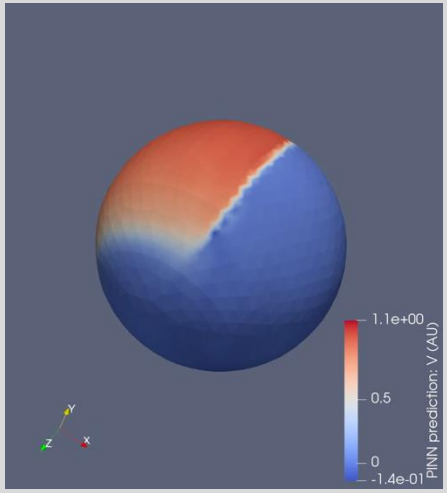
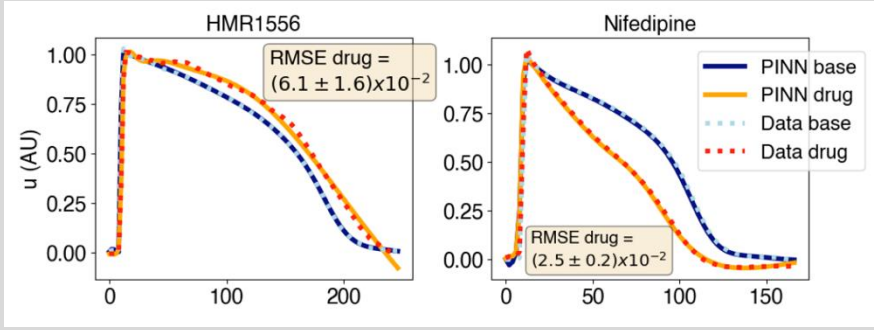
PINNs Tutorial

- Introduction to PINNs
  - What are PINNs?
  - Considerations when designing PINNs
- Google Colab Exercises
  - PINNs to estimate cerebral blood flow
  - PINNs to model cardiac electrophysiology
- Recent developments in PINNs
  - Problems with PINNs
  - How to address them

Introduction to Physics-Informed Neural Networks





Annie Cing-En Chiu & Marta Varela

ching-en.chiu18@imperial.ac.uk

marta.varela@imperial.ac.uk

IMPERIAL

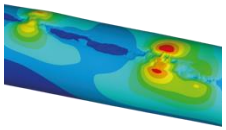
Physics-Informed Machine Learning

Aims

- Model and understand the physiology of the system (organ, tissue)
- Identify, predict and characterise failure modes (pathology)
- Optimally control the system (design and assess treatments)

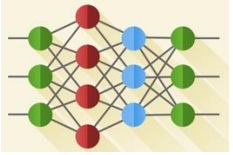
- Differential equations known
- All parameters identified
- Initial and boundary conditions known

- A lot of data is available
- Data perfectly reproduces the system and is representative of the population



Finite elements, finite differences …

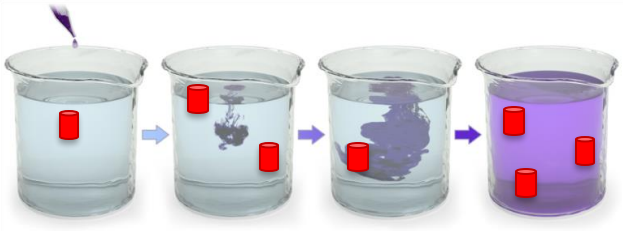
Neural networks …



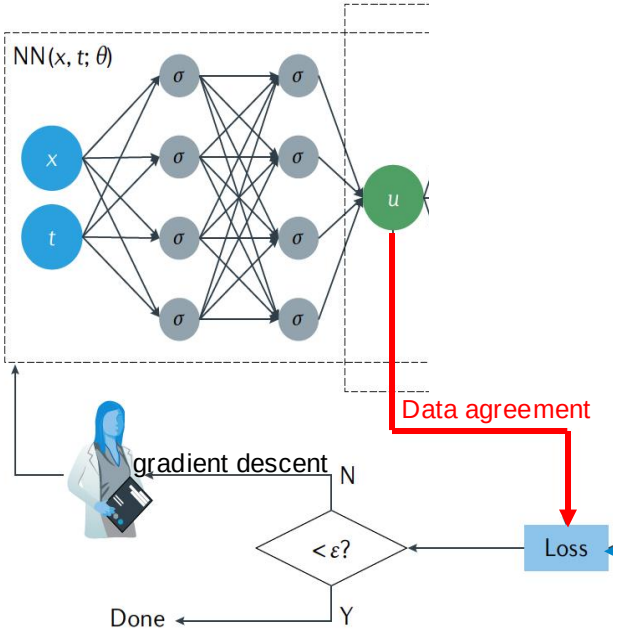
Physics-Informed Machine Learning (e.g., PINNs)

IMPERIAL

Physics-Informed Neural Networks (PINNs)



$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}$$ **Diffusion Equation**



$$\mathcal{L} = w_{data}\mathcal{L}_{data} + w_{PDE}\mathcal{L}_{PDE},$$

where

$$\mathcal{L}_{data} = \frac{1}{N_{data}}\sum_{i=1}^{N_{data}}(u(x_i, t_i) - u_i)^2 \quad \text{and}$$

$$\mathcal{L}_{PDE} = \frac{1}{N_{PDE}}\sum_{j=1}^{N_{PDE}}\left(\frac{\partial u}{\partial t} - D\frac{\partial^2 u}{\partial x^2}\right)^2_{|(x_j,t_j)}.$$

