A beginner's guide to Active Inference

- Practical Part

Philipp Schwartenbeck

Wellcome Trust Centre for Human Neuroimaging, UCL

Computational Psychiatry Course 2018, TNU, Zurich

Requirements

Matlab

SPM12

‘Practical_*’ files and adjusted functions (‘Z_*’)

See also https://github.com/schwartenbeckph/CPC_ActiveInference2018

What sort of problems can we address?

Partially observable Markov Decision Processes

- Inference on hidden states and policies
- Planning

Exploitation-Exploration problems

- Information-gain, active Learning

Any task with discrete choices, reaction time data, neural data…

- From economic choices to visual search paradigms
- Ideally problems that involve *behaviour*

Computational Phenotyping in active inference

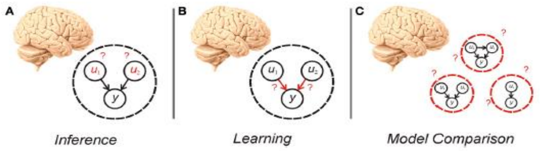
All models are wrong, but some are useful - for understanding how things can break

Failures in inference

- ‘Suboptimal’ priors induce pathological behaviour based on ‘optimal’ inference
- E.g., ‘suboptimal’ preferences, transition probabilities, observation model, …, that underlie inference

Failures in learning

- Computational basis of information gain about hidden states and model parameters
- E.g., inability to update after bad experiences, inability to optimise one’s model



Failures in model building

- State space underlying inference and learning

cf., FitzGerald, Dolan & Friston, 2014; Huys, Guitart-Masip, Dolan & Dayan, 2015; Dayan, 2014

A Markovian generative model

$P(\boldsymbol{A}) = Dir(a)$ $\quad P(\boldsymbol{D}) = Dir(d)$

$P(\boldsymbol{B}) = Dir(b)$ $\quad P(\gamma) = \Gamma(1, \beta)$ Full priors

$P(\pi|\gamma) = \sigma(-\gamma \cdot \boldsymbol{Q})$ – control states

$\boldsymbol{Q} = \sum_t -\mathbb{E}\,[\mathrm{H}[P(o_t|s_t)]] + H[Q(o_t|\pi)] + \mathbb{E}[lnP(o_t)]$

$P(\tilde{s}|\pi) = P(s_t|s_{t-1}, \pi) \ldots P(s_1|s_0, \pi)\ P(s_0)$

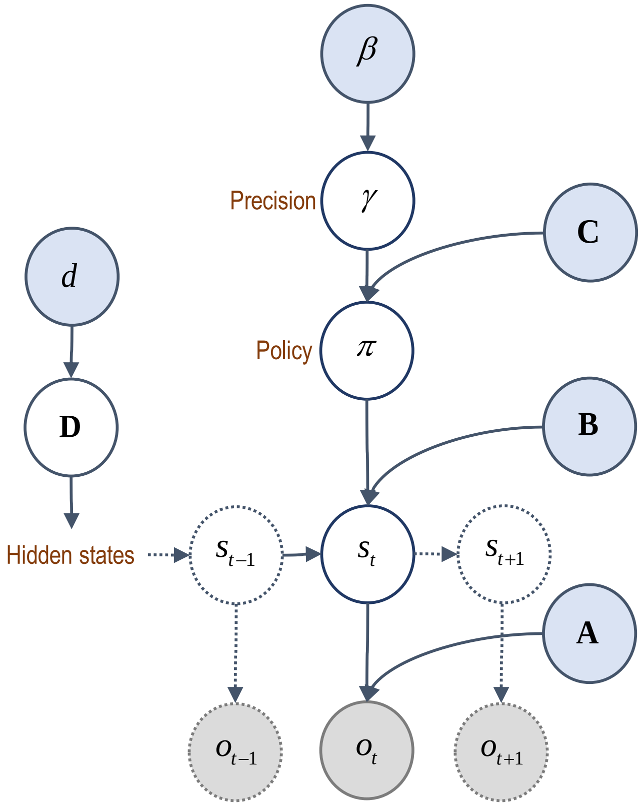
$P(s_{t+1}|s_t, \pi) =: \boldsymbol{B}(u = \pi(t))$

$P(s_0) =: \boldsymbol{D}$

$P(o_t) =: \sigma(\boldsymbol{C})$ Empirical priors – hidden states

$P(\tilde{o}|\tilde{s}) = P(o_0|s_0)P(o_1|s_1) \ldots P(o_t|s_t)$

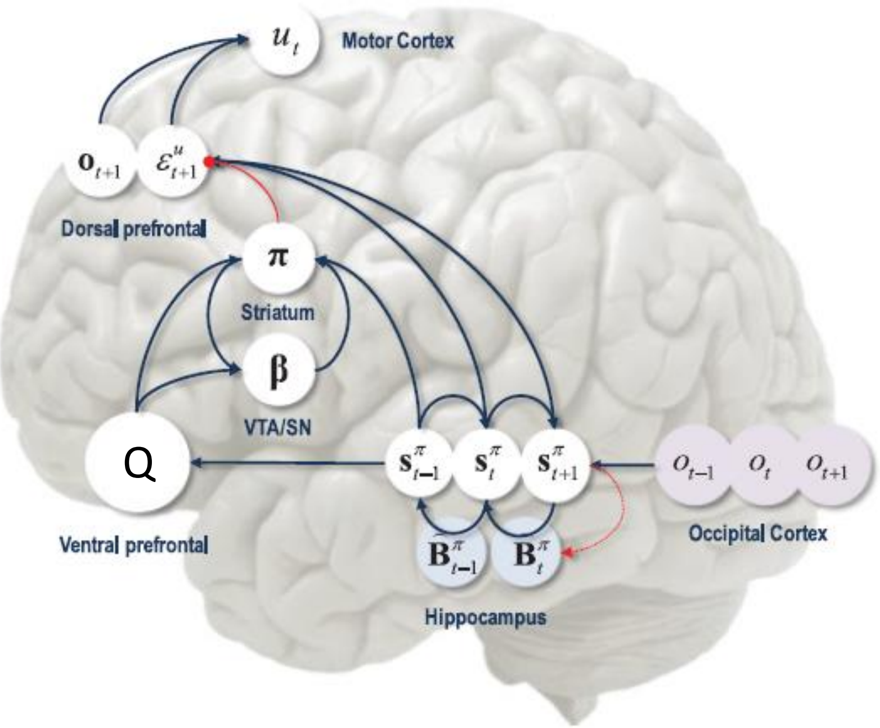
$P(o_t|s_t) =: \mathbf{A}$ Likelihood



Inference

Belief updating

Perception	$\hat{s}_t = \sigma(\log A \cdot o_t + \log B^{\pi}_{t-1} \cdot \hat{s}_{t-1})$	$\frac{\partial}{\partial_s} F$
Policy selection	$\hat{\pi} = \sigma(-\gamma \cdot Q)$	$\frac{\partial}{\partial_\pi} F$
Precision	$\hat{\beta} = \beta - \hat{\pi} \cdot Q$	$\frac{\partial}{\partial_\gamma} F$



Derive via variational free energy with respect to hidden states $x = \{s_t, \pi, \beta\}$:

$$Q(x) = \underset{Q(x)}{\arg\min}\, F(\tilde{o}, x, Q) \approx P(x|\tilde{o})$$

$$F(\tilde{o}, x, Q) = -E_{Q(\tilde{s}_t,\pi,\beta)}[\ln P(\tilde{o}, \tilde{s}_t, \pi, \beta | m)] - E_{Q(\tilde{s}_t,\pi,\beta)}[\ln Q(\tilde{s}_t, \pi, \beta)]$$

$$= \hat{s}_t \cdot (\ln \hat{s}_t - \ln A \cdot o_t - \ln B^{\pi}_{t-1} \cdot \hat{s}_{t-1}) + \cdots$$

cf., Friston et al., 2015, 2017; Bogacz, 2017

Inference: a closer (more conceptual) look

Belief updating

Perception

$$\hat{s}_t = \sigma(\log A \cdot o_t + \log B^{\pi}_{t-1} \cdot \hat{s}_{t-1})$$

"Likely **hidden states** given what I **observe**"

"Based on what I believe about the **previous state**"

Policy selection

$$\hat{\pi} = \sigma(-\gamma \cdot Q)$$

"**Confidence**"

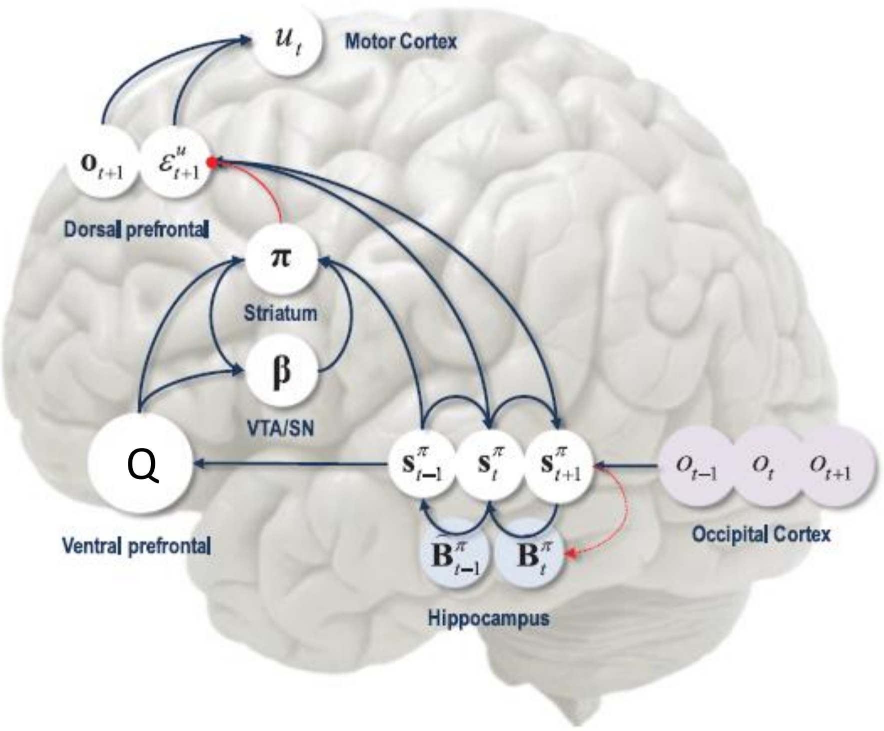
"**Value** of policies"

Precision

$$\hat{\beta} = \beta - \hat{\pi} \cdot Q$$

"**Prior** on precision"

"**Value** of **inferred** policy"



cf., Friston et al., 2015, 2017

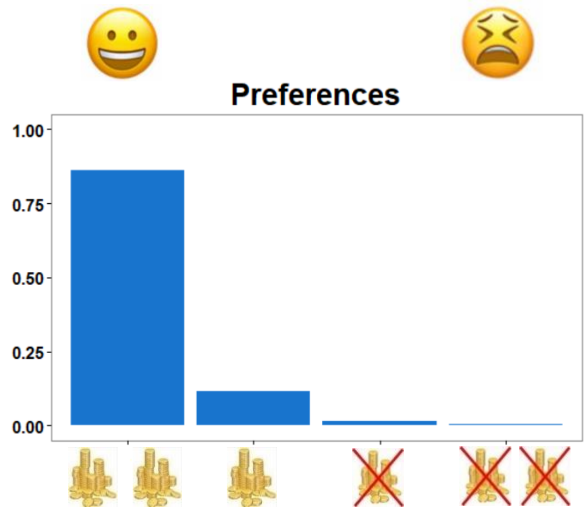
Inference: a closer look on policy selection

Policies become valuable if they

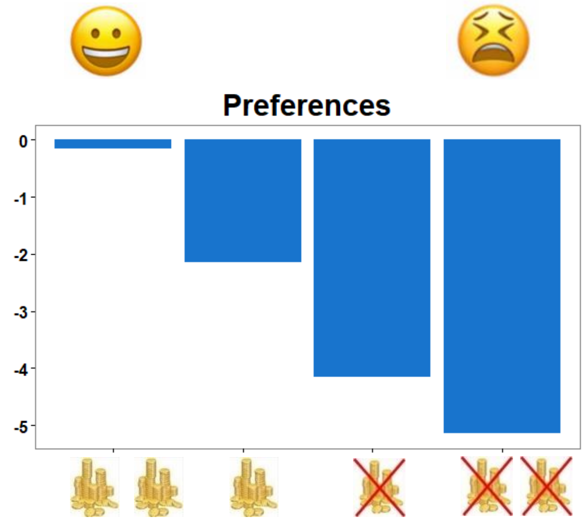
- Maximise reward/utility
- Keep options open
- Allow us to minimise uncertainty about the world

Utilities or preferences are defined as log-expectations over outcomes:

$P(o_t)$



$\longrightarrow \log P(o_t)$



Fulfilling these preferences minimises surprise $-\log P(o_t)$!

- This can be approximated with variational free energy!

