

Computational Modeling of Decision-Making Tasks With a Single Line of Coding:

Modeling Can Be as Easy as Doing a T-Test

Woo-Young (Young) Ahn & Nate Haines



Computational Clinical Science Lab
Department of Psychology
The Ohio State University
<https://ccs-lab.github.io/>



<https://github.com/ccs-lab/aps2017-workshop>

General Outline

*Part I: Introduction to computational modeling and
hBayesDM (hierarchical Bayesian modeling of
Decision-Making tasks)*



*Part II: Hands-on tutorials on Bayesian modeling /
hBayesDM*



github.com/ccs-lab/aps2017-workshop

CCS-Lab / APS2017-workshop

Code Issues 0 Pull requests 0 Projects 0 Insights ▾

"Computational Modeling of Decision-Making Tasks With a Single Line of Coding: Modeling Can Be as Easy as Doing a T-Test"

aps2017 workshop hbayesdm

27 commits 1 branch 0 releases 2 contributors GPL-3.0

Branch: master ▾ New pull request Find file Clone or download ▾

youngahn committed on GitHub Update README.md

Data --all
R_Codes/tutorial removed answer to gng question (will upload later)
LICENSE Initial commit
README.md Update README.md

Clone with HTTPS ⓘ
Use Git or checkout with SVN using the web URL.
<https://github.com/CCS-Lab/APS2017-workshop> ↗

Open in Desktop Download ZIP 2 days ago

README.md

Learning objectives

Participants will...

- *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 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*

Software installation (1/2)

- *Install the latest R (+3.3.2):* <https://cloud.r-project.org/>
- *Install RStudio:*
<https://www.rstudio.com/products/rstudio/download2/>

Windows users

- *Install Rtools:* <https://cran.r-project.org/bin/windows/Rtools/Rtools34.exe>
- *Carefully follow the instructions:* <https://github.com/stan-dev/rstan/wiki/Install-Rtools-for-Windows>

Mac users

- *Install Xcode* (<https://developer.apple.com/xcode/>)

Software installation (2/2)

hBayesDM tutorial: <http://rpubs.com/CCSL/hBayesDM>

“How to install hBayesDM” (Google “hBayesDM rpubs”)

Windows users

- *Install hBayesDM from CRAN. Takes ~10 seconds.*

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

Mac/Linux users

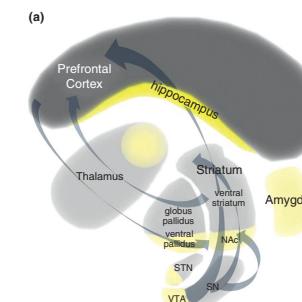
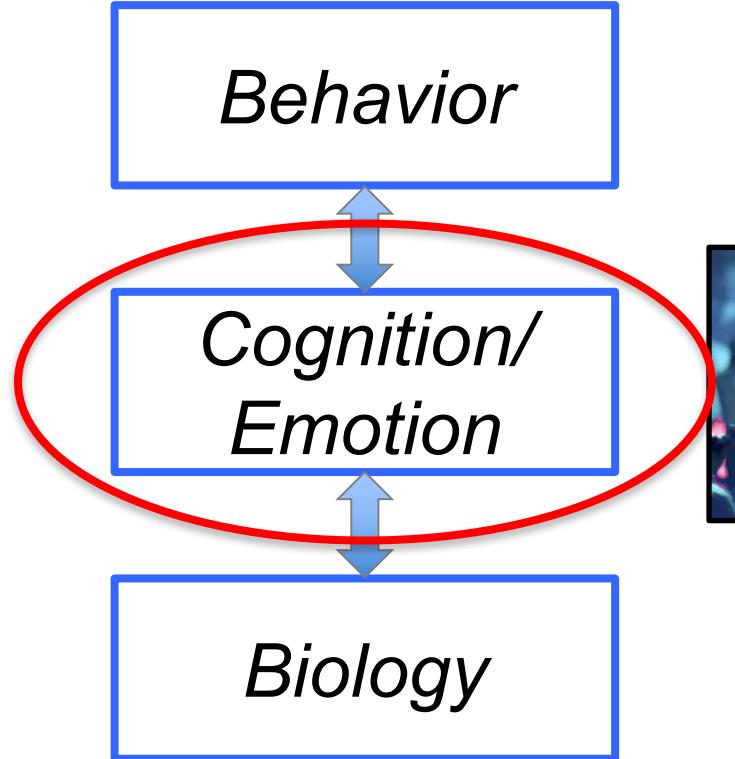
- *Install hBayesDM from GitHub. Takes ~5-10 minutes.*

```
# install 'devtools' if required
if (!require(devtools)) install.packages("devtools")
devtools::install_github("CCS-Lab/hBayesDM")
```

