Application of DFQ in Machine Learning: Neural Ordinary Differential Equations

Presentation by Jasmine Sellers

University of Washington Tacoma

TMATH 207 Fall 2023

Instructor: Duong Than

Outline

Topic Overview

Development and Use of Differential Equations

Application

Goal: Learn about differential equations in machine learning, specifically neural ODEs.

TOPIC OVERVIEW

Learning Target:

- What is artificial intelligence, machine learning, and deep learning?
- How are neural networks connected?
- What is neural ODEs?

Background

Why did I choose this topic?

I wanted to know …

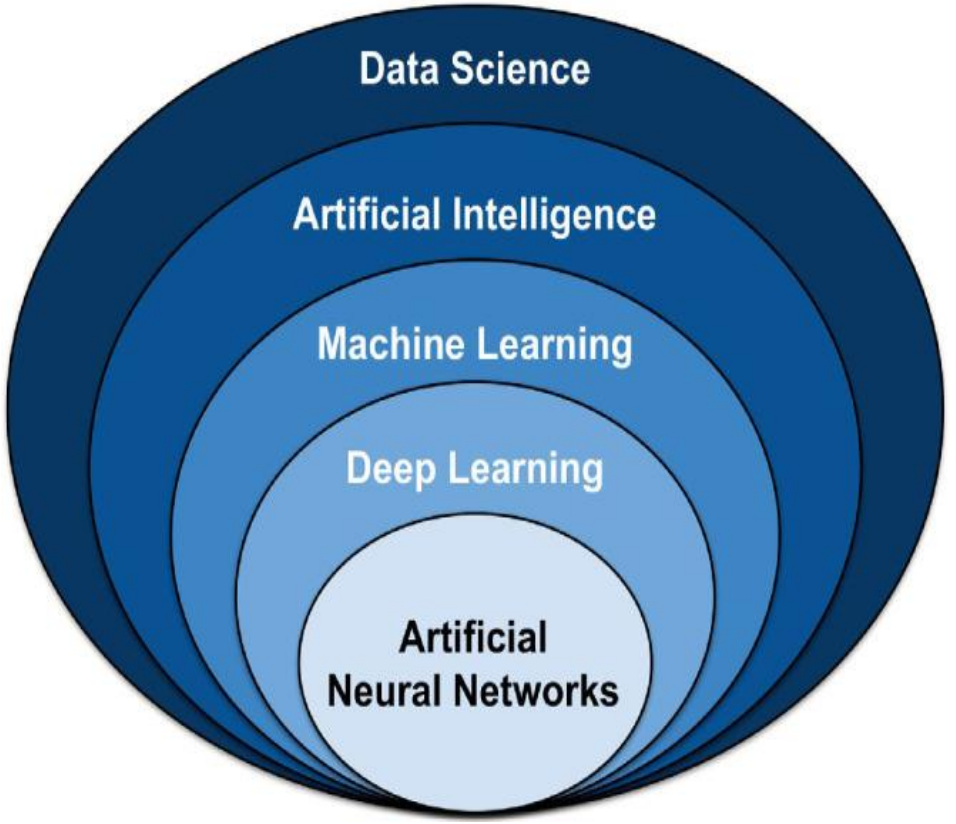
How can algorithms be used to make predictions and decisions?

What role do differential equations play in machine learning tools?

Why is it important to study this topic?

Implementation of AI is applicable in a vast majority of fields such as biology, business, and engineering.

