



Practical Part: Choice Behaviour as Active Inference

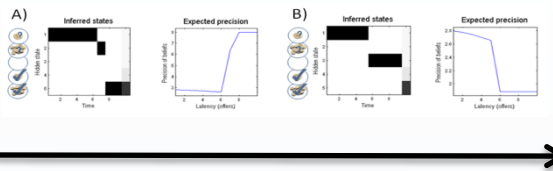
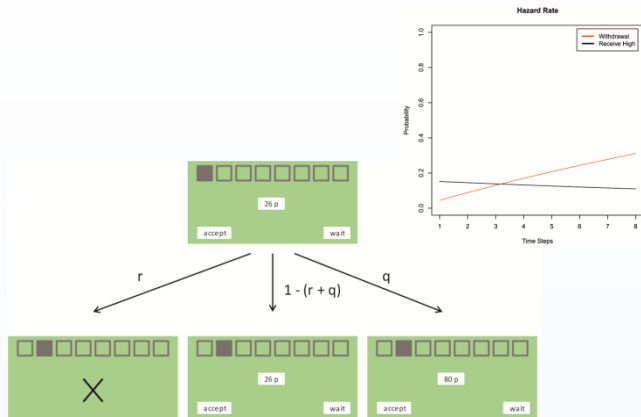
Philipp Schwartenbeck

Centre for Cognitive Neuroscience, Salzburg
Wellcome Trust Centre for Neuroimaging, UCL

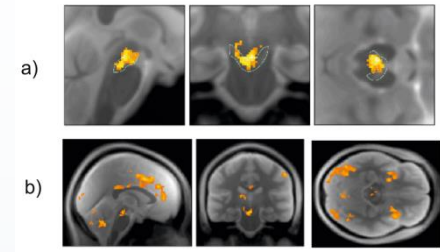
Computational Psychiatry Course, TNU, Zurich

What can we use the active inference toolbox for?

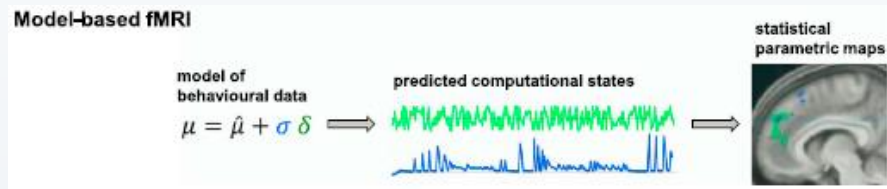
1. Task & Behaviour



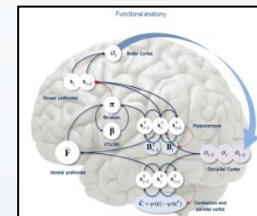
2. Invert generative model & Predict brain function



