# Workshop on the HGF-toolbox

Tore Erdmann, Lilian Weber

Zurich Computational Psychiatry Course
14.09.2018

# Overview

- General framework
- Analysis workflow
- Intro to toolbox
- Response models
- Start with exercises

08.30-09.30

- Break

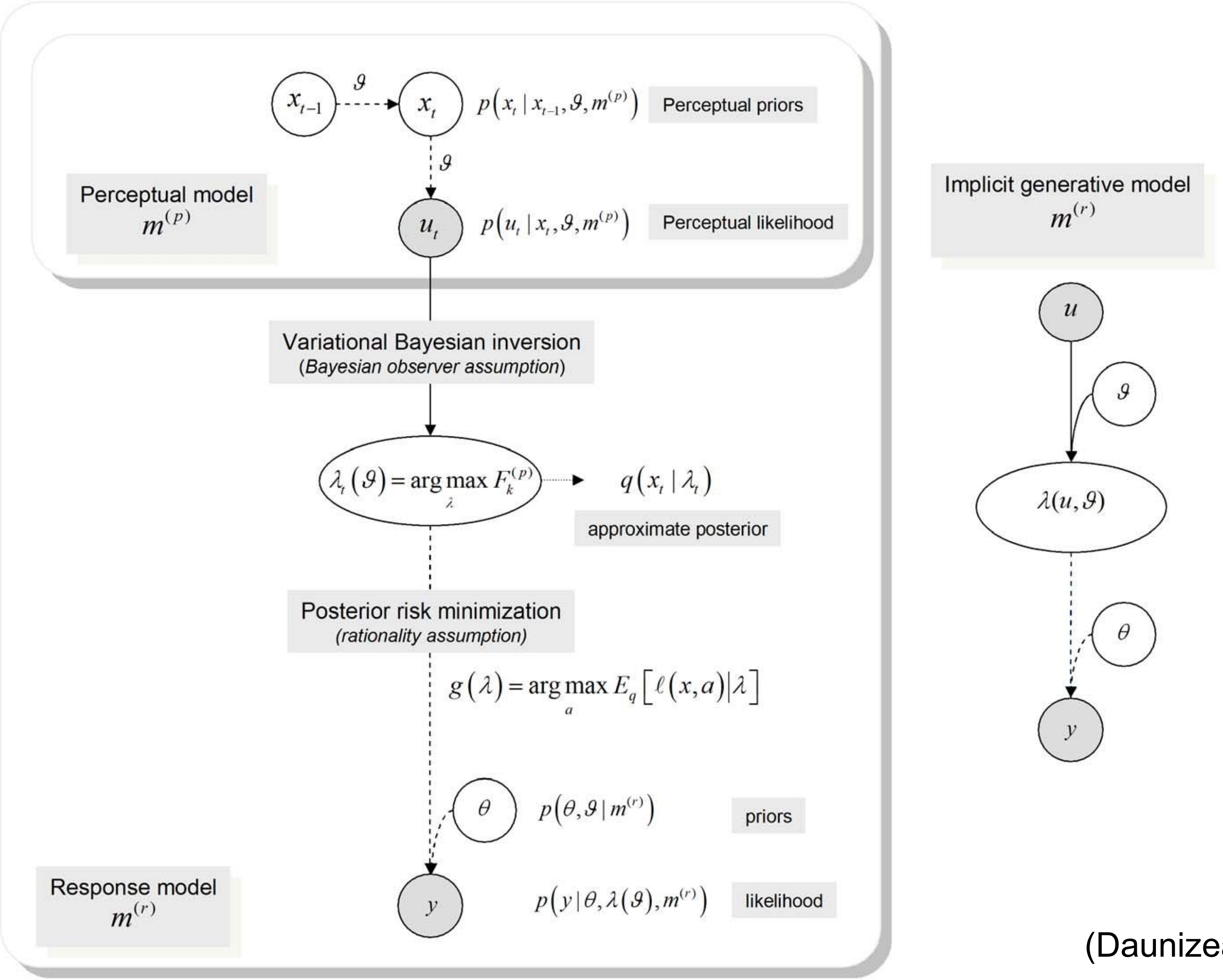
09.30-10.00

- Continue exercises
- Discussion / questions

10.00-11.30

# Introduction

- We want to understand learning
- Learning = belief updating
- **Beliefs** here are (gaussian) probability distributions, and are updated based on inputs and a **perceptual model**
- **Responses** at each time are based on the current belief, as specified by a **response model**
- Together, these models provide a **generative model** of the responses of the learner, which **allows** us to do **inference on the model parameters**