Inference: a closer look on policy selection

Values of policies $\boldsymbol{Q(\pi)}$ defined as expected free energy:

$Q(\tilde{s}, \pi, A, B, D, \gamma) = Q(s_1|\pi) \ldots Q(s_T|\pi)\boldsymbol{Q(\pi)}Q(A)Q(B)Q(D)Q(\gamma)$

Approximate posterior

$$\boldsymbol{Q(\pi)} = \sum_t \mathbb{E}_Q[\ln P(o_\tau, s_\tau\,|\pi) - \ln Q(s_\tau|\pi)]$$
$$= \cdots$$
$$= \sum_t -\mathbb{E}_Q[\mathrm{H}[P(o_t|s_t)]] - D_{KL}[Q(o_t|\pi)||\,P(o_t)]$$

Friston et al., 2015 Appendix A

$$\boldsymbol{Q(\pi)} = \sum_t \underbrace{\underbrace{-\mathbb{E}\,[\mathrm{H}[P(o_t|s_t)]]}_{\text{Expected uncertainty } (H=0 \Leftrightarrow o_t = s_t)} + \underbrace{H[Q(o_t|\pi)]}_{\text{Entropy over outcomes}}}_{\text{Epistemic or intrinsic value}} + \underbrace{\underbrace{\mathbb{E}[lnP(o_t)]}_{\text{Expected utility}}}_{\text{Extrinsic value}}$$

Friston et al., 2015; 2017; Parr & Friston, 2017

Inference: a closer look on policy selection

Imagine you are a Tennis player and your opponent serves…

You want to find a position where…

Expected utility: $\mathbb{E}[lnP(o_t)]$

...there is a high chance of getting the ball, …

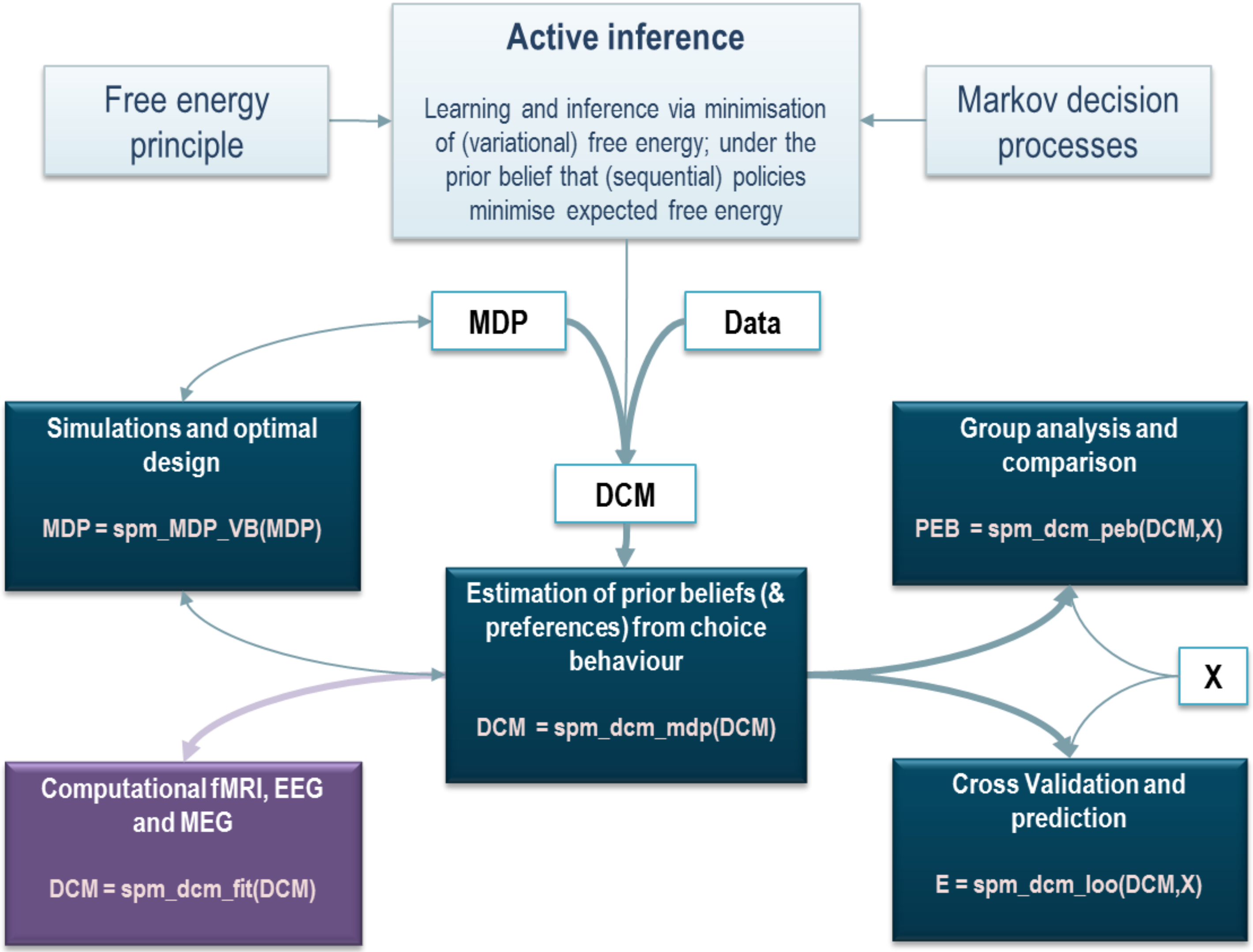
Entropy over outcomes: $H[Q(o_t|\pi)]$

... you have a good chance of getting a ball that was unexpected, …

...and where you can minimise uncertainty about the character of your opponent.

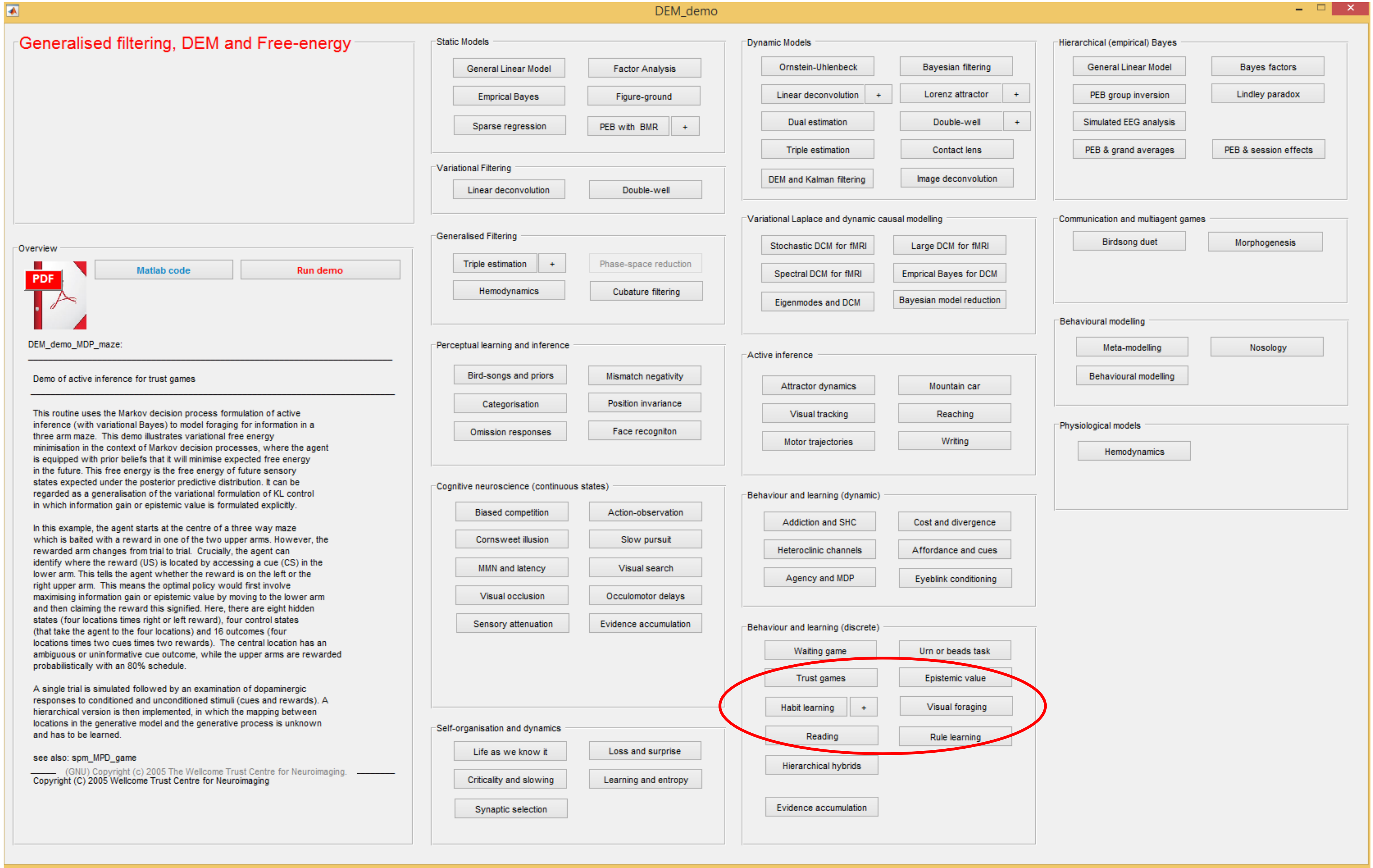
Ambiguity: $-\mathbb{E}[\mathrm{H}[P(o_t|s_t)]]$

Computational Phenotyping: Overview



Schwartenbeck & Friston, 2016 *eNeuro*

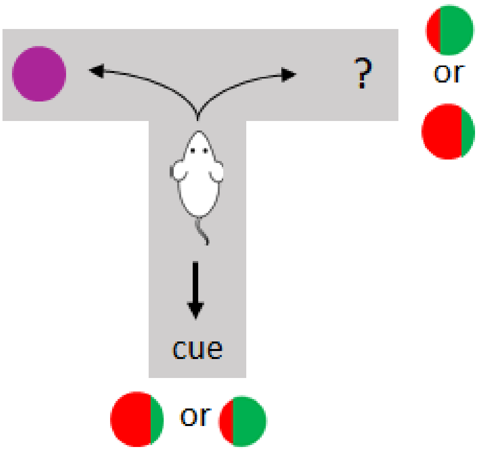
DEM Toolbox


DEM_demo
Generalised filtering, DEM and Free-energy
Overview
PDF
Matlab code
Run demo
DEM_demo_MDP_maze:
Demo of active inference for trust games
This routine uses the Markov decision process formulation of active inference (with variational Bayes) to model foraging for information in a three arm maze. This demo illustrates variational free energy minimisation in the context of Markov decision processes, where the agent is equipped with prior beliefs that it will minimise expected free energy in the future. This free energy is the free energy of future sensory states expected under the posterior predictive distribution. It can be regarded as a generalisation of the variational formulation of KL control in which information gain or epistemic value is formulated explicitly.
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 16 outcomes (four locations times two cues times two rewards). The central location has an ambiguous or uninformative cue outcome, while the upper arms are rewarded probabilistically with an 80% schedule.
A single trial is simulated followed by an examination of dopaminergic responses to conditioned and unconditioned stimuli (cues and rewards). A hierarchical version is then implemented, in which the mapping between locations in the generative model and the generative process is unknown and has to be learned.
see also: spm_MPD_game
(GNU) Copyright (c) 2005 The Wellcome Trust Centre for Neuroimaging.
Copyright (C) 2005 Wellcome Trust Centre for Neuroimaging
Static Models
General Linear Model
Factor Analysis
Emprical Bayes
Figure-ground
Sparse regression
PEB with BMR
+
Variational Filtering
Linear deconvolution
Double-well
Generalised Filtering
Triple estimation
+
Phase-space reduction
Hemodynamics
Cubature filtering
Perceptual learning and inference
Bird-songs and priors
Mismatch negativity
Categorisation
Position invariance
Omission responses
Face recogniton
Cognitive neuroscience (continuous states)
Biased competition
Action-observation
Cornsweet illusion
Slow pursuit
MMN and latency
Visual search
Visual occlusion
Occulomotor delays
Sensory attenuation
Evidence accumulation
Self-organisation and dynamics
Life as we know it
Loss and surprise
Criticality and slowing
Learning and entropy
Synaptic selection
Dynamic Models
Ornstein-Uhlenbeck
Bayesian filtering
Linear deconvolution
+
Lorenz attractor
+
Dual estimation
Double-well
+
Triple estimation
Contact lens
DEM and Kalman filtering
Image deconvolution
Variational Laplace and dynamic causal modelling
Stochastic DCM for fMRI
Large DCM for fMRI
Spectral DCM for fMRI
Emprical Bayes for DCM
Eigenmodes and DCM
Bayesian model reduction
Active inference
Attractor dynamics
Mountain car
Visual tracking
Reaching
Motor trajectories
Writing
Behaviour and learning (dynamic)
Addiction and SHC
Cost and divergence
Heteroclinic channels
Affordance and cues
Agency and MDP
Eyeblink conditioning
Behaviour and learning (discrete)
Waiting game
Urn or beads task
Trust games
Epistemic value
Habit learning
+
Visual foraging
Reading
Rule learning
Hierarchical hybrids
Evidence accumulation
Hierarchical (empirical) Bayes
General Linear Model
Bayes factors
PEB group inversion
Lindley paradox
Simulated EEG analysis
PEB & grand averages
PEB & session effects
Communication and multiagent games
Birdsong duet
Morphogenesis
Behavioural modelling
Meta-modelling
Nosology
Behavioural modelling
Physiological models
Hemodynamics

Part I: Maze task with ‘hidden state exploration’

- active *inference*

Example I



Two-step maze task

- Rat in a T-shaped maze
- Sample safe option (left) or risky option (right)
- At any trial, the risky option either has a high (75%) or low (25%) probability of containing a reward
- The rat starts in the middle of the maze and can decide to go left or right – or sample a cue at the bottom first
- The cue signifies the reward probability of this trial
- The left and right arm are *absorbing states* (the rat cannot sample both)

Thus, the rat needs to solve a trade-off between maximising reward and gaining information

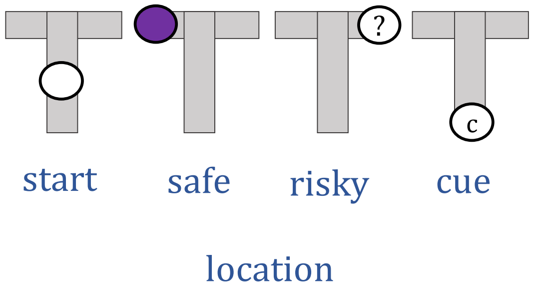
- In case of high uncertainty, it should sample the cue first

