HIGH SCHOOL EDITION
Journal of Student Research
Volume 12 Issue 4 (2023)

Modeling the Water Temperature Profile of Lakes: A Physics Informed Neural Network (PINN) Approach

Audrey Creighton

Lexington High School

ABSTRACT

Water temperature plays an important role in our environment and is applicable to nearly all limnology research, as the temperature of a body of water affects biological activity and growth of organisms such as algae and bacteria. Certain organisms have a preferred temperature range within which they can survive, while others become dormant or die when the water reaches extreme temperatures. The temperature of water also governs the maximum dissolved oxygen concentration of water. Dissolved oxygen in water is important for aquatic life because of its vital role in cellular respiration. Predicting water temperature is also an important factor in determining whether a body of water is acceptable for human use. Warm bodies of water may contain pathogens that can be dangerous to humans. Our research presents a computational method of determining lake water temperature through a novel technique known as physics-informed neural networks (PINNs). PINNs can be used to model and forecast the temperature of water over a specific time period by training a neural network using the data points derived from the discrete form of a partial differential equation and taking into account the boundary conditions. Several factors such as wind, precipitation, and solar energy effects on water temperature were investigated. By using a computer simulation in place of an analytical mathematical model, a tremendous increase in run time speed can be achieved. The results can be used to determine the patterns in water temperature throughout a year, demonstrating the advantages of a PINN over an analytical model.

Introduction

Computer simulations are important tools in many areas of research. Generally, computer simulations are accomplished using traditional numerical methods, such as the finite difference method, finite element method, or computational fluid dynamics. While these methods are accurate, they can be slow and time-consuming. Traditional methods use a process called “time stepping” that requires the solution to be computed one time step at a time in chronological order. For example, if we wanted to compute the solution at time 1000 seconds, we must know the solution at time 0.1 seconds, time 0.2 seconds, time 0.3 seconds, all the way up to time 1000 seconds. Recently, scientists have found that one way to make computer models faster is to solve them using neural networks instead of traditional numerical methods. What makes neural networks so much faster is that they can jump ahead to any time step within their training domain. This means that, with a neural network, we can compute the solution at time 1000 seconds as quickly as you can compute the solution at time 0.1 second.

In this research, our goal is to use a neural network to solve for the temperature profile of a lake. The neural network can compute the temperature profile at any time of the year. Additionally, the neural network can compute the solution magnitudes faster than a traditional numerical solver. Our research aims to prove the ability of neural networks to produce practical and accurate computer models that are amongst the fastest in the world.

Methods

In order to train a neural network to predict the temperature profile of a lake, we conduct two major processes:

Method 1: Build a custom, discretized 3D mathematical model of transient heating using partial differential equations to model the temperature profile of the lake;

Method 2: Subsequently, apply the resulting data and use it to train a neural network. The end result will be a neural network that performs nearly the exact same computations as an analytical mathematical model but with a tremendous speed advantage.

A. Method 1 of 2: Build a mathematical model of the transient heating of a lake in order to collect collocation data.

Step 1: Represent the lake as discrete points using Microsoft Excel CSV

To determine the transient heating inside of a lake, we will convert our lake profile into a 2D grid with 51 rows and 151 columns. We will convert the continuous nature of the lake profile into discrete points. That way, we only compute the temperature at a finite number of points.