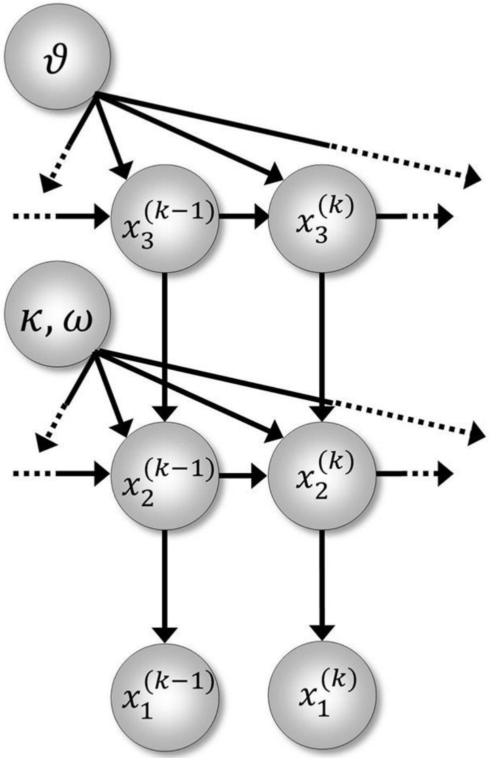


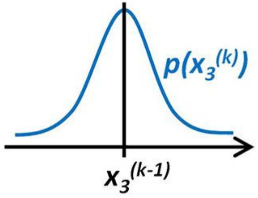
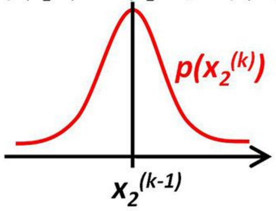
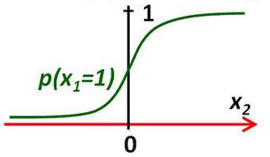

(Daunizeau et al., 2010)

# Example: Reversal learning paradigm

- Two slot machines: Each trial, the participant makes a binary choice and is rewarded or not rewarded
- There is always one good machine, having 0.8 probability to give reward and can be reversed
- To do well, the agent needs to estimate (and update following a reversal) the probability for winning with each machine
- Tracking a single probability is enough because we have P(right is better) = 1 - P(left is better)
- Initially the agent is uncertain about which is better, which can be expressed as the belief P(left is better) = .5

# The generative model at the base of the learner's perceptual model




| State of the world | Model | |
|---|---|---|
| Log-volatility **$x_3$** of tendency | Gaussian random walk with constant step size $\vartheta$ | $p(x_3^{(k)}) \sim N(x_3^{(k-1)}, \vartheta)$   |
| Tendency **$x_2$** towards category "1" | Gaussian random walk with step size $\exp(\kappa x_3+\omega)$ | $p(x_2^{(k)}) \sim N(x_2^{(k-1)}, \exp(\kappa x_3+\omega))$   |
| Stimulus category **$x_1$** ("0" or "1") | Sigmoid trans-formation of $x_2$ | $p(x_1=1) = s(x_2)$ $p(x_1=0) = 1-s(x_2)$   |

# Example: Reversal learning paradigm

At the bottom of the generative model, the learners observations are produced by:

$$u^{(k)} \sim Ber(x_1^{(k)})$$

which leads to these updates for the belief about the latent process,

$$\mu_2^{(k)} = \mu_2^{(k-1)} + \frac{1}{\pi_2^{(k)}}\delta_1^{(k)}$$

and which directly implies the degree of belief for the next observation:

$$\mu_1^{(k)} = s(\mu_2^{(k)})$$

# Example: Reversal learning paradigm

This update is similar in form to a Rescorla-Wagner update: it adjusts the old prediction by a weighted prediction error:

$$\mu_2^{(k)} = \mu_2^{(k-1)} + \frac{1}{\pi_2^{(k)}} \delta_1^{(k)}$$

However, the weighting factor for the HGF is dynamic and reflects different kinds of uncertainty:

$$\frac{1}{\pi_2^{(k)}} = \frac{1}{\sigma^{(k-1)} + t^{(k)} \exp(\kappa \mu_3^{(k-1)} + \omega)} + \frac{1}{\hat{\mu}_1^{(k)} (1 - \hat{\mu}_1^{(k)})}$$

