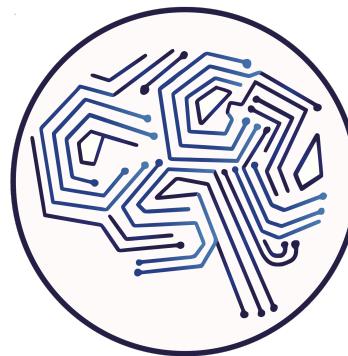


Reinforcement Learning & Decision-Making with hBayesDM



*Woo-Young (Young) Ahn
Department of Psychology
Seoul National University
ccs-lab.github.io*

Installation for R (actually the hardest part)

- *For MAC users:*
 1. *Make sure you have Xcode installed:* <https://developer.apple.com/xcode/>
 2. *Run the following commands through the R console:*

```
install.packages("hBayesDM", dependencies = T)
```

- *For Windows users:*
 1. *Download R tools:* <https://cran.r-project.org/bin/windows/Rtools/>
 - *For more details, see:* <https://github.com/stan-dev/rstan/wiki/Installing-RStan-on-Windows>
 2. *Run the following command through the R console:*

```
install.packages("hBayesDM", dependencies = T)
```

Learning objectives

Participants will...

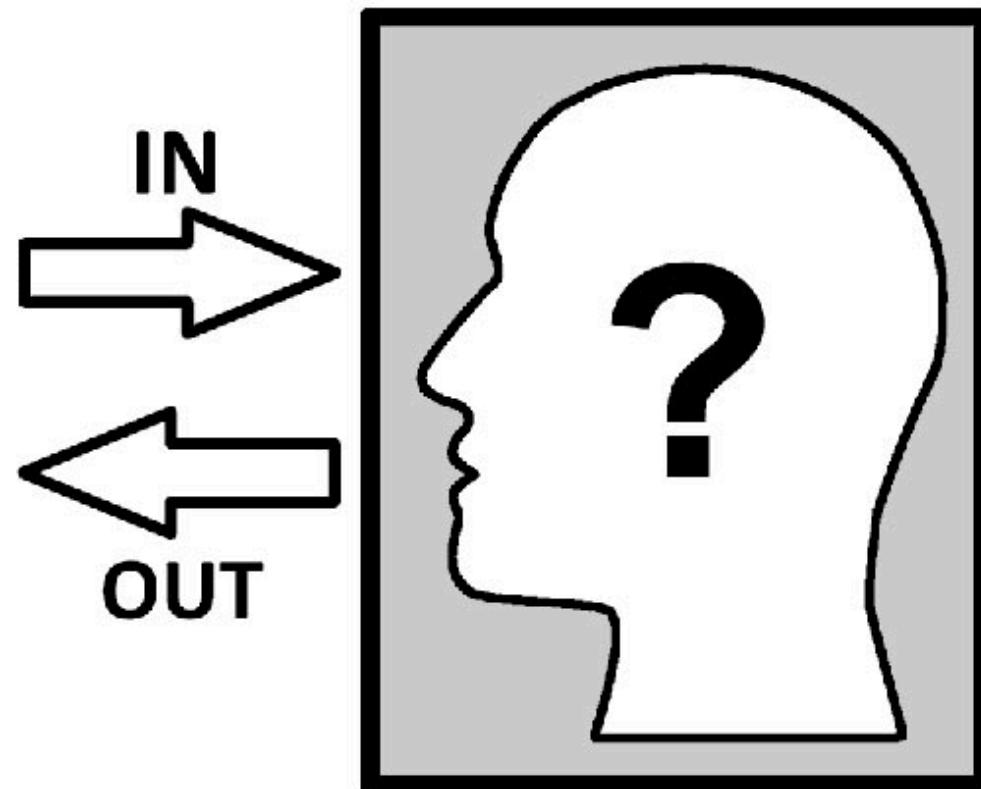
- *Know popular and cutting-edge methods for fitting computational models*
- *Review the concept of Bayesian data analysis*
- *Evaluate outputs from Bayesian data analysis*
- *Use hBayesDM package to model reinforcement learning and decision-making tasks*

Outline for Part I

- *What is computational modeling (a.k.a. cognitive modeling)?*
- *Why/how do we lower the barrier to computational modeling?*
 - *Brief introduction to hBayesDM*
- *How to fit a computational model?*
 - *Maximum likelihood estimation (MLE)*
 - *Bayesian analysis & MCMC sampling*
 - *Hierarchical Bayesian analysis*
 - *Tools for Bayesian data analysis*
- *Things to know when performing MCMC sampling*

Outline

- *Understand the concepts of computational modeling*
- *Know popular and cutting-edge methods for fitting computational models / Understand the concept of Bayesian data analysis*
- *Things to know when performing MCMC sampling*
- *Use hBayesDM package to model several tasks*

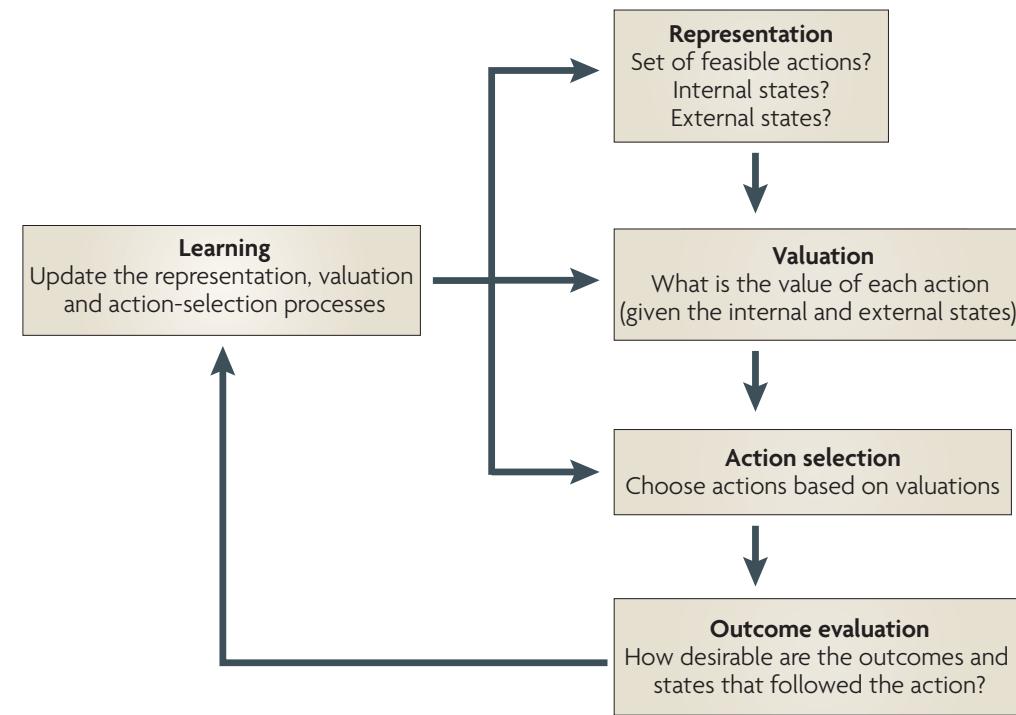


Reinforcement learning and Decision-making (RLDM)

A framework for studying the neurobiology of value-based decision making

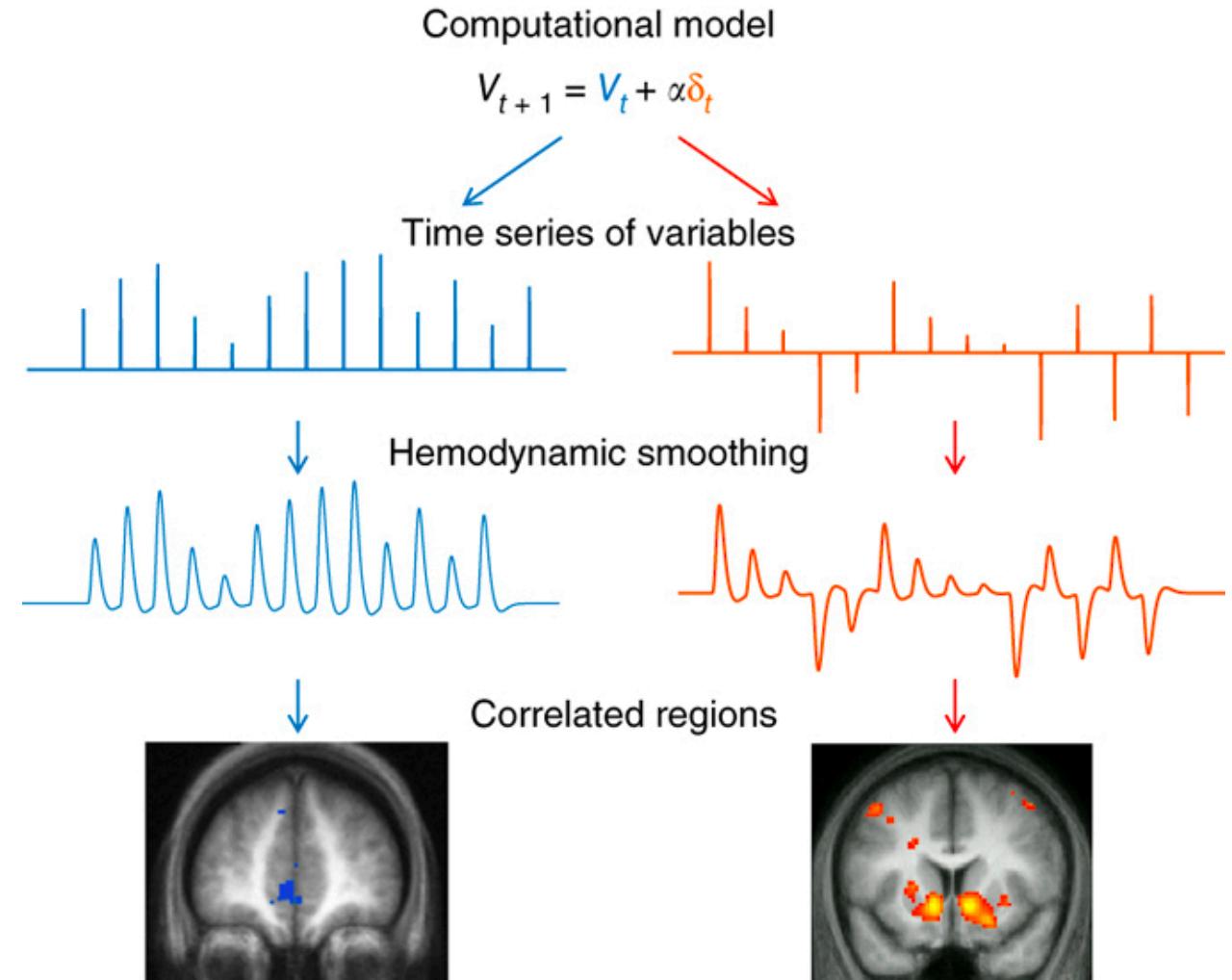
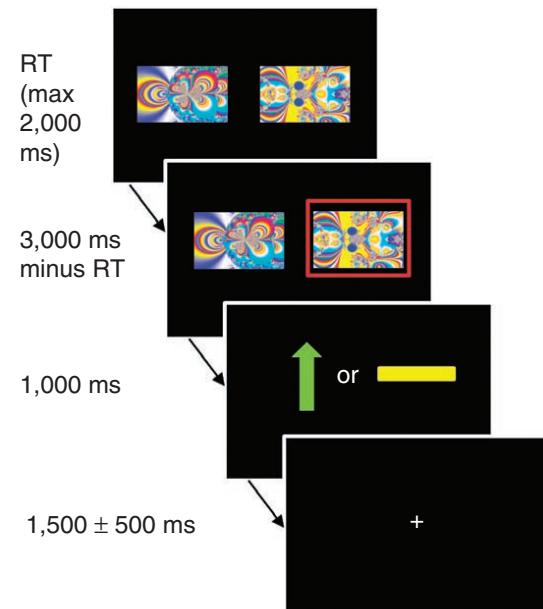
Antonio Rangel*, Colin Camerer* and P. Read Montague[†]

Rangel et al (2008) *Nature Rev. Neuro*



Model-based fMRI/EEG

e.g., Forstmann & Wagenmakers (2015); O'Doherty et al (2007); Cohen et al (2017)



Special Issue: Cognition in Neuropsychiatric Disorders

Computational psychiatry

P. Read Montague^{1,2}, Raymond J. Dolan², Karl J. Friston² and Peter Dayan³



Computational psychiatry: the brain as a phantastic organ

Karl J Friston, Klaas Enno Stephan, Read Montague, Raymond J Dolan



Available online at www.sciencedirect.com

ScienceDirect

Current Opinion in
Neurobiology

Computational approaches to psychiatry
Klaas Enno Stephan^{1,2,3} and Christoph Mathys³

*Computational accounts of
abnormal cognition &
its biological underpinnings*

The screenshot shows the header of the Nature journal website. The main title "nature" is in large white letters, with "International weekly journal of science" in smaller text below it. A navigation bar includes links for Home, News & Comment, Research, Careers & Jobs, Current Issue, Archive, Audio & Video, and a search bar. Below the header, a breadcrumb trail indicates the article's path: Archive > Volume 539 > Issue 7627 > News: Q&A > Article. The main headline reads "US mental-health chief: psychiatry must get serious about mathematics".