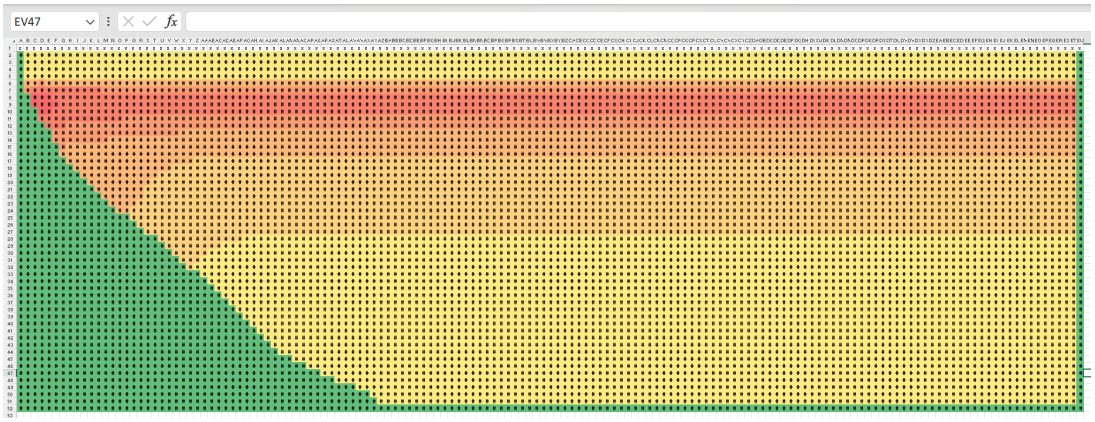


Fig. 1 Outline and initial temperature profile of a “deep” lake

The image, which was created in a Microsoft Excel CSV file, consists of 51 rows (excluding the header) and 151 columns. Thus, we have broken our environment into 51*151 = 7701 discrete points. However, the 1183 green-colored cells represent the area that lies outside of the lake, i.e., rocks and soil. This means that our lake is broken into 7701-1183 = 6518 discrete points. The green cells to the left and bottom of the image represent earth material while the green cells to the right of the image represent an axis of symmetry. Establishing an axis of symmetry is a mathematical trick that allows us to simulate only half the lake, thereby reducing the needed number of computations.

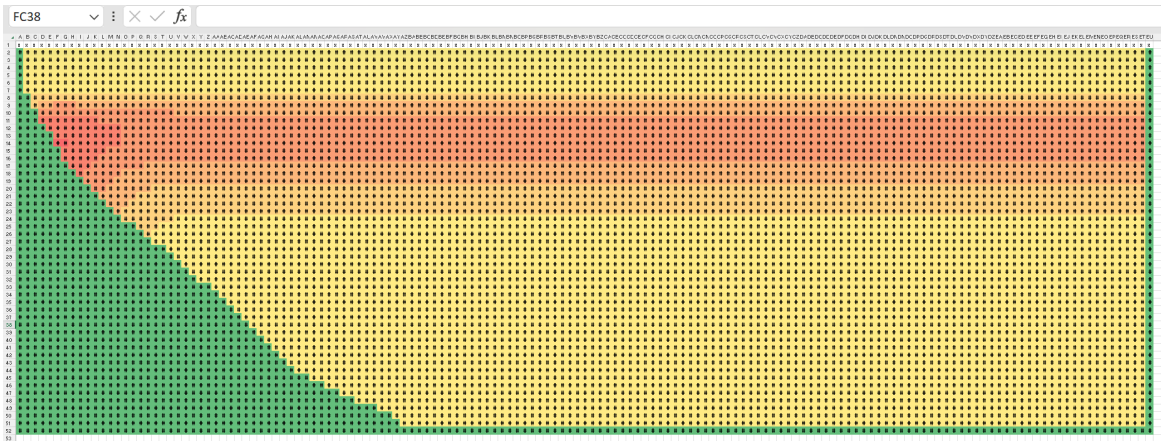


Fig. 2 . Outline and initial temperature profile of a "shallow lake."

Again, the image consists of 51 rows (excluding the header) and 151 columns. The outline of the "shallow lake" is exactly the same as that for the "deep lake." The only difference is in the initial temperature profile of the lake. Because the lake is shallow (meaning that the differential spacing in the y-direction is smaller), the red band of warmer water is more spread out in the shallow lake profile than in the deep lake profile.

Step 2: Derive a formula to compute the temperature at every discrete point in the *interior* of our lake.

There is a partial differential equation known as the transient heat equation that represents the transient heating of a continuous material. (Pina, H.L.G., Fernandes, J.L.M. 1984) That formula is given here:

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}+\frac{\dot{q}}{k}=\frac{\rho C_p}{k}+\frac{\partial T}{\partial t}$$

This equation illustrates how the temperature of any point in 3D space varies according to the spatial coordinates x, y, and z, and the temporal coordinate t. However, our lake profile is 2D (which is equivalent to prismatic 3D), so we can drop the z-axis. This equation also includes a heat source q, which the *interior* of our lake does not have. Thus, for the *interior* points of our lake, we will use the following modified equation:

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}=\frac{\rho C_p}{k}+\frac{\partial \partial^2 T}{\partial t}$$

This equation is in a "continuous" form. Since we "discretized" our lake into discrete points on a 51 x 151 grid, we will need to replace the continuous form of the transient heat equation with a discrete form. This size grid was selected in order to properly display the lake proportions, while minimizing the number of points to make sure the neurological network would not be overloaded. We accomplish this by replacing every spatial and time derivative with its finite difference form.

