and schools.

For teachers, teaching material is particularly interesting if it can be used to fulfill curriculum requirements easily. For example, the STEM subjects in high schools require that 10th grade students “[...] develop systems in increasingly complex contexts and [...] know about the dynamics and interactions in these systems” [14]. For this purpose, they should, for example, understand the importance of “model building and simulation” and how the computer is used as a tool, for example, to simulate dynamic systems [14]. Teachers might have issues with the curriculum because precise instructions, didactic approaches, or examples of simulations are not very concrete. *Simulated Worlds* addresses the curriculum and the challenges by elaborating material with practical help and develops concrete teaching materials that can be booked by schools in various offerings (see Section 3.1).

Compared to other digital tools, didacticians attested that learning about simulation provides the greatest increase in students’ competencies in digital literacy [1]. In addition, simulations can help students visualize a wide range of topics, both in terms of content (e.g., climate simulations, energy transition) and science and mathematics (e.g., growth models, stochastics). They facilitate the recognition of interrelationships and offer alternative ways of representation (e.g., through various graphics) [7]. Furthermore, mathematical modeling allows students to work on problems that are close to their everyday life [8].

Other projects with a similar goal to that of *Simulated Worlds*, for example, try to bring students to a standard level of knowledge in information technology [4]. It is more beneficial if the first, possibly uniform experiences are created much earlier, namely at school. International projects that target schoolchildren try to familiarize them with programming and computer science [5, 6].

Simulated Worlds complements other existing, wide-ranging offerings for students in STEM fields. What distinguishes it, however, is its focus on the complex themes of simulation, HPC, and mathematical modeling. Computers are used as tools, while programming takes a back seat. At the same time, the project implements the curriculum of the schools and thus also addresses their needs. *Simulated Worlds* supports teachers, enabling them to implement the topics actively and independently in the classroom.

Based on these requirements, each of the offerings of *Simulated Worlds* includes answers to the following questions:

- Where can scientific simulations be found in everyday life? In which decisions do simulation results have an impact and are thus politically, economically, and socially relevant?
- Why do scientists carry out simulations?
- How do scientists carry out simulations?
- What are the opportunities and risks associated with simulations?

There is a pool of examples that can be used to flexibly embed these questions into the respective teaching topics. Examples include simulations of heat conduction, human behavior during panic, the spread of viral diseases, traffic patterns, or the shift to renewable energy sources (see Table 1, Section 3.1).

In order to make the principles of scientific simulation, supercomputing, and mathematical modeling accessible to students and teachers, the project team primarily offers free classroom events that can be held either in schools or at computing centers. Depending on the location, different emphases are then set (see Section 3.1). Similarly, the duration and level of detail of the face-to-face offerings is very scalable, ranging from 90-minute school

double lessons to half-day workshops, such as the so-called do-IT day, to a six-month scholarship. Section 3.1 describes these offerings and their content in more detail, and Section 4 uses concrete examples—simulations of heat conduction and a commuter train system—to show how young people can try things out for themselves in the do-IT day and in the scholarship.

2.2 Supercomputing Academy—Blended Learning for Professionals

Since April 2018, the Supercomputing Academy¹⁶ has been offering professionals a part-time, modular, continuing education program that enables them to develop expertise in HPC. The program is aimed at computer scientists, engineers, and IT-oriented career changers. The focus is on employees of small and medium-sized enterprises (see Table 1, Target groups). The aim is to enable participants in the Supercomputing Academy to independently develop solutions for problems in the field of HPC after qualification.

The continuing education program is organized in a blended learning format. The combination of online and offline phases allows maximum time flexibility and location independence, making it possible for participants to learn at an individual pace whenever and wherever they find time alongside work and family commitments. Weekly virtual seminars focusing on previous learning units provide some structure to the learning process and enable learners and teachers to exchange views on the learning material from the previous week. The assignments and exercises are discussed intensively and help the learners to keep up with the curriculum.

Each module (see Figure 3) starts and ends with an attendance day, where speakers from industry and HLRS introduce or conclude the module topics with technical presentations. All learning materials are accessible via an open-source learning management system, the Integrated Learning, Information and Work Collaboration System (ILIAS)¹⁷. Practical exercises (hands-on) that require computing resources are conducted on the HPC training cluster, a scaled-down version of an HLRS production system. Each participant who has completed the exercises and virtual seminars contained in a course module receives a qualified certificate of participation. Following the successful completion of a test, graduates receive a certificate of achievement.

Experts at HLRS are familiar with current scientific developments in HPC and simulation. However, in order to incorporate the viewpoint of industry—the target group of this program—into the continuing education program, and to enable a transfer of knowledge into daily work, identifying the needs of businesses and addressing them in the curriculum of the continuing education program is of vital interest. Roundtable discussions involving HLRS experts, scientists, and industry experts—so-called expert workshops—as well as interviews with industry experts were held to determine the need for HPC know-how in industry. In addition, the requirements, topics, and learning objectives were outlined. The identified requirements form the basis for the learning content,

which is created and continuously developed by HLRS experts. The module contents cover all HPC-relevant topics (see Table 1, Modules).

The competencies to be acquired are formulated in concrete, action-oriented, and verifiable terms and translated into learning objectives [12, 17]. Bloom's taxonomy of learning objectives is used for operationalization [2, 3]. Learning objectives have different functions. They guide teaching and learning processes in terms of content. They help to reflect and plan the instructor's intentions. Finally, they also serve to review the achievement of objectives and to clarify the way forward [18].

The “3-2-1 model of didactic elements in hybrid learning arrangements” [11] is used to develop the structure and content of each module. This model provides guidance for the organization and design of learning arrangements in a blended learning environment. It can be used to identify three mandatory elements, two supplementary elements, and one optional element, as described below.

The content of the three mandatory elements is determined by the learning objectives. Orienting information for each module and for the lessons within a module are provided on the learning platform. They are supplemented by learning control tasks and enriched by exercises. The exercises require the participants to apply the acquired knowledge, to demonstrate mastery of it, and to use it in a new context. Thus, the exercises promote the ability to transfer and apply what has been learned to other situations [16]. Once participants have completed all the exercises, they receive a qualified certificate of participation at the end as proof of the work they have done.

During the online phase, learning is self-directed in terms of time, scope, and speed of the learning process. It promotes self-regulation, and metacognitive processes such as monitoring and regulation can be stimulated [16].

The two supplementary elements, communication and cooperation, are established at a social event organized on the face-to-face day that gives participants a chance to meet one another and to make first contact with the instructors. In the weekly virtual seminars, the communication element is strongly emphasized. The lecturers reflect on the learning material of the previous week and give the participants the opportunity to ask questions and share and discuss problems. The discussion session is moderated by the module supervisors. Opportunities for cooperation are provided via an online forum. Here, participants have the opportunity to support each other when problems arise and to help their partners with suggestions for solutions. These dialogical and dialectical interactions can increase motivation and support the acquisition of knowledge and skills [22].

The optional element in the modules is the test. It takes place on the last attendance day of each module. Participants who complete the exam successfully receive a certificate as confirmation.

To increase motivation for the test and to recognize achievements, four awards have been introduced:

- HPC User
- HPC Developer
- HPC Administrator
- HPC Expert

Participants who have successfully completed three modules and the corresponding tests receive an award reflecting their chosen direction (HPC user, developer, or administrator). Participants who

¹⁶ The universities of Stuttgart, Freiburg, and Ulm in Germany and SICOS BW GmbH have joined forces in the MoeWE project to found and establish the “Supercomputing Academy” based at HLRS. The accompanying project MoeWE (2016–2021) was funded by the Ministry of Science, Research and the Arts of the State of Baden-Württemberg and the European Social Fund (ESF).

¹⁷ <https://www.ilias.de/en/>. ILIAS is widely used in Germany as a learning management system and is constantly being further developed by the ILAS community.

successfully complete two more modules receive the HPC Expert award (see also Table 1, Award). More detailed information about the modules and awards can be found in Section 3.2.

The effort required to set up this continuing education program amounted to six and a half full-time positions during the entire four years. Coordination, didactics, conception, production, and implementation required about 3/4 of the positions. Dissemination and contact management, acquisition, and public relations accounted for 1/4.

2.3 Profile of Both Programs

Simulated Worlds and the Supercomputing Academy are united above all by the goal of bringing the topics of HPC and simulation to society. The different target groups—teachers and students on the one hand and professionals on the other—result in different offerings that nevertheless have commonalities. Table 1 summarizes both programs.

3. CONTENTS OF THE TWO PROGRAMS

In the previous section, the concepts of Simulated Worlds and the Supercomputing Academy were presented. Both project teams developed their offerings using an exploratory approach in which they directly involved the respective target groups through focus groups and expert workshops, respectively.

Section 3.1 will provide overviews of their content. Two examples of the contents are then explained in detail in Section 3.2.

3.1 Contents of Simulated Worlds

In order to reach as many schools as possible in Baden-Württemberg, diverse formats with varying degrees of difficulty are necessary. The offerings of Simulated Worlds are based on the so-called “knowledge mountain” (see Figure 2) according to which the formats can be assigned to three knowledge levels; the first is called “Basic Knowledge,” the second “Exemplary Knowledge,” and the third “Detailed Knowledge.” All formats can be offered to interested schools individually as they are independent of each other in terms of prior knowledge. The formats differ in duration, depth of knowledge, and target group (see Table 1, Target group). All three levels of knowledge are explained here with examples.

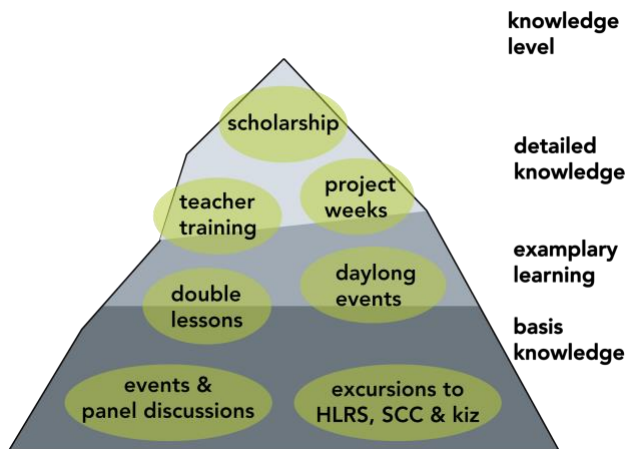


Figure 2. “Knowledge Mountain”—Locating the offerings of Simulated Worlds.

Table 1. Comparison of the programs of Simulated Worlds and Supercomputing Academy.