Limitations

- *Overly simplified “toy” problems*
 - *Violated assumptions (e.g., discrete space/action & Markov..) , (Gershman & Daw, 2016, Annu Rev Psych)*
- *One-shot learning with sparse data*
 - *Episodic memory (hippocampus) (Gabrieli, 1998; Eichenbaum et al., 1999)*
- *Modeling of even toy problems is hard for many people*

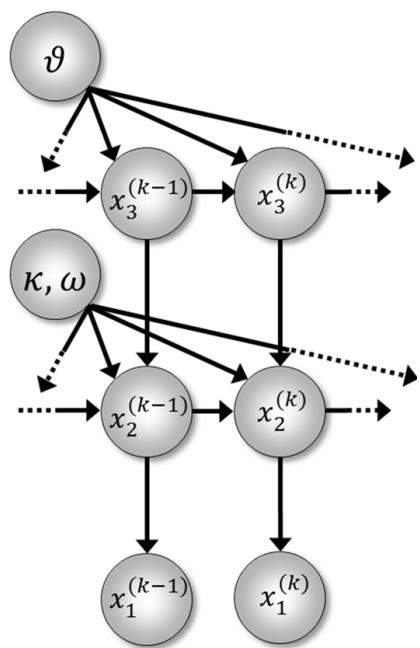
Steps of computational modeling

Busemeyer and Diederich (2010)

- *Formulate assumptions into a mathematical language*
- *Estimate “free” parameters*
- *Model comparisons*
 - *Compare predictions of competing models in terms of their ability to explain empirical data*
- *Reformulate and construct new models in the light of the feedback...*
- *Use computational modeling for the analysis of neuroimaging data (e.g., model-based fMRI)*
- *“All models are wrong but some are useful”*

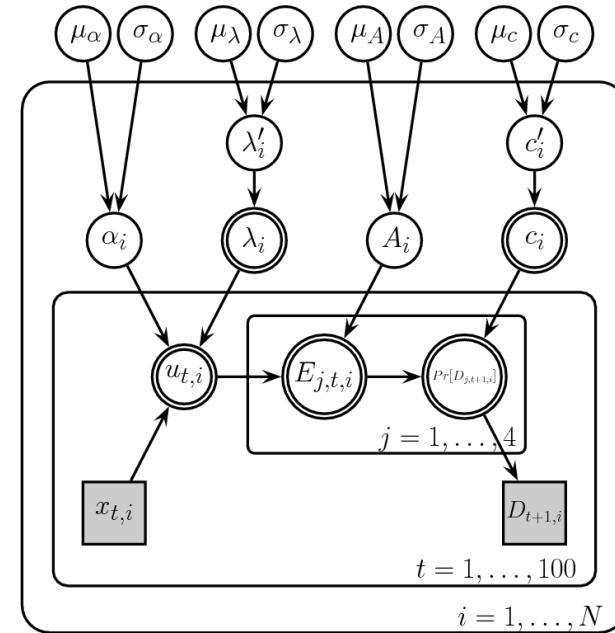
Box (1976)

I like the idea of modeling



Mathys et al (2011) Frontiers

But...

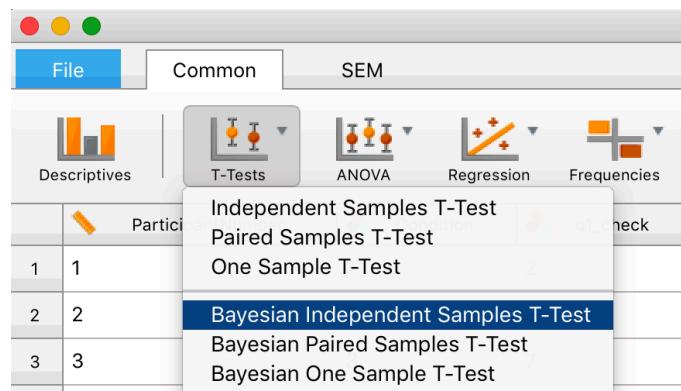


Ahn et al (2011) JNPE

Can we make it easy to do computational modeling?

Q) As easy as *doing a T-test*?

JASP, SPSS



R

```
Console ~/Desktop/ ↗  
> t.test(group1, group2)
```

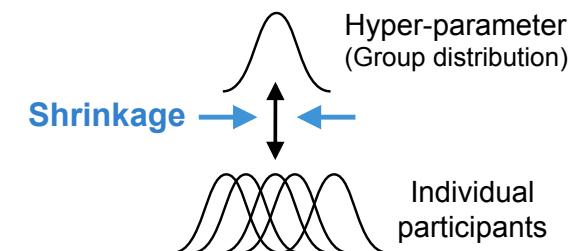
Several packages exist, but ...

Matze et al (2013) Frontiers; Wabersich & Vandekerckhove (2014); Wiecki et al (2013) Frontiers;
Daunizeau et al (2014) PLoS Comp Biol

hBayesDM (hierarchical Bayesian modeling of Decision-Making tasks) Package

- *Models for “many” tasks/paradigms (next slide)*
- *Single-line of coding in R*
 - *Model fitting, visualization, model comparisons*
- *Based on the advanced Bayesian software, Stan (<https://mc-stan.org>).*
- *Hierarchical Bayesian modeling*
- *All codes are publicly available*

<https://github.com/CCS-Lab/hBayesDM>



What tasks and models are available? in CPC2017

Ahn & Busemeyer (2016) Curr Opin Behav Sci

- *Choice reaction time* → sequential sampling models
- *Delay Discounting* (*e.g., Mazur, 1987*)
- *Iowa Gambling* (*Bechara et al, 1994*)
- *(Orthogonalized) Go/Nogo* (*Guitart-Masip et al, 2012*)
- *Two-armed Bandit (Experience-based) including Reversal Learning*
(e.g., Erev et al, 2010)
- *Four-armed Bandit (Experience-based)* (*e.g., Seymour et al, 2012*)
- *Two-choice Description-based* (*e.g., Sokol-Hessner et al, 2009; Tom et al, 2007*)
- *Ultimatum Game* (*e.g., Xiang et al, 2013*)

*What tasks and models are available *in CPC2019*?*

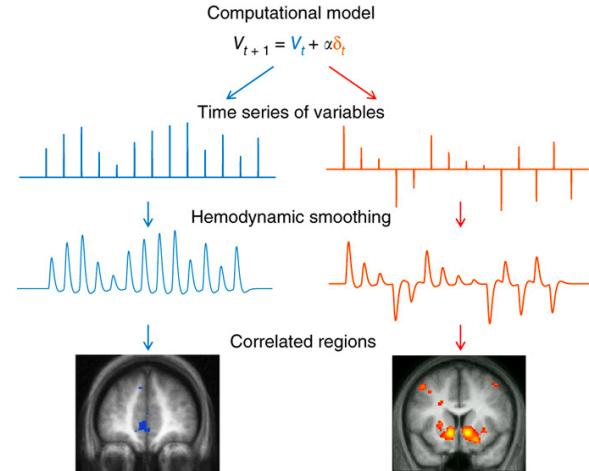
Ahn & Busemeyer (2016) Curr Opin Behav Sci

- *Balloon Analogue Risk Task (BART)* ([Wallsten et al, 2005](#))
- *Choice under Risk and Ambiguity* ([Levy et al, 2010](#))
- *Cambridge Gambling Task* ([Rogers et al, 1999](#))
- *Choice reaction time → sequential sampling models*
- *Delay Discounting* ([e.g., Mazur, 1987](#))
- *Description-based tasks with a probability weighting function* ([e.g., Erev et al., 2010; Hertwig et al., 2004; Jessup et al., 2008](#))
- *Iowa Gambling* ([Bechara et al, 1994](#))
- *(Orthogonalized) Go/Nogo* ([Guitart-Masip et al, 2012](#))
- *Peer influence task* ([Chung et al, 2015](#))
- *Probabilistic Selection task* ([e.g., Frank et al, 2007](#))
- *Two-armed Bandit (Experience-based) including Reversal Learning* ([e.g., Erev et al, 2010](#))
- *Four-armed Bandit (Experience-based)* ([e.g., Seymour et al, 2012](#))
- *Two-choice Description-based* ([e.g., Sokol-Hessner et al, 2009; Tom et al, 2007](#))
- *Ultimatum Game* ([e.g., Xiang et al, 2013](#))
- *Two Step task* ([Daw et al, 2011](#))
- *Wisconsin Card Sorting task* ([Bishara et al, 2010](#))

hBayesDM in Python!

- *Easy to use with Python-based neuroimaging packages (e.g., Nipype)*

The screenshot shows the homepage of the hBayesDM Python documentation. The top navigation bar includes a logo for 'hBayesDM v1.0.1', a search bar labeled 'Search docs', and links for 'Docs' and 'Edit on GitHub'. Below the navigation is a main content area with the title 'hBayesDM' and a subtitle: 'This is the Python version of *hBayesDM* (hierarchical Bayesian modeling of Decision-Making tasks),'. On the left sidebar, there are links for 'Models' and 'Diagnostics'.



Cohen et al (2017)
Nature neuro.

Much easier to contribute to hBayesDM now!

<https://github.com/CCS-Lab/hBayesDM/wiki>

Home

Woo-Young Ahn edited this page now · 3 revisions

Welcome to the hBayesDM wiki!

Here we show how to add new tasks/models to the hBayesDM package.

Check the [Repository Structure](#) and carefully follow instructions in the following sections:

[How to contribute](#)

- [How to add new model](#)

▶ Pages 6

- [Home](#)
- [Repository Structure](#)
- [How to contribute](#)
- [How to add new model](#)

To-do list

- More tasks
 - *Hierarchical Gaussian Filtering* ([Mathys et al., 2011](#))
 - *More models for the delay discounting task.*
 - *More sequential sampling models (e.g., drift diffusion models with different drift rates for multiple conditions).*
 - *Passive avoidance learning task* ([Newman & Kossen, 1986; Newman et al., 1985](#))
 - *Stop Signal Task (SST)*
 - *Model-based regressors* ([O'Doherty et al., 2007](#)) from more tasks.

How can I use it?

Tutorials available at

https://ccs-lab.github.io/hBayesDM/articles/getting_started.html (*R*)

```
install.packages("hBayesDM", dependencies=TRUE)
```

<https://hbayesdm.readthedocs.io/> (*Python*)

```
pip install hbayesdm # Install using pip
```

Brief step-by-step tutorials

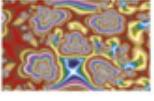
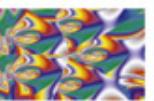
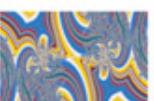
* Part II for detailed tutorials

1. *Prepare raw (trial-by-trial) data*
2. *Fit candidate models*
3. *Plot (visualize) and inspect model parameters*
4. *Compare models (if there exist competing models)*

The Orthogonalized Go/Nogo task

- gng_m1
- gng_m2
- gng_m3
- gng_m4

*Guitart-Masip et al, 2012 Neuroimage
Cavanagh et al, 2013 J Neuro*

	punishment	reward
go	go to avoid losing 	goto win 
nogo	nogo to avoid losing 	nogo to win 