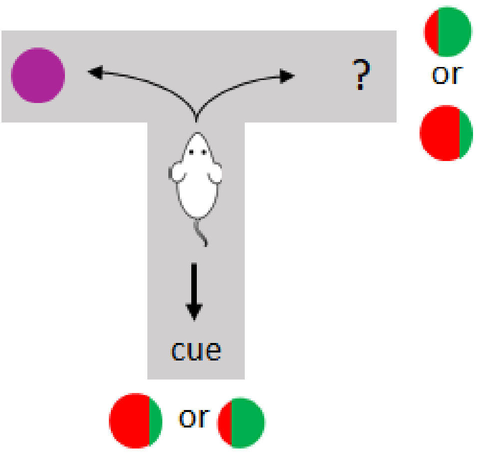
Now, define subjective generative model

- Start with available actions, (hidden) states and observations

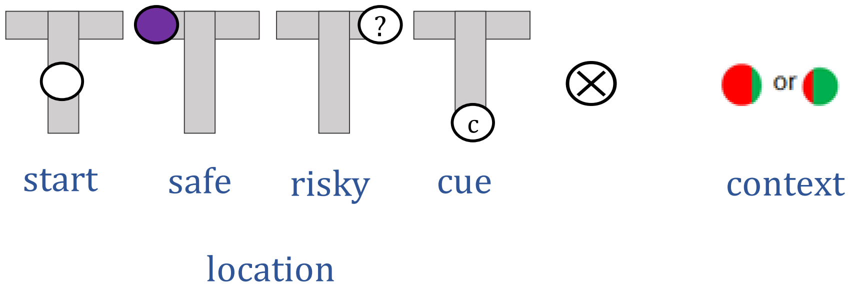
Subjective Generative Model

Action = moving to a Location

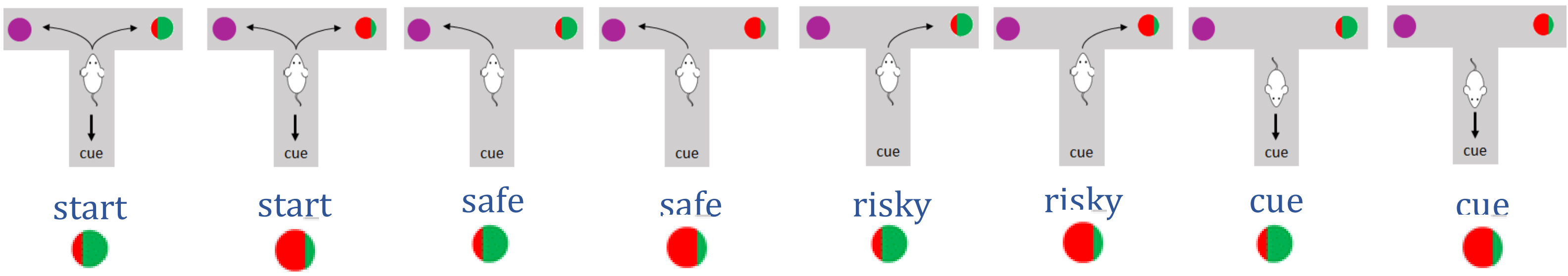




Hidden states

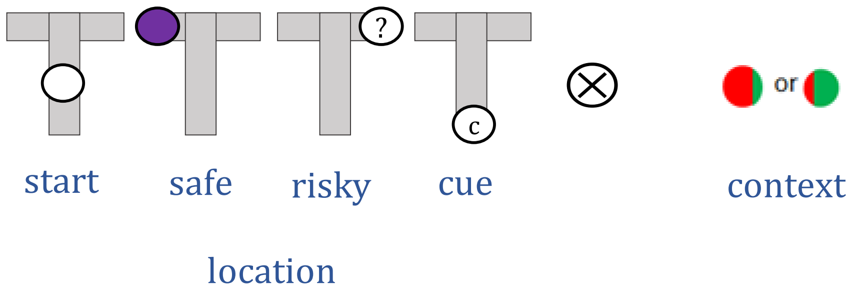


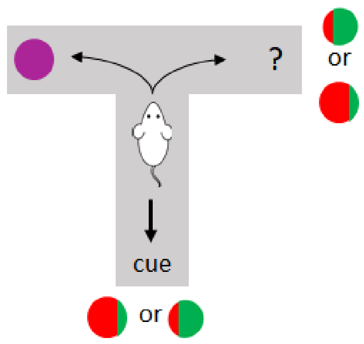
=> 8 hidden states:



Subjective Generative Model

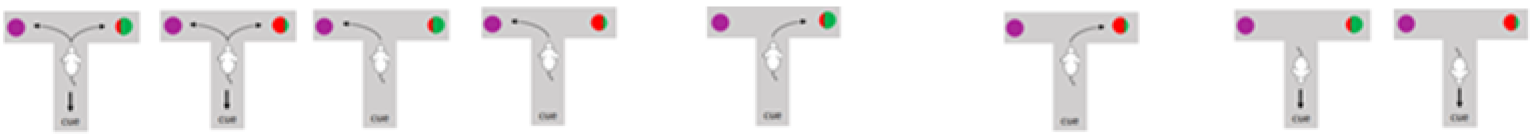
Hidden states



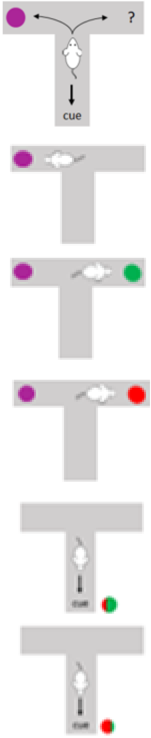


Outcomes

A – Mapping from hidden states to outcomes

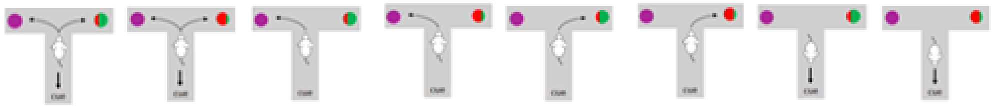


$$A = P(o_t|s_t) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.75 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.25 & 0.75 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

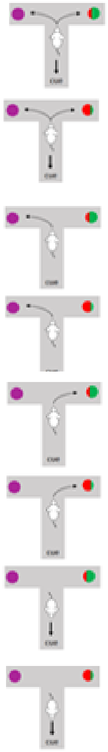


Implemented in Practical_I.m

B – Transition Probabilities



$$B(risky) = P(s_{t+1}|s_t, risky) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



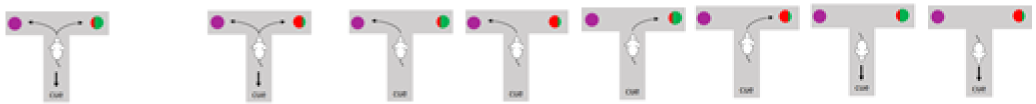
Implemented in Practical_I.m

c – Preferences over outcomes
d – Prior over initial state

Outcomes

$c = \ln P(o_t) = [0$ 🙂 😀 🙁 $0 \quad 0]$ $\quad \Rightarrow \ln P(o_t)$

States



$$d = \ln P(s_t) = [0.5 \quad 0.5 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$$

Implemented in Practical_I.m

Model structure

```matlab
%% 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 = .75;
b = 1 - a;

A{1}   = [1 1 0 0 0 0 0 0;    % ambiguous starting position (centre)
          0 0 1 1 0 0 0 0;    % safe arm selected and rewarded
          0 0 0 0 a b 0 0;    % risky arm selected and rewarded
          0 0 0 0 b a 0 0;    % risky arm selected and not rewarded
          0 0 0 0 0 0 1 0;    % informative cue - high reward prob
          0 0 0 0 0 0 0 1];   % informative cue - low reward prob
```

```matlab
% priors: (utility) C
%--------------------------------------------------------------------------
% Finally, we have to specify the prior preferences in terms of log
% probabilities. Here, the agent prefers rewarding outcomes
%--------------------------------------------------------------------------
cs = 2^1; % preference for safe option
cr = 2^2; % preference for risky option win

% preference for: [staying at starting point | safe | risky + reward | risky + no reward | cue context 1 | cue context 2]
C{1}  = [0 cs cr -cs 0 0]';

% now specify prior beliefs about initial state
%--------------------------------------------------------------------------
D{1}  = kron([1/4 0 0 0],[1 1])';

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

```matlab
% 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.
%--------------------------------------------------------------------------

% move to/stay in the middle
B{1}(:,:,1) = [1 0 0 0 0 0 1 0;
               0 1 0 0 0 0 0 1;
               0 0 1 0 0 0 0 0;
               0 0 0 1 0 0 0 0;
               0 0 0 0 1 0 0 0;
               0 0 0 0 0 1 0 0;
               0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0];

% move up left to safe  (and check for reward)
B{1}(:,:,2) = [0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0;
               1 0 1 0 0 0 1 0;
               0 1 0 1 0 0 0 1;
               0 0 0 0 1 0 0 0;
               0 0 0 0 0 1 0 0;
               0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0];
% move up right to risky (and check for reward)
B{1}(:,:,3) = [0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0;
               0 0 1 0 0 0 0 0;
               0 0 0 1 0 0 0 0;
               1 0 0 0 1 0 1 0;
               0 1 0 0 0 1 0 1;
               0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0];

% move down (check cue)
B{1}(:,:,4) = [0 0 0 0 0 0 0 0;
               0 0 0 0 0 0 0 0;
               0 0 1 0 0 0 0 0;
               0 0 0 1 0 0 0 0;
               0 0 0 0 1 0 0 0;
               0 0 0 0 0 1 0 0;
               1 0 0 0 0 0 1 0;
               0 1 0 0 0 0 0 1];
```

Implemented in Practical_I.m

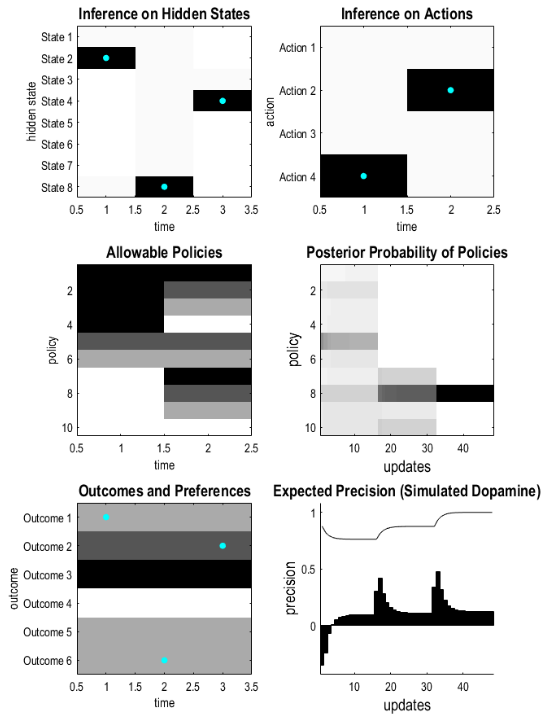
Model structure

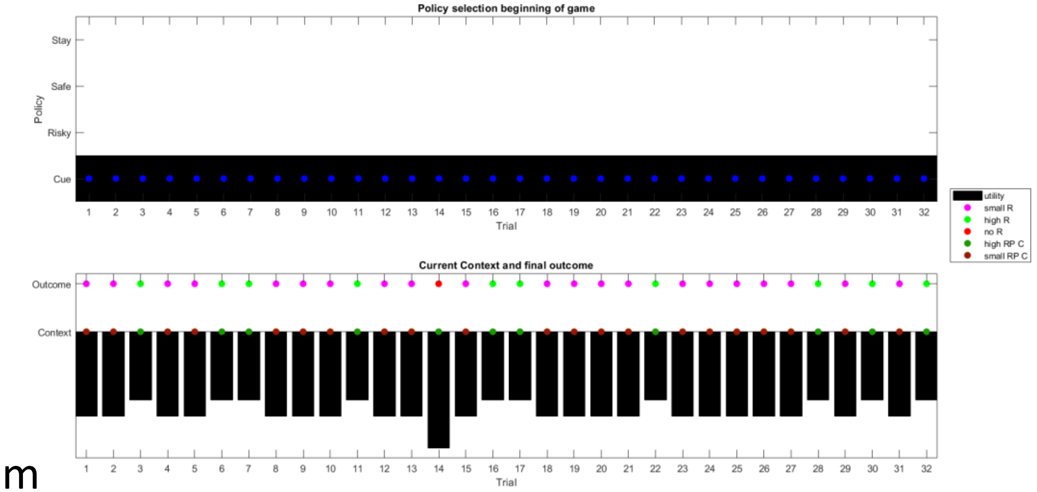
```
%% MDP Structure - this will be used to generate arrays for multiple trials
%==========================================================================
mdp.V = V;                      % allowable policies
mdp.A = A;                      % observation model
mdp.B = B;                      % transition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

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

mdp = spm_MDP_check(mdp);

% true parameters
%--------------------------------------------------------------------------
n          = 32;                % number of trials
i          = rand(1,n) > 1/2;   % randomise hidden states over trials

MDP        = mdp;

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

Implemented in Practical_I.m

Apply the model and generate data

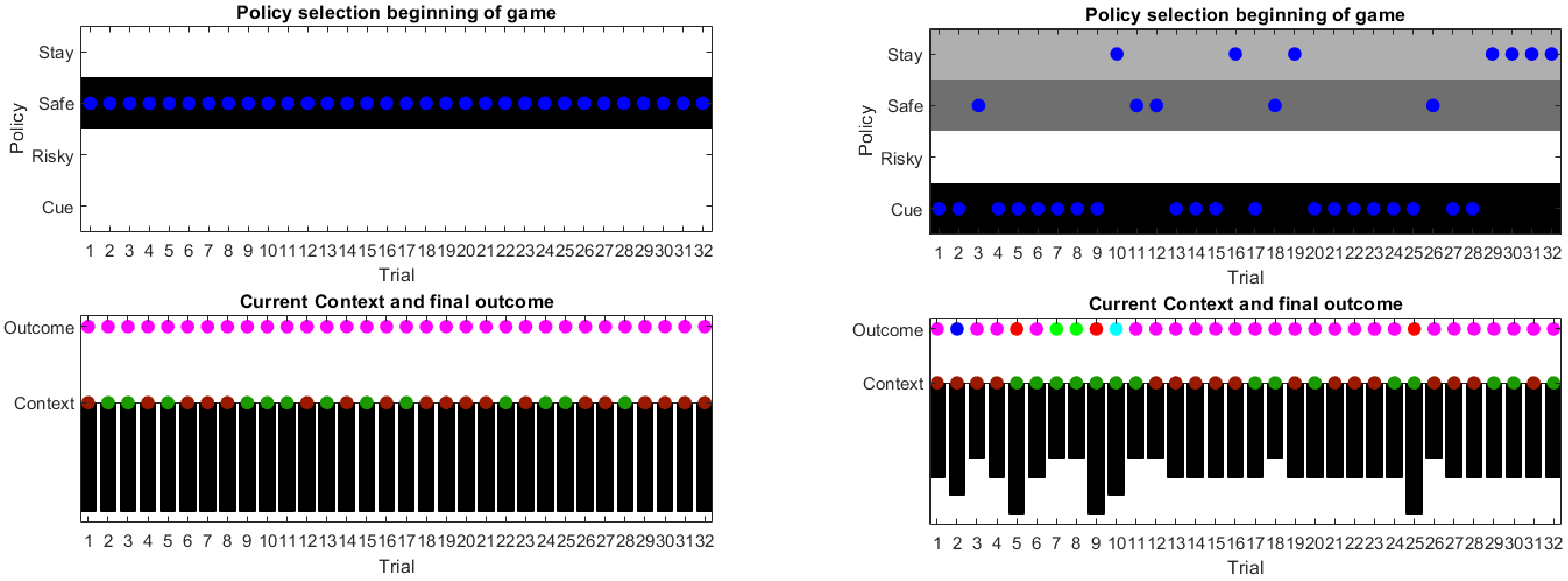
```
%==========================================================================
%==========================================================================
% 3.1 Fairly precise behaviour WITH information-gain:
%==========================================================================
%==========================================================================

% number of simulated trials
%--------------------------------------------------------------------------
n         = 32;                % number of trials
i         = rand(1,n) > 1/2;   % randomise hidden states over trials

MDP       = mdp;

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

[MDP(1:n).beta]  = deal(1);             % inverse precision of policy selection
[MDP(1:n).alpha] = deal(16);            % precision of action selection

MDP  = Z_spm_MDP_VB_X(MDP);

% illustrate behavioural responses - single trial
%--------------------------------------------------------------------------
spm_figure('GetWin','Figure 1a'); clf
Z_spm_MDP_VB_trial(MDP(1));

% illustrate behavioural responses over trials
%--------------------------------------------------------------------------
Z_spm_MDP_VB_game_EpistemicLearning_state(MDP);
```





