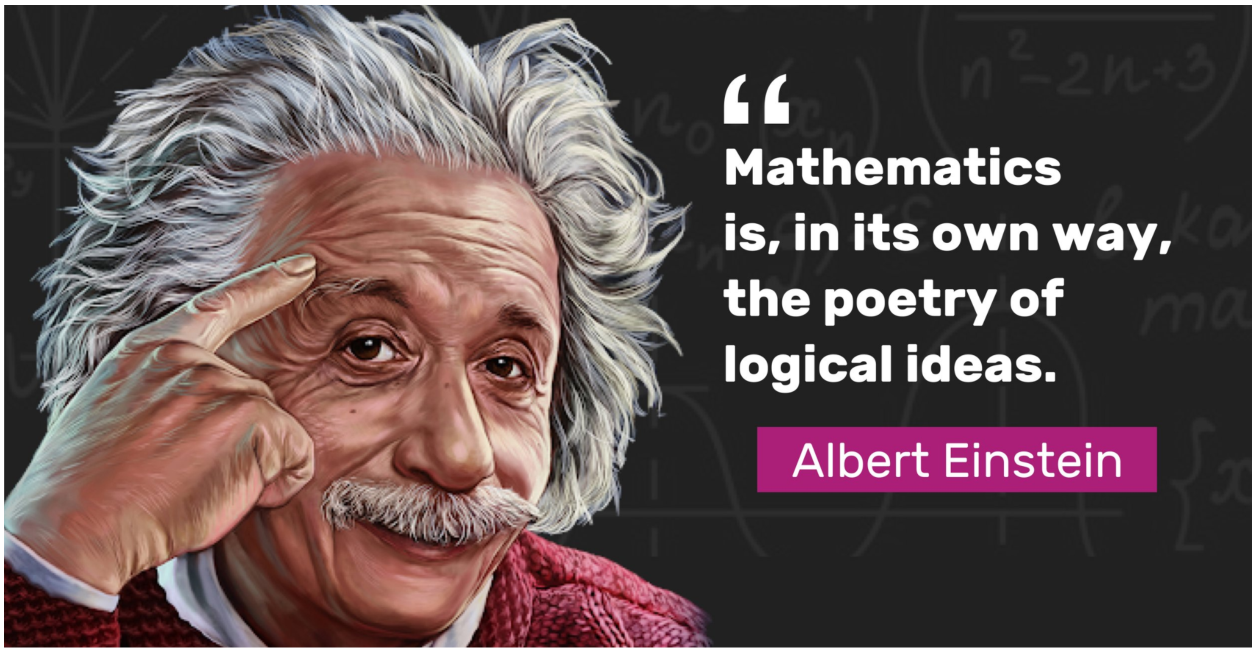
Laplace based Neural networks for solving differential equations



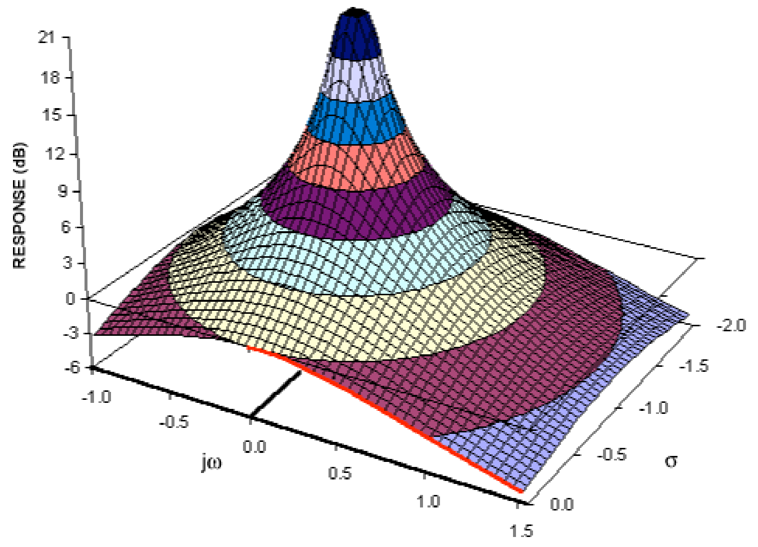
https://images.squarespace-cdn.com/content/v1/54905286e4b050812345644c/ee9b8637-0dfa-4329-ad4f-db6aa0faabe5/EinsteinQuote+copy.jpg

Mathematics extends to every corner of life - from the fibonacci spiral in flowers to the astonishing fractals in trees. But it doesn't stop there - math is also a critical part of the technological advancements involving Neural Networks and Deep learning. As neural networks grab the spotlight for computer science researchers, let's explore a fascinating link between deep nets and laplace transforms which might give prospects for increased efficiency in automating the solutions of complex partial differential equations.

Before we look at the application of laplace in neural networks, let's define some of the key properties of laplace transforms.

The Laplace transform:

The laplace transform is defined for a function $f(t)$ where $t \in \mathbb{R}$ and $t > 0$ as
$L[f(t)] = \int_0^\infty e^{-st} f(t)dt$ where $s \in \mathbb{C}$.



