Neural Discovery

Patrick AK Reinbold

1 Introduction

We want to discover models for dynamical systems, and we can use neural networks to do so. The inputs shall be (snippets of) trajectories and the outputs shall be the model parameters. In other words, we modify the multi-label classification problem to suit a model discovery framework. For instance, say that our dynamical system can be described by 2 variables x and y, and their evolution is governed by

$$\begin{aligned} \dot{x} &= \sum_n a_n f_n(x,y), \\ \dot{y} &= \sum_n b_n f_n(x,y), \end{aligned} \tag{1}$$

where f_n can be of arbitrary complexity, and even have its own parameterization.

Let us pose the learning problem as training a model

$$\mathcal{M} : \{x, y\}_t \to \{a, b\}_n, \tag{2}$$

where $\{x, y\}_t$ denotes observations of some trajectory, and $\{a, b\}_n$ denotes the parameters of the governing equation. The specifics of the discretization of the trajectory and the parameters will be discussed in subsequent sections.

2 Fitting Parameters with Fixed ODE Trajectory

To start, let us only consider learning parameters that we know are present, and train on trajectories that are always observed at the same points in time.

2.1 Systems

For example, consider the damped physical pendulum, whose torque balance equation can be written as

$$\ddot{\theta} + \frac{b}{m}\dot{\theta} - \frac{g}{r}\sin\theta = 0. \tag{3}$$

Reframing this as a first order equation:

$$\dot{\theta} = \omega \tag{4}$$

$$\dot{\omega} = c_1\omega + c_2\sin\theta \tag{5}$$

where $c_1 = -b/m$ and $c_2 = g/r$. Let us try and learn c_1 and c_2 using observations of ω and θ. Each training sample z_i will then be a stacking of observations:

$$z_i = \begin{bmatrix} \theta_i(t_1) \\ \theta_i(t_2) \\ \vdots \\ \theta_i(T) \\ \omega_i(t_1) \\ \omega_i(t_2) \\ \vdots \\ \omega_i(T) \end{bmatrix}, \tag{6}$$

where i denotes a particular simulation with parameters $\vec{c}_i$. Then all we must do is train our neural net

$$\mathcal{M} : \{z\}_i \to \{\vec{c}\}_i \tag{7}$$

2.2 Neural Net Parameter Scaling

- Layers
- Activation Function
- Neurons
- Amount of Training Data

2.3 Trajectory Observation Scaling

3 Fitting A Generic Library for Fixed ODE Trajectory

4 Fitting Parameters with Flexible ODE Trajectory