estimation uncertainty

estimated volatility of the environment (affected by parameters)

irreducible uncertainty about the outcome

## Update equations for expectations

| | | | |
|---|---|---|---|
| Level 3 | $\Delta\mu_3 = \sigma_3 \cdot \frac{\kappa}{2} \cdot w_2 \cdot \delta_2$ | with | $\Delta\mu_3 = \mu_3^{(k)} - \mu_3^{(k-1)}$<br>$\sigma_3 = \sigma_3^{(k)}$<br>$w_2 = \frac{e^{\kappa\mu_3^{(k-1)}+\omega}}{\sigma_2^{(k-1)} + e^{\kappa\mu_3^{(k-1)}+\omega}}$<br>$\delta_2 = \frac{\sigma_2^{(k)} + \left(\mu_2^{(k)} - \mu_2^{(k-1)}\right)^2}{\sigma_2^{(k-1)} + e^{\kappa\mu_3^{(k-1)}+\omega}} - 1$ |
| Level 2 | $\Delta\mu_2 = \sigma_2 \cdot \delta_1$ | with | $\Delta\mu_2 = \mu_2^{(k)} - \mu_2^{(k-1)}$<br>$\sigma_2 = \sigma_2^{(k)}$<br>$\delta_1 = \mu_1^{(k)} - s\left(\mu_2^{(k-1)}\right)$ |

Expectation update
(Unweighted) learning rate
Weighting factor
Prediction error

## Update equation for posterior means

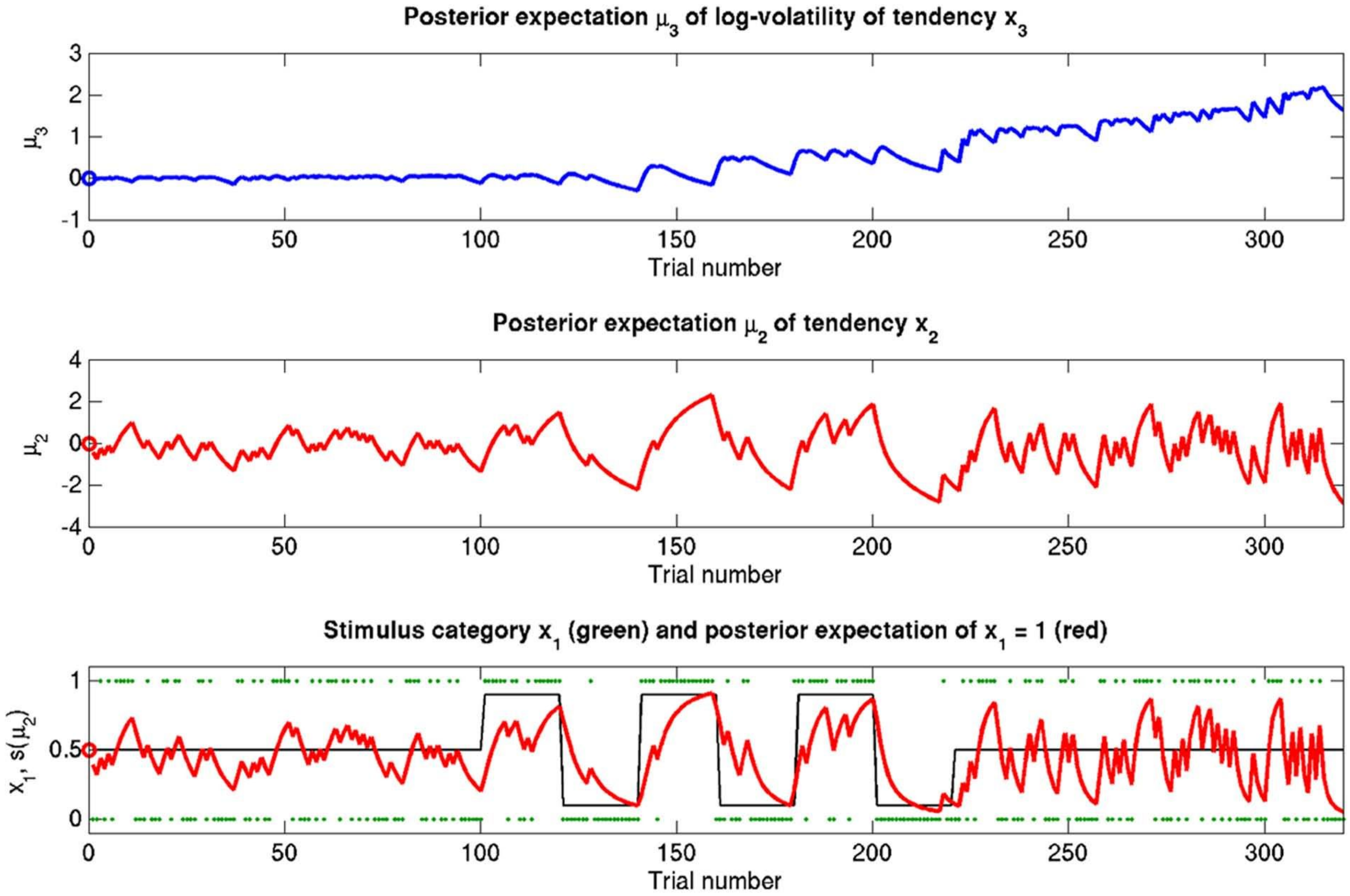
$$\Delta\mu_i^{(k)} = \frac{1}{2}\kappa_{i-1} v_{i-1}^{(k)} \frac{\hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)}$$

