Geothermal Sustainability

New Mexico Supercomputing Challenge Final Report

April 3, 2011

Team 62

Los Alamos High School

Team Members

Martin Loncaric

Adam Lang

Teacher Sponsor

Leroy Goodwin

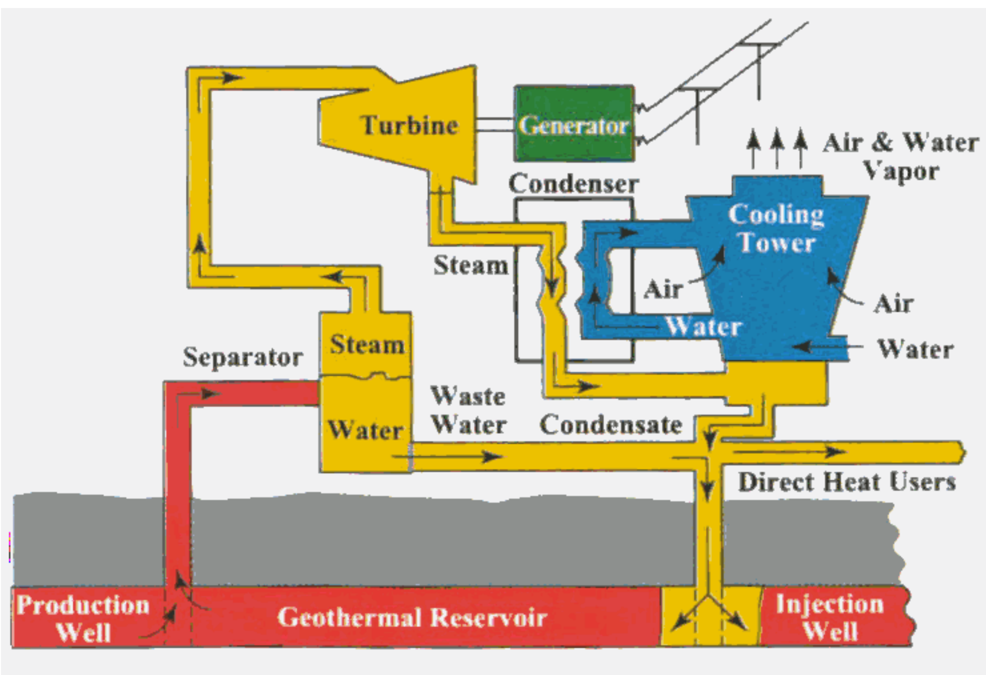
Table of Contents

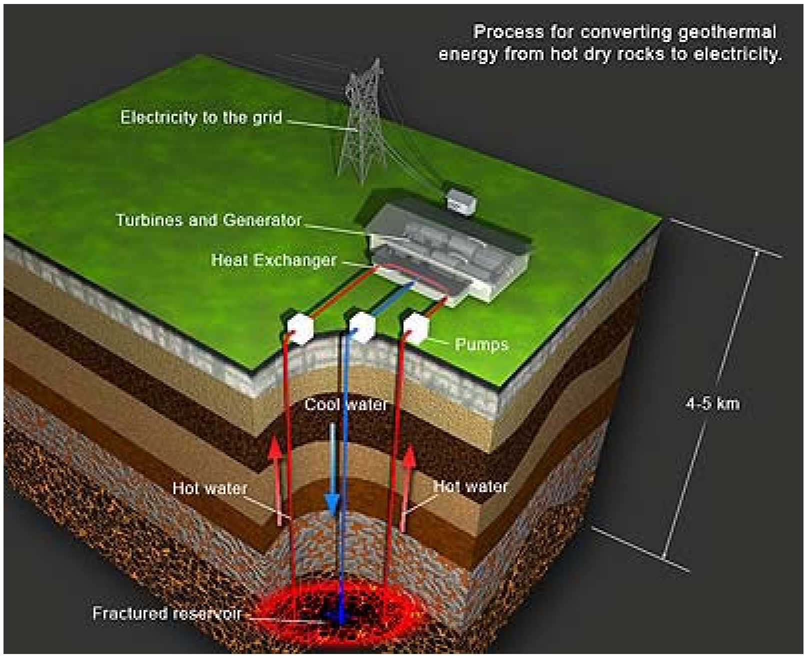
Executive Summary

In this project we chose to model a geothermal power plant by using Java in a grid-based simulation. Our particular sort of plant pumps cold water into the ground until it become hot or turns into steam. The goal was to find the simplest layout of pipes that would not deplete the power output of the rock or exceed the bounds of practicality for a construction project. For the former condition, we used our test over a much longer time period with longer increments of time, after accelerating conditions to match those in a long-standing geothermal reservoir. For the latter, we limited our pipeline to 2000 meters in length once the ideal depth of 5.5 kilometers had been reached. All simulations were done exclusively at this depth for the sake of control. After much struggle, our code worked exactly as it was expected to. Some of its simpler results can be directly verified by equations engineers use to model the flow of a moving fluid in a pipe. We collected more data to verify our program's precision and proceeded to test for sustainability of different productive configurations. We found that, for a flow rate of 0.01 cubic meters/second, a 0.3 meter radius pipe is best, and that an underground configuration of pipes in a horizontally zigzagging pattern is most practical by a small margin. These results may be generalized to predict that every flow rate has an optimal radius, and that horizontal patterns that do not block the lower heat source are preferable.