We insert the finite difference form of each term into the transient heat equation (Recktenwald, 2004), setting $\Delta x=\Delta y$, and rearranging, we obtain:

$$T_{i,j}^{k+1}=T_{i,j}^{k}+\Delta t\cdot\frac{k}{p\cdot c_p}\cdot\frac{1}{(\Delta X^2)}[T_{i-1,j}^{k}+T_{i+1,j}^{k}+T_{i,j-1}^{k}+T_{i,j+1}^{k}-4T_{i,j}^{k}$$

In this equation, the subscript "i" represents the index in the x-direction, the subscript "j" represents the index in the y-direction, and the superscript "k" represents the index in time. Thus, this equation allows us to find the new temperature *k+1* at any point I,j using only the current temperature at time *k*. Specifically, we see that the temperature of any point in space is dependent on the temperature of the points in space around it. Given our point in space, the formula requires that we sum the current temperature at the point in space directly above, below, to the right, and to the left. Then, from this sum, we must subtract four times the current temperature at our point in space. This mathematical process simply represents the heat that a point can gain due to its temperature gradient with the four points around it. This new value is then multiplied by constant: $\frac{k}{p*c_p}*\frac{1}{(x\Delta)^2}$ and Δt and added to the initial value of the temperature. This formula is simplified down from its original form to reflect a case where Δx is equal to Δy.

Step 3: Derive a formula to compute the temperature at every discrete point along the side and bottom edges of our lake.

In this step, we will derive a formula to compute the temperature along the left, right, and bottom edges. To accomplish this, we first have to understand what kind of boundary to apply to the left, right, and bottom edges. We will presume that, for the left and bottom edges of our lake, the heat inside the lake cannot escape. In other words, the left and bottom edges of our lake are well insulated. To prevent the heat from escaping, we apply a first-order zero-flux Neumann boundary to the left and bottom edges:

$$q_s = -k\frac{dT(0,t)}{dx} = 0$$

Because the right edge of the lake is an axis of symmetry, no heat should escape out of the right edge of our lake. So, we can also say that the right edge of our lake is well insulated. Thus, we apply a first-order zero-flux Neumann boundary to the right edge as well.

Step 4: Derive a formula to compute the temperature at every discrete point along the top edge of our lake.

Finding the formula for the discrete points along the top edge of the lake is the most difficult task, because we have a non-zero-flux Neumann boundary. This means that heat is allowed to enter and leave the lake's surface. In other words, heat transfer into and out of the lake is constantly occurring. The amount of heat transfer at any point in time is dependent on numerous weather conditions and other factors. The process to compute the temperature along the top of the lake is as follows.

q_total, the total heat flux into the lake in W/m2, is given by:

$$q_{solar} = q_{solar} + q_{sensible} + q_{latent}$$

q_solar is the solar radiation, q_sensible is the sensible heat flux, and q_latent is the latent heat flux.

q_solar, the solar radiation, combines the short wave radiation q_short and the long wave radiation q_long:

$$q_{solar} = q_{short} \cdot (1 - a) + q_{long}$$

α, the albedo:

$$a = p^{(c \sin b + 1)}$$

where

a = hourly albedo,
c = roughness coefficient,
p = color coefficient,
b = solar angle, in degrees.

H = solar hour angle, degree. (H = 0 at solar noon, 15° per hour deviation from solar noon, "+" in the afternoon, "-" in the morning)

The short-wave and long-wave radiation values are obtained from weather data The color coefficient p and the roughness coefficient c are found using the below table:

Surface and condition	p	c	Average R^2
Lakes and ponds, clear water			
waves, none	0.13	0.29	0.817
waves, ripples up to 1 inch	0.16	0.70	0.741
waves, 1 inch or more with occasional whitecaps	0.23	1.25	0.827
waves, frequent whitecaps	0.30	2.00	0.852
Lakes and ponds, green water,			
waves, ripples up to 1 inch	0.22	0.70	0.902
Lakes and ponds, muddy water			
waves, none	0.19	0.29	0.758

Fig 3. Values for the coefficients p and c (For 0.3 micron to 3-micron wavelengths)

The solar angle B and solar declination angle D are computed using these two formulas, where N is the Julian day:

$$B = \sin^{-1}(\sin D \sin L + \cos D \cos L \cos H)$$
$$D = \sin^{-1}\{0.39797 \cos[0.98563(N - 173)]\}$$