Differential equations allow for flexible and adaptive frameworks for modeling data that change continuously



<u>Key Concepts</u>

Artificial Intelligence: intelligence of machines/software

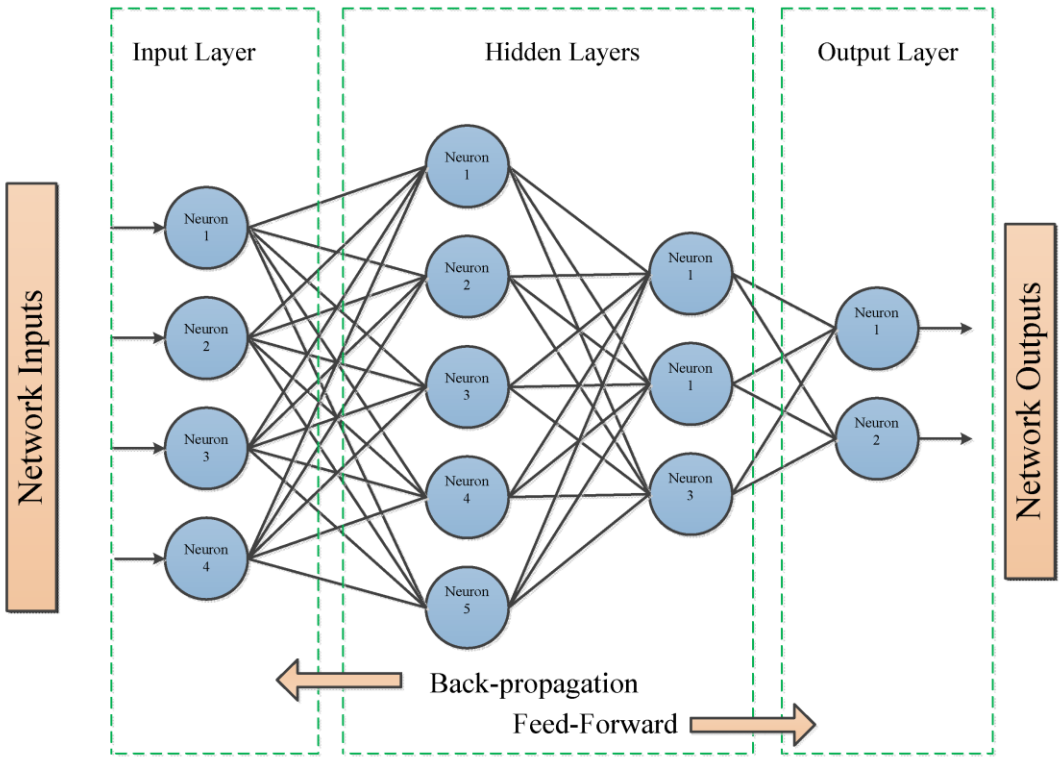
Machine Learning: uses data and algorithms to learn