Forward Mode

- NN to solve the differential equation in entire domain (find function u(x,t))
- No discretization of domain performed

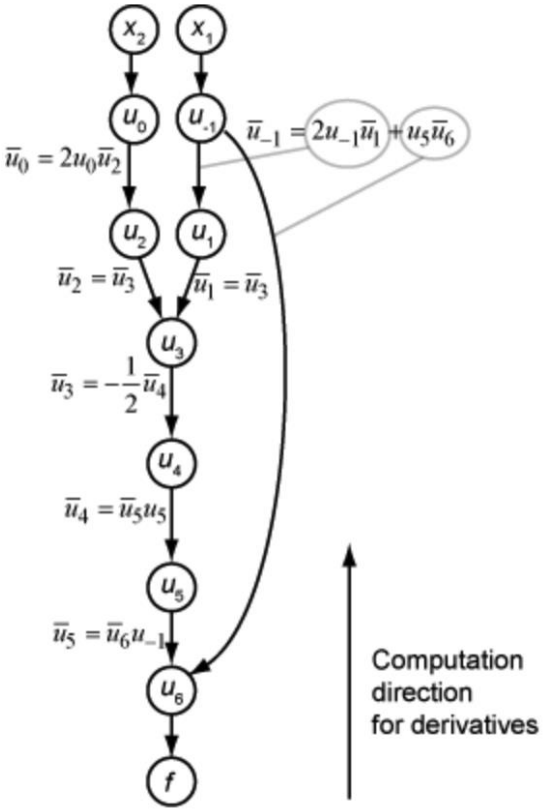
Inverse Mode

- Also estimate equation parameters, e.g. diffusion coefficient, D
- Adjusted using gradient descent

Raissi et al, J Comp Phys, 2019 Karniadakis et al, Nat Phys Rev, 2021

Calculate Derivatives Fast: Auto Diff

- Automatic Differentiation (AutoDiff):
  - The workhorse of backpropagation for NN training
  - More widely applicable than symbolic diff
  - No discretization errors, discrete solutions as in numerical diff
  - Faster, especially for high-order and multivariate diff
- Based on the algebra of dual numbers and the chain rule.
- In the PINNs framework, equation derivatives come for free!



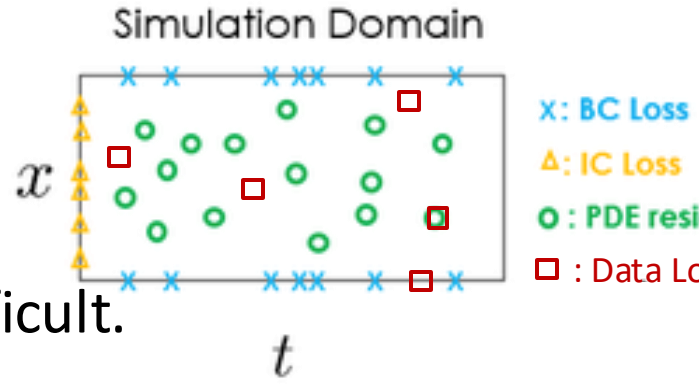
Baydin et al, JMLR, 2018

Count Your Losses

- Loss terms typically use the mean squared error (MSE).
- Agreement with equations is computed at pre-assigned points ($N_{PDE/ODE}$). Unrelated to the number of data points (N_{data}).
- Different loss terms can have different weights.
- Can also adaptatively refine location of points where $L_{PDE/ODE}$ is calculated to minimise it[1].
- PINNs can work in forward mode

with no experimental data (just BCs, ICs), but...

  - Convergence is slow(er) and (more) difficult.
  - If you have extra data, use it!



Lu et al, SIAM Review, 2021

On the Edge: Initial & Boundary Conditions

- Initial and/or boundary conditions should be included in the loss function (soft enforcement).

$$\mathscr{L} = \mathscr{L}_{PDE/ODE} + \mathscr{L}_{IC} + \mathscr{L}_{BC} + \mathscr{L}_{data}$$