Simulated Worlds	Supercomputing Academy
Objective	
To impart knowledge about HPC and simulation	To impart knowledge about HPC and simulation
Target group	
Teachers and students (especially at high schools), interested public	Working professionals: System administrators, software developers, users from industrial and academic fields
Offerings / Modules	
Topic-specific offerings of different duration: panic simulation, spreading of diseases, do-IT day on heat conduction, bridge simulation, crane simulation, etc.	Advanced training to become an HPC expert: Parallel Programming, Simulation, HPC cluster, Performance Optimization, Economics and Sustainability, Visualization, Data: Management & Analysis, Basics
Formats	
Attendance events: Double lessons, lectures, day sessions, advanced training, project week, scholarship	Blended Learning: Face-to-face days: kick-off & closing events; online phase with weekly virtual seminars.
Self-study	
In the scholarship besides meetings with supervisors	During online phase
Duration per module	
Depending on module: 90 min (double hour) to 6 months, each 3 h per week (scholarship)	Per module: 12 weeks, each 10 h per week
Certification / Award	
Certificate at the end of the support scholarship Certificate of participation for teachers after advanced training	Certificates Qualified certificate of participation Awards: HPC administrator, HPC user, HPC developer (three modules) HPC expert (five modules)
Learning progress documentation	
Learning journal in the scholarship	ILIAS: Learning progress tracking + learning control questions

3.1.1 “Basic Knowledge”—Lectures and Excursions to Computing Centers

The events that belong to the Basic Knowledge category serve as an introduction to the topic and are offered at many schools. For students and teachers, these classroom events are their first contact with the topic of simulation.

Lectures by HPC experts on the topics of simulation and HPC, which can be booked by schools, are popular. Participants learn about current scientific projects and can ask questions about the state of research and engage in conversations with scientists. The HPC experts are informed about the level of knowledge of the classes before the lecture so they can better adapt the level of the lecture to the school classes (grades 8 to 10).

In addition, school classes can visit the HPC centers and receive a guided tour of the computer hall. Additionally, at HLRS, the CAVE can be toured—a walk-in projection chamber where groups of people can experience 3D visualizations simultaneously.

These offerings for schools are complemented by evening lectures for the interested public that are organized by the project team. Here, simulation experts provide insight into current research topics and answer questions from participants. Thus, although the focus of Simulated Worlds is on teachers and students, addressing interested members of the public is a supplementary activity.

3.1.2 “Exemplary Knowledge”—Double Lessons and Daylong Events

This level of knowledge transfer combines academic science with everyday examples to teach simulation in a comprehensible way. To promote deeper understanding, a hands-on phase is always integrated, in which students run a simulation themselves. Here, Simulated Worlds again offers two formats: double lessons and do-IT days.

In a double lesson, a member of the project team introduces a specific topic on simulation, e.g., the spread of a disease or the development of a panic situation in a nightclub. A short lecture outlines what a simulation is, how it works, and what risks are associated with it. Afterwards, in the hands-on phase (approx. 45 minutes), the students are allowed to run a simulation themselves using a program such as NetLogo¹⁸. This offering is designed for school classes from grades 7 to 10. The level of difficulty varies depending on the age of the students.

The do-IT day is an approximately four-hour workshop, in which students gain deeper insights. The workshop provides a brief overview of computer-aided simulation and its many possible applications in the worlds of work and science. Due to the more complex requirements in the hands-on phase (e.g., programming in Python), students in grades 9 through 11 are targeted here.

Simulations are introduced as the fundamental link between theory and experiment in science. The following form of the simulation cycle is explained to them:

1. In modeling, parts of reality or a physical state are described mathematically.
2. To obtain a prediction of the future state of the object, mathematical equations are solved in complex or repeated cases with the computer.
3. The result of the calculation is compared with the observed reality.
4. Relevant differences lead to a change in the modeling. Thereupon, the cycle is restarted to improve the match.

Further details on the tasks of the do-IT day can be found in Section 4.

¹⁸ For all programs, the project team makes sure to use open-source programs to minimize the financial burden on schools and enable them even to host such an event again themselves.

Link: <https://ccl.northwestern.edu/netlogo/>

3.1.3 “Detailed Knowledge”—Scholarship, Project Weeks, Internships, Training for Teachers

The aim of Detailed Knowledge is to provide students and teachers with in-depth insights into the topics of simulation and HPC. Students, in particular, deal intensively with a specific problem over a longer period of time and try to solve it independently. Teachers are provided with extensive learning and teaching material so that they can implement the topic of simulation themselves in the classroom.

Students in the 11th grade with a special interest in STEM fields can receive a scholarship at a high-performance computing center. The scholarship lasts about six months, during which approximately ten students spend three to four hours a week working on a topic of their choice in addition to their classes. The topics are provided by employees of the respective computing centers.

The aim of the scholarship is to give the students insights into the work of a high-performance computing center. In the process, they also learn how to work scientifically. The work in the scholarship is primarily characterized by independent work on a project. This includes research as well as familiarization with new programs, e.g., by means of tutorials.

Organizationally, there are three mandatory, official meetings with all students: a kick-off meeting; an intermediate meeting to address problems and present the current status of their projects; and a final, official presentation of their results to a public audience (parents, friends, teachers, etc.).

During the scholarship, students keep a learning journal [21] in which they record their assignments, their positive and negative experiences, and how they overcame challenges. Students are guided by questions that provide direction. At the end, they draw a personal conclusion. The main goal of the learning journal is for students to reflect on their work experiences and their methods of problem solving. This experience is generally transferrable to the classroom, where students also use the skills they have acquired during the scholarship. At the same time, this experience should prepare them for future working situations, e.g., in their studies at university. A concrete example of a scholarship can be found in Section 4.

Another option for delivering Detailed Knowledge occurs during project weeks for students held at computing centers. Here, too, particularly interested students from grades 10 and 11 can work on a scientific problem. An additional option is through internships, where 9th or 10th grade students are asked to spend a week doing an internship at a company. Students spend time in a department at a supercomputing center and are assigned to a specific project, gaining insight into the process and tasks that projects entail as well as general information about the workings of a data center.

The fourth offering created to deliver Detailed Knowledge is aimed at teachers in STEM fields. Here, advanced training courses are offered, which are conducted by HPC Ambassadors together with scientists. In these one-day or multi-day training courses, participants receive an introduction to simulations and programming with a programming language suitable for students. Procedures and teaching materials are presented as examples so that teachers can independently integrate the topic of simulation into their lessons.

3.2 Contents of the Supercomputing Academy

The Supercomputing Academy has divided its offerings into seven modules. During the expert workshops with HLRS and industry experts, three main target groups were identified for the Supercomputing Academy. These are users of simulation programs, developers of simulation programs, and IT administrators interested in HPC (see also Table 1). Professionals in all three target groups can earn the corresponding awards already mentioned in Section 2.2 if they successfully complete the associated modules plus one additional module of their own choosing including the related tests. The seven modules are shown in Figure 3 in combination with the awards to be achieved. Their contents are briefly presented below.

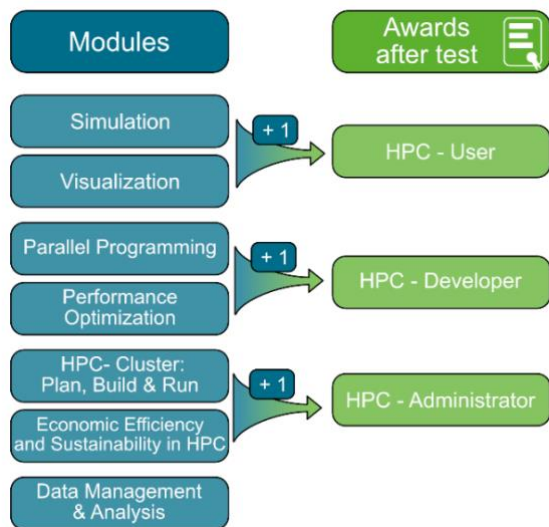


Figure 3. The modules and their assignment to the HPC User, HPC Developer, and HPC Administrator awards. “+1” denotes an additional module of free choice.

The Simulation and Visualization modules are primarily aimed at users of simulation programs.

The Simulation module is one of the central modules of the Supercomputing Academy. Its first learning unit presents the entire simulation cycle while focusing on the heat conduction equation. In addition to providing an introduction to the topic, the purpose of this learning unit is to establish a common knowledge base among participants (see Section 4.1.2 for more details). In the subsequent learning units of the Simulation module, the fundamentals of computational fluid dynamics (CFD) and structural mechanics are provided. The Navier-Stokes equations and the equations of structural mechanics are introduced, and their discretization methods are explained. Finite difference (FDM), finite element (FEM), and finite volume (FVM) methods are introduced with simple examples and practiced using provided implementations.

Subsequently, the numerical methods used in the simulations—primarily methods for solving systems of linear equations—are covered with emphasis on HPC-specific aspects, tested on the HLRS training cluster, and compared. The topics covered in this module are completed by learning units on particle-based methods and on molecular dynamics simulations as well as a discussion of the societal impact of simulation.

The module Simulation is followed by the module Visualization that consists of three topics. In the first topic area, “Introduction

and Motivation,” students learn about the many uses of visualizing measured data and data computed in simulations in combination with virtual reality (VR) and augmented reality (AR).

In the second topic area, “Virtual and Augmented Reality in Technical-Scientific Visualization”, topics such as human perception, colors and vision, graphics hardware, and interfaces are covered in depth. Finally, in the third topic area, “Interactive Remote Visualization on Clusters,” interactive exercises using the open-source programs ParaView¹⁹ and COVISE²⁰ for technical scientific visualization are conducted on the HLRS training cluster.

Completion of the Simulation and Visualization modules, as well as one additional module of free choice, is mandatory for earning the HPC User award (see also Figure 3).

For software developers who want to parallelize and optimize numerical methods and simulation programs, the Supercomputing Academy offers the modules Parallel Programming and Performance Optimization.

After an introduction to parallel thinking and the parallelizability of algorithms and software, participants in the Parallel Programming module learn the central concepts and commands of OpenMP and practice them using simple programs. This is followed by learning units on MPI for beginners and advanced users including exercises focusing on parallel I/O and hybrid programming. Since the focus of the module is on the use of software on HPC clusters; the topics of scalability, domain decomposition, and load balancing as well as parallel design patterns, heterogeneous computing systems, and graphics cards are also covered.

High-performance computing is a matter not only of running parallelized software on HPC clusters and supercomputers but also of emphasizing its efficiency, i.e., its optimization for the available hardware. Participants in the Performance Optimization module, another central module of the Supercomputing Academy, learn about this.

The Performance Optimization module consists of four parts. In the first, the basics regarding CPUs and the parallel programming models MPI and OpenMP are provided, also serving as prerequisites for understanding the following parts. The second part introduces the methodology of performance optimization.

