**Toolbox for Estimating Parameters of Reinforcement Learning Models**

Created by Jee Hoon Yoo, University of Bristol, September 2008

**Overview**

The reinforcement learning models in Computational neuroscience are designed to fit the human behavioural data. They have been successfully used to predict human behaviours in exploration/exploitation dilemma. Also, they are useful to explain some effects observed in neural activity.

These models are controlled by a number of parameters (e.g. learning rate, exploration parameter of Softmax method, initial action values, etc.). In order to describe human behaviour in an experiment, the values of these parameters have to be estimated from behavioural data.

Below you can download MATLAB scripts that attempt to find the best fit parameters to the behavioural data and negative log likelihoods of data given models, and generate information useful for analysis of fMRI (i.e. temporal difference prediction error for each trial and classification if the trial is explorative or exploitative). The toolbox can be used for analysis of data from experiments consisting of sequences of choice trials, such that in each trial participant makes a choice between a certain number of alternatives and receives a numerical feedback indicating amount of reward or loss.

**Models**

The scripts allow estimating parameters of 3 different models:

* **The indirect actor** :  The standard basic reinforcement learning model described in Chapter 9 of [1].  Updating procedure: *mi* = *mi* + εδ where δ = *ri* - *mi* In the equation, *mi* is the action value of the chosen alternative. ε denotes the learning rate,  and δ represents the difference between received reward *ri* and action value *mi*.
* **The indirect with the decay of action value** :  The decay of action values based on [2] is added to the standard indirect actor.  The chosen alternative's action value is updated by above updating procedure. Also, all  the action values are decreased by following equation regardless of choice: *mi* = λ*mi* + (1 - λ)θ In the equation, θ is the value towards converging point which action values decay,  and λ represents decaying rate.
* **Kalman filter** :  All details are described in [2].

**Download**

[Click here to download MATLAB script for *the indirect actor* model](http://www.cs.bris.ac.uk/home/rafal/rltoolbox/indirectActor.zip)

In order to find out how to use the above script, download and uncompress the zip file, and then type in MATLAB:

help indirectActorAuto

The above link also includes a very simple example of a behavioural data. To see an example how to use the indirect actor model for this simple behavioural data, type in MATLAB:

indirectActorAuto('example');

To run the simple example, type in MATLAB:

indirectActorAuto('testrun');

[Click here to download MATLAB script for *the indirect actor with the decay of action value* model](http://www.cs.bris.ac.uk/home/rafal/rltoolbox/indirectActorDecay.zip)

In order to find out how to use the above script, download and uncompress the zip file, and then type in MATLAB:

help indirectActor2Auto

The above link also includes a very simple example of a behavioural data. To see an example how to use the indirect actor model for this simple behavioural data, type in MATLAB:

indirectActor2Auto('example');

To run the simple example, type in MATLAB:

indirectActor2Auto('testrun');

[Click here to download MATLAB script for *Kalman filter* model](http://www.cs.bris.ac.uk/home/rafal/rltoolbox/kalmanFilter.zip)

In order to find out how to use the above script, download and uncompress the zip file, and then type in MATLAB:

help kalmanAuto

The above link also includes a very simple example of a behavioural data. To see an example how to use Kalman filter model for this simple behavioural data, type in MATLAB:

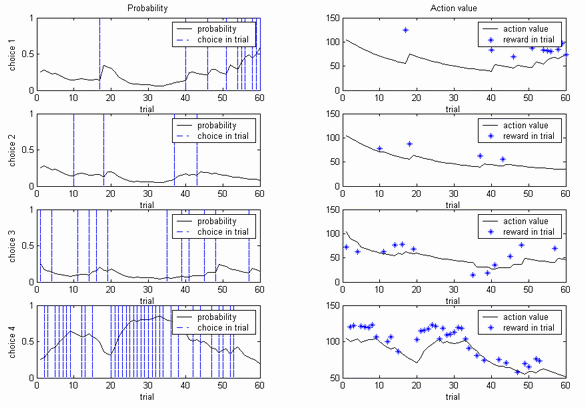
kalmanAuto('example');

To run the simple example, type in MATLAB:

kalmanAuto('testrun');

**Visualization**

Each script also performs visualization for describing the changes of probabilities predicted by the model and action values.



Two plots in each row represent the changes in probabilities predicted by the model and action values of one choice. For example, in the 1st row, there are two plots for 1st choice.

In the graph of probability, the black line describes the changes in probabilities, and the vertical line indicates that the participant chose the corresponding alternative in a given trial. In the graph of action value, the black line depicts the changes in action values, and the star represents the reward in each trial. For instance, the participant visualized above chose the 2nd alternative on 10th trial, hence two graphs corresponding to the 2 alternative (in 2nd row) show the vertical line and the star in 10th trial respectively.

In order to find out how to use the visualization script, type in MATLAB:

help resultplot

To see an example how to use visualization script for this simple behavioural data, type in MATLAB:

resultplot('example');

**Estimation method in MATLAB scripts**

The scripts maximize the likelihood of behavioural data given the model expressed by:

L = ∏Nk=1∏Tt=1P(choicek,t)

where

N - number of participants

T - number of trials per participant

P(choicek,t) - the likelihood of choice made by participant k at trial t given the model (computed using Softmax on the basis of mi).

The parameters maximizing the likelihood L are found using MATLAB's function 'fminsearch'.

**References**

[1] Peter Dayan and L. F. Abbott, Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems, The MIT Press, London, 2001

[2] Nathaniel D. Daw, John P. O'Doherty, Peter Dayan, Ben Seymour and Raymond J. Dolan, Cortical substrates for exploratory decisions in humans, Volume 441, 876 - 879, 15 June 2006 Nature.