3. Model-based Neuroimaging

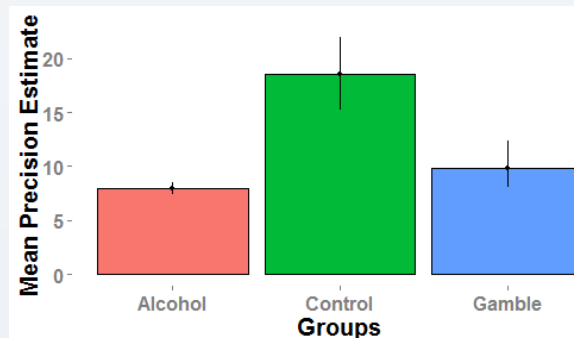


Stephan et al. 2015 *Neuron*



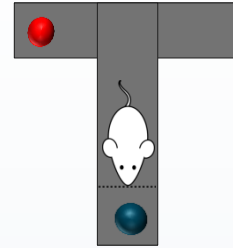
Friston et al., in preparation

4. Apply to psychiatry!



Schwartenbeck et al., 2015, *Cerebral Cortex*

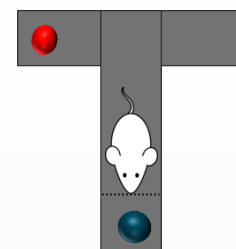
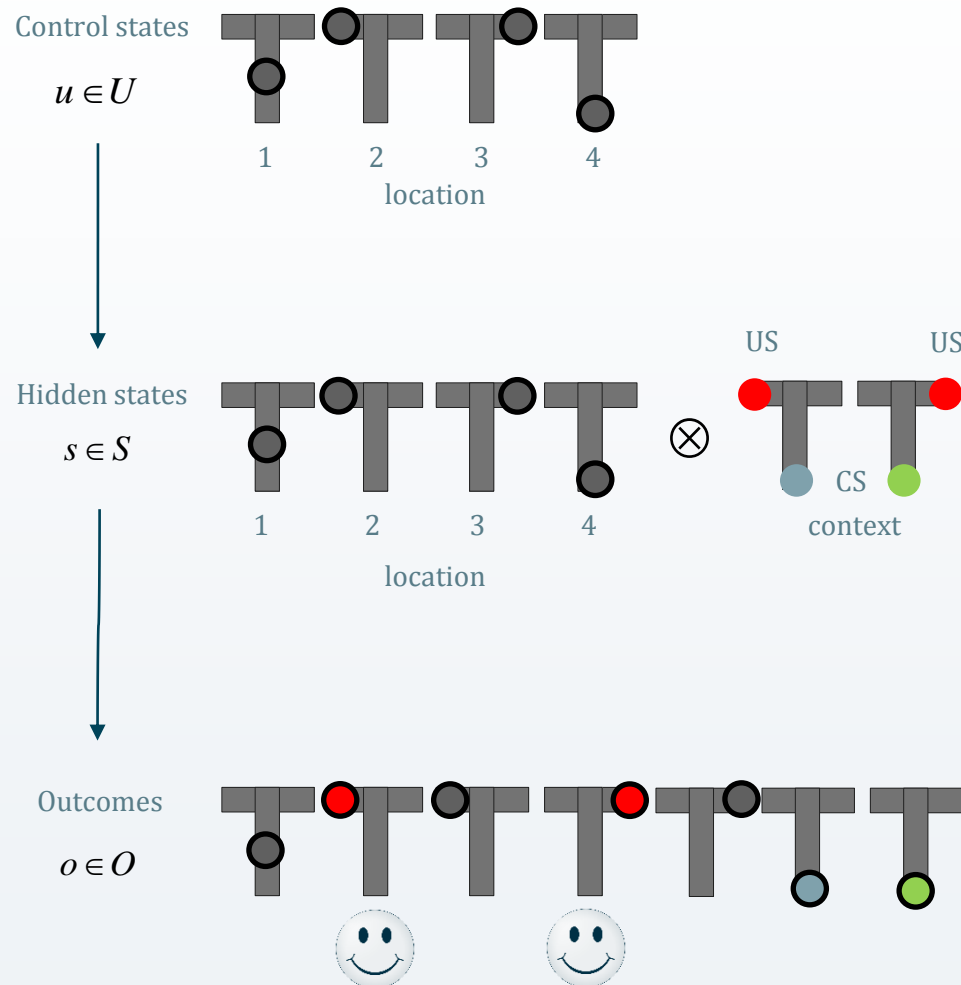
Practical example: a two-step maze task



Overview

- ❖ How to model a task involving epistemic and exploitative behaviour
- ❖ Inversion of single subject/ group data to obtain subject-/group-specific parameters
- ❖ Perform hierarchical (empirical) Bayesian inference on second level (group) effects