Introduction

As the use of petroleum and coal continues on a decline, the world needs to find alternative methods of cheap energy. The United States Department of Energy, privatized companies in America, est. explore alternative methods of energy production such as nuclear, wind, and solar energy. However America spends (comparatively) little effort to profit from the raw power from the Earth's crust. The United States of America is the largest producer of geothermal power [13]; however, all that electricity generated can only account for about 0.3% of the total yearly consumption [10]. Countries like the Philippines, Costa Rica, New Zealand, El Salvador, and Kenya all use geothermal energy to produce over 10% of their countries national production [10]. However the country that stands out as the flagship for geothermal energy is Iceland with about 30% if their energy coming from local hot springs [10]. While some environmentalists may claim that geothermal plants have taken away from the landscape of Iceland, it is no secret geothermal plants – surprisingly – take up significantly less space than coal plants and wind farms that produce the same quantity of watts. On another note, Iceland's – and geothermal plants in general – are considered beautiful. They release steam and traces of minerals and gasses trapped in the rock which leave virtually no ecological footprint [1]. Geothermal Energy is also considered to be a very stable power source, and contributes to a stable economy overall. The only downside to geothermal plants is that there is a potential for seismic activity due to tapping into the energy in the earth's core. Sweden had a problem when they began to use a geothermal plant. The earthquakes experienced were 3.4 on the Richter scale [5]. This disastrous instance is just by far the minority however. Countries like the Philippines continue to utilize earth safely and effectively. The Philippines's hydroelectric plants and geothermal plants are used for the majority of electricity used in the country. On the contrary, America has many locations for geothermal plants not utilized. Areas like the Jemez of Northern New Mexico, some parts of Lower New Mexico and Arizona, many locations along the entire west coast, Hawaii, Alaska, and much to all of the state of Nevada and Western Utah are all suitable for installation of a geothermal plant [12]. The United States is coming to a point where if it does not do anything to change its reliance on fossil fuels, the American way of life will be a fossil – one that could not survive the test of time. Some more information on this subject is that America has the ability to radically reform its energy reliance. The money spent on the war in Iraq alone is enough to invest in alternative energy enough to rid ourselves of reliance on petroleum [7]. Geothermal is arguably the cleanest alternative energy for its energy output, and is still not extensively modeled for several factors.





Before we continue, however, it is important to note exactly how a modern geothermal plant works (excluding those that take hot water directly from existing natural sources). It begins with cold water that flows down into a reservoir or deeper series of pipes, absorbing heat until it is either returned to the surface for use as hot water, or until it is turned into steam for use in a generator (we are primarily concerned with the latter use). Then the pressure from the steam is used to spin a turbine, returning the steam into a cooling tower where it is reused. The turbine spins a coil of wire in a magnetic field, producing electricity that goes out through the power lines. The most efficient geothermal plants operate at 40% efficiency [15], and improving this was our original goal. However, we realized that our model could not estimate efficiency, which has little to do with thermodynamics, but would only tell whether the pipeline produced the proper amount of steam. With the new goal of providing this necessary quantity of steam with the minimal use of pipe, we completed our simulation.

To fully understand and prove why something, like a geothermal plant, works the way it does and easily experiment to minimize cost while maintaining output, one must create a computer model. In our supercomputing project we sought to obtain an output of 10 kg/s steam with the simplest configuration of pipes possible by utilizing thermodynamics equations. We made a graphics portion of the project to show how the presence of the pipe affected the temperature of the rock around after water begins to flow through, and to visualize the water as it flows through the pipe. In the end the computer model was accurate to the equations we used and successfully modeled the thermodynamics involved in geothermal plants.

Description

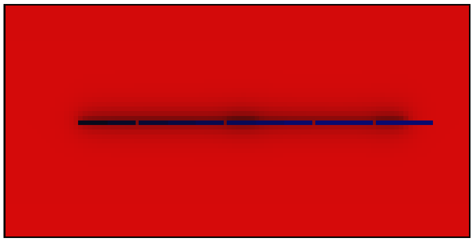
The basic concept of our Java program: chunks of water go into the hot ground, warming up as the chunks of Rock around it cools. At the end of its Pipeline, this fluid disappears, used by the geothermal plant. Our idea was to measure the decay in energy output from the ground for a relatively small model, changing only the variables of concern to us, pipe configuration and radius. The goal was to find the simplest sort of route to build that will produce sustainable energy. The code we wrote to solve this problem is primarily a mathematical model, cycling through thousands of iterations in a grid of rock. In some respect, it is also agent based, because Rock and Fluid interact with each other. The whole process, however, is much more complex than this simple ideal.

I will explain the steps our code goes through as it begins. The first command in the main class is to declare a Pipe, giving its pressure, radius, starting coordinates, ending coordinates, and flow rate (cubic meters/second). Pressure was originally of concern to us, but then we realized that affecting the pressure would be practically impossible. After this construction, pipe length is calculated for efficiency's sake in future computations. More Pipes may be declared if the Pipeline is to be nonlinear, and fewer variables are required for the constructors of these as much of their data is already known by the first pipe. We only concerned ourselves with the pipes at depth, since the ones that take water to that depth and return it would not vary much. Then a Pipeline is created based on these Pipes, so that Fluids moving along each Pipe know where to go once they reach the end.

Next a Grid is produced, given an x dimension, starting depth (best at about 5.5 km underground [2]), y dimension after that depth, z dimension (z goes into the screen and needs not be more than 10 blocks for any sort of computation), dp, and dt. The variable dp measures the smallest unit of distance for the chunks of the Grid, and dt measures the smallest unit of time that can pass. At this point, we may create a JFrame for the grid so that we can visualize it. This is, however, only to verify the functionality of the code, and is not necessary for data collection. Then the Grid's Populate() function is called, filling it with Rocks at each (x,y,z) coordinate, setting each Rock's temperature based on the depth (approximately (298 + 0.0292*depth in meters) Kelvin [2]), giving each Rock a specific heat of 840 J/(kg * K) [14], and having each Rock pre-calculate its neighbors for efficiency's sake. Rocks along the edges of the Grid get virtually infinite specific heat, as we must draw the Grid's end at some point and these Rocks are practically unaffected by the distant cold Pipeline.

Then that Pipeline, which we have already assigned, is added to the grid. At this point, to make calculations simpler, each Pipe in the Pipeline divides itself into Slots, each of which pre-computes its neighboring Rocks and the amount of surface area shared with each. The surface area calculation is perfectly exact, based on the equations of the Rock and of the circular cross section of the Pipe. It would still, however, be impractical to try this on each rock only to discover than nearly every Rock does not touch the Slot. Therefore we use a getNearbies() method to narrow the search down to about 12 – 30 Rocks, depending on the Slot's size.

