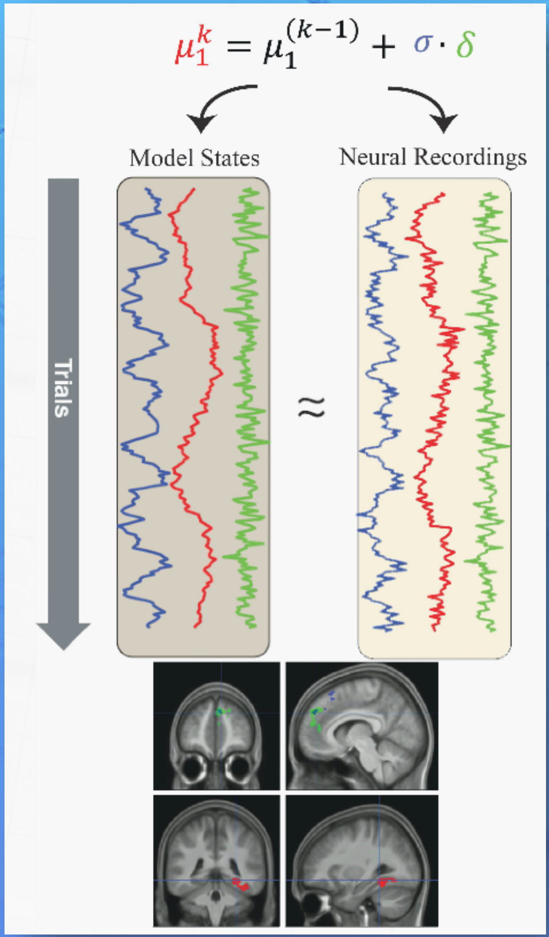
Computational Neuroimaging

Andreea Diaconescu

Methods & Models 2016



Dark Room Experiment

Translational Neuromodeling Unit

What is it all about?

Translational Neuromodeling Unit

- Why do we use functional magnetic resonance imaging?
  - To measure brain activity
- When does the brain become active?
  - When it learns
    i.e., when its predictions about the world have to be adjusted
- Where do these predictions come from?
  - A model

Advantages of model-based neuroimaging

Translational Neuromodeling Unit

- Model-based neuroimaging permits us to:
  - **Infer** the computational mechanisms underlying brain function
  - **Localize** such mechanisms
  - **Compare** different models

Explanatory Gap

Translational Neuromodeling Unit

Biological

- Molecular
- Neurochemical

Cognitive

- Computational
- “cognitive/
- computational phenotyping”

Phenomenological

- Performance Accuracy
- Reaction Time
- Choices, preferences

Computational Models

Three Levels of Inference

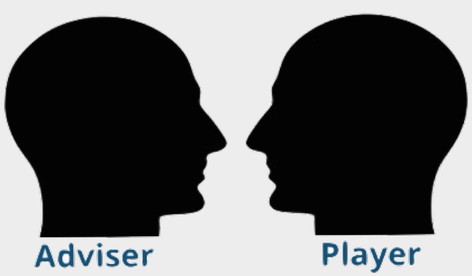
Translational Neuromodeling Unit

- *Computational Level*: predictions, prediction errors
- *Algorithmic Level*: reinforcement learning, hierarchical Bayesian inference, predictive coding
- *Implementational Level*: Brain activity, neuromodulation

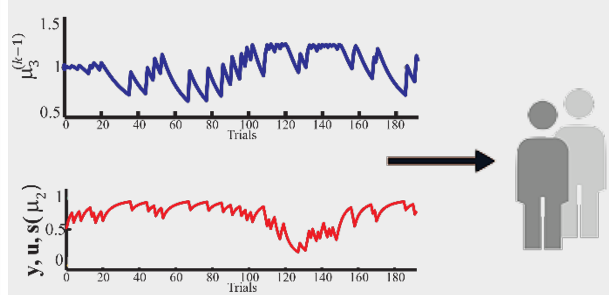
David Marr, 1982

- **3 ingredients**:

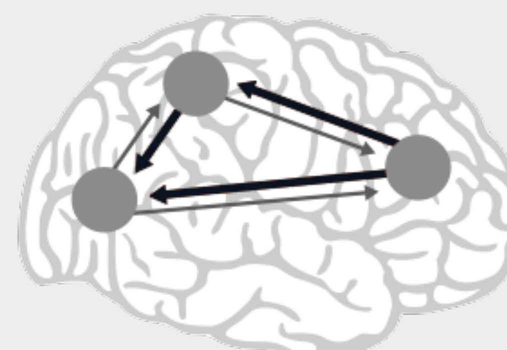
1. Experimental paradigm:



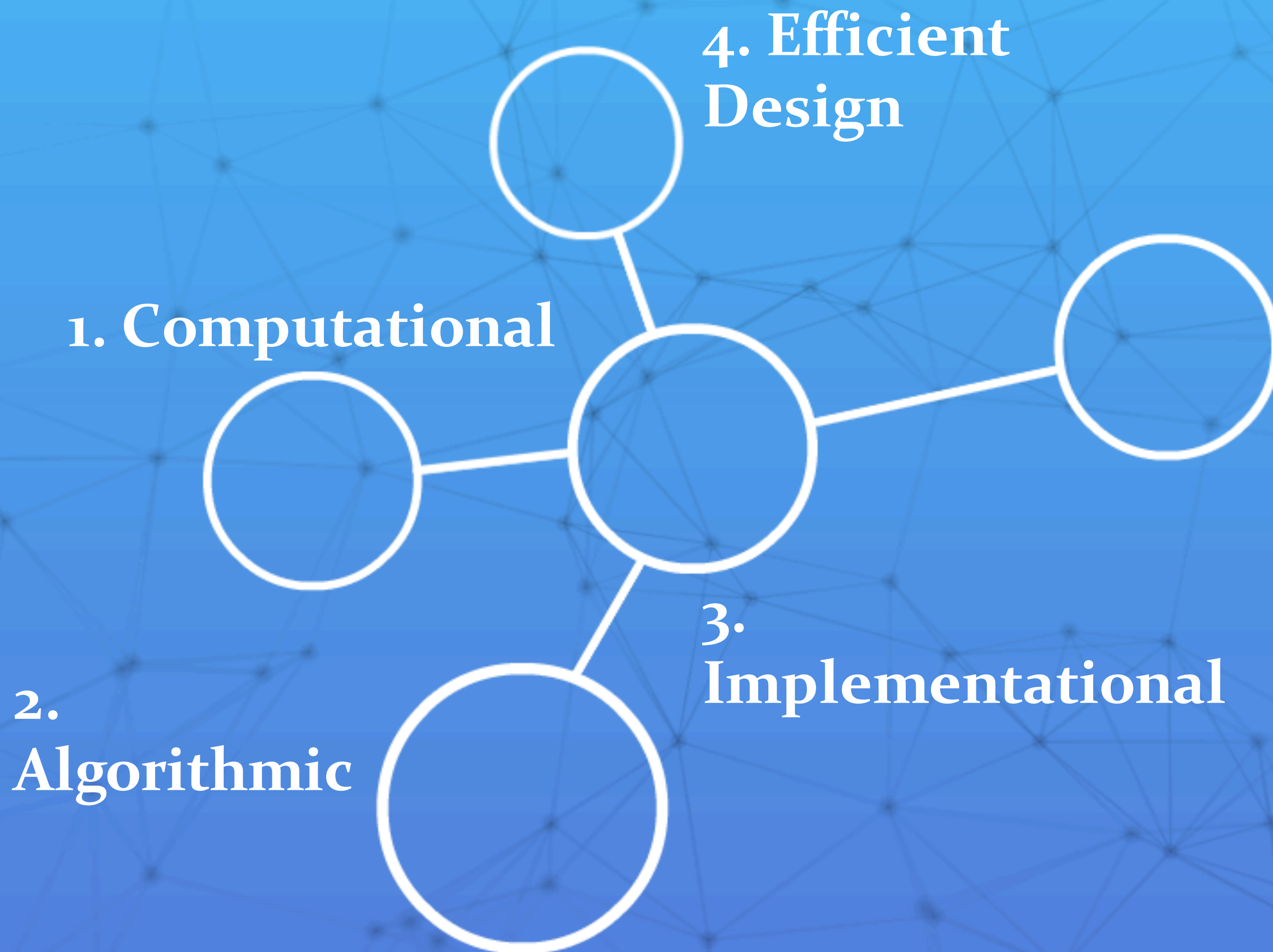
2. Computational model of learning:



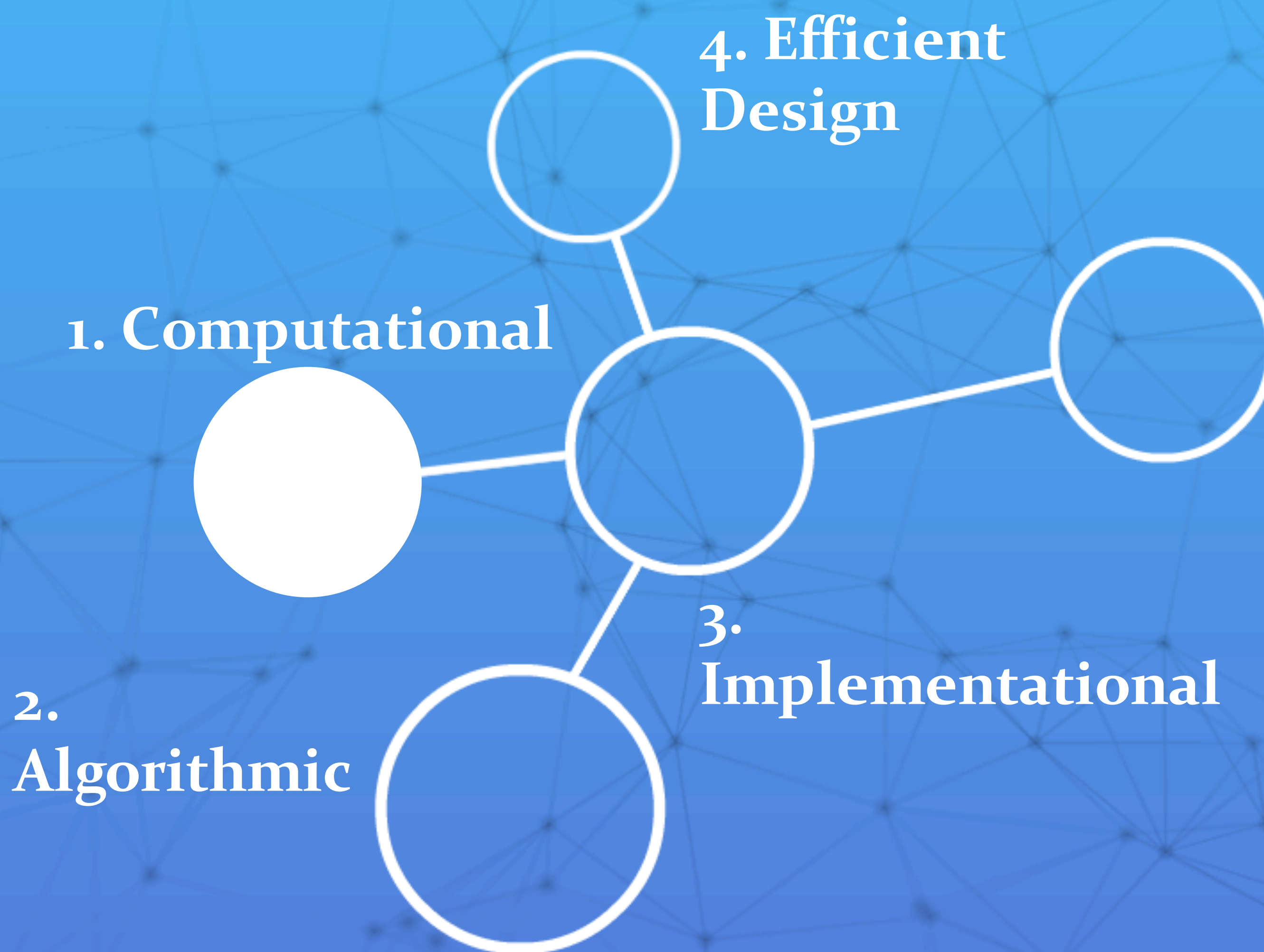
3. Model-based fMRI analysis:



Outline

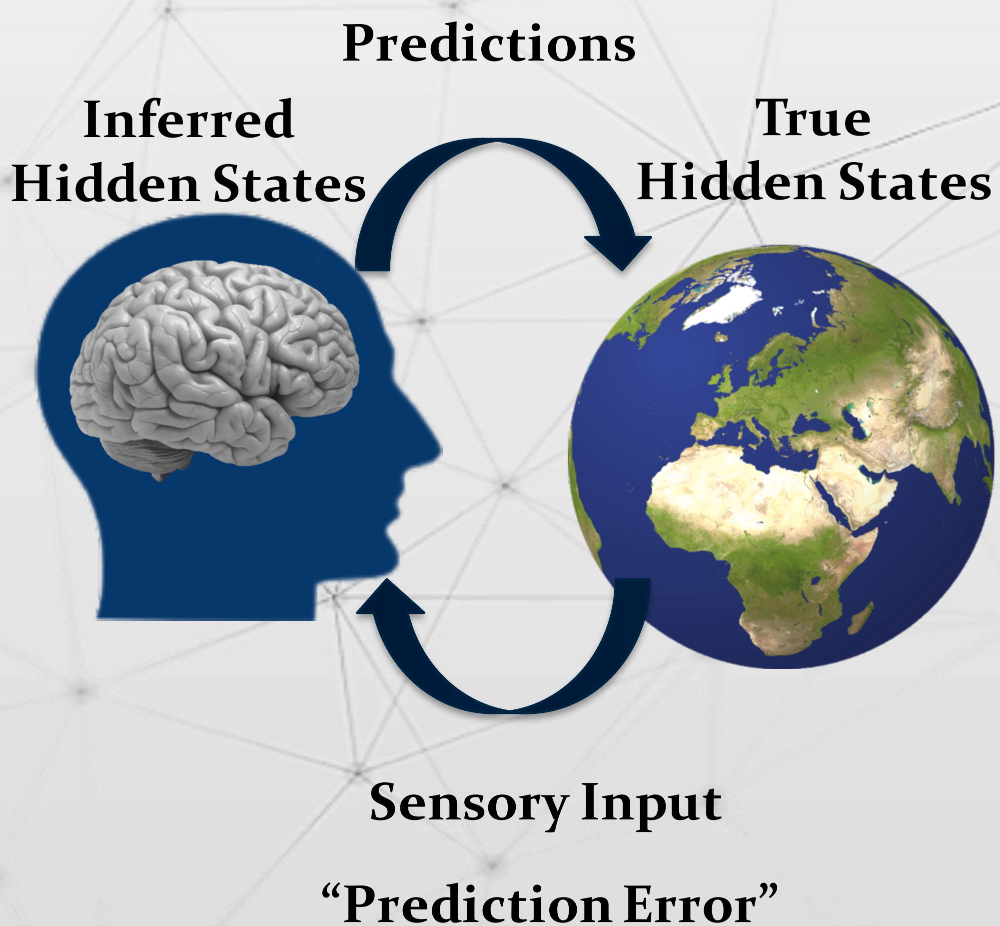

4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

Outline


4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

How to build a model

Translational Neuromodeling Unit



Example of a simple model

Translational Neuromodeling Unit

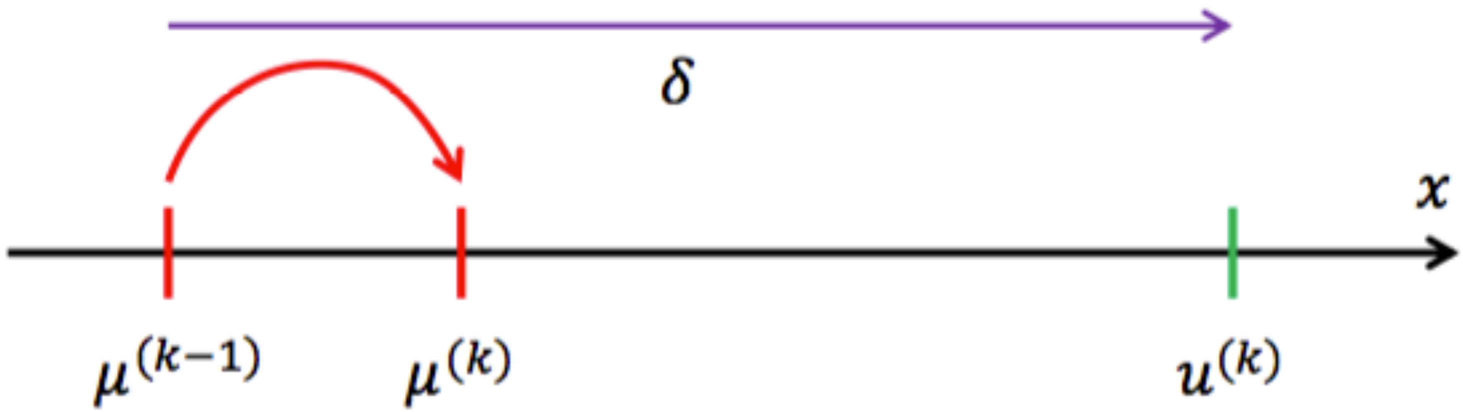
Rescorla-Wagner Learning:

Prediction Error

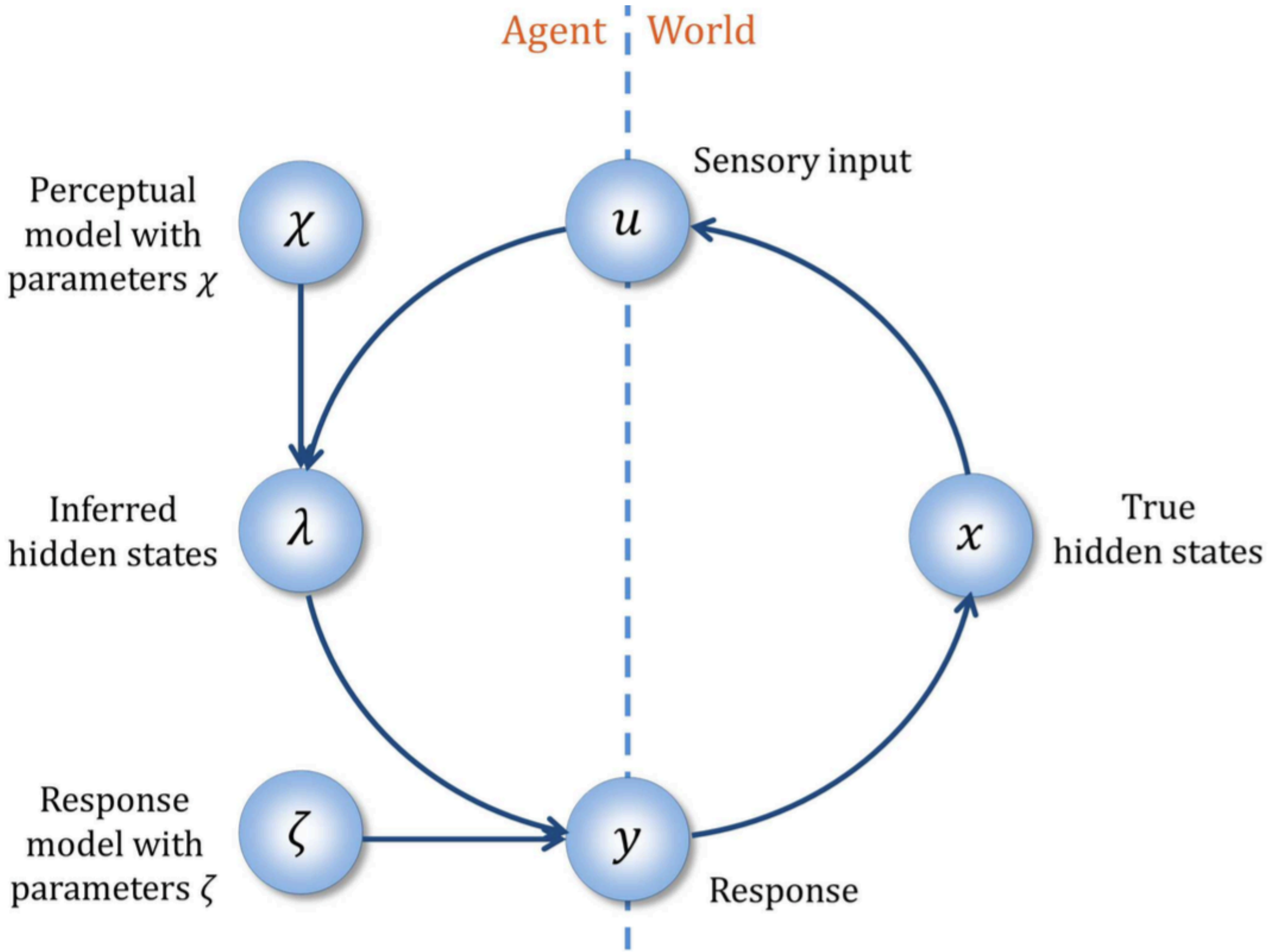
$$\mu^{(k)} = \mu^{(k-1)} + \alpha(u^{(k)} - \mu^{(k-1)})$$

Inferred States

New Input

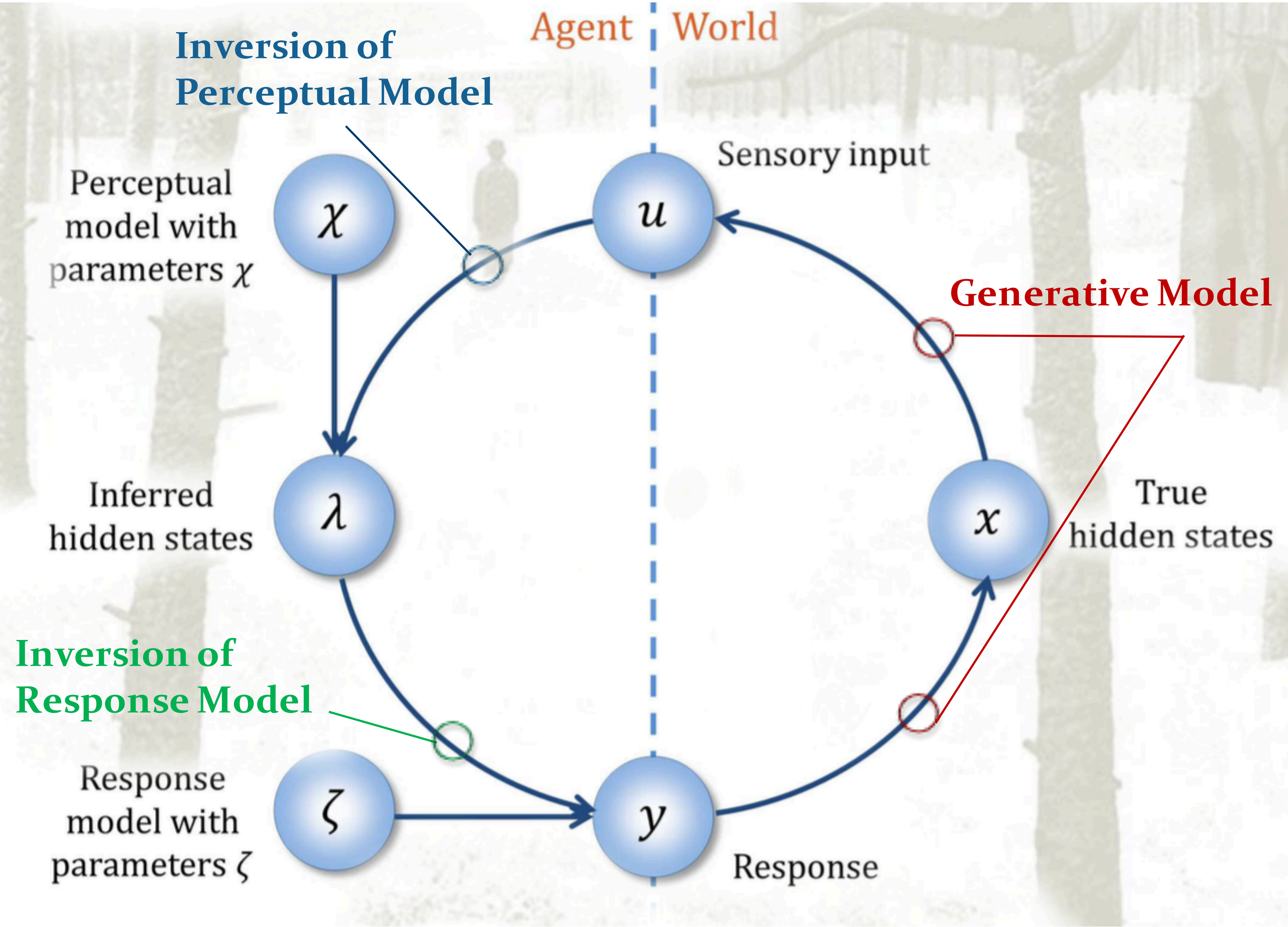


From perception to action



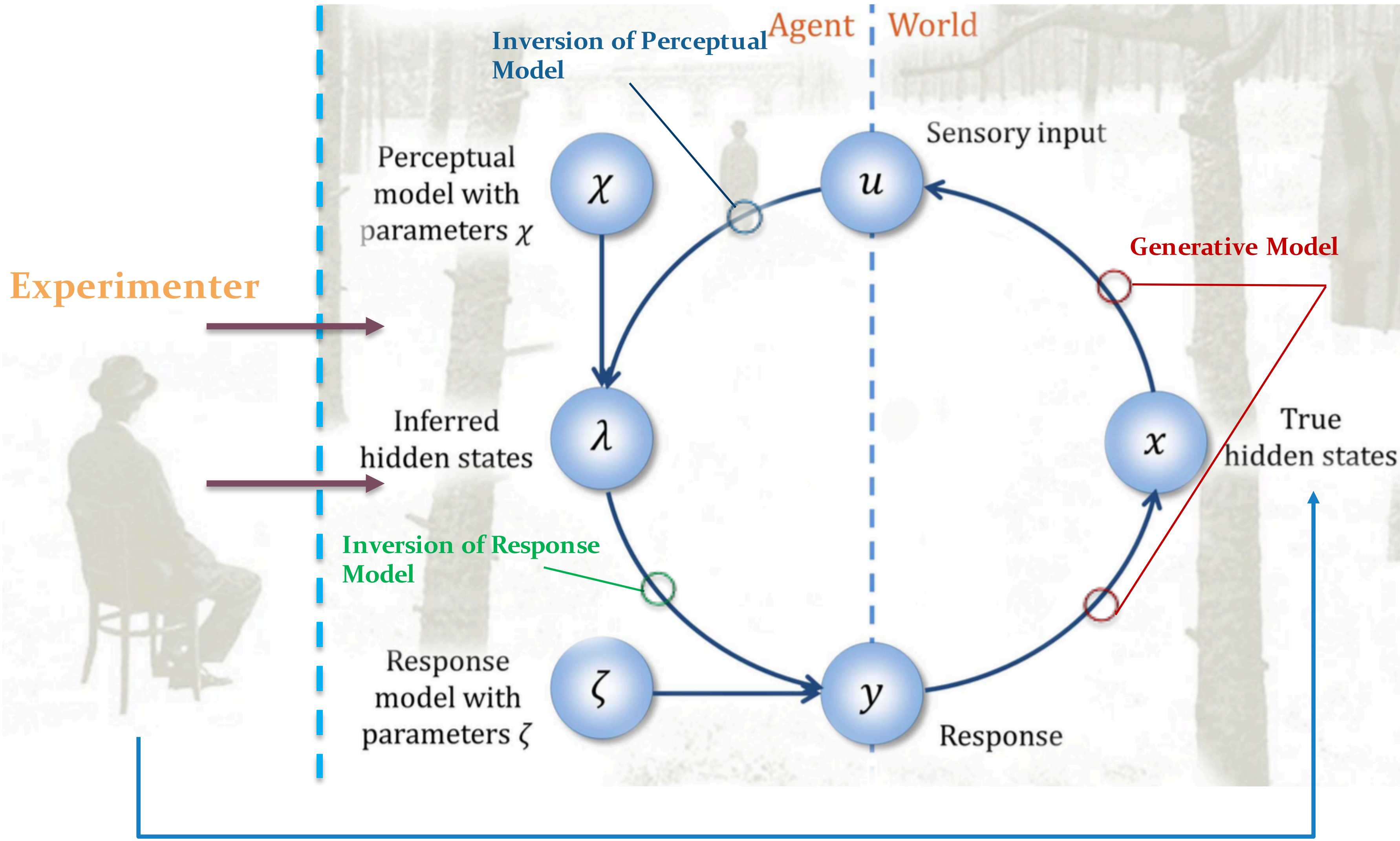
From perception to action

Translational Neuromodeling Unit



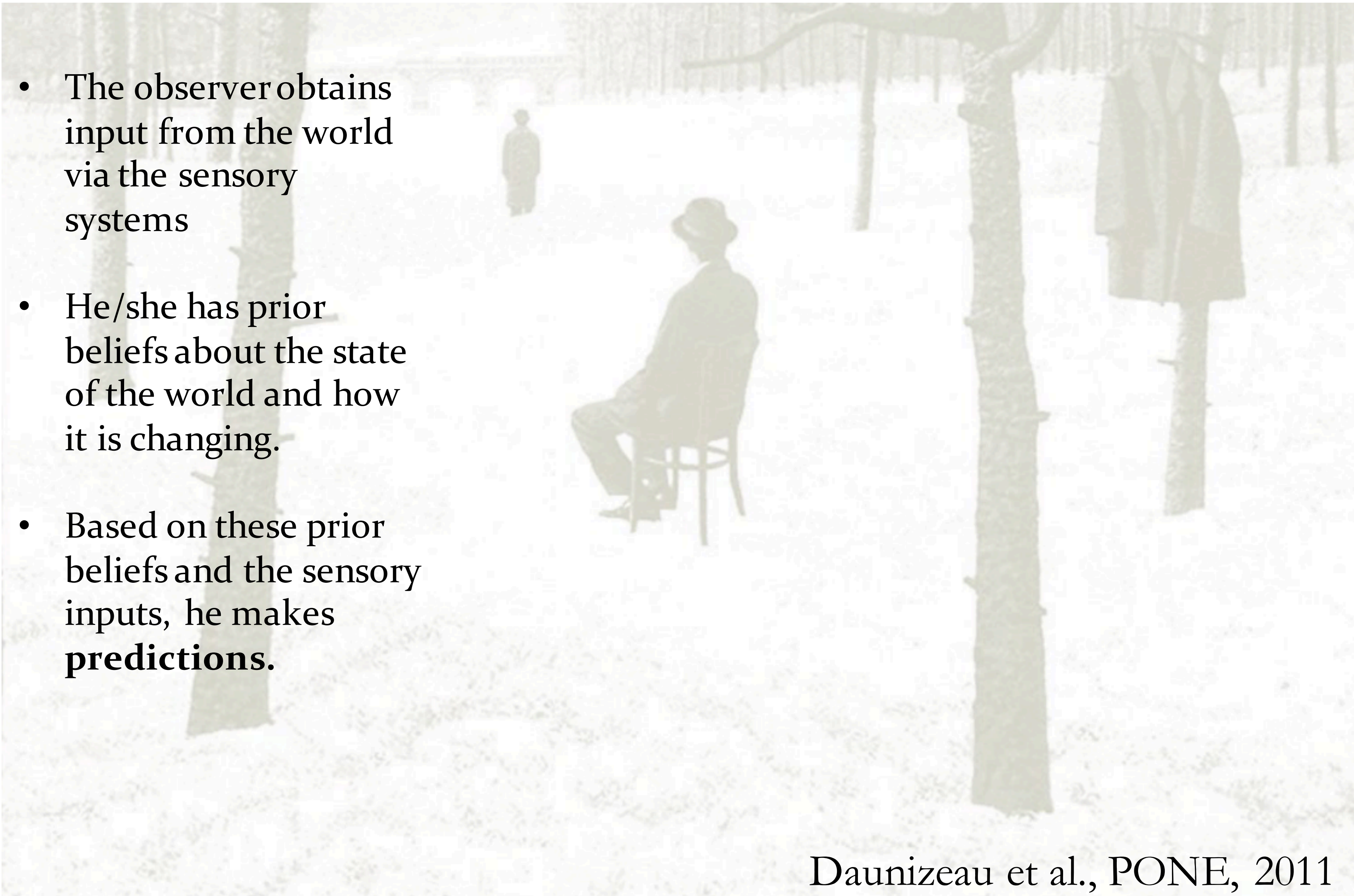
From perception to action to observation

Translational Neuromodeling Unit



Observing the observer

Translational Neuromodeling Unit



- The observer obtains input from the world via the sensory systems
- He/she has prior beliefs about the state of the world and how it is changing.
- Based on these prior beliefs and the sensory inputs, he makes **predictions.**

Daunizeau et al., PONE, 2011

Observing the observer

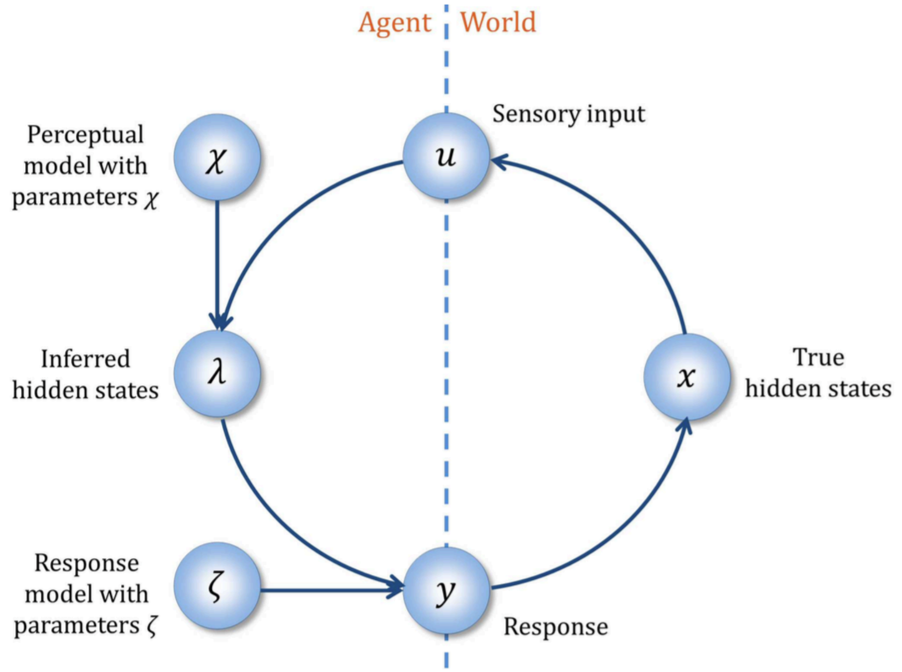
Translational Neuromodeling Unit

- The observer obtains input from the world via the sensory systems
- He/she has prior beliefs about the state of the world and how it is changing.
- Based on these prior beliefs and the sensory inputs, he makes **predictions.**

- As the experimenter, we want to infer on what the observer is thinking …
- But all we can observe is his/her behaviour.
- We invert the observer's beliefs from his/her behaviour: computational model

Daunizeau et al., PONE, 2011

From perception to action



- In behavioural tasks, we observe actions a
- How do we use them to infer on beliefs λ?
- Answer: we invert (estimate) a **response model**

Example of a simple response model

Translational Neuromodeling Unit

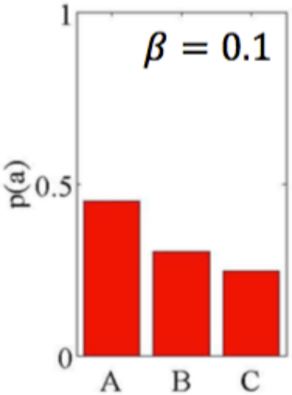
- Options A, B and C have values: $v_A = 8, v_B = 4, v_C = 2$

- We translate these values into action probabilities via a *Softmax* function:

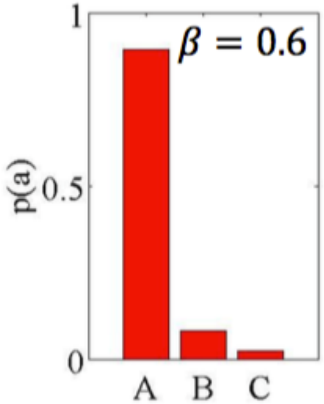
$$p(a = A) = \frac{e^{\beta v_A}}{e^{\beta v_A} + e^{\beta v_B} + e^{\beta v_C}}$$