In the third part, “Node-Level Performance Optimization”, important aspects of performance optimization at the compute node level are introduced including vectorization and the importance of efficient data transport between main memory, caches, and the processor. After the presentation of modern processor and memory microarchitectures, the functionality of the AVX units and central processes such as memory allocation are analyzed in provided benchmarks on the training cluster. The fourth part, “Optimization of Communication,” deals with the analysis and optimization of data transport between computing nodes.

Participants must successfully complete the Parallel Programming and Performance Optimization modules—in addition to a third module of free choice—to receive the HPC Developer award (see Figure 3).

For system administrators—and for users and software developers with an interest in building HPC clusters—the module HPC Cluster: Plan, Build, Run is suitable. Its focus is on hardware. In

¹⁹ <https://www.paraview.org>

²⁰ <https://www.hlrs.de/covise>

the first part, “Fundamentals,” participants learn about the main components of any HPC cluster: compute nodes, a high-performance data network, and data storage. In addition, information is provided on operating systems, compilers, libraries, and software management. In order to be able to discuss performance aspects with software developers, administrators also learn basic performance facts.

Building on these fundamentals, the second part of the module involves designing an HPC cluster on paper and learning about aspects of its technical integration into a company. Additional knowledge about batch systems, monitoring, and IT security that are necessary for the integration of the cluster into other IT systems and for its operation are taught in the third part of the module. The most important matters concerning virtualization, containerization, and cloud computing are compiled in the fourth part.

Business issues surrounding HPC clusters are the subject of the module, “Economics and Sustainability,” designed for IT administrators. This module looks at business management and environmentally sustainable aspects of HPC and explains why business viability and sustainability are not contradictory. The central question of this module is how the goals of efficient computing and sustainability can be achieved simultaneously.

In the topic area, “Economic Efficiency,” the basics of investment decision-making and cost accounting are introduced, as well as multi-criteria decision support. The knowledge acquired is then applied to the HPC operating models “in-house cluster” and “cloud utilization” as well as mixed forms of the two approaches. The basics of these operating models and the associated costs and dependencies are explained.

The module also provides an introduction to the issue of sustainability. This includes providing insights into environmental certifications according to ISO 14001 and ISO 50001 as well as the standards EMAS²¹ and the Blue Angel²² for energy-efficient data center operation. Other important topics include power supply and efficient cooling as well as sustainable procurement and end-of-life management. Participants learn how closely profitability and sustainability are linked in the various practical exercises.

Successful completion of the HPC Cluster: Plan, Build, Run; Economics; and Sustainability in HPC modules and one elective module are required for the HPC Administrator award.

The seventh module, “Data: Management and Analysis,” is of interest to all three target groups and is a very popular elective module. It consists of two topic areas. The first topic area, “Management,” contains an introduction to data and familiarizes participants with the basics. They are also introduced to the technical basics, ontology, and model design of metadata and methods and tools for handling data in repositories. Finally, the topics of dark data and data accountability are addressed.

The second topic area focuses on the analysis of large amounts of data using artificial intelligence and HPC. For this purpose, current container technologies, database technologies, and explorative data analysis are introduced. Finally, the basics of machine learning are covered, introducing participants to the topic of neural networks and deep learning on HPC systems. As part of emerging fields, exploratory data analysis is presented in Section 4.2.

Participants who successfully complete five modules will receive the HPC Expert award (see Section 2.2).

In contrast to students, the participants in the Supercomputing Academy are a very heterogeneous group of professionals with individual prior knowledge, experience, preferences, and goals. In order to achieve an approximately homogeneous learning group, the necessary prior knowledge is communicated to prospective students in advance of the start of each course module. This gives them the opportunity to catch up on missing knowledge or to refresh their knowledge. This ensures that participants can successfully complete the modules.

Linux knowledge is a prerequisite in all modules, as high-performance computers are only operated with Linux operating systems. In addition, basic knowledge of computer hardware is beneficial for all modules. For some modules—Simulation, Parallel Programming, and Performance Optimization—programming knowledge in Fortran or C is also a prerequisite. Some modules also require special prior knowledge. For example, the Simulation module expects a basic understanding of vector analysis and linear algebra from the participants.

4. TRAINING AND TEACHING MATERIALS

The following two examples, focusing on simulations of heat conduction and suburban train delays, provide a deeper insight into the contents of the Supercomputing Academy and Simulated Worlds. In each case, first the similarities and then the differences in the level of detail and procedure are presented.

4.1 Heat Conduction

Both the do-IT day on heat conduction developed by Simulated Worlds and the Simulation module of the Supercomputing Academy start with the presentation of the simulation cycle and explain it using the heat conduction equation (cf. Section 3). This equation has the advantage that it can be used to show principles that are essential for simulations (e.g., spatial discretization and the role of boundary conditions), and at the same time it is comparatively simple.

4.1.1 Do-IT Day on Heat Conduction from Simulated Worlds

For many students, computer simulations are initially abstract and difficult to conceptualize. In order to make simulation more vivid, the do-IT day focuses on the visualization of heat conduction in a real classroom (see Figure 4 and Figure 5). Didactically, creating computer simulations that visualize the flow of thermal energy provides a very good opportunity to teach otherwise abstract physical concepts [23] and apply them to things that can be experienced by students.

Due to the need to adapt the content of the do-IT days to the knowledge that students have previously acquired in mathematics and physics classes, only the steady-state heat conduction equation is used. Necessary knowledge about partial differential equations is introduced and explained.

The theme of the do-IT day makes it possible to create a sequence of tasks that promote step-by-step learning by having the simulation approximate the real situation. Starting from the thermal equilibrium between walls and windows, additional heat sources (radiators, red rectangles in Figure 4) and then heat sinks (air conditioning, light blue in Figure 4) are introduced into the two-dimensional space.

²¹ <https://www.emas.de/en>

²² <https://www.blauer-engel.de/en/products/electric-devices/data-centers>

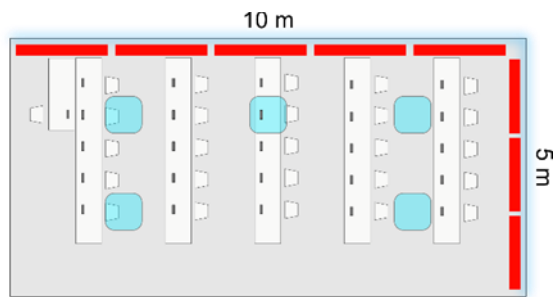


Figure 4. Sketch of the classroom.

The simulation and visualization code is written in Python. Since few students learn this computer language in schools, the do-IT day introduces the basics in about 30 minutes. Furthermore, the code is not completely written by the students, but they are given a code framework with gaps at critical points similar to a cloze. In this way, students' attention is focused on implementing modeling decisions rather than producing many lines of code. An example of the final output after all steps is shown in Figure 5.

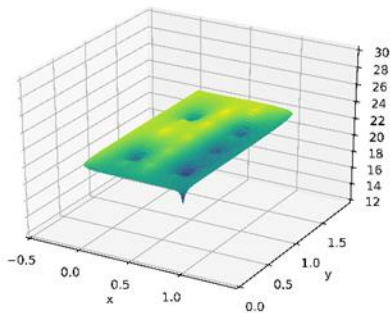


Figure 5. Visualization of the static temperature field in the classroom as a 3D heat map (z-axis in degrees Celsius).

After the simulation, the students reflect on and compare their simulation results, discussing the shortcomings and inadequacies of their models. Furthermore, they can test the limits of the computer hardware used by maximizing the spatial resolution.

4.1.2 Learning Unit “The Simulation Concept” of the Supercomputing Academy

The objectives of the ten-hour learning unit, “The Simulation Concept,” at the beginning of the Simulation module are to illustrate the simulation cycle by means of a real technical and engineering problem and to create a common knowledge base among the participants by introducing important concepts and procedures.

At the beginning of the course unit, the problem is introduced: A windowpane is to be installed in a concrete wall using an aluminum profile. The question arises whether, under the given climatic conditions, a cold bridge is created that promotes mold growth on the inside of the wall. In this case, the aluminum profile would have to be thermally insulated against the wall. This question determines the subsequent identification of the physical quantities involved and the physical-mathematical modeling—in this case, the heat conduction equation with suitable boundary conditions.

Now, the time-dependent heat conduction equation in three dimensions and its spatial and temporal discretization are explained. Furthermore, Dirichlet and von Neumann boundary

conditions, the elementary solution methods (explicit and implicit Euler method), and the Courant-Friedrichs-Lewy number (CFL number) are introduced.

This is concretized in the subsequent exercise and applied to a one-dimensional example: a windowpane with constant temperatures on the inside and on the outside. The participants calculate the temperature distribution of the pane. Since only three support points are used, this task can be solved with a calculator. It corresponds to the third level of Bloom’s taxonomy (application).

Then, participants are provided with a Fortran program that calculates the temperature distribution in the above-mentioned concrete wall in one dimension. The participants fill in gaps in the program code (Bloom’s Taxonomy Level 3). The correctness of the code is verified by the correct temperature calculated at a given point in the wall. The program can be run by participants on the training cluster or on their laptops.

In further tasks, two variants of the code are examined, showing variations in what is called sustained performance. These observations and their explanations lead to the topics of the following learning units.

The author of this learning unit has used commercial software to run a simulation that expands the system boundaries to include the entire window, the aluminum profile, and the wall, as well as areas of the interior (all two-dimensional). The graphical outputs of this simulation are provided to the participants, and the evaluation of the time-dependent temperature field using ParaView is explained. The result of the simulation is related to the present problem described at the beginning; this closes the simulation cycle.

4.2 Analysis of Public Transport Data

In another unit, an exploratory data analysis is used to analyze Stuttgart commuter rail (“S-Bahn”) data to identify and predict correlations that lead to delays. The purpose of this example is to deal with a given data set, i.e., data manipulation and data preparation. Thus, HPC is connected with data science. In addition to machine learning, the focus is on pre-processing and feature engineering. The machine learning pipeline (train, validate, test) and machine learning performance optimization are used.

The main question to the participants is whether the given data set makes it possible for travelers to better plan their travel in the Stuttgart “S-Bahn” using machine learning or perhaps deep learning. Participants approach the solution by completing tasks that facilitate recognition of the correlations and enable the analyses to be carried out. The intensity and duration of the exercises are adapted to the target group.

4.2.1 Scholarship “S-Bahn” Delay from Simulated Worlds

The example is worked on by students in the context of the six-month scholarship of Simulated Worlds and is thus on the level of the Detailed Knowledge of the Knowledge Mountain. During the scholarship, meetings are held at HLRS at regular intervals in which HLRS staff members present the required theory. Following this, students have the opportunity to present their progress and problems and receive direct help from the HLRS employees.