Generative Model: a two-step maze task

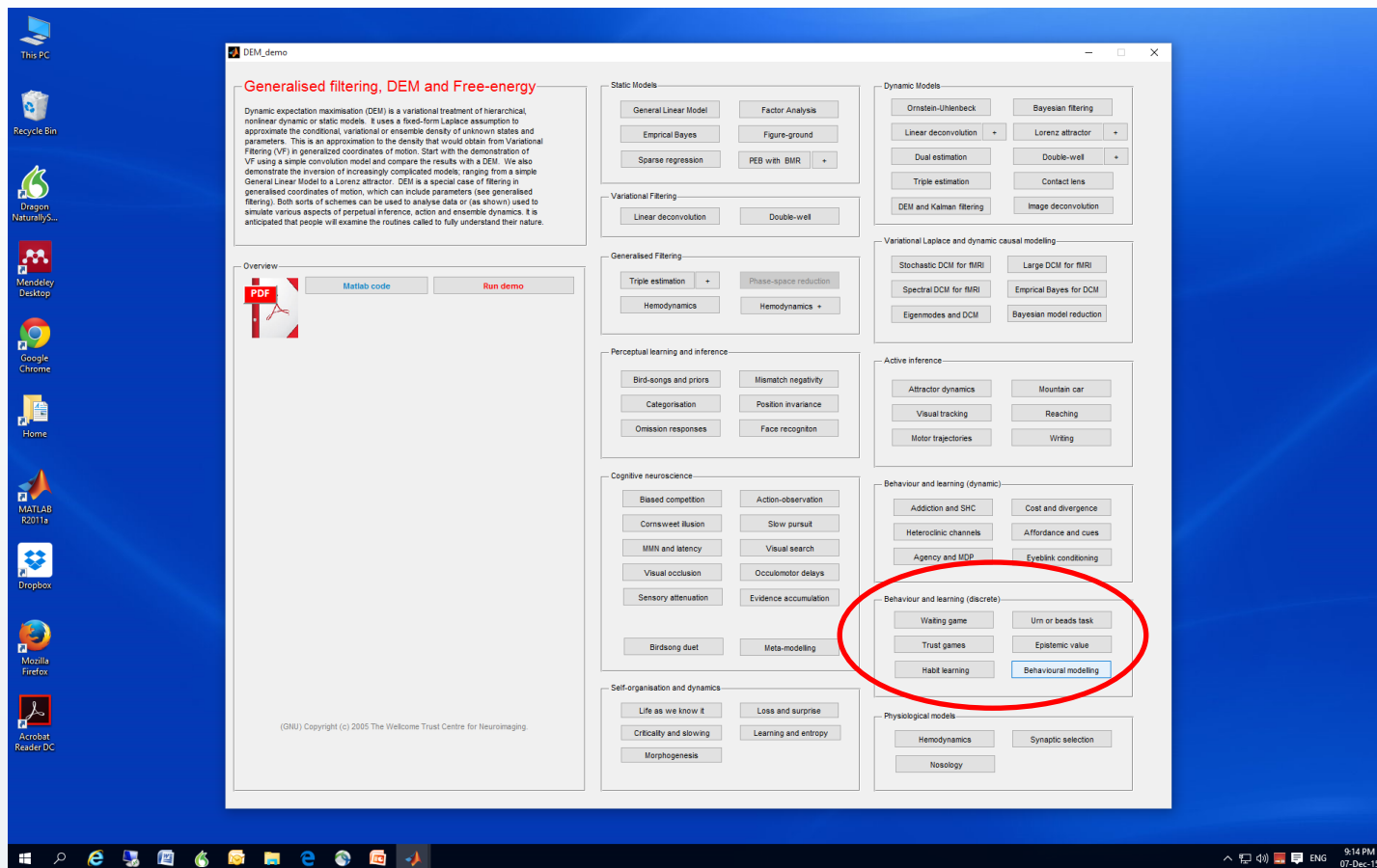


Generative model

Now let's take a look at the MDP toolbox!

- ❖ The ABC of (MDP) model specification
- ❖ Simulating choice behaviour (`spm_MDP_VB`)
- ❖ Fitting single subject choices (`spm_dcm_mdp`)
- ❖ Empirical Bayes for group analyses (`spm_dcm_peb`)
- ❖ Out of sample (CV) estimates (`spm_dcm_loo`)

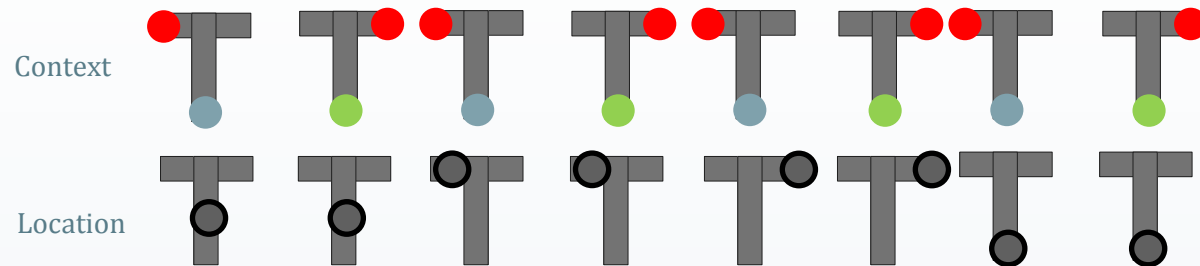
Now let's take a look at the MDP toolbox!



Mapping from states to outcomes (A-Matrix)

$$p = 0.95$$


$$q = 1 - p$$

































$$A = P(o_t | s_t) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & p & q & 0 & 0 & 0 & 0 \\ 0 & 0 & q & p & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q & p & 0 & 0 \\ 0 & 0 & 0 & 0 & p & q & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Outcomes

Transition-Probabilities (B-Matrix)



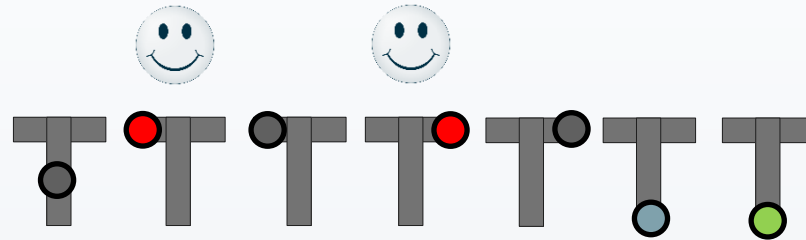
$$B(u = 2) = P(s_{t+1}|s_t, u = 2) =$$

Context									
Location									
	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0		
	1	0	1	0	0	0	1		
	0	1	0	1	0	0	1		
	0	0	0	0	1	0	0		
	0	0	0	0	0	1	0		
	0	0	0	0	0	0	0		
	0	0	0	0	0	0	0		

Also for $B(u = 1)$, $B(u = 3)$ and $B(u = 4)$...