$q_{sensible}$, the sensible heat flux through the lake surface:

$$Q_s = -p_a \cdot c_p \cdot C_s(T_s - T_a)W$$

T_v, the virtual temperature:

$$T_v = T(\frac{1 + \frac{W}{E}}{1 + W})$$

ρ_a, the density of the moist air mass:

$$P_a = \frac{P}{(R_d \times T_v)}$$

ω, the mixing ratio:

$$W = \frac{Ee}{P - e}$$

e_a, the vapor pressure above the lake's surface:

$$e = e_0 \exp\left[\left(\frac{L_v}{R_v}\right)\left(\frac{1}{t_0} - \frac{1}{T_d}\right)\right]$$

L_v, the latent heat of vaporization of water:

$$L_v = 2500297.8 - 2369T$$

$c_{p,a}$, the specific heat of moist air in J/kg-K:

$$C_p = C_{p0}(\frac{1 + W\frac{C_{pv}}{C_{p0}}}{1 + W})$$

c_s, the sensible heat transfer coefficient:

$$C_p = \begin{cases} W < 8\, ms^{-1}: (0.720 + [0.0175\, W\, (T_s - T_a)]) \cdot 10^{-3} \\ W \geq 8\, ms^{-1}: (1.000 + [0.0015\, W\, (T_s - T_a)]) \cdot 10^{-3} \end{cases}$$

q_latent, the latent heat flux through the lake surface:

$$Q_e = -P_a \cdot L_v \cdot C_e(q_s - q_a)W$$

q_s, the specific humidity at the lake surface:

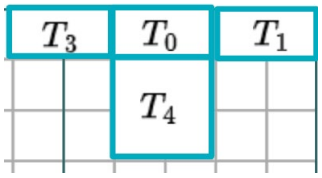
$$q = \frac{Ee}{P - (1 - E)e}$$

C_e, the evaporative heat transfer coefficient:

$$C_e = 1.5 \cdot 10^{-3}$$

Now that we know q_total (the total heat flux into the lake), the temperature of the nodes along the top edge of the lake can be computed using the finite volume method given below:

$$\dot{q_s}, T_\infty = 300, H_c$$



$$K \cdot \frac{T_1 - T_0}{\Delta x}\left(\frac{1}{2}\Delta y \cdot \delta\right) + K \cdot \frac{T_3 - T_0}{\Delta x}\left(\frac{1}{2}\Delta y \cdot \delta\right) + K \cdot \frac{T_4 - T_0}{\Delta y}\left(\frac{1}{2}\Delta x \cdot \delta\right) + q_s(\Delta x \cdot \delta) + h(T_\infty - T_0)(\Delta x \cdot \delta) = P\left(\frac{1}{2} \cdot \Delta x \cdot \Delta y \cdot \delta\right) \cdot \frac{T_0^{new} - T_0}{\Delta t}$$

To use this formula, we only need two last variables: T-infinity and h. T-infinity is just the common notation for the air temperature. So, we will obtain that at 15-minute intervals from our weather data. The last variable, h, is the convective heat transfer coefficient. This is given by:

$$h_c = 10.45 - v + 10^{\frac{1}{2}}$$

In this formula, "v" is just the wind speed (measured at 10 meters above the surface of the lake) in units of meters per second.

Step 5: Find the weather and material data necessary to run the model.

The formulas above require that we obtain certain data. For example, the transient heating equations require that we know the mass density ρ, specific heat capacity c_p, and thermal conductivity k of our lake water. These values, in turn, depend on the salinity of our lake. Assuming that our lake has a salinity of 15%, the aforementioned values will be ρ = 1171.45 kg/m^3, c_p = 3681.75 J/kg-K, and k = 0.60 W/m-K.

Additionally, to create a transient heat modeling for the temperature profile of our lake for a 12-month period, we will need a detailed set of weather data for a 12-month period. So, we downloaded a weather set that has weather data from January 3, 2021, to December 31, 2021, collected at every 15-minute increment. This weather data contains the date, time, dew point temperature, air temperature at 10 meters above the ground, wind speed at 10 meters above the ground, and short-wave solar radiation (q_{short}). We will make the general assumption that the long-wave solar radiation q_{long} is some percentage of q_{short}, e.g., $q_{long} = 0.45*q_{short}$. This assumption is informed by field data showing that the pattern of short-wave and long-wave radiation appear to follow one another, but with the long-wave radiation being a fraction as intense.