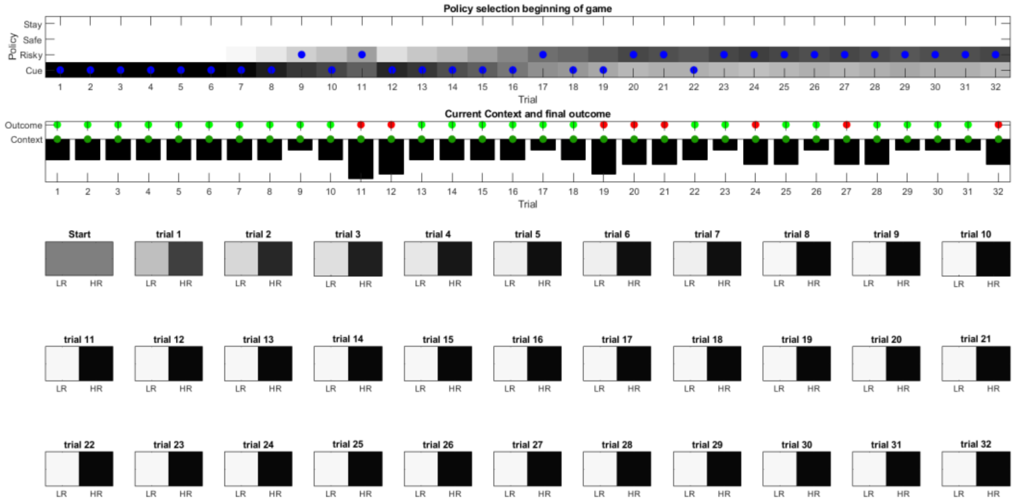
Implemented in Practical_I.m

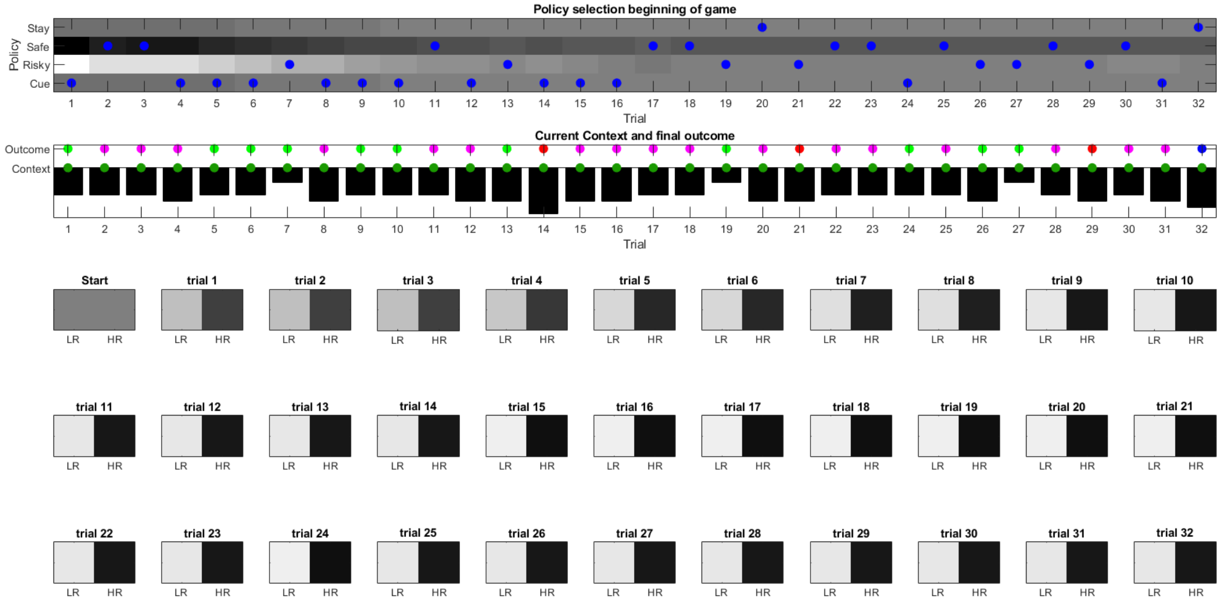
Apply the model and generate data

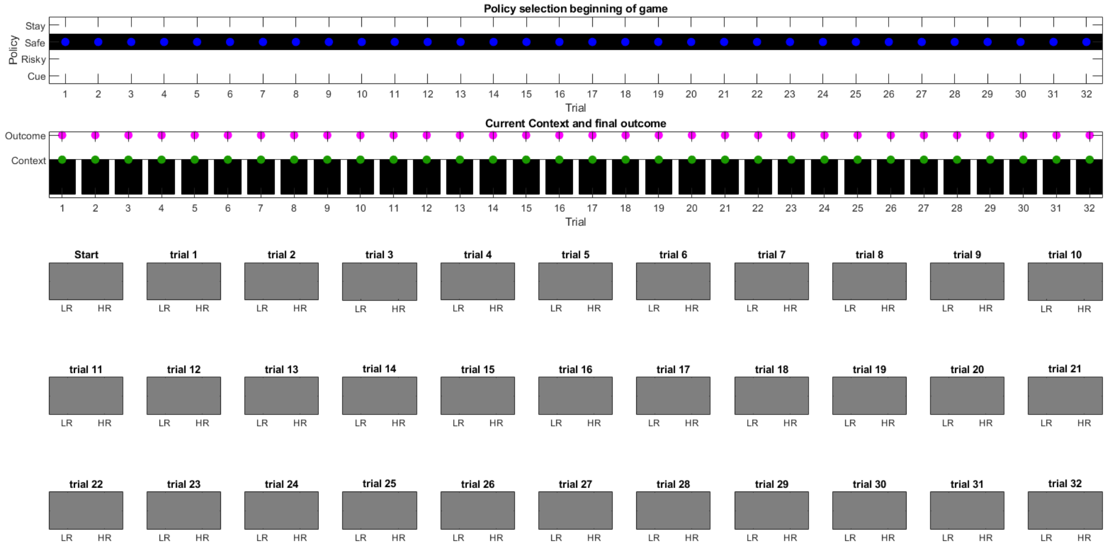


Implemented in Practical_I.m

Apply the model and generate data

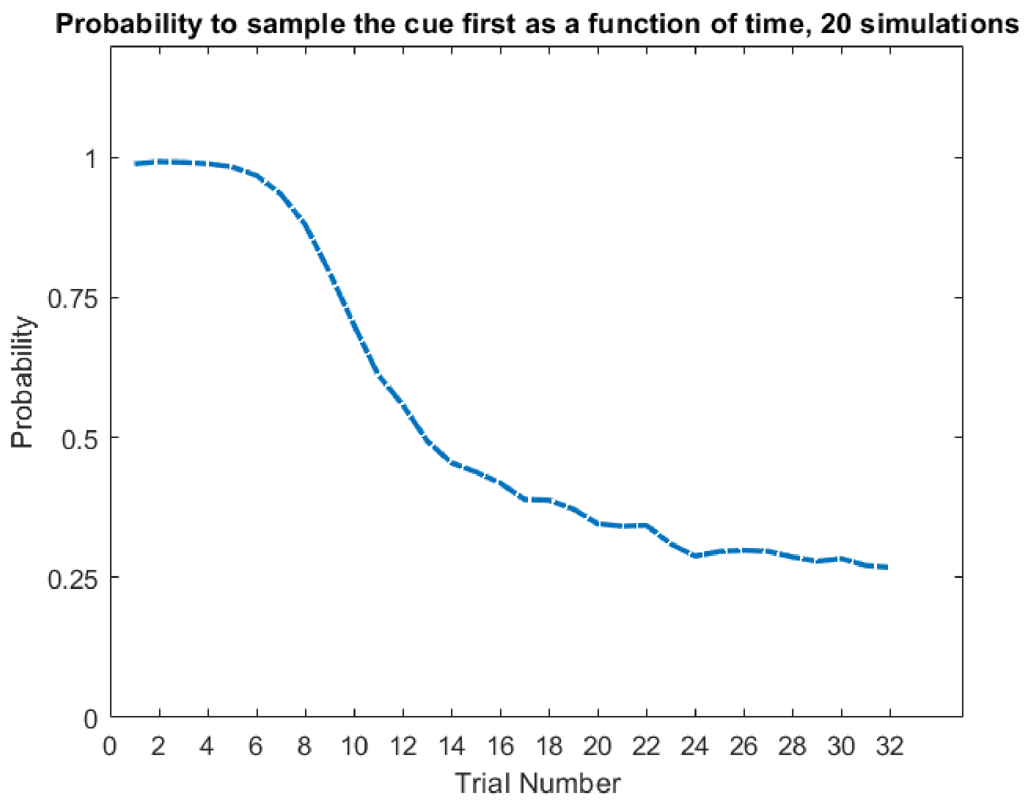


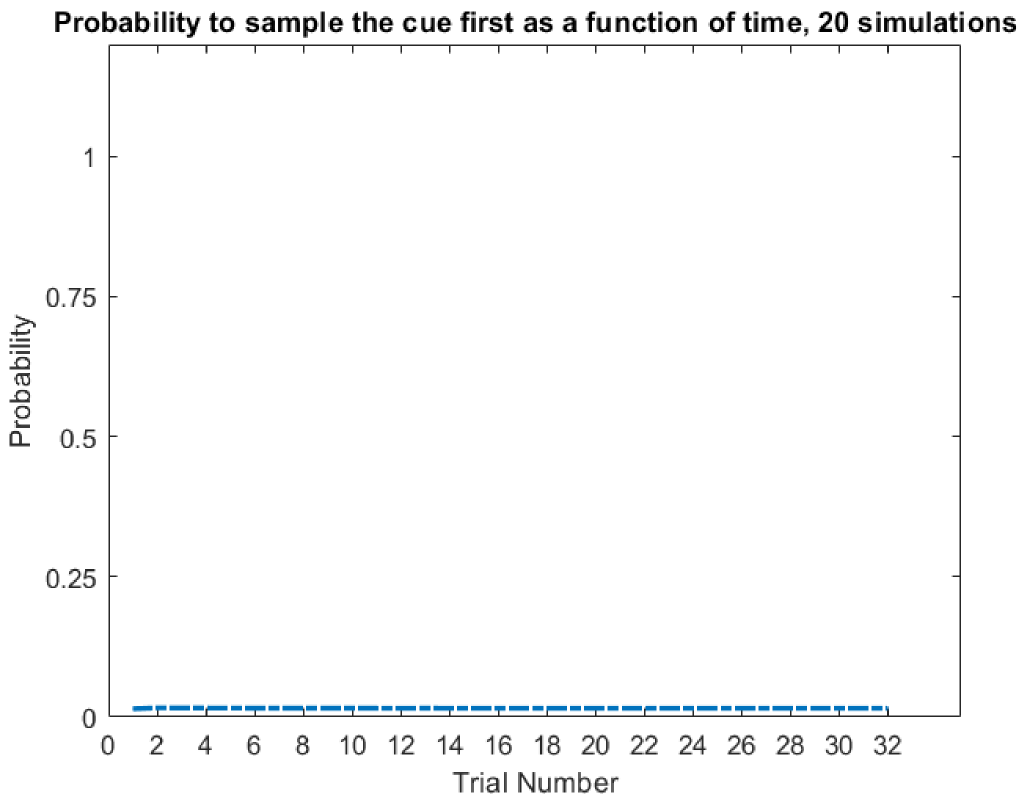




Implemented in Practical_I.m

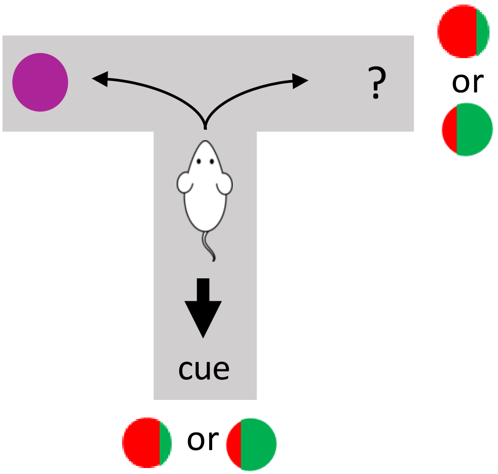
Apply the model and generate data

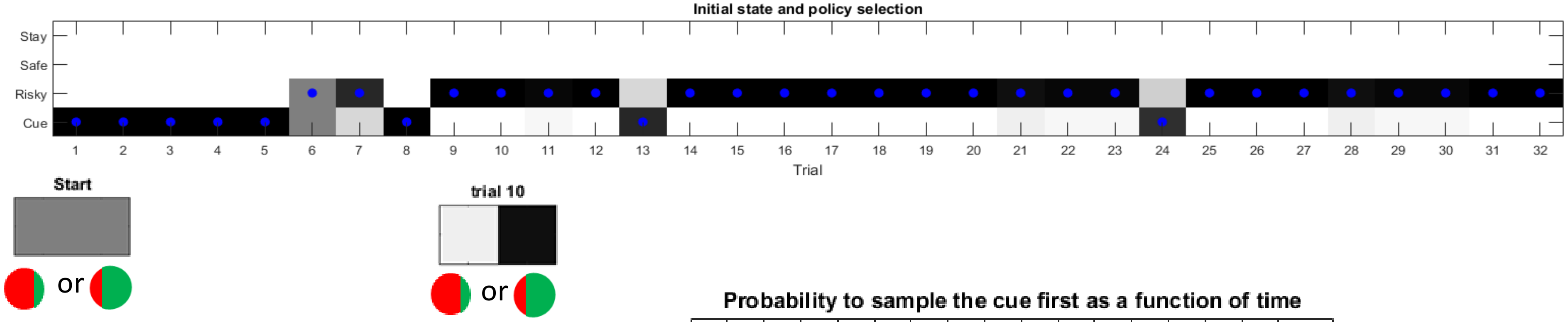




Implemented in Practical_I.m

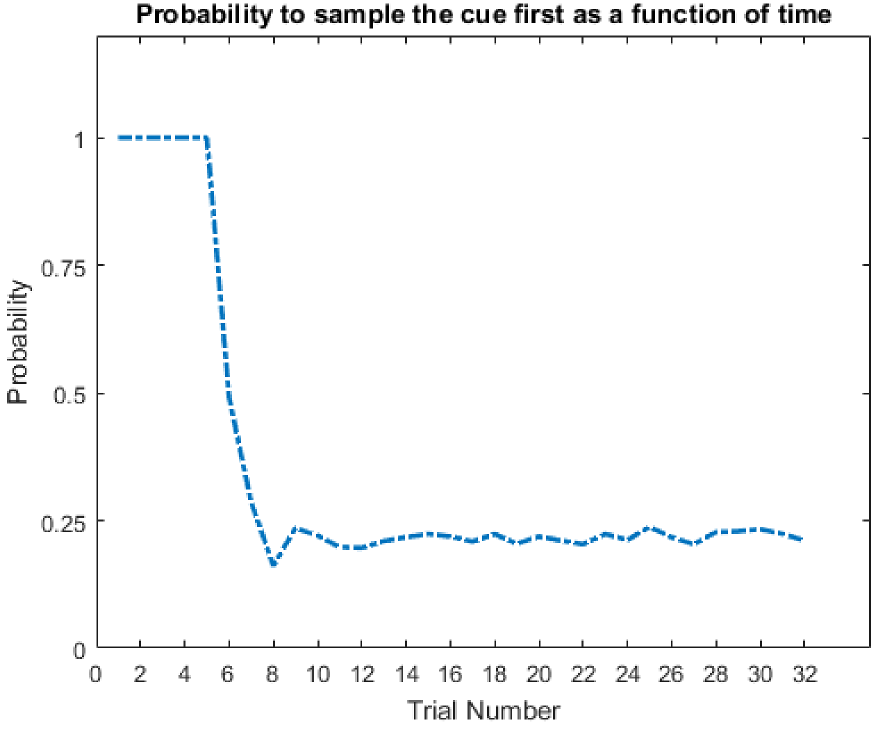
Summary 'hidden sate exploration'





