Bayesian Statistics and Hierarchical Bayesian Modeling for Psychological Science

Lecture 10

Lei Zhang
Social, Cognitive and Affective Neuroscience Unit (SCAN-Unit)
Department of Cognition, Emotion, and Methods in Psychology

lei.zhang@univie.ac.at
lei-zhang.net
@lei_zhang_lz

https://github.com/lei-zhang/BayesCog_Wien

universität wien
Fakultät für Psychologie

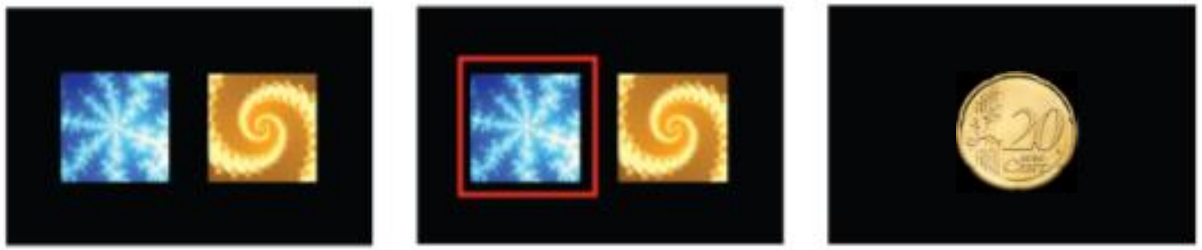
Bayesian warm-up?

Rescorla-Wagner (1972)

- The idea: error-driven learning
- Change in value is proportional to the difference between actual and predicted outcome

Robert A. Rescorla

Allan R. Wagner



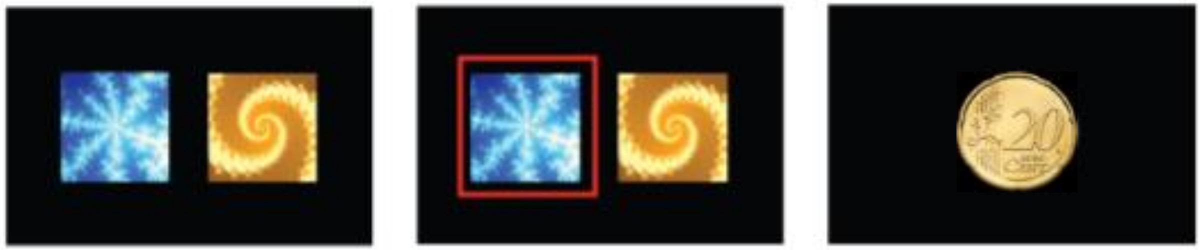
Value update: $V_t = V_{t-1} + \alpha * \mathrm{PE}_{t-1}$

Prediction error: $\mathrm{PE}_{t-1} = R_{t-1} - V_{t-1}$

α – learning rate
PE – reward prediction error
V – value
R – reward

*Expectations on the next trial = the expectation on the current trial + learning rate * prediction error (reward – current expectation)*

Choice rule: softmax

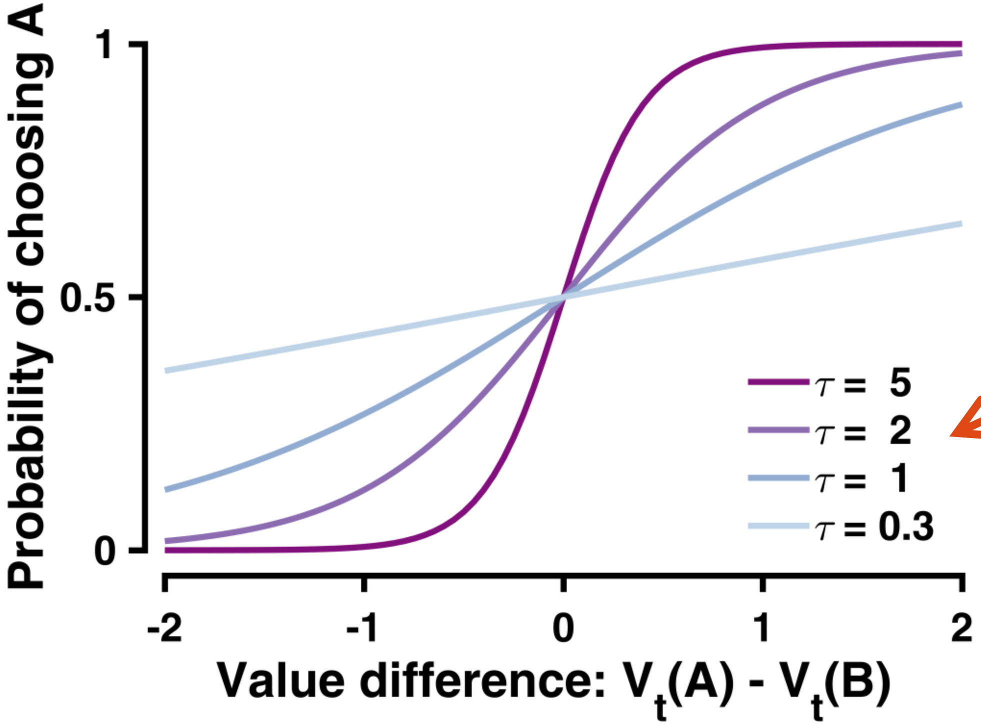


V()

V()