Preferences over Outcomes (c-vector)

Outcomes



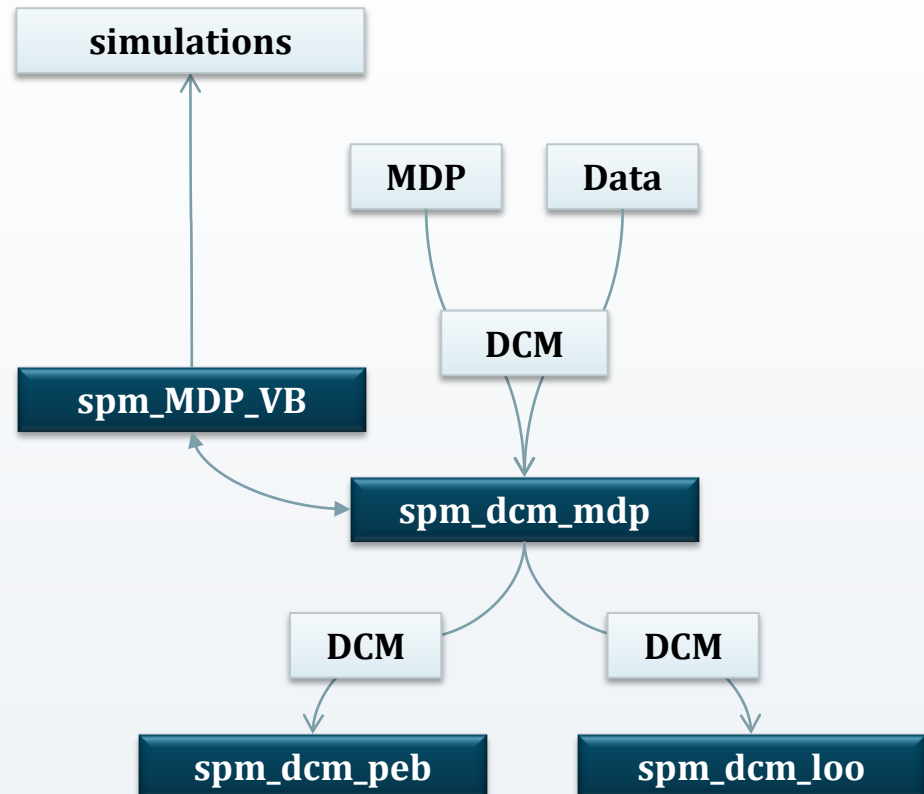
$$c = \ln P(o_t) = [0 \quad 3 \quad -3 \quad 3 \quad -3 \quad 0 \quad 0]^T$$

Code & Results

- ❖ The ABC of (MDP) model specification
- ❖ Simulating choice behaviour (spm_MDP_VB)
- ❖ Fitting single subject choices (spm_dcm_mdp)
- ❖ Empirical Bayes for group analyses (spm_dcm_peb)
- ❖ Out of sample (CV) estimates (spm_dcm_loo)