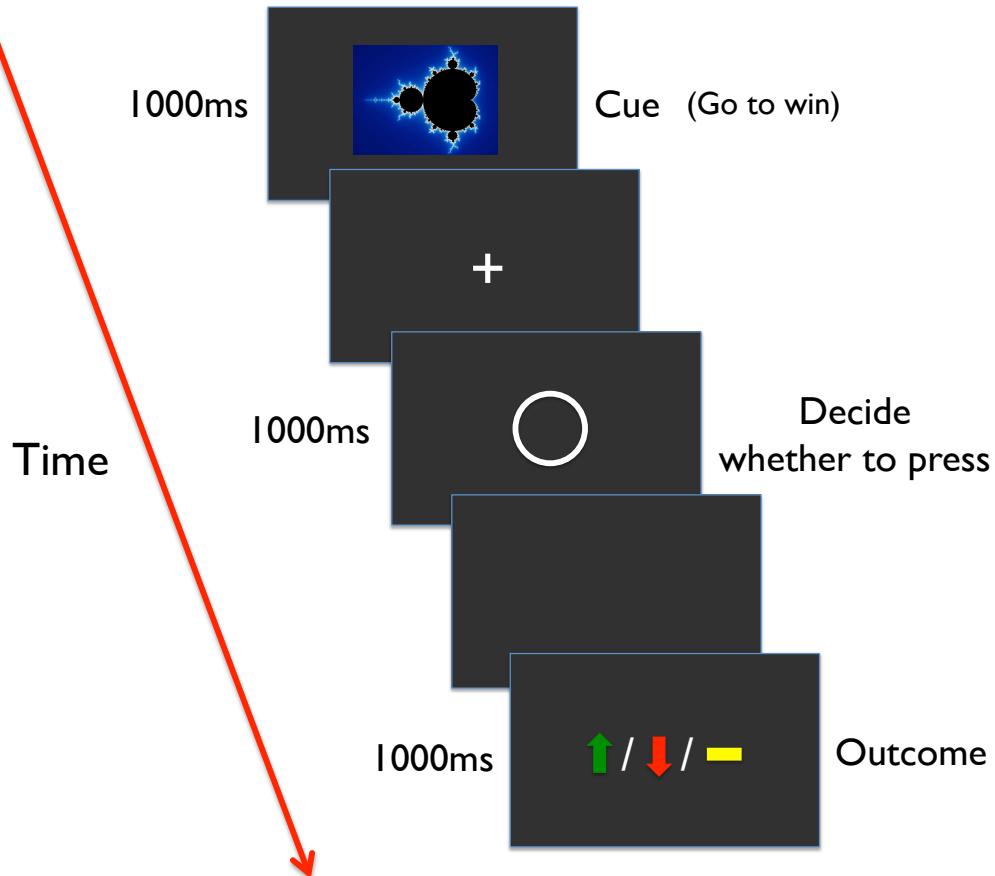
Orthogonalized Go/Nogo task

	Loss	Gain
Go	Go to avoid	Go to win
Nogo	Nogo to avoid	Nogo to win



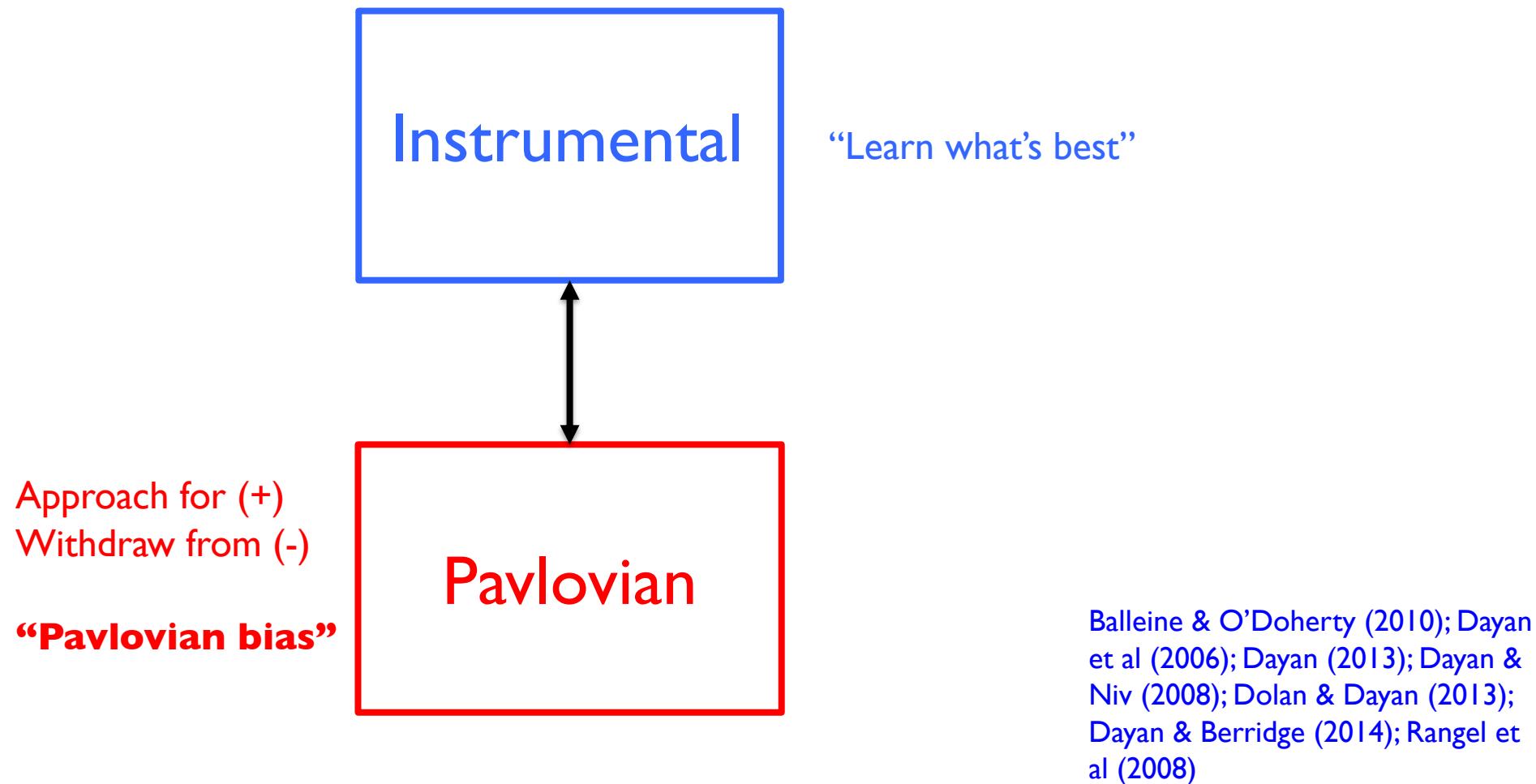
- 4 cues (conditions)

2 actions (Go / Nogo) x
2 valence (Gain / Loss)

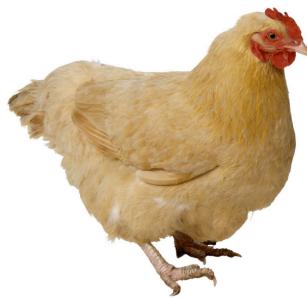


Orthogonalized Go/Nogo task

Pavlovian-Instrumental competition



Hungry
Chicken

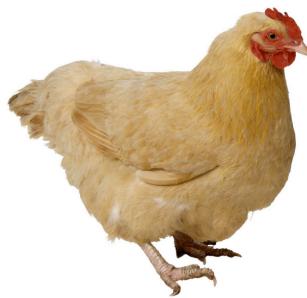


Food!



Hershberger (1986)

Hungry
Chicken

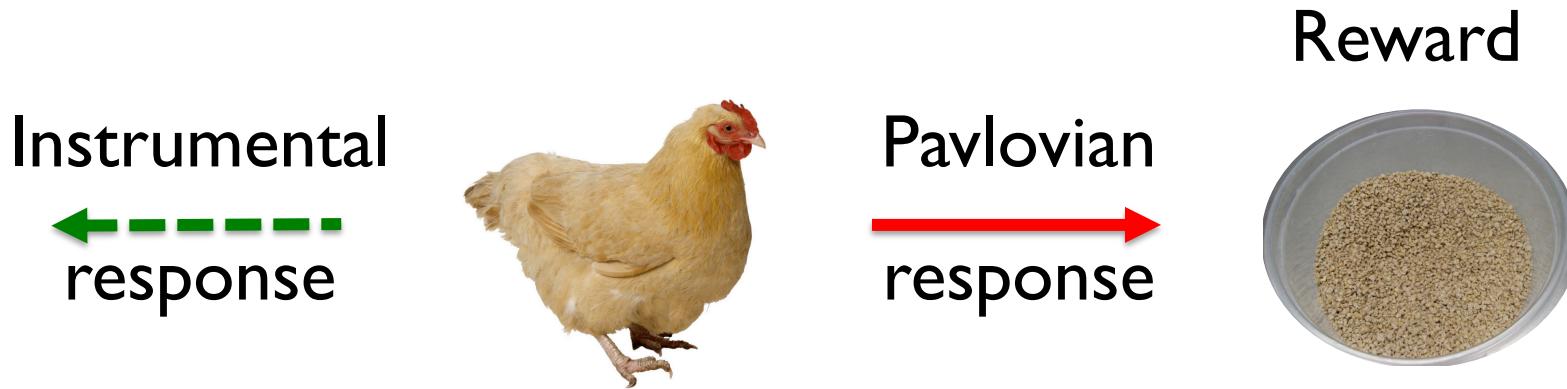


Food!



Hershberger (1986)

Pavlovian-Instrumental competition



Hershberger (1986)

Pavlovian-Instrumental competition

Nogo

*Instrumental
response*



Young on diet

Go

*Pavlovian
response*

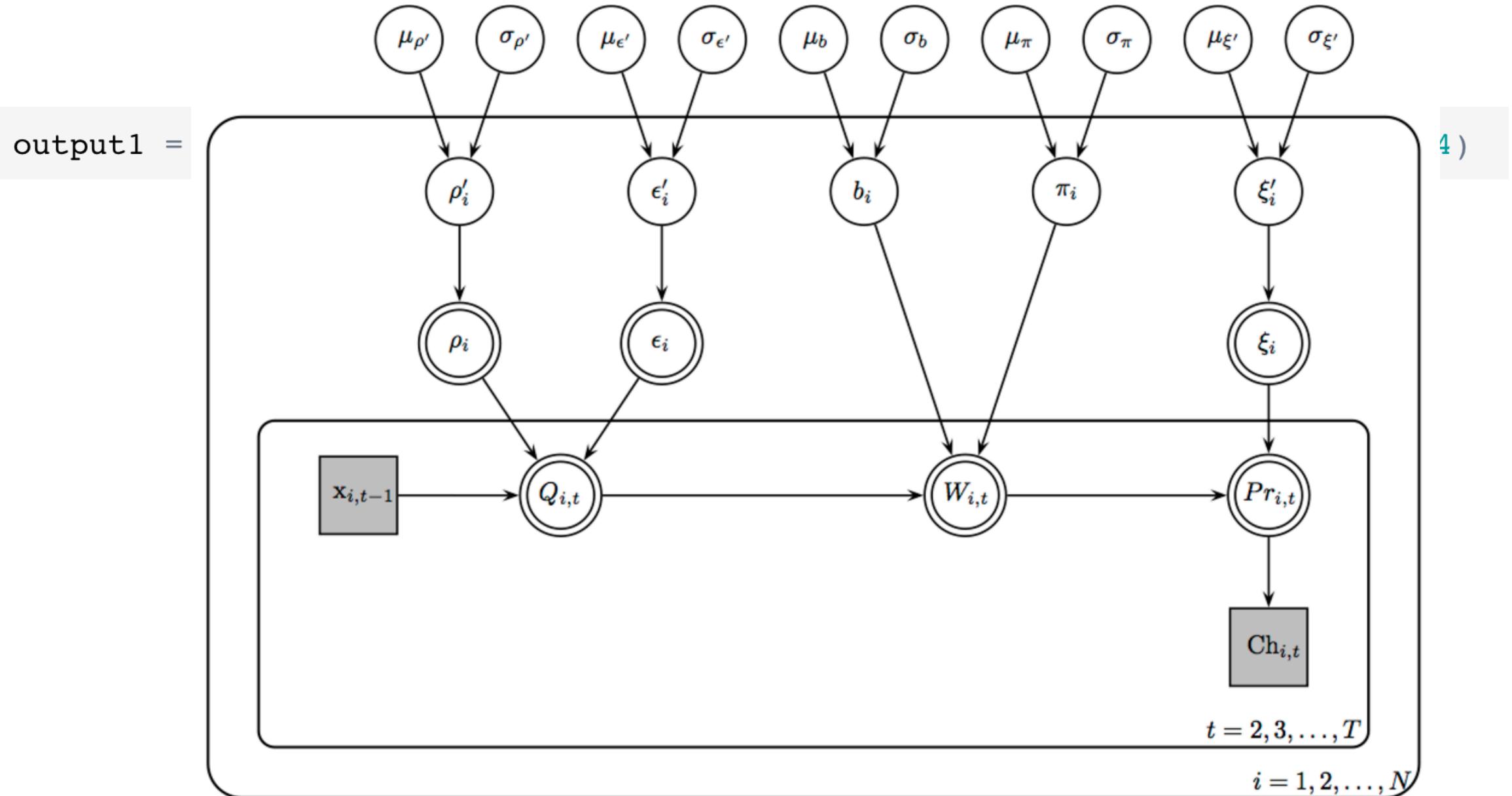


Instrumental goal: To lose weight

1. Prepare raw (trial-by-trial) data as a single text file

A	B	C	D	E	F	G	
1	trialNum	cue	keyPressed	success	congruentOutcome	outcome	subjID
2	1	1	1	1	2	0	1
3	2	2	0	1	1	1	1
4	3	4	0	1	1	0	1
5	4	4	1	0	1	-1	1
6	5	4	0	1	1	0	1
7	6	1	1	1	1	1	1
8	7	3	0	0	1	-1	1
9	8	1	1	1	1	1	1
10	9	3	1	1	1	0	1
11	10	3	0	0	1	-1	1
12	11	4	0	1	1	0	1
13	12	4	0	1	1	0	1
14	13	4	0	1	1	0	1

2. Fit candidate models



Details:

```
# of chains          = 4
# of cores used     = 4
# of MCMC samples (per chain) = 2000
# of burn-in samples = 1000
# of subjects        = 10
# of (max) trials per subject = 240
```

** Loading a precompiled model **

starting worker pid=75130 on localhost:11950 at 08:25:48.905

starting worker pid=75138 on localhost:11950 at 08:25:49.101

SAMPLING FOR MODEL 'gng_m1' NOW (CHAIN 1).