- Parameter β determines sensitivity to value differences:

low β



high β

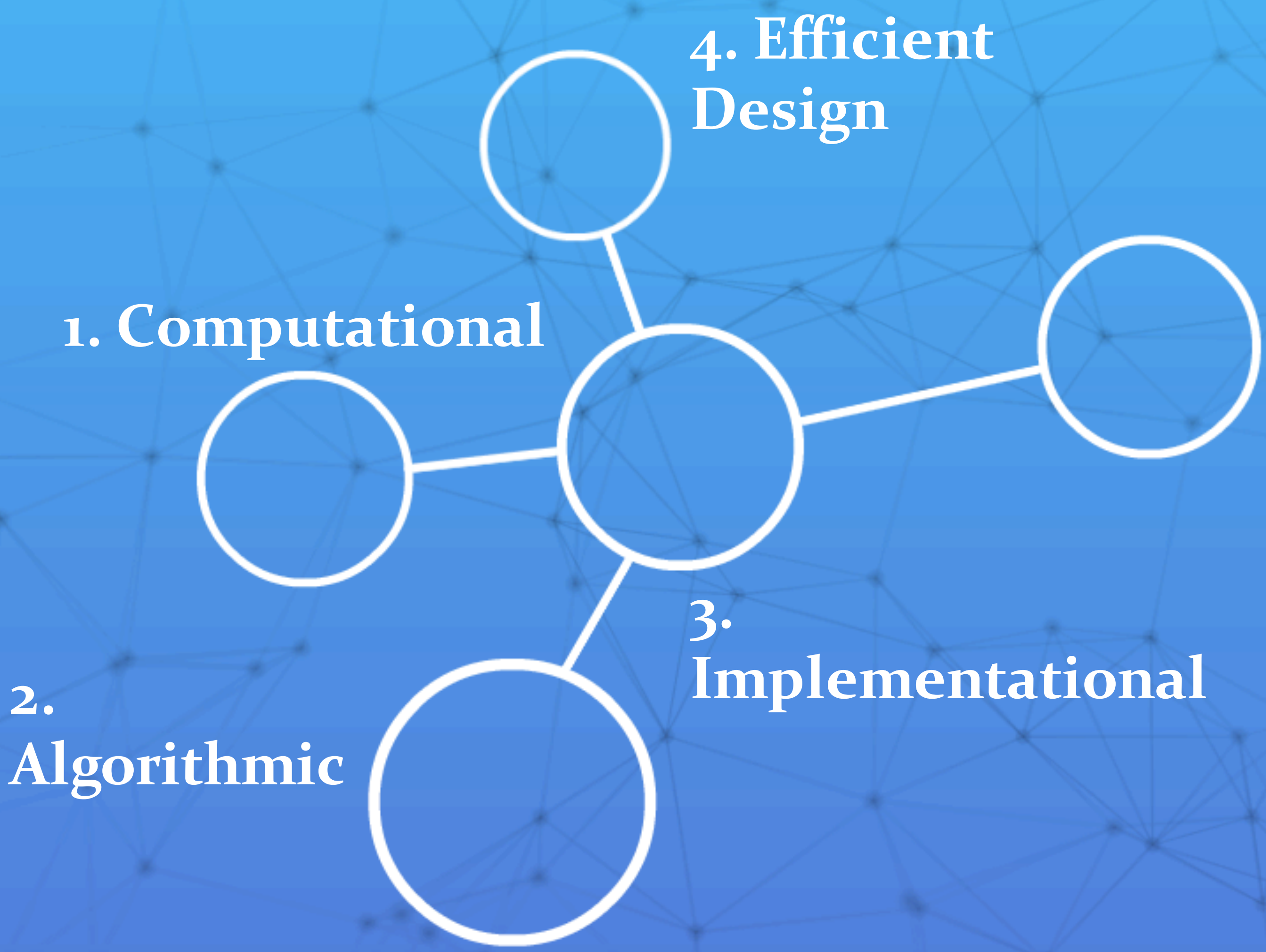


All the necessary ingredients

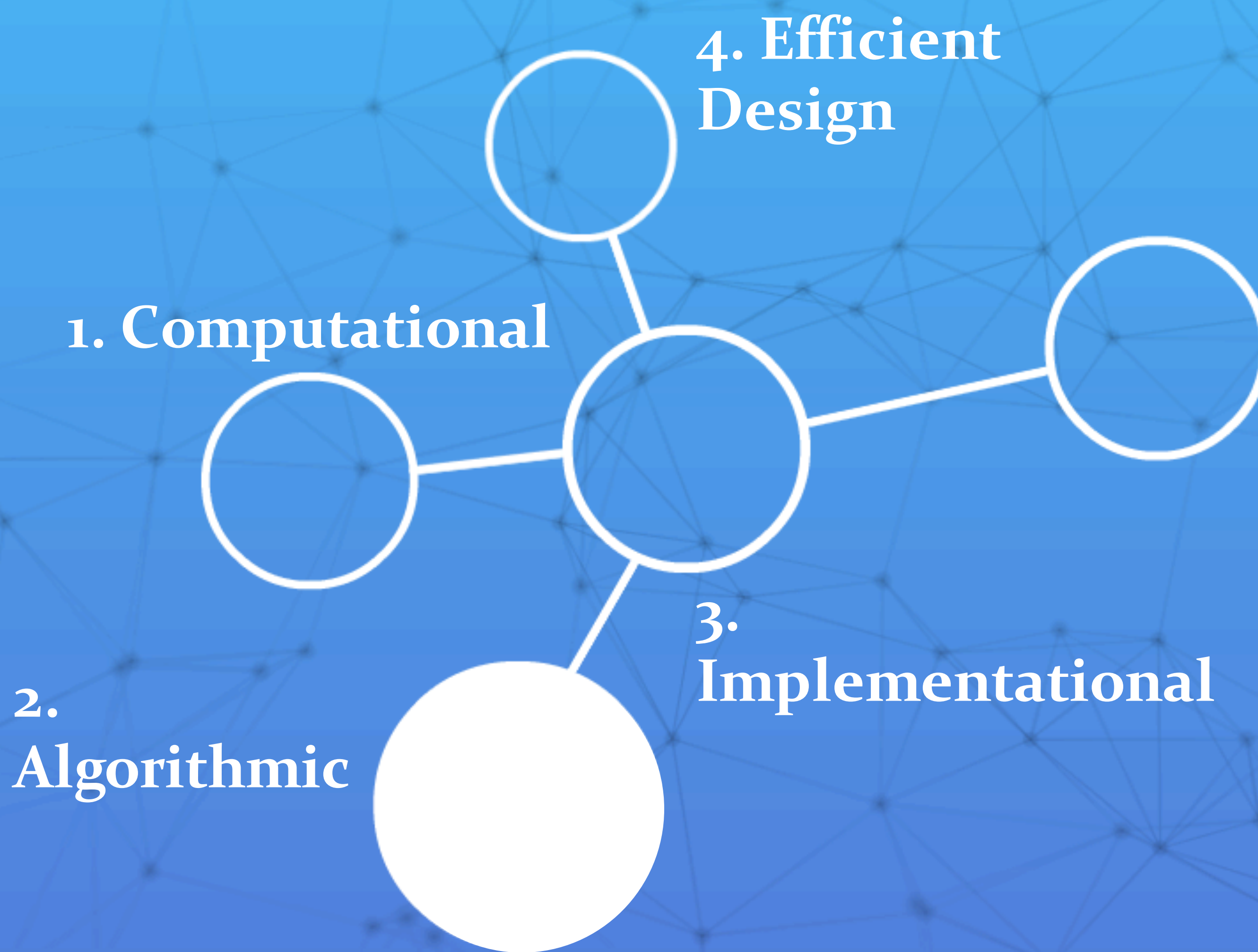
Translational Neuromodeling Unit

- Perceptual model (updates based on prediction errors)
- Value function (inferred state to action value)
- Response model (action value to response probability)

Outline


4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

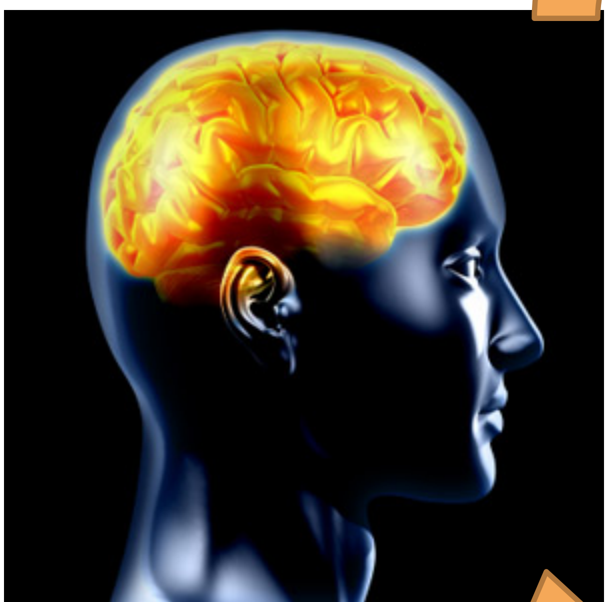
Outline


4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

Perception (learning) via hierarchical interactions

Translational Neuromodeling Unit

Top-down; Predictions



Filter

Update

Bottom-up; Sensations/Prediction Error

Hierarchical Bayesian Models

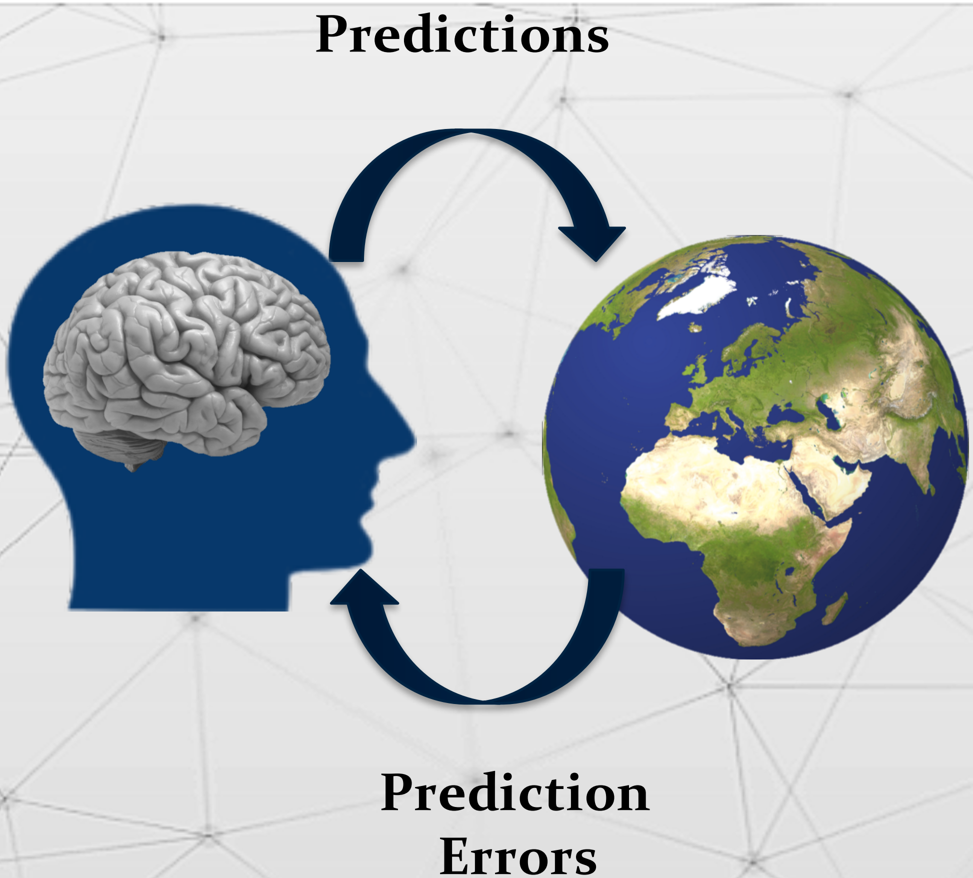
Translational Neuromodeling Unit

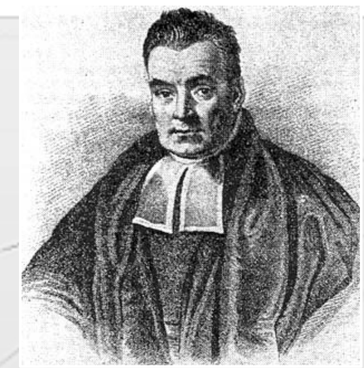
- Inference on the state of the world
- Beliefs are represented via probability distributions
  - Therefore: uncertainty (variance of the distribution) affects belief-updating
- Hierarchy of beliefs: state of the world and its volatility
- Efficient implementation in the brain promoted by evolutionary selection:
  - e.g. hierarchical architecture

Bayesian Models

The Bayesian Brain

Translational Neuromodeling Unit





- The brain is an inference machine
- Conceptualise beliefs as probability distributions
- Updates via Bayes' rule:

Prior Belief — Sensory Data

$$p(\Theta|y,m) = \frac{p(\Theta|m)p(y|\Theta,m)}{\int p(\Theta|m)p(y|\Theta,m)d\Theta}$$

Posterior Belief — Evidence

Bayesian Models

likelihood prior

$$p(x|u) = \frac{p(u|x)p(x)}{\int p(u|x')p(x')dx}$$

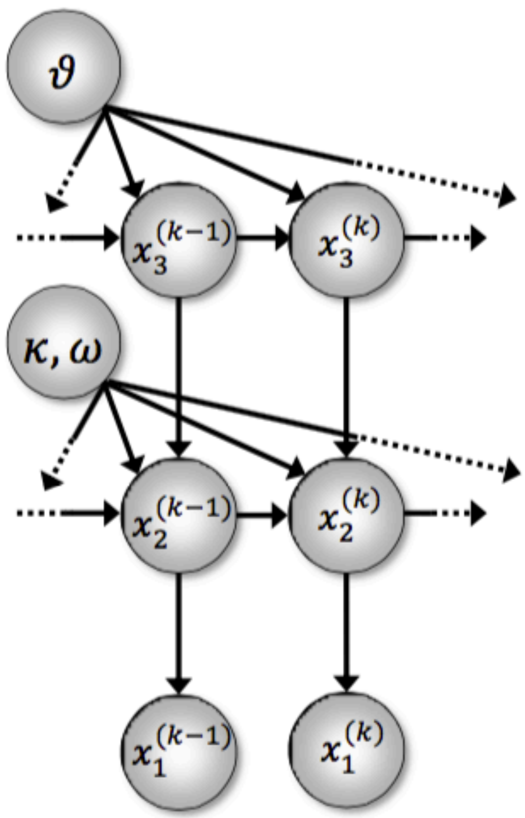
posterior

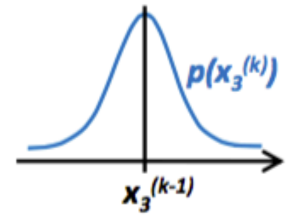
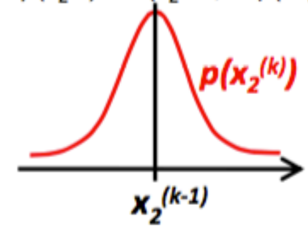
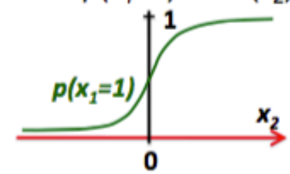
evidence

- In all but the simplest cases, the equation for the model evidence has no closed-form solutions.
- One way to deal with this is to introduce approximations.
- One possible and plausible approximation to the model evidence is variational free energy (cf. Friston, 2007; Feynman, 1972)

The hierarchical Gaussian filter (HGF): a computationally tractable model for individual learning under uncertainty

Translational Neuromodeling Unit



State of the world	Model	
Log-volatility x_3 of tendency	Gaussian random walk with constant step size ϑ	$p(x_3^{(k)}) \sim N(x_3^{(k-1)}, \vartheta)$ 
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	$p(x_1=1) = s(x_2)$ $p(x_1=0) = 1-s(x_2)$ 

Mathys et al., Frontiers, 2011

HGF: Variational inversion and update equations

Translational Neuromodeling Unit

- Inversion proceeds by introducing a mean field approximation and fitting quadratic approximations to the resulting variational energies.
- This leads to simple one-step update equations for the sufficient statistics (mean and precision) of the approximate Gaussian posteriors of the hidden states x_i.
- The updates of the means have the same structure as value updates in Rescorla-Wagner learning:

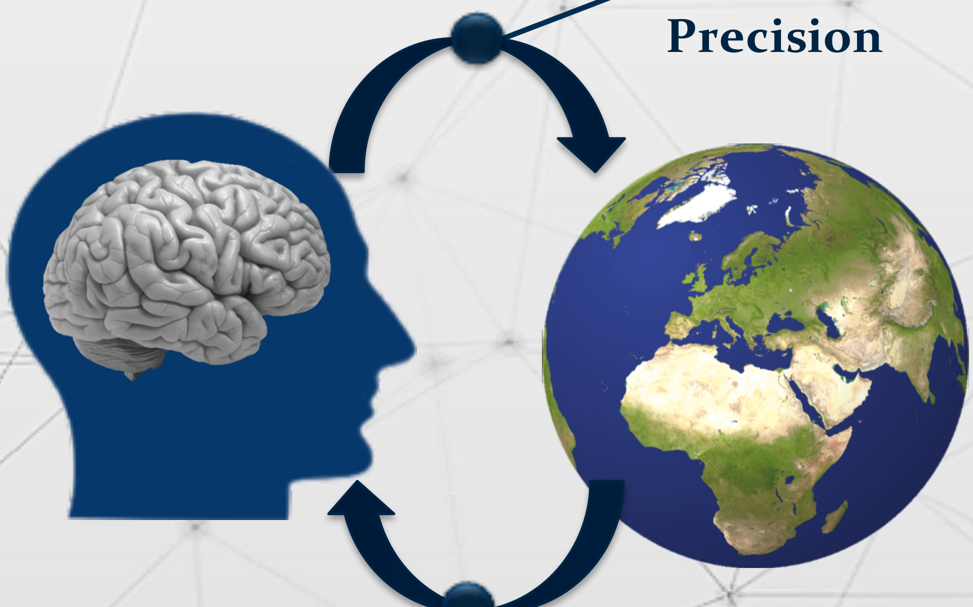
$$\Delta\mu_i = \frac{\hat{\pi}_{i-1}}{\pi_i}\delta_{i-1}$$