Graphical interpretation of the laplace transform (source: https://i.stack.imgur.com/1cuMb.gif)

Some of the properties of laplace transforms useful for solving differential equations are:

$$\text{Linear Property: } L[af(t)+bg(t)]dt = aL[f(t)] + bL[g(t)]$$

$$\text{Change of scale: If } L^{-1}[F(s)] = f(t) \text{ then } L^{-1}[F(as)] = \frac{1}{a}f\left(\frac{t}{a}\right)$$

Another critical aspect of laplace lies in the convolution theorem according to which if $f(t)$ and $g(t)$ are two functions and if $L^{-1}[G(t)] = g(t)$ then

$$L^{-1}[F(s)G(s)] = \int_0^t f(u)g(t-u)du = \int_0^c f(t-u)g(u)du]$$

where f * g is called the “convolution of f and g”.

In general, convolutions are commutative (i.e., f * g = g * f), associative (i.e.f(t) * [g(y) * h(t)] = [f(t) * g(t)] * h(t)) and distributive (i.e. f * (g + h) = f * (g) + f * (k)).

Classically, the laplace transform is used to solve differential equations. Here’s an example of how.

Given a differential equation defined as:

$$\frac{d^2y(t)}{dt^2} + \frac{2dy(t)}{dt} + 5y(t) = e^{-t}\sin t \text{ for } 0 < t < \infty$$

To solve this, the first step is to apply the laplace transform on both sides of the equation. Doing so, we get:

$$L\left[\frac{d^2y(t)}{dt^2}\right] + 2L\left[\frac{dy(t)}{dt}\right] + 5L[y(t)] = L[e^{-t}\sin t]$$

Now, we introduce the gamma function which is defined as $\gamma(s) = \int_0^\infty y(t)e^{-st}dt$. Plugging this in followed by some algebraic simplification, we get:

$$\gamma(s) = \frac{s^2+2s+3}{(s^2+2s+2)(s^2+2s+5)}.$$

Now, taking the inverse laplace transform, we have:

$$y(t) = L^{-1}(\gamma(s)) = L^{-1}\left[\frac{s^2+2s+3}{(s^2+2s+2)(s^2+2s+5)}\right].$$

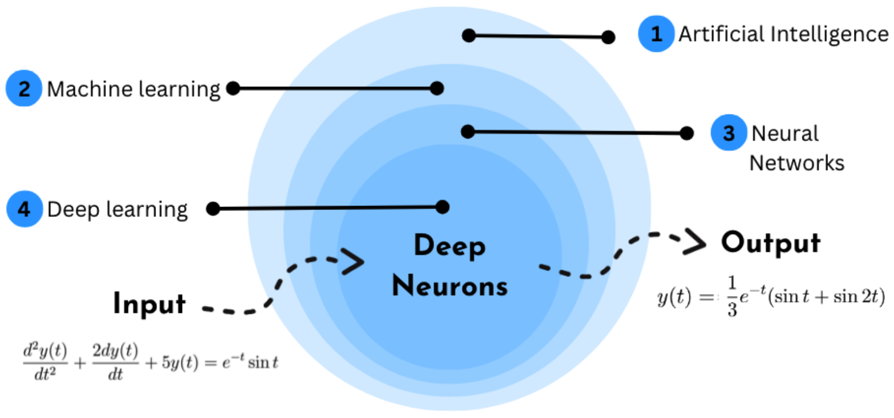
We can now solve this using partial fraction decomposition and arrive at the final answer which is:

$$\boxed{y(t) = L^{-1}[\gamma(s)] = \frac{1}{3}e^{-t}(\sin t + \sin 2t)}$$

That was a demonstration of how laplace transform can be manually used to solve differential equations. Unfortunately though, arriving at the solution is not always this easy. So, what if we embedded laplace transforms in deep neural networks to solve more complex differential equations?

Laplace transform artificial neurons:

When a two-year old is given a simple arithmetic to solve, they most likely make no sense of the numbers. A 10-year old might understand numerals and attempt several standard operations including addition, subtraction, multiplication, division, etc. An 18-year old is capable of learning complex ideas and notations including calculus, geometry, etc. Now let's take a pause. The first time gap (between 2 and 10 years old) is the same as the second one (10 and 18 years old). Yet, the learning rate differs greatly between the two lapses. This nonlinear growth curve simulates how neural nets use laplace transform for various applications. How? Let's find out!


1 Artificial Intelligence
2 Machine learning
3 Neural Networks
4 Deep learning
Deep Neurons
Input
$\frac{d^2y(t)}{dt^2} + \frac{2dy(t)}{dt} + 5y(t) = e^{-t}\sin t$
Output
$y(t) = \frac{1}{3}e^{-t}(\sin t + \sin 2t)$

Laplace based deep neural networks are right at the center of the chart

The three standard ways in which any machine learning model functions are supervised learning, unsupervised learning, and reinforcement learning. For this article, we'll be using a supervised learning approach to demonstrate the link between laplace and artificial neurons.

The LTAN (Laplace transform artificial neurons) is a neural nets model which consists of several layers of iteration for modification to minimize the value of Δs given as Z(s) - Y(s)Z(s). The key idea here is to be able to repeatedly apply the transform on the differential equation derived from the given sets of inputs until a solution with lowest error bounds is recognized. As this is done within a deep neural network, it's essential for the weights of each intermediate input to be altered based on the raw input weights.

Laplace transforms merged with deep neural networks can be helpful in solving more complex differential equations. But what is their application in the real world? For starters, laplace based neurons are much more effective in image processing and are already put in several test models for analyzing geological anomalies. They can also be used for monitoring natural bodies such as rivers, lakes, peaks, etc. to observe and predict any future catastrophes in the coastal regions.

Neural networks are becoming more popular by the second and it’s only a matter of time before they are integrated with other growing tech such as quantum computers. With the progressive research in differential equations, the potential of AI and ML will soon approach infinity.