During the course of the scholarship, students first familiarize themselves with the software systems used (Python, Docker, Jupyter Notebook) and conduct independent research to analyze the data. They then prepare the data, learning about different approaches for doing so. To get a better overview and deeper

understanding of the data, students then perform statistical analysis (minima, maxima, averages, standard deviations of delays of different commuter rail lines, and box plots of events). Up to this point, students can list and name what they have learned as well as explain and summarize it.

In preparation for the next steps, students receive an introduction to machine learning from the experts including explanations of how machine learning algorithms work. In addition, the use of classification and regression algorithms to calculate delay forecasts and the extension of the model by additional data sets are explained. The latter are, for example, weather data and data on high traffic volumes caused by large public events such as festivals.

In the last phase, the students have the additional opportunity to apply what they have learned by independently calculating the “S-Bahn” example on a compute cluster at HLRS, enabling them to establish a first contact with HPC.

4.2.2 Example “S-Bahn” at the Supercomputing Academy

At the Supercomputing Academy, the commuter rail (“S-Bahn”) example is part of the module, “Data: Management and Analysis.” The total duration of the module is 12 weeks, of which two weeks are devoted to this example. In the four weeks preceding the “S-Bahn” example, relevant theory is introduced and illustrated with two other examples. Participants are assumed to have prior knowledge in using Python and tools confidently. The entire “S-Bahn” example is run on the training cluster. In two virtual seminars, the participants exchange information and results with the experts and can discuss open questions.

During the “S-Bahn” unit, the question to be answered is whether it is possible to achieve the goal of a minute-by-minute prediction of “S-Bahn” arrivals using machine learning. To determine the answer, participants follow a guided approach to perform an exploratory analysis, make a first interpretation, and then extract first simple statistical values for the delay—all done using Python. This learning objective corresponds to the first and second levels of Bloom’s Taxonomy, knowledge and understanding.

In the “S-Bahn” example, the participants are confronted with problems, e.g., the data set is quite sparse, and the initial feature set is small. Pre-processing and feature engineering clean up noisy data in a first step. In the next step, some data preparation is performed, such as data augmentation and data fusion. Before running the model, linear relationships are determined with a correlation map. After the model is run, unnecessary features that reduce model performance are filtered out. These learning objectives correspond to the third level of Bloom’s taxonomy, application.

Predicting train delays with machine learning involves supervised learning using a set of features called labels. This allows the use of (linear) regression models to predict a delay with high accuracy. Building these models is a difficult task for participants because nonlinear relationships may go undetected or features may be missing. From a set of features, participants predict a discrete label, e.g., delay yes or no. To do this, participants use a variety of tools. This is the fourth level of Bloom’s taxonomy, analysis.

For post-processing and optimizing the prediction of commuter rail delays, participants are asked what options are available to improve the model. Possible solutions include choosing different combinations of features or analyzing the impact of ratio training versus test data (learning curve). This corresponds to the fifth and sixth stages of Bloom’s Taxonomy, synthesis and evaluation.

5. RESULTS

For quality assurance purposes, the offerings of Simulated Worlds and the modules of the Supercomputing Academy are regularly evaluated. Due to the different structures of the two programs, the methods used for this differ in detail.

5.1 Evaluation Results of Simulated Worlds

Due to the wide range of different characteristics, Simulated Worlds evaluates its offerings using various methods to ensure and continuously improve quality. In addition, parts of the program—e.g., the teacher training courses—are subject to continuous quality control by the state of Baden-Württemberg; training courses must first be approved by the Center for School Quality and Teacher Education Baden-Württemberg (ZSL) and have their own evaluation questionnaire at the end of the events. Furthermore, the quality of the courses is checked and, if necessary, improved by continuous feedback from didactic experts, both internally and externally. Internal experts are the teachers employed in the project, the HPC Ambassadors. The external ones are the active teachers, who are always present as observers during programming conducted in schools and classes or are participants of a further training.

In addition, the activities of the project are made evaluable through documentation of the implemented offerings. Table 2 summarizes the third, three-year funding phase of the project with 53 one-day events, 9 project weeks, and 8 double lessons as an example.

Table 2. Overview “Simulation Package” of the third funding phase.

Module	Year 1	Year 2	Year 3	Total
Daylong events**	13	24	16	53
Project weeks**	3	2	4	9
Scholarships*	22	24	20	66
Double lessons**	2	3	3	8
Internships*	6	7	7	20
Teacher trainings*	47	61	21	129
Lectures, final events **	7	13	5	26

Notes: As of June 29, 2021; * Number of participants; ** Number of events; Source: Own representation.

Standardized questionnaires were used for the evaluation of the do-IT day program. In the case described of the do-IT day on heat conduction (see Section 4.1.1), fourteen participating students between the ages of fifteen and seventeen were surveyed. 57% of them were female and 43% male.

The do-IT day was rated very good by 35% of the participants and good by 65%. In response to the open question of which parts of the workshop they particularly enjoyed, the most frequent response was the opportunity to program (four times). Getting a simulation to run themselves, rather than just listening to a lecture, was specifically liked by three participants. It was also emphasized that this type of workshop gives students an idea of what it will be like to work with simulations and computers at a university.

For the (didactic) preparation of such a workshop, it is always a significant challenge for Simulated Worlds organizers to flexibly adapt the schedule to students’ individual abilities and needs by including or excluding additional tasks (such as basics of Python programming or offering additional tasks for some students).

The scholarship also shows evidence of successful use of this strategy. Over the years, more and more schools from the same area have applied. Among the successful applications through 2021, the scholarship had 44 students from 19 high schools in the Stuttgart area; 45 students from 20 high schools in the Karlsruhe area; and, already in its first year of the offering in Ulm, 10 students from 8 different schools in the Ulm area. This includes 19 schools that had already had successful applications in different years, proving that information about the educational enrichment program was passed on within the schools either by the teachers or the students.

Two variants of outcome evaluation were used in the scholarship. The first is the learning journal, mentioned earlier in Section 2.2. Its evaluation makes it possible to continually improve the scholarship concept and adapt it to the needs of students and supervisors.

The narrative self-evaluation in the learning journals shows how the students respond to each task. In one specific example (see Section 4.2.1), a fellow decided to incorporate weather data into a cross-data analysis with commuter rail data to increase forecast accuracy. In the conclusion of the journal, the fellow praised the freedom he had in finding his own solution to expand the data set. He also showed some pride in having found his own way to integrate the weather data into the given database. Such successes greatly enhance the student's confidence in his own abilities and judgment.

The second method of evaluation within the framework of the scholarship was via questionnaires. For this purpose, former scholarship holders were asked about their career development about two years after completing the scholarship. Due to the delay of two years, only the follow-up survey of the first year with a total of 10 scholarship holders is currently available. With a response rate of 70%, an astonishingly large number of former scholarship holders were motivated to respond, considering that the ensuing years involved many personal upheavals including the end of school, taking up studies, or entering professional life, along with the associated relocations.

The survey showed that satisfaction with the scholarship was high in many areas. For the thematic organization of the scholarship, the respondents gave an average grade of 2.0²³; for the meetings with their supervisors, a 2.1; and for the joint meetings they experienced, even a 1.9; in each case without a grade lower than a 3 being given. Only the joint work among the students scored worse, with an average of 2.5. This was because one respondent was not satisfied with his/her scholarship partner and awarded a 5.

The scholarship did particularly well in questions about whether the content was correctly adapted to the performance level of the respondents. Here, the respondents awarded an average of 1.9. Very good scores were also given for the question of whether they would recommend the scholarship to other 11th grade students, with an average of 1.6 (the scores awarded were in the range between one and three).

Meanwhile, all seven respondents are studying at a university, six of them at one in Baden-Württemberg, and one outside Germany. Six are studying a classic STEM subject, one medicine.

The open-ended survey revealed that new skills were acquired during the scholarship. Two respondents said they had learned time

and project management, one person highlighted the experience gained in lecture techniques, and another cited working together in a team as a meta-skill learned.

So far, efforts to ensure the quality of the Simulated World offerings have been successful. The project has also received two awards: in 2013 from the Bosch Foundation and in 2014 from the Kreissparkasse Esslingen.

The biggest challenge for the project will be to respond to changing demands of the educational landscape in the post-pandemic period and to bring the existing offerings back into schools. Many schools are signaling that additional educational offerings will have to be scaled back significantly to compensate for cancelled class time. Nevertheless, the project team has already received a number of event requests for the second half of 2021.

5.2 Results of the Supercomputing Academy

To determine the quality of the modules and the satisfaction of the participants at the Supercomputing Academy, participants are asked to take part in regular surveys. Their standardized questions are based on the rules for successful questionnaire design [20]. In accordance with the criteria for learning success [10], the following aspects are assessed:

- Perceived quality of the offering
- Achievement of learning objectives
- Learning behavior: duration of learning, acquisition of confirmations of participation and certificates
- Transfer to the professional environment
- Subjective satisfaction

144 people took part in the eight module runs carried out within the MoeWE project phase. Almost half of the participants attended two or more modules. 105 (73%) qualified certificates of participation (see also Section 2.2) were issued. 86 participants (60%) successfully completed the final exam and received a certificate. Five HPC developer awards, seven HPC administrator awards, and four HPC user awards were issued. Six participants received the highest award, HPC Expert.

To determine perceived quality and subjective satisfaction, participants provided feedback on each lesson in weekly surveys. At the end of each module, participants gave a final evaluation of the entire module. The surveys were mostly conducted online.

The results presented below are for the final surveys of all completed modules. Responses were on a 5-point scale. The questions for each criterion are summarized in Table 3. A total of 69 people participated in the surveys.

On average, the quality was rated as good. Likewise, the participants stated on average that they had achieved the learning objectives. Overall, the participants were satisfied with the modules. The learning duration was longer than expected, and 60% confirmed the transfer of what they had learned to their professional environment.

In the part on the simulation concept of the Simulation module (see also Section 4.1.2), half of the participants needed more than 10 hours for the learning content. The other participants mastered the learning content within five to ten hours. Here it became particularly clear that the participants had different prior knowledge. Therefore, it is important for successful participation to communicate the prerequisites for a module more clearly in the future.

²³ German school grades range from 1.0 "very good" to 6.0 "unsatisfactory," with at least 4.0 having to be achieved in order to be promoted to the following grade.

Table 3: Survey on the quality assurance of the modules of the Supercomputing Academy.

Criterion	Examples of questions and statements
Quality	“The common thread was clearly recognizable.”
	“The learning content was consistent with the module description.”
Satisfaction	“How satisfied or not satisfied were you with the learning offered in the module?”
	“My expectations were met in the module.”
Learning duration	“My personal time commitment was higher than expected.”
Achieving the learning objectives	“I was able to achieve my set goal.”
Transfer	“I can apply the learning content well in my professional practice.”