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*

How to study latent (hidden) processes?



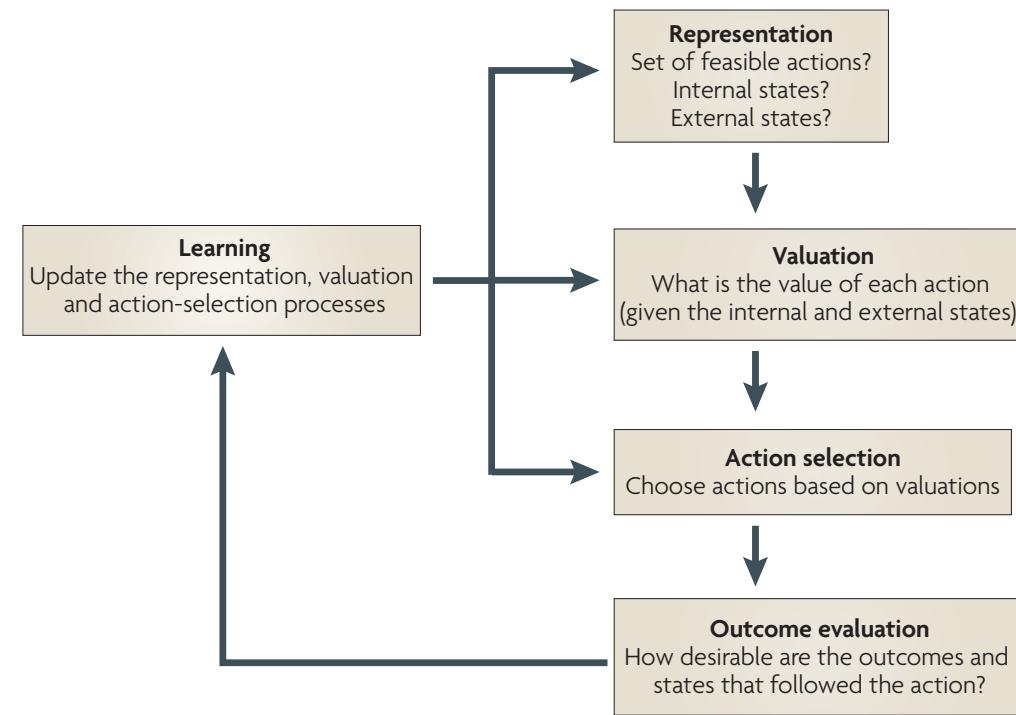
Volkow et al (2013)
Curr Opin Neurobiol

Reinforcement learning and Decision-making (RLDM)