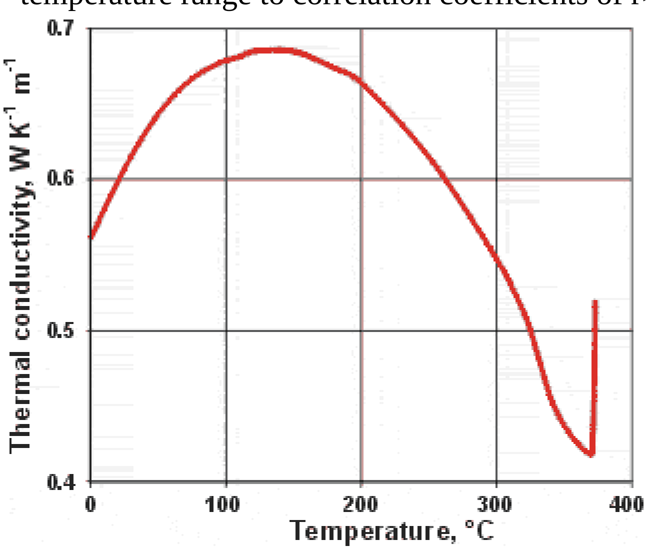
Then the graphics come into play, drawing the initial setting onto a Panel. Various buttons and slide-bars pop up that occasionally help in data collection. The panel adjusts to match the Grid's size, displaying the XY plane that goes through the middle z coordinate. Then This gets painted onto the JFrame that was initialized earlier, and begins to update as the next and most important step of the process starts.



At left is a sample image from one run of our code; the black and blue dashed lines in the middle indicate Fluids, and the red surrounding it indicates Rock. Brightness increases with temperature for both Fluid and Rock. Notice the darker red nearer to the Pipeline.

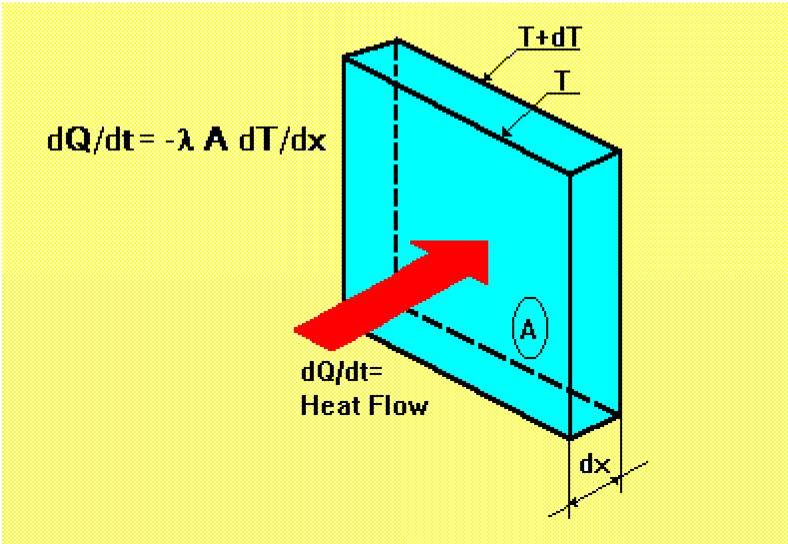
The bulk of the CPU usage begins as the Fluids and Rocks interact among each other. One small loop in the main class runs until the virtual time, as measured by the Grid, reaches a limit we set. Typically this limit may range between 1 second and 15 hours. For each run through the loop, the whole Grid takes one step and the graphics update. In this step, the first thing to happen is the creation of a new Fluid (always water) in each Pipeline (we only ever use one Pipeline, but the code is generalized). The Fluid is given its starting temperature (288 Kelvin typically), the rate of flow in the Pipeline, its "height" (the Fluid is theoretically a little cylinder in a Pipe), how much pressure it is under, and which Pipeline and Grid it is in. From this, it calculates everything else it needs to know, such as boiling point. Then every Fluid in the Grid moves according to its calculated information and dt, moving to the next Pipe in the Pipeline if necessary, and removing itself from the Grid if it has reached the end. It calculates which of the Pipe's Slots it is closest to, based solely on its position variable, which resets each time it moves onto a different Pipe.

Then each Fluid gains heat from the hot Rocks around it, computing its energy change according to Newton's Law of Cooling (and Heating) for each Rock, which involves dt, the water's thermal conductivity (a function of temperature), the difference in temperature between Rock and Water, the shared surface area (remembered by the Slot), the radius, a constant of 0.023, the Reynold's number to the 0.8th power (a function of radius, velocity, density, and viscosity, which varies with temperature), and the Prandlt number to the 0.4th power (a very complicated function of temperature) [11]. For the thermal conductivity, viscosity, and Prandlt number, we were able to compute various regressions from data tables to match the points in our temperature range to correlation coefficients of r>0.99.



Here we used a linear regression over T=0C to T=100C [3].

Our code also includes a similar calculation for losing heat to rock. These equations, developed by engineers for Fluids moving through a Pipe, are the most complicated ones we needed to research for our project. Once the Fluid knows its new total heat, it calculates its temperature, and, if hot enough, its process through the vaporization process. A proportion variable (ranging from 0, steam, to 1, liquid water) allows us to account for this in the equation. Then all the Rocks transfer heat among themselves. Fourier's Law gives us a much nicer equation for this, only involving dt, side length, difference in temperature between rocks, and thermal conductivity [] (which we choose to leave at 7 W/(m * K), common of heavy rocks [4]). To be more