Table 1. Weather Data

date_time	td_avg	airt_avg	winds_avg	precip_tb	so-larw_avg	julian	hour	albedo
1/1/2021 0:00	19.4	22.8	2.503424	0	0	1	0	0.228450037
1/3/2021 0:00	18.7	20.9	0.849376	0	0	3	0	0.228338727
1/3/2021 0:15	17.2	19.3	0.312928	0	0	3	0.25	0.228137098

1/3/2021 0:30	16.5	18.4	0.715264	0	0	3	0.5	0.22753414
1/3/2021 0:45	16.8	18.7	0.715264	0	0	3	0.75	0.226535602
1/3/2021 1:00	15.3	17.4	0.268224	0	0	3	1	0.225150948
1/3/2021 1:15	12.5	14.8	0.759968	0	0	3	1.25	0.223393182
1/3/2021 1:30	12.2	14.6	0.357632	0	0	3	1.5	0.221278617
1/3/2021 1:45	12.2	14.6	0.491744	0	0	3	1.75	0.218826576
1/3/2021 2:00	12.9	15.2	0.44704	0	0	3	2	0.216059063
1/3/2021 2:15	12.6	14.9	0.089408	0	0	3	2.25	0.213000386
1/3/2021 2:30	12	14.4	0.625856	0	0	3	2.5	0.209676754
1/3/2021 2:45	11.9	14.3	0.581152	0	0	3	2.75	0.206115864
1/3/2021 3:00	13	15.3	0	0	0	3	3	0.202346477
1/1/2021 0:00	19.4	22.8	2.503424	0	0	1	0	0.228450037

Step 6: Run the mathematical model.

To run our mathematical model, we set the time step to 0.1 seconds. The dataset that we obtained has weather information for 15-minute increments. Thus, for every set of 15-minute weather data, we must run our model for 9000 steps. One year should have (365 days/year) *(24 hours/day) *(4 15-increments/hour) = 35,040 15-minute increments. However, our data only had 34,321 15-minute increments since a few days of data in the month of January were missing. In addition, other data points spread throughout the year were missing. Given that we have 34,321 15-minute increments, and our algorithm computes the new temperature values 9000 times per 15-increment, this means that our algorithm ends up computing the temperature of our lake profile a total of (34,321)*(9000) = 308,889,000 times. Given that our lake contains 6518 (of the 7701 total points), this means that the total possible number of collocation data points that we can generate is (6518)*(308,889,000) = 2,013,338,500,000.

Step 7: Sample the solution space to generate a collocation data set.

Our model, which computes the temperature profile of our lake for a 12-month period, generates an incredible 2,013,338,500,000 colocation data points. However, we need far fewer data points to train our neural network. With too much data, the neural network can become stuck at a local minimum during stochastic gradient descent. Additionally, the training time will be too high given the number of computations and batch sizes. So, we created an algorithm that samples the solution space and only takes a relatively small number of collocation points. As a result, we ultimately chose 10,502,532 collocation points. For each point, there are three inputs: time, x, and y (where x and y are simply the Cartesian coordinates of the points and are

computed using the row and column number of our array). For each point, there is only one output: the temperature T in Kelvin.

Step 8: Scale down the collocation dataset.

To scale down the collocation dataset, we apply the formula:

$$Scaled\ Value = \frac{(value - middle)}{(\max - middle)}$$

In the formula, "value" is the original x or y value, "max" is the value with the biggest magnitude in the given data set, "middle" is the middle value of the given data set, i.e., $[(max + min)/2]$, and "scaled_value" is the scaled-down version of the original value. Using this process, the data will be scaled down between -1 and 1. When training a neural network, it is important to scale down all given inputs and outputs for two reasons: (1) to center all of the outputs to zero so that we can initialize the neural network weights to zero; (2) to enable faster convergence since all of inputs and outputs are centered in the middle of the activation function. This method also helps a network avoid getting stuck in any local minimums or maximums during training.

B. Method 2 of 2: Build a neural network to learn the collocation data

Overview: When using a dataset to train a neural network, there are initially too many data points for the network to be efficient. First, a training set must be made by selecting a fraction of the original data points to show the network what the desired output is for a corresponding input. In the beginning, the weights the network used in the hidden layers are set randomly. However, because we scaled our outputs to be centered at 0, it makes sense that we should initialize all the weights to near zero. As the network trains more batch sizes, these weights are adjusted according to the pattern the network recognizes through the training set, which should result in the mean squared error (MSE) decreasing. At the beginning of training, the total error will be fairly high, but the more training the neural network performs, the closer the MSE will get to zero.