A demo to illustrate the basic ideas (DEM Toolbox)

```
% This routine uses a Markov decision process formulation of active
% inference (with variational Bayes) to model foraging for information in a
% three arm maze. This demo illustrates the inversion of a single subject
% and group data to make inferences about subject specific parameters -
% such as their prior beliefs about precision and utility.
%
% We first generate some synthetic data for a single subject and illustrate
% the recovery of key parameters using variational Laplace. We then
% consider the inversion of multiple trials from a group of subjects to
% illustrate the use of empirical Bayes in making inferences at the between
% subject level - and the use of Bayesian cross-validation to retrieve out
% of sample estimates (and classification of new subjects)
%
% In this example, the agent starts at the centre of a three way maze
% which is baited with a reward in one of the two upper arms. However, the
% rewarded arm changes from trial to trial. Crucially, the agent can
% identify where the reward (US) is located by accessing a cue (CS) in the
% lower arm. This tells the agent whether the reward is on the left or the
% right upper arm. This means the optimal policy would first involve
% maximising information gain or epistemic value by moving to the lower arm
% and then claiming the reward this signified. Here, there are eight hidden
% states (four locations times right or left reward), four control states
% (that take the agent to the four locations) and seven outcomes (three
% locations times two cues plus the centre). The central location has an
% ambiguous or uninformative cue outcome, while the upper arms are rewarded
% probabilistically.
%
% see also: spm_MPD_VB.m, spm_dcm_mdp.m and spm_nlsi_Newton.m
%
% Copyright (C) 2005 Wellcome Trust Centre for Neuroimaging
%
% Karl Friston
```



```
%% set up and preliminaries: first generate synthetic (single subject) data
%=====
rng('default')

% outcome probabilities: A
%-----
% We start by specifying the probabilistic mapping from hidden states
% to outcomes.
%-----
a      = .95;
b      = 1 - a;
A      = [1 1 0 0 0 0 0 0;    % ambiguous starting position (centre)
          0 0 a b 0 0 0 0;    % left arm selected and rewarded
          0 0 b a 0 0 0 0;    % left arm selected and not rewarded
          0 0 0 0 b a 0 0;    % right arm selected and not rewarded
          0 0 0 0 a b 0 0;    % right arm selected and rewarded
          0 0 0 0 0 0 1 0;    % informative cue - reward on right
          0 0 0 0 0 0 0 1];   % informative cue - reward on left

% controlled transitions: B{u}|
%-----
% Next, we have to specify the probabilistic transitions of hidden states
% under each action or control state. Here, there are four actions taking the
% agent directly to each of the four locations.
%-----
B{1}   = [1 0 0 1; 0 1 0 0; 0 0 1 0; 0 0 0 0];    % move to the middle
B{2}   = [0 0 0 0; 1 1 0 1; 0 0 1 0; 0 0 0 0];    % move up left (and check for reward)
B{3}   = [0 0 0 0; 0 1 0 0; 1 0 1 1; 0 0 0 0];    % move up right (and check for reward)
B{4}   = [0 0 0 0; 0 1 0 0; 0 0 1 0; 1 0 0 1];    % move down (check cue)

for i = 1:4
    B{i} = kron(B{i},eye(2));
end
```

```
% priors: (utility) C
%-----
% Finally, we have to specify the prior preferences in terms of log
% probabilities. Here, the agent prefers rewarding outcomes
%-----
c      = 2;
C      = [0 c -c c -c 0 0]';

% now specify prior beliefs about initial state, in terms of counts
%-----
d      = kron([1 0 0 0],[1 1])';

% allowable policies (of depth T). These are just sequences of actions
%-----
V      = [1 1 1 1 2 3 4 4 4 4
          1 2 3 4 2 3 1 2 3 4];
```

❖ The ABC of (MDP) model specification

- ❖ Simulating choice behaviour (spm_MDP_VB)
- ❖ Fitting single subject choices (spm_dcm_mdp)
- ❖ Empirical Bayes for group analyses (spm_dcm_peb)
- ❖ Out of sample (CV) estimates (spm_dcm_loo)

```
%% MDP Structure - this will be used to generate arrays for multiple trials
%=====
mdp.V = V;                % allowable policies
mdp.A = A;                % observation model
mdp.B = B;                % transition probabilities
mdp.C = C;                % preferred states
mdp.D = d;                % prior over initial states
mdp.s = 1;                % initial state

mdp.alpha = 2;            % precision of action selection
mdp.beta  = 1;            % inverse precision of policy selection

% true parameters
%-----
n      = 128;              % number of trials
i      = rand(1,n) > 1/2; % randomise hidden states over trials
P.beta = log(2);
P.C     = log(2);

MDP     = mdp;
MDP.C   = mdp.C*exp(P.C);
MDP.beta = mdp.beta*exp(P.beta);

[MDP(1:n)] = deal(MDP);
[MDP(i).s] = deal(2);
```

```
%% Solve to generate data
%-----

MDP = spm_MDP_VB(MDP);

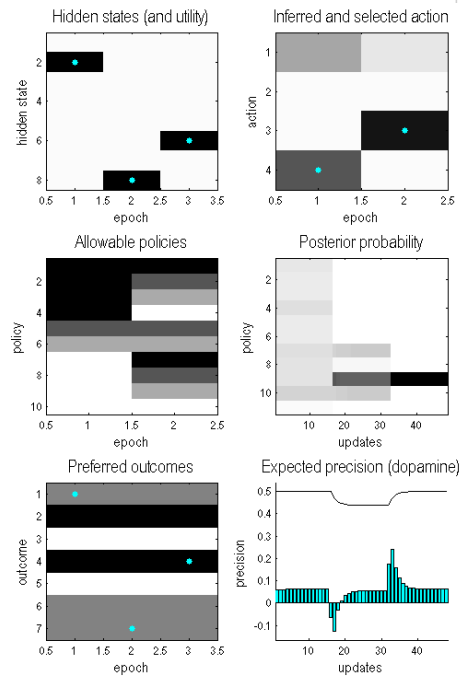
% illustrate behavioural responses - single trial
%-----

spm_figure('GetWin','Figure 1a'); clf
spm_MDP_VB_trial(MDP(1));

% illustrate behavioural responses and neuronal correlates over trials
%-----

spm_figure('GetWin','Figure 1b'); clf
spm_MDP_VB_game(MDP);

%-----
% This completes the generation of data. We now turn to the estimation of
% subject specific preferences and precision encoded by the parameters
% beta and C. Model parameters here are log scaling parameters that allow
% for increases or decreases in the default prior values.
%-----
```