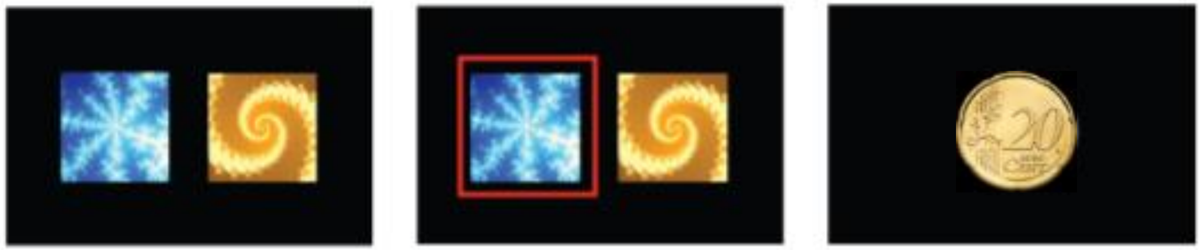
$$\boxed{p_t(A) = \frac{e^{\tau * V_t(A)}}{e^{\tau * V_t(A)} + e^{\tau * V_t(B)}}}$$

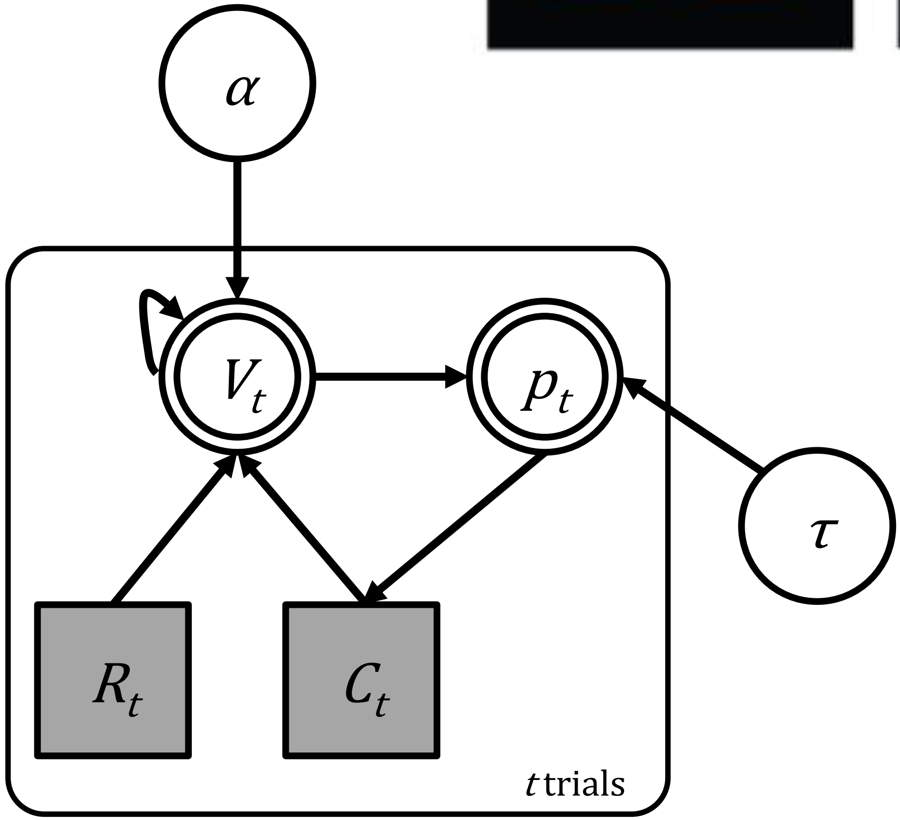
$$= \frac{1}{1 + e^{-\tau * (V_t(A) - V_t(B))}}$$



cognitive model
statistics
computing

RL – Implementation





$$\alpha \sim Uniform(0,1)$$

$$\tau \sim Uniform(0,3)$$

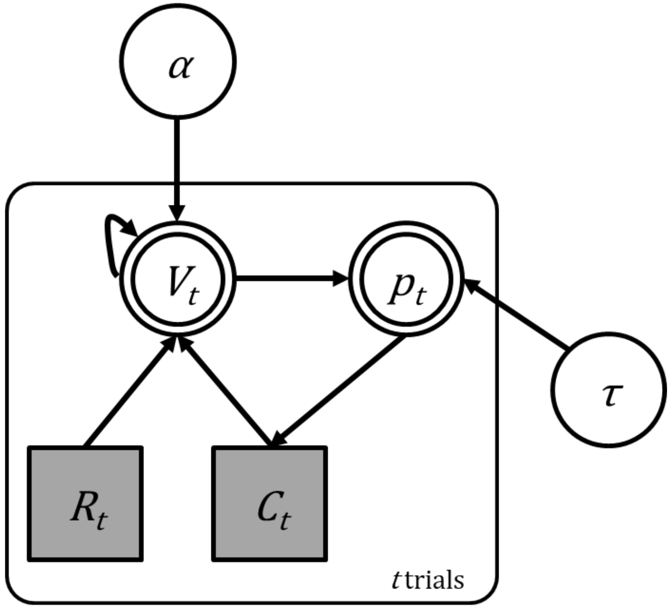
$$p_t(C = A) = \frac{1}{1 + e^{\tau(V_t(B) - V_t(A))}}$$

$$V_{t+1}^{c} = V_t^{C} + \alpha\left(R_t - V_t^{C}\right)$$

 deterministic variable

RL – Implementation

cognitive model
statistics
computing



$$\alpha \sim Uniform(0,1)$$

$$\tau \sim Uniform(0,3)$$

$$p_t(C = A) = \frac{1}{1 + e^{\tau(V_t(B) - V_t(A))}}$$

$$V_{t+1}^{c} = V_t^C + \alpha(R_t - V_t^C)$$

```
transformed data {
  vector[2] initV;
  initV = rep_vector(0.0, 2);
}

model {
  vector[2] v[nTrials+1];
  real pe[nTrials];

  v[1] = initV;

  for (t in 1:nTrials) {
    choice[t] ~ categorical_logit( tau * v[t] );

    pe[t] = reward[t] - v[t,choice[t]];

    v[t+1] = v[t];
    v[t+1, choice[t]] = v[t, choice[t]] + lr * pe[t];
  }
}
```