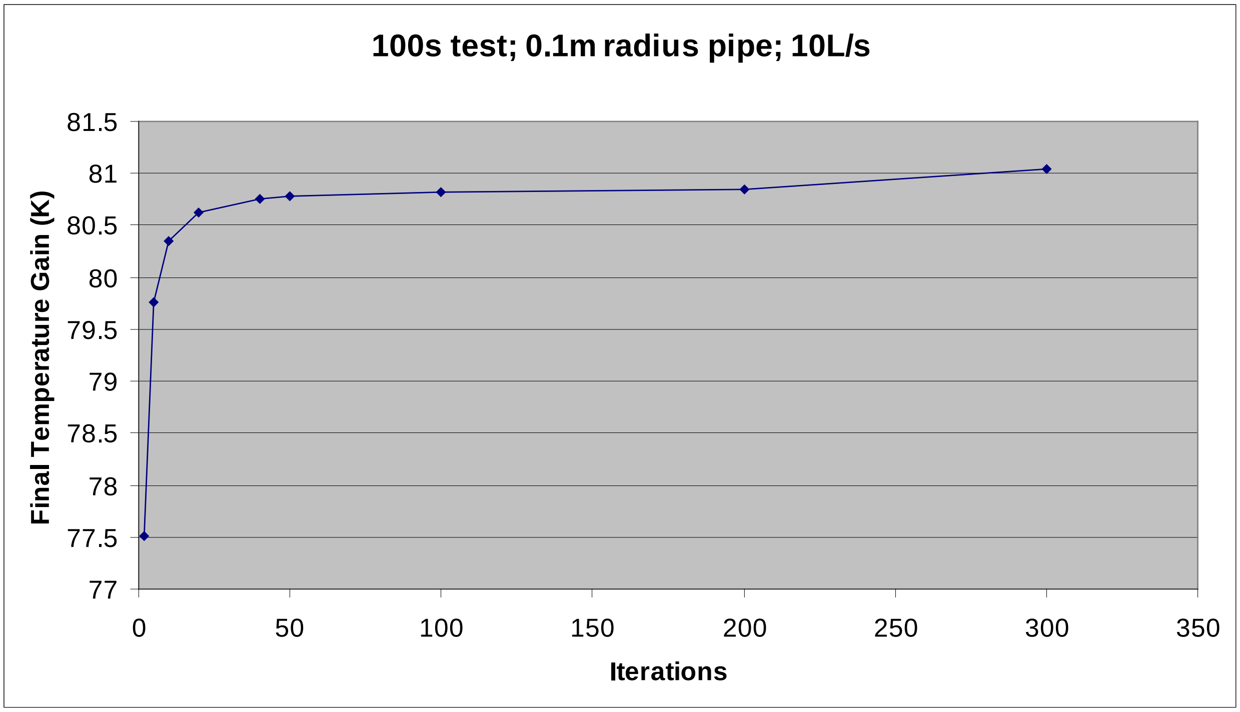
Fourier's Law. For our program, dT=temperature difference between rocks, and dx=dp.

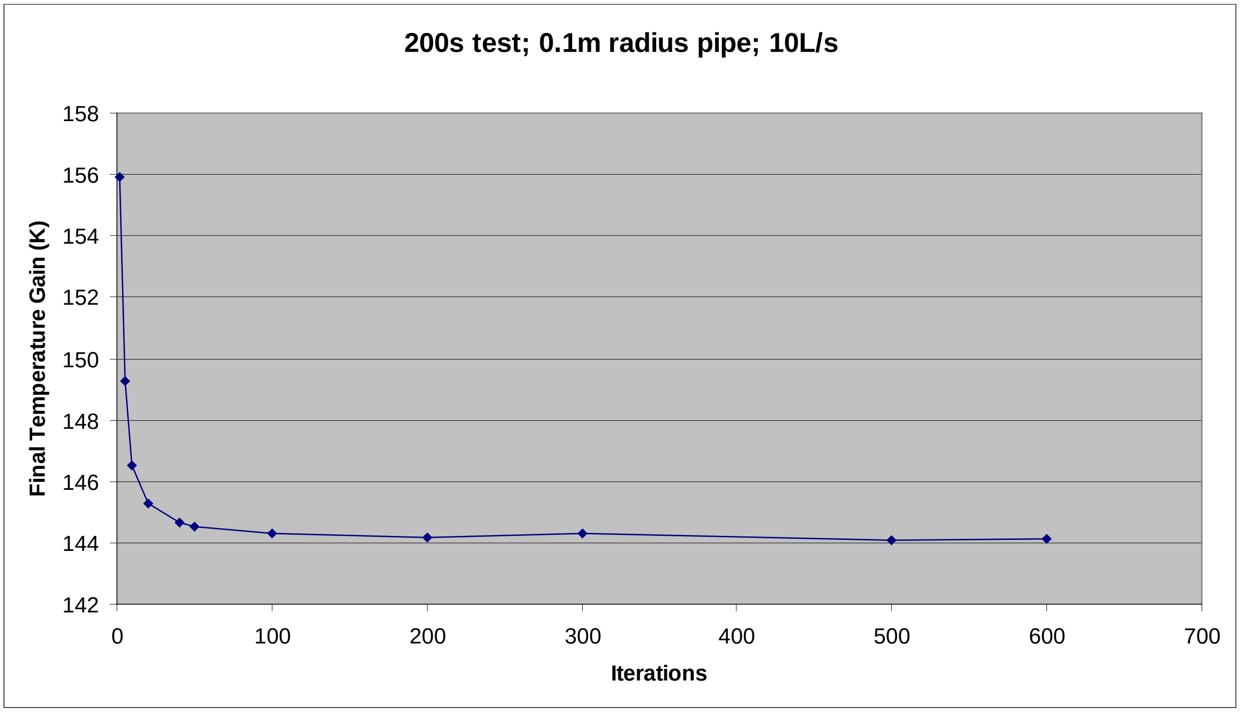
precise, we calculated each Rock's gain in temperature before applying it so that changes would not affect each other. Finally, the Grid increments the time elapsed by dt.

This is the gist of the code's methodology, but we collected data in several ways. At first, we simply measured the temperature gained by a small Fluid in a small period of time. As we grew more confident about the abilities of our code, we implemented graphics to visualize the effects within the Rock structure and the gradual loss of heat output to the water, over several-hour time spans. To collect real numerical data on this output decay, we measured the heat put into a chunk of Fluid at a specific point in the Pipe at different times (for instance, checking how close the Fluid 510 meters along is to vaporization at time=10000s, 20000s, etc.) Finally, we measured the minimal heat output (output once the Rock bordering a Slot is in equilibrium, gaining as much temperature from the Rock below as it loses to the Fluid) for various configurations and radii. This allowed us to determine the most efficient radius for a given flow rate and Pipe length (larger radius means more energy gained total, but is less practical to construct) and the most permanent setup of pipes in relation to each other (the Pipeline must return at nearly the same position it started at, so there are limitations to how it can be oriented). The geothermal system in Reykjavik uses 0.45m radius pipes that lead down to a reservoir to heat up; essentially a Pipe of extremely large radius, so we had our starting point [8].