Prediction Error

Precisions determine the learning rate

Hierarchical Gaussian Filter

Translational Neuromodeling Unit



Mathys et al., *Frontiers Human Neurosci* 2011
Diaconescu et al., *PLoS Computational Biology,* 2014

Hierarchy

- Updates as precision-weighted prediction errors

$$\Delta\mu_i^{(k)} \propto \frac{\hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)}$$

Belief Update ($\Delta\mu_i^{(k)}$)

Sensory Precision ($\hat{\pi}_{i-1}^{(k)}$)

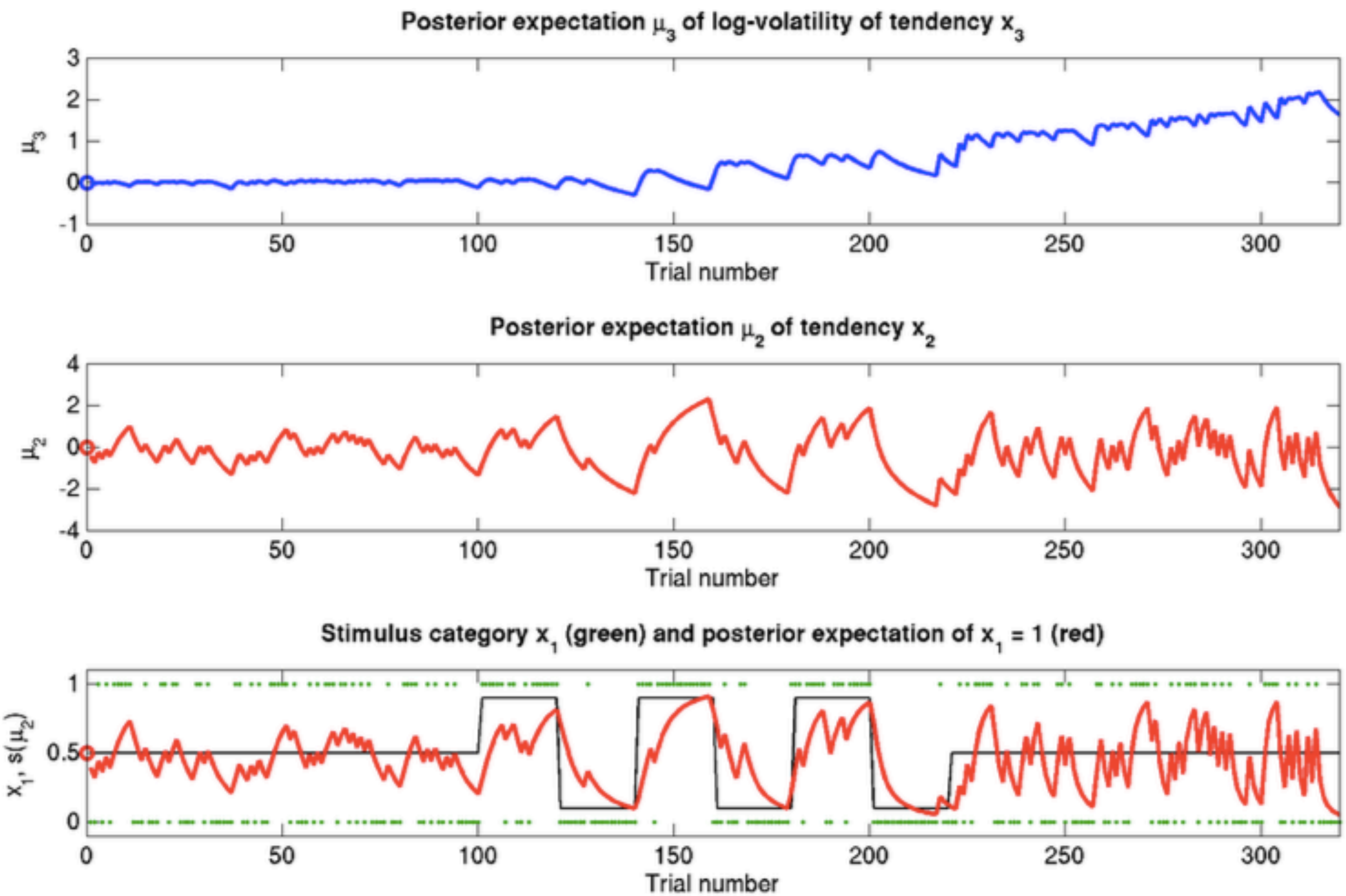
Belief Precision ($\pi_i^{(k)}$)

PE ($\delta_{i-1}^{(k)}$)

Hierarchical Learning

Translational Neuromodeling Unit

Simulations: $\vartheta = 0.5,\ \omega = -2.2,\ \kappa = 1.4$



HGF: Hierarchical Precision-weighted PEs

Translational Neuromodeling Unit

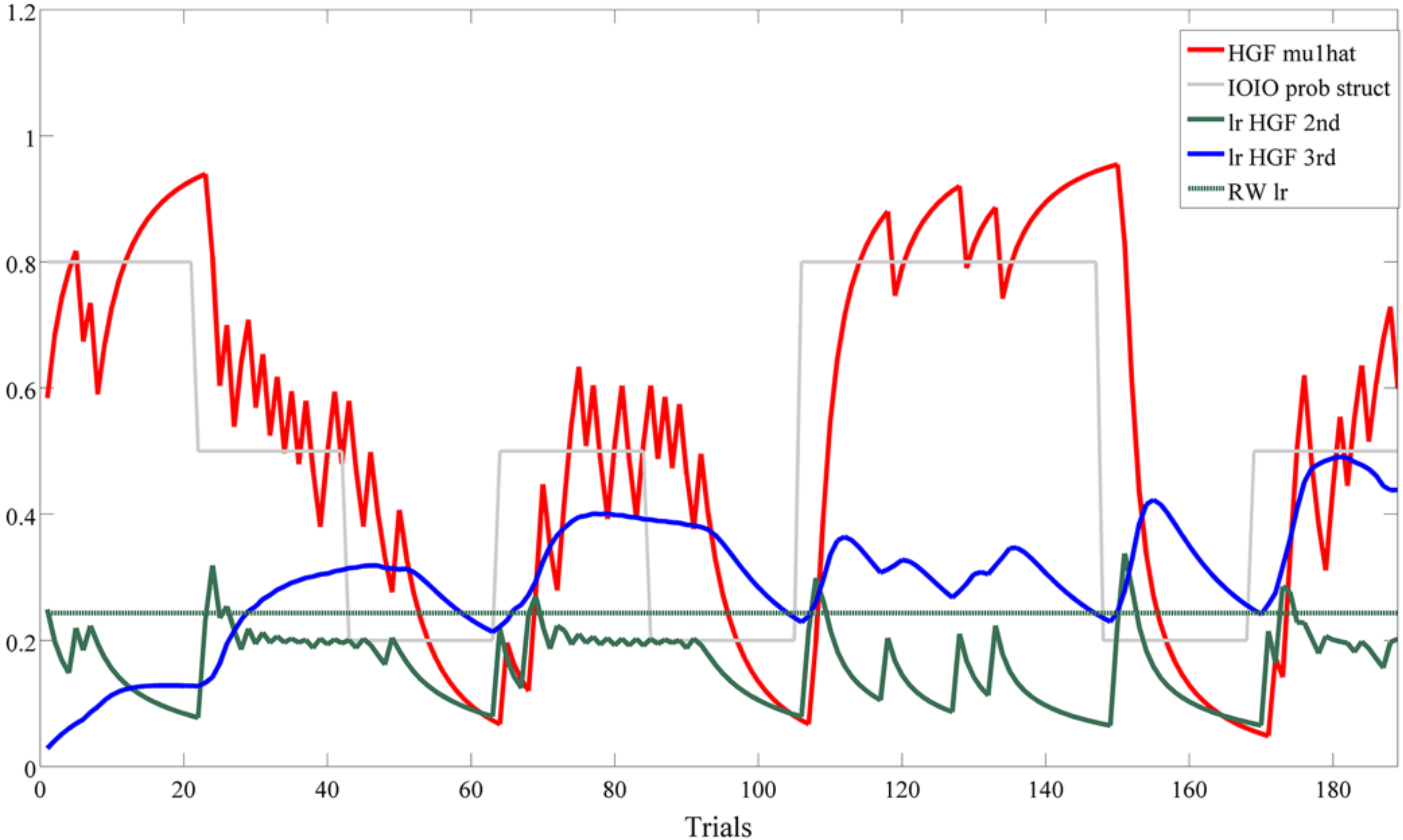
1. Value Update: $\Delta\mu_2 = \boxed{\frac{1}{\pi_2}} \cdot \delta_1$ where $\pi_2 = \hat{\pi}_2 + \frac{1}{\hat{\pi}_1}$

2. Volatility Update: $\Delta\mu_3 = \boxed{\frac{\kappa}{2} \cdot \frac{1}{\pi_3} \cdot w_2} \cdot \delta_2$

RL models: $\Delta v = \boxed{\alpha} \cdot \delta$

HGF: Dynamic Learning Rates

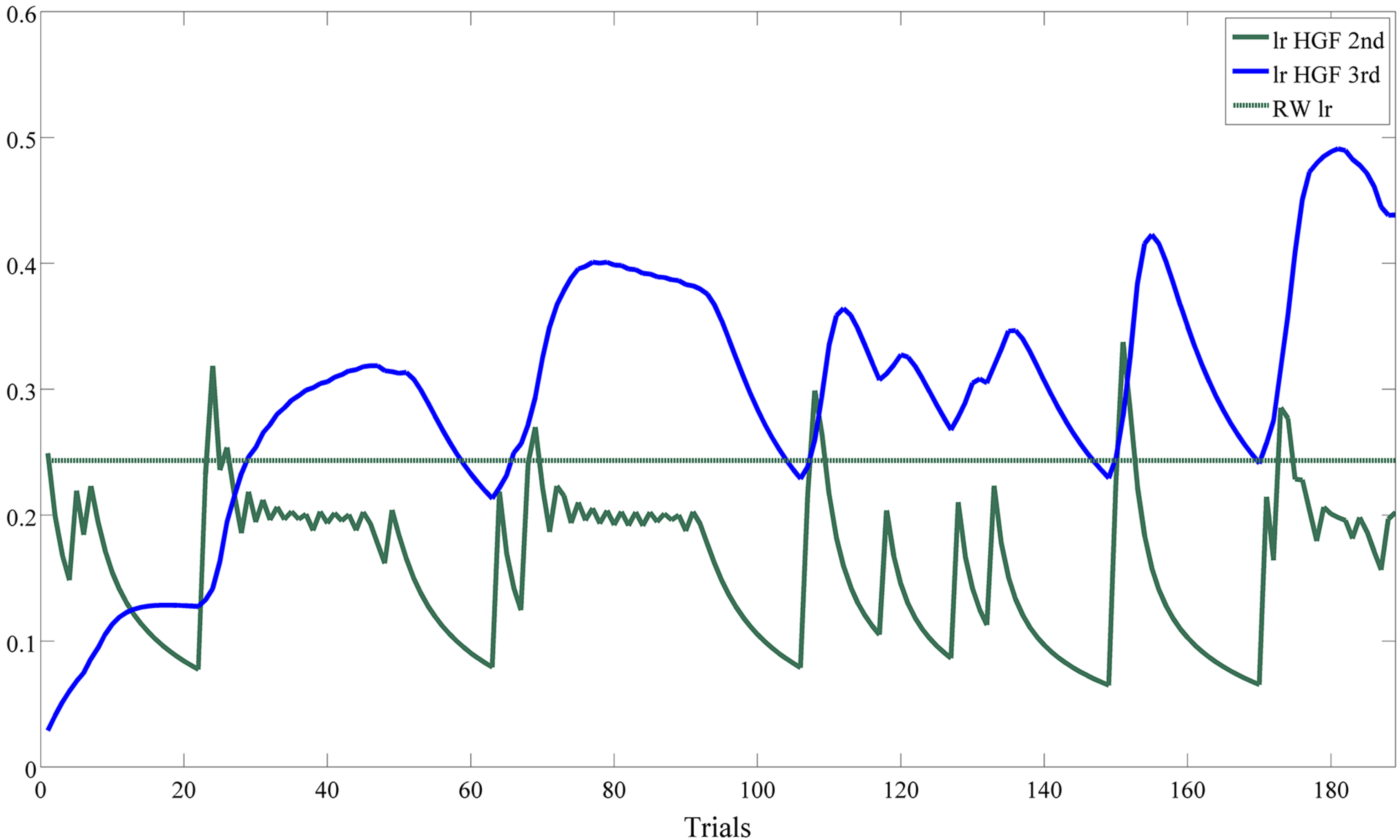
Translational Neuromodeling Unit



Diaconescu et al., 2014

HGF: Dynamic Learning Rates

Translational Neuromodeling Unit



Diaconescu et al., 2014

Which model is better?

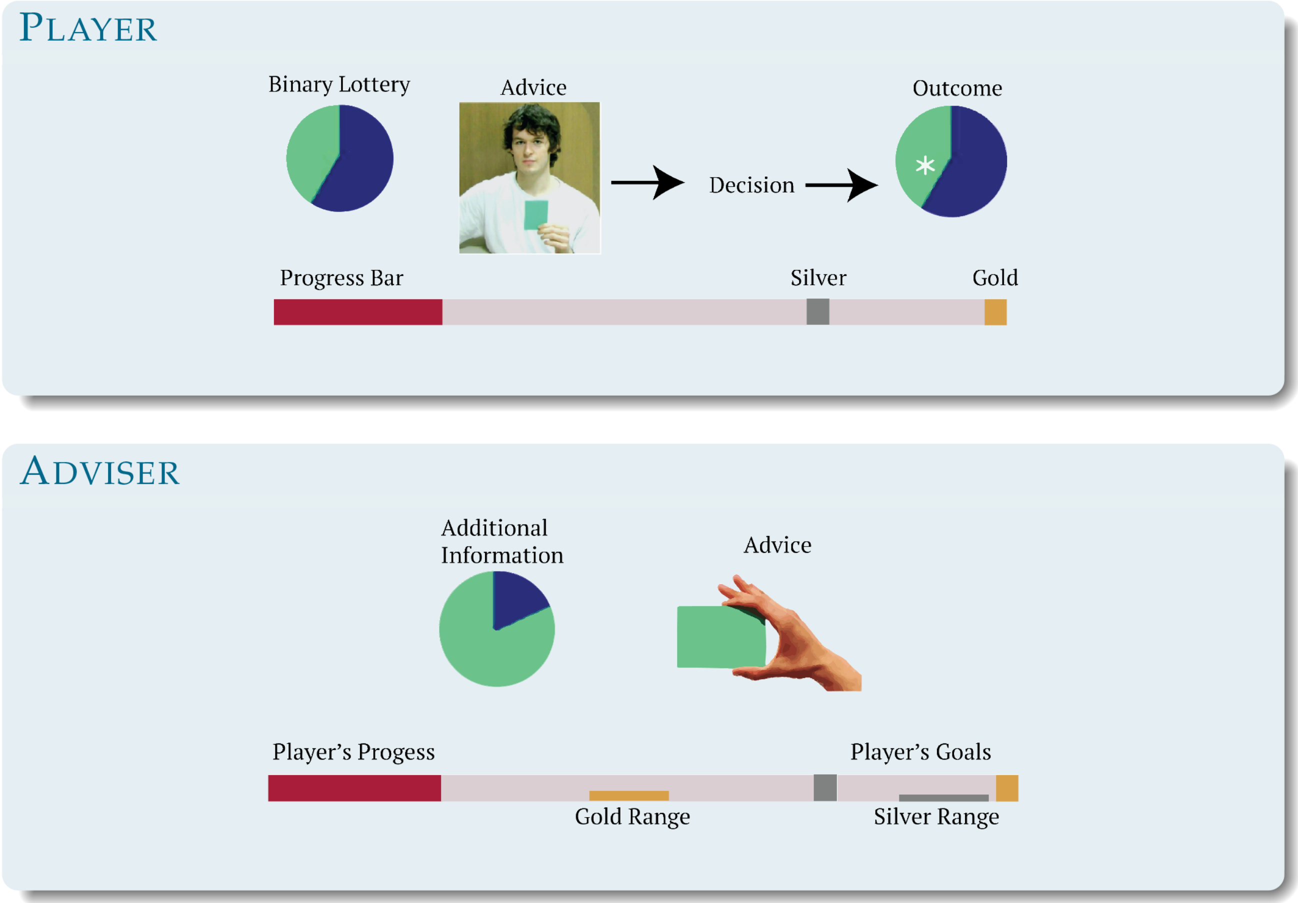
Translational Neuromodeling Unit

- Reinforcement Learning?
- Hierarchical Bayesian Model?