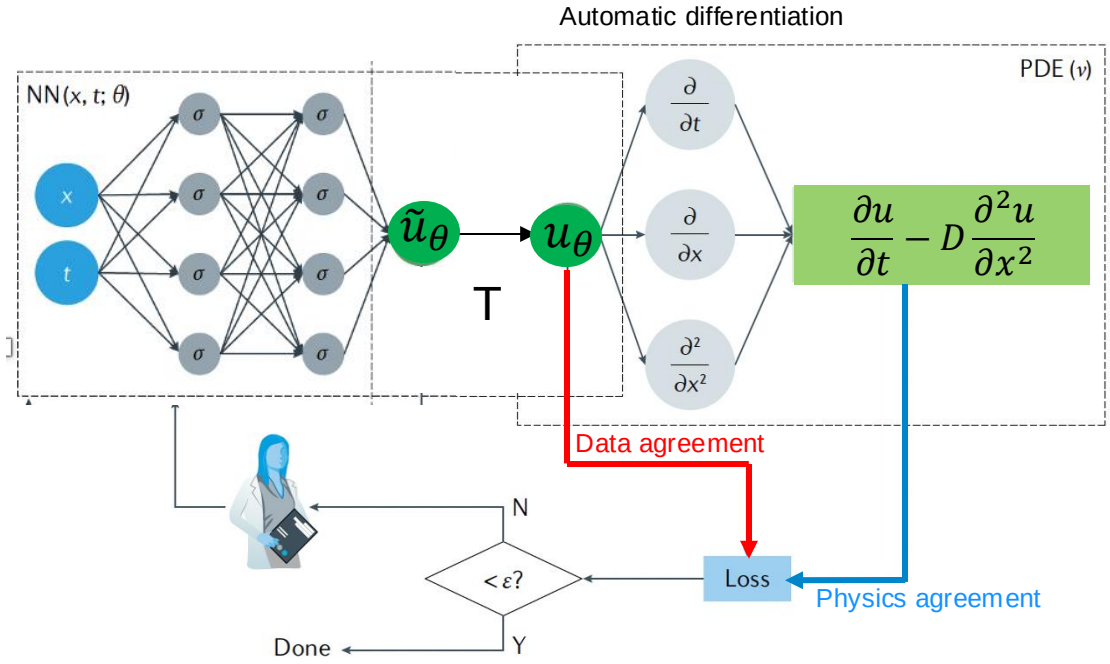
  - The IC and BC losses resemble data losses in specific conditions.

$$\mathcal{L}_{IC} = \frac{1}{N_{ptsIC}} \sum_{i=1}^{N_{ptsIC}} (u_\theta(x_i, 0) - u(x_i, 0))$$

$$\mathcal{L}_{BC} = \frac{1}{N_{ptsBC}} \sum_{i=1}^{N_{ptsBC}} (u_\theta(x_{BC}, t_i) - u(x_{BC}, t_i))$$

- For hard enforcement, we can apply transforms to the NN solutions so that ICs and BCs are obeyed.

Hard BCs, ICs



- Examples:

$$\frac{\partial y}{\partial t} = \frac{\partial^2 y}{\partial x^2} - e^{-t}(\sin(\pi x) - \pi^2 \sin(\pi x)), \qquad x \in [-1, 1], \quad t \in [0, 1]$$

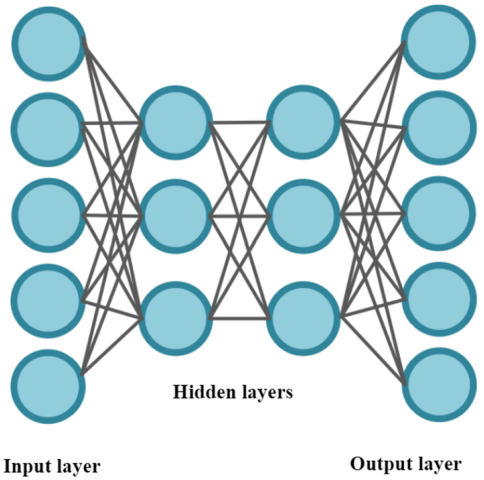
with $y(x, 0) = \sin(\pi x)$ and $y(-1, t) = y(1, t) = 0.$

gives e.g., $T = t * (1 - x^2) * y + \sin(\pi x)$

Example taken from:
https://deepxde.readthedocs.io/en/latest/demos/pinn_forward/diffusion.1d.exactBC.html

Build It: Architecture for PINNs

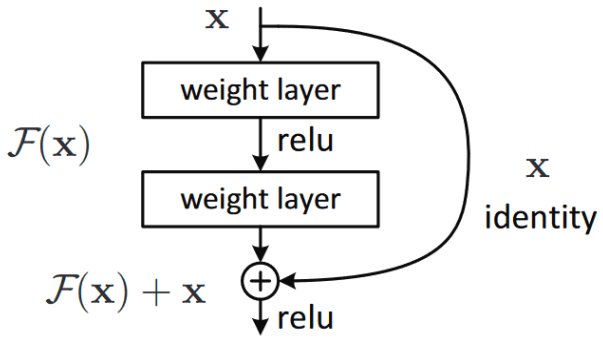
- Fully-connected NNs (= multilayer perceptrons, MLPs) are most popular.



Problems:

- Vanishing gradients for deep networks
  - ReLU as activation function not commonly used (problems differentiating it!)
  - Tanh (and Swish) most often used
- Skip (=residual) connections sometimes added
- Normalisation of inputs, outputs helps balance gradients