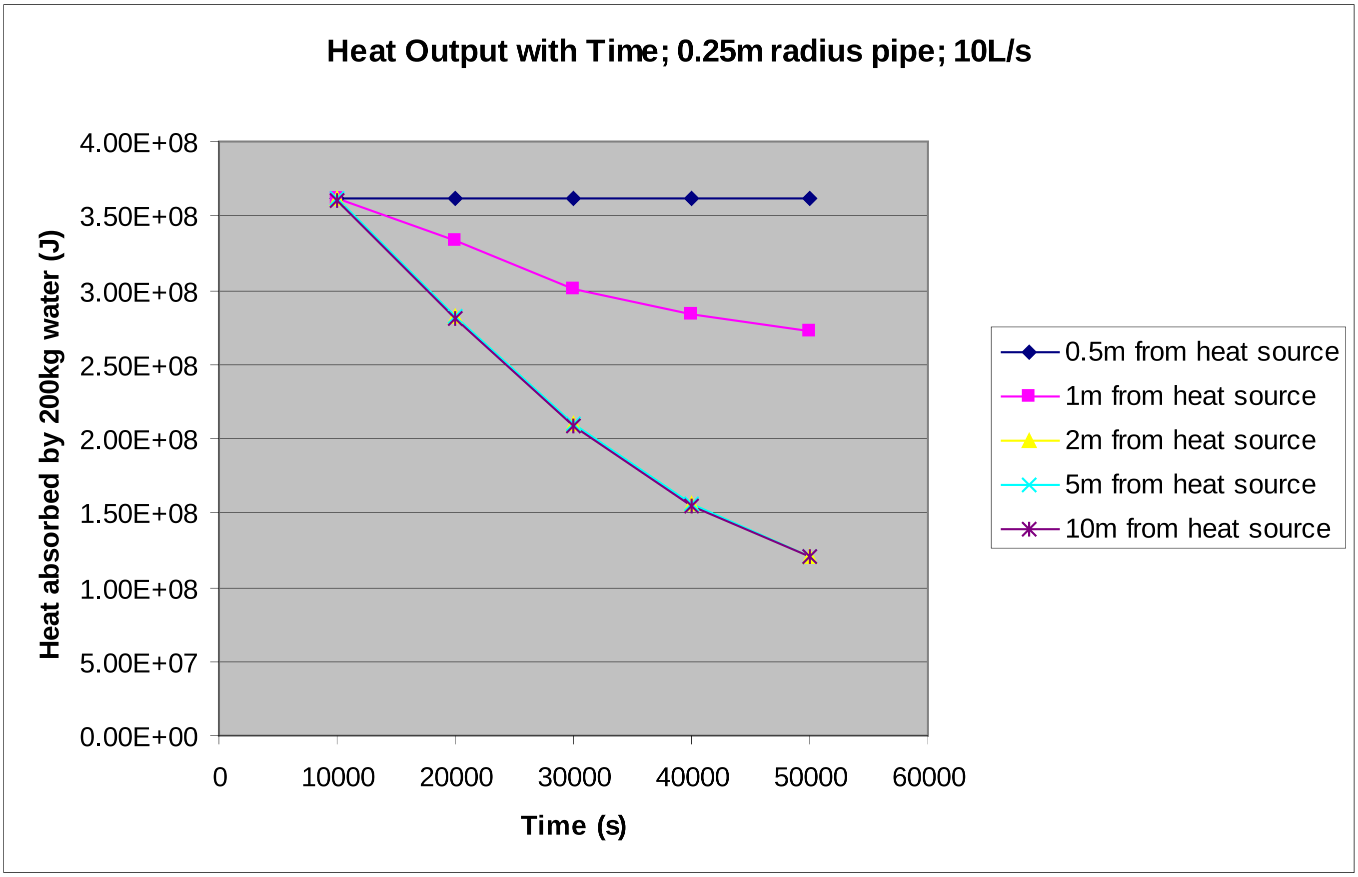
Results

At first, we aimed our code a verifying itself. There were countless hitches at first, but eventually we got it to exactly match the theoretical data from a one-second interval as a 0.1m radius pipe runs at 0.01 cubic meters/second, producing a temperature increase of 0.728 Kelvin. This theoretical data is known to fall within a few percent of reality. From there, we extrapolated for somewhat longer intervals, watching as temperature converged to a specific value for an increasing number of iterations for a long test:

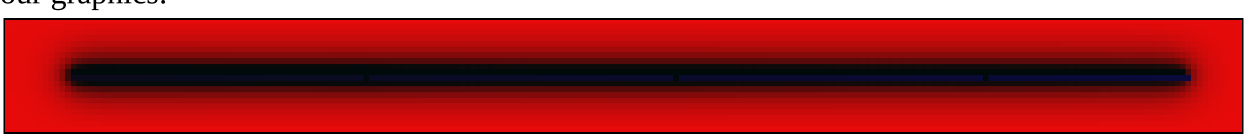




Notice that even for 2 iterations of 100 seconds each, the error was barely even 8%, and that 10 iterations of 10 seconds produced less than 1% error. After numerous tests, we decided that error increases both with dt and with the inverse of the number of iterations total, so scaling time would mean that larger dt intervals could be used. Even so, we decided not to use intervals of dt>20 seconds so that we could be precise in our knowledge of further results. There is clearly some bound beyond which you should not go, as too long a time interval could mean transferring so much heat into the Fluid that it becomes hotter than the surrounding Rock. So next we tested the necessary distance from the heat source (the Grid's edge):