Chain 1, Iteration: 1 / 2000 [0%] (Warmup)

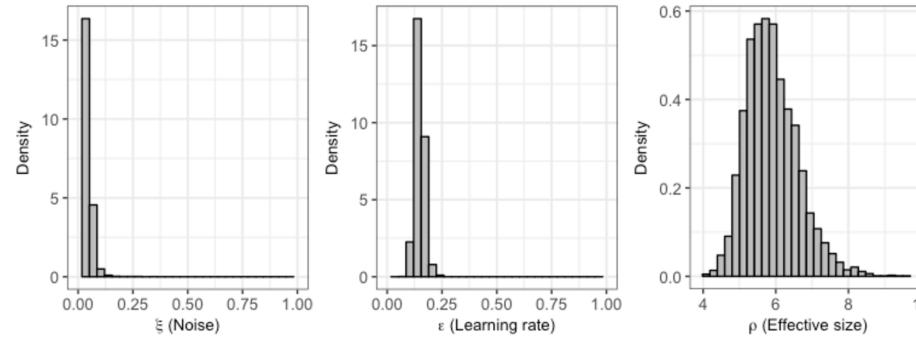
SAMPLING FOR MODEL 'gng_m1' NOW (CHAIN 2).

...

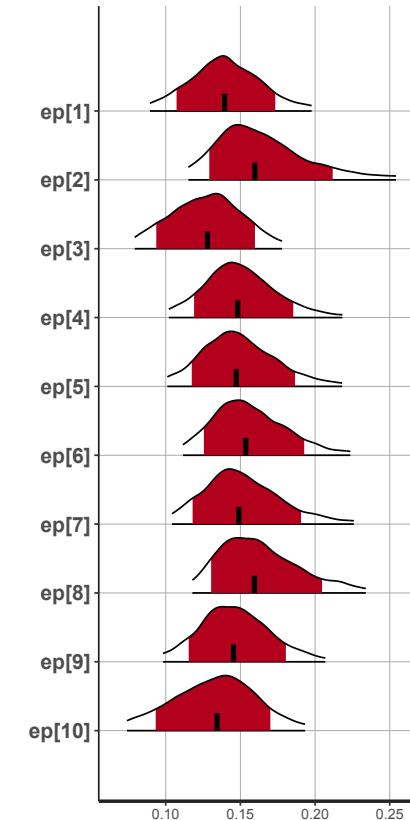
**** Model fitting is complete! ****

3. Plot (visualize) and inspect model parameters

```
plot(output1)
```



```
plotInd(output1, "ep")
```



```
> output1$allIndPars
```

	xi	ep	rho	subjID
1	0.03688558	0.1397615	5.902901	1
2	0.02934812	0.1653435	6.066120	2
3	0.04467025	0.1268796	5.898099	3
4	0.02103926	0.1499842	6.185020	4
5	0.02620808	0.1498962	6.081908	5
...				

Python version

```
# Load hBayesDM

from hbayesdm import rhat, print_fit
from hbayesdm.models import gng_m1
```

```
# Run the model and store results in "output"
output = gng_m1(data='example', niter=2000, nwarmup=1000, nchain=4, ncore=4)
```

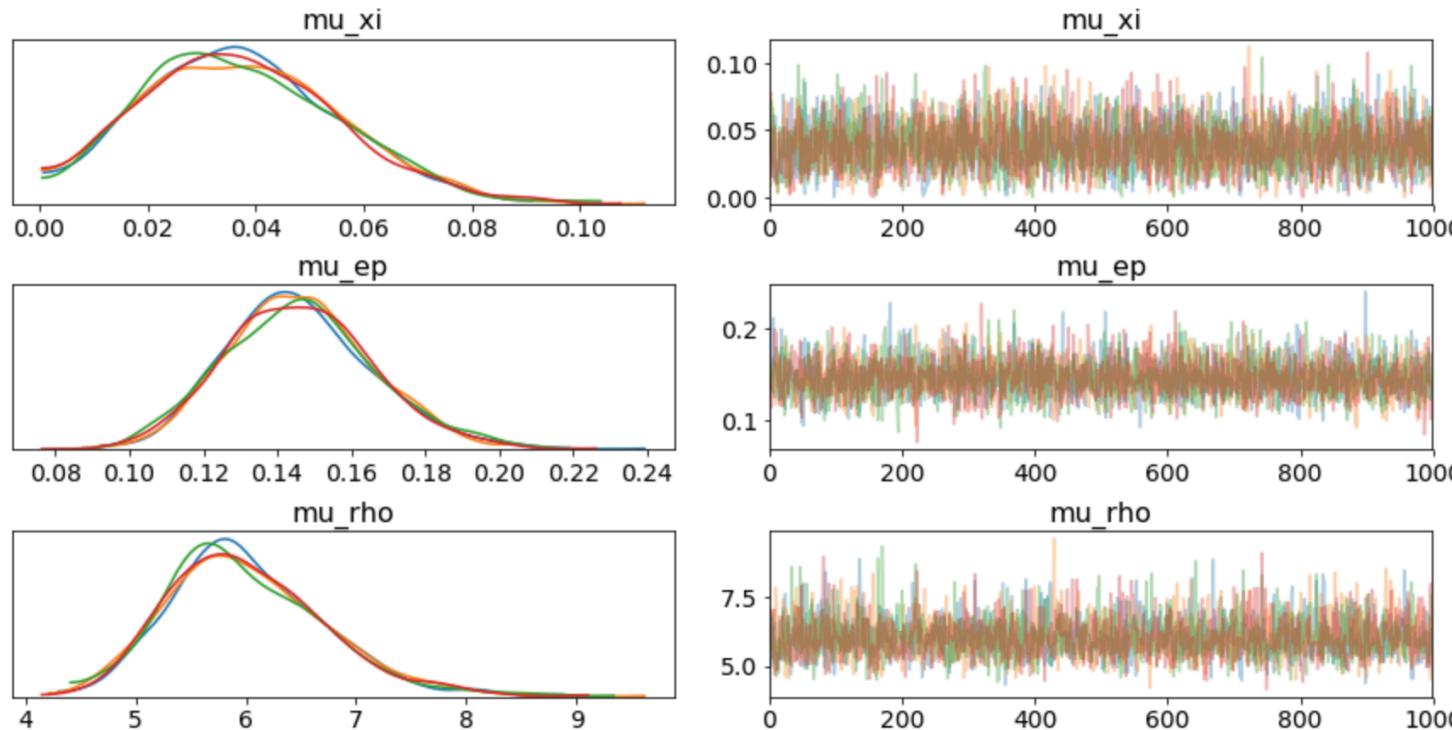
INFO:pystan:COMPILING THE C++ CODE FOR MODEL gng_m1_5ae9a279bb0dc4d00c887e56e86cead2 NOW.

Model = gng_m1
Data = example

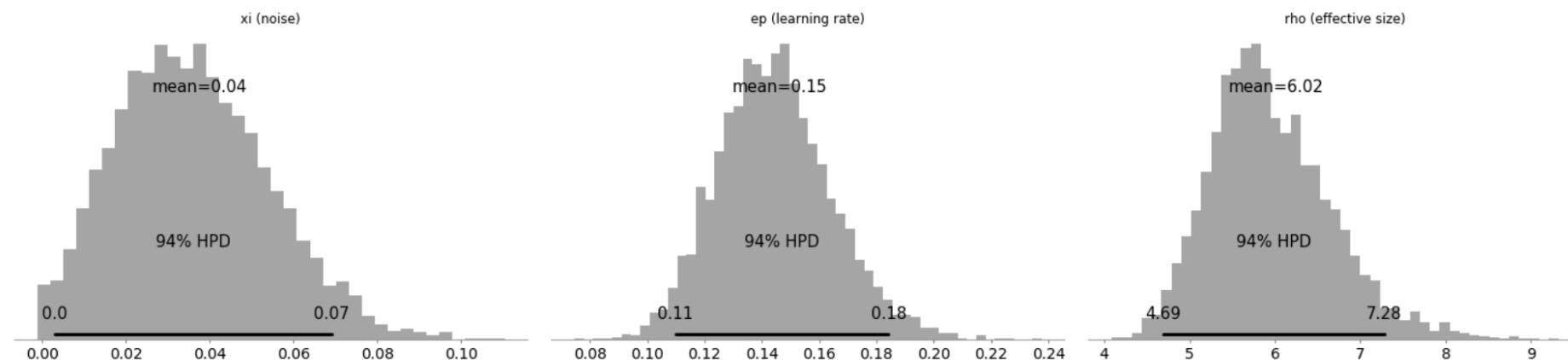
Details:

# of chains	= 4
# of cores used	= 4
# of MCMC samples (per chain)	= 2000
# of burn-in samples	= 1000
# of subjects	= 10
# of (max) trials per subject	= 240

```
# Visually check convergence of the sampling chains (should look like "hairy caterpillars")
output.plot(type='trace')
```



```
# Plot posterior distributions of the hyper-parameters (distributions should be unimodal)
output.plot()
```



4. Bayesian model comparisons

Vehtari et al. (2016)

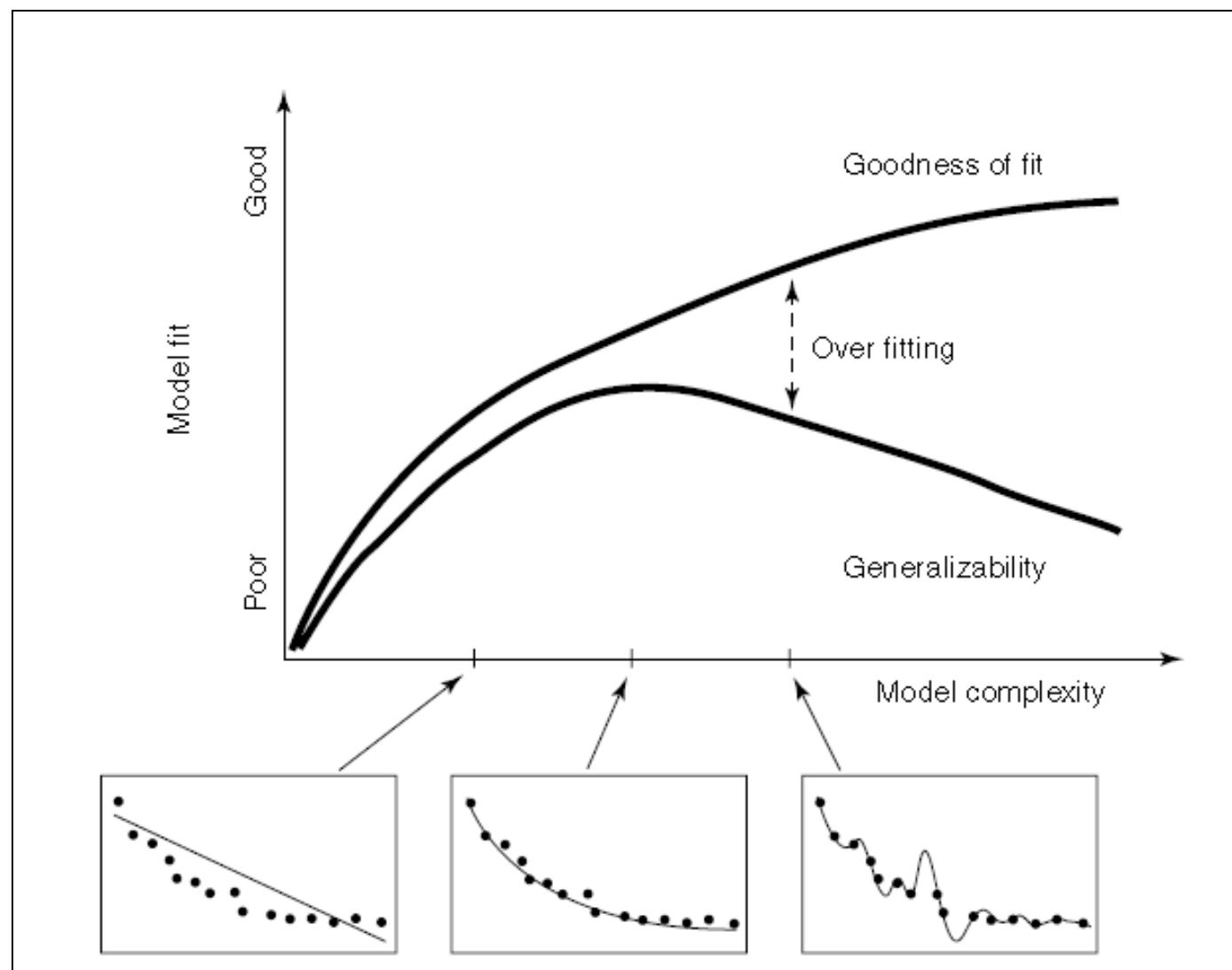
Leave-One-Out Information Criterion (LOOIC) - default

Widely Applicable Information Criterion (WAIC)

```
> printFit(output1, output2, output3, output4)
```