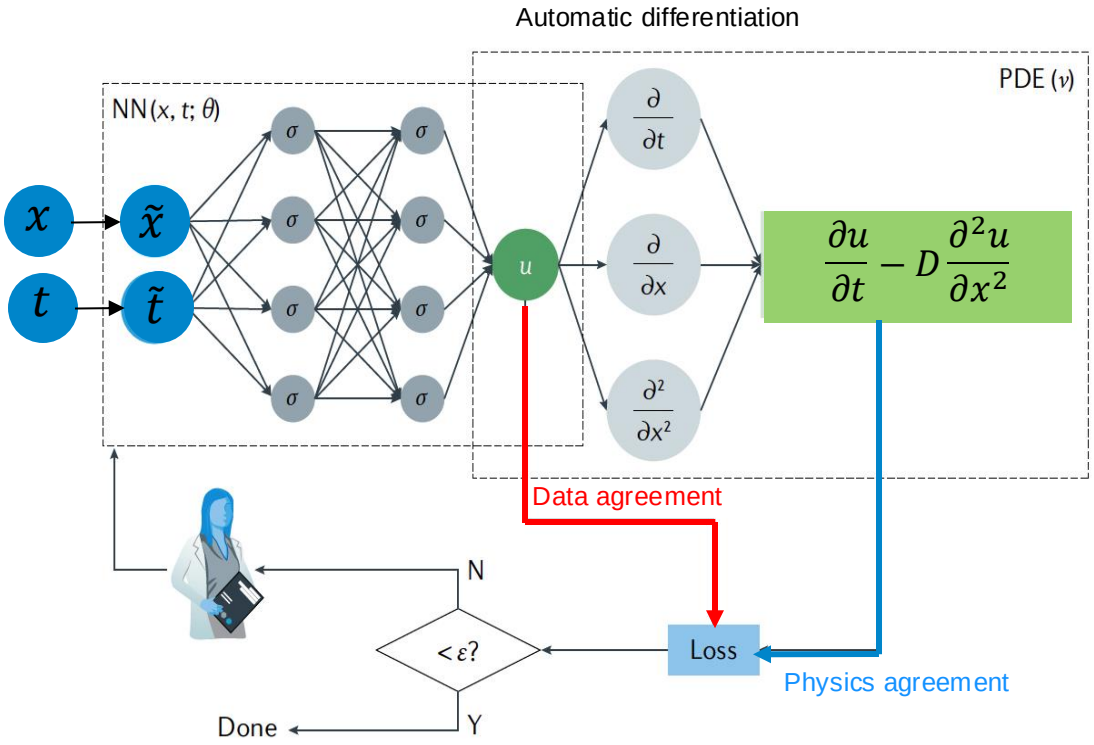
- Depth and width adjusted manually
  - 2-6 layers
  - 32-516 neurons/layer



Wang et al, 2023, arxiv.org/abs/2308.08468

High-Frequency Modes

- FCNNs suffer from spectral bias: they preferentially learn low-frequency functions.
  - Can be mitigated by changing activation function
  - In PINNs, more often addressed by applying a (non-learned) Fourier transform to the inputs



Fourier Mapping

$$\tilde{t} = \sum_{k=1}^{N_{modes}} \sin(kt)$$

Random Fourier Mapping

$$\tilde{t} = [\sin(Bt), \cos(Bt)]$$

with $B \in \mathbb{R}^{N_{modes}}$, $B_k \sim \mathcal{N}(0, \sigma^2)$

Choose the Best: Optimiser for PINNs

- Adam is most commonly used.
  - Low learning rates work best. (Can be reduced iteratively if needed.)
- An L-BFGS pass following Adam sometimes used.
- Batch learning possible in PINNs, but only useful if a lot of training points are used.
  - Full-batch learning is very common.
- NN parameters are typically initialised using Glorot initialisation, although statistical properties of PINNs ‘samples’ are hard to calculate.

Wang et al, 2023, arxiv.org/abs/2308.08468

Inverse Mode

- PINNs can also estimate one or more unknown model parameters given some experimental data for the system.
- These parameters are included in the AutoDiff-based optimisation, playing a similar role to network parameters.
- It is a good idea to perform identifiability analyses before running PINNs in inverse mode.

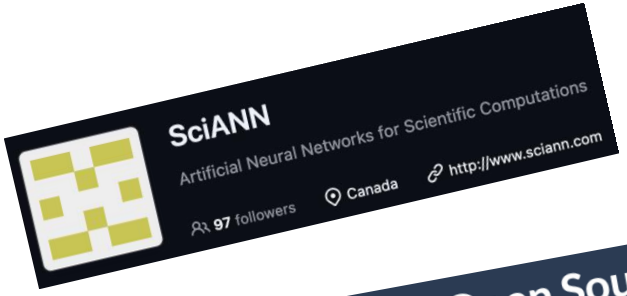
Raissi et al, J Comp Phys, 2019 Karniadakis et al, Nat Phys Rev, 2021

Make Your Life Easier: Libraries for PINNs

- Here, we will use an open-source Python library (with a TensorFlow backend):

- Other PINN libraries (Python, Julia, Jax):

NVIDIA Modulus



SciML: Open Source Software for Scientific Machine Learning

Pinns is a python library which creates neural networks that can solve differential equations.

PINNs

JAX-PI

- Can also write your own code from scratch!