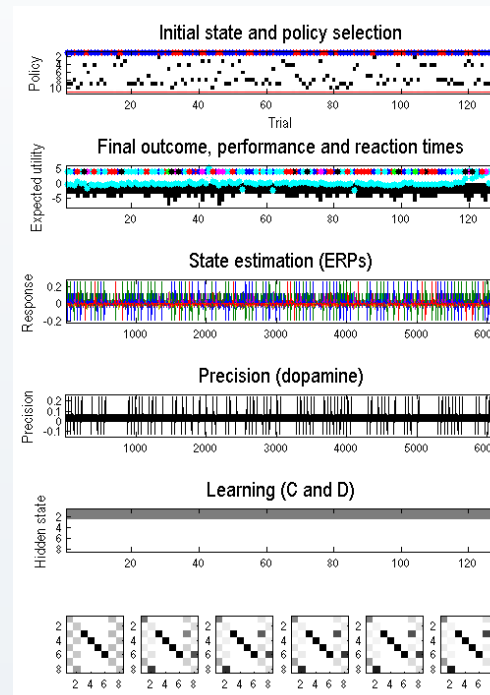
❖ The ABC of (MDP) model specification

❖ Simulating choice behaviour (spm_MDP_VB)

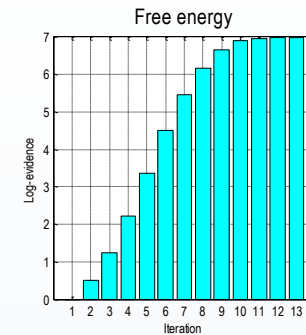
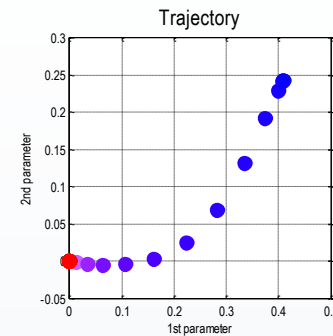
❖ Fitting single subject choices (spm_dcm_mdp)

❖ Empirical Bayes for group analyses (spm_dcm_peb)

❖ Out of sample (CV) estimates (spm_dcm_loo)



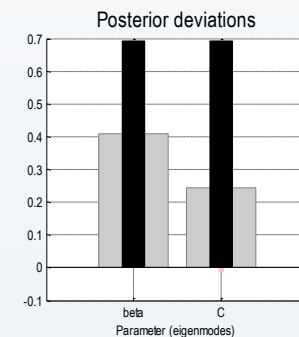
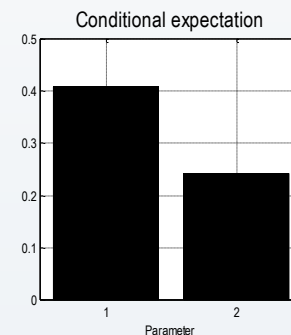
- ❖ The ABC of (MDP) model specification
- ❖ Simulating choice behaviour (spm_MDP_VB)
- ❖ Fitting single subject choices (spm_dcm_mdp)
- ❖ Empirical Bayes for group analyses (spm_dcm_peb)
- ❖ Out of sample (CV) estimates (spm_dcm_loo)



```
%% Invert to recover parameters (preferences and precision)
%-----
DCM.MDP = mdp; % MDP model
DCM.field = {'beta','C'}; % parameter (field) names to optimise
DCM.U = {MDP.o}; % trial specification (stimuli)
DCM.Y = {MDP.u}; % responses (action)

DCM = spm_dcm_mdp(DCM);

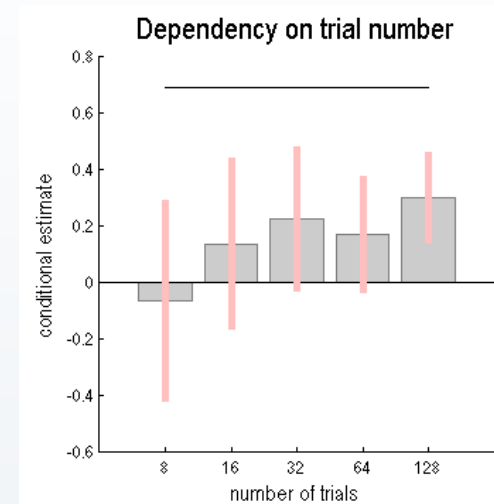
% compare true values with posterior estimates
%-----
subplot(2,2,4),hold on
bar(spm_vec(P),1/4)
set(gca,'XTickLabel',DCM.field)
set(gcf,'Name','Figure 2','Tag','Figure 2')
```