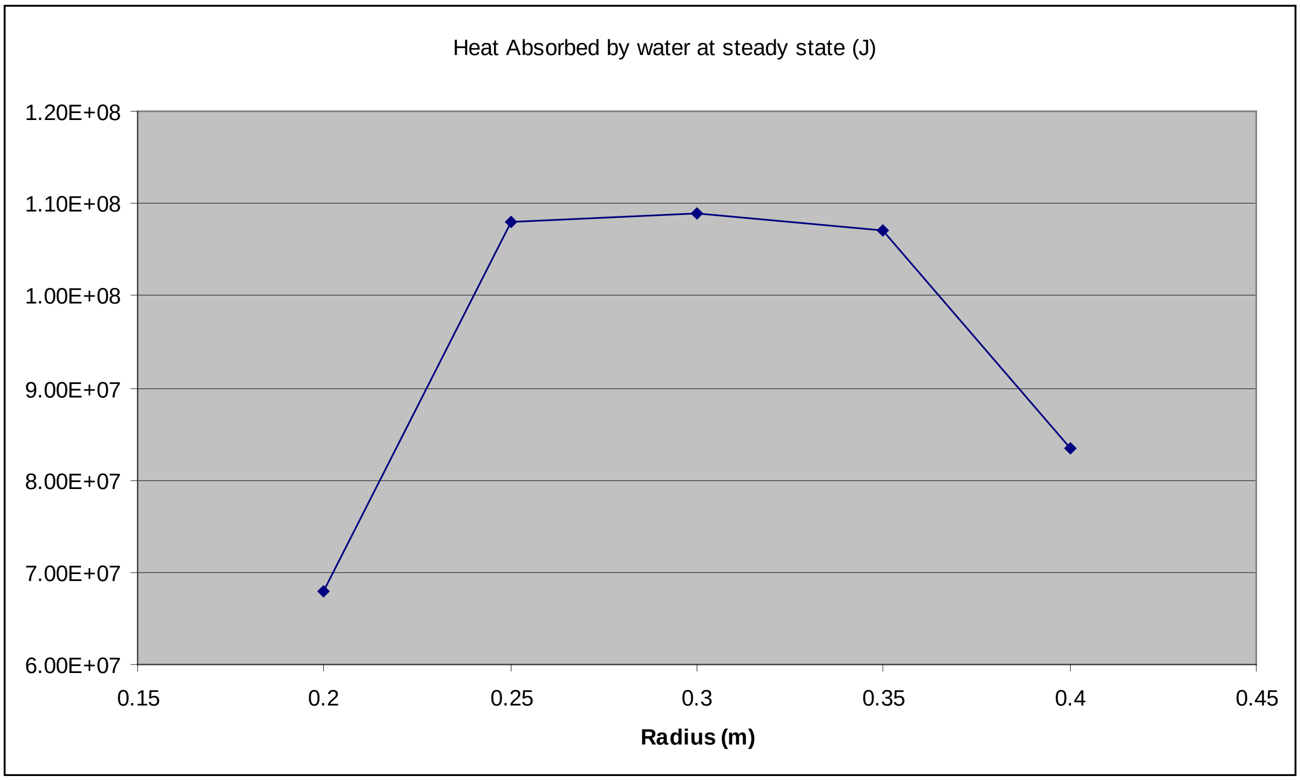


There are two conclusions one may draw: first, that distances of 2 meters or greater are practically unaffected by their proximity to the heat source (for this particular setup), and second, that this test (14 virtual hours) was not long enough to be affected by deeper conditions. At first glance, these seemed good, but in reality, the latter indicated that our test needed a change before it could address the slow depletion that some geothermal plants develop over months. Thus, we devised a clever way to reach such conditions more quickly: hold the Rock's specific heat at a lower value for a good part of the test, then return it to normal before collecting data. This would drastically increase the effect each Rock has on its neighbors, so we tried it, watching with our graphics:



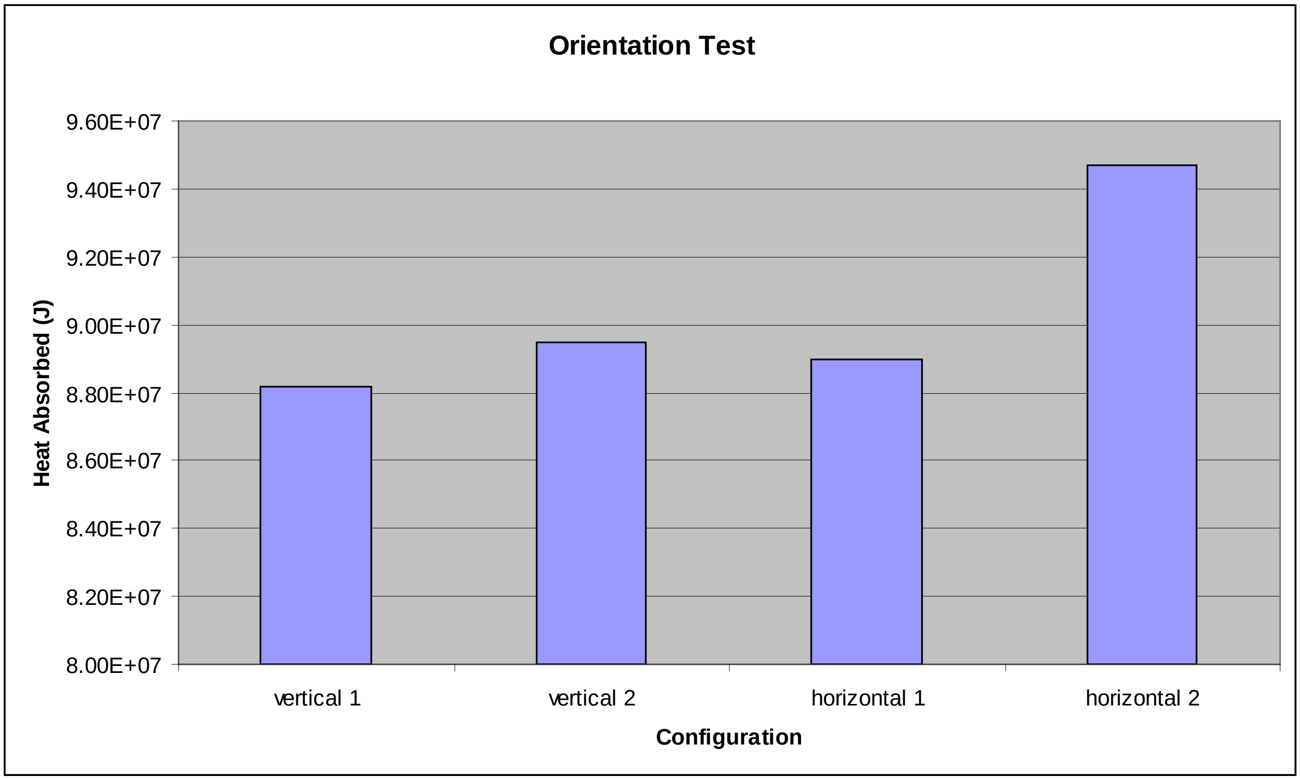
While the original test penetrated about 1one meter of distance, this new method allowed us to penetrate 7-8, which indicates that $(7\text{-}8)^2$ as much time has (equivalently) passed – over one month, instead of 15 hours! What's more, this method approaches the same equilibrium that would normally occur, since specific heat does not amount of heat transferred, only the temperature drop; the Rocks bordering the water still lose as much energy from the Fluid and gain as much from other Rocks as if specific heat were accurate. So, when we return to normal specific heat we are able to collect data. The Pipeline shown above was 200 meters, and we next made a 1,600 meter Pipeline for the conclusive radius test.