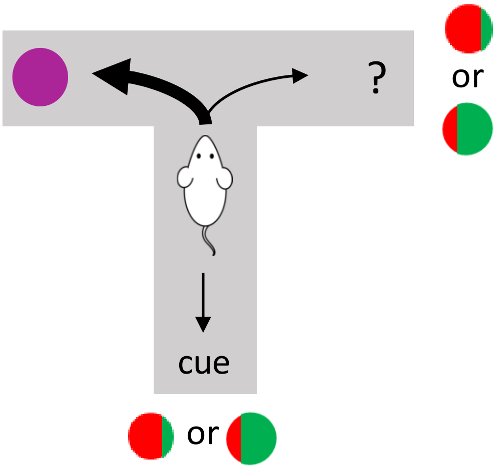
$$\boldsymbol{Q}(\boldsymbol{\pi}) = \underbrace{-\mathbb{E}_Q[\mathrm{H}[P(o_t|s_t)]]}_{\text{Minimising uncertainty}} - \underbrace{D_{KL}[Q(o_t|\pi)||P(o_t)]}_{\text{Realising preferences}}$$

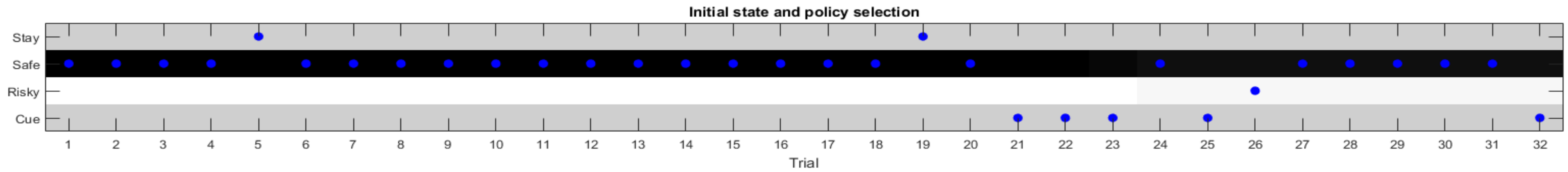
Exploring hidden states

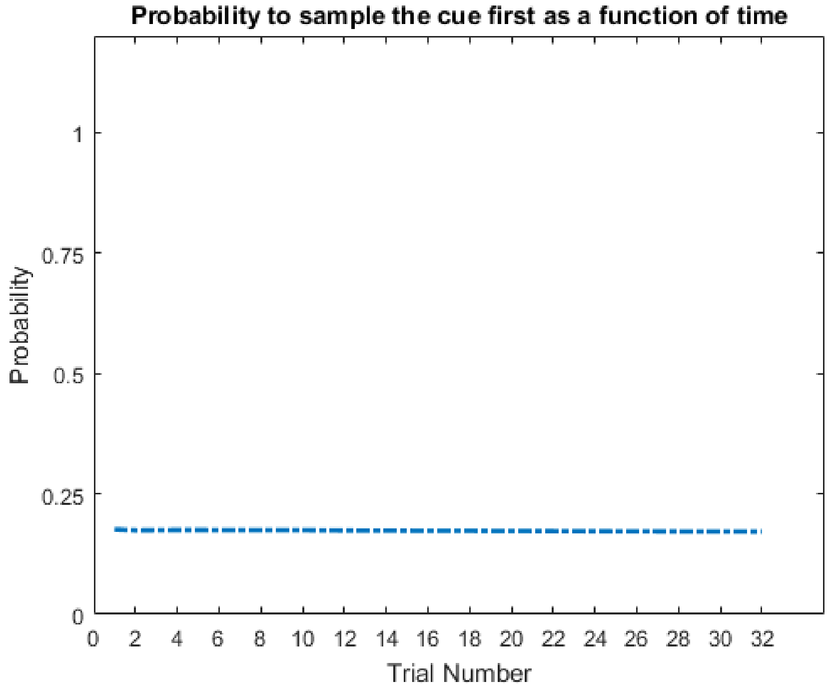


Schwartenbeck et al., 2018, bioRxiv

Summary **broken** ‘hidden sate exploration’







$$\boldsymbol{Q}(\boldsymbol{\pi}) = -\underbrace{\mathbb{E}_Q[\mathrm{H}[\cancel{\;}_t|s_t)]]}_{\text{Minimising uncertainty}} - \underbrace{D_{KL}[Q(o_t|\pi)||P(o_t)]}_{\text{Realising preferences}}$$

↑

$\boxed{\text{Exploring } \cancel{\;}\text{len states}}$

Schwartenbeck et al., 2018, bioRxiv

Tasks

1) What happens if the cue now provides random information – how could you simulate that?
(Random or stable context)

2) Can you think of (and simulate) prior preferences that induce pathologic behaviour? E.g., what happens if you induce ‘flat’ preferences?
(Random or stable context)

3) How does behaviour depend on the prior over initial states (‘D’)? What happens if you give the agent the right/wrong prior (use extreme values)?
(Random or stable context)

4) What happens if you change the learning rate (eta) in this task?
(Stable context)

5) Can you simulate a reversal learning task with a change of context after 16 trials?
(*‘Stable’ context)*

6) What is the role of precision (alpha)? Are there situations where imprecise (i.e. random) behaviour can be beneficial?
(Random or stable context)

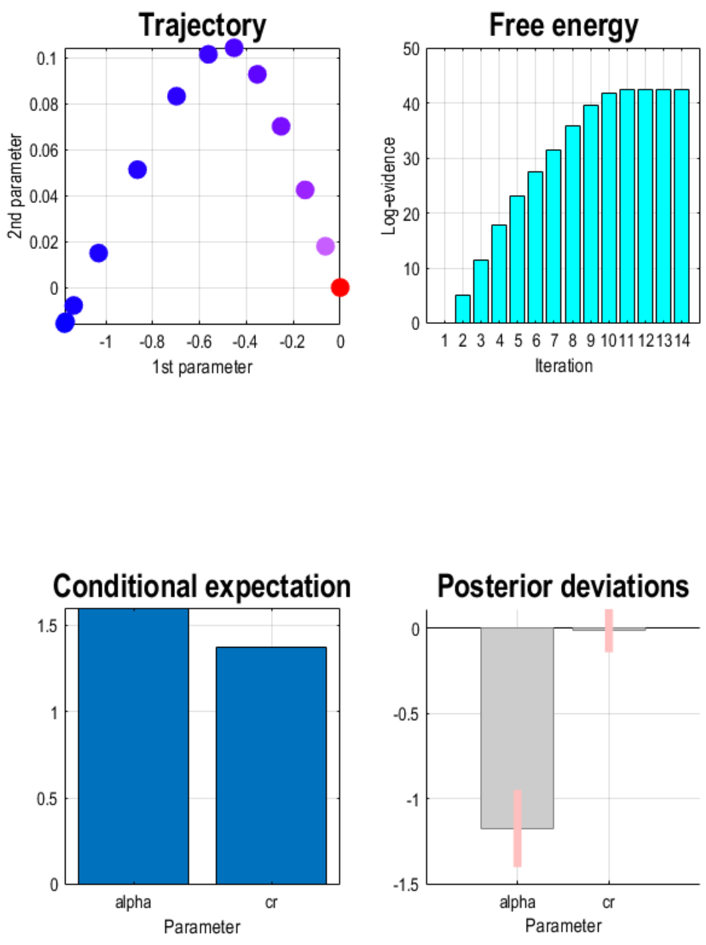
7) [How critical is the Markov property here? Can you think of an easy way to relax the Markov property, e.g. by also taking the previous time step into account?]

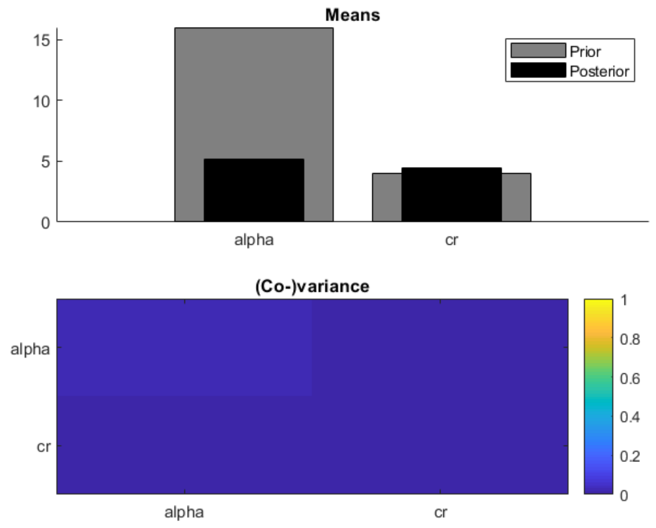
Part II: Model Inversion

- based on task of part I

IV Model inversion and computational phenotyping

```
%========================================================================
%========================================================================
% 6.2 Invert model and try to recover original parameters:
%========================================================================
%========================================================================

DCM.MDP   = mdp;                    % MDP model
DCM.field = {'alpha','cr'};         % parameter (field) names to optimise
DCM.U     = {MDP.o};                % trial specification (stimuli)
DCM.Y     = {MDP.u};                % responses (action)

DCM       = Z_spm_dcm_mdp(DCM);

subplot(2,2,3)
xticklabels(DCM.field),xlabel('Parameter')
subplot(2,2,4)
xticklabels(DCM.field),xlabel('Parameter')
```





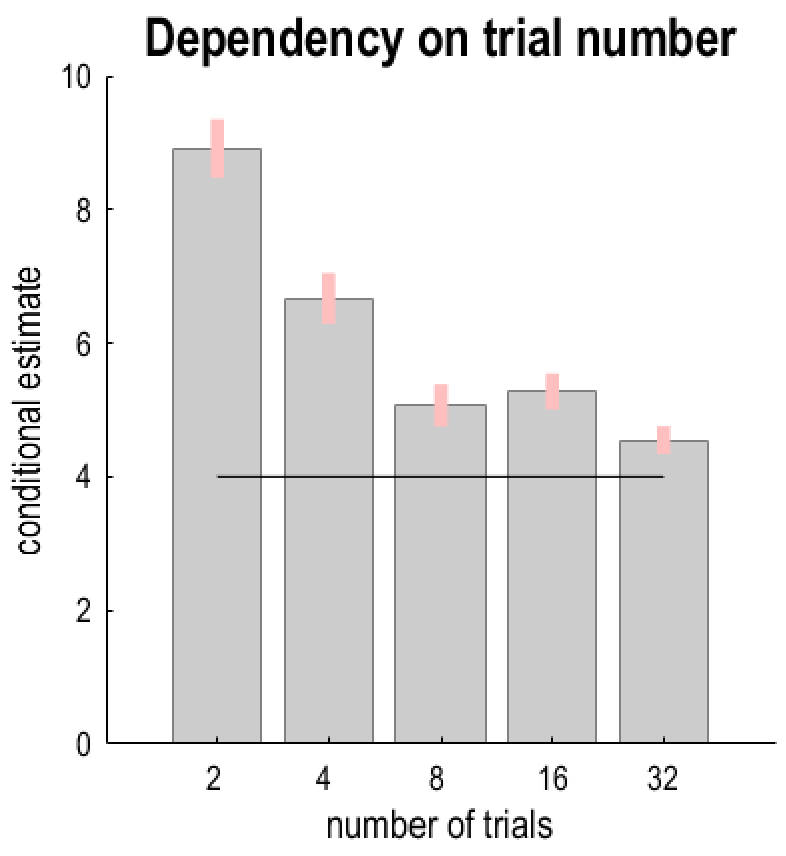
Model inversion (i.e. parameter estimation) based on variational Bayes

Central idea: maximise (negative) free energy, which reflects log-likelihood of data under parameters (i.e. accuracy) minus shift from prior to posterior over parameters (i.e., complexity)

Implemented in Practical_II.m

IV Model inversion and computational phenotyping

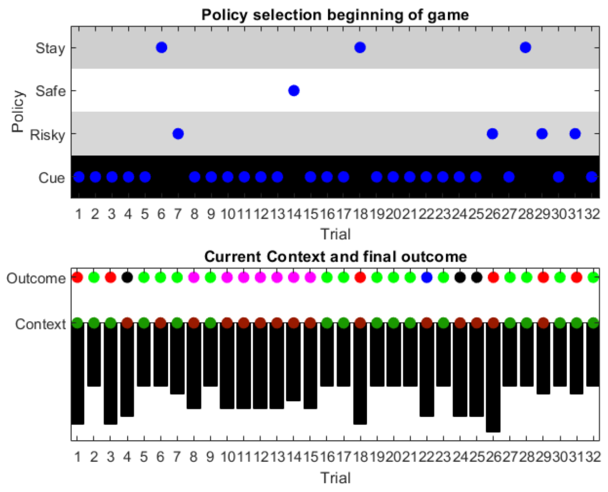
```
%% 7. Now repeat using subsets of trials to illustrate effects on estimators - design optimisation!
DCM.MDP   = mdp;                    % MDP model
DCM.field = {'alpha'};

n         = [2 4 8 16 32];

for i = 1:length(n)

    DCM.U = {MDP(1:n(i)).o};
    DCM.Y = {MDP(1:n(i)).u};
    DCM   = Z_spm_dcm_mdp(DCM);

    Ep(i,1) = DCM.Ep.alpha;
    Cp(i,1) = DCM.Cp;

    fprintf('### Simulated parameter recovery with %d trials ###\n',n(i))

end

% plot results
%--------------------------------------------------------------------------
spm_figure('GetWin','Figure 3'); clf
subplot(2,1,1), spm_plot_ci(exp(Ep(:)),Cp(:)), hold on
plot(1:length(n),(n - n) + MDP(1).alpha,'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
```