Neural Networks: interconnected, layered processing units

Deep Learning: uses neural networks to learn

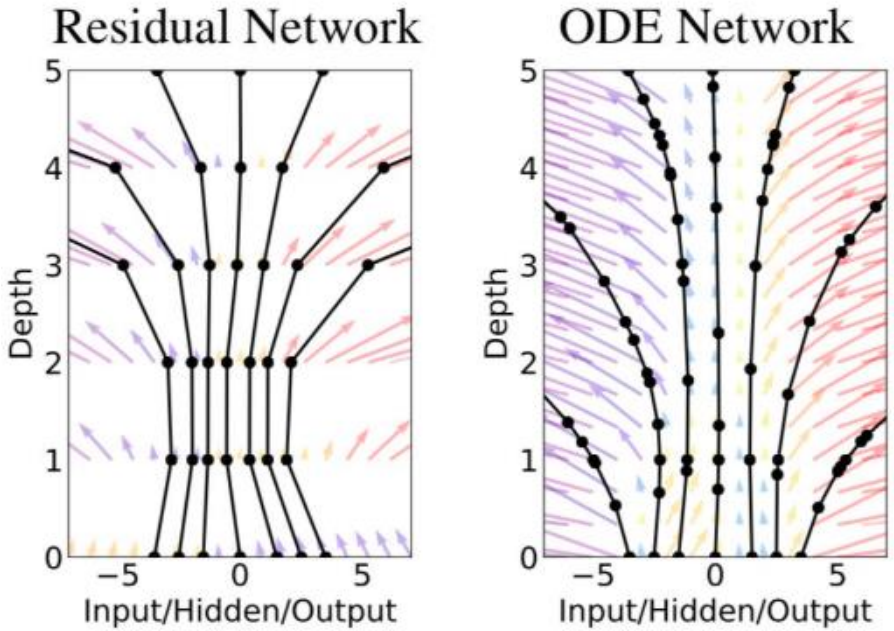
Aspects of Networks

- ***Neuron/Nodes***: receive input, perform calculations, and produce output
- ***Layers***: contain nodes that process and transform data
- ***Weights and Biases***: determine the strength of connections between neurons
- ***Feedforward Propagation***: flow data from the input layer to the output layer
- ***Backpropagation***: trains algorithm to adjust network weights and biases in accordance with error in the output; uses gradients and updates parameters to minimize the loss function.
- ***Loss Function***: quantifies the error between the network's prediction and actual values
- ***Gradients***: rate of change of the loss with respect to the model's parameters



Neural Ordinary Differential Equations (ODEs) Networks

- Parametrize the derivative of the hidden state
- $\frac{dh(t)}{dt} = f(h(t), t, \theta)$
  - h(t) = hidden layer function
  - t = time
  - θ = learnable parameters
  - $f(h(t), t, \theta)$ = neural network function



	Neural ODEs
Processing	Continuous data
Architecture	Dynamic/adaptive
Application	Physics-based modeling, physiology

DEVELOPMENT

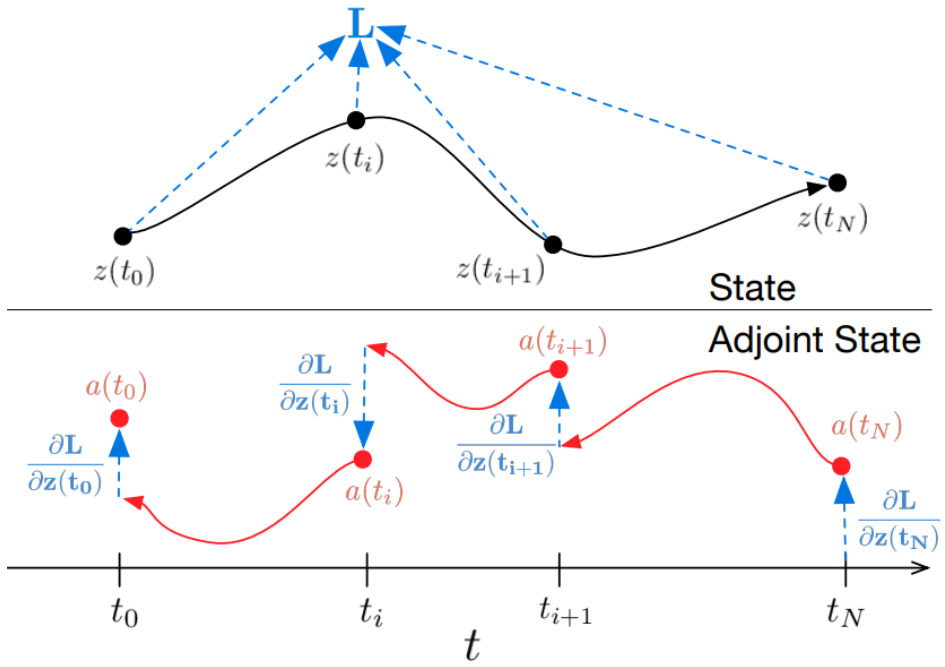
Learning Target:

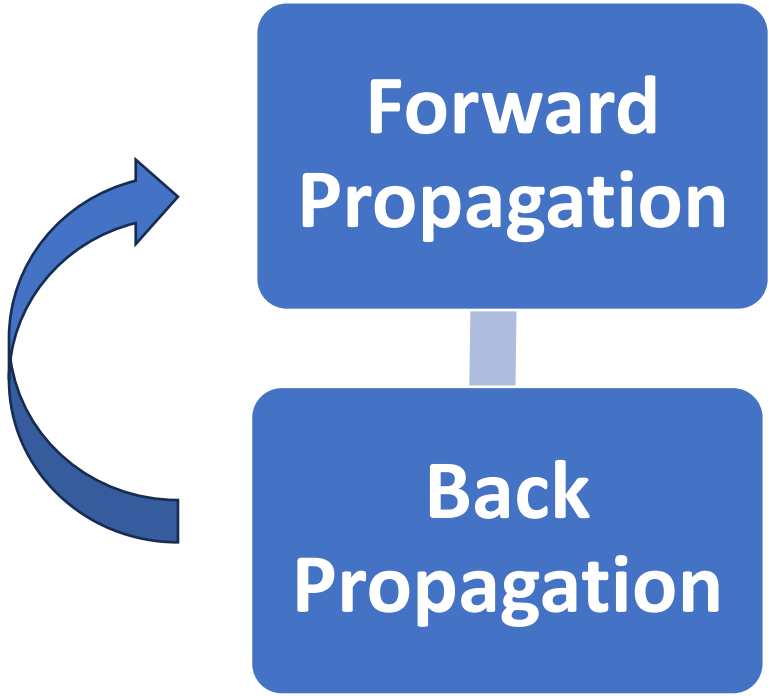
- What differential equations are involved in neural ODE?

Overview

Given: A parametrize derivative of the hidden state
$f(h(t), t, \theta)$

Output: state at the end time





Forward Propagation

$$h(t_1) = h(t_0) + \int_{t_0}^{t_1} f(h(t), t, \theta)dt$$

<u>Method</u>: Separable differential equations with an initial value $\boldsymbol{h(t_0) = f(h(t_0), t_0, \theta)}$

- recall: $\frac{dh(t)}{dt} = f(h(t), t, \theta)$

➢ $dh(t) = f(h(t), t, \theta)dt$

➢ $\int_{h(t_0)}^{h(t)} dh(s) = \int_{t_0}^{t} f(h(s), s, \theta)ds$ Integrate with an arbitrary constant of integration

➢ $h(s)|_{h(t_0)}^{h(t)} = \int_{t_0}^{t} f(h(s), s, \theta)ds$

➢ $h(t) - h(t_0) = \int_{t_0}^{t} f(h(s), s, \theta)ds$

➢ $h(t) = h(t_0) + \int_{t_0}^{t} f(h(s), s, \theta)ds$

<u>Variables:</u>

- h(t) = hidden layer function
- t = time, t_0 = initial time, t_1 = final time of interest
- θ = learnable parameters
- $f(h(t), t, \theta)$ = neural network function

Backpropagation: Adjoint sensitivity method

Adjoint sensitivity method:

- Captures the sensitivity of the function with respect to the parameters of the model
- <u>What is involved</u>:
  - Adjoint, a(t), a gradient of the loss function (L) depends on the hidden state h(t)
  - Gradient of the loss function (L) with respect to the systems parameters (θ)

1. $$\frac{\partial L}{\partial h(t_0)} = \frac{\partial L}{\partial h(t_1)} - \int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h}\, dt$$

   - Used to adjust hidden states based on loss

2. $$\frac{\partial L}{\partial \theta} = -\int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial \theta}\, dt$$

   - Used to adjust the model's parameters based on loss

<u>**Variables:**</u>

- h(t) = hidden layer function
- t = time, t_0 = initial time, t_1 = final time of interest
- θ = learnable parameters
- $f(h(t), t, \theta)$ = neural network function
- $a(t)^T = transpose\ of\ the\ adjoint\ (\frac{dL}{dh(t)})$

Big Question: What is the loss function?