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

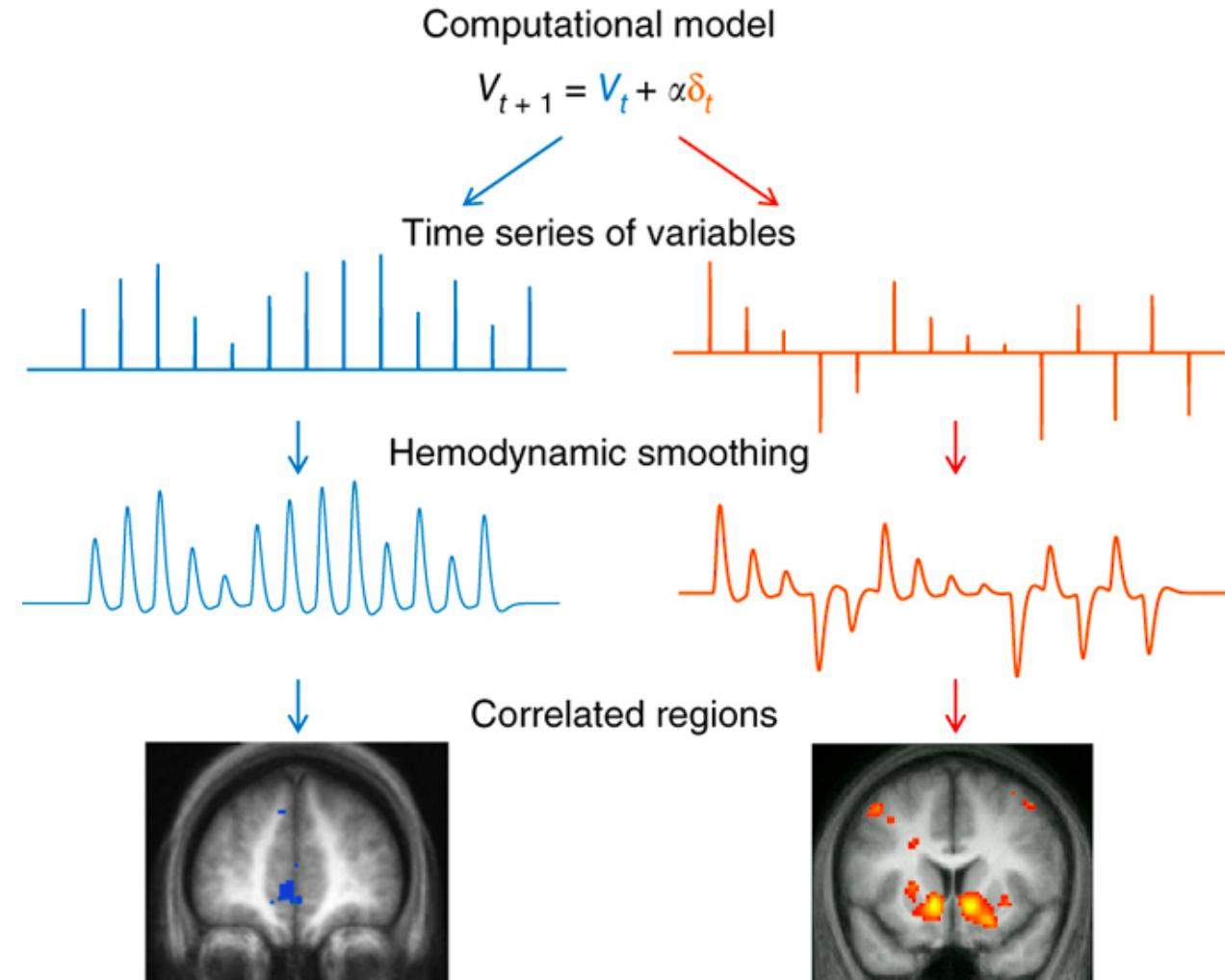
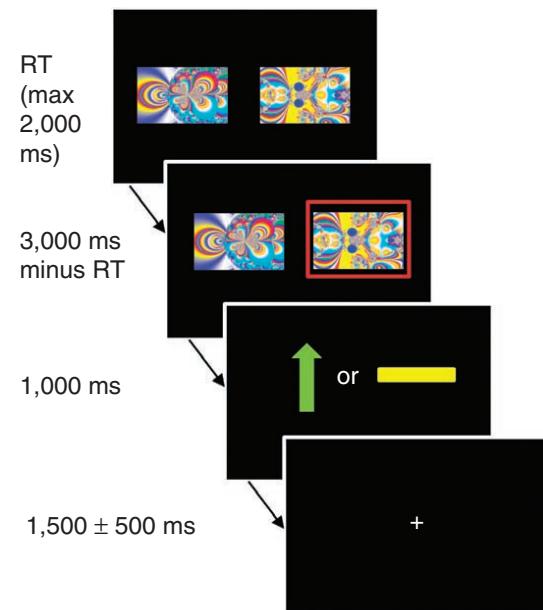
Antonio Rangel*, Colin Camerer* and P. Read Montague†