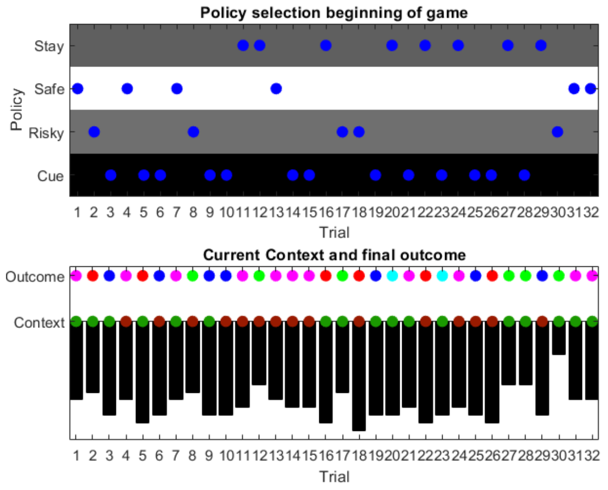


Implemented in Practical_II.m

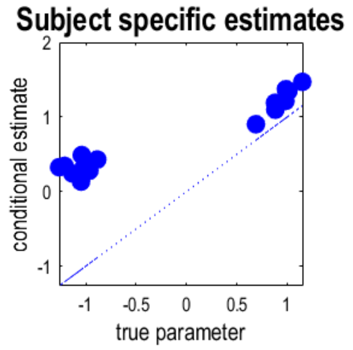
IV Model inversion and computational phenotyping

```matlab
%% 8. Now repeat but over multiple subjects with different alpha

% 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     = 32;                             % number of trials
i     = rand(1,n) > 1/2;                % randomise hidden states

clear MDP

[MDP(1:n)]       = deal(mdp);
[MDP(i).s]       = deal(2);
% [MDP(1:n).alpha] = deal(16);

reward = zeros(n,size(X,1));
for i = 1:size(X,1)

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

    % solve to generate data
    %----------------------------------------------------------------------
    DDP        = Z_spm_MDP_VB_X(MDP);      % realisation for this subject

    DCM.field  = {'alpha'};
    DCM.U      = {DDP.o};               % trial specification (stimuli)
    DCM.Y      = {DDP.u};               % responses (action)
    GCM{i,1}   = DCM;

    for kk=1:length(DCM.U)
        if DCM.U{kk}(end)==2 || DCM.U{kk}(end)==4 % outcome 2 or 4 == reward
            reward(kk,i)=1;
        end
    end

    % plot behavioural responses
    %----------------------------------------------------------------------
    Z_spm_MDP_VB_game_EpistemicLearning_state(DDP);drawnow

    fprintf('### Simulated data for subject %d of %d ###\n',i,size(X,1))

end
```





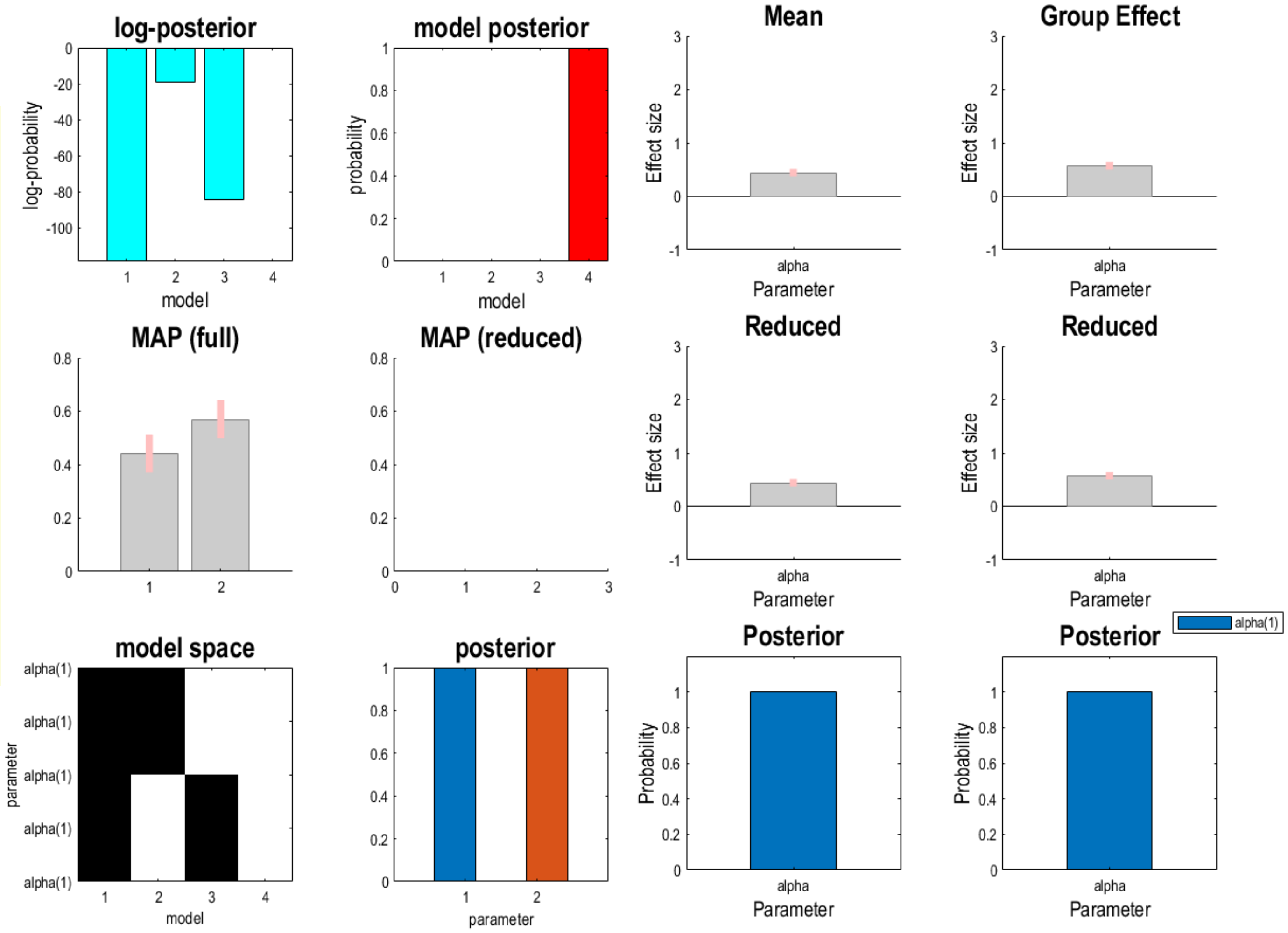

```matlab
%% Bayesian model inversion
%==========================================================================
GCM   = Z_spm_dcm_fit(GCM);

% 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.alpha;
end

plot(alpha,alpha,':b',alpha,qP,'.b','MarkerSize',32)
% 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
```



Implemented in Practical_II.m

IV Model inversion and computational phenotyping

```matlab
%% 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(3,2,1), set(gca,'XTickLabel',DCM.field), title('Mean'), ylim([-1,3])
subplot(3,2,2), set(gca,'XTickLabel',DCM.field), title('Group Effect'),ylim([-1,3])
subplot(3,2,3), set(gca,'XTickLabel',DCM.field), ylim([-1,3])
subplot(3,2,4), set(gca,'XTickLabel',DCM.field), ylim([-1,3])
subplot(3,2,5), set(gca,'XTickLabel',DCM.field), ylim([0,1.2])
subplot(3,2,6), set(gca,'XTickLabel',DCM.field), ylim([0,1.2])
```

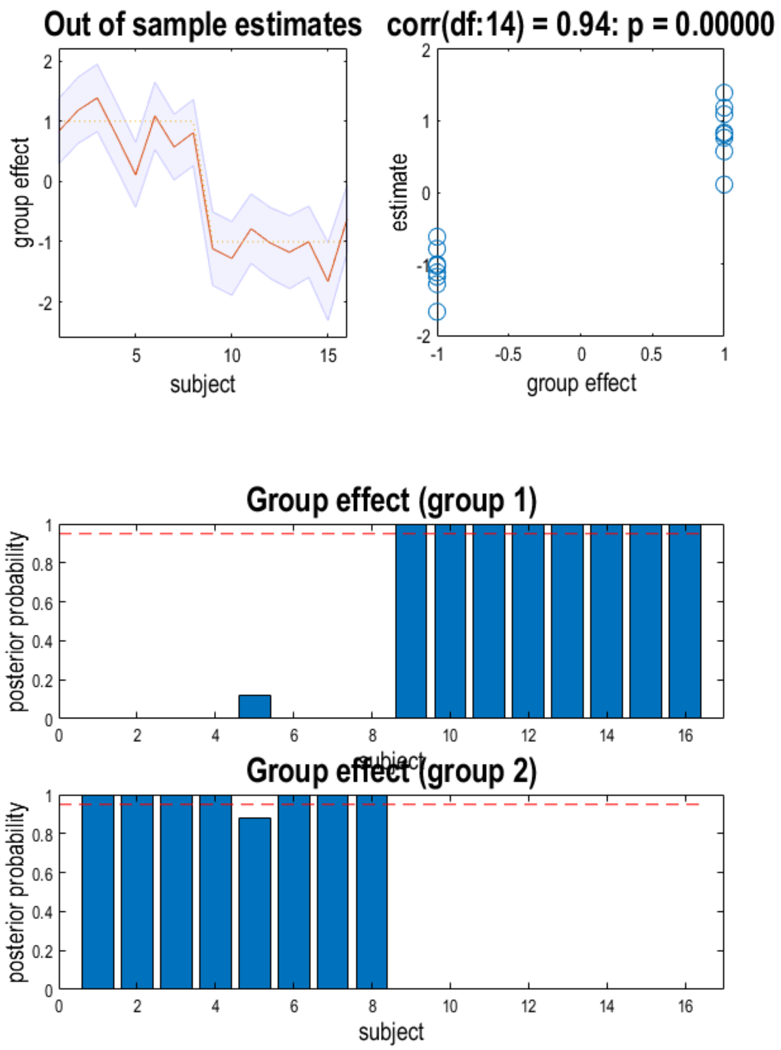


Implemented in Practical_II.m

Friston, Litvak, Oswal, Razi, Stephan, van Wijk, Ziegler, & Zeidman, 2016

IV Model inversion and computational phenotyping

```
%% posterior predictive density and cross validation
%==========================================================================
spm_dcm_loo(GCM,M,'alpha');
```



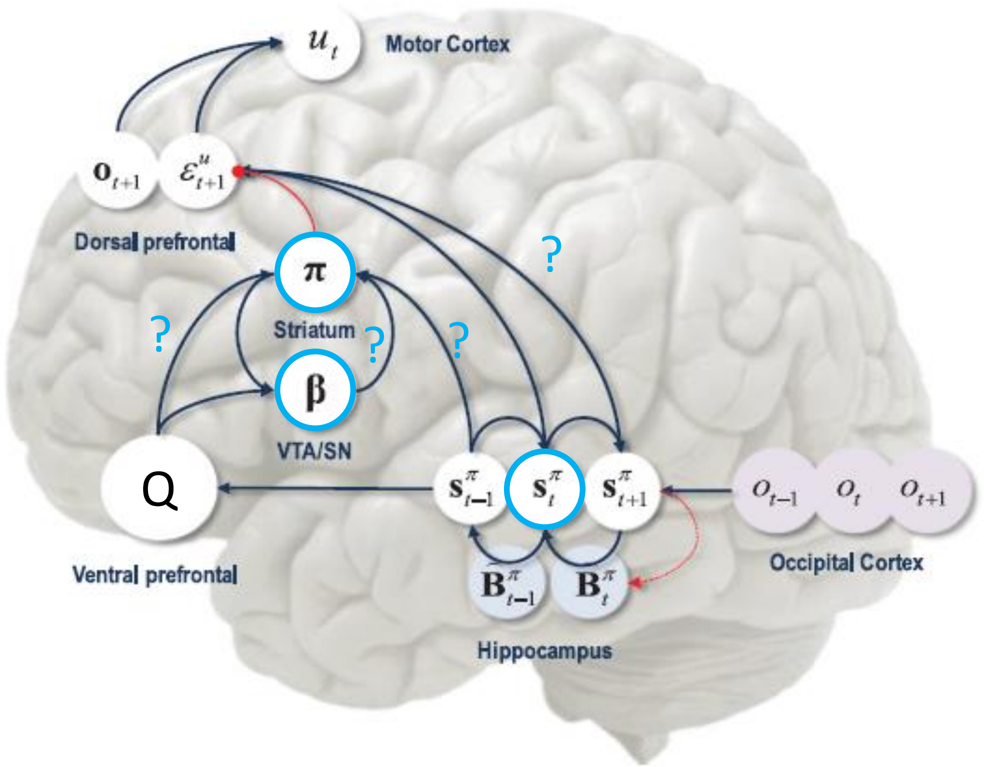
Implemented in Practical_II.m

Part III: Maze task with ‘model parameter exploration’

- active *learning*

Active inference and *active learning*

We use generative models to perform inference on hidden states.



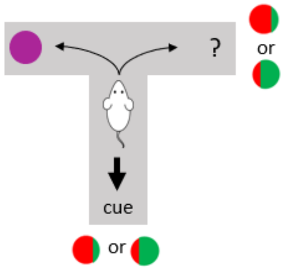
But we also learn and update our models.