RL – Implementation

cognitive model
statistics
computing

```
model {
  vector[2] v[nTrials+1];
  real pe[nTrials];

  v[1] = initV;

  for (t in 1:nTrials) {
    choice[t] ~ categorical_logit( tau * v[t] );
    pe[t] = reward[t] - v[t,choice[t]];

    v[t+1] = v[t];
    v[t+1, choice[t]] = v[t, choice[t]] + lr * pe[t];
  }
}
```

```
model {
  vector[2] v;
  real pe;

  v = initV;

  for (t in 1:nTrials) {
    choice[t] ~ categorical_logit( tau * v );
    pe = reward[t] - v[choice[t]];

    v[choice[t]] = v[choice[t]] + lr * pe;
  }
}
```

RL – Fitting with Stan

cognitive model
statistics
computing

```
…/06.reinforcement_learning/_scripts/reinforcement_learning_single_parm_main.R
```

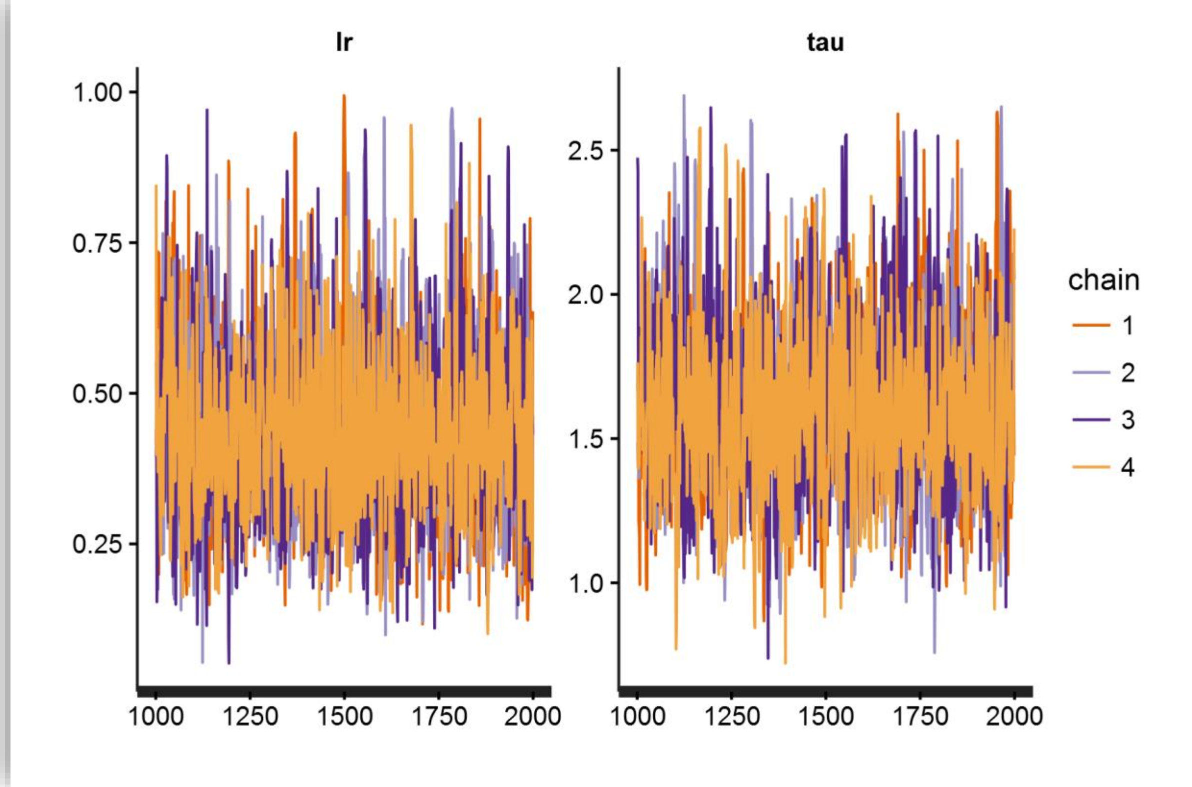
TASK: fit the model for single participants