*Model #4 is the best model
(in terms of LOOIC)*

Goodness of Fit vs Model Complexity



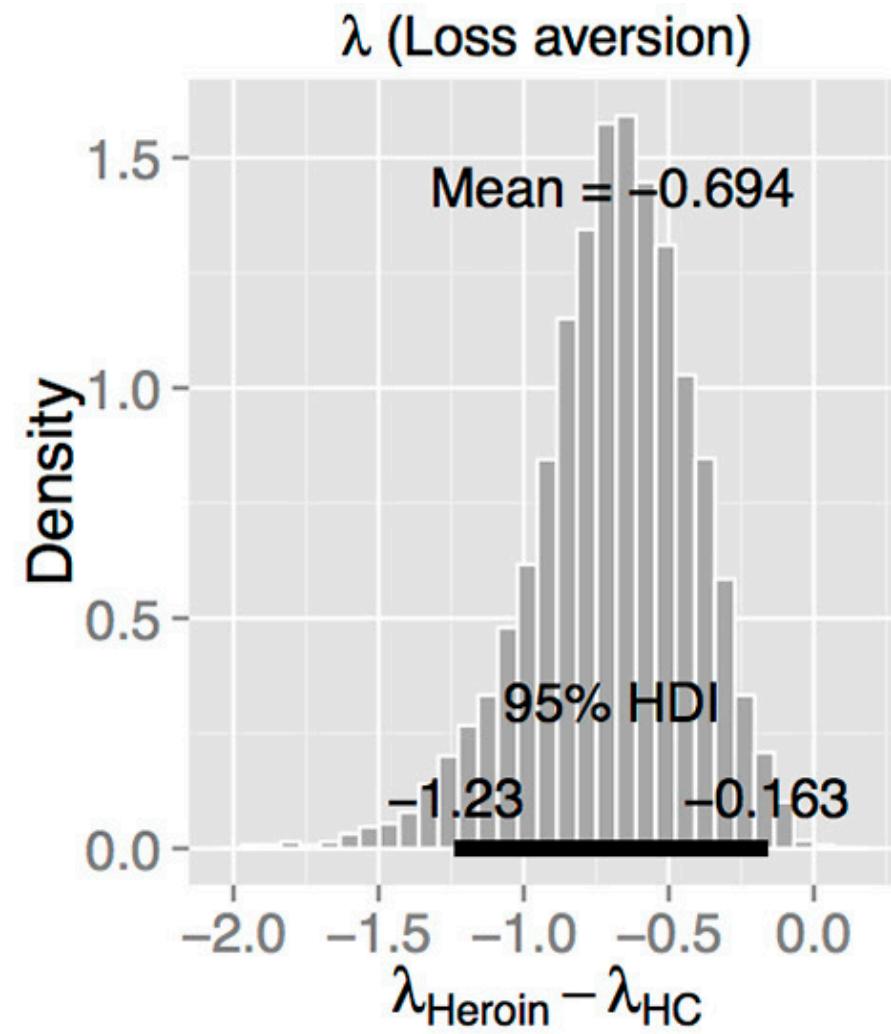
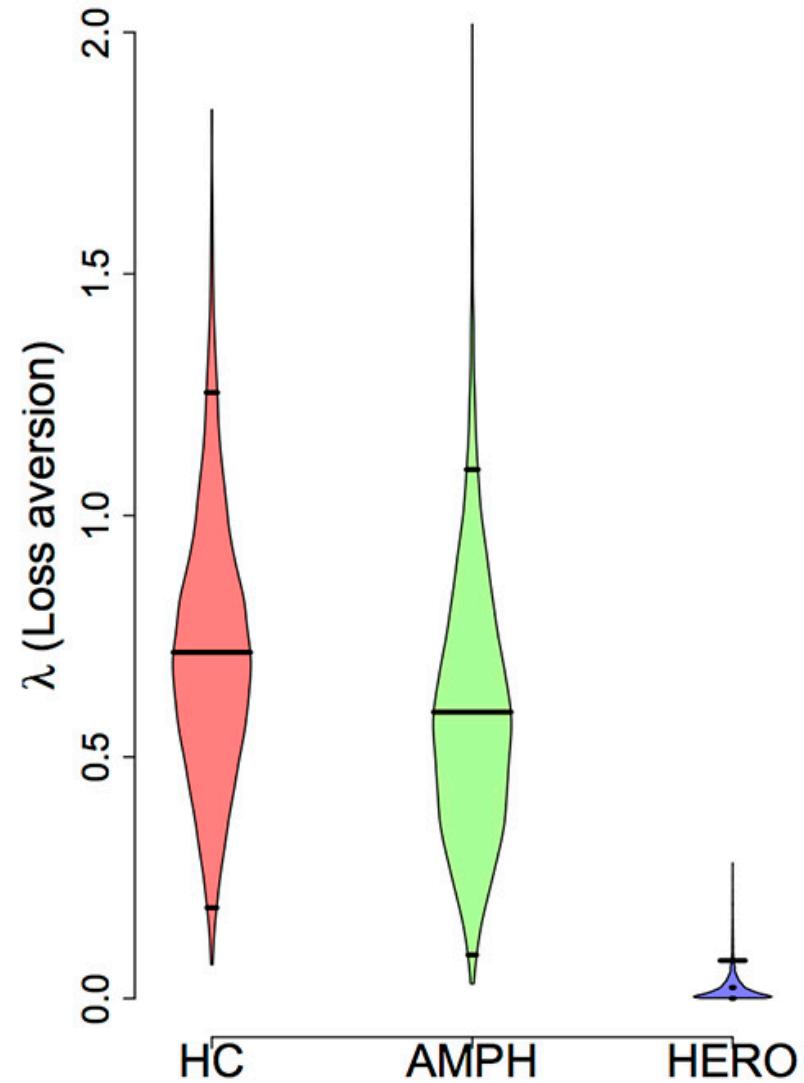
5. Group comparisons

```
data_group1 = "~/Project_folder/gng_data_group1.txt" # data file for group1
data_group2 = "~/Project_folder/gng_data_group2.txt" # data file for group2

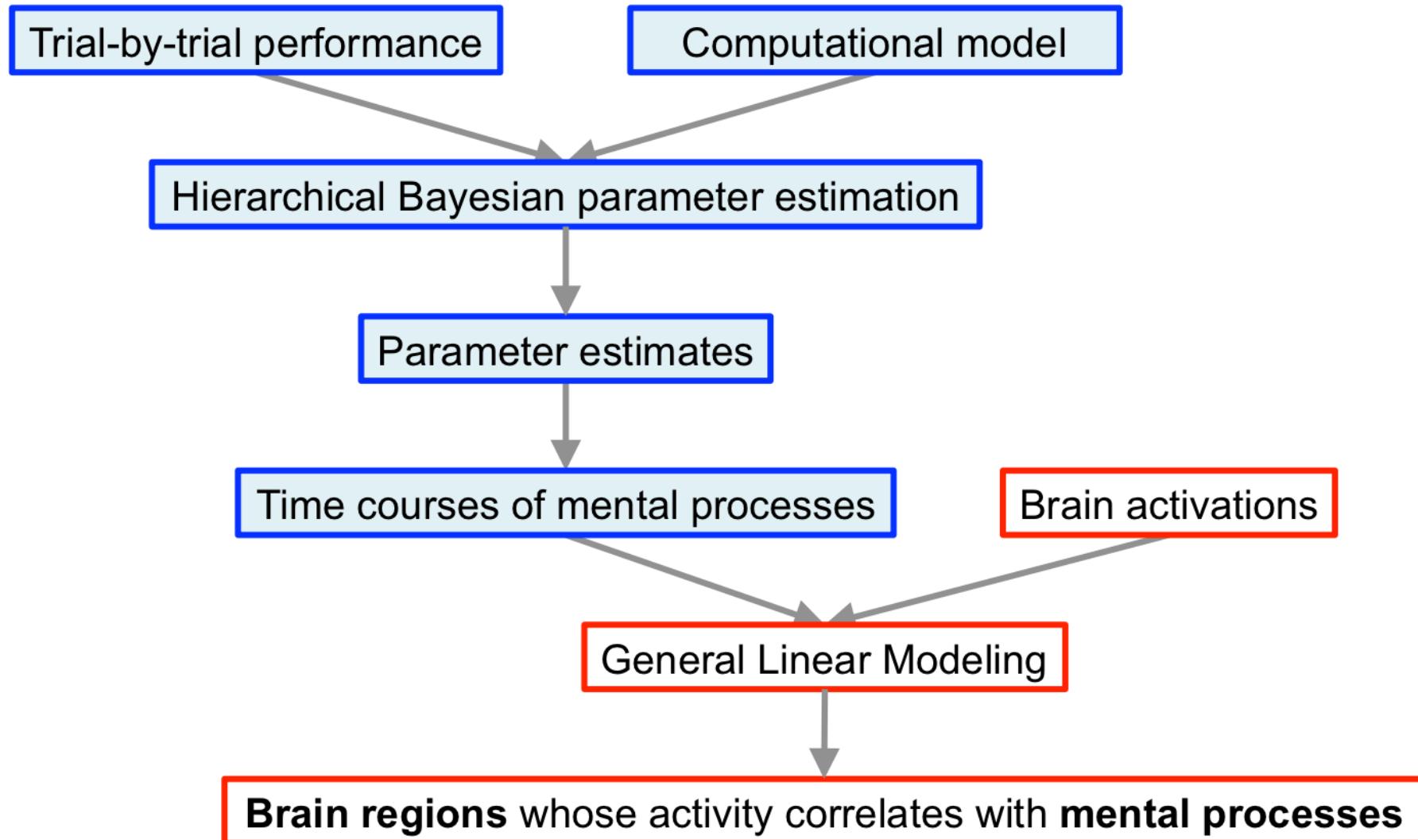
output_group1 = gng_m4(data_group1) # fit group1 data with the gng_m4 model
output_group2 = gng_m4(data_group2) # fit group2 data with the gng_m4 model

# After model fitting is complete for both groups,
# evaluate the group difference (e.g., on the 'pi' parameter) by examining the posterior distribution of
# group mean differences.

diffDist = output_group1$parVals$mu_pi - output_group2$parVals$mu_pi # group1 - group2
HDIofMCMC( diffDist ) # Compute the 95% Highest Density Interval (HDI).
plotHDI( diffDist ) # plot the group mean differences
```



6. Model-based fMRI/EEG



Model-based fMRI/EEG

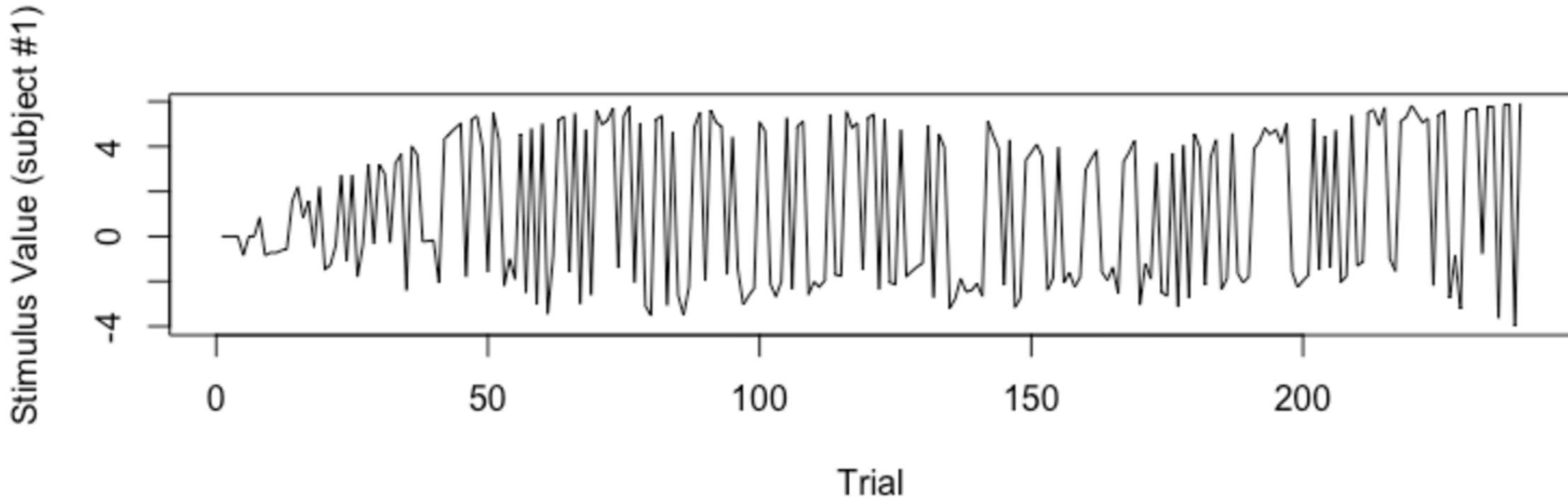
- *Trial-by-trial regressors are readily available*

```
# fit example data with the gng_m3 model
output = gng_m3(data="example", niter=3000, nwarmup=1000, nchain=3, ncore=3, modelRegressor=TRUE)
```

```
# store all subjects' stimulus value (SV) in 'sv_all'  
sv_all = output$modelRegressor$SV  
  
dim(output$modelRegressor$SV) # number of rows=# of subjects (=10), number of columns=# of trials (=240)
```