- **Loss function:** quantifies the error between the network's prediction and actual values
- The loss function (L) used depends on the purpose of the neural ODE

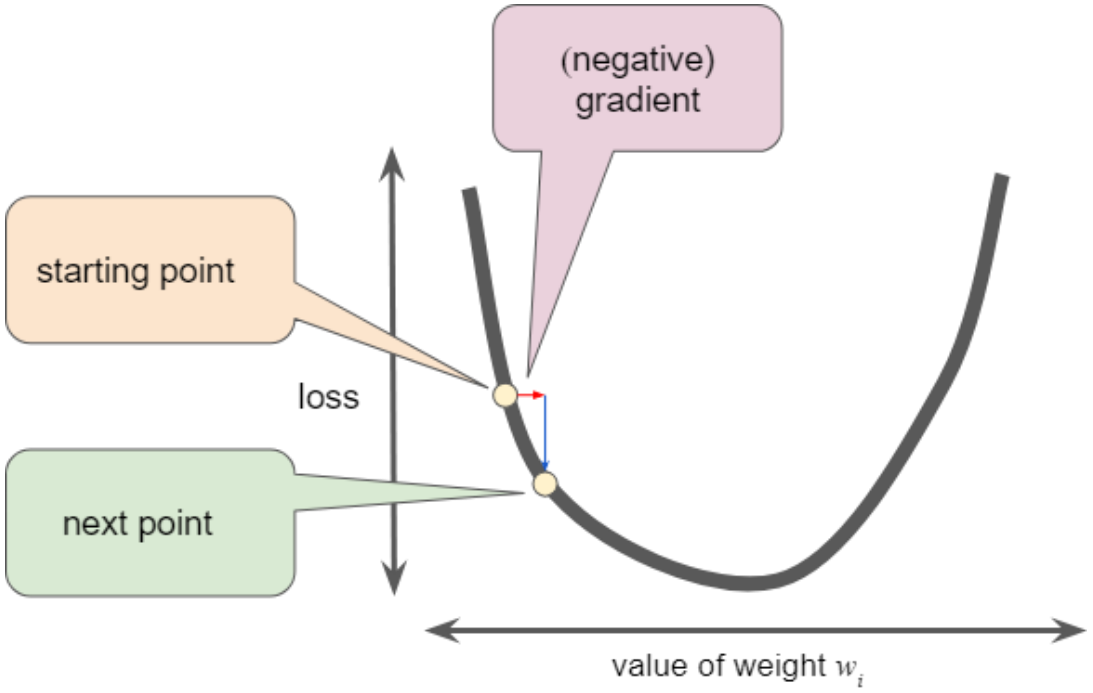
- A few methods:
  - **Quadratic Mean Square Error (MSE)**: penalizes large errors more
  - **Log-Cosh Loss**: used for data containing many outliers and noise
  - **Poisson Loss**: used if interested in the number of times an event occurred
- Note we won't go deep into this topic since this is a statistics-focused topic

Big Question: How do we optimize the model?

- The gradients are used to adjust the model's parameters (i.e., train the model)
  - $\frac{\partial L}{\partial h(t_0)}, \frac{\partial L}{\partial \theta}$
- One method: Gradient Descent

- Note we won't go deep into this topic since there is no direct application of differential equations