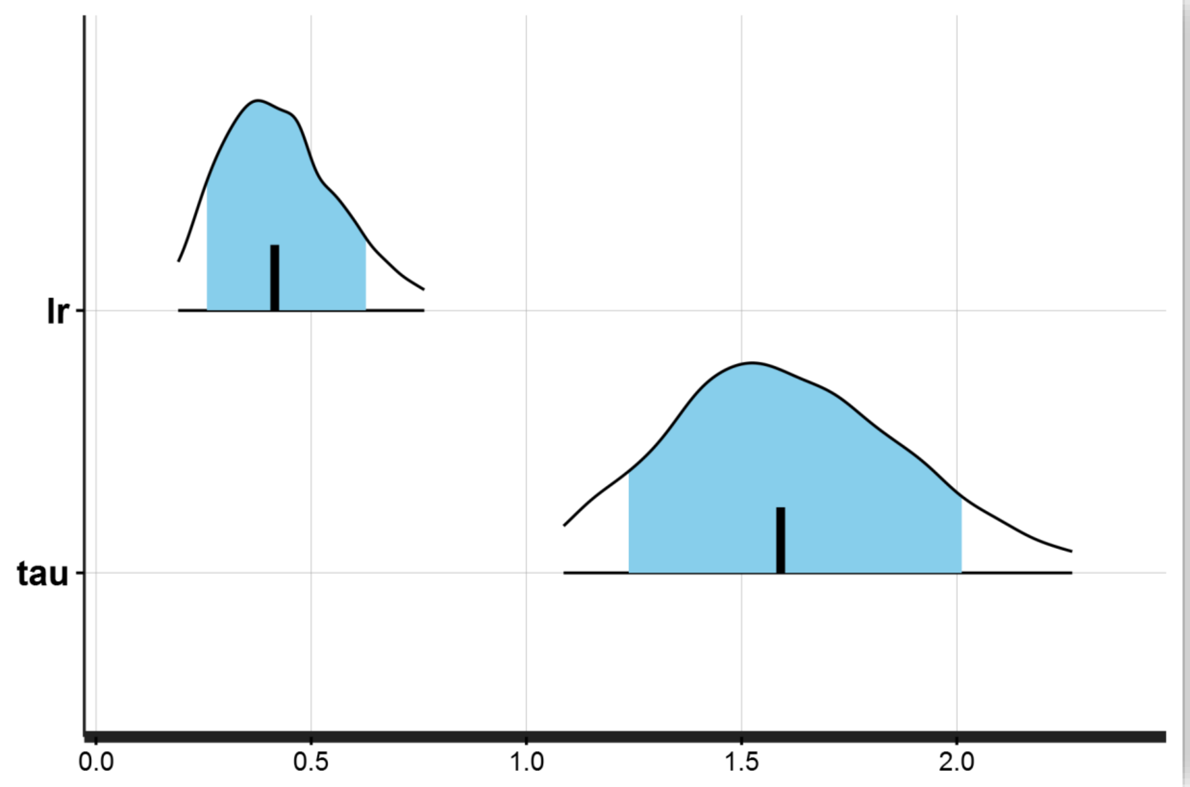
```
> source('_scripts/reinforcement_learning_single_parm_main.R’) # a function

> fit_rl1 <- run_rl_sp(multiSubj = FALSE)
```

```
> load('_data/rl_sp_ss.RData')
> head(rl_ss)
     [,1] [,2]
[1,]    2   -1
[2,]    1    1
[3,]    1    1
[4,]    1    1
[5,]    2   -1
[6,]    1    1
```

RL – MCMC Output

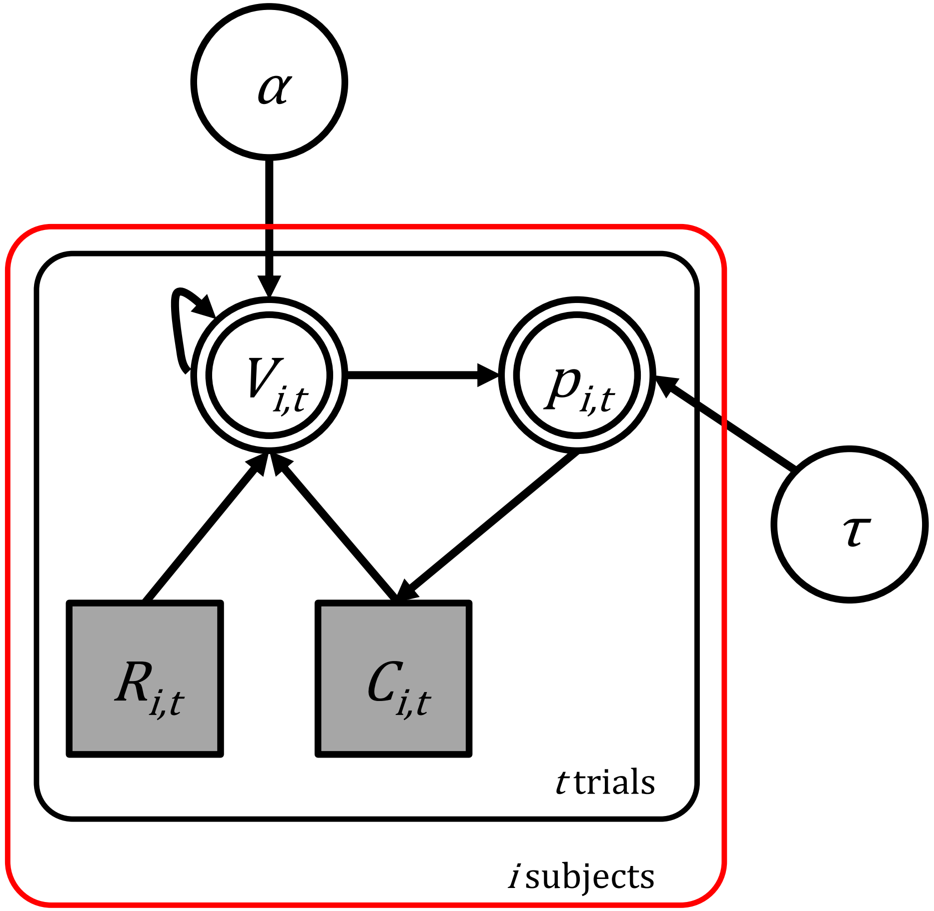
cognitive model
statistics
computing





Fitting Multiple Participants as ONE

cognitive model
statistics
computing




```
model {
  for (s in 1:nSubjects) {
    vector[2] v;
    real pe;
    v = initV;

    for (t in 1:nTrials) {
      choice[s,t] ~ categorical_logit( tau * v );
      pe = reward[s,t] - v[choice[s,t]];
      v[choice[s,t]] = v[choice[s,t]] + lr * pe;
    }
  }
}
```

