Physics Informed Neural Networks

Starting Point: Ordinary Diff Eqs (ODE)'s

<u>Problem</u>: Find $u(t)$ on $[0,1]$
given $u(0)$ and $u'(t) = f(u,t)$

Demo ode_simple.jl

One classical Method

Assume a finite dimensional vector space as a solution guess

e.g. $u(t) = \sum_{k=0}^{n} c_k t^k$ or $u(t) = \sum c_k \sin 2\pi k t$ etc

Take n points $0 < t_1 < \dots < t_n \le 1$

Solve
$$c_0 = u(0)$$
$$\sum_{k=1}^{n} k c_k t_i^{k-1} = f\left(\sum_{k=0}^{n} c_k t_i^k, t_i\right)$$
for $i = 1, \dots, n$

This is linear if $u'(t)$ ~~only~~ depends on t

For a PINN we assume $u(t)$ is a neural network

The Universal Approximation Theorem

In layman's terms, any (nice) function from $\mathbb{R}^n$ to $\mathbb{R}^m$ can be well approximated by a neural network if you allow yourself enough parameters and layers

Polynomials have this property

$F(x) = \sum \cancel{x \cdot x}$

$$c_{[\,]} + \sum c_i x_i + \sum c_{ij} x_i x_j + \sum c_{ijk} x_i x_j x_k + \cdots$$

but polynomials tend to need exponentially (in the # parameters) more terms as compared to neural networks

Machine Learning not from data but from functions