Gradient Descent

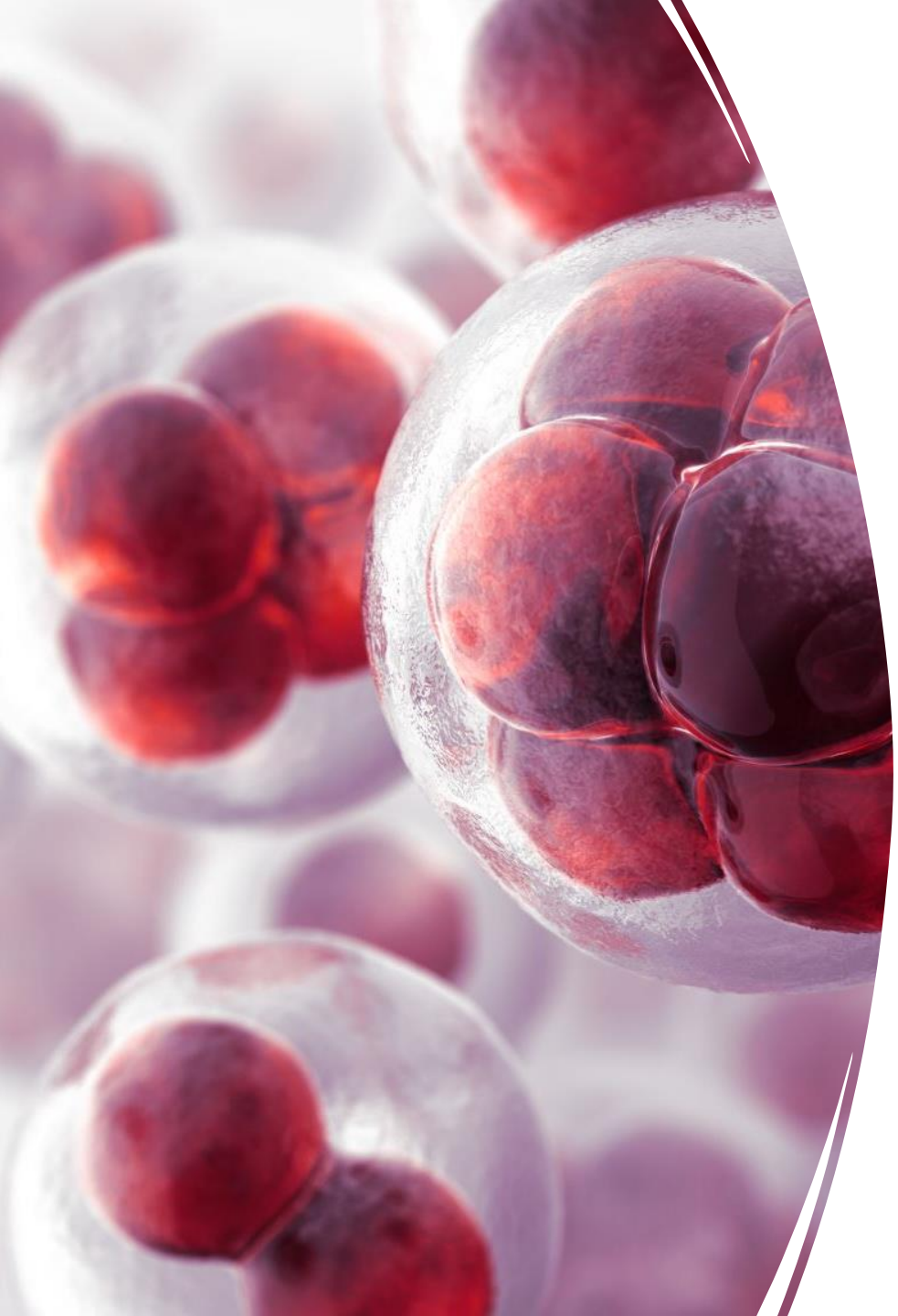


- Uses the direction (+/-) to determine which direction will reduce loss
- Some fraction of the gradient's magnitude is added to the 'starting point'

APPLICATION

Learning Target:

- Walk through setting up problems using neural ODEs.



Example 1: Tumor Dynamic Modeling

$$\frac{dh(t)}{dt} = f(h(t), t, \theta)\ t \in [0, T]$$

- t represents time
  - T is the final simulation time of interest
- θ patient-specific kinetic parameters (e.g., chemical reactions and biological processes)
- h(t) is the time-continuous solution representing the state of the tumor
  - h(0) patient-specific initial state
  - h(T) patient's predicted state at time T

These metrics can be used to predict a patient's overall survival.