Step 1: For the structure of our neural network, we use the following: an input layer consisting of 3 nodes (one each for time, x, and y); four hidden layers of 100 nodes each, with each node being hyperbolic tangent; and one output layer consisting of 1 node (for temperature).

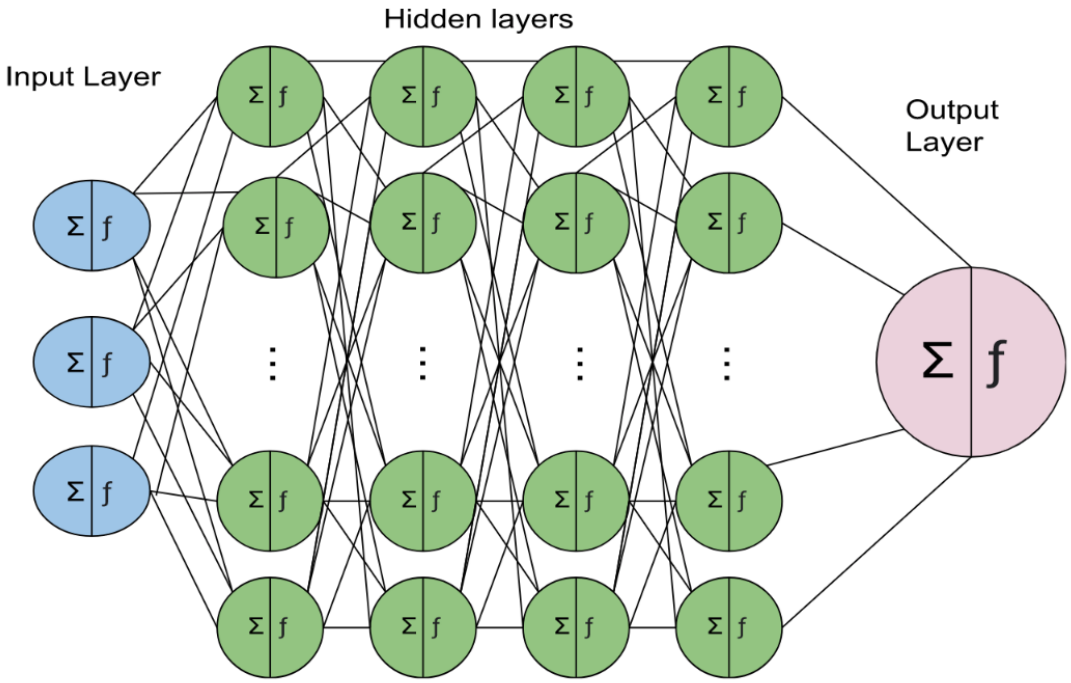


Fig. 4 Neural Network

Step 2: As we indicated previously, the loss function of our neural network is simply the mean squared error. This means that the neural network will adjust its weights so that the predicted answer is as close as possible to the correct answer.

Step 3: Our neural network takes the 3 inputs of time, x, and y, and gives as its output the temperature of the water at that single point. Our neural network can predict the temperature at every possible point in the lake and generalize this to any point on the grid for any given time step. To accomplish this, we will input 7701 combinations of (t,x,y) into the neural network all at the same time and compute the temperature at all 7701 points. However, we know that the lake water only comprises 6518 out of the 7701 points. That means that we should ignore 7701-6518 = 1183 of the answers, which correspond to all the points that fall into the green-colored cells in our CSV file.

Results

The criteria used during the neural network training was the mean squared error (MSE), which measures how close the neural network's predicted answer is to the traditional math model's correct answer. For the shallow lake (where we used slightly smaller dx and dy values), the MSE was 0.001520. For the deep lake, where we used bigger dx and dy values, the MSE was 0.001067. Given that we used real-life weather data, which is subjected to errors and noise, these MSE values are quite respectable. Additionally, by examining the neural network outputs, we can see that the 2D graphic passes the eye test and demonstrates a reasonable temperature profile.

The speed advantage of the neural network depends on the time step for which we want the solution to be computed. The neural network is able to compute the solution for time step 0 to time step 34,320. Because each time step represents a 15-minute increment, the total of 34,320 time steps represents roughly a one-year period (the original

data was missing the first few days of the year). The neural network can compute the solution at any time step in 0.192 seconds.