The number of participants is still too low to obtain statistically significant figures. Nevertheless, the initial trends are recognizably positive, and the evaluation results for the first run of the modules are very satisfactory overall. The learning content was evaluated as didactically well prepared, which can be seen in the results of the quality and the achievement of the learning objectives.

In addition, the positive trend is confirmed by the numerous participants who attended more than one module and also successfully completed them.

Overall, the modules were very well attended. On average, 18 participants attended the modules. In this particular blended learning format, the maximum limit was set at 20 participants in order to be able to ensure individual support for the learners and to create an optimal learning situation. The current pandemic situation generally favors an online format. However, the number of participants did not increase significantly. The particularly high attendance in the module, “Data: Management and Analysis,” can be credited to the fact that this is currently a hot topic. Here, 28 participants were admitted as an exception.

Public relations activities focused on publications of course offerings, the project website, and networking activities. Existing contacts to HLRs’ industrial partners were also used to acquire interested participants. Many participants approached the Supercomputing Academy by searching for suitable training courses on the Internet or received recommendations from colleagues and superiors. In the future, even more attention will be paid to direct marketing in order to achieve the maximum with these measures.

6. THE NEXT STEPS

Both programs, Simulated Worlds and the Supercomputing Academy, will add new topics to their programs and continuously update and improve the content of their modules and refine their methods.

The continuous development process of Simulated Worlds’ offerings must take into account changes in public school curricula. In addition, more topics will be found that can be used to introduce simulations to students. The adaptation of the programming tasks to the different abilities of the students is also an ongoing task. For cost reasons, it must be ensured that only open source software is used. In addition, the program plans to develop new offerings for (vocational) secondary schools and for teachers in training.

Since the beginning in 2021, the Supercomputing Academy is self-financing through cost-covering registration fees. This poses a major challenge to continue to achieve the good participant numbers seen during the grant-subsidized project phase.

With the relaunch, the modules that previously ran over a period of three months have been divided into smaller modules with a duration of up to six weeks. By focusing on selected content, these smaller modules will meet the individual needs of interested parties and improve the transfer of knowledge and skills to the professional environment.

Content required in several modules, such as instructions on how to use the training cluster and information to provide an understanding of the hardware, will be combined in a basic module and made available to participants for self-study.

In addition, new modules on topics such as machine learning and artificial intelligence are planned. Another consideration is to offer the modules to interested companies as in-house training. In addition, it would be desirable to be able to structure online teaching materials in a more needs-oriented way, for example, by integrating an adaptive learning system [13] that goes beyond the structures provided by the learning management system ILIAS.

7. SUMMARY

Simulated Worlds and the Supercomputing Academy (developed during the MoeWE project) achieved the goal of expanding the audience for training in topics related to simulation and HPC. Both projects offer continuing education programs that are tailored to the needs of the respective target groups including students, teachers, and professionals. By consulting with external experts from schools for Simulated Worlds and from industry for MoeWE and the Supercomputing Academy, the project teams succeeded in developing suitable and demand-oriented offerings. Both the scientists and the experts from industry and schools were enthusiastic about the collaboration as the exchange of insights was enriching for everyone involved.

Simulated Worlds has a large portfolio of offerings for students at various levels of knowledge, from lectures in school classrooms to half-day workshops to scholarships that allow students to conduct small research projects at high-performance computing centers alongside their normal school lessons. Teachers are trained to be able to integrate the topic of simulation into their lessons. Evening lectures are offered to the interested public to draw attention to the advantages and limitations of simulation.

Because the Supercomputing Academy / MoeWE targets working people, it uses a blended learning format for teaching. This gives participants as much independence as possible in choosing the time and place in which they learn in order to enable them to learn successfully while managing work and family responsibilities. The continuing education program is modular in structure and offers continuing education opportunities focusing on key topics in simulation and HPC. Currently, the program portfolio consists of seven modules that cover all relevant topics and can be easily expanded. Participants can earn HPC awards depending on the focus of their continuing education when they successfully complete the appropriate modules.

The learning activities related to heat conduction and the Stuttgart commuter rail system demonstrate a wide range of tasks and exercises for participants of Simulated Worlds and the Supercomputing Academy, offering them the opportunity to

achieve the optimal learning progress in each case through tailored challenges and levels of complexity.

The offerings of both programs are popular with the target groups. Thus, the number of participating schools is steadily increasing. The number of participants in the Supercomputing Academy modules has also risen steadily since the project began, and demand for modules following the project phase is high. Both programs will continue to be offered in the future.

The concepts presented related to these two projects can be transferred very well to other areas of learning because they are independent of the subject matter.

8. ACKNOWLEDGEMENTS

Thanks to funding from the Ministry of Science, Research and the Arts of the State of Baden-Württemberg; the Simulated Worlds project has been successfully established and implemented since 2011.

The project MoeWE (2016–2020) was developed and implemented thanks to funding from the Ministry of Science, Research and the Arts of the State of Baden-Württemberg and the European Social Fund (ESF).

Contributors to the work done in the Simulated Worlds project are Dr. Almut Zwölfer, Jochen Barwind, Peter Lürßen, and Adrian Kaiserauer.

Contributors to the work done in the MoeWE project are Dr. Brigitte Lorenz, Dr. Geneveva Schmidmann, Dr. Nicole Wöhrle, Nicole Prange, Markus Klietmann, Inna Wöckener, Bernd Aumann, Dr. Andreas Wierse, Dr. Norbert Conrad, Christopher Williams, Dr. Ralf Schneider, Dennis Hoppe, Li Zhong, and Dr. Lorenzo Zanon.

9. REFERENCES

- [1] Acatech and Körber-Stiftung (Ed.). 2021. MINT Nachwuchsbarometer 2021. (May 6, 2021). Retrieved July 9, 2021 from: <https://www.acatech.de/publikation/mint-nachwuchsbarometer-2021/download-pdf?lang=de>
- [2] Benjamin S. Bloom, Max D. Engelhart, Edward J. Furst, Walker H. Hill, and David R. Krathwohl. 1956. *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain*. David McKay Company, New York.
- [3] Charles Xie. 2012. Interactive Heat Transfer Simulations for Everyone. In *The Physics Teacher* 50, 237 (April 2012), American Association of Physics Teachers (Ed.). DOI: <https://doi.org/10.1119/1.3694080>
- [4] Christoph Meier. 2019. KI-basierte, adaptive Lernumgebungen (Beitrag Handbuch E-Learning). (July 2019). Retrieved July 9, 2021 from <https://www.scil.ch/2019/07/05/ki-basierte-adaptive-lernumgebungen-beitrag-fuer-handbuch-e-learning/>
- [5] Freydis Vogel and Frank Fischer. 2018. Computerunterstütztes kollaboratives Lernen. *Lernen mit Bildungstechnologien*. Helmut Niegemann and Armin Weinberger (Eds.). Springer Reference Psychologie. Springer, Berlin, Heidelberg. 1–24. DOI: https://doi.org/10.1007/978-3-662-54373-3_3-1
- [6] Gilbert Geefrath and Hans-Georg Weigand. 2012. Simulieren: Mit Modellen experimentieren. *Mathematik lehren, Pädagogische Zeitschriften (in Zusammenarbeit mit Klett)* 174 (October 2012), 2–6.
- [7] Horst Otto Mayer, Johannes Hertnagel, and Heidi Weber. 2009. *Lernzielüberprüfung im eLearning*. Oldenbourg Wissenschaftsverlag, München. DOI: <https://doi.org/10.1524/9783486848984>
- [8] Jörg Hilpert, Rüdiger Berlich, Peter Lürßen, Almut Zwölfer and Jochen Barwind. 2015. Teaching Simulations and High Performance Computing at Secondary Schools in the German State of Baden-Württemberg. In *Proceedings of the 2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, May 25–29, 2015, Hyderabad, India, 731–738. DOI: <https://doi.org/10.1109/IPDPSW.2015.61>
- [9] Juan Chen and Li Shen. 2018. Design of Paper CPU Project to Improve Student Understanding of CPU Working Principle. In *Proceedings of ACM Turing Celebration Conference — China (TURC 2018)*, May 19–20, 2018, Shanghai, China, 96–102. DOI: <https://doi.org/10.1145/3210713.3210735>
- [10] Lorin W. Anderson and David R. Krathwohl (Eds.). 2001. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman, White Plains, NY.
- [11] Maren Hattebuhr, Martin Frank and Christina Roeckerath. 2018. Komplexe Modellierung: Trumpf gegen die Wissenschaft — Gibt es den Klimawandel wirklich? *Gesellschaft für Didaktik der Mathematik, Beiträge zum Mathematikunterricht (2018)*, 731–734. Retrieved June 22, 2021 from: https://eldorado.tu-dortmund.de/bitstream/2003/37393/1/BzMU18_HATTEBUHR_Klimamodellierung.pdf
- [12] Maria Opfermann, Tim N. Höffler and Annett Schmeck. 2019. Lernen mit Medien: ein Überblick. *Lernen mit Bildungstechnologien*. Helmut Niegemann and Armin Weinberger (Eds.). Springer Reference Psychologie. Springer, Berlin, Heidelberg. 1–14. Retrieved July 9, 2021 from https://link.springer.com/content/pdf/10.1007%2F978-3-662-54373-3_2-1.pdf DOI: https://doi.org/10.1007/978-3-662-54373-3_2-1
- [13] Maria Reichelt, Frauke Kämmerer and Ludwig Finster. 2019. Lehrziele und Kompetenzmodelle beim E-Learning. *Lernen mit Bildungstechnologien*, Helmut Niegemann and Armin Weinberger (Eds.). Springer Reference Psychologie. Springer, Berlin, Heidelberg. 1–16. DOI: https://doi.org/10.1007/978-3-662-54373-3_15-1
- [14] Marie des Jardins and Michael Littman. 2010. Broadening Student Enthusiasm for Computer Science with a Great Insights Course. In *SIGCSE '10: Proceedings of the 41st ACM technical symposium on Computer science education*, March 2010, New York, USA, 157–161. DOI: <https://doi.org/10.1145/1734263.1734317>
- [15] Marlen Schulz. 2012. Quick and easy?! Fokusgruppen in der angewandten Sozialwissenschaft, *Fokusgruppen in der empirischen Sozialwissenschaft: Von der Konzeption bis zur Auswertung*, Marlen Schulz, Birgit Mack and Ortwin Renn (Eds.). Springer VS, Wiesbaden, 9–22. DOI: https://doi.org/10.1007/978-3-531-19397-7_1