Example 2: Taxicab Rooftop Advertising

$$\frac{dh(t)}{dt} = f(h(t), t, \theta)\ t \in [0, T]$$

- t represents time
  - T is the final simulation time of interest
- θ parameters in the system (e.g., start location, drop-out rate of passengers, distance traveled)
- h(t) taxi location
  - h(0) initial taxi location
  - H(T) taxi location at time T

The taxi's location can be used to target advertisement.

Example 3: Pharmacokinetics

- $\frac{dh(t)}{dt} = f(h(t), t, \theta)\ t \in [0, T]$
  - t represents time
    - T is the final simulation time of interest
  - θ measurable parameters in the system (e.g., absorption rate, distribution volume, clearance, and half-life)
  - h(0) initial drug concentration
  - h(t) drug concentration at time t

Understanding the dynamics of drug concentration can allow for the optimization of drug therapy


Questions?

Very Briefly

If you are interested, these are the proofs of the gradients

Proving Gradient (1)

Let $a(t) = \frac{\partial L}{\partial h(t)}$ We will show: $\frac{da(t)}{dt} = -a(t)\frac{\partial f(h(t),t,\theta)}{\partial h(t)}$

1. Note $\frac{\partial L}{\partial h_t} = \frac{\partial L}{\partial h_{t+1}} * \frac{\partial h_{t+1}}{\partial h_t}$ and recall the forward propagation formula $h(t_1) = h(t_0) + \int_{t_0}^{t_1} f(h(t), t, \theta)dt$ which will be adusted for the interval $[t, t+\varepsilon]$ thus $h(t+\varepsilon) = h(t) + \int_{t}^{t+\varepsilon} f(h(t), t, \theta)dt$ which we can represent as a transformation, $T_\varepsilon(h(t), t)$.

2. This can then be applied to the $\frac{\partial L}{\partial h_t}$ formula to get $\frac{\partial L}{\partial h(t)} = \frac{\partial L}{\partial h(t+\varepsilon)} * \frac{\partial h(t+\varepsilon)}{\partial h(t)} = a(t+\varepsilon)\frac{\partial T_\varepsilon(h(t),t)}{\partial h(t)} = a(t)$.

3. We can then use the definition of the derivative to rewrite $\frac{da(t)}{dt} = \lim_{\varepsilon \to 0^+} \frac{a(t+\varepsilon)-a(t)}{\varepsilon}$ and substitute $a(t)$ with what we derived previously to eventually get $\frac{da(t)}{dt} = -a(t)\frac{\partial f(h(t),t,\theta)}{\partial h(t)}$

If we solve for $a(t_0)$ using separable equations with an initial value we get $a(t_0) = a(t_1) - \int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h} dt$

which is equivalent to $\frac{\partial L}{\partial h(t_0)} = \frac{\partial L}{\partial h(t_1)} - \int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h} dt$

Proving Gradient (2)

$$We\ will\ show: \frac{dL}{d\theta} = -\int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial \theta} dt$$