Model Comparison: An example

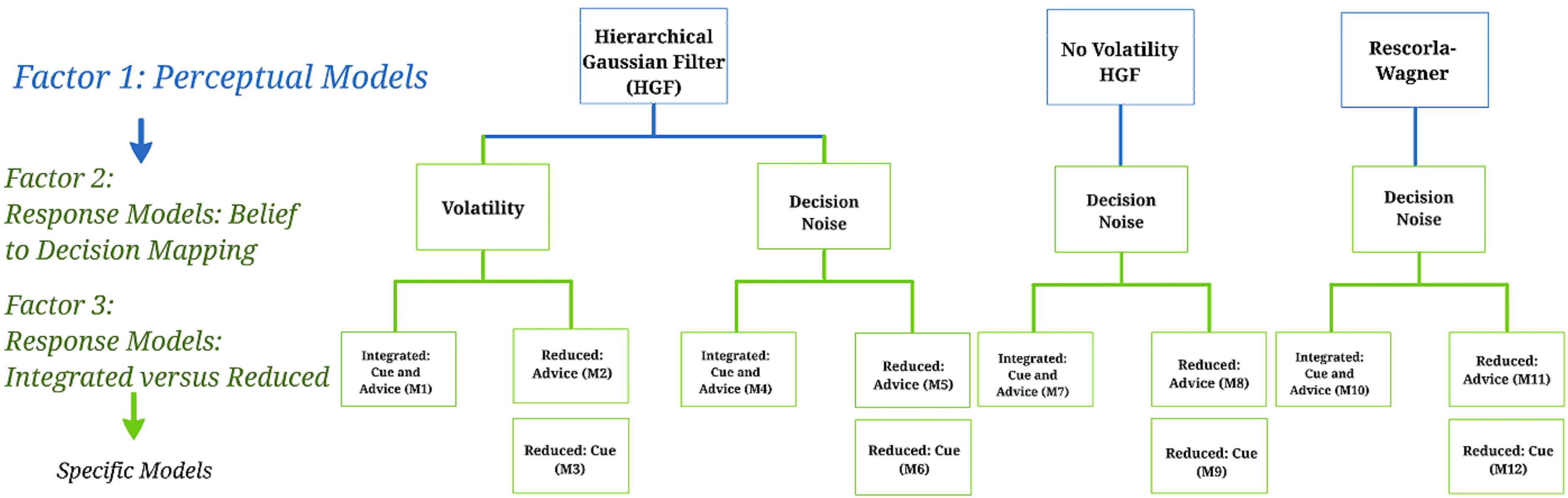
Translational Neuromodeling Unit

- Advice-Taking Task:



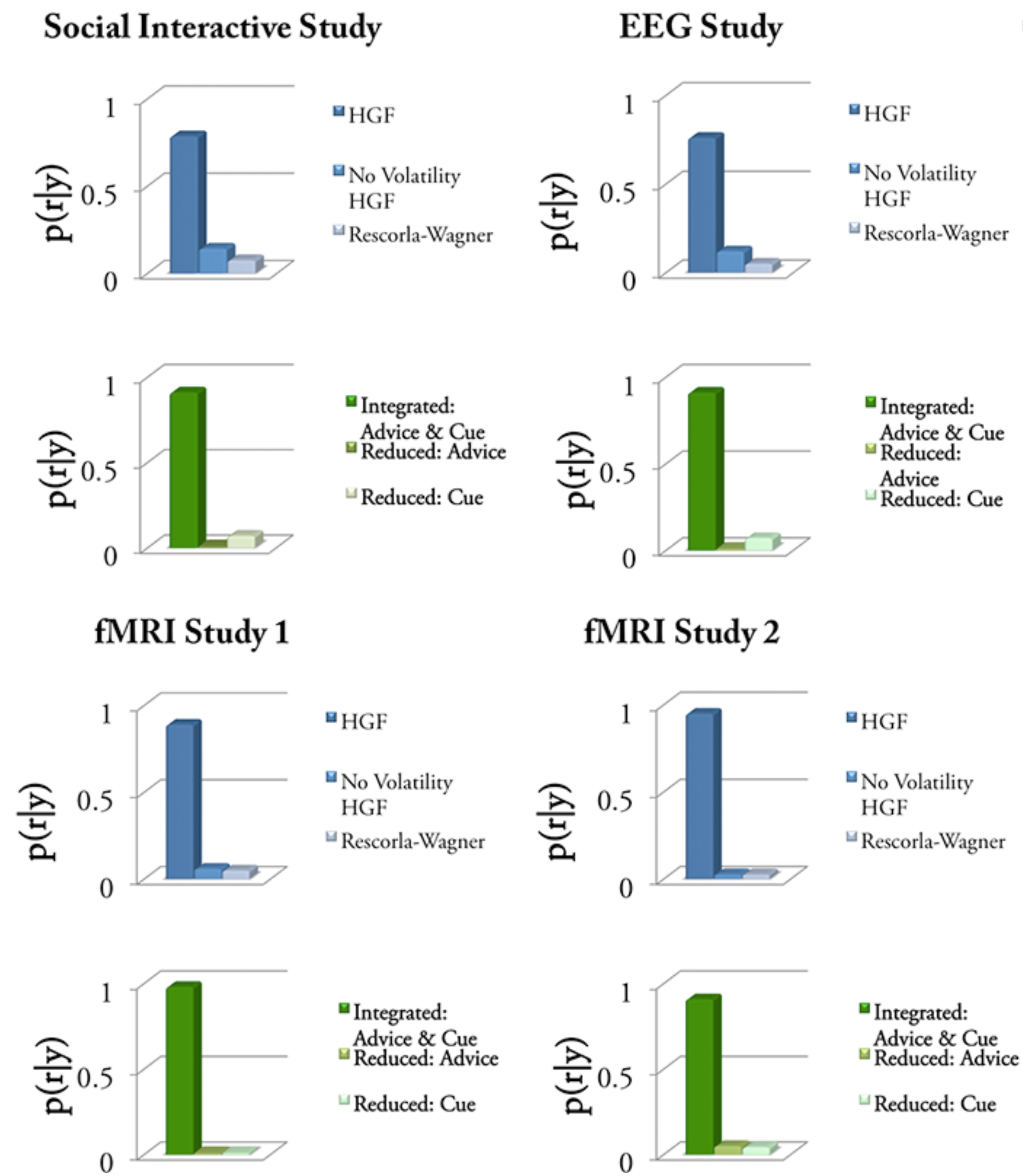
Diaconescu et al., *PLoS Computational Biology,* 2014

Model Space

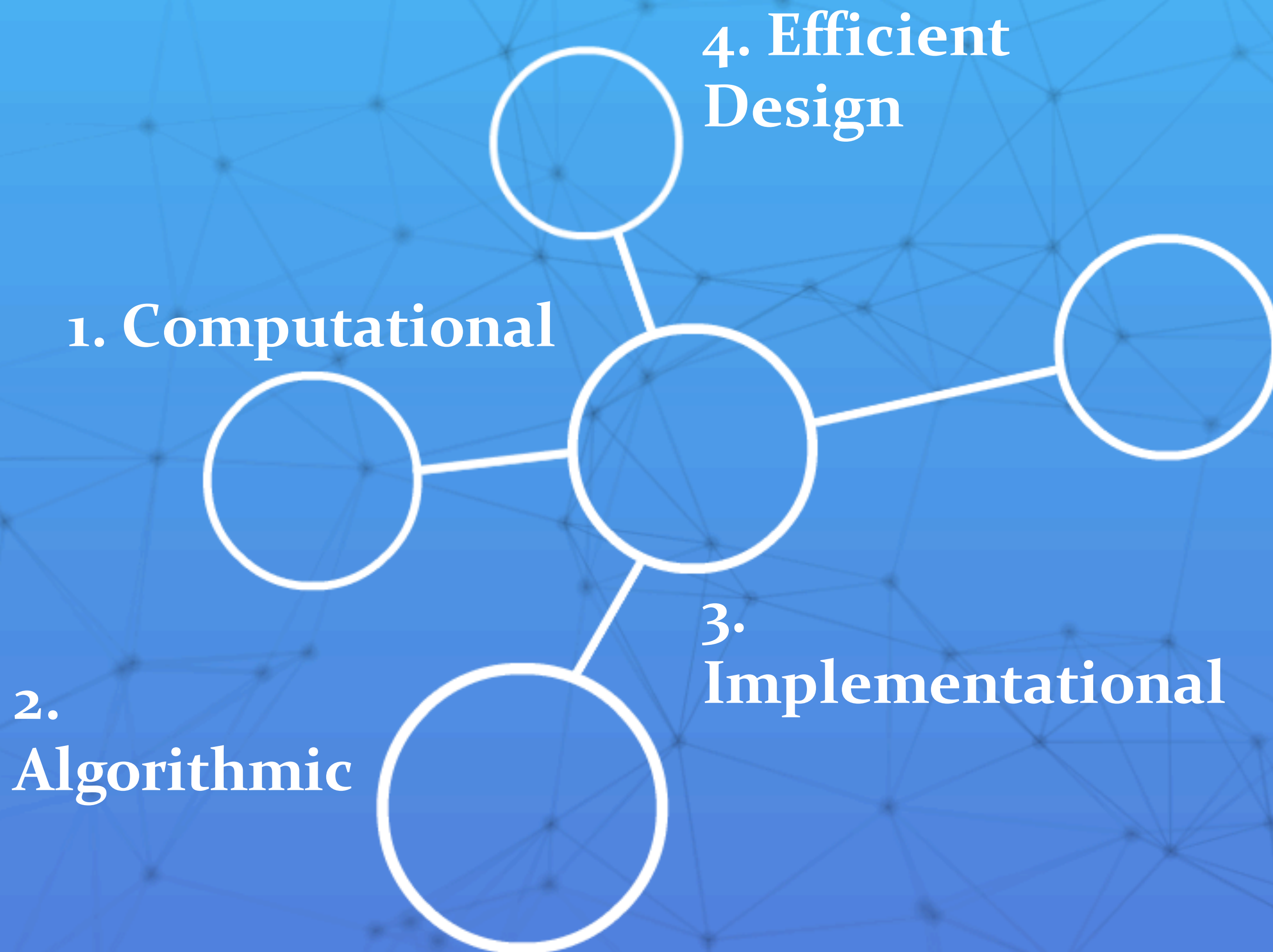


Diaconescu et al., *PLoS Computational Biology,* 2014

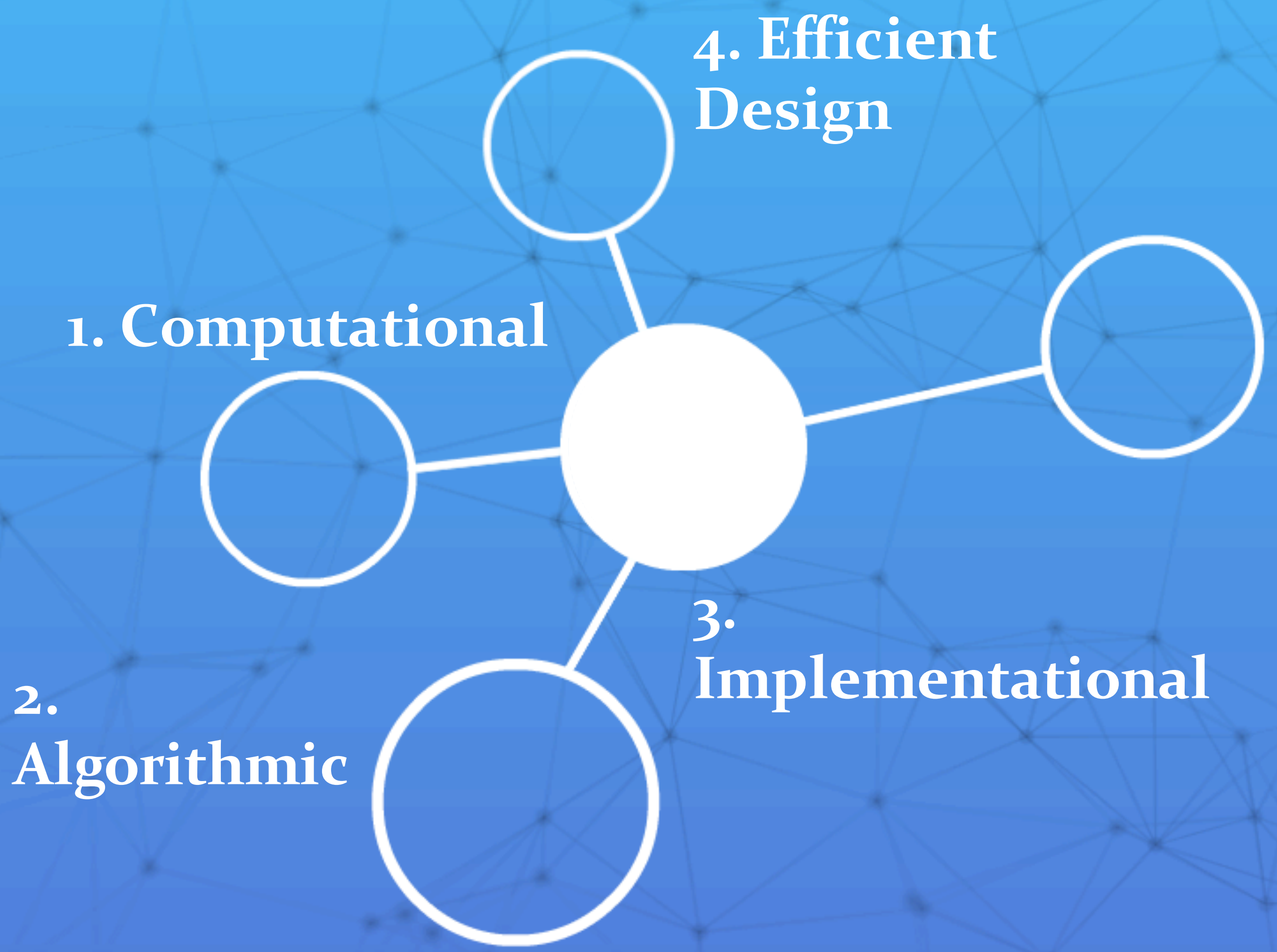
Winning model



Outline


4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

Outline


4. Efficient Design
1. Computational
3. Implementational
2. Algorithmic

Model-based fMRI: The advantage

Translational Neuromodeling Unit

The question event-related/block designs answer:

- Where in the brain do particular experimental conditions elicit BOLD responses?

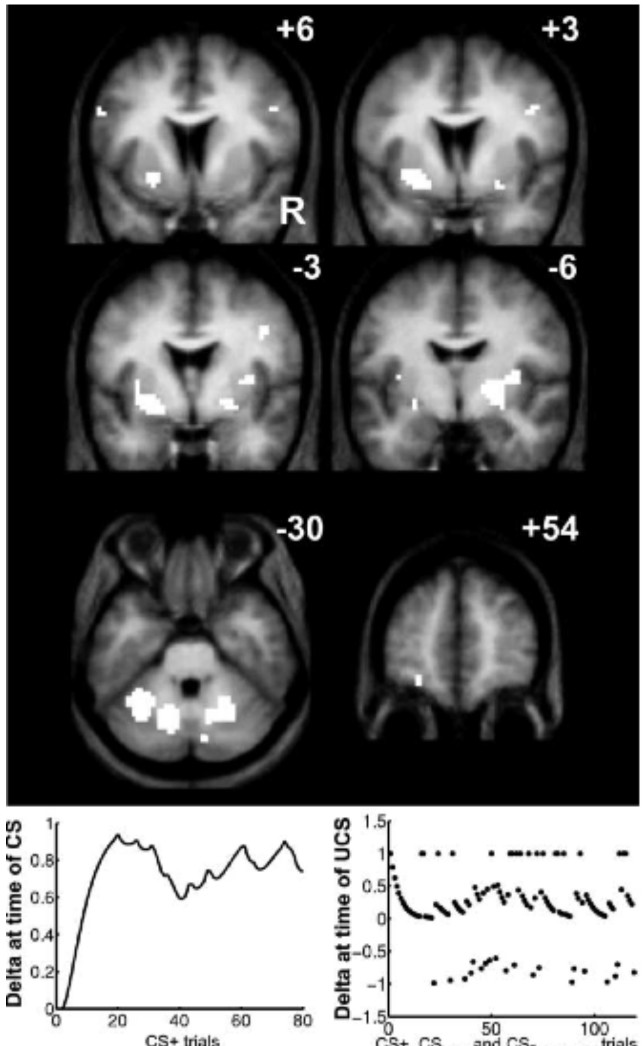
The question model-based fMRI answers:

- How (i.e., by activation of which areas) does the brain implement a particular cognitive process?

It is able to do so because its regressors correspond to particular cognitive processes instead of experimental conditions.

Example of a simple learning model

Translational Neuromodeling Unit



- Pavlovian conditioning:
  - abstract visual stimuli paired with sweet/neutral taste

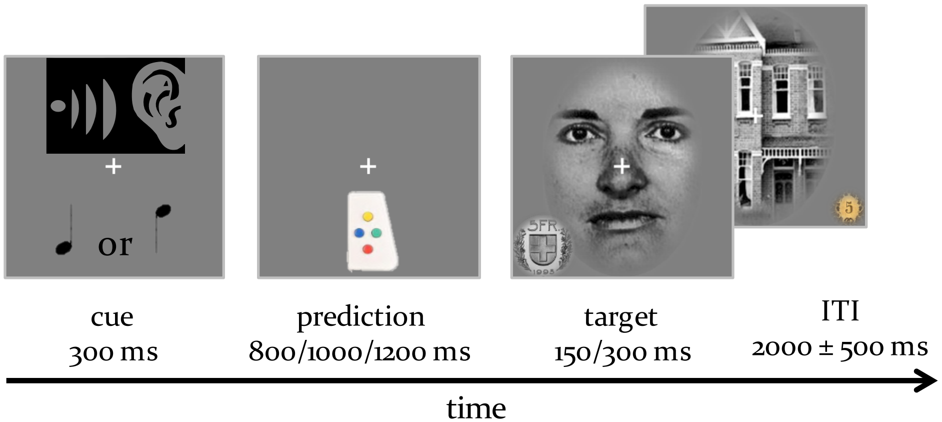
$$\mu^{(k)} = \mu^{(k-1)} + \alpha(u^{(k)} - \mu^{(k-1)})$$

- Signed PE with a fixed learning rate:
  - striatum, OFC, and cerebellum

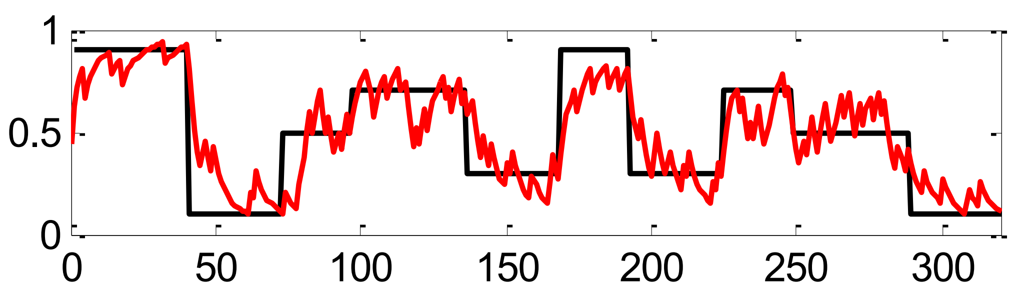
O'Doherty et al., *Neuron*, 2003

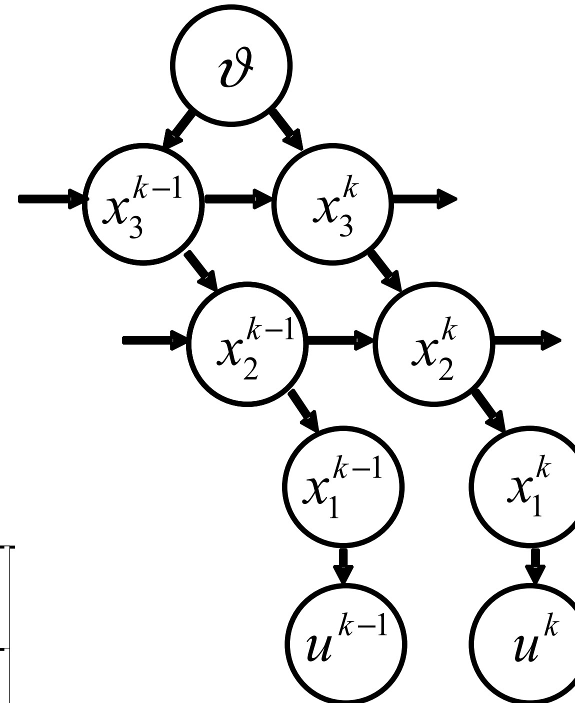
Application of the HGF: Sensory Learning