Tutorial Exercise 1

- Estimate cerebral perfusion (=cerebral blood flow, CBF) in an infant using data from arterial spin labelling (ASL) MRI.
- Tutorial exercise GitHub page:

- https://github.com/annien094/PINNs-tutorial-MICCAI-2024/blob/main/PINNs_ASL.ipynb

PINNing Cerebral Blood Flow: Analysis of Perfusion MRI in Infants using Physics-Informed Neural Networks

Christoforos Galazis [1,2,*], **Ching-En Chiu** [2,3], **Tomoki Arichi** [4], **Anil A. Bharath** [5] **and Marta Varela** [2]

[1] *Department of Computing, Imperial College London, London, UK*
[2] *National Heart & Lung Institute, Imperial College London, London, UK*
[3] *Department of Electrical Engineering, Imperial College London, London, UK*
[4] *Centre for the Developing Brain, King's College London, London, UK*
[5] *Department of Bioengineering, Imperial College London, London, UK*

Correspondence*:
Christoforos Galazis
c.galazis20@imperial.ac.uk

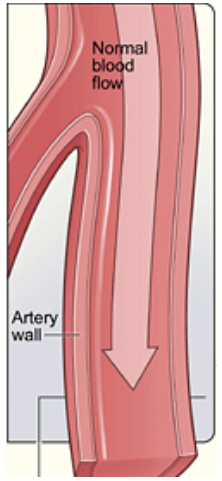
(under review)

https://github.com/cgalaz01/supinn

IMPERIAL

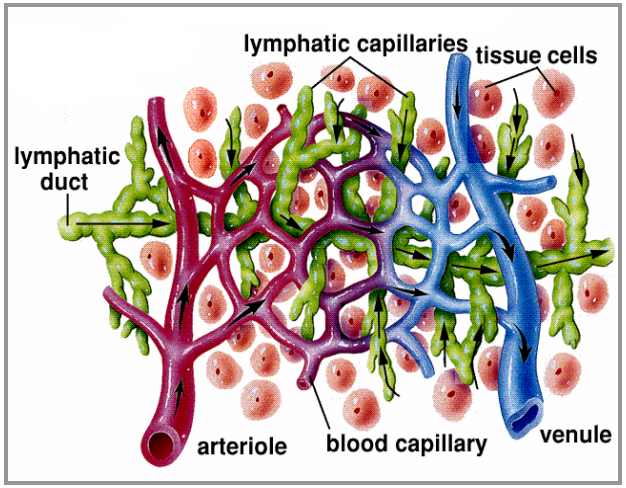
What is Perfusion?

- **Blood Flow in a Vessel:** volume of blood transported per unit time (mL blood/s)



- **Perfusion:** volume of blood delivered to a given amount of tissue per unit time (mL blood/100 g tissue/min)

- Information about:
  - Quality of local blood supply
  - Health/activity of the tissue



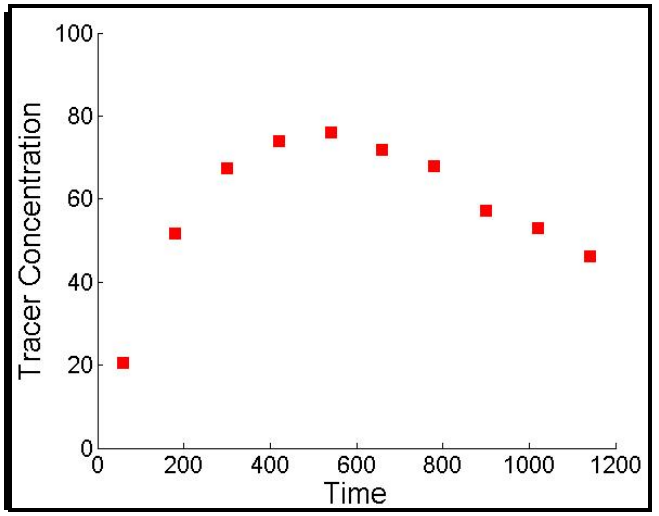
Cerebral Perfusion ⇔ Cerebral Blood Flow (CBF)

How can Perfusion be Measured?

1. Inject blood tracer
2. Monitor concentration

in tissue over time

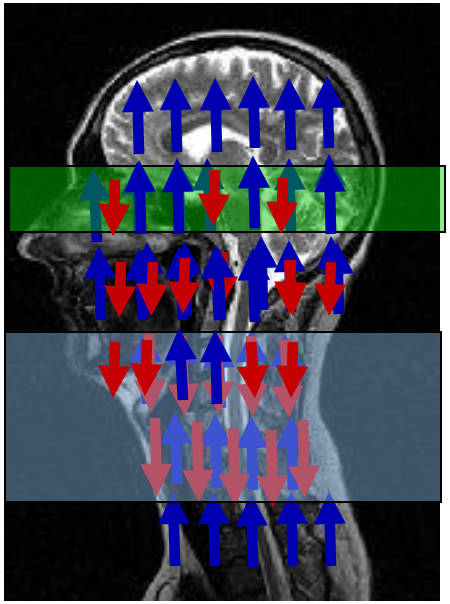
3. Compute CBF using a model



The ASL Experiment

1. Labelling: Invert magnetization in the neck

- Labelled blood moves away
- Magnetization recovers with tissue-specific time constant T_1