This test would run water through a pipeline of varying radius until it is sure to have reached an almost steady state (this time increases with radius, as it takes longer for water to reach the end of a wider Pipe). Once there, the specific heat would switch back and the heat and proportion of a chunk of water moving to the end would be measured. From these numbers, we calculated the energy absorbed to gauge the practicality of the Pipe. We expected this to simply increase with radius, but were shocked to find the following result:



We found a simple, relative maximum near a radius of 0.3 meters. Looking back onto the expanded form of Newton's Law of Cooling (and Heating) we used, a larger radius will indeed cause some factors to drop, mostly because of the reduction in convection caused by low speeds within the Fluid. Happy with this explanation, we moved on.

The next test involved using four pipes of equal length in four different configurations. The three we tested were the short vertical zigzag (where the parallel pipes are close together), the long vertical zigzag (where they are far apart), the short horizontal zigzag (where they are parallel in the XZ plane instead of XY), and the long horizontal zigzag. We wholly expected the long horizontal one to have the highest output in the end, as its pipes would not inhibit each others' access to the lower heat source. Our expectations were confirmed, but not as dramatically as we had hoped:



Notice that each value is within 8% of the others. No setup seems remarkably better than any other. Nevertheless, one must change the theory to meet the facts and not the other way around. The lower heat source is, after all, only very slightly hotter than the other edges of the Grid, and that is compensated for by the slightly colder upper heat source that, as one might view it, is blocked by the vertical configurations. Thus, the XZ plane must be hardly any better than the XY plane. We had our final results.

Conclusions

It seems that pipeline configuration does indeed make a difference in sustainability and practicality when choosing a geothermal plant. This is the culmination of our labor – from conception to research to beginnings to running code to working code to results. Experimentation and calculation has found the optimum depth [2], location [9], and method of distribution [8] for a geothermal pipeline and its product, but no one before has calculated the ideal radius of such a line. Our data was, however, constricted to a 0.01 cubic meter/second flow rate, and does not offer that ideal radius for any other value. Plus, only a few configurations were tried. Nonetheless, we can make a few conclusions with confidence:

(1) Our code approximated a real-world situation. Many geothermal plants have very similar setups to the ones we tried, and our equations told us we got the right answer for simple tests. The percent error, as we mentioned earlier, decreases with virtual time, so we may extrapolate this to say that our data were not too far off from reality.

(2) There is an ideal radius for a given length of pipe with a given flow rate. A few other, less thorough tests we have done state that the length of the pipe barely makes a difference, as long as the flow is just quick enough to be convective and match our equations. Thus, the quickest way to boil water in a geothermal pipeline is through an optimal radius of the sort our computer program can compute through trial and error.

(3) Some configurations are better than others. Pipes make the rock around themselves cold, so they should be placed far enough apart so as not to affect each other much. We saw this with the difference in output even between the short and long configurations we tried (still separated by as much as 5 and 7 meters, respectively). Much closer and there would be a bigger discrepancy. Plus, each pipe should border, in some way, the heat source below. These two sub-factors make a small but noticeable difference

The final tests of our code did not actually boil the water; based on how much water it did absorb, though, a pipeline of about 10km underground length would be necessary. This is not unreasonable, given that the total length of Reykjavik's pipelines is 2,157km [8].

So one thing is for certain: we could have used more time to do testing.

Recommendations

There were definitely at least two sources of error in our project. First of all, the equation used for Newton's Law of Cooling (and Heating) is partly based on a correlation that can vary by up to 15% from the true value [11]. Second, our code was limited by a CPU's ability; it would have been impractical to do the test on a finer scale of dp (we left it at 1 meter). To make the scale twice as fine would mean almost 8 times as much computation. However, these issues mean almost nothing to our results; the numbers indicate that one radius is better than another, regardless of whether or not both are 15% too much.

Leaving accuracy behind, there are several improvements that could be made to the code to make it more efficient on a larger scale. Given more time the code is easily made parallel. By cutting each step of dt time into several pieces, the work could be distributed. Then one processor could easily fit the pieces together. Five processors would not mean five times as swift computation, but there would definitely be improvement. Supercomputers are the of tests like ours that might benefit from a finer resolution of Grid. In addition the graphics could also benefit from using an open source parallel visualization program. Open source parallel visualization programs like ParaView have been used to visualize 3D data sets such as our grid. If we had the opportunity for expansion, this is undoubtedly the route we would take to make both the code and the graphics more efficient.

Software

The bulk of how our code works has already been elaborated upon, so we will go into further depth about graphics and a few other details.