Active inference and *active learning*

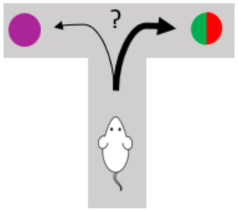
Goal-directed exploration can reflect active inference or active learning

$$
\begin{aligned}
Q(\pi) &= \mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}[\ln Q(s_\tau, A|\pi) - \ln P(o_\tau, s_\tau, A|\pi)] \\
&= \cdots \\
&= \underbrace{-\mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}\big[H[P(o_\tau|s_\tau)]\big]}_{\text{Minimising uncertainty}} - \underbrace{D_{KL}[Q(o_\tau|\pi)||P(o_\tau)]}_{\text{Obtaining preferred outcomes}} + \underbrace{\mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}[\ln Q(A) - \ln Q(A|s_\tau, o_\tau, \pi)]}_{\text{Expected model-update}}
\end{aligned}
$$

Exploring hidden states

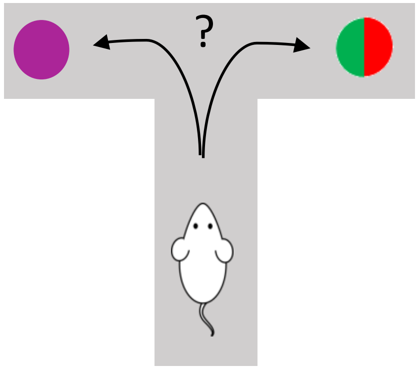


Exploring model parameters



Schwartenbeck et al., 2018, bioRxiv

Example II



Two-step maze task

- Rat in a T-shaped maze
- Sample safe option (left) or risky option (right)
- Probability of obtaining a reward is stable but unknown
- The rat starts in the middle of the maze and can decide to go left or right
- The left and right arm are *absorbing states* (the rat cannot sample both)

Thus, the rat needs to solve a trade-off between maximising reward and gaining information

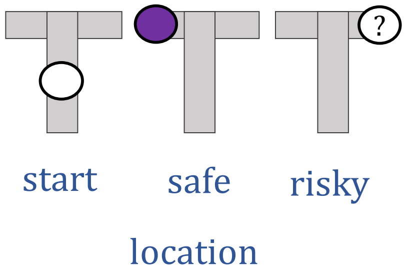
- In the beginning of the experiment, it should sample the risky option to learn about the reward statistics

Now, define subjective generative model

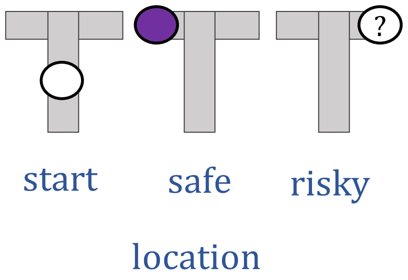
- Start with available actions, (hidden) states and observations

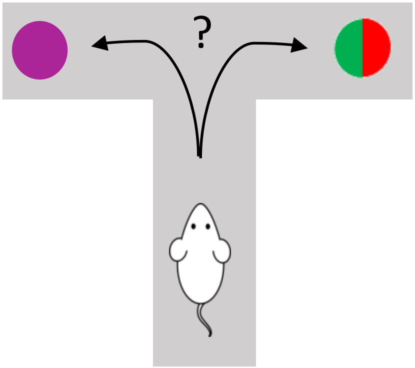
Subjective Generative Model

Action = moving to a Location

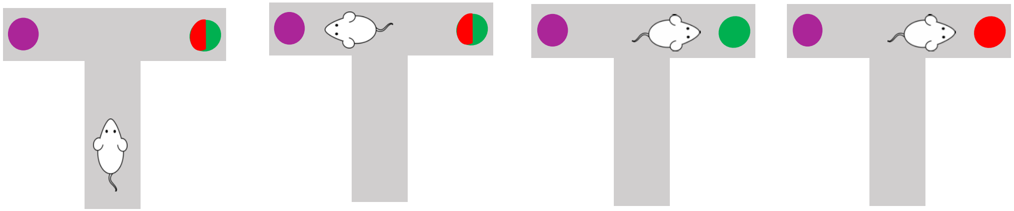


(Hidden) states

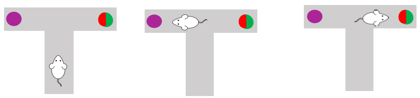




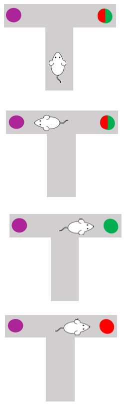
Outcomes



A – Mapping from hidden states to outcomes

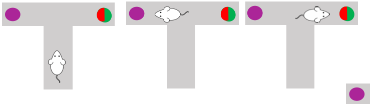


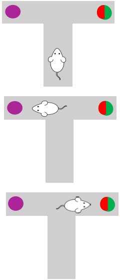
$$A = P(o_t | s_t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \\ 0 & 0 & 0.5 \end{bmatrix}$$



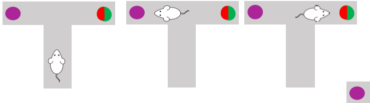
Implemented in Practical_III.m

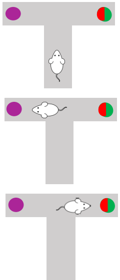
B – Transition Probabilities





$$B(safe) = P(s_{t+1}|s_t, safe) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$





$$B(risky) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

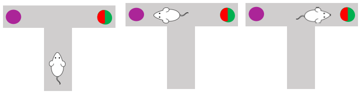
Implemented in Practical_III.m

c – Preferences over outcomes
d – Prior over initial state

Outcomes

$c = \ln P(o_t) = [0$ 🙂 😀 🙁 $] \quad \Rightarrow \ln P(c)$

States



$$d = \ln P(s_t) = [1 \quad 0 \quad 0]$$

Implemented in Practical_III.m

```
%% 1. Set up model structure

rng('shuffle')

%==========================================================================
%==========================================================================
% 1.1 Outcome probabilities: A
%==========================================================================
%==========================================================================

a = 0.5;
b = 1 - a;

% Location and Reward, exteroceptive - no uncertainty about location, interooceptive - uncertainty about reward prob
%--------------------------------------------------------------------------
% That's were true reward prob comes in
A{1} = [1 0 0          % reward neutral  (starting position)
        0 1 0          % low reward      (safe option)
        0 0 a          % high reward     (risky option)
        0 0 b];        % negative reward (risky option)

clear('a'),clear('b')
%==========================================================================
%==========================================================================
% 1.2 Beliefs about outcome (likelihood) mapping
%==========================================================================
%==========================================================================

%--------------------------------------------------------------------------
% That's where learning comes in - start with uniform prior
%--------------------------------------------------------------------------

a{1} = [1 0 0          % reward neutral  (starting position)
        0 1 0          % low reward      (safe option)
        0 0 1/4        % high reward     (risky option)
        0 0 1/4];      % negative reward (risky option)
%==========================================================================
%==========================================================================
% 1.3 Controlled transitions: B{u}
%==========================================================================
%==========================================================================

%--------------------------------------------------------------------------
% Next, we have to specify the probabilistic transitions of hidden states
% for each factor. Here, there are three actions taking the agent directly
% to each of the three locations.
%--------------------------------------------------------------------------
B{1}(:,:,1) = [1 1 1; 0 0 0;0 0 0];     % move to the starting point
B{1}(:,:,2) = [0 0 0; 1 1 1;0 0 0];     % move to safe  option (and check for reward)
B{1}(:,:,3) = [0 0 0; 0 0 0;1 1 1];     % move to risky option (and check for reward)
```

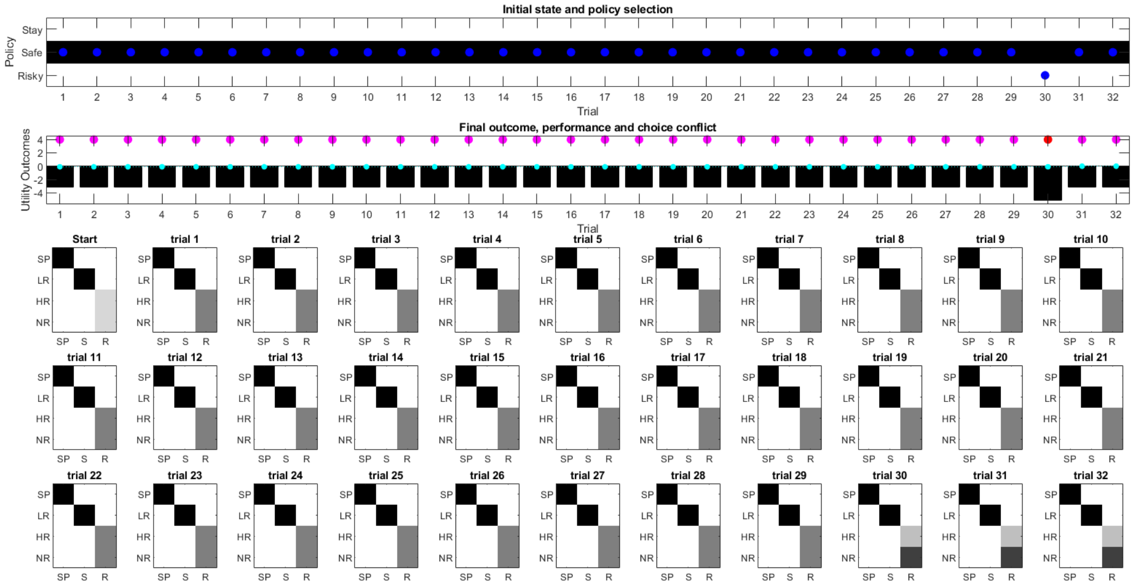
```
%==========================================================================
%==========================================================================
% 1.4 Priors:
%==========================================================================
%==========================================================================

%--------------------------------------------------------------------------
% Finally, we have to specify the prior preferences in terms of log
% probabilities over outcomes. Here, the agent prefers high rewards over
% low rewards over no rewards
%--------------------------------------------------------------------------

cs = 2^1; % preference for safe option
cr = 2^2; % preference for risky option win

C{1}  = [0 cs cr -cs]'; % preference for: [staying at starting point | safe | risky + reward | risky + no reward]

%--------------------------------------------------------------------------
% Now specify prior beliefs about initial state
%--------------------------------------------------------------------------
D{1}  = [1 0 0]'; % prior over starting point - rat 'starts' at starting point (not at safe or risky option)

%==========================================================================
%==========================================================================
% 1.5 Allowable policies (of depth T).  These are sequences of actions
%==========================================================================
%==========================================================================

V     = [1 2 3]; % stay, go left, go right
```

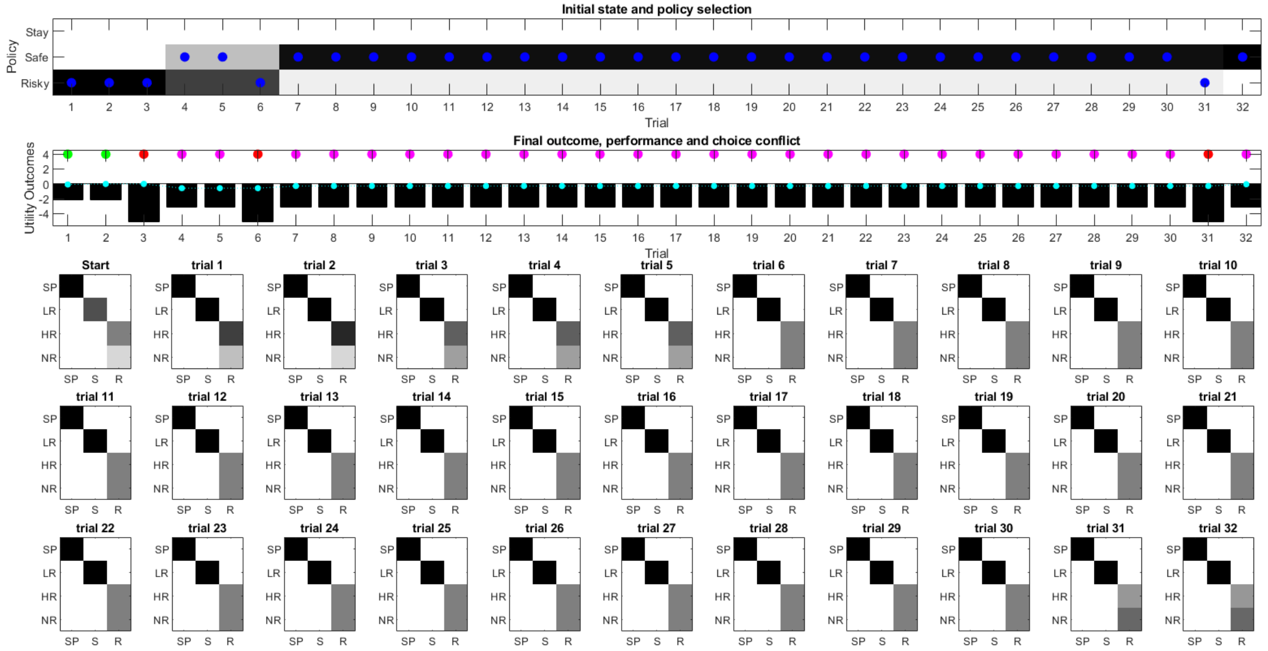
Implemented in Practical_III.m

```
%% 2. Define MDP Structure

mdp.V = V;                      % allowable policies
mdp.A = A;                      % observation process
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;

mdp.eta   = 0.5;                % Learning rate

%--------------------------------------------------------------------------
% Check if all matrix-dimensions are correct:
%--------------------------------------------------------------------------
mdp = spm_MDP_check(mdp);

%--------------------------------------------------------------------------
% Having specified the basic generative model, let's see how active
% infernece solves different tasks
%--------------------------------------------------------------------------
```