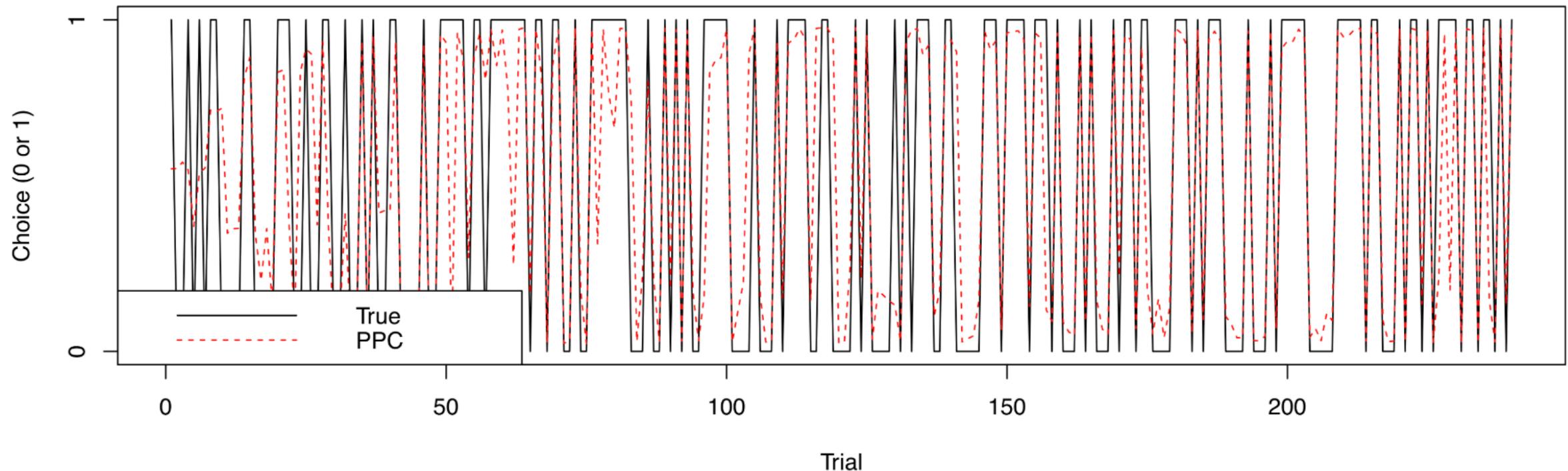
```
## [1] 10 240
```

```
# visualize SV (Subject #1)  
plot(sv_all[1, ], type="l", xlab="Trial", ylab="Stimulus Value (subject #1)")
```



Posterior predictive checks

```
# fit example data with the gng_m3 model and run posterior predictive checks  
x = gng_m3(data="example", niter=2000, nwarmup=1000, nchain=4, ncore=4, inc_postpred = TRUE)
```



Outline

- *Understand the concepts of computational modeling*
- *Know popular and cutting-edge methods for fitting computational models / Understand the concept of Bayesian data analysis*
- *Things to know when performing MCMC sampling*
- *Use hBayesDM package to model several tasks*

Popular methods for model fitting

- *Maximum likelihood estimation (MLE)*
- *Bayesian parameter estimation (or Bayesian data analysis)*
- *Hierarchical Bayesian analysis*

Maximum likelihood estimation

- Ronald A. Fisher (1920s): finds the value of a parameter that makes the observed data “most likely”.
- Likelihood function
 - $L(\text{data} \mid \theta)$: likelihood of getting data given θ
- Analytical solutions often don’t exist
- Nonlinear optimization
 - gradient decent
 - `fminsearch` (Matlab), `optim` (R), etc.
- MLE estimates = point estimates

Quantify uncertainty in parameter estimates

- In MLE
 - Bootstrapping: samples with replacement from the original data
 - Distribution of point estimates → classical measure of uncertainty
 - Hessian matrix
- In Bayesian
 - Plotting posterior distributions

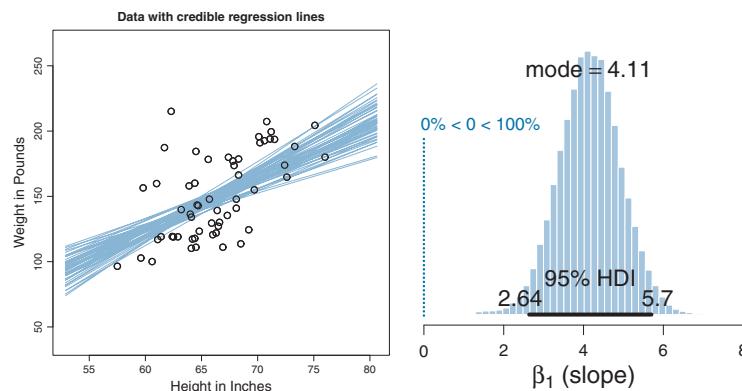
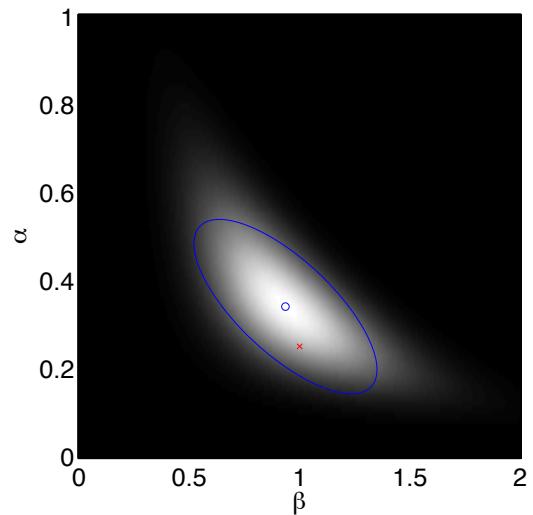


Figure 2.5 Data are plotted as circles in the scatter plot of the left panel. The left panel also shows a smattering of credible regression lines from the posterior distribution superimposed on the data. The right panel shows the posterior distribution of the slope parameter (i.e., β_1 in Equation 2.1).

Kruschke (2015)



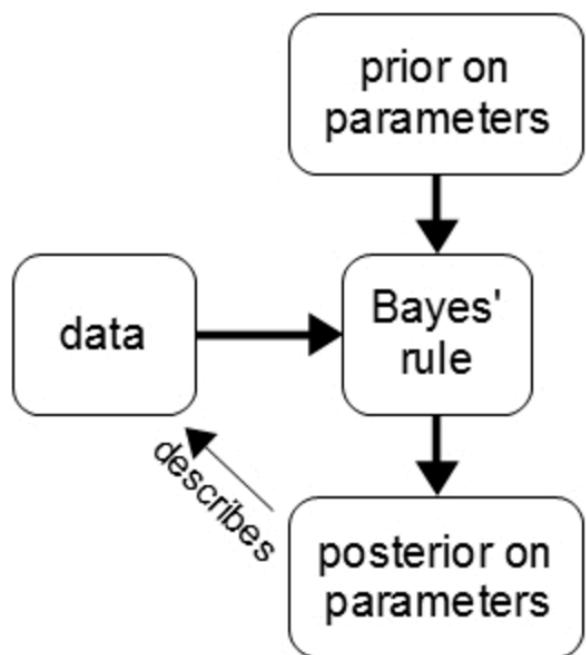
"Likelihood surface" from Daw (2011)

*Brief Introduction to
Bayesian data analysis*

Bayesian data analysis

Bayesian data analysis vs. *Bayesian model of mind*

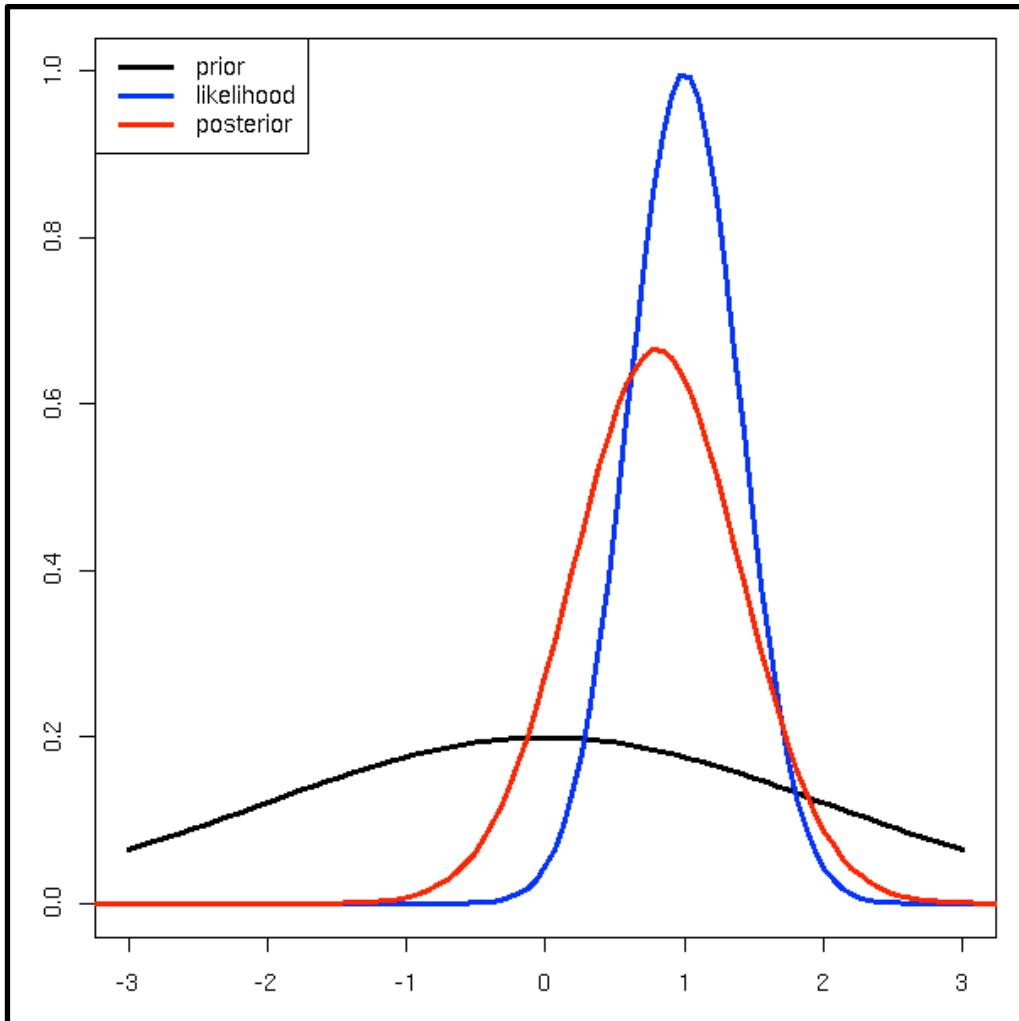
<http://doingbayesiandataanalysis.blogspot.hk/2011/10/bayesian-models-of-mind-psychometric.html>



Θ : Model parameters (e.g., discounting rate, learning rate, etc.)

A diagram illustrating the Bayesian formula. At the top, three boxes are labeled: "Posterior" (with an upward arrow), "Likelihood" (with an upward arrow), and "Prior" (with a red arrow pointing to a box). Below these, the Bayesian formula is shown: $P(\Theta_i | D_i) = \frac{P(D_i | \Theta_i)P(\Theta_i)}{P(D_i)}$. The term $P(D_i | \Theta_i)P(\Theta_i)$ is enclosed in a red box. The term $P(D_i)$ is enclosed in another red box. Below the formula, the word "Evidence" is written with a red arrow pointing downwards.

$$P(\Theta_i | D_i) = \frac{P(D_i | \Theta_i)P(\Theta_i)}{\int P(D_i | \Theta_{i'})P(\Theta_{i'})d\Theta_{i'}}$$



Bayesian data analysis

Update prior distributions for model parameters into posterior distributions given the data (e.g., trial-by-trial choices and outcomes) using Bayes' rule

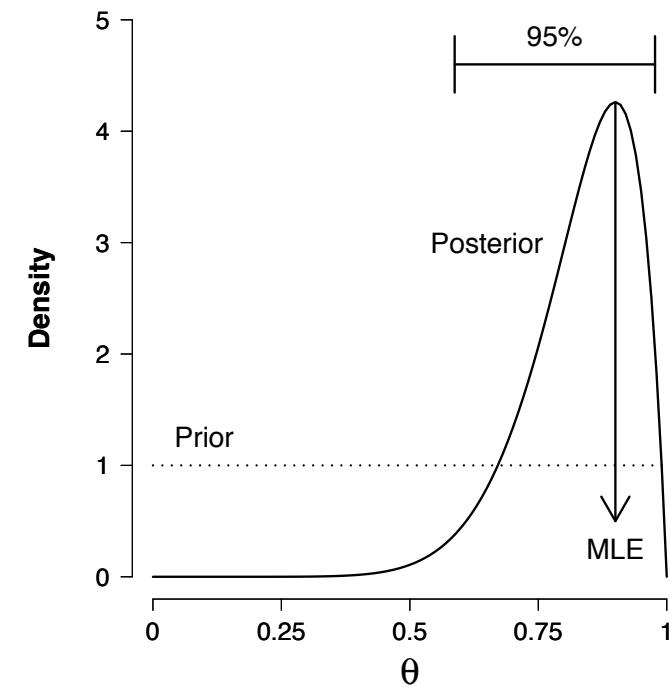
Relationship between MLE and Bayesian data analysis