Translational Neuromodeling Unit



Changes in cue strength (black), and posterior expectation of visual category (red)



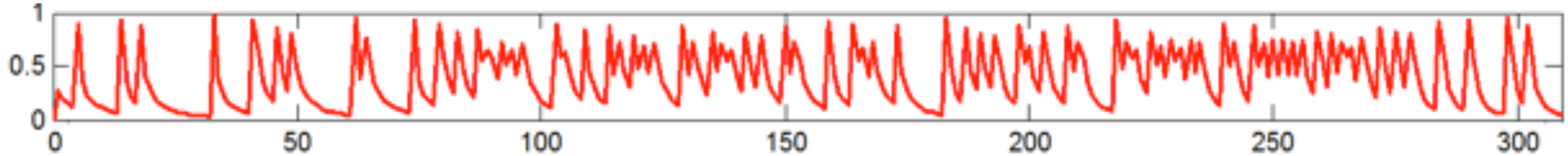


Iglesias et al., *Neuron*, 2013

Application of the HGF: Two types of PEs

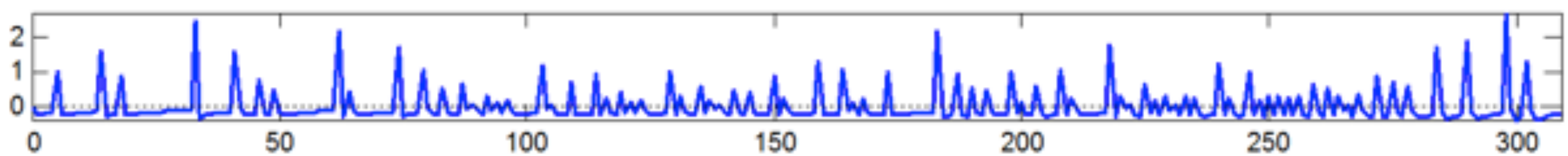
Translational Neuromodeling Unit

1. Outcome PE



ϵ_2

2. Cue-Outcome Contingency PE



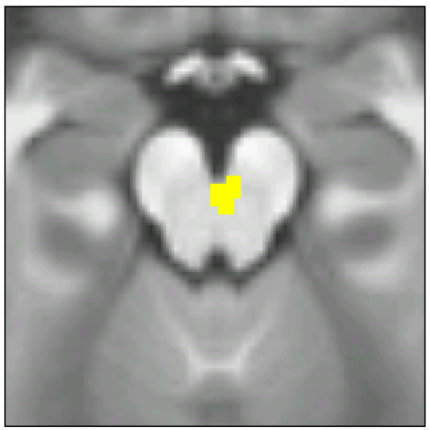
ϵ_3

Iglesias et al., *Neuron*, 2013

Application of the HGF: Representation of precision-weighted PEs

Translational Neuromodeling Unit

1. Outcome PE

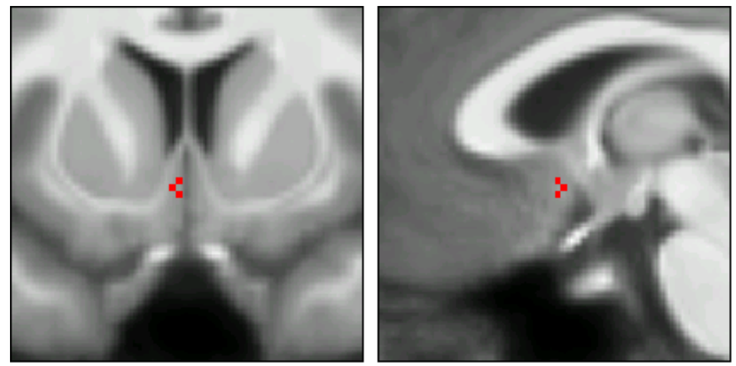


z = -18

- right VTA

Dopamine

2. Probability PE



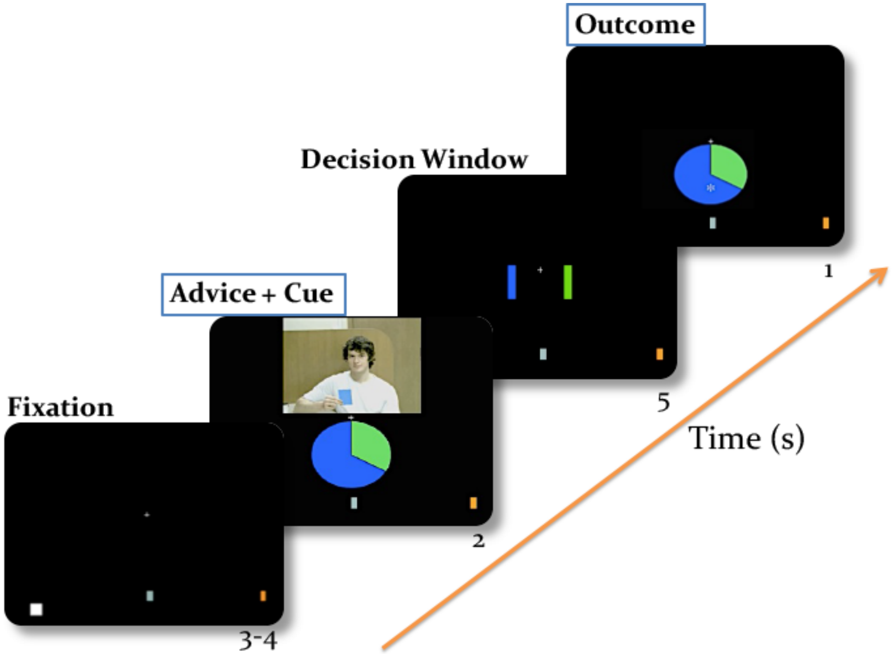
- left basal forebrain

Acetycholine

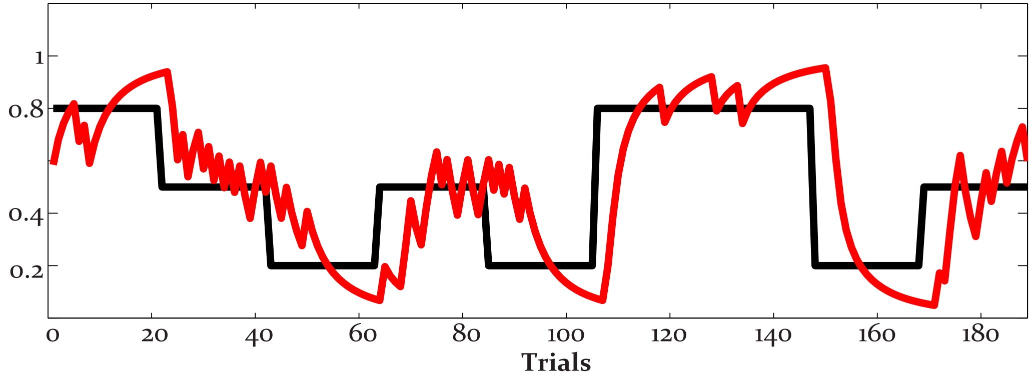
Iglesias et al., *Neuron*, 2013

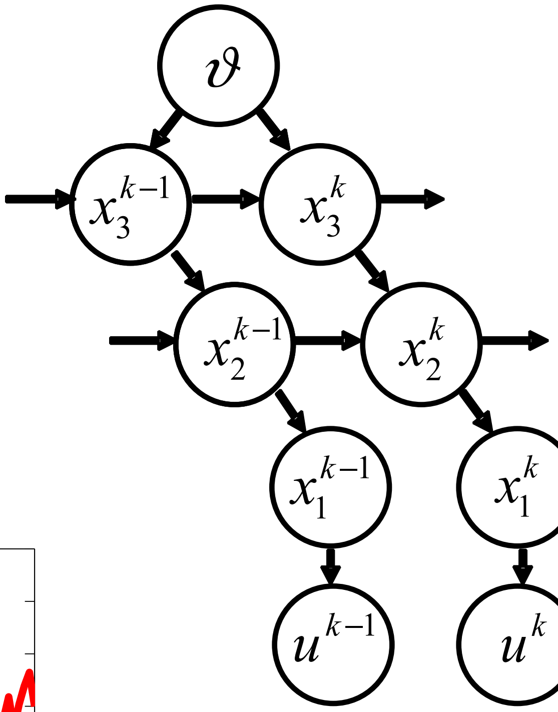
Application of the HGF: Social Learning

Translational Neuromodeling Unit



Changes in advice accuracy (black), and posterior expectation of adviser fidelity (red)