2. Labelled Image: Collect image in thin slice in the head

- Signal comes both from labelled blood and static tissue

4. Control Image: Acquire an image without labelling blood

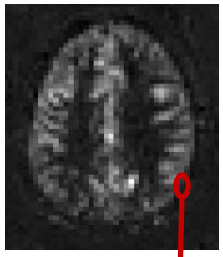
The ASL Kinetic Model

Control Image

-

Labelled Image

=

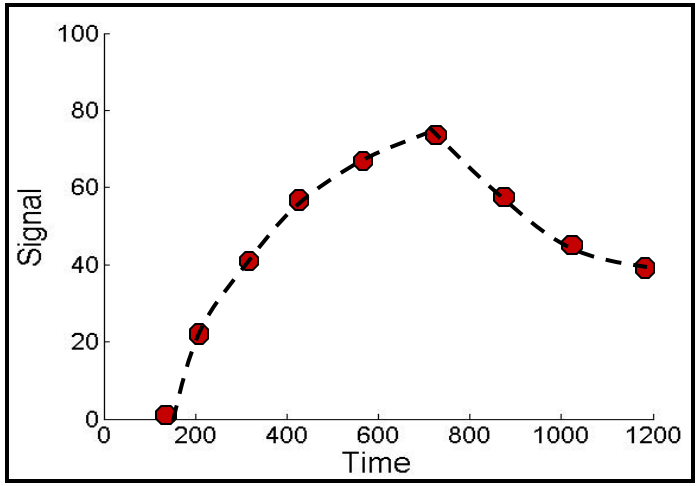


Perfusion Weighted Image

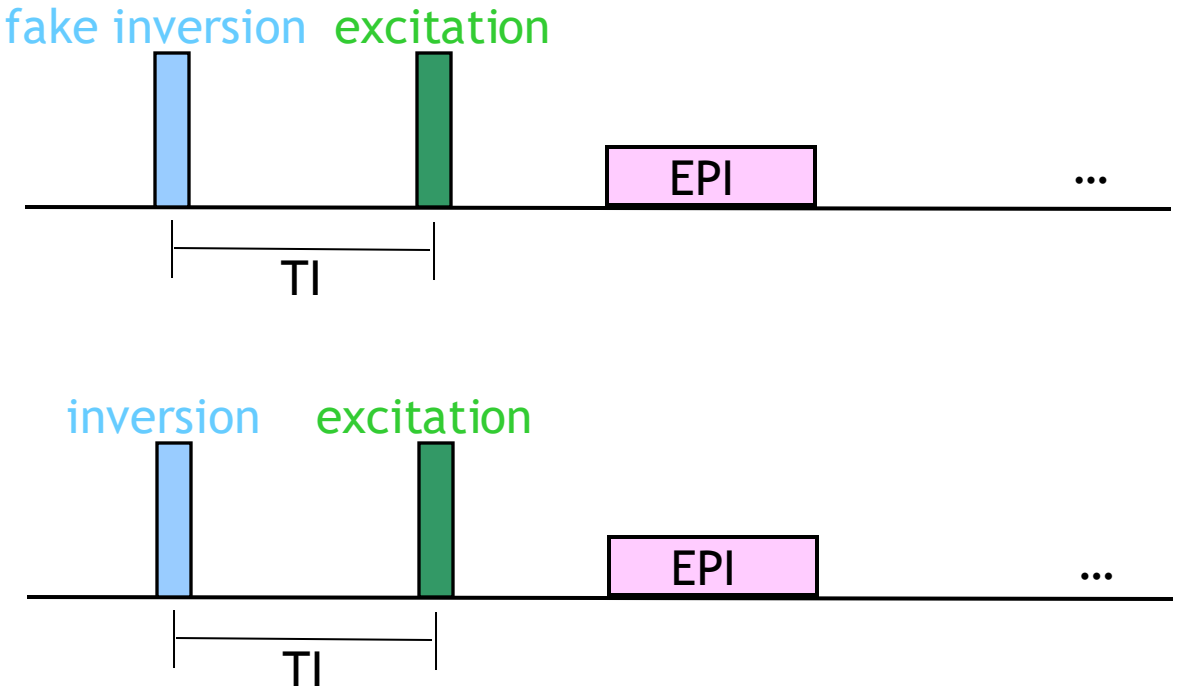
(100 averages)

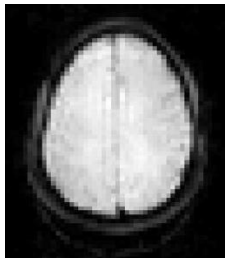
5. Repeat the experiment changing time gap between labelling and image acquisition

6. Fit data to kinetic model to extract CBF



Perfusion Weighted Images



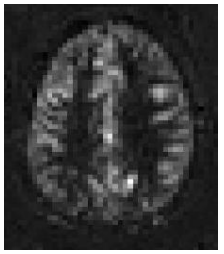


Control Image

-

Labelled Image

=



Perfusion Weighted Image
(100 averages)