Implemented in Practical_III.m

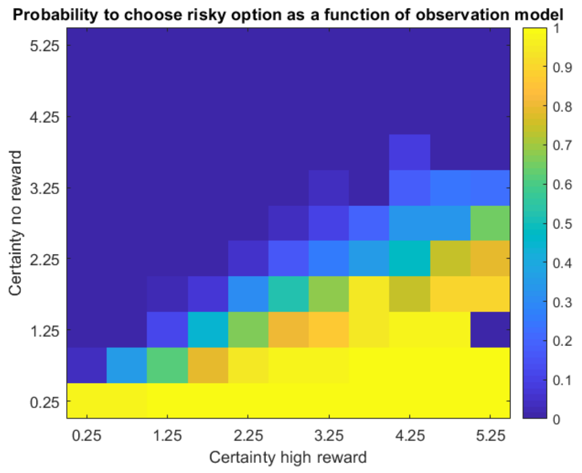
V Apply the model

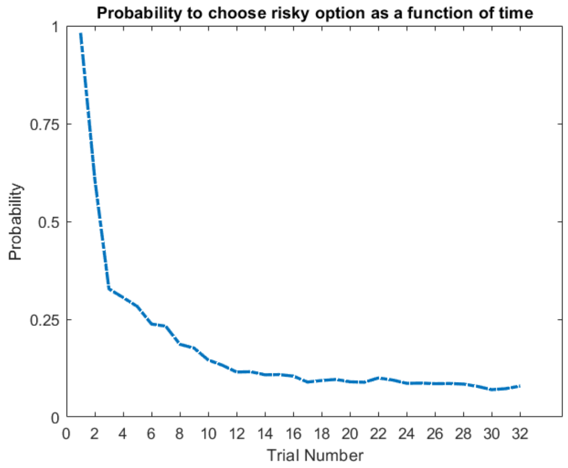


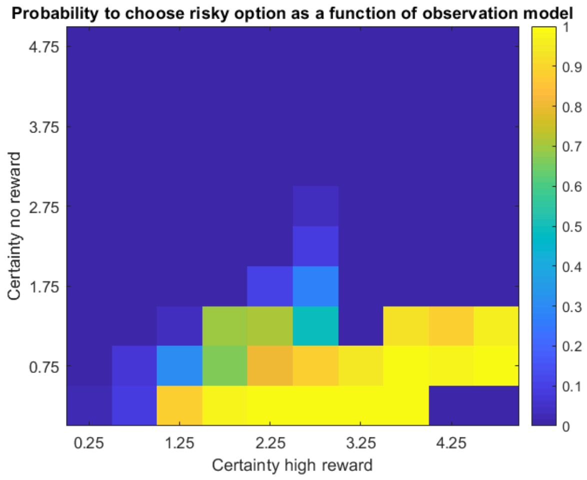


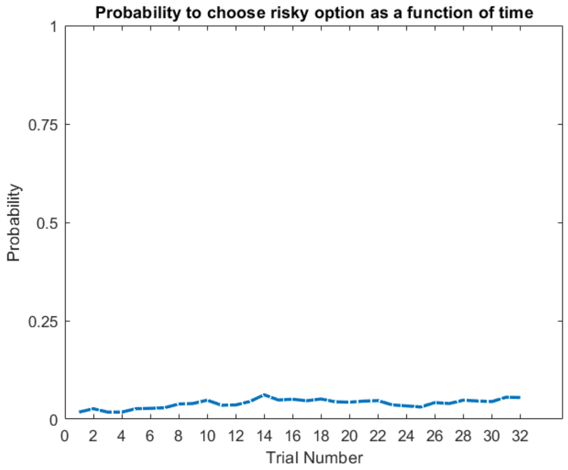
Implemented in Practical_III.m

V Apply the model



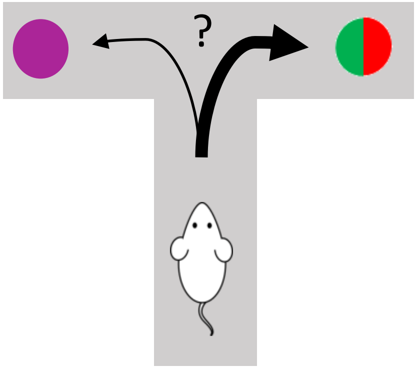


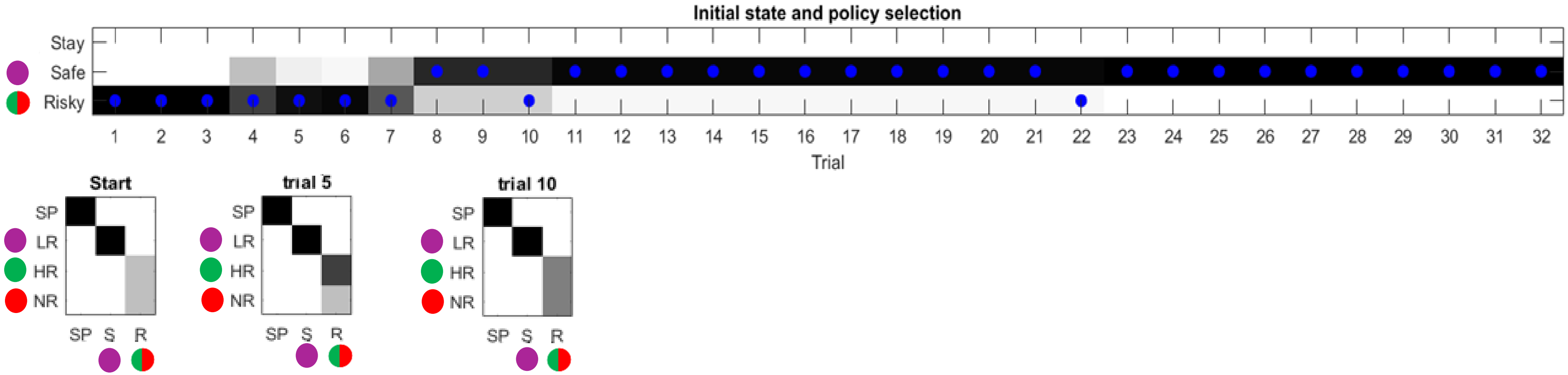




Implemented in Practical_III.m

Summary ‘model parameter exploration’



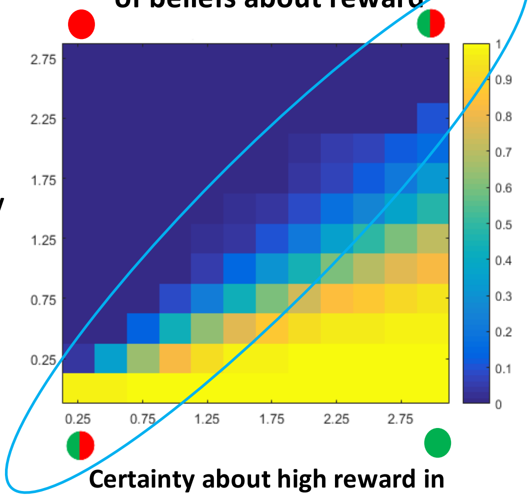


$$Q(\pi) = \mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}[\ln Q(s_\tau, A|\pi) - \ln P(o_\tau, s_\tau, A|\pi)]$$
$$= \cdots$$
$$= \underbrace{-\mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}[H[P(o_\tau|s_\tau)]]}_{\text{Minimising uncertainty}} - \underbrace{D_{KL}[Q(o_\tau|\pi)||P(o_\tau)]}_{\text{Obtaining preferred outcomes}} + \underbrace{\mathbb{E}_{Q(o_\tau, s_\tau, A|\pi)}[\ln Q(A) - \ln Q(A|s_\tau, o_\tau, \pi)]}_{\text{Expected model-update}}$$

Exploring model parameters

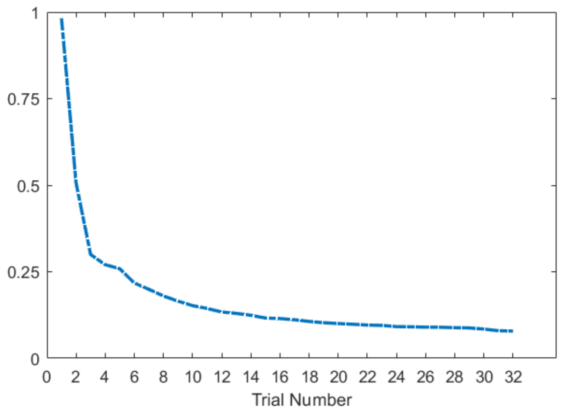
Prob choose risky as a function of beliefs about reward

Certainty about no reward in risky option



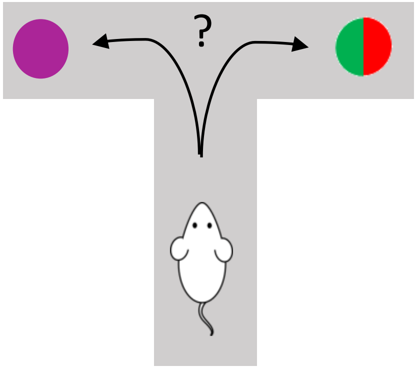
Certainty about high reward in risky option

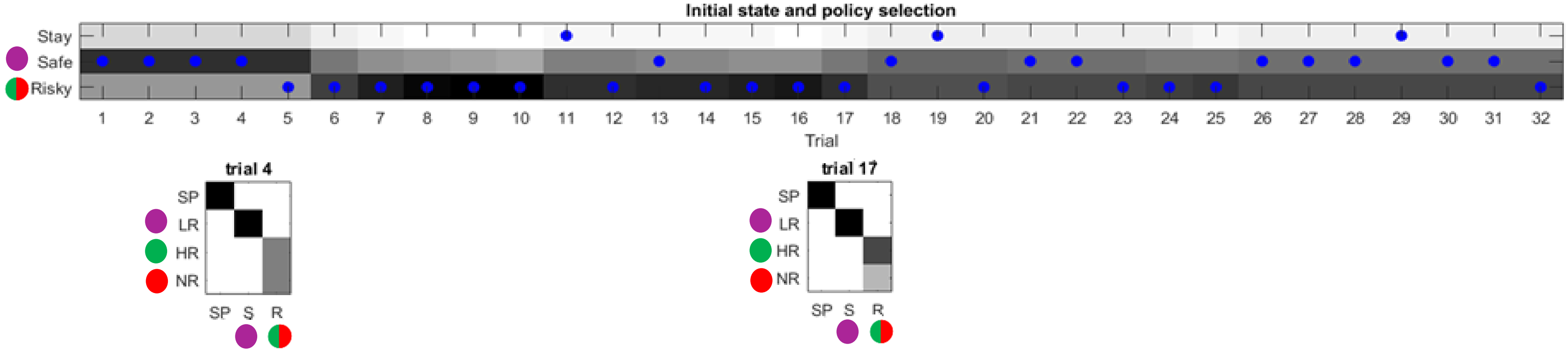
Prob choose risky as a function of time



Schwartenbeck et al., 2018, bioRxiv

Summary **broken** ‘model parameter exploration’



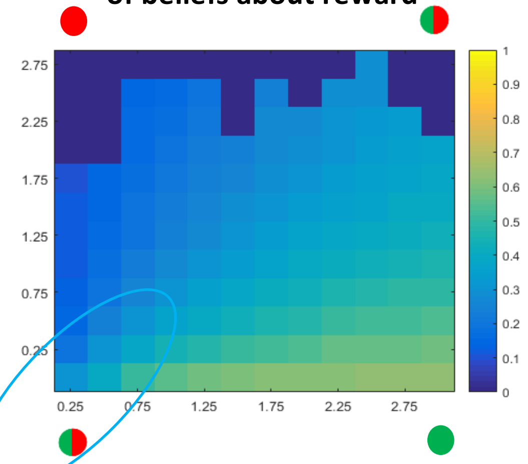


$$Q(\pi) = \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(s_\tau,A|\pi) - \ln P(o_\tau,s_\tau,A|\pi)]$$
$$= \cdots$$
$$= \underbrace{-\mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[H[P(o_\tau|s_\tau)]]}_{\text{Minimising uncertainty}} - \underbrace{D_{KL}[Q(o_\tau|\pi)||P(o_\tau)]}_{\text{Obtaining preferred outcomes}} + \underbrace{\mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\cancel{\ln Q(A|\pi) -} \ln Q(A|s_\tau,o_\tau,\pi)]}_{\text{Expected model-update}}$$

~~Exploring model parameters~~

Prob choose risky as a function of beliefs about reward

Certainty about no reward in risky option



Certainty about high reward in risky option

Schwartenbeck et al., 2018, bioRxiv

Take home messages

Active inference combines **probabilistic inference**, **Markov Decision Processes** and **information theory**

- Actions fulfil expectations ⇔ minimise surprise ⇔ maximise model evidence

Approximate inference takes place based on variational Bayes

- Inference on the current state, policy and confidence

Defining the value of policies as expected free energy predicts that agents try to

- Realise preferences (maximise utility)
- Solicit information from the world

Provides a computational framework for *active inference* and *active learning* - and how this might break

- Exploration of hidden states and model parameters

Thank you!

Supervisors
Karl Friston
Martin Kronbichler
Ray Dolan
Tim Behrens

Collaborators
Tom FitzGerald
Christoph Mathys
Johannes Passecker
Tobias Hauser
Dorothea Hammerer
Rick Adams
Matthew Nour
Lilla Horvath
Thomas Parr
Daniel Freinhofer
Shirley Mark
Yunzhe Liu
Zeb Kurth-Nelson

Derive expected free energy for example I

$$
\begin{aligned}
F &= \mathbb{E}_{Q(x)}[\ln Q(x) - \ln P(x, \tilde{o})] \\
&= \mathbb{E}_{Q(x)}[\ln Q(x) - \ln P(x|\tilde{o}) - \ln P(\tilde{o})] \\
&= D_{KL}[Q(x)||P(x|\tilde{o})] - \ln P(\tilde{o})
\end{aligned}
$$

$$Q(\pi, \tau) = \mathbb{E}_{Q(o_\tau, s_\tau|\pi)}[\ln P(o_\tau, s_\tau\ |\pi) - \ln Q(s_\tau|\pi)]$$

$$\longleftarrow \quad P(o_\tau, s_\tau|\pi) = Q(s_\tau|o_\tau, \pi)P(o_\tau|m)$$

$$= \mathbb{E}_{Q(o_\tau, s_\tau|\pi)}[\ln Q(s_\tau\ |o_\tau, \pi) + \ln P(o_\tau|m) - \ln Q(s_\tau|\pi)]$$

$$= \underbrace{\mathbb{E}_{Q(o_\tau|\pi)}[\mathrm{D}_{KL}[\,Q(s_\tau\ |o_\tau, \pi)\,||Q(s_\tau|\pi)]}_{\text{Epistemic or intrinsic value}} + \underbrace{\mathbb{E}_{Q(o_\tau|\pi)}[\ln P(o_\tau|m)]}_{\text{Extrinsic value}}$$

Derive expected free energy for example II

$$G(\pi,\tau) = \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(o_\tau, s_\tau, A\ |\pi) - \ln P(o_\tau, s_\tau, A|\pi)]$$

$$\leftarrow Q(o_\tau, s_\tau, A|\pi) = Q(s_\tau, A|\pi)P(o_\tau|s_\tau, A)$$

$$G(\pi,\tau) = \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(s_\tau, A|\pi) - \ln P(o_\tau, s_\tau, A|\pi)]$$

$$= \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(A) + \ln Q(s_\tau|\pi) - \ln P(A|s_\tau, o_\tau, \pi) - \ln P(s_\tau|o_\tau,\pi) - \ln P(o_\tau)]$$

$$\approx \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(A) + \ln Q(s_\tau|\pi) - \ln Q(A|s_\tau, o_\tau, \pi) - \ln Q(s_\tau|o_\tau,\pi) - \ln P(o_\tau)]$$

$$= \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}\left[\ln Q(A) - \ln Q(A|s_\tau, o_\tau, \pi)\right] + \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(s_\tau|\pi) - \ln Q(s_\tau|o_\tau,\pi)] - \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}\left[\ln P(o_\tau)\right]$$

$$\leftarrow Q(o_\tau, s_\tau|\pi) = Q(s_\tau|\pi)P(o_\tau|s_\tau)$$

$$= \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}\left[\ln Q(A) - \ln Q(A|s_\tau, o_\tau, \pi)\right] + \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(o_\tau|\pi) - \ln P(o_\tau|s_\tau)] - \mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}\left[\ln P(o_\tau)\right]$$

$$= \underbrace{\mathbb{E}_{Q(o_\tau,s_\tau,A|\pi)}[\ln Q(A) - \ln Q(A|s_\tau, o_\tau, \pi)]}_{\text{Expected model-update}} + \underbrace{D_{KL}[Q(o_\tau|\pi)||P(o_\tau)]}_{\text{Realising preferences}} + \underbrace{\mathbb{E}_{Q(o_\tau,s_{\tau,}A|\pi)}[H[P(o_\tau|s_\tau)]]}_{\text{Minimising uncertainty}}$$