- [16] Michael Kerres. 2018. Chapter 3 — Gründe für das Lernen mit Medien. *Mediendidaktik*. De Gruyter, Oldenburg, 87–126. DOI: <https://doi.org/10.1515/9783110456837-099>
- [17] Michael Kerres. 2002. Online- und Präsenzelemente in hybriden Lernarrangements kombinieren. *Handbuch E-Learning*. Andreas Hohenstein and Karl Wilbers (Eds.). Deutscher Wirtschaftsdienst, Köln, 7–12.
- [18] Ministerium für Kultus, Jugend und Sport Baden-Württemberg (Ed.). 2004. Bildungsplan 2004: Allgemein bildendes Gymnasium, 400–402. Retrieved June 22, 2021 from: http://www.bildungsplaene-bw.de/site/bildungsplan/get/documents_E978621370/lbw/Bildungsplaene/Bildungsplaene-2004/Bildungsstandards/Gymnasium_Bildungsplan_Gesamt.pdf
- [19] Miriam Venn. 2011. Lerntagebücher in der Hochschule. In *Journal Hochschuldidaktik* 1, 9–12. Retrieved October 10, 2016 from: http://www.zhb.tu-dortmund.de/hd/journal-hd/20111/journalhd20111_venn.pdf
- [20] OECD, European Union and UNESCO Institute for Statistics (Eds.). 2015. ISCED 2011 Operational Manual: Guidelines for Classifying National Education Programmes and Related Qualifications. (March 20, 2015). OECD Publishing. DOI: <http://dx.doi.org/10.1787/9789264228368-en>
- [21] Teaching & Learning Academy. 2019. 1.3 Learning Outcomes. Wirtschaftsuniversität Wien (Ed.) (November 2019). Retrieved June 22, 2021 from: <https://learn.wu.ac.at/open/tlac/learningoutcomes>
- [22] Rolf Porst. 2000. Question Wording — zur Formulierung von Fragebogen-Fragen. *GESIS-How-to*, 2, Zentrum für Umfragen, Methoden und Analysen — ZUMA (Ed.), Mannheim. Retrieved November 12, 2020 from: <https://nbn-resolving.org/urn:nbn:de:0168-ssoar-201334>
- [23] Thomas J. Cortina. 2007. An Introduction to Computer Science for Non-Majors Using Principles of Computation. *ACM SIGCSE Bulletin* 39, 1 (March 7, 2007), 218–222. DOI: <https://doi.org/10.1145/1227504.1227387>

Infusing Fundamental Competencies of Computational Science to the General Undergraduate Curriculum

Ana C. González-Ríos
University of Puerto Rico - Mayagüez
anacarmen.gonzalez@upr.edu

ABSTRACT

The growing need for a workforce that can analyze, model, and interpret real-world data strongly points to the importance of imparting fundamental concepts of computational and data science to the current student generation regardless of their intended majors. This paper describes the experiences in developing and implementing a course in computation, modeling, and simulation. The main goal of the course was to infuse fundamental competencies of computational science to the undergraduate curriculum. The course also aimed at making students aware that modeling and simulation have become an essential part of the research and development process in the sciences, social sciences, and engineering. The course was targeted to students of all majors.

Keywords

Computational science, Flipped classroom, Computational and data science literacy, General education, Python, Curriculum development

1. INTRODUCTION

Computational science (CS) is an interdisciplinary field that can be defined as the intersection of the domain area of the problem of interest, computer science, and mathematics. It requires mathematical modeling capability and the skills for its efficient implementation using computing techniques. CS allows us to build models, visualize phenomena, and conduct experiments difficult or impossible in the laboratory. So, it plays a critical role in the future of the scientific discovery process and engineering design [11, 8, 12].

It is important to reach not only the undergraduate curriculum in science, technology, engineering, and mathematics (STEM). All undergraduate students should be exposed to the fundamentals of CS. Hence, students can be better equipped to apply CS techniques in their fields and to better understand society and their environment [8, 12].

All students, independent of their area of study, should have access to a course that will help them develop fundamental CS competencies. After reviewing the curriculum at a public Minority Serving Institution, it was observed that the courses that address the topics of modeling, simulation, and data science have multiple advanced pre-requisites, so it is hard for many students to take those

courses. Sometimes students who do have the pre-requisites cannot fit the courses in their curricula. It was also noted that these courses are offered randomly depending on enrollment and professor availability.

With the main goal of infusing fundamental concepts of computational science into the undergraduate general education curriculum; we proposed, developed, and implemented a new elective course with only college algebra as a pre-requisite.

All undergraduate students could benefit from this course including students who planned on participating in undergraduate research. Pre-service teachers were also an important target so that they would be better able to integrate CS in their K–12 courses. Computer science students had the opportunity to see how programming skills can be applied to science, social sciences, and engineering. The course could provide a panoramic view of the field and act as a catalyst to continue with more advanced courses.

This paper describes the set of basic learning outcomes upon which the course content rested, course description, course objectives, resources developed, and overall format of the course. Observations on students' perception of the course are also discussed.

2. COURSE LEARNING OBJECTIVES

The primary goal of the course was to introduce basic concepts of computational science to a diverse student body. The aim was not to produce experts in computational science but to provide the tools and skills that can benefit the personal and professional lives of individuals and allow them to better collaborate with computational scientists. With this mindset and the fact that the course would only require college algebra, we proceeded to determine the learning outcomes of the course. The learning outcomes represented the framework for the course content.

The computational science educational competencies that guided the course design derived from the work developed as part of a project at the Ohio Supercomputer Center sponsored by the National Science Foundation [17]. The competencies were created by the participating faculty and then reviewed by a business advisory committee [9].

The group identified seven competency areas shown in Table 1. As shown in Figure 1, each area was further subdivided to describe the level of the skills and knowledge necessary to master the competencies [17].

Consistent with [11, 7, 9], we believed that an introduction to computational science must equip learners with a basic understanding and an integration of both modeling and computer programming principles. Learners should be provided with experiences on how to translate the relationships within a system being modeled into a set of mathematical functions that accurately portray the behavior of that system and then translate the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

© 2021 Journal of Computational Science Education
DOI: <https://doi.org/10.22369/issn.2153-4136/12/3/3>

mathematics into computer code that correctly simulates those relationships.

Table 1. Competency areas [17].

1	Simulation and Modeling
2	Programming and Algorithms
3	Differential Equations and Discrete Dynamical Systems
4	Numerical Methods
5	Optimization
6	Parallel Programming
7	Scientific Visualization

- Area 1: Simulation and Modeling [-]**
- Explain the role of modeling in science and engineering:[+]
 - Analyze modeling and simulation in computational science:[+]
 - Create a conceptual model:[+]
 - Examine various mathematical representations of functions:[+]
 - Analyze issues in accuracy and precision:[+]
 - Understand discrete and difference-based computer models:[+]
 - Demonstrate computational programming utilizing a higher level Mathematics, other):[+]
 - Assess computational models:[+]
 - Verification, Validation, and Accreditation:[+]
 - Complete a team-based, real-world model project:[+]
 - Demonstrate technical communication:[+]
-
- Area 2: Programming and Algorithms [-]**
- Describe the fundamentals of problem solving[+]
 - Understand and write Pseudo code:[+]
 - Use subprograms in program design:[+]
 - Write code in a Programming language:[+]
 - Use different approaches to data I/O in a program:[-]
 - Explain the advantages and disadvantages of file I/O
 - Describe the syntax for file I/O in your programming language
 - Compare binary and ASCII file I/O
 - Write code using file I/O and keyboard/monitor I/O
 - Understanding and use of fundamental programming Algorithms:
 - Explain various approaches to Program Design:[+]
 - Possible Additions - Understand basic concepts of parallel program

Figure 1. Subset of competency subdivisions [17].

Table 2. Course learning outcomes.

1.	Explain the role of modeling in the sciences and engineering.
2.	Explain the terms of modeling in the sciences and engineering.
3.	Create a conceptual model.
4.	Write code in a programming language.
5.	Use subprograms in program design.
6.	Understand and write pseudocode.
7.	Describe the fundamentals of problem solving.
8.	Use different approaches to data I/O in a program.
9.	Understanding and use of fundamental programming algorithms.
10.	Understand discrete and difference-based computer models.
11.	Understand the use of empirical data.
12.	Understand the modeling process.
13.	Verification and validation.
14.	Technical communication.
15.	Demonstrate computational programming.

We believed that an introductory course should cover the skills and knowledge described in the first two areas: simulation and modeling, programming, and algorithms. The selected learning outcomes as shown in Table 2 represented what the students would be able to do at course completion.

Guided by the HPC University [17] competencies, the learning outcomes as shown in Tables 3a and 3b were further subdivided for a more detailed description of the skills and knowledge that needed to be acquired.

Table 3a. Learning outcomes subdivision.

1	Explain the role of modeling in the sciences and engineering. a) Describe the importance of modeling to science and engineering. b) Describe the history and need for modeling. c) Describe the cost effectiveness of modeling. d) Describe the time-effect of modeling.
2	Explain the terms of modeling in the sciences and engineering. a) Define modeling terms. b) List questions that would check/validate model results. c) Describe future trends and issues in science and engineering. d) Identify specific examples of modeling in science.
3	Create a conceptual model. a) Utilize the modeling process to identify key parameters of a model. b) Estimate model outcomes. c) Use Python to implement the mathematical representation of the model.
4	Write code in a programming language. a) Understand the concept of syntax in a programming language. b) Describe the syntax of the programming language constructs. c) Understand the difference between a compiled and interpreted language. d) Write and run basic programs in the language of choice. e) Understand how to de-bug code. f) Understand the numerical limits of various data types and the implications for numerical accuracy of results.
5	Use subprograms in program design. a) Describe how logical tasks can be implemented as subprograms. b) Explain the control flow when a function is called. c) Explain how function output is used. d) Understand how languages handle passed data into functions and subprograms, especially one- and two-dimensional arrays.
6	Understand and write pseudocode. a) List the basic programming elements of pseudocode. b) Explain the logic behind an if/then/else statement. c) Understand the iterative behavior of loops. d) Describe the difference between several looping constructs. e) Write pseudocode to solve basic problems.

Table 3b. Learning outcomes subdivision.