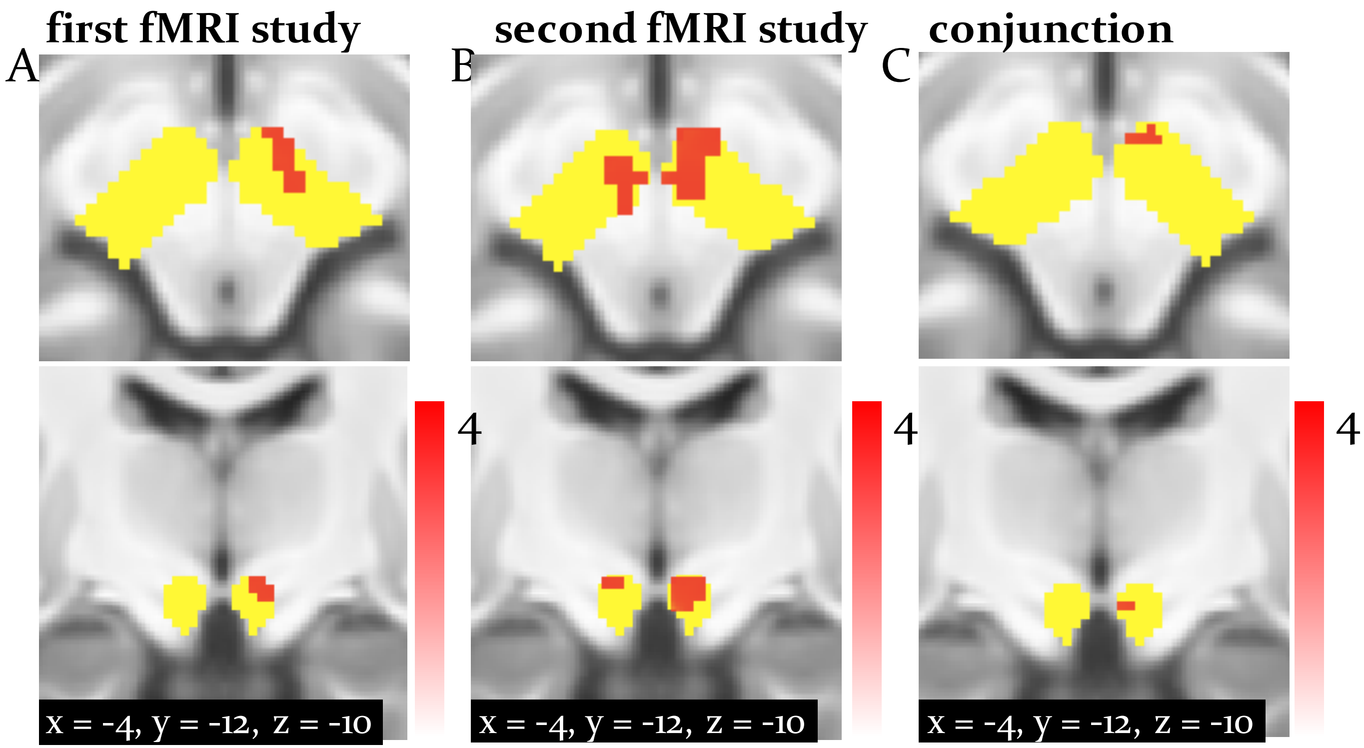


Diaconescu et al., *SCAN*, 2016

Representation of precision-weighted PEs in the social domain

Translational Neuromodeling Unit

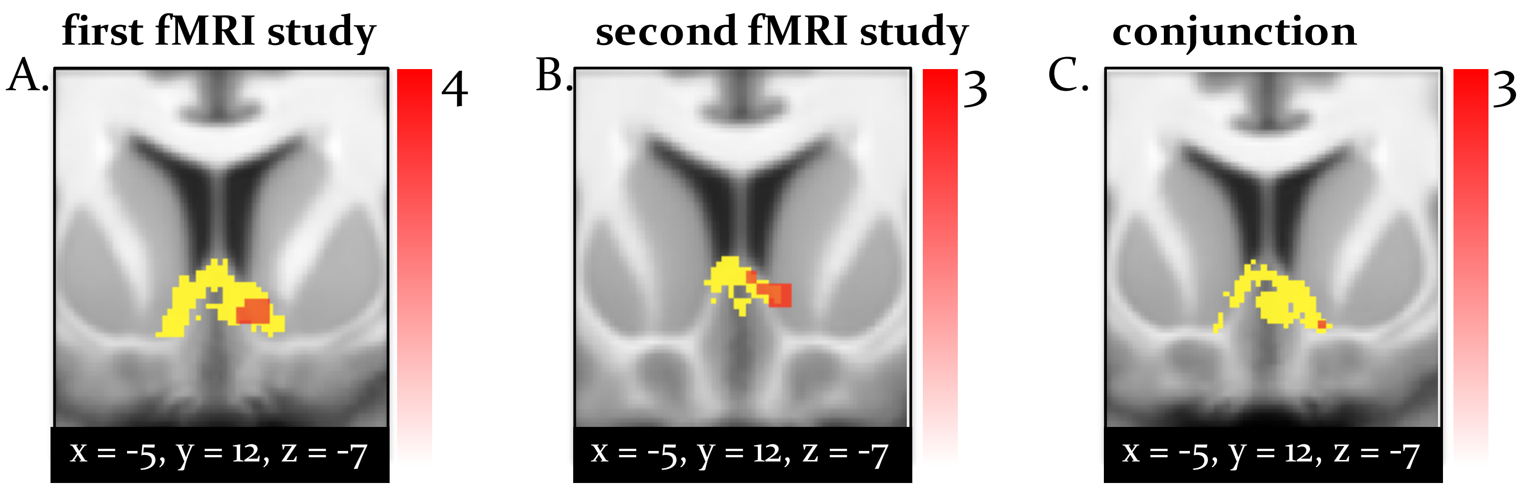
1. Outcome PE



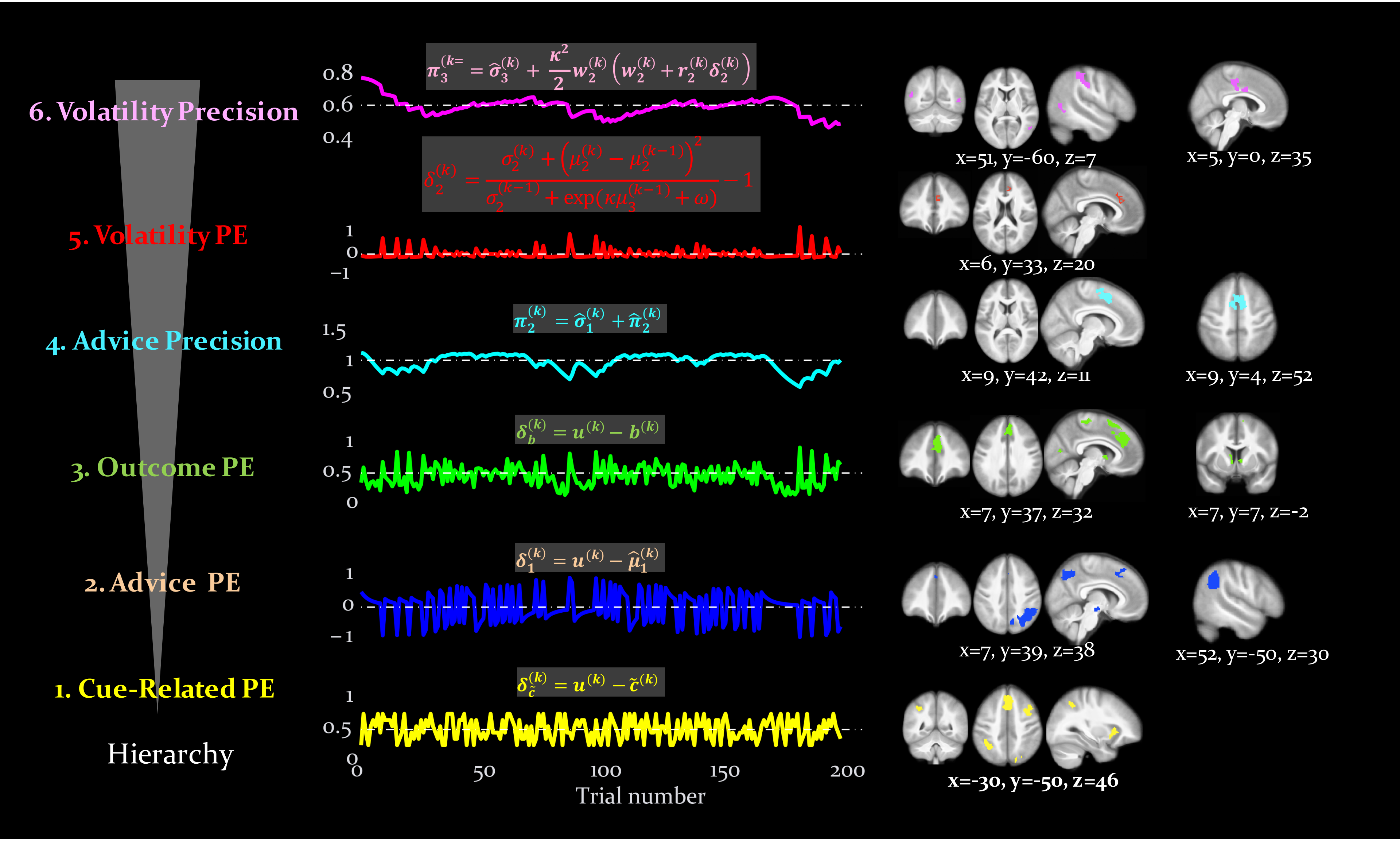
Diaconescu et al., *SCAN*, 2016

Representation of precision-weighted PEs in the social domain

2. Probability PE



Diaconescu et al., *SCAN*, 2016

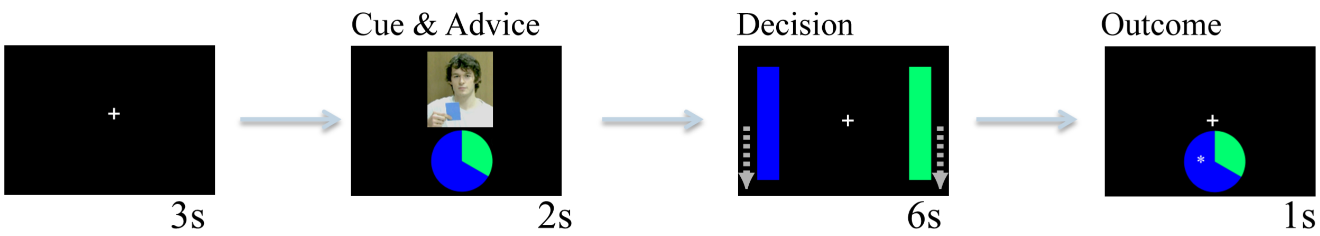


Diaconescu et al., *In Prep.*

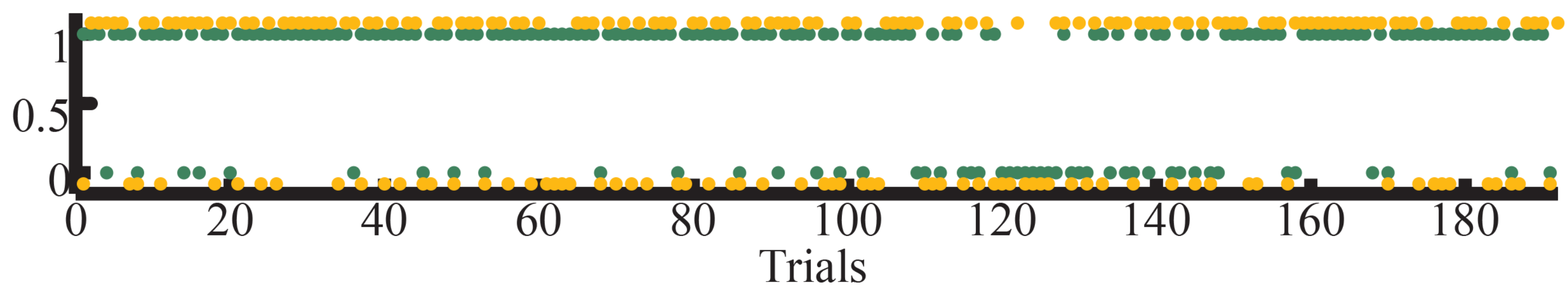
How do we construct regressors that correspond to cognitive processes and use them in SPM?

Translational Neuromodeling Unit

1. Pass individual subject trial history into SPM:



Response y (orange=1 advice was taken), input u (green=1 advice was accurate)



How do we construct regressors that correspond to cognitive processes and use them in SPM?

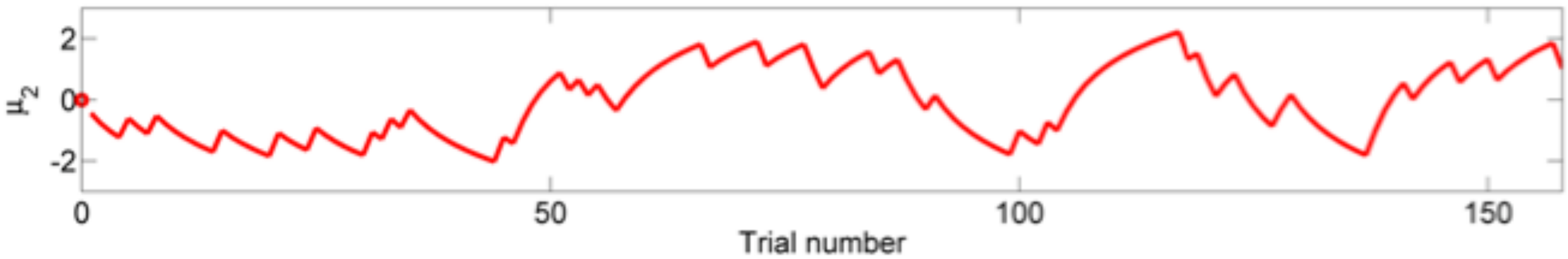
Translational Neuromodeling Unit

2. Estimated subject-by-subject model parameters:
   - Model Inversion:

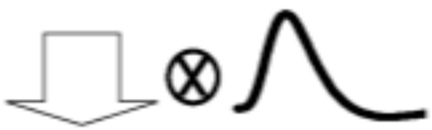
```
runnning model/param combination 4 of 546
Irregular trials: none
Ignored trials: none
Irregular trials: none

Optimizing...

Calculating the negative free energy...

Results:
     mu2_0: 1.0665
     sa2_0: 1.4966
     mu3_0: 1
     sa3_0: 1
        ka: 0
        om: -10
        th: 1.0000e-18
         p: [1.0665 1.4966 1 1 0 -10 1.0000e-18]
    ptrans: [1.0665 0.4032 1 0 -22.3327 -10 -34.5388]

       ze1: 0.8816
       ze2: 48.0000
         p: [0.8816 48.0000]
    ptrans: [2.0073 3.8712]

Negative free energy F: -82.9603
```

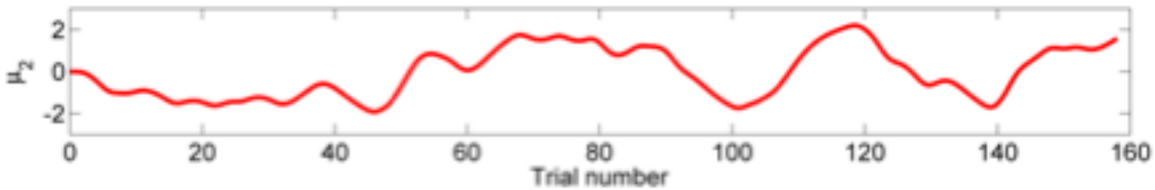
How do we construct regressors that correspond to cognitive processes and use them in SPM?

3. Generate model-based time-series:



3. Convolve them with HRF:



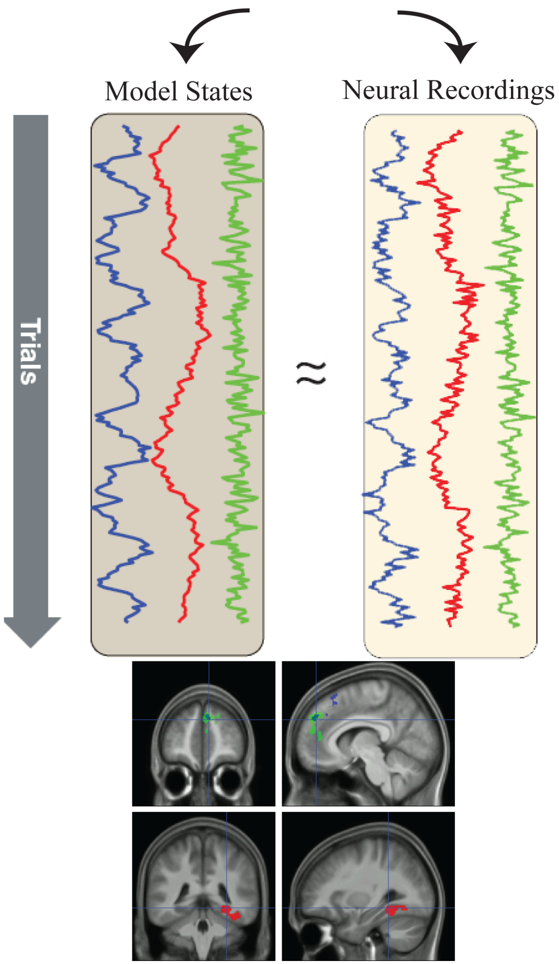


Adapted from O'Doherty et al., 2007

How do we construct regressors that correspond to cognitive processes and use them in SPM?

5. Construct your GLM:

$$\mu_1^k = \mu_1^{(k-1)} + \sigma \cdot \delta$$



Adapted from Behrens et al., 2010

Estimate: single subject

Translational Neuromodeling Unit

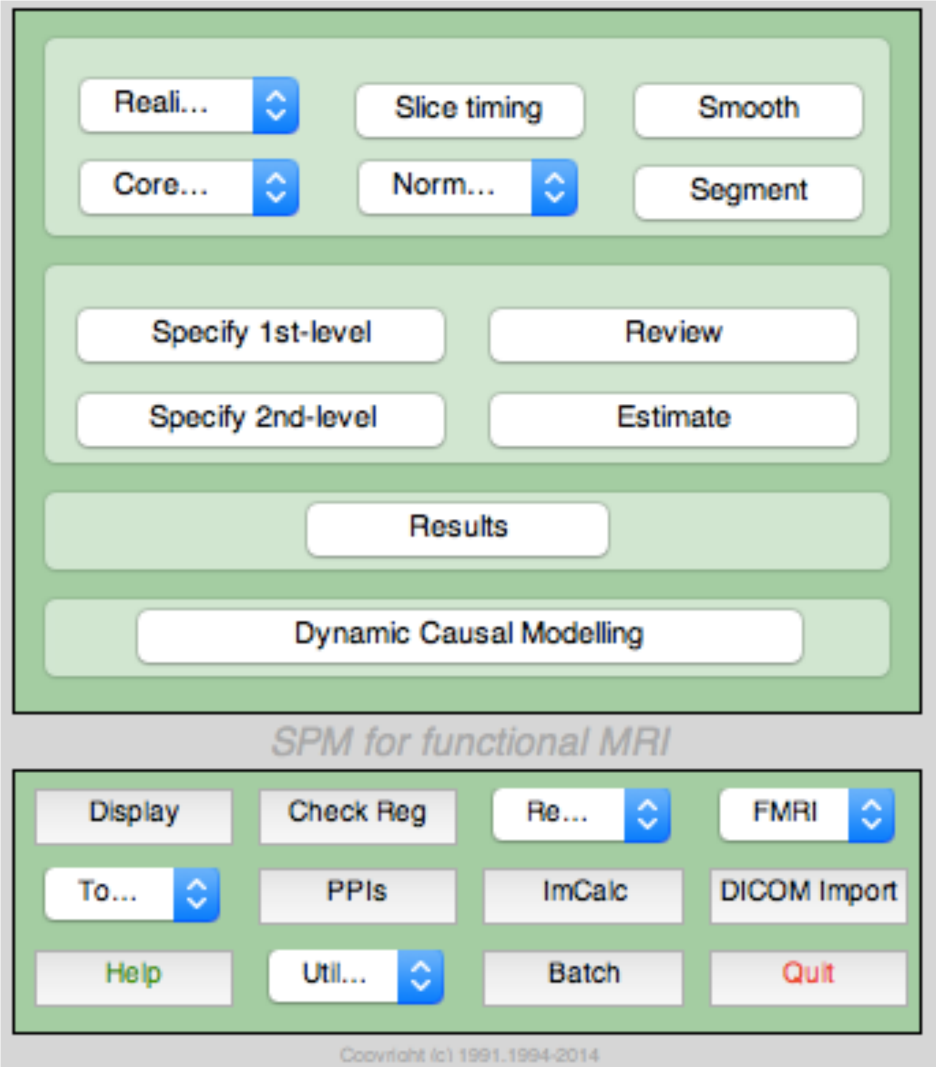
6. First-level analysis:
   - Load your regressors:

```
reg1 =

    [1x189 double]
    [1x189 double]
    [1x189 double]
```

mu1hat	<1x189 double>
positive_PE	<1x189 double>

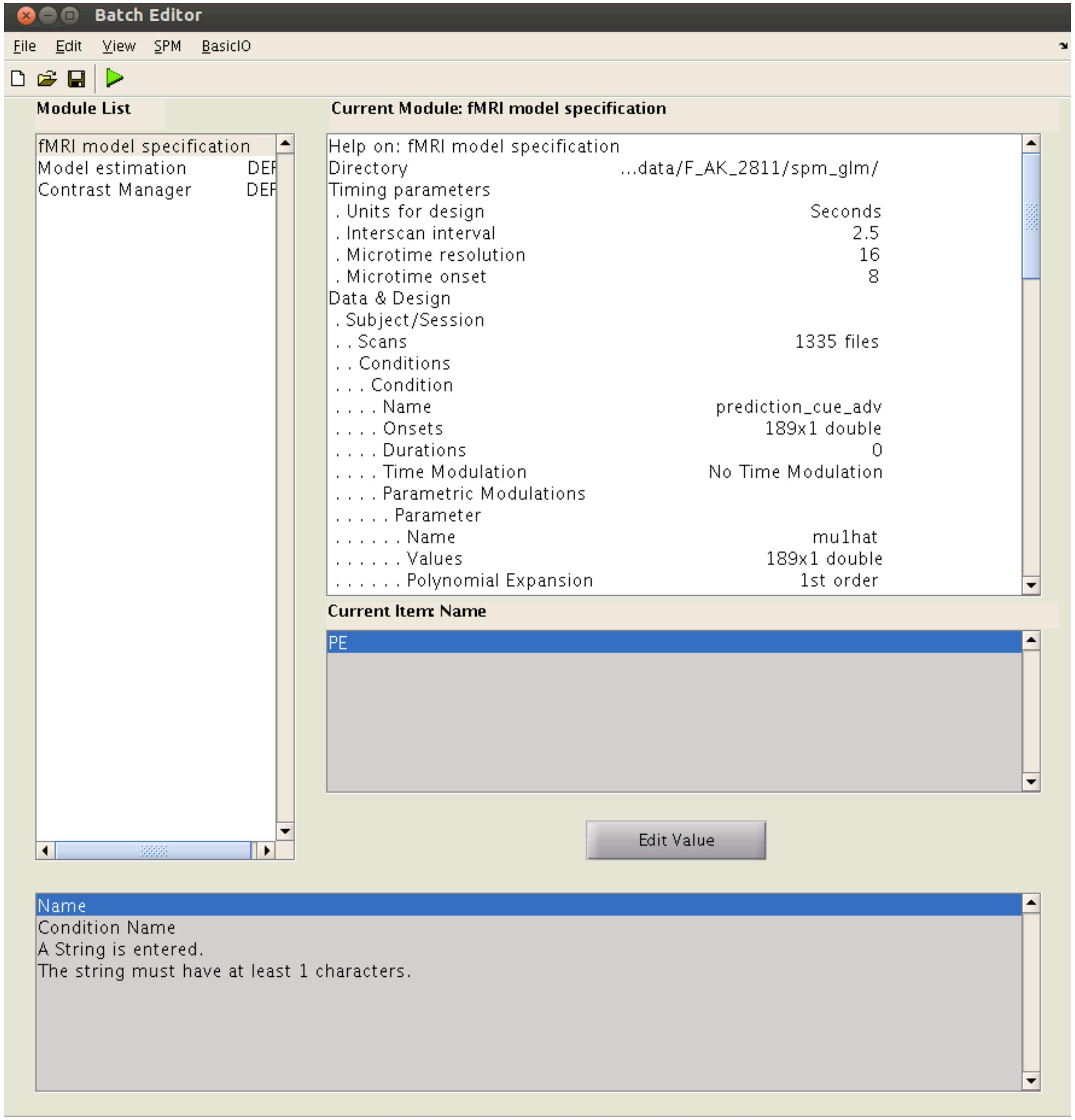
Estimate: single subject

6. First-level analysis:
   - Open SPM: Specify first level analysis



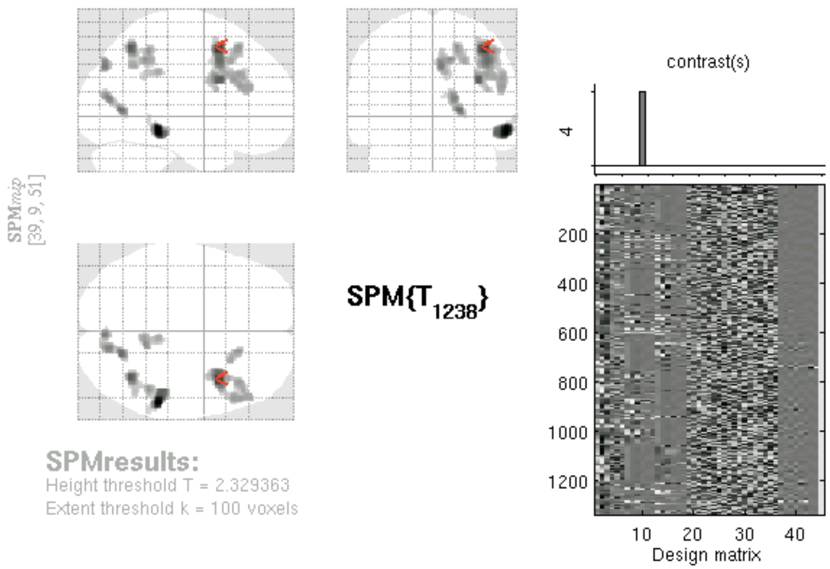
Estimate: single subject

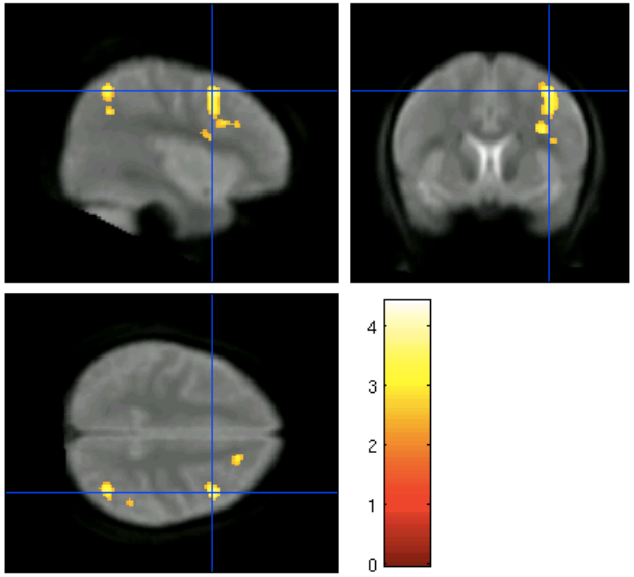
6. First-level analysis:
   - Load Design matrix into Batch editor



