Tutorial 2
(Neural Computing)

Artur S. d'Avila Garcez
Department of Computer Science, City, University of London

EXERCISE 4 (hands-on tutorial using the MATLAB Neural Networks Toolbox)
A single hidden layer neural network is to be trained to balance a pole in gravity by applying a suitable force. In our model (see figure below), we assume that the pole itself is massless, and has masses fixed to each end. The movement of the pole is confined to a vertical plane. The foot of the pole can only be moved horizontally by applying a force F. In our units, the gravitational constant, the length of the pole, and the masses are assumed to be 1.

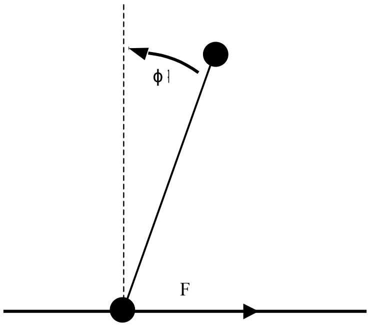

Figure 1: Pole in gravity

The movement of the pole is given by Equation 1, where the angle $\phi(t)$ at time t is measured counter-clockwise against the vertical, $\dot{\phi}(t)$ is the angular velocity and $\ddot{\phi}(t)$ is the angular acceleration. It turns out that applying the force F given by Equation 2 balances the pole reasonably well within certain angular limits.

$$(1+\sin^2\phi(t))\ddot{\phi}(t) = -\frac{1}{2}\sin(2\phi(t))\dot{\phi}(t)^2 + 2\sin\phi(t) - F(t)\cos\phi(t) \tag{1}$$

$$F(t) = 5\sin\phi(t) + \dot{\phi}(t) \tag{2}$$

Train a neural network with standard *backpropagation* to compute the force F. Hint: create a training and a test set with random input pairs $(\phi, \dot{\phi}) \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, and output F taken from Equation 2. Alternatively, use the examples available from the module's webpage on Moodle (file pole_balancing.txt), which were obtained from a real pole balancing experiment, to create training and validation sets.