- $\frac{da(t)}{dt} = -a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h}$ can be used to obtain gradients with respect to θ and t (recall θ are our parameters)
  - $\frac{d\theta(t)}{dt} = 0\ and\ \frac{dt(t)}{dt} = 1$ since we are assuming our parameters are constants and dt/dt = 1
    - $\frac{d}{dt}\begin{bmatrix} h \\ \theta \\ t \end{bmatrix}(t) = f_{aug}([h,\theta,t]) = \begin{bmatrix} f([h,\theta,t]) \\ 0 \\ 1 \end{bmatrix}$
  - $a_{aug} = \begin{bmatrix} a \\ a_\theta \\ a_t \end{bmatrix}\ where\ a_\theta(t) = \frac{dL}{d\theta(t)}, a_t = \frac{dL}{dt(t)}$
  - $\frac{\partial f_{aug}}{\partial [h,\theta,t]} = \begin{bmatrix} \frac{\partial f}{\partial h} & \frac{\partial f}{\partial \theta} & \frac{\partial f}{\partial t} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}(t)$, by the Jacobian matrix of f
  - Recall $\frac{da(t)}{dt} = -a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h}$, thus $\frac{da_{aug}}{dt} = -a_{aug}^T \frac{\partial f_{aug}}{\partial [h,\theta,t]} = -[a, a_\theta, a_t] \begin{bmatrix} \frac{\partial f}{\partial h} & \frac{\partial f}{\partial \theta} & \frac{\partial f}{\partial t} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ (t)= $- \left[\frac{a\partial f}{\partial h}, \frac{a\,\partial f}{\partial \theta}, \frac{a\,\partial f}{\partial t}\right]$ (t)
  - Recall $a_\theta(t) = \frac{dL}{d\theta(t)}$. *Suppose* $a_\theta(t_N) = 0$ *thus by using the second element in the matrix,* $\frac{da_\theta}{dt} = a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial \theta}$ *, and integrating:*
    - $\frac{dL}{d\theta} = a_\theta(t_0) - a_\theta(t_N) = -\int_{t_N}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial \theta}$ dt

References

1. Abdolrasol, M. G., Hussain, S. M., Ustun, T. S., Sarker, M. R., Hannan, M. A., Mohamed, R., Ali, J. A., Mekhilef, S., & Milad, A. (2021). Artificial Neural Networks Based Optimization Techniques: A Review. Electronics, 10(21), 2689. https://doi.org/10.3390/electronics10212689
2. Bräm, D. S., Nahum, U., Schropp, J., Pfister, M., & Koch, G. (2023). Low-dimensional neural ODEs and their application in pharmacokinetics. Journal of Pharmacokinetics and Pharmacodynamics, 1-18.
3. Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. Advances in neural information processing systems, 31.
4. Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Campbell, J. P. (2020). Introduction to machine learning, neural networks, and deep learning. Translational vision science & technology, 9(2), 14-14.
5. Finlay, C., Jacobsen, J. H., Nurbekyan, L., & Oberman, A. M. (2020). How to train your neural ode. *arXiv preprint arXiv:2002.02798*, *2*.
6. Google. (n.d.). Reducing loss: Gradient descent | machine learning | google for developers. Google. https://developers.google.com/machine-learning/crash-course/reducing-loss/gradient-descent
7. Grogan S, Preuss CV. Pharmacokinetics. [Updated 2023 Jul 30]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2023 Jan-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK557744/
8. Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31-44.
9. Jeon, J., Kang, S., Jo, M., Cho, S., Park, N., Kim, S., & Song, C. (2021, October). Lightmove: A lightweight next-poi recommendation fortaxicab rooftop advertising. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 3857-3866).
10. Laurie, M., & Lu, J. (2023). Explainable Deep Learning for Tumor Dynamic Modeling and Overall Survival Prediction using Neural-ODE. arXiv preprint arXiv:2308.01362.
11. Lebl, J. (2019). Separable equations. In Notes on diffy QS: Differential Equations for Engineers (6.5, pp. 33–37). essay, CreateSpace Independent Publishing Platform.
12. Oke, S. A. (2008). A literature review on artificial intelligence. *International journal of information and management sciences*, *19*(4), 535-570.
13. Ota, R., & Yamashita, F. (2022). Application of machine learning techniques to the analysis and prediction of drug pharmacokinetics. *Journal of Controlled Release*, *352*, 961-969.
14. Terven, J., Cordova-Esparza, D. M., Ramirez-Pedraza, A., & Chavez-Urbiola, E. A. (2023). Loss Functions and Metrics in Deep Learning. A Review. arXiv preprint arXiv:2307.02694.