An example from Lee & Wagenmakers (2012)

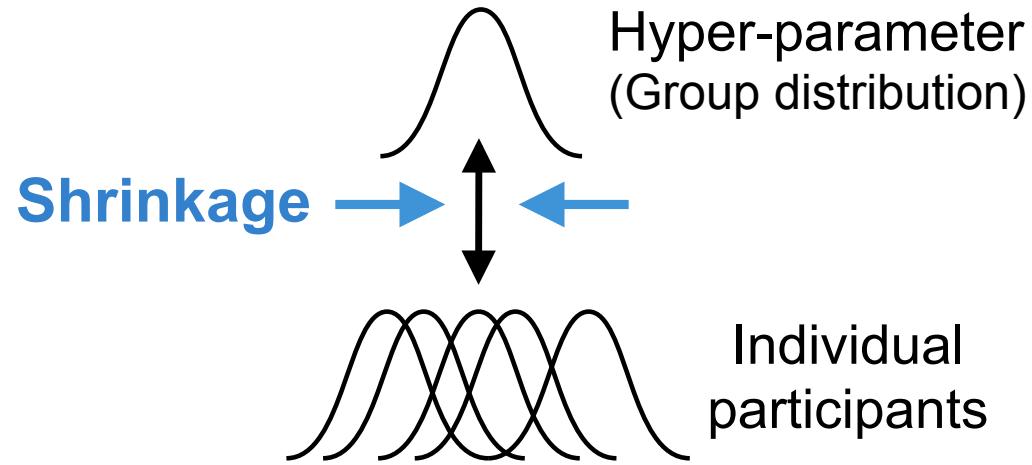
- *A test with 10 questions*
- *Parameter of interest: your ability ($0 < \theta < 1$)*
- *Data: 9 correct answers out of 10 ($k=9$, $n=10$)*
- *Likelihood function: Bernoulli*

Bayesian $p(\theta | D) = \frac{p(D | \theta) p(\theta)}{p(D)}$

MLE $\hat{\theta} = k/n = 0.9$



Hierarchical Bayesian analysis



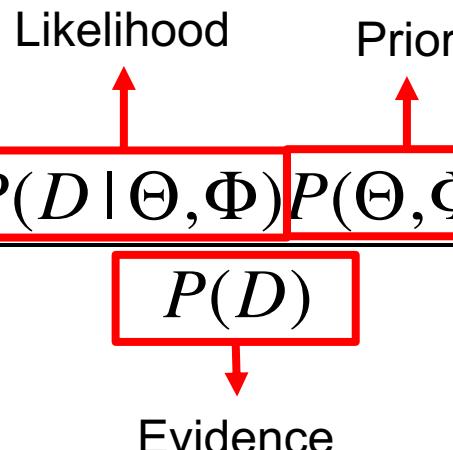
- Similarities & differences across participants
- Small amount of data from each subject

$$P(\Theta, \Phi | D)$$

Posterior

$$\Theta_i = \{\alpha_i, \beta_i, \gamma_i, \dots\}$$

$$\Phi = \{\mu_\alpha, \mu_\beta, \mu_\gamma, \sigma_\alpha, \sigma_\beta, \sigma_\gamma, \dots\}$$

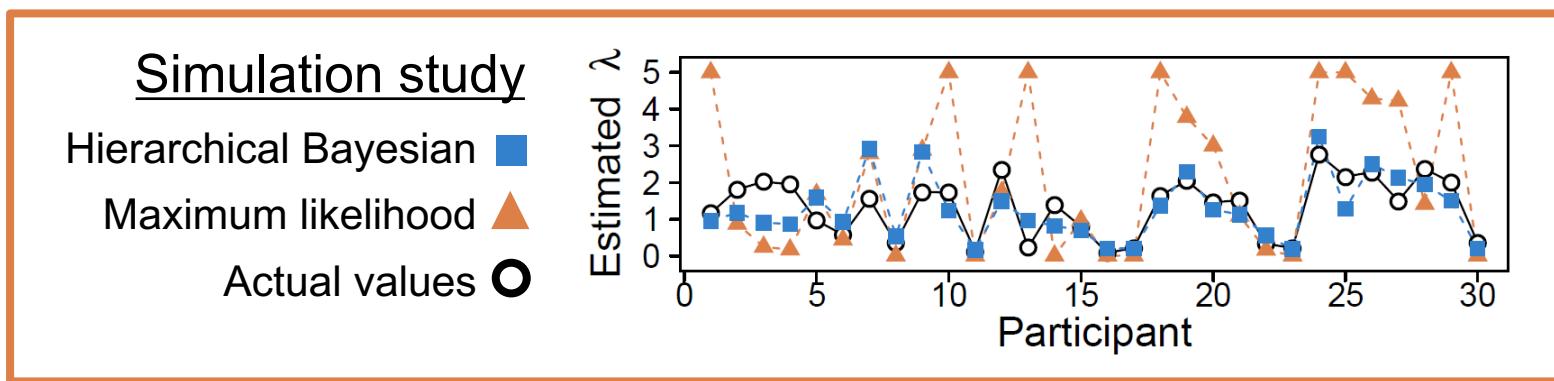


$$\frac{P(D | \Theta, \Phi)P(\Theta, \Phi)}{P(D)}$$

Evidence

$$\propto P(D | \Theta)P(\Theta | \Phi)P(\Phi)$$

Hierarchical Bayesian data analysis leads to more accurate parameter estimates



Ahn et al. (2011, JNPE)

Limitations

- Assumption of a single hyper-group
- Shrinkage → regression toward the group mean

Ahn, W.-Y., Krawitz, A., Kim, W., Busemeyer, J. R., & Brown, J. W. (2011). A Model-Based fMRI Analysis with Hierarchical Bayesian Parameter Estimation. *Journal of Neuroscience, Psychology, and Economics*, 4(2), 95-110.

Okay sounds good... but how can we actually do Bayesian updating?

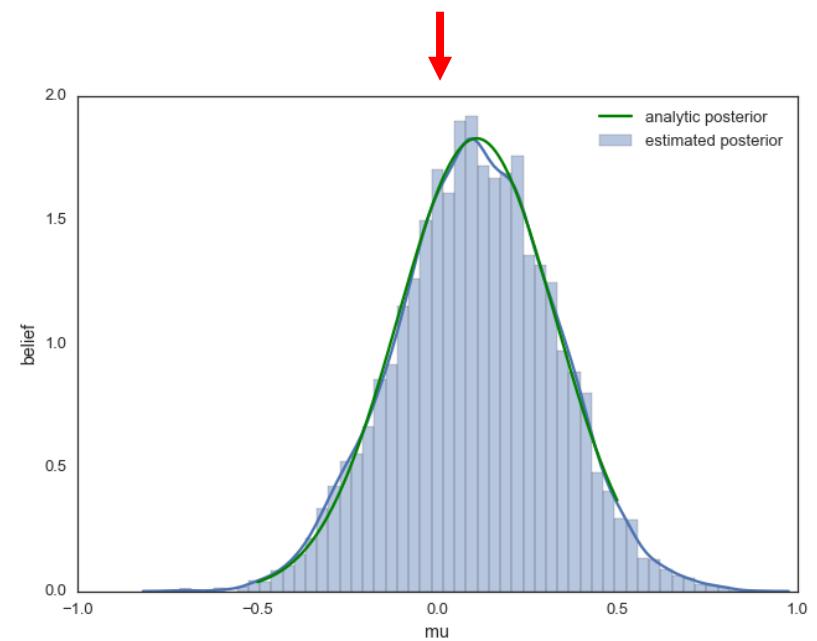
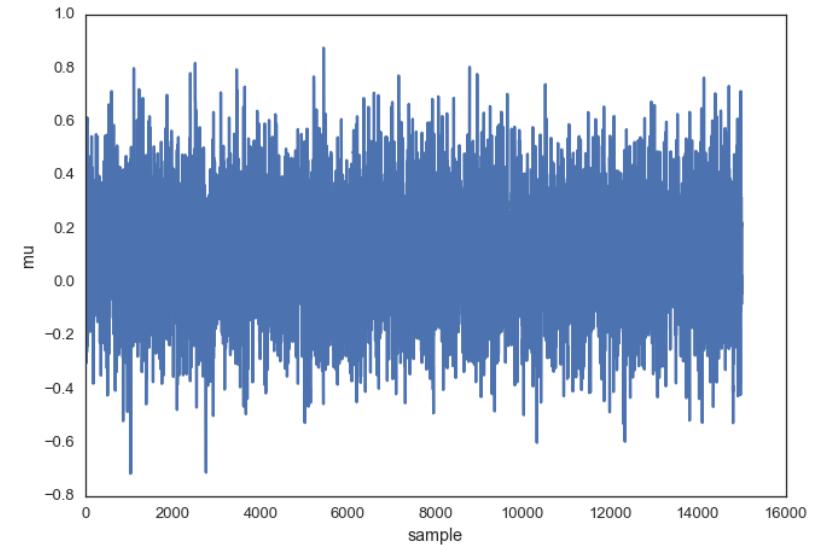
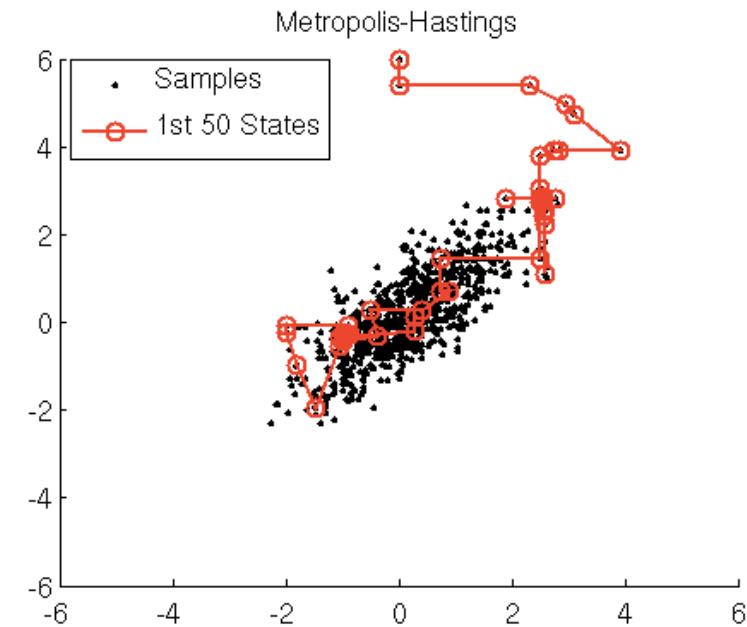
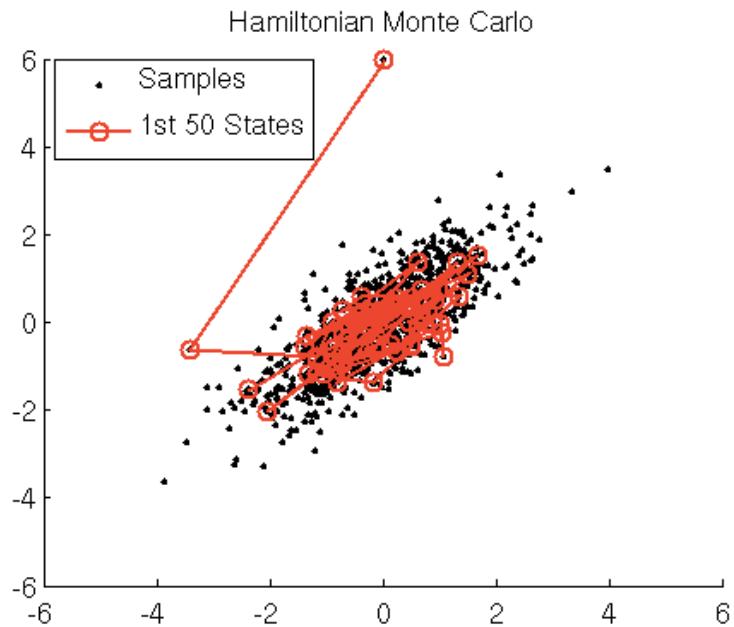
$$P(\Theta_i | D_i) = \frac{P(D_i | \Theta_i)P(\Theta_i)}{P(D_i)} = \frac{P(D_i | \Theta_i)P(\Theta_i)}{\int P(D_i | \Theta_{i'})P(\Theta_{i'})d\Theta_{i'}}$$

1. *Analytical solutions (often don't exist)*
2. *Grid approximation*
3. *Variational Bayes*
4. *Markov Chain Monte Carlo (MCMC)*

Markov Chain Monte Carlo (MCMC)

Approximate posterior distributions by drawing samples from the posterior distributions (after reaching an equilibrium).

Some samplers are more efficient than others!