Using simulations to optimise design parameters

```
%% now repeat using subsets of trials to illustrate effects on estimators
%=====
DCM.field = {'beta'};
n         = [8 16 32 64 128];
for i = 1:length(n)
    DCM.U = {MDP(1:n(i)).o};
    DCM.Y = {MDP(1:n(i)).u};
    DCM   = spm_dcm_mdp(DCM);
    Ep(i,1) = DCM.Ep.beta;
    Cp(i,1) = DCM.Cp;
end

% plus results
%-----
spm_figure('GetWin','Figure 3'); clf
subplot(2,1,1), spm_plot_ci(Ep(:),Cp(:)), hold on
plot(1:length(n), (n - n) + P.beta, 'k'), hold off
set(gca, 'XTickLabel', n)
xlabel('number of trials', 'FontSize', 12)
ylabel('conditional estimate', 'FontSize', 12)
title('Dependency on trial number', 'FontSize', 16)
axis square
```



Simulating group effects

```
%% now repeat but over multiple subjects with different beta
%=====

% generate data and a between subject model with two groups of eight
% subjects
%-----
N      = 8;                                % numbers of subjects per group
X      = kron([1 1; 1 -1], ones(N,1));    % design matrix
h      = 4;                                % between subject log precision
n      = 128;                              % number of trials
i      = rand(1,n) > 1/2;                  % randomise hidden states

clear MDP
[MDP(1:n)] = deal(mdp);
[MDP(i).s] = deal(2);

for i = 1:size(X,1)

    % true parameters - with a group difference of one quarter
    %-----
    beta(i) = X(i,:) * [0; 1/4] + exp(-h/2) * randn; % add random Gaussian effects to group means -> BMR and PEB
    [MDP.beta] = deal(exp(beta(i)));

    % solve to generate data
    %-----
    DDP      = spm_MDP_VB(MDP); % realisation for this subject
    DCM.U     = {DDP.o};        % trial specification (stimuli)
    DCM.Y     = {DDP.u};        % responses (action)
    GCM{i,1} = DCM;

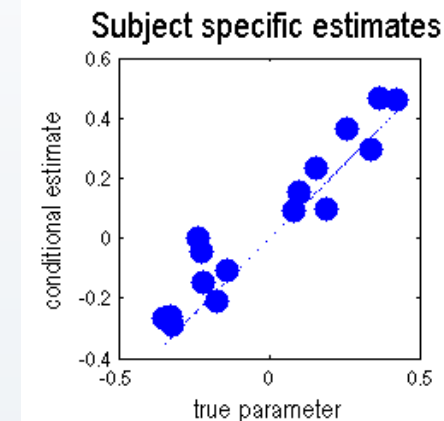
    % plot behavioural responses
    %-----
    spm_figure('GetWin', 'Figure 4'); clf
    spm_MDP_VB_game(DDP); drawnow

end
```


- ❖ The ABC of (MDP) model specification
- ❖ Simulating choice behaviour (spm_MDP_VB)
- ❖ Fitting single subject choices (spm_dcm_mdp) for multiple subjects (spm_dcm_fit)
- ❖ Empirical Bayes for group analyses (spm_dcm_peb)
- ❖ Out of sample (CV) estimates (spm_dcm_loo)

```
%% Bayesian model inversion
%=====
GCM = spm_dcm_fit(GCM);

% plot subject specific estimates and true values
%-----
spm_figure('GetWin','Figure 4');
subplot(3,1,3)
for i = 1:length(GCM)
    qP(i) = GCM{i}.Ep.beta;
end
plot(beta,beta,':b',beta,qP,':b','MarkerSize',32)
xlabel('true parameter','FontSize',12)
ylabel('conditional estimate','FontSize',12)
title('Subject specific estimates','FontSize',16)
axis square
```



```
%% hierarchical (empirical) Bayes
```

```
=====
```

```
% second level model
```

```
-----
```

```
M = struct('X',X);
```

```
% BMA - (second level)
```

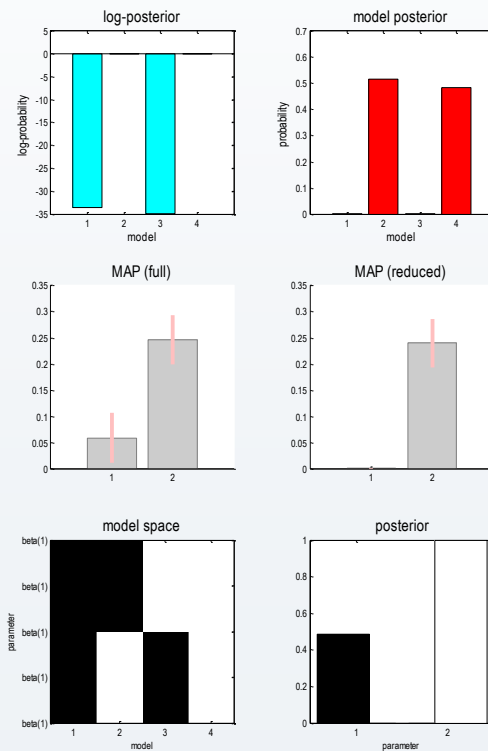
```
-----
```

```
PEB = spm_dcm_peb(GCM,M);
```

```
BMA = spm_dcm_peb_bmc(PEB);
```

```
subplot(3,2,4),hold on, bar(1,1/4,1/4), set(gca,'XTickLabel',DCM.field)
```

```
subplot(3,2,2),hold on, bar(1,1/4,1/4), set(gca,'XTickLabel',DCM.field)
```