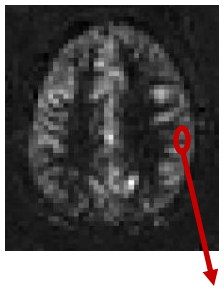
5. Do many averages to increase SNR

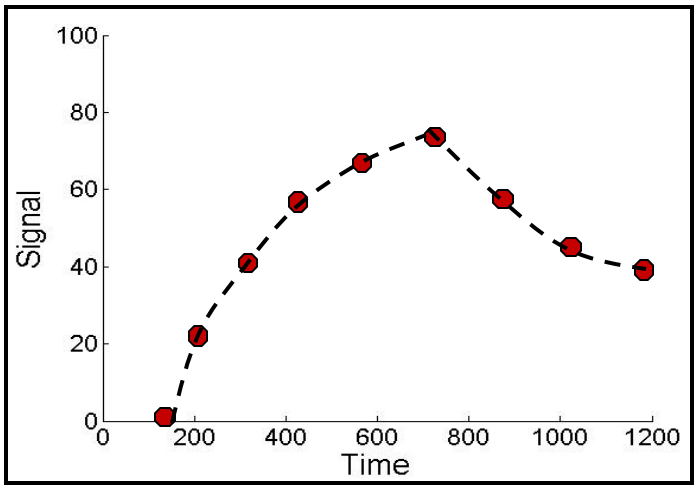
Acquiring Images at Multiple TIs

5. Repeat the experiment changing time gap between labelling and image acquisition

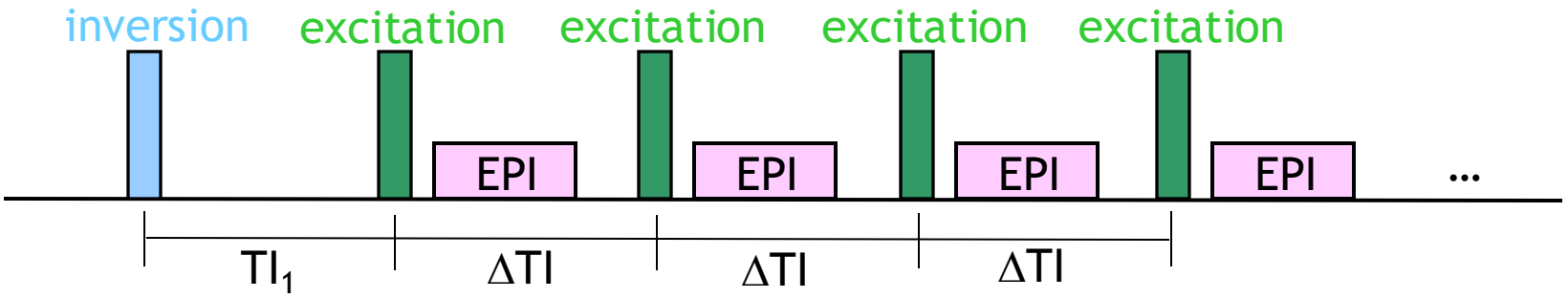
6. Fit data to kinetic model to extract CBF

Perfusion Weighted Image





Or… faster (but giving lower SNR):



(Simplified) ASL Model

Parameter		Healthy Adults	Neonates
T_{1b}	Longitudinal Recovery Time of Arterial Blood	1500 ms	Very variable
CBF	Cerebral Blood Flow	60 mL blood/100 g tissue/min	Age dependent
τ	Bolus Duration		
Δt	Bolus Arrival Time	-	

$$\frac{dS}{dt} = \begin{cases} 0 & \text{if } t < AT \\ CBF \times e^{-t/T_{1b}} \times \left(1 - \frac{t-AT}{T_{1b}}\right) & \text{if } AT \leq t < AT + \tau \\ -CBF \times e^{-t/T_{1b}} \times \frac{\tau}{T_{1b}} & \text{if } AT + \tau \leq t \end{cases}$$

Tutorial Exercise 2

- Model the propagation of electrical signals in cardiac tissue in a 2D rectangular geometry.

- https://github.com/annien094/PINNs-tutorial-MICCAI-2024/blob/main/PINNs_AP2D.ipynb

Scan for the exercise notebook

IMPERIAL

IMPERIAL

Physics-informed neural networks for cardiac electrophysiology

in 3D and fibrillatory conditions

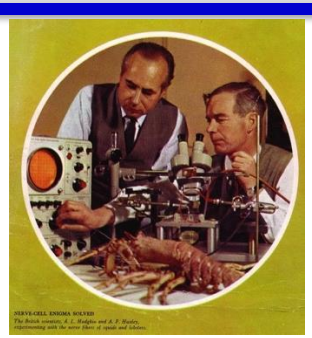
Annie Ching-En Chiu, Aditi Roy, Sarah Cechnicka,
Arieh Levy, Ashvin Gupta, Christoforos Galazis, Kim Christensen,
Danilo Mandic, Marta Varela