<http://twiecki.github.io/blog/2015/11/10/mcmc-sampling/>

Tools for MCMC sampling

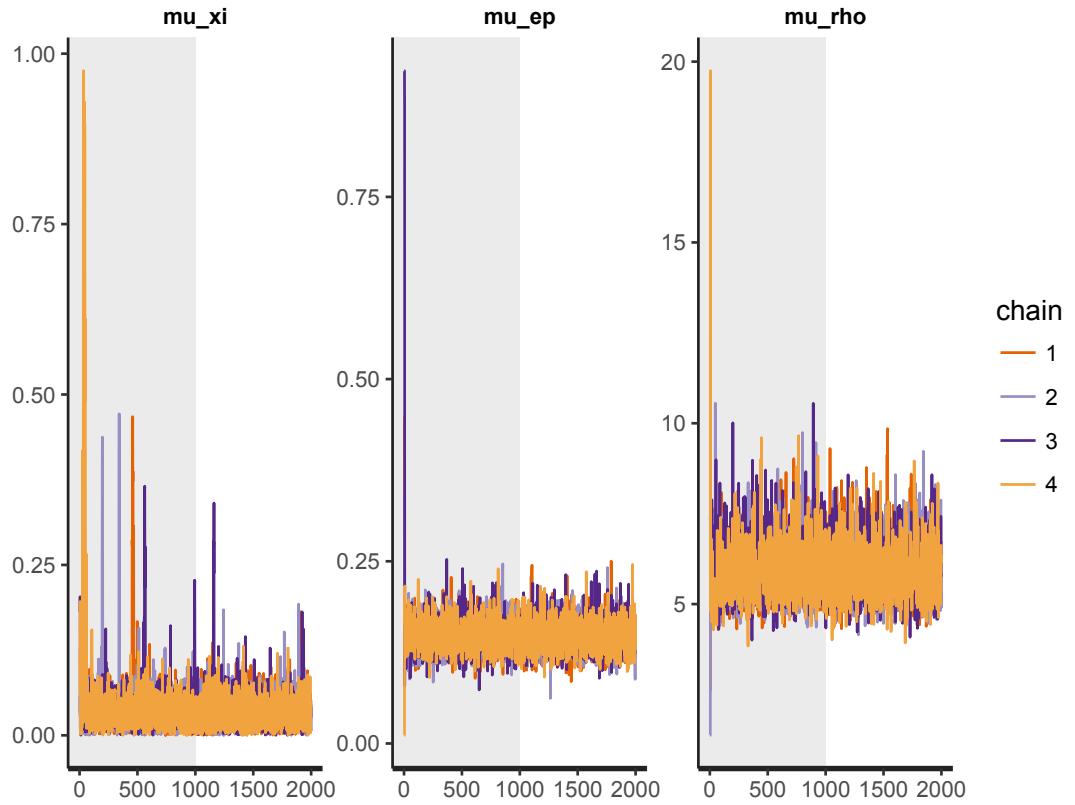
1. *Program your own!?*
2. *User-friendly packages*
 - *WinBUGS/OpenBUGS*
 - *JAGS*
 - *PyMC3 (in Python)*
 - *Stan*
3. *hBayesDM utilizes Stan (<http://mc-stan.org/>), which is well-maintained by a team led by Andrew Gelman.*
4. *Stan & hBayesDM support VB as well!*

Outline

- *Understand the concepts of computational modeling*
- *Know popular and cutting-edge methods for fitting computational models / Understand the concept of Bayesian data analysis*
- ***Things to know when performing MCMC sampling***
- *Use hBayesDM package to model several tasks*

Things to know when performing MCMC sampling

“Beware: *hBayesDM* can be dangerous!”

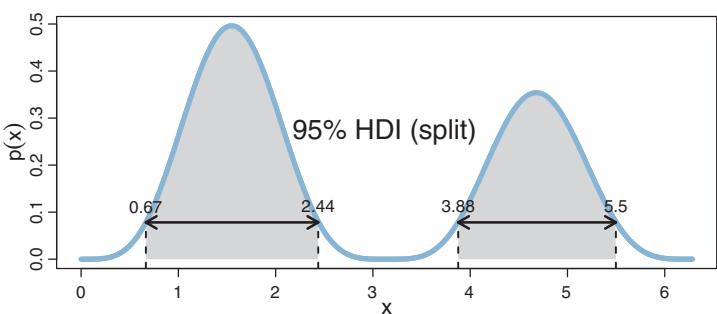
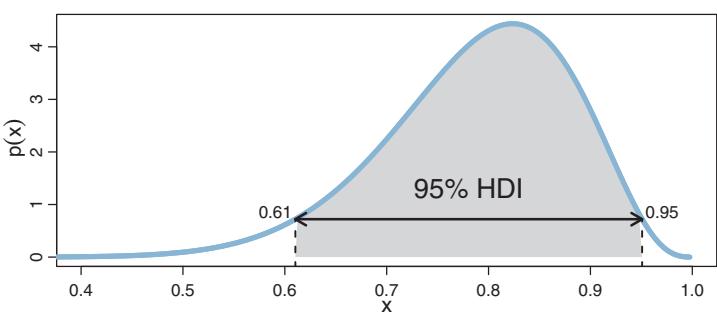
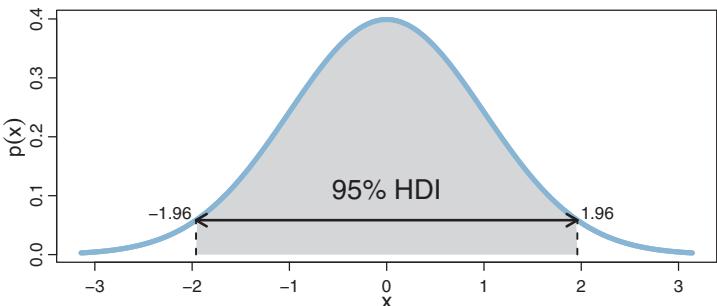


Check the convergence!

- Number of chains → at least 4
- Burn-in (warm-up) samples
- Convergence index (e.g., \hat{R}) → Close to 1.00
At least less than ~1.10

```
plot(output1, type="trace", inc_warmup=T)
```

Highest Density Interval (HDI)

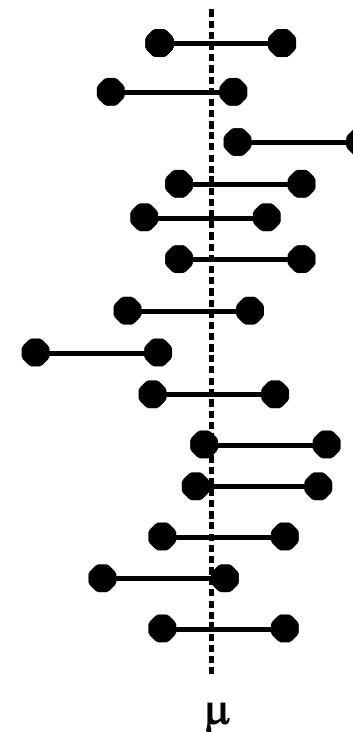


- “*Which points of a distribution are most credible*”
- *Skewed vs non-skewed*

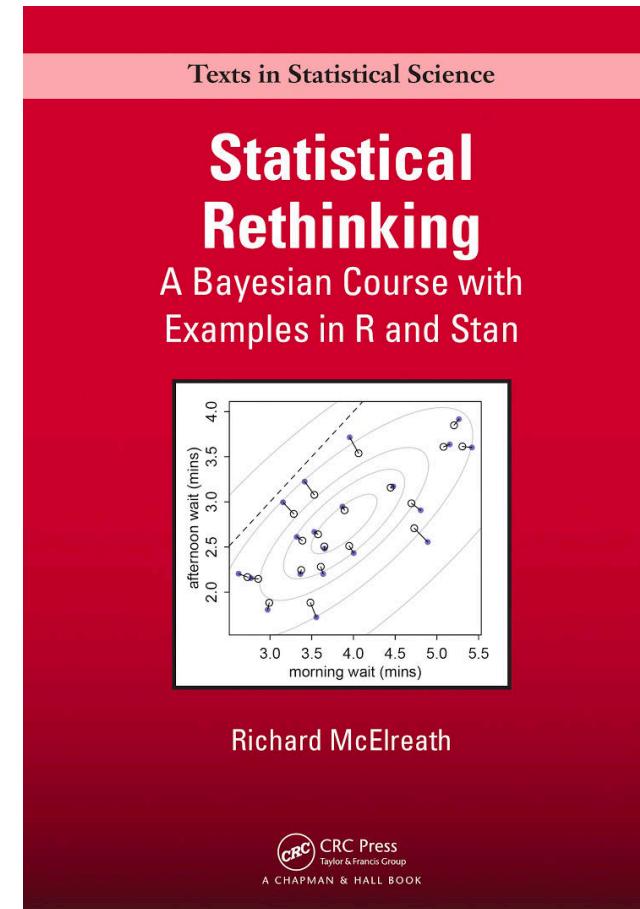
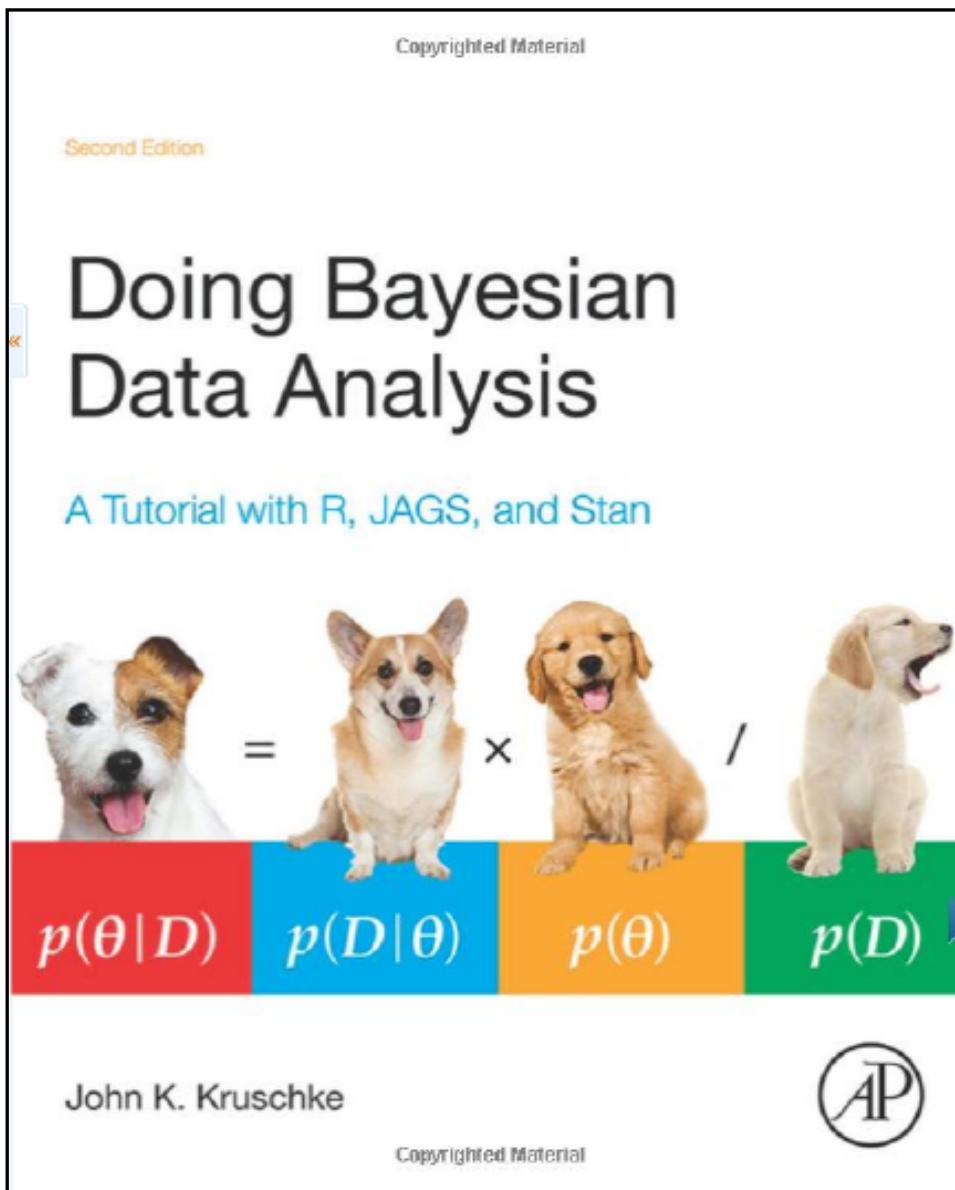
*It is important to note that a HDI is an **actual probability** interval, unlike a confidence interval in frequentist statistics.*

Interpretation of a BCI: $0.09 < \theta < 0.81$

“The probability that the parameter falls between 0.09 and 0.81 is 0.95.”



Doing Bayesian Data Analysis, 2nd ed. (2014)



Outline

- *Understand the concepts of computational modeling*
- *Know popular and cutting-edge methods for fitting computational models*
- *Understand the concept of Bayesian data analysis*
- *Evaluate outputs from Bayesian data analysis*
- ***Use hBayesDM package to model several tasks (Part II)***
 - *If you learn how to model one task, you know how to model all!*