Exercise X

cognitive model
statistics
computing

```
…/06.reinforcement_learning/_scripts/reinforcement_learning_single_parm_main.R
```

TASK:

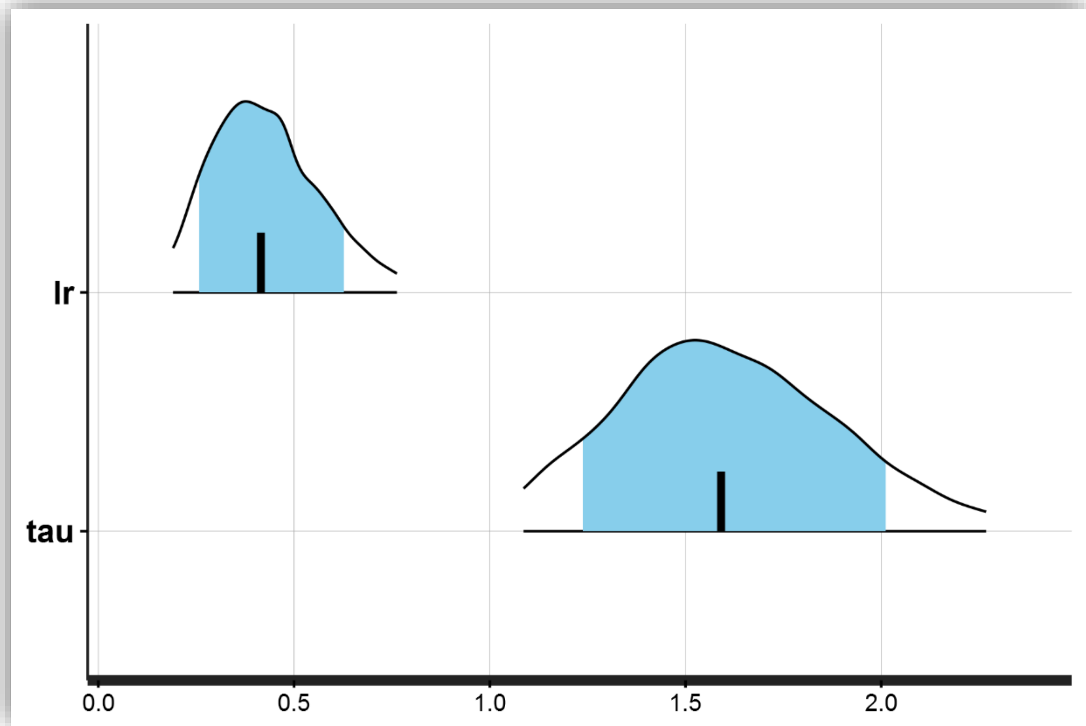
(1) complete the model (Tip: the for-loop)

(2) fit the model for multiple participants (assuming same parameters)

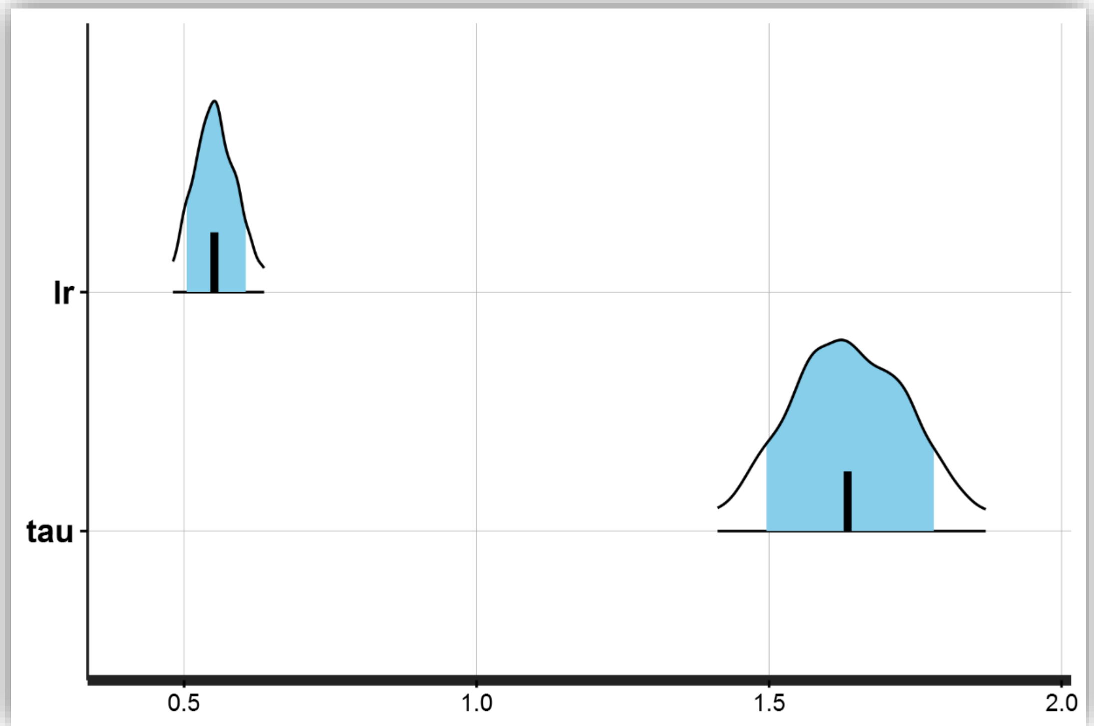
```
> source('_scripts/reinforcement_learning_single_parm_main.R')

> fit_rl2 <- run_rl_sp(multiSubj = TRUE)
```