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

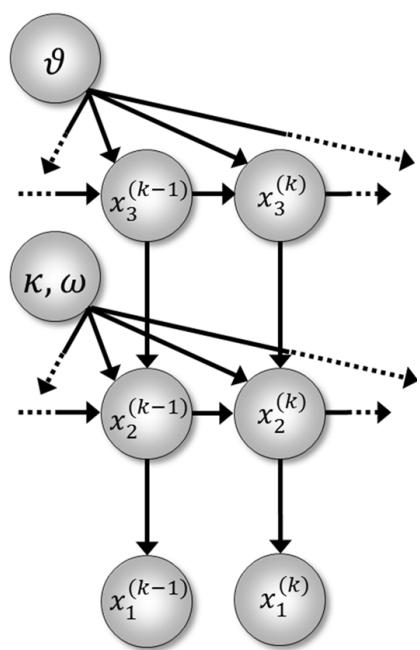
Computational psychiatry

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

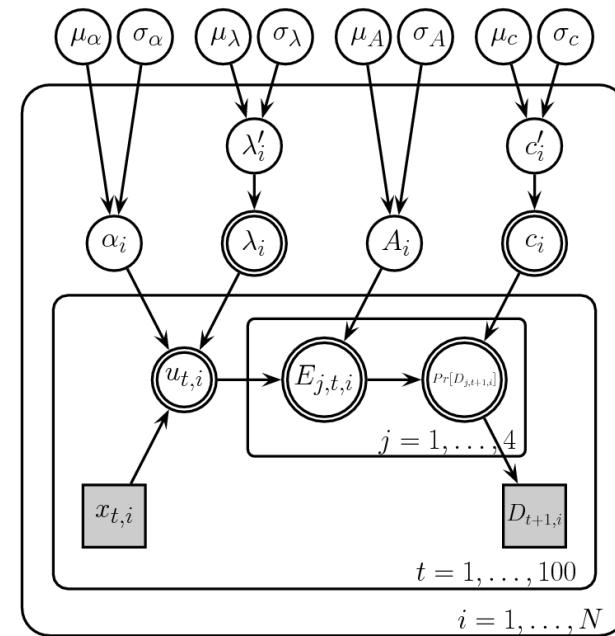
*Computational accounts of
abnormal cognition &
its biological underpinnings*

The screenshot shows the homepage of the journal *nature*. The header includes the title "nature" and the subtitle "International weekly journal of science". Below the header is a navigation bar with links: Home, News & Comment, Research, Careers & Jobs, Current Issue, Archive, Audio & Video, and a search icon. A secondary navigation bar shows the current path: Archive > Volume 539 > Issue 7627 > News: Q&A > Article. The main content area features a news article titled "NATURE | NEWS: Q&A" with the headline "US mental-health chief: psychiatry must get serious about mathematics".

I like the idea of modeling



But...