$$\Longrightarrow \quad \boxed{\Delta\mu_i \propto \frac{\hat{\pi}_{i-1}}{\pi_i} \delta_{i-1}}$$

with

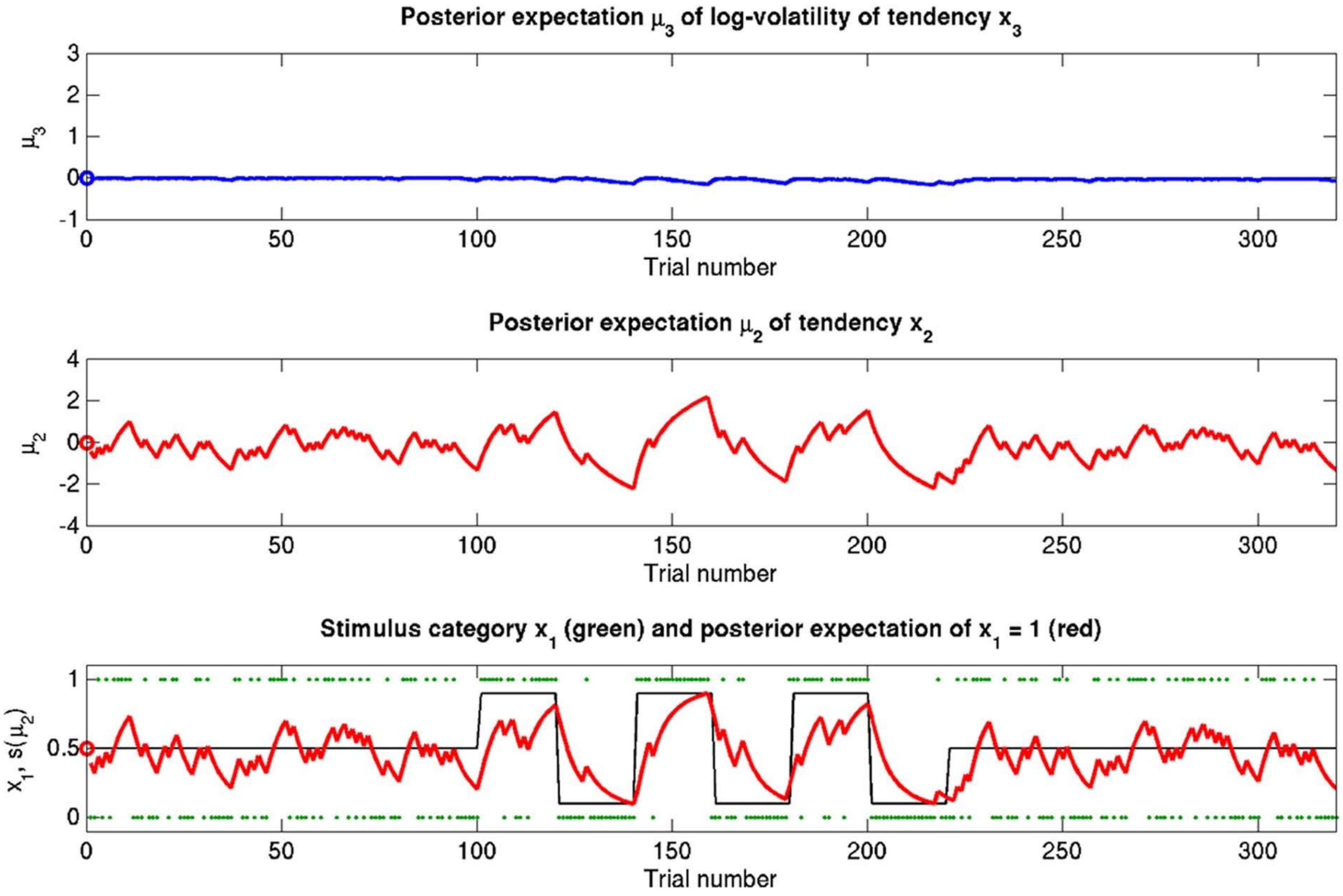
| | |
|---|---|
| $\hat{\pi}_{i-1}$ | Precision of the prediction onto the level below |
| $\pi_i$ | Posterior precision at the current level |
| $\delta_{i-1}$ | Prediction error for the volatility of the level below |