7	Describe the fundamentals of problem solving. a) Understand top-down thinking and program design. Discuss breaking up a problem into its component tasks. Understand how tasks acquire data. b) Describe how tasks should be ordered. c) Represent tasks in a flow-chart style format.
8	Use different approaches to data I/O in a program. a) Explain the advantages and disadvantages of file I/O. b) Describe the syntax for file I/O in your programming language. c) Write code using file I/O and keyboard/monitor I/O.
9	Understanding and use of fundamental programming algorithms. a) Explain an algorithm as an ordered series of solution steps. b) Describe an algorithm for a simple programming problem. c) Describe what a software library is. d) Construct difference-based computer models.
10	Understand discrete and difference-based computer models. a) Write simple Python programs performing numerical calculations as needed for modeling and simulation. b) Implement finite difference modeling equations and create simulations in Python.
11	Understand the use of empirical data. a) Visualize empirical data and the fitting function using Python. b) Use data science techniques to illustrate data relevant to social change issues and interpret the results.
12	Understand the modeling process. a) Identify different types of models and simulations. b) Describe iterative development of a model. c) Explain use of models and simulation for hypothesis testing.
13	Verification and validation. a) Discuss methods for reviewing models their verification and validation. b) Describe the differences between predictions of model, actual results, and relevance of these differences to the problem. c) Suitability/limits of models.
14	Technical communication. a) Document the development and implementation of a model and present it in oral and written form.
15	Demonstrate computational programming. a) Describe the computational programming system environment. b) Define elementary representations, functions, matrices and arrays, script files. c) Explain relational operations, logical operations, condition statements, loops, debugging programs. d) Create tabular and visual outputs. e) Translate the conceptual models to run with this system and assess the model results.

3. COURSE DESCRIPTION, COURSE OBJECTIVES

Based on the selected learning outcomes, the course description as stated in the University's Course Catalog was as follows:

Introduction to the principles of modeling and simulation; progressive introduction of programming principles and skills using a high-level programming language; application of programming skills to the solution of different classes of models.

The course did not require programming experience and the mathematics pre-requisite was college algebra.

The course objectives were:

1. Provide a background for more advanced modeling courses.
2. Provide the students with an introduction to modeling and its importance to current practices in different subject domains; like science, social sciences, and engineering.
3. Introduce programming principles and apply them to the solution of different classes of models.
4. Provide an overview of the modeling process and the terminology associated with modeling and simulation.
5. Study the mathematical representation of different classes of models.
6. Introduce techniques for fitting a function to an experimental data set.
7. Provide the opportunity for students to document the development and implementation of a model and present it in oral and written form.

4. COURSE ELEMENTS

Tables 4a and 4b show a mapping of the course topics and course learning outcomes. Each topic was covered as a unit that included instructional materials, learning activities, and assessments.

Table 4a. Course topics and learning outcome mapping.

	Topic	Learning Outcomes
1	Introduction to modeling; modeling concepts and definitions	1a,b,c,d 2a,b,c,d 13c 12c
2	Introduction to the Programming Environment	4a,b,c,d,e 9c
3	Deterministic Linear Models	3a,b,c 12a,b 4f
4	Array Mathematics and Python	15a,b 5d
5	Plotting in Python	11a,b 15d
6	Problem solving	7a,b,c 9a,b 6a,e
7	Conditional Statements	6b,15c
8	Iteration and Loops	4b,6c,d,e
9	Nonlinear and Dynamic Models	10a,b 12a,b 9d

Table 4b. Course topics and learning outcome mapping.

10	Estimating Models from Empirical Data	3a,b,c 8a,b,c 11a,b 15d,e
11	Stochastic Models	12a,b
12	Functions in Python	5a,b,c,d
13	Verification, Validation, and Errors	13a,b,c 12c
14	Project implementation, Students are given in-class time to work in groups in their final project presentations	14a

4.1 Course Textbook

The textbook selected for the course was *Introduction to Modeling and Simulation with MATLAB® and Python* by Steven I. Gordon and Brian Guilfoos [11]. The reason for selecting the book was that its content focuses on meeting the set of basic modeling and simulation competencies as defined in HPC University [17], and it uses a just-in-time approach to introduce both programming and modeling concepts.

4.2 Python Programming Language

Python [15] and the Spyder integrated development environment [19] were selected as the platform to develop the programming skills.

Python is a popular general-purpose programming language; it is Free and Open-Source Software that can be acquired and used at no cost. It provides powerful tools and libraries like SciPy and NumPy for scientific computing and Matplotlib and Seaborn for data visualization. Python has a rich web ecosystem of pedagogical resources.

5. PEDAGOGICAL MODEL

The course was designed to combine face-to-face interventions with online, web-enabled strategies, a pedagogical model known as flipped classroom [6]. The flipped classroom switches the typical face-to-face lectures and homework elements of a course. Instead of attending the traditional lecture, students engage with short video presentations or other multimedia content asynchronously before the class period [13]. The in-class period is spent in student-centered learning strategies such as active learning activities like discussions, assignments, laboratories, and mini projects; allowing the instructor to spend more time guiding and supporting the students' progress [16]. This teaching approach has been documented to benefit students' learning outcomes [10, 2, 1].

6. RESOURCES DEVELOPED

The development of all course materials was guided by the course learning outcomes and by taking into consideration the expected students' mathematical background. The created instructional material consisted of videos (equivalent to a lecture but divided in segments of 5-to-15 minutes), practice exercises, walkthroughs, multiple choice questions, and recorded demonstrations. These as well as the course evaluations were made available through the course management system, Moodle [14]. The course and instructional materials were designed following the best practices for using technology and the theory for the creation of courses [5, 4]. The center of resources for distance education, CREAD [3], at the author's institution offered the necessary guidance and support during the design and development of the instructional materials and assessments.

7. COURSE FORMAT

The course met twice a week, Tuesdays and Thursdays, for two hours each time. New material was made available in cycles of one week. A new cycle started at the end of the class period on Thursdays. During the in-class time of the second half of a cycle (Thursdays), the students took a 15-minute, multiple-choice quiz of the material on the current cycle. The idea was to provide an incentive to the students to stay on top of the material.

The students also had to work and submit what was denoted as the module's activity (a list of which is provided in Appendix A). The purpose was to practice and develop the skills introduced in the module. On occasion, the activity was a mini project in which the students organized and developed their work to present/share at the next in-class time. The students were given an outline of what was expected in the presentation, for example, to include four slides and the topic to discuss in each slide.

The students were allowed to work by themselves or in groups of at most three students. They were encouraged to share ideas with one another. The instructor was always available to answer questions and give feedback as needed.

A solution was discussed at the end of the period. On occasion, a student volunteer presented his/her solution to the rest of the class.

During the first half of the cycle (Tuesdays), the students were expected, but not required, to have studied all the online materials. At the beginning of the period, the students could ask questions about the module's materials. The instructor also delivered a short lecture, of at most 15 minutes, that covered core concepts and examples from the videos. This lecture also aimed at focusing and guiding the students on how to study videos and other online materials. Students could spend time working on the practice activities. Practice exercises were similar to what was discussed in the video. The students were also asked to modify the mathematical models or code and to comment on the results. On the other hand, the module's activity and mini project could be described as an open-ended problem. The students needed to recognize the mathematical model and corresponding coding from the videos and other online materials to develop their solution.

7.1 Examples of Course Materials

Some examples of course materials are presented in Appendix C. Individuals interested in reviewing additional materials or resources from other topics are encouraged to contact the author.

8. COURSE EVALUATIONS

Two types of evaluations of the students were offered: those that required the students' individual efforts and those that could be done collaboratively.

The individual evaluations were in the form of timed, multiple-choice questions quizzes. One 15-minute quiz was offered at the end of each cycle. Toward the end of the semester, the students took a 60-minute, multiple-choice exam that covered all the material discussed to the that point. This type of evaluation assessed students' knowledge on terminology and basic concepts.

The collaborative evaluations included the module's activity, the mini projects, and the end of semester project.

9. FINAL PROJECT: COMMENTS

As a final evaluation, students were required to develop, document, and present a project. Keeping in mind that this is a foundational course, the aim of the project was to provide the means for students

consciously to go through all the steps of the modeling process. The objective of the project was to build and test a model of a system and use that model to derive insights into the system behavior [11]. The project specifications indicated that the code be written in Python and that students must use graphs to discuss part of the results.

Two weeks before the end of the semester, the students had to submit a project proposal. In the proposal, the students explained the goals and motivations of the project, why the project was of interest, and what question(s) they would like to answer or what problem(s) they were trying to solve.

To help guide in the project topic selection and development, students were directed to the course textbook where the last chapter provides a list of project topics and a set of reference materials for each topic [11]. Other reference materials were available at Shodor [18]. The students could design, develop, and implement the final project either in groups or individually. A written report and an oral presentation were required. A list of project topics is presented in Appendix B.

10. FINAL DISCUSSION

There have been two offerings of the course, the first during spring 2019 and the second during spring 2020. The first time, it was offered in the format as described in the paper. The second time, due to the COVID-19 pandemic, it was offered completely online. On both occasions, the course was advertised through the University internal e-mail system.

In spring 2019, 16 students registered to take the course, and only one withdrew due to health conditions. For the spring 2020 cohort, there were a total of 13 students. The breakdown by year is shown in Table 5.

Table 5. Year breakdown

Year	Spring 2019	Spring 2020
Freshman	0	0
Sophomore	3	5
Junior	6	7
Senior	7	1

At the beginning of the semester, the students were asked to write two or three paragraphs describing why they had taken the course. Of the spring 2019 group, there was one student from the Physics Department, one student from the Economics Department, and the rest of the students were from the Computer Science program. Their answers to why they were taking the course can be described as follows: the student from the Physics Department mentioned that the course would provide an introduction to more advanced courses in the topic of modeling and simulation and would help develop programming skills. The student from the Economics department mentioned that, according to the search she had done on necessary skills for the modern workforce, this course would help her acquire some of those competencies. Seven of the students from the Computer Science program mentioned that they had taken the course to fulfill a required elective and secondly to learn about Python. The rest of the students also took the course to fulfill an elective requirement and said they were curious about the modeling and simulation topics and felt that learning the skills of problem solving and Python would be advantageous for their careers.

In the spring 2020 group, there were six students from the Physics department, one student from the Business faculty, one student from Mathematics, and the rest of the students from the Computer Science program. Their answers to why they were taking the course can be described as follows. The students from the Physics department were taking the course because it was recommended by other students who had taken the course or had heard about the course. According to their comments, these students were looking for an introductory course in computational science that would provide experience in implementing solutions using a programming language rather than focusing on the programming language itself. The rest of the students were taking the course to fulfill an elective requirement. They also mentioned being interested in learning about the programming language Python.

A list of selected questions from the end-of-semester student survey is shown in Table 6. The answers to the first two questions enumerate the preferred activities and course concepts most frequently mentioned. From the answers to the last three questions, it can be said that by the end of the semester the students preferred the flipped classroom model to face-to-face. It was often mentioned that they appreciated the immediate feedback and guidance from the instructor. The high percentages of affirmative answers to recommending the course to other students and mentioning in the resume having taken the course suggest that they valued and recognized the importance of the skills and concepts acquired through the course.

Table 6. Selected questions.