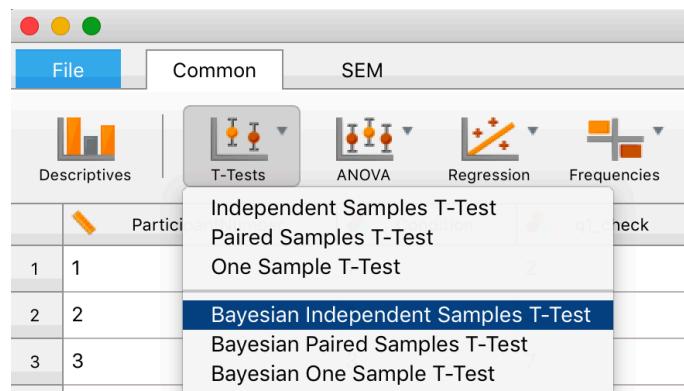
Mathys et al (2011) Frontiers

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)
```

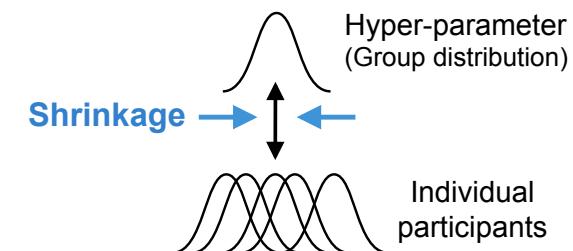
Several packages exist, but ...

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 9 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?

Ahn & Busemeyer (2016) Curr Opin Behav Sci

- *Choice reaction time*
- *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*)
- **Two-Step* (*Daw et al, 2011*)

*Version 0.4.1. or later

How can I use it?

Tutorials available at

<http://rpubs.com/CCSL/hBayesDM>

(you can find it by Googling ‘hBayesDM’)

Install it just like other R packages

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