Using traditional math methods, the time needed to compute the solution is directly proportional to the time step for which the solution is desired. For example, computing the solution at time step 50 will be 25 times longer than computing the solution at time step 2. It follows that computing the last time step (time step 34320) will take the longest time; indeed, the traditional numerical method required 7 hours. This time was derived from y points calculated through a math model and used to train the neurological network. This means that our neural network can be as much as (7 hours)*(3600 seconds/hour)/(0.192 seconds) = 131,250 times faster than the traditional numerical method. This speed advantage is a huge breakthrough for scientists and will help enable a much faster investigation of the impact of climate change on the temperature of lakes, which has enormous consequences for biodiversity, human consumption, and pathogens and bacteria.

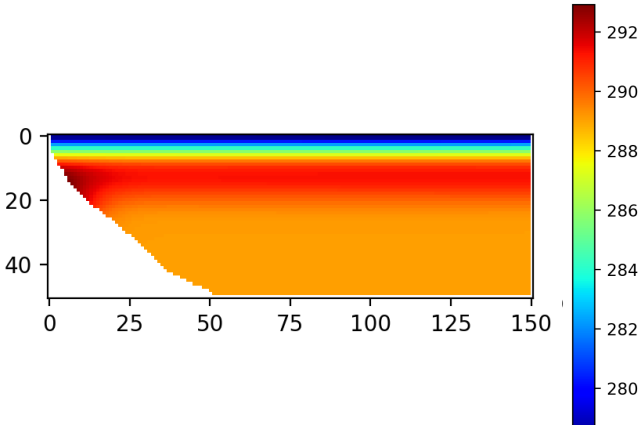


Fig. 5 The results are shown for Jan 31, 2021, at midnight, which is Excel row 2690 and a time-step 2688.

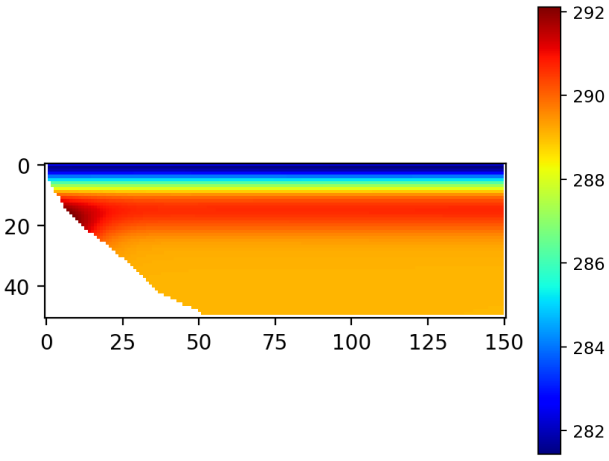


Fig. 6 The following diagram shows the result for February 28, 2021, at midnight, which is row 5474 with a time step of 5472.

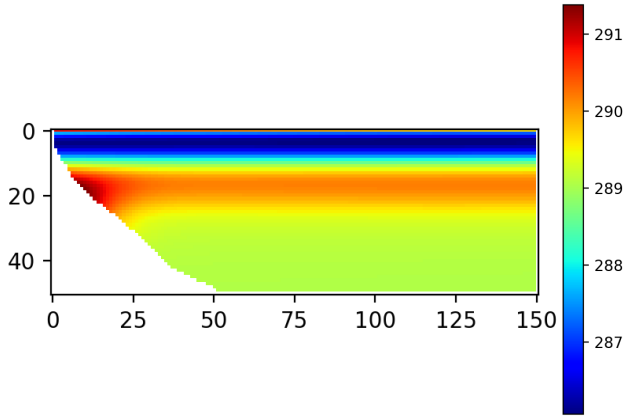


Fig. 7 The following diagram shows the result for March 31, 2021 at midnight, which is excel row 8446 with a time step 8444. The picture number is 8444/40 = 211.

Discussion and Conclusion

The data previously collected on the changes in temperature in a lake over time, the amount of albedo, daily precipitation, daily winds, and air temperature, once scaled, serve as inputs to calculate the new temperature of the top of the lake. Tracking the temperatures of lake water is important as it affects many aspects such as biodiversity, water hygiene, and aquatic activity. Water temperatures change the metabolic rates and biological activity of aquatic life. Fluctuations in temperature will change behavioral patterns such as migration routines, predator-prey interactions, and organism relocation. Major temperature changes can also prohibit plant respiration and photosynthesis. This affects dissolved oxygen levels, which can put stress on aquatic life if they drop below 5.0 mg/l. Water temperatures also affect biodiversity as all organisms have an optimal temperature range. Temperature increases also lead to changes in bacteria and fungi composition, which can affect water quality (Bacteria grows fastest at 41 and 135 degrees F).