Questions	Most frequent answers	
1. From the module activities and mini projects, which were the ones that you enjoyed the most?	Concept map, linear models plotting with Python, linear regression, curve fit, modeling process.	
2. List three concepts that you remember from the course.	The Traffic Model, calculating prices of car rentals, compound interest, designing an app for the best gas station selection.	
	Affirmative Answers	
	Spring 2019	Spring 2020
3. Do you prefer the flipped model vs. the face-to-face?	77%	N/A
4. Would you recommend this course to other students?	100%	90%
5. Do you think that you can benefit from mentioning in your resume that you have taken this course?	92%	80%

After offering the course in both the flipped classroom and online modalities, two final observations are described. First, for the end-of-semester project, the students had to submit both written and oral reports. The oral presentation needed to be supported by a PowerPoint (PPT) presentation. For spring 2019, the oral presentation was face-to-face. It was noticed that most of the students for the PPT did a copy and paste of parts of the written report, tended to read from the PPT, and had problems keeping to the presentation time limits. For spring 2020, the students formed virtual groups, and for the oral presentation, they had to submit an audio recording. This worked well because they were forced to reflect about their work and organize their thoughts to prepare a script before recording. Each member of the group had to submit his/her own audio recording.

Secondly, as a distance course, it was observed that only a few students made use of the virtual office hours, and the main feedback they received was from the graded evaluations as opposed to the immediate feedback obtained from the flipped classroom model.

To conclude, the experience of developing and implementing a course in computation, modeling, and simulation has been very gratifying. We believe that such a course provides the means to facilitate, to a diverse student body, the access to the much-needed fundamental CS competencies. From the students' comments, we believe that they recognize the importance of having knowledge in modeling and simulation as well as the leading edge it provides in entering today's workforce. The students also valued the pedagogical model that allowed for active learning activities as well as immediate feedback from the instructor. We are hoping that for our next cohort we can impact pre-service teachers as well as provide professional development to K–12 teachers.

11. ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation funded Grant CyberTraining: CUI: Computational and Data Science Literacy Exchange (CSE-1829717). The author would like to acknowledge the support of Linda Akli from SURA and Katharine Cahill from OSC and the rest of the C²Exchange team for providing a five-star space for our curriculum development discussions.

12. REFERENCES

- [1] Netravathi Basavaraj Angadi, Avinash Kavi, Kimi Shetty, and Nayana Kamalnayan Hashilkar. 2019. Effectiveness of flipped classroom as a teaching-learning method among undergraduate medical students — An interventional study. *Journal of Education and Health Promotion*, 8, Article 211 (Oct. 2019). DOI: https://doi.org/10.4103/jehp.jehp_163_19
- [2] R. Brewer and S. Movahedazarhouligh. 2018. Successful stories and conflicts: A literature review on the effectiveness of flipped learning in higher education. *Journal of Computer Assisted Learning* 34 (Feb. 2018), 409–416. DOI: <https://doi.org/10.1111/jcal.12250>
- [3] Centro de Recursos para la Educación a Distancia. 2019. CREAD. Retrieved September 8, 2021 from <https://www.uprm.edu/cread/>
- [4] Centro de Recursos para la Educación a Distancia. 2019. Mejor Prácticas. Retrieved June 11, 2021 from <https://www.uprm.edu/cread/mejores10practicass/>
- [5] José Ferrer. 2017. *Diseño y creación de materiales educativos: Guía de la mínimo a lo óptimo para cursos en línea* (Spanish Edition).
- [6] Flipped Learning Network (FLN). 2014. What Is Flipped Learning? Retrieved September 8, 2021 from https://flippedlearning.org/wp-content/uploads/2016/07/FLIP_handout_FNL_Web.pdf
- [7] Steven I. Gordon. 2010. Creating Computational Science Programs for the Existing and Future Workforce. In *Proceedings ACM / IEEE SC2010 — International Conference for High Performance Computing, Networking, Storage and Analysis SC10*, Nov. 13–19, 2010, New Orleans, LA.
- [8] Steven I. Gordon and Katharine Cahill. 2020. The State of Undergraduate Computational Science Programs. *Journal of Computational Science Education* (Apr. 2020), 7–11. DOI: <https://doi.org/10.22369/issn.2153-4136/11/2/2>
- [9] Steven I. Gordon, Kate Carey, and Ignatios Vakalis. 2008. A Shared, Interinstitutional Undergraduate Minor Program in Computational Science., *Comput. Sci. Eng.*, 10, 5 (Aug. 2008), 12–16. DOI: <https://doi.org/10.1109/MCSE.2008.127>
- [10] Steven I. Gordon, James Demmel, Lizanne Destefano, and Lorna Rivera. 2015. Implementing a Collaborative Online Course to Extend Access to HPC Skills, *Comput. Sci. Eng.* 18, 1 (Dec. 2015), 73–79. DOI: <https://doi.org/10.1109/MCSE.2016.6>
- [11] Steven I. Gordon and Brian Guilfoos. 2017. *Introduction to Modeling and Simulation with MATLAB® and Python* (1st. ed.). Chapman and Hall/CRC. London, UK.
- [12] Scott A. Lathrop, Katharine Cahill, Steven I. Gordon, Jennifer Houchins, Robert M. Panoff, and Aaron Weeden. 2020. Preparing a Computationally Literate Workforce. *Comput. Sci. Eng.* 22, 4 (May 2020), 7–16. DOI: <https://doi.org/10.1109/MCSE.2020.2994763>
- [13] Michigan State University. What, Why, and How to Implement a Flipped Classroom Model. Retrieved September 8, 2021 from <https://omerad.msu.edu/teaching/teaching-skills-strategies/27-teaching/162-what-why-and-how-to-implement-a-flipped-classroom-model>
- [14] Moodle Pty Ltd. 2021. Moodle: Online Learning with the World's Most Popular LMS. Retrieved September 8, 2021 from <https://moodle.com/>
- [15] Python Software Foundation. 2021. Welcome to Python.org. Retrieved September 14, 2021 from <https://www.python.org/>
- [16] Ashley Radder-Renter. 2020. The Flipped Classroom Model: What It Is and How It Works. (Sept. 2020). Retrieved September 8, 2021 from <https://www.yeseep.org/blog/the-flipped-classroom-model-what-it-is-and-how-it-works>
- [17] Shodor. 2011. HPC University: Minor Program in Computational Science Competency/Topic Overview. Retrieved June 7, 2021 from <http://hpcuniversity.org/educators/undergradCompetencies/>
- [18] Shodor. 2021. Shodor: A National Resource for Computational Science Education. Retrieved August 26, 2021 from <http://www.shodor.org/>
- [19] Spyder Website Contributors. 2021. Home — Spyder IDE. Retrieved September 14, 2021 from <https://www.spyder-ide.org/>

APPENDIX A

List of Module Activities and Mini Projects

1. Watch a video and:
 - a. Describe how the scientific method and simulation were mentioned in the video
 - b. Describe the hypothesis mentioned in the video
 - c. Describe if experiments are used to confirm or deny the hypothesis
2. Given an open-ended problem (best gas station option), design an app, and state:
 - a. List of assumptions, concept map, mathematical model
3. Converting mathematical expressions to Python expressions
 - a. Creating and editing Matrices.
 - b. Using IDE Spyder.
 - c. Using Numpy arrays.
4. Deterministic Linear Models: Modeling and Implementing a Traffic Model
5. Arrays Mathematics in Python using NumPy
6. Visualization with Python, storytelling with a dataset
7. Describe an algorithm for the problem of solving the real roots of a quadratic equation, create a flow chart as part of the solution, Calculating car rental price
8. Implementing different situations, where repetition structures are required, implement a guessing game.
9. Practice exercises on population growth, bank account interest rates, use of visualization to answer what-if questions.
10. Best fit from empirical data: implementing a model to predict the weight of a dog at any time during its life.
11. Using a dataset find the coefficients for a linear model using at least two of the different procedures in Python.

APPENDIX B

List of some of the topics of the final project

1. Finding the fastest route from home to UPRM, using traffic data and precipitation data
2. Effect of altitude on projectile moving at fast speeds
3. Ball Toss
4. Modeling predator prey population
5. SIR Model of COVID-19 in the USA
6. Prey-Predator Model with carrying capacity
7. Natural Disaster Model for Future Economic Losses

APPENDIX C**Topic 1***Activity 1*

Watch the video in the provided link:

https://youtu.be/T9qoU9_tGhA

And answer the following questions:

1. Describe how the scientific method and simulation are mentioned in this video.
2. What is the hypothesis mentioned in the video?
3. Is it possible to make experiments to confirm or deny the hypothesis? Explain.

Activity 2

(based on an Example presented by Daniel Teague of North Carolina School of Science and Mathematics)

Suppose there is a local radio station that broadcasts the locations and prices for all the gas stations in your area. The question is, which should you buy from?

Make a PPT. Imagine you want to make an app for the phone, that can be used to answer that question. Turn in five PPT slides to illustrate your ideas.

Slide 1: List the assumptions.

Slide 2: Show the concept map.

Slide 3: Show the mathematical model.

Slides 4 & 5:

Create screens for an app based on your model. The first screen should request essential information from the user, and the second should show the app's response.

Attach your PPT document in the Moodle platform. In addition to the PPT, each student will also submit a script consisting of the explanation of your work. It is recommended that you also submit an audio version of the script.

Topic 2

Sample exercise:

Using IDE Spyder, write code to:

1. Create a NumPy array with values 1, 7, 13, 105 and determine the size of the memory occupied by the array.
2. Create and print a NumPy array of integers from 30 to 70 in steps of 10.

Convert a list of numeric values into a one-dimensional NumPy array.

Topic 5

A figure of what is shown in Moodle

Lesson 5: Introduction to Visualization with Python Matplotlib



En este modulo se provee una introducción a visualización usando Python.



Walkthrough: Introduction to Visualization with Python Matplotlib

[View](#)



Video: DemoSaving Graph

[View](#)



Video: Demo Graphics Preferences

[View](#)



Module5 Activity PracticeExercises

[Mark as done](#)



Video: Import Data

[View](#)



MiniProject Story Telling

[View](#)

[Make a submission](#)

[Receive a grade](#)



Lesson5Quiz

[View](#)

[Receive a grade](#)

Topic 7

If else demo:

<https://youtu.be/F8QwAhOiuq4>

Topic 8

Sample of video (while):

<https://youtu.be/a72XiszmKE8>

Topic 10

Sample of videos:

<https://youtu.be/bsopgKEdBf0>

<https://youtu.be/BzvYO13HDuk>

<https://youtu.be/6I9d7FVGEoo>

December 2021

Volume 12 Issue 3

ISSN 2153-4136 (online)