```
devtools::install_github("CCS-Lab/hBayesDM")
```

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	nogoto avoid losing 	nogoto win 

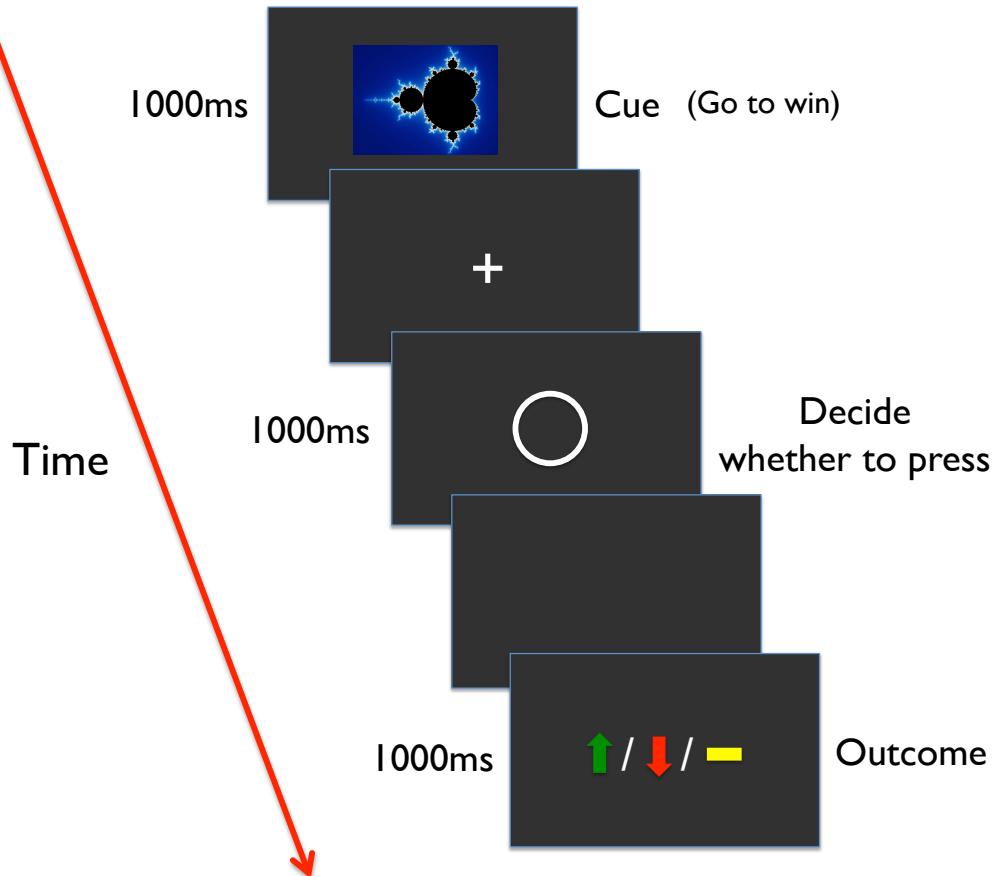
Orthogonalized Go/Nogo task

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



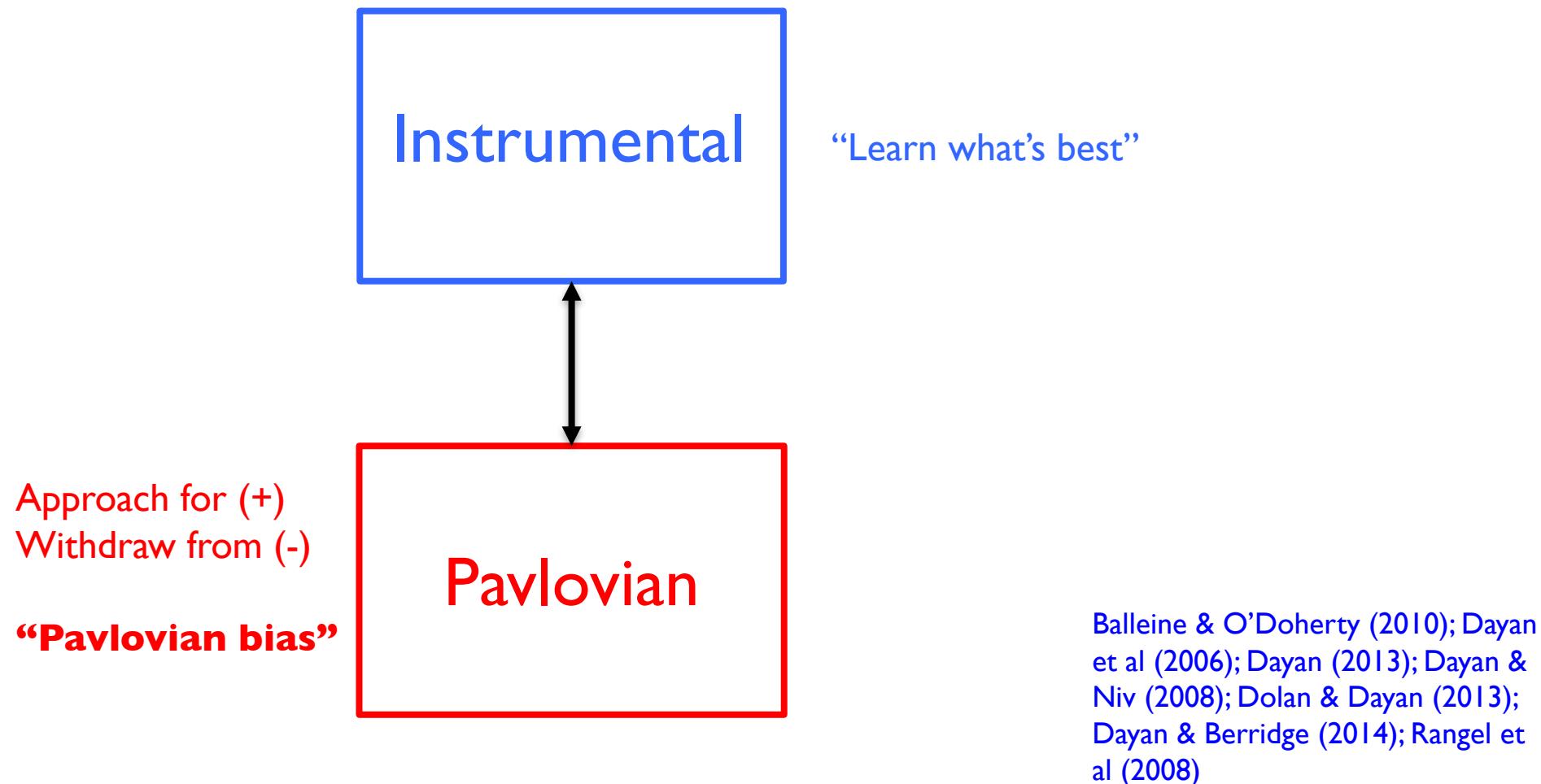
- 4 cues (conditions)

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