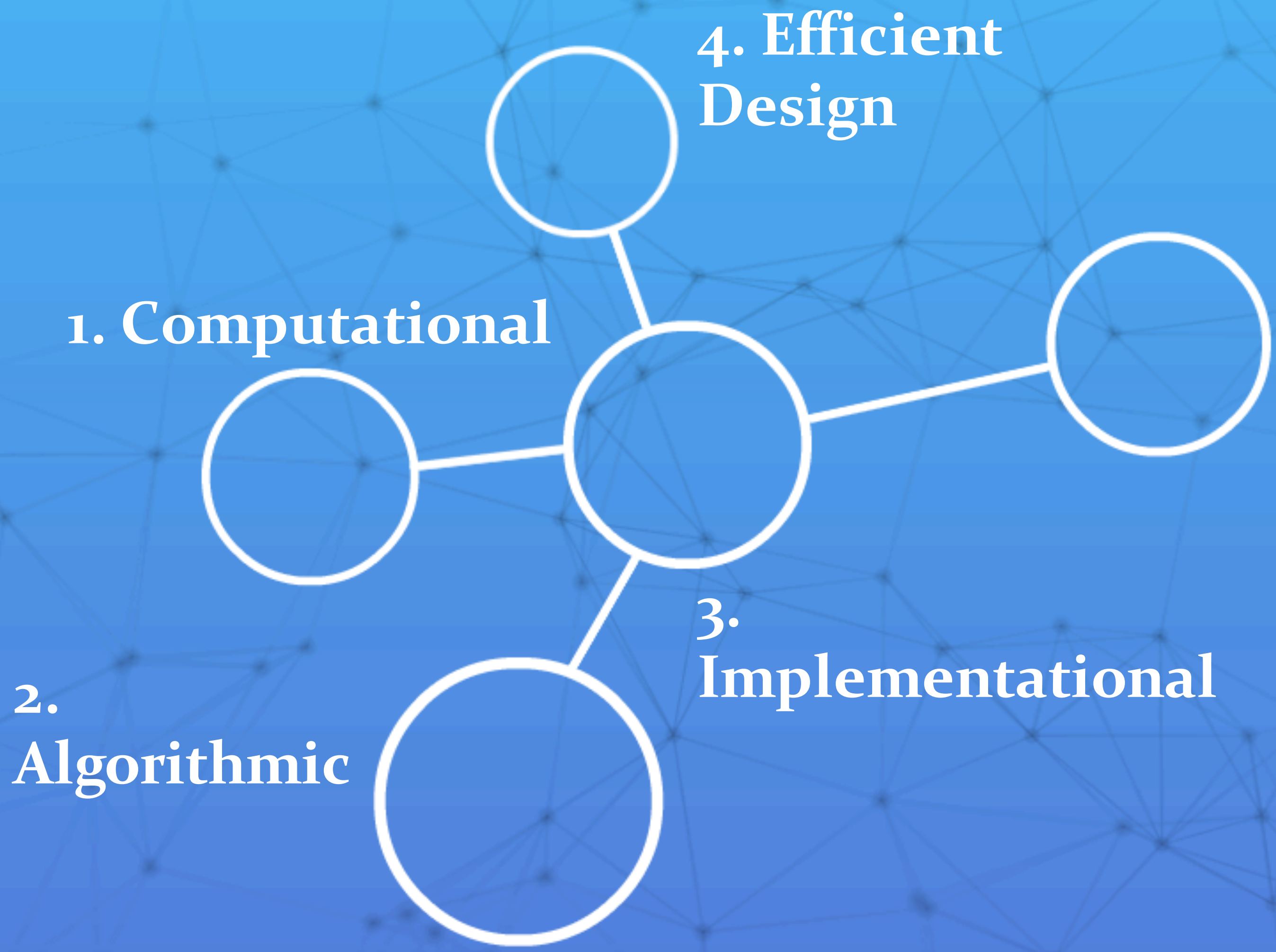
Estimate: single subject

6. First-level analysis:
   - Examine results:
     - PE

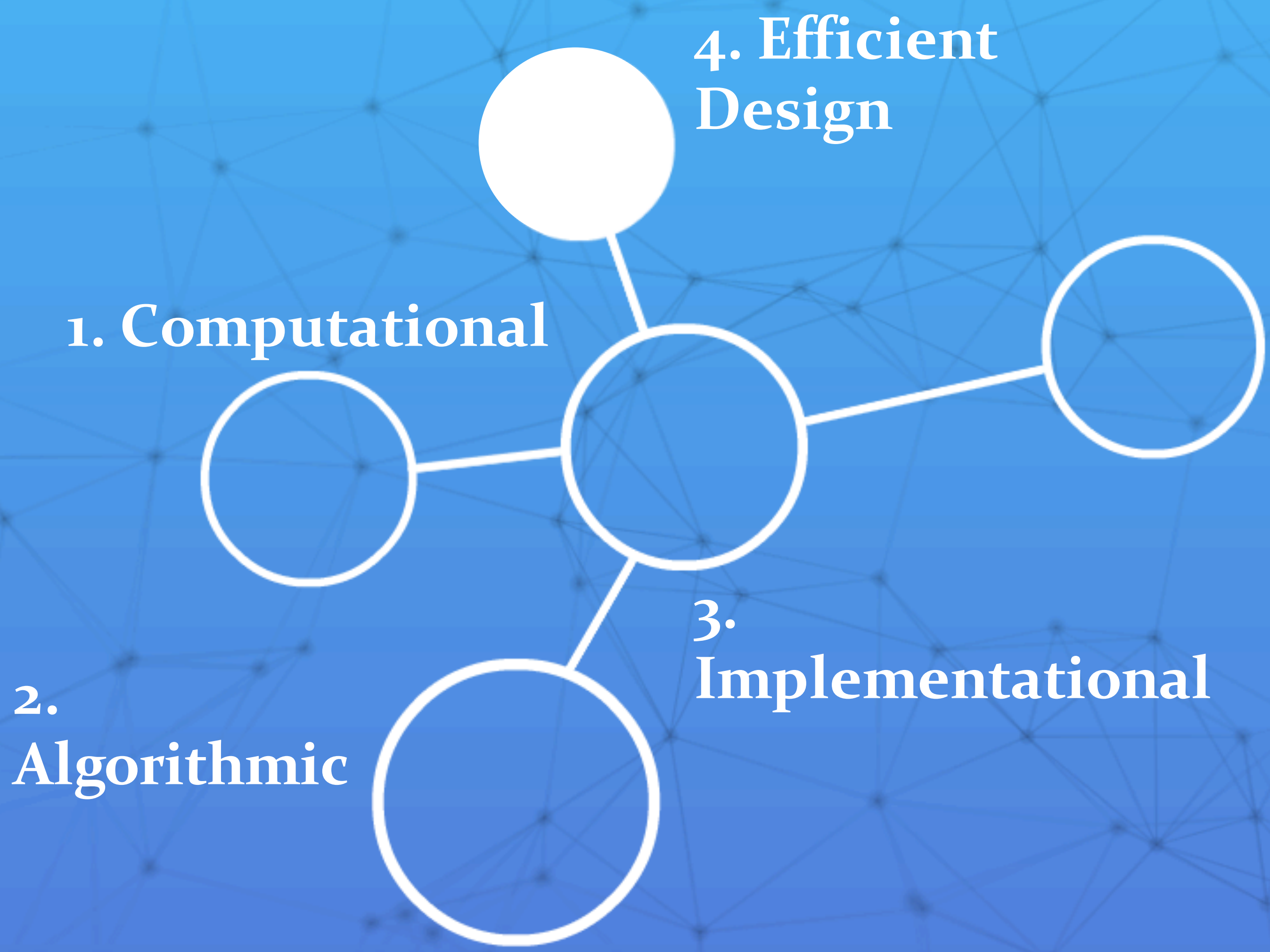




Outline


4. Efficient
Design
1. Computational
3.
Implementational
2.
Algorithmic

Outline


4. Efficient Design
1. Computational
2. Algorithmic
3. Implementational

Tips for efficient experimental design

Translational Neuromodeling Unit

1. Design your "model space" before designing your experiment:

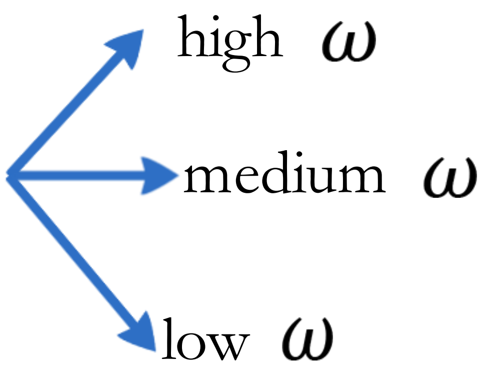
   - The research question and set of hypotheses will determine your model space
   - Formalize your hypotheses mathematically: these will become your models

2. Use simulations to design your "optimal" input structure

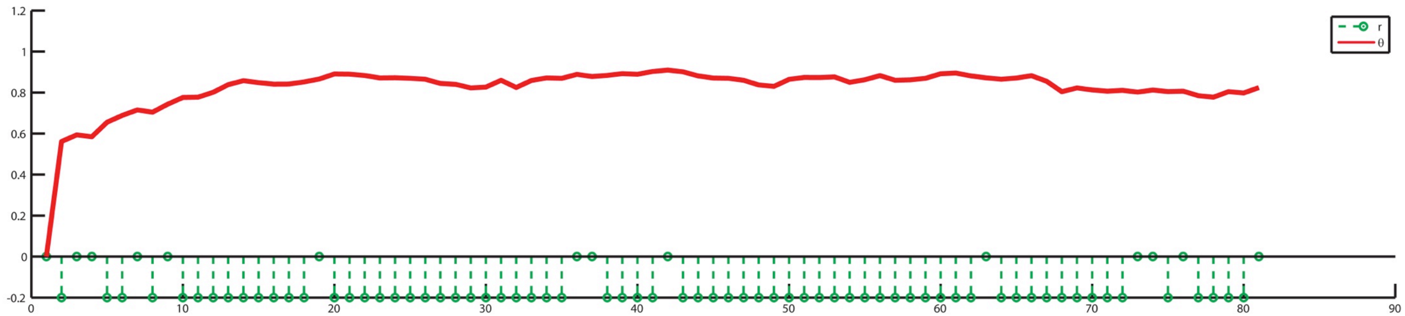
   - Input structure which best allows you to identify your parameters of interest

Example: Social learning experiment

- Simulations: under what conditions can we recover our parameters of interest?



No Volatility: 80% adviser reliability



Example: Social learning experiment

Translational Neuromodeling Unit

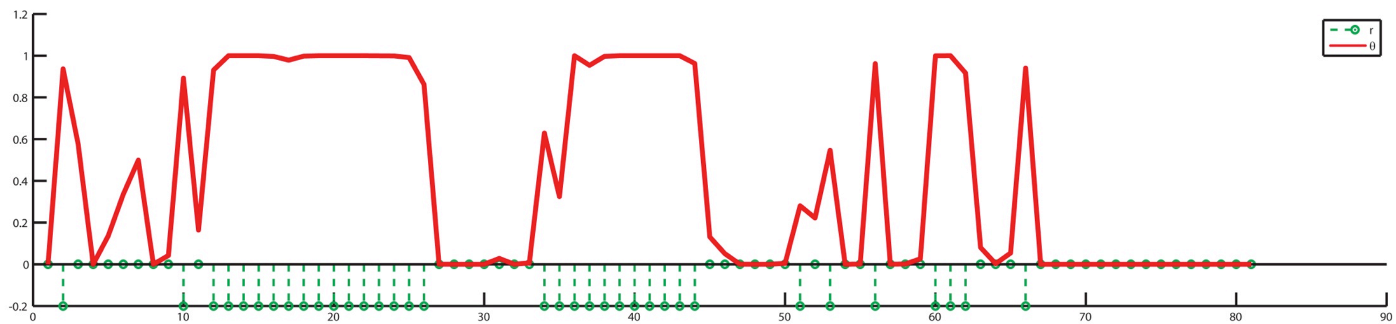
- Simulations: under what conditions, can we recover our parameters of interest?

high ω

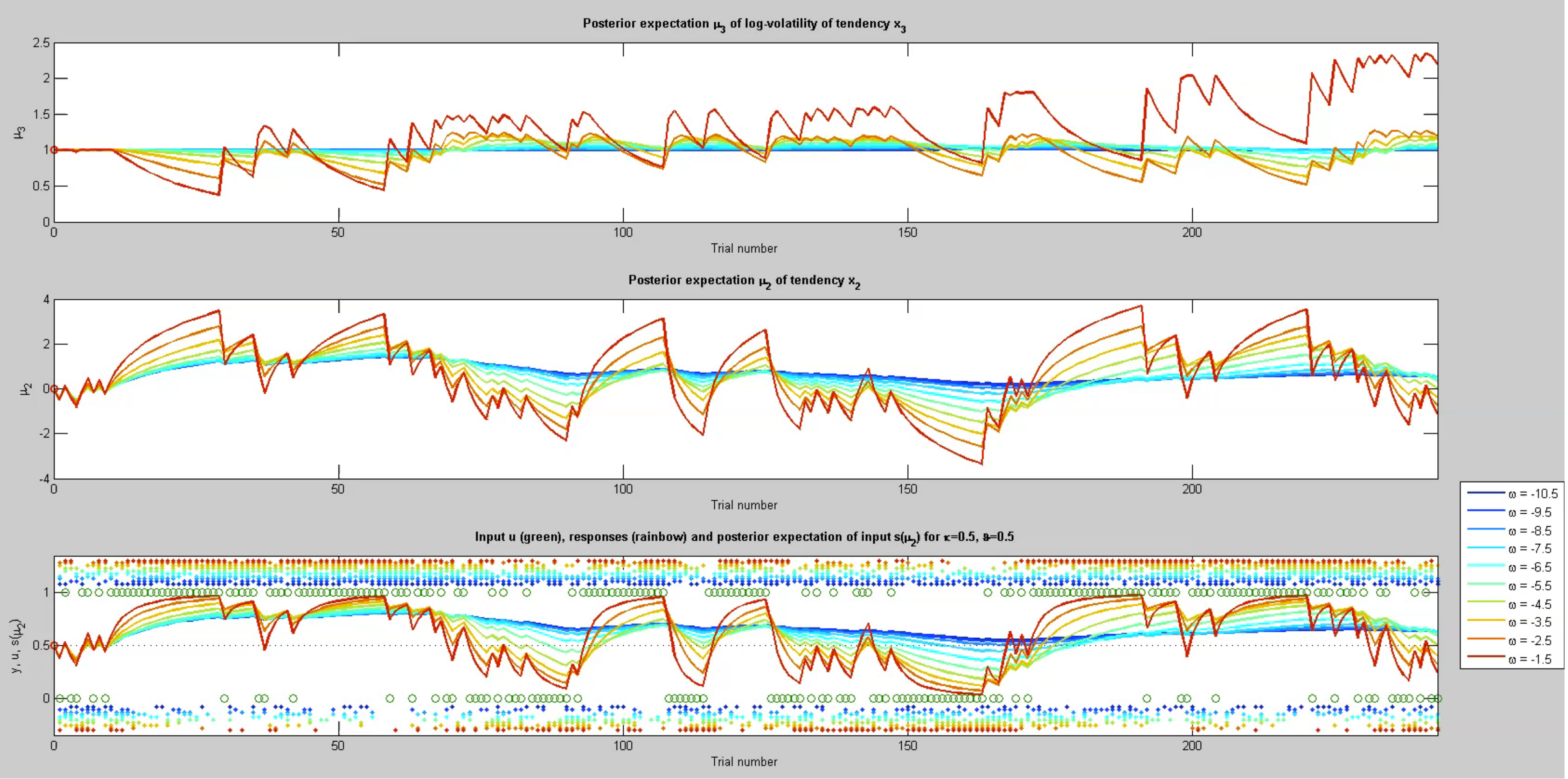
medium ω

low ω

High Volatility: 80% adviser reliability



Simulation Results: Demo



Take-Home Message

Translational Neuromodeling Unit

- Efficient experimental design is formalizing hypotheses in terms of mathematical models.
- Model-based regressors allow for investigation of mechanisms in the brain that are not accessible via direct observation.
- Abstract model-based quantities such as prediction error have shown to correlate with strong neuronal activation.
- In SPM, model-based regressors are treated just like any other parametric modulation.

References

1. Diaconescu, A.O., Mathys, C., Weber, L.A.E., Daunizeau, J., Kasper, L., Lomakina, E.I., Fehr, E., and Stephan, K.E. (2014). Inferring on the Intentions of Others by Hierarchical Bayesian Learning. PLoS Comput Biol *10*, e1003810.
2. Feynman, Richard P. (1972). Statistical Mechanics: A Set of Lectures. Reading, Mass: W. A. Benjamin.
3. Friston, K., Mattout, J., Trujillo-Barreto, N., Ashburner, J., and Penny, W. (2007). Variational free energy and the Laplace approximation. NeuroImage *34*, 220–234.
4. Iglesias, S., Mathys, C., Brodersen, K.H., Kasper, L., Piccirelli, M., den Ouden, H.E.M., and Stephan, K.E. (2013). Hierarchical Prediction Errors in Midbrain and Basal Forebrain during Sensory Learning. Neuron *80*, 519–530.
5. Mathys, C., Daunizeau, J., Friston, K.J., and Stephan, K.E. (2011). A Bayesian foundation for individual learning under uncertainty. Front Hum Neurosci *5*.
6. Mathys, C.D., Lomakina, E.I., Daunizeau, J., Iglesias, S., Brodersen, K.H., Friston, K.J., and Stephan, K.E. (2014). Uncertainty in perception and the Hierarchical Gaussian Filter. Front. Hum. Neurosci. *8*.
7. Montague, P.R., Hyman, S.E., and Cohen, J.D. (2004). Computational roles for dopamine in behavioural control. Nature *431*, 760–767.
8. Rescorla, R.A., and Wagner, A.R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement. (New York: Appleton-Century-Crofts).