10th Oct 2024 @ STACOM workshop

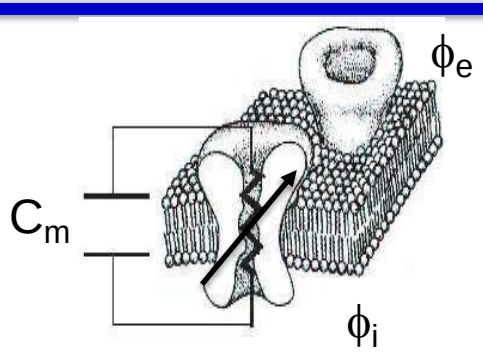
MICCAI 2024 Marrakesh MOROCCO

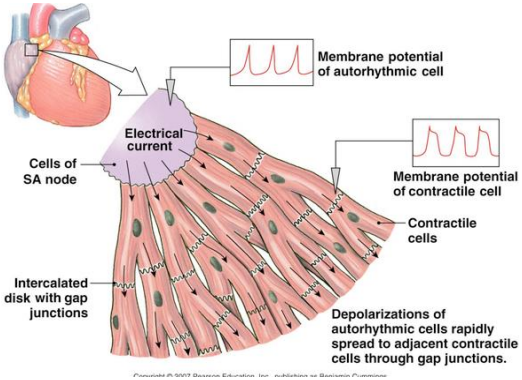
https://arxiv.org/pdf/2409.12712

Propagating Action Potentials



A. Hodgkin & A. Huxley,
Nobel for Prize Medicine
& Physiology, 1962

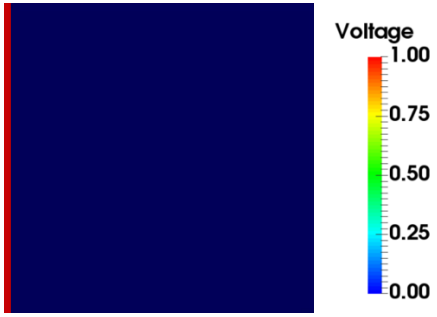




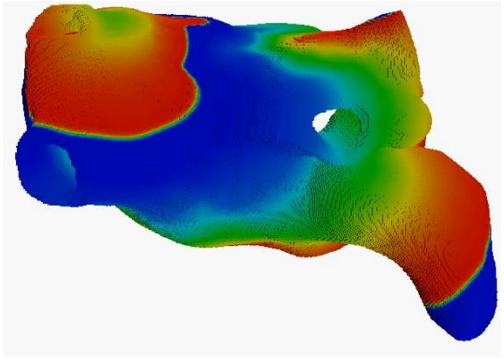
V "diffuses" to neighbouring cells.

$$\frac{\partial V}{\partial t} = \nabla.(D\nabla V) - \frac{I_{ion}(V,t)}{C_m}$$

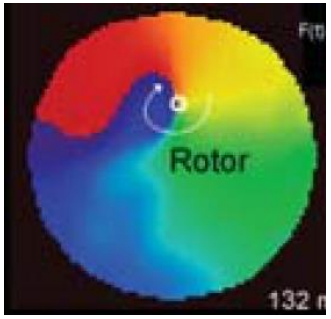
D : Diffusion tensor



Travelling Wave Solutions

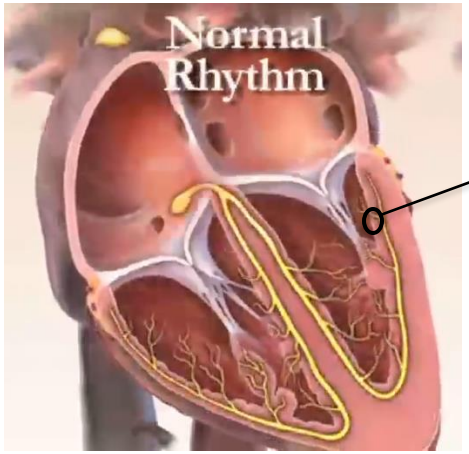


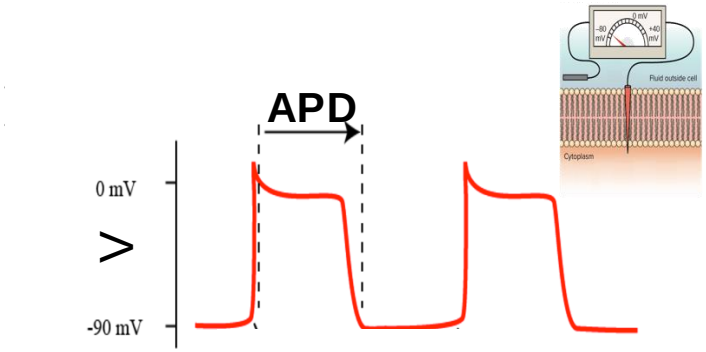
Re-entrant waves (rotors) as drivers of arrhythmias.



IMPERIAL

Modelling Cardiac Electrophysiology



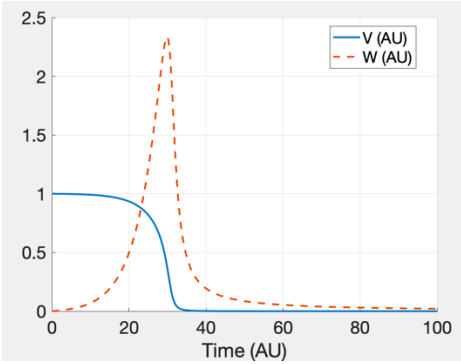


$$\frac{\partial V}{\partial t} = \vec{\nabla}.(D\vec{\nabla}V) - kV(V - a)(V - 1) - VW$$

$$\frac{\mathrm{d}W}{\mathrm{d}t} = (\epsilon + \frac{\mu_1 W}{V + \mu_2})(-W - kV(V - b - 1))$$

We can estimate:

- D is the (scalar) diffusion coefficient
- a "controls" the steepness of the action potential
- b "controls" action potential duration (APD)



Aliev & Panfilov, Chaos, 1996

IMPERIAL