Orthogonalized Go/Nogo task

Pavlovian-Instrumental competition



Pavlovian-Instrumental competition

Nogo

*Instrumental
response*



Young on diet

Go

*Pavlovian
response*

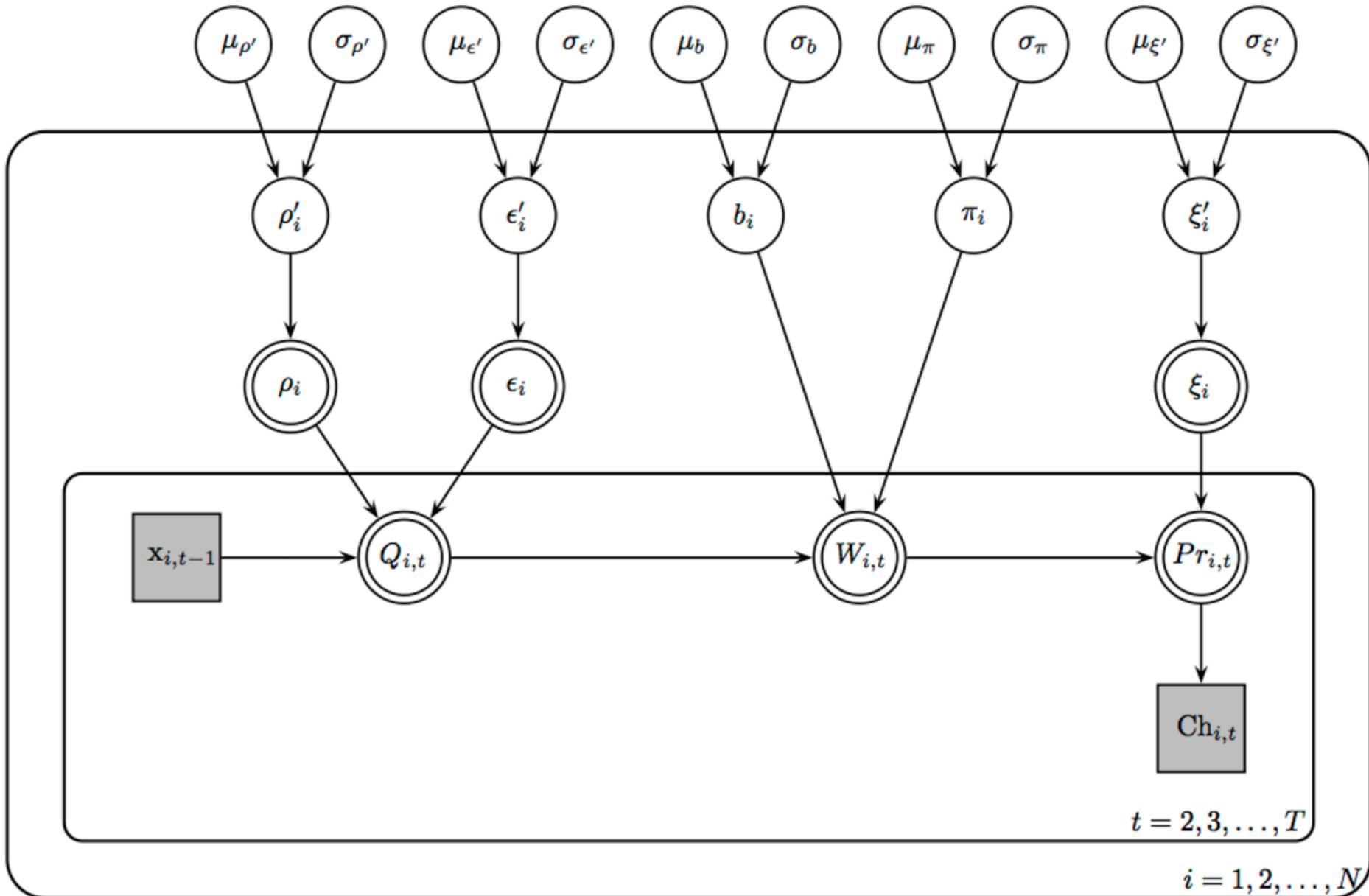


Instrumental goal: To lose weight

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

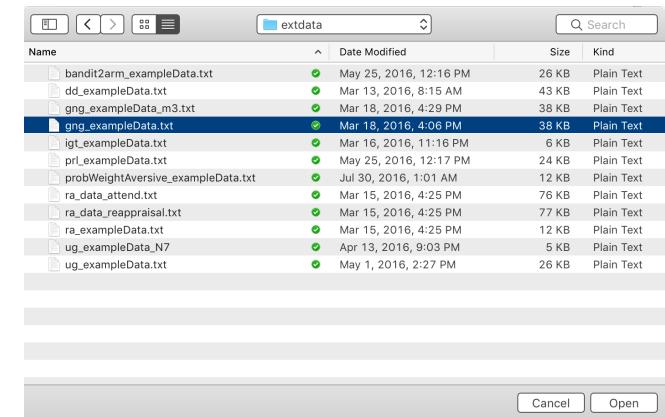


2. Fit candidate models