The initial value of the temperature of the rest of the lake is assigned based on the day, but can be altered if necessary. From here, the data serves as inputs into a simulation that utilizes the finite difference transient heat equation and the first-order Neumann boundary in order to calculate the new temperature after a certain duration of time has passed, excluding the top layer. A separate math model is used to calculate the new value of the top layer, as it must take into account albedo, daily precipitation, daily winds, and air temperature when creating the next value. The initial temperature value of a part of the lake becomes the x value, while the calculated new temperature after a certain period of time becomes the output or y value. These are then made into a dataset in order to train the neural network.

Science often relies on mathematical models of physical behavior. Usually, math models are solved using "numerical methods." The most common include the finite difference method and the finite volume method. These methods are very accurate, but they can be slow. In fact, the model in this research (simulating the temperature profile of a lake for a 12-month period) took 7 hours to complete. The reason is that we can only step forward in time one step at a time. So, if we want the answer for t = 1000, we need to know the answer for t = 0.1, 0.2, 0.3, ….., 999.8, 999.9, and then 1000. There is a new area of AI technology called "physics-informed neural networks," which are neural networks that can understand some kind of physical behavior. Neural networks can act as ultra-fast solvers that approximate analytical partial differential equations. A neural network is a function that solves our model for any given point in time. So, if we want the solution for t = 1000, we simply input this to the neural network and it will provide the answer just as easily as it would if we gave it t = 0. The further out in time we want to see, the better the speed advantage of the neural network over the traditional numerical method. In this paper, we demonstrate that our neural network at the final step (t = 34320) is 131,250 times faster than the traditional numerical method. By predicting

the water temperature profile of lakes using much faster computational methods, we can determine in real-time the impact of climate change on biodiversity, water hygiene, and aquatic activity.

References

Pina, H. L. G., & Fernandes, J. L. M. (1984, January 1). *Applications in transient heat conduction*. SpringerLink. Retrieved January 28, 2023, from https://link.springer.com/chapter/10.1007/978-1-4899-2877-1_3#citeas

Recktenwald, G. W. (2004). Finite-difference approximations to the heat equation. *Mechanical Engineering*, *10*(01).

Szekeres, B. & Izsák, F. (2015). A finite difference method for fractional diffusion equations with Neumann boundary conditions. *Open Mathematics*, *13*(1), 000010151520150056. https://doi.org/10.1515/math-2015-0056

Hannabas, Bryon. (1989). *ESTIMATING ALBEDO FOR EVAPOTRANSPIRATION MODELS*. Retrieved January 29, 2023, from https://ttu-ir.tdl.org/bitstream/handle/2346/22458/31295005414007.pdf

Millar, S., Mupparapu, P., Brown, W., & Bub, F. (n.d.). *Convex air-sea heat flux calculations*. RetrievedoJanuaryo28,o2023,fromohttp://www.smast.umassd.edu/OCEANOL/reports/CONVEX/Heat_Flux_TR.dir/Heat_Flux_Tech_Report.html

Bintanja, R. (1996). The parameterization of shortwave and longwave radiative fluxes for use in zonally averaged climate models. *Journal of Climate*, *9*(2), 439–454. https://doi.org/10.1175/1520-0442(1996)009<0439:tposal>2.0.co;2

Hetrick, W. A., Rich, P. M., Barnes, F. J., & Weiss, S. B. (1993). GIS-based solar radiation flux models. In ACSM ASPRS ANNUAL CONVENTION (Vol. 3, pp. 132-132). AMERICAN SOC PHOTOGRAMMETRY & REMOTE SENSING+ AMER CONG ON.

Jasak, H. (1996). *error analysis and estimation for the finite volume method with applications to fluid flows. Phd thesis, Imperial College London, London. - references - scientific research publishing*.Retrieved January 28, 2023, from https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=1943696

RIZZO, F. J., & SHIPPY, D. J. (2012). A method of solution for certain problems of transient heat conduction. *AIAA Journal*, *8*(11), 2004–2009. https://doi.org/10.2514/3.6038

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.