Computer graphics are an essential component to display results and produce results that are available to the public. Data visualization shows the program as it runs through time steps, giving both insight on how the programs runs and a nice display of what is actually happening. Visualization is meant to be interactive by nature (such as: going through time steps and manipulating variables). In our code we used JFrames (after importing java.swing.*) to create a window where a slice of our grid could be represented from a 2D perspective. To do this we set the z axis halfway through the grid in order to draw the middle of the grid with the pipe and capture the most water. Then we go through the Grid and draw first all the layers of rocks – colored by temperature (adjusted to be somewhere between 0 and 255). After the rock is drawn we draw the pipe and color it dark to light blue based on temperature. What this showed us, is how the water goes through the pipe and heats up over time. It also shows that the rock around the pipe is cooling how it is supposed to. This enables the programmer to actually see the effects the geothermal plant has on the surrounding rock after it has been in use for lengthy periods of time. JButtons, JTextField and JSliders (after importing java.swing.*) are easy to manipulate variables in order to get instant results. Without a graphics element, it would take longer to try out different ranges of variables while looking for a particular result. If we needed to look at a range of variables all we would do is set the ActionListener for each JSlider, JTextField or JButton to manipulate a starting variable and use the repaint() method. Data visualization is useful for double checking that the model is working correctly, creating data that can be shared with more than just the programmers, and to more easily scroll through a range of variable settings.

We have written nearly 2,000 lines of code for this project on two files (graphics is separate). Including lines deleted and replaced for the many different tests, the count is surely over 2,000. Our original code would give a HeapSpace error for using too much of the computer’s memory, but since then we have greatly trimmed down its usage. The run time for 3,200 iterations on a 162,000 block Grid is under 3 minutes.

Achievements

Before this project, we researched other work on geothermal energy (sources [2], [4], [8], [9], [15]), but could find nothing on the actual underground configuration and sizing of pipes. Optimal depth and location selection have been understood for many years, but efforts to sustain this energy have sometimes failed [2]. The error isn't in the location; it is, at least partially, in the pipe. Our project has done something entirely original by proving that conclusion. Regardless of sources of error, the data consistently show there is an ideal radius for a given flow rate. Furthermore, pipeline configuration should to be considered when planning a new geothermal plant. Computer simulations will be the future of many aspects of energy production, and this is one look forward into our society's progress in that direction.

Simply having a program working without bugs is satisfying. Knowing that it stays fairly accurate to what is seeks to simulate is even better. And hoping that someday, someone might be able to learn from your research is sublime. This is our hope with this project.

Acknowledgements

First, we would like to sincerely thank our teacher sponsor Leroy Goodwin for watching over our project as it developed and giving us a pace to go by. He deserves our respect for helping with something he has no stake in.

On a more sarcastic note, we would like to insincerely thank our prior colleagues Allen Wu and Thomas Liu for claiming they would do work. They bolstered our hopes before admitting each time that they had done nothing. We recently removed them from the team for promising to write half the report and to do research weeks ago, and only on April 3rd confessing that they had actually done nothing. We maintained frequent contact with them, but leadership does not work on unmotivated people when one can offer no punishment or reward.

Bibliography

[1] Bertani, Ruggero and Thain, Ian. "Geothermal Power Generating Plant CO2 Emission Survey." *IGA News(International Geothermal Association)* July, 2002. (49): 1–3, retrieved 2009-05-13

[2] Brady, Robert; Ducea, Mihai; Kidder, Steven; and Saleeby, Jason. “The Distribution of Radiogenic Heat Production as a Function of Depth...” *University of Arizona Geosciences*. 26 July, 2004. <http://www.geo.arizona.edu/tectonics/Ducea/publications/Bradyetal.pdf>.

[3] Chaplin, Martin. “Explanation of Thermodynamic Anomalies of Water.” *Water Structure and Science*. 11 August, 2009. <http://www.btinternet.com/~martin.chaplin/explan4.html>.

[4] Clauser, Christoph and Huenges, Ernst. “Thermal Conductivity of Rocks and Minerals.” *Geophysik*. 1995. <http://www.geophysik.rwth-aachen.de/Downloads/pdf/cl_hu_1995_s.pdf>.

[5] Deichmann, N. et al. “Seismicity Induced by Water Injection for Geothermal Reservoir Stimulation 5 km Below the City of Basel, Switzerland.” *American Geophysical Union*. July, 2007.

[6] Elert, Glenn. “Conduction – The Physics Hypertextbook.” *The Physics Hypertextbook*. 1998-2010. <http://physics.info/conduction/>.

[7]Gordon, Jason. “How Much Money is Being Spent on the war in Iraq”. *E-How*. <http://www.ehow.com/facts_5042946_much-money-being-spent-iraq.html>.

[8] Lund, John. “Hitaveita Reykjavikur and the Nesjavellir Geothermal Co-Generation Power Plant.” *Geo-Heat Center*. June, 2005. <http://geoheat.oit.edu/bulletin/bull26-2/art5.pdf>.

[9] Jonsson, Kjaernested and Palsson, H. “A Methodology for Optimal Geothermal Pipeline Route Selection...” *Stanford School of Earth Sciences*. February, 2011. <http://pangea.stanford.edu/ERE/pdf/IGAstandard/SGW/2011/kjaernested.pdf>.

[10] McLarty, Lynn; Reed, Marshall J. "The U.S. Geothermal Industry: Three Decades of Growth.” *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*. October, 1992. London: Taylor & Francis.

[11] Nahle, Nasif. “Heat Transfer, Conduction, Convection and Radiation.” *Biology Cabinet*. April, 2006. <http://www.biocab.org/Heat_Transfer.html>.

[12] (no author given). “Geothermal Education Office”. *Geo*. October 12, 1977. <http://geothermal.marin.org/map/usa.html>.

[13](no author given). “Geothermal Energy: International Market Update .“ *Geothermal Energy Association*. May 2010. p. 7.

[14] (no author given). “Solids – Specific Heat Capacities.” *The Engineering Toolbox*. <http://www.engineeringtoolbox.com/specific-heat-solids-d_154.html>.

[15] Rafferty, Kevin. “Geothermal Power Generation”. *Geo-Heat Center*. January, 2000. < http://geoheat.oit.edu/pdf/powergen.pdf>.