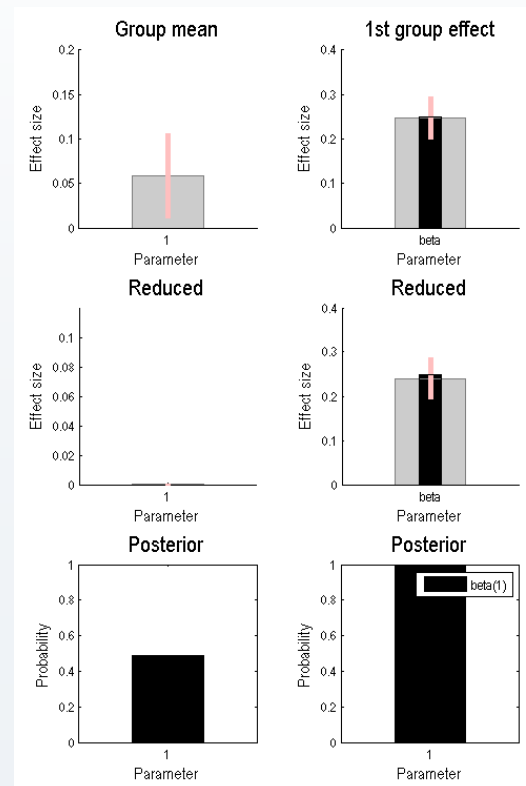
❖ The ABC of (MDP) model specification

❖ Simulating choice behaviour (spm_MDP_VB)

❖ Fitting single subject choices (spm_dcm_mdp)

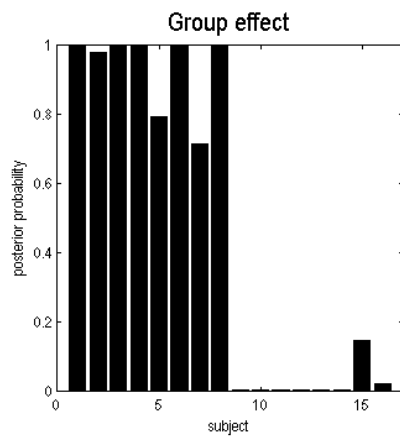
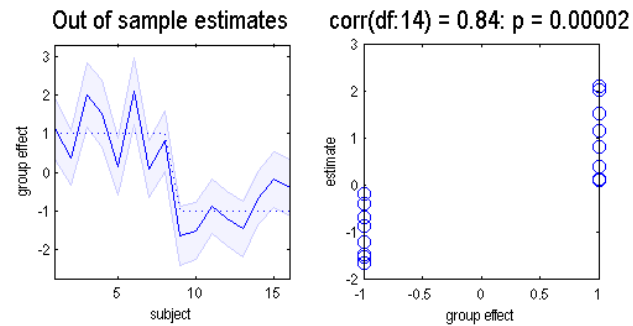
❖ Empirical Bayes for group analyses (spm_dcm_peb)

❖ Out of sample (CV) estimates (spm_dcm_loo)



```
%% posterior predictive density and cross validation
```

```
=====
spm_dcm_loo(GCM,M,DCM.field);
```



- ❖ The ABC of (MDP) model specification
- ❖ Simulating choice behaviour (`spm_MDP_VB`)
- ❖ Fitting single subject choices (`spm_dcm_mdp`)
- ❖ Empirical Bayes for group analyses (`spm_dcm_peb`)
- ❖ Out of sample (CV) estimates (`spm_dcm_loo`)

Active inference: Summary

1. Why is this model useful?
 - Cast decision-making as Bayesian inference
 - Distinction between inference and learning
 - Rests on a normative theory based on belief updating
2. Where can we use it?
 - Any sort of inference or planning problem
 - Particularly, problems that involve *decisions* or *choices*
3. Where can't we use it?
 - Until recently: learning
 - Continuous state space models (that call for predictive coding)
4. What do we like about it?
 - Applies free energy principle to decision-making
 - Highlights role of (prior) beliefs in behaviour
5. What are the most common mistakes?
 - Inappropriate state space specification

Many thanks to:



Thank you for listening!