Exercise X

N = 1

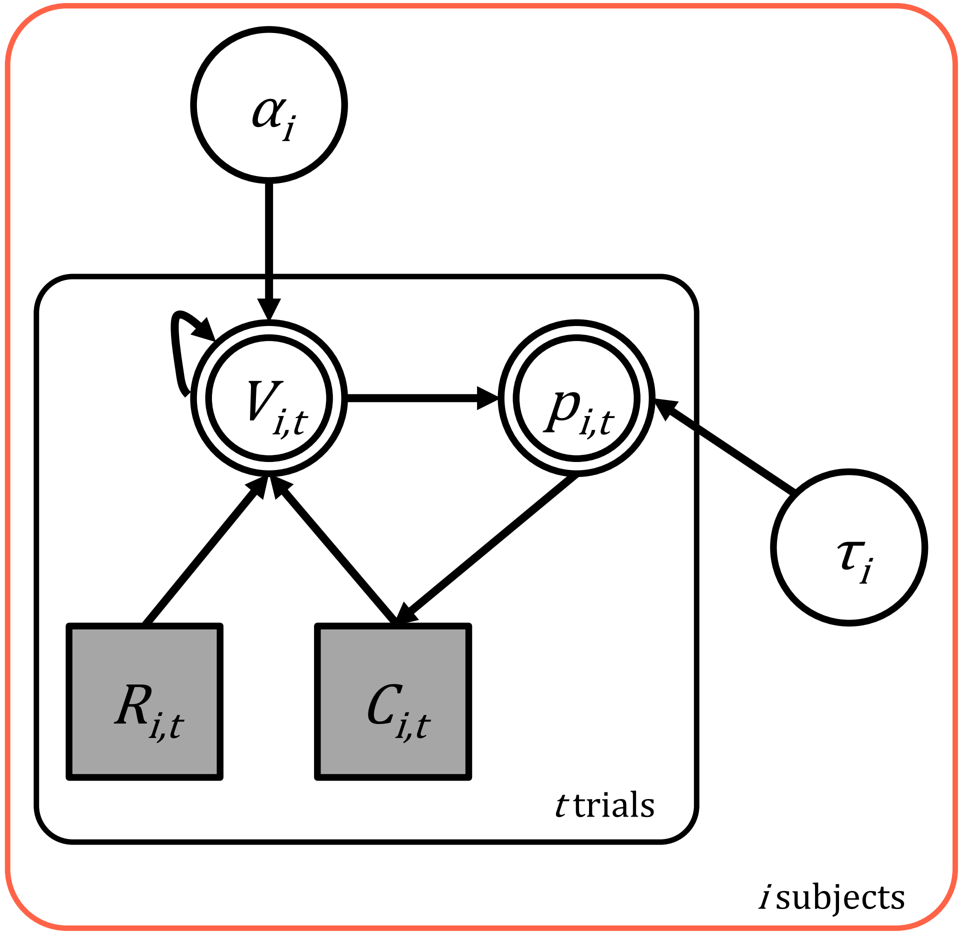


N = 10



Fitting Multiple Participants Independently

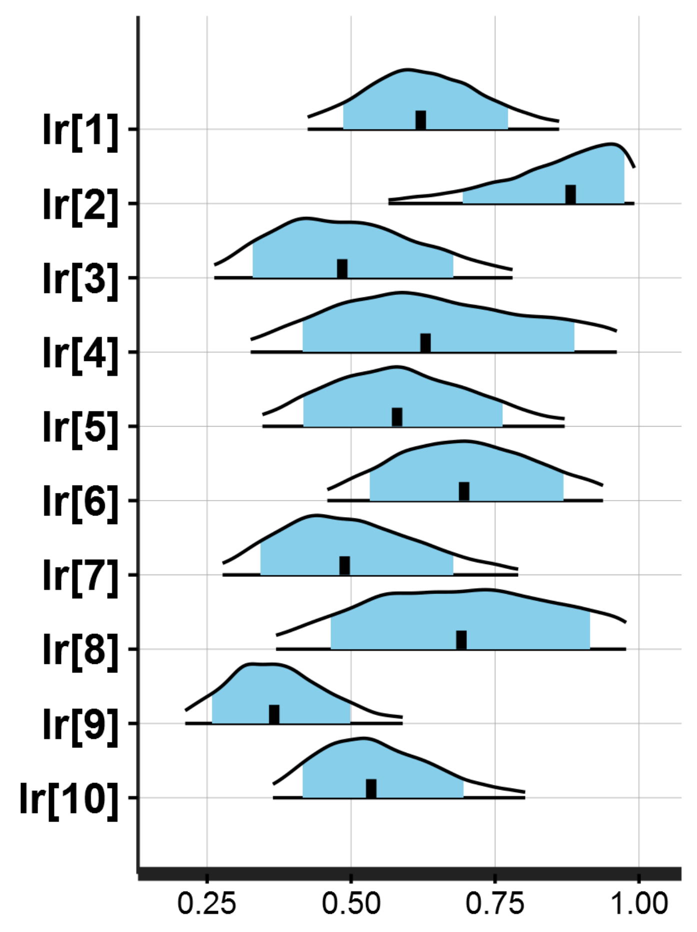
cognitive model
statistics
computing

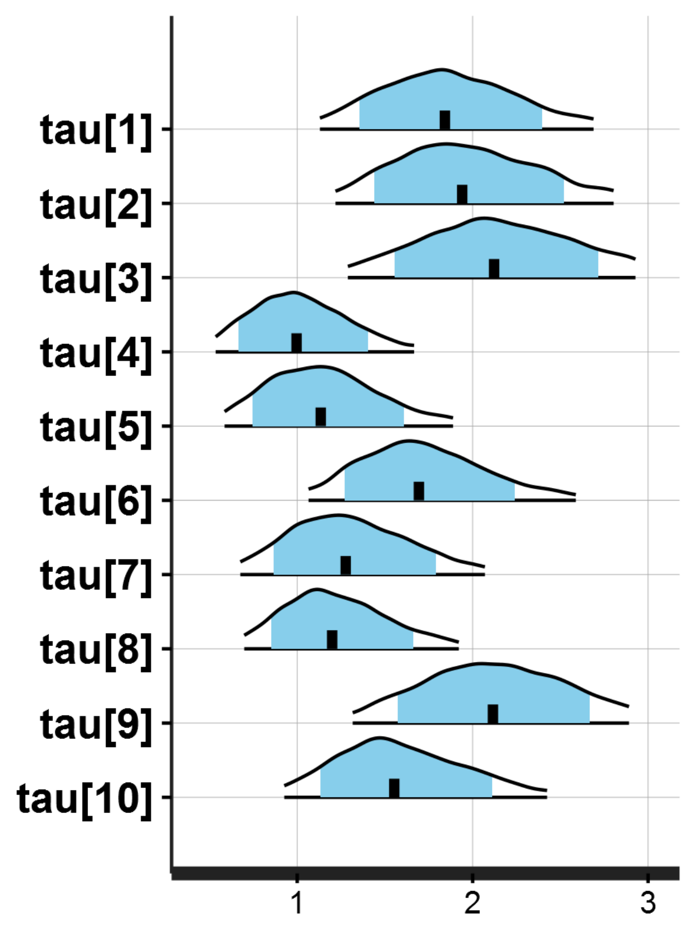


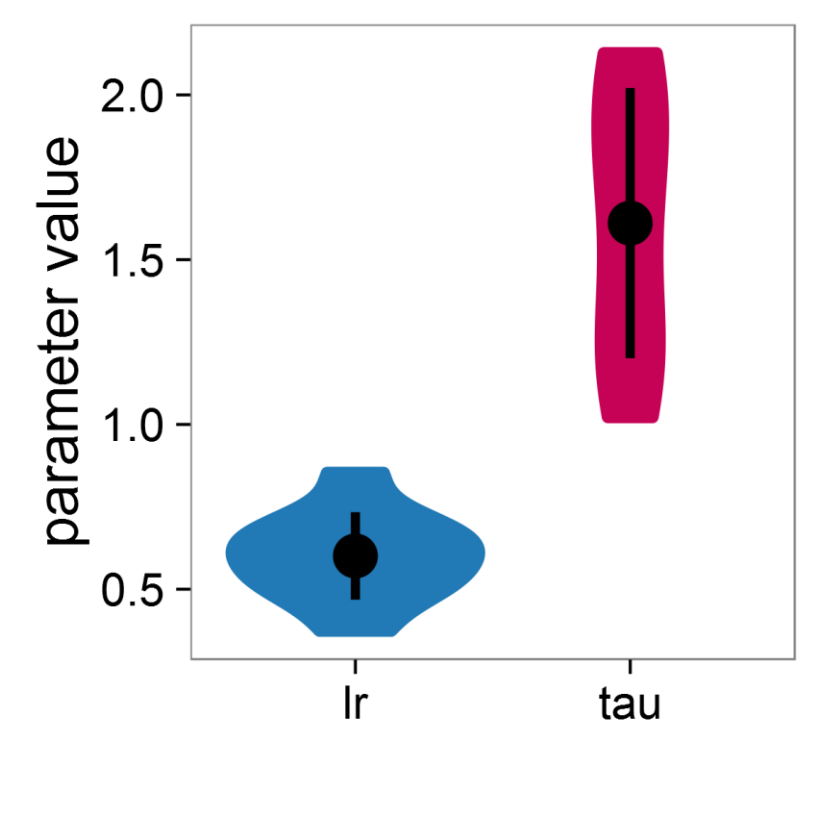

```
model {
  for (s in 1:nSubjects) {
    vector[2] v;
    real pe;
    v = initV;

    for (t in 1:nTrials) {
      choice[s,t] ~ categorical_logit( tau[s] * v );
      pe = reward[s,t] - v[choice[s,t]];
      v[choice[s,t]] = v[choice[s,t]] + lr[s] * pe;
    }
  }
}
```