# Binary HGF

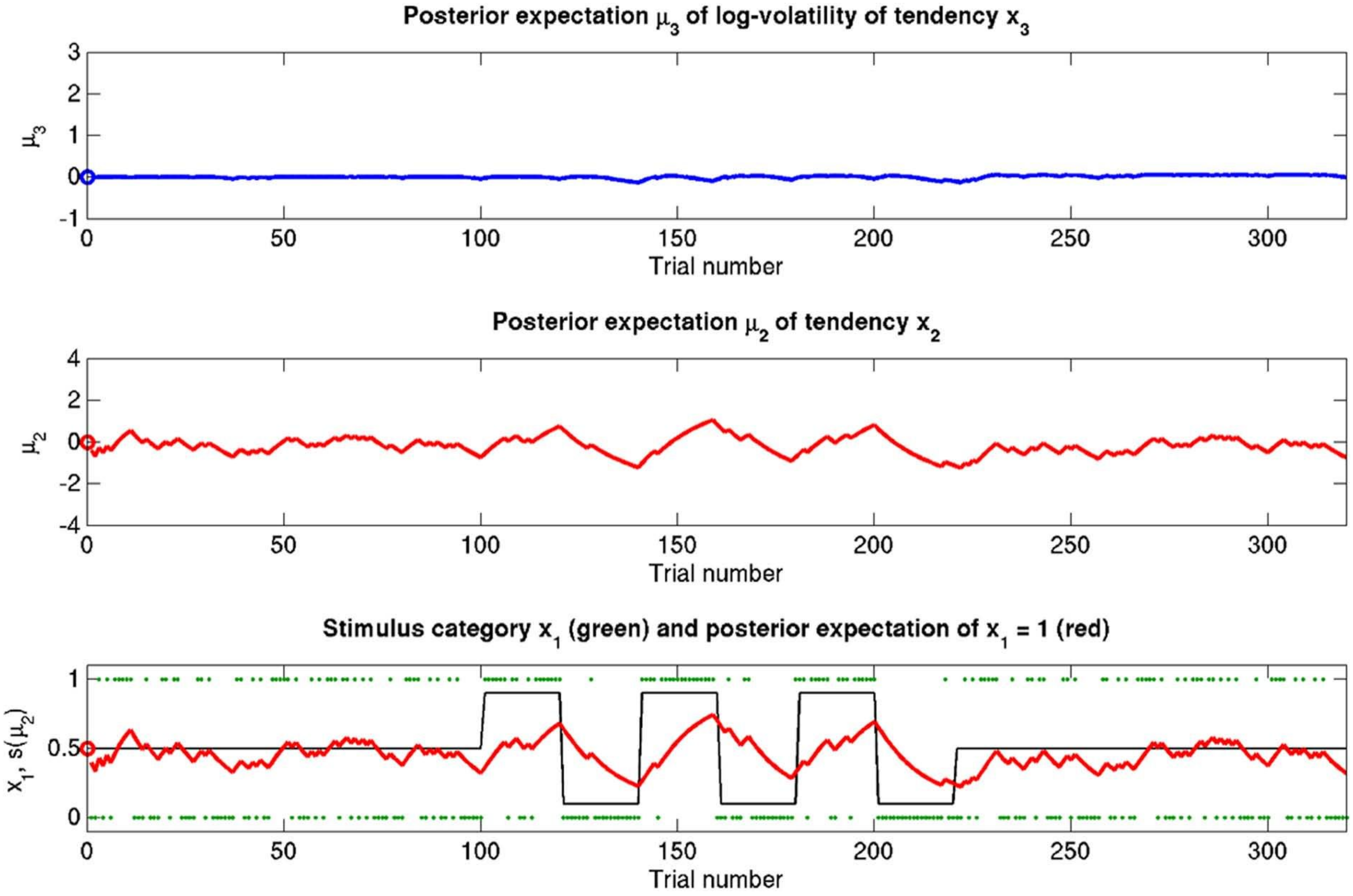


Posterior expectation $\mu_3$ of log-volatility of tendency $x_3$
3
2
1
0
-1
$\mu_3$
0
50
100
150
200
250
300
Trial number
Posterior expectation $\mu_2$ of tendency $x_2$
4
2
0
-2
-4
$\mu_2$
0
50
100
150
200
250
300
Trial number
Stimulus category $x_1$ (green) and posterior expectation of $x_1$ = 1 (red)
1
0.5
0
$x_1$, $s(\mu_2)$
0
50
100
150
200
250
300
Trial number

# Binary HGF: reduced theta



# Binary HGF: reduced omega



# Adding a response model

- Ok, so we have a model about how to update beliefs
- Now we need another model to map beliefs to responses
- Some simple suggestions for binary:
  - If p > .5 do action 1
  - Sample action ~ bernoulli(p)
  - Binary softmax with ‘decision temperature’ → allows modeling exploration / noisiness of responses
- Response models can also be regression models, drift diffusion, LBAs ...

# Types of input- and response variables

- Think about the data analysis before starting the data collection
- Binary inputs/responses is not as informative as graded inputs/responses

| | **Input: binary** | **Input: continuous** |
|---|---|---|
| **Output: binary** | Input: blue/green card<br>Choice: yes/no | Input: pitch<br>Choice: yes/no |
| **Output: continuous** | Input: blue/green card<br>Choice: log(RT) | Input: pitch<br>Choice: log(RT) |

# Analysis workflow

1. Specify model (and design experiment)
2. Setting priors / prior predictive checking
3. (Get data)
4. Fit model
5. Check model
6. Good enough? Publish. Else, adjust model and go to 1.

# HGF-toolbox

- Functions for fitting, simulation, diagnostics
- Various choices for perceptual and response models (and can be extended):

| **Belief trajectory:** | **Response models:** |
|---|---|
| Binary | Binary (square sigmoid, binary softmax) |
| Categorical | Categorical (softmax) |
| Continuous | Continuous (gaussian, linear regression) |