Individual Fitting

cognitive model
statistics
computing



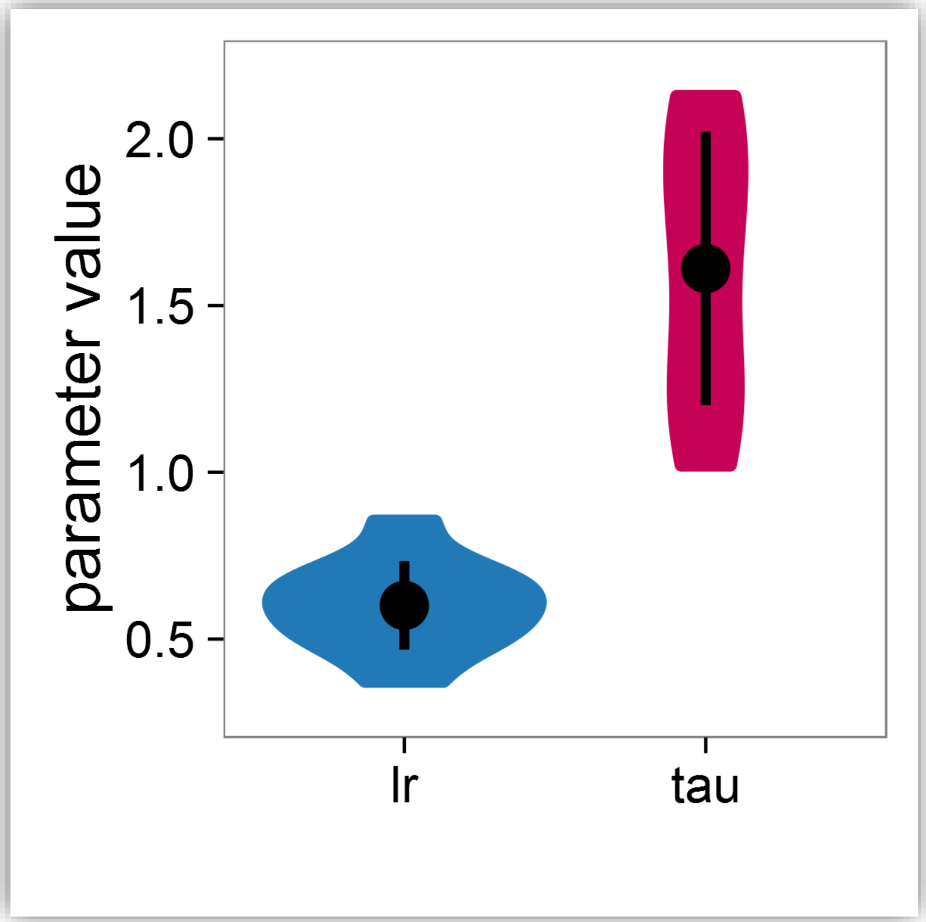




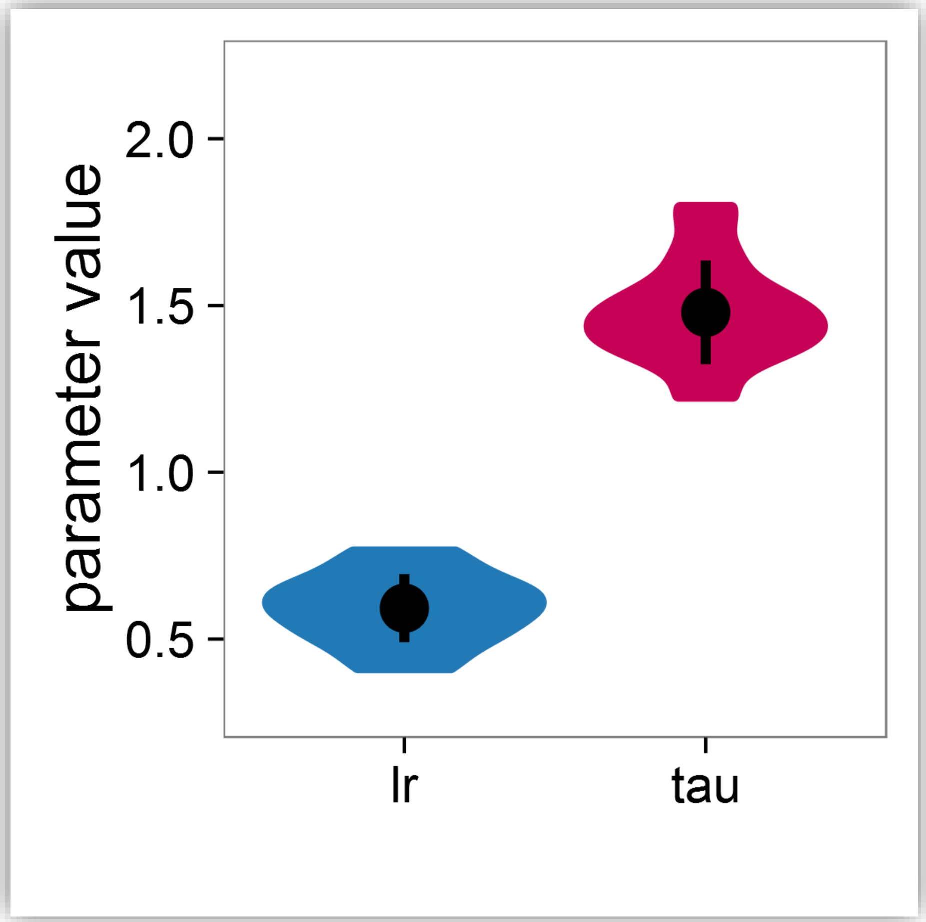
Comparing with True Parameters

cognitive model
statistics
computing

Posterior Means



True Parameters




ANY
QUESTIONS
?

Happy Computing!