# Important functions

- Fitting the model

```
fit = tapas_fitModel(responses, inputs, <prc_model_config>,
                    <obs_model_config>);
```

- Simulation from the model

```
sim = tapas_simModel(inputs, <prc_model>, prc_model_parameters, ...
                            <obs_model>, obs_model_parameters);
```

- Plotting belief trajectories / simulated responses

```
tapas_hgf_plotTraj(fit);

tapas_hgf_binary_plotTraj(fit);
```

# Using `tapas_fitModel`

This call estimates parameters for given inputs and responses:

```
fit = tapas_fitModel(responses, inputs, ...
                     <prc_model_config>, ...
                     <obs_model_config>, ...
                     <opt_algo>);
```

‘tapas_hgf’
‘tapas_hgf_binary’
‘tapas_rw_binary’
‘tapas_hgf_categorical’

‘tapas_gaussian_obs_config’
‘tapas_softmax_binary_config’

# Using `tapas_simModel`

This call simulates responses for given inputs and parameters:

```
sim = tapas_simModel(inputs, ...
                     <prc_model>, ...
                     prc_parameters, ...
                     <obs_model>, obs_parameters);
```

```
'tapas_hgf'
'tapas_hgf_binary'
'tapas_rw_binary'
'tapas_hgf_categorical'
```

```
'tapas_gaussian_obs'
'tapas_softmax_binary'
```

# Model comparison

- When to choose between models?
  - Finding the right model
  - Justifying your model
- How to choose a model?
  - Do prior predictive checks
  - Only keep sensible models, you don’t need try every possible model
  - Do posterior predictive checks: is the model missing something?
  - Compare models using the model evidence p(y|M)
- \`tapas_fitModel\` computes the model evidence for you
  - est.optim.LME
  - Fixed-effects analysis: for each model sum the LME values of each participant and take exp of the difference
  - Random-effects analysis: use \`spm_BMS\` from the SPM toolbox

On to the exercises!