```
output1 = gng_m1(data="example", niter=2000, nwarmup=1000, nchain=4, ncore=4)
```

```
output1 = gng_m1("example", ncore=4)
```

```
output1 = gng_m1()
```



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)

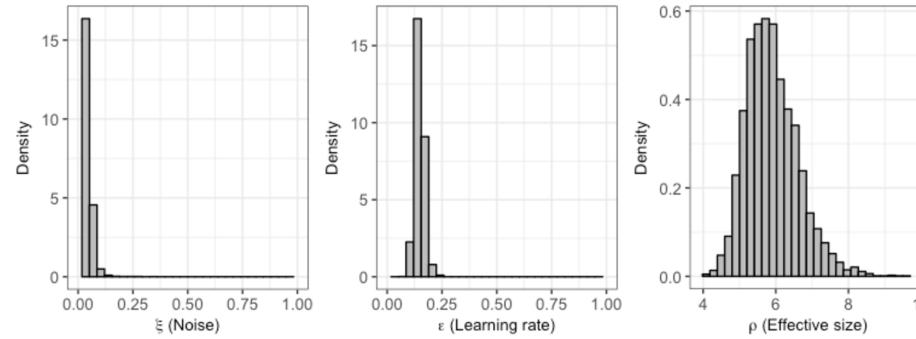
SAMPLING FOR MODEL 'gng_m1' NOW (CHAIN 2).

...

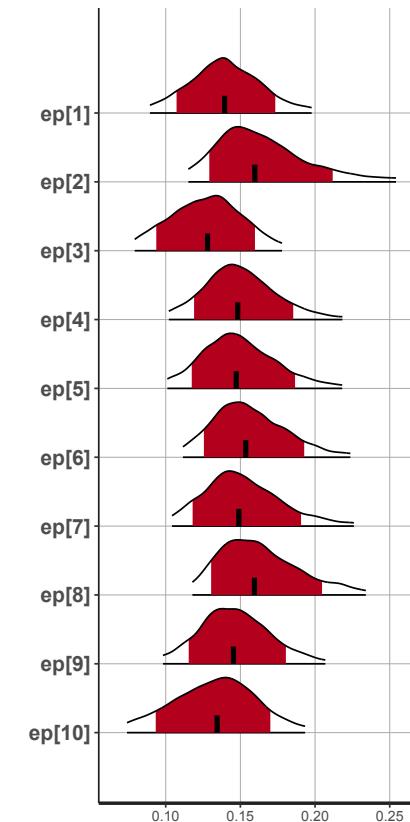
**** Model fitting is complete! ****

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
...				

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      LOOIC
 1 gng_m1  1588.843
 2 gng_m2  1571.129
 3 gng_m3  1573.872
 4 gng_m4  1543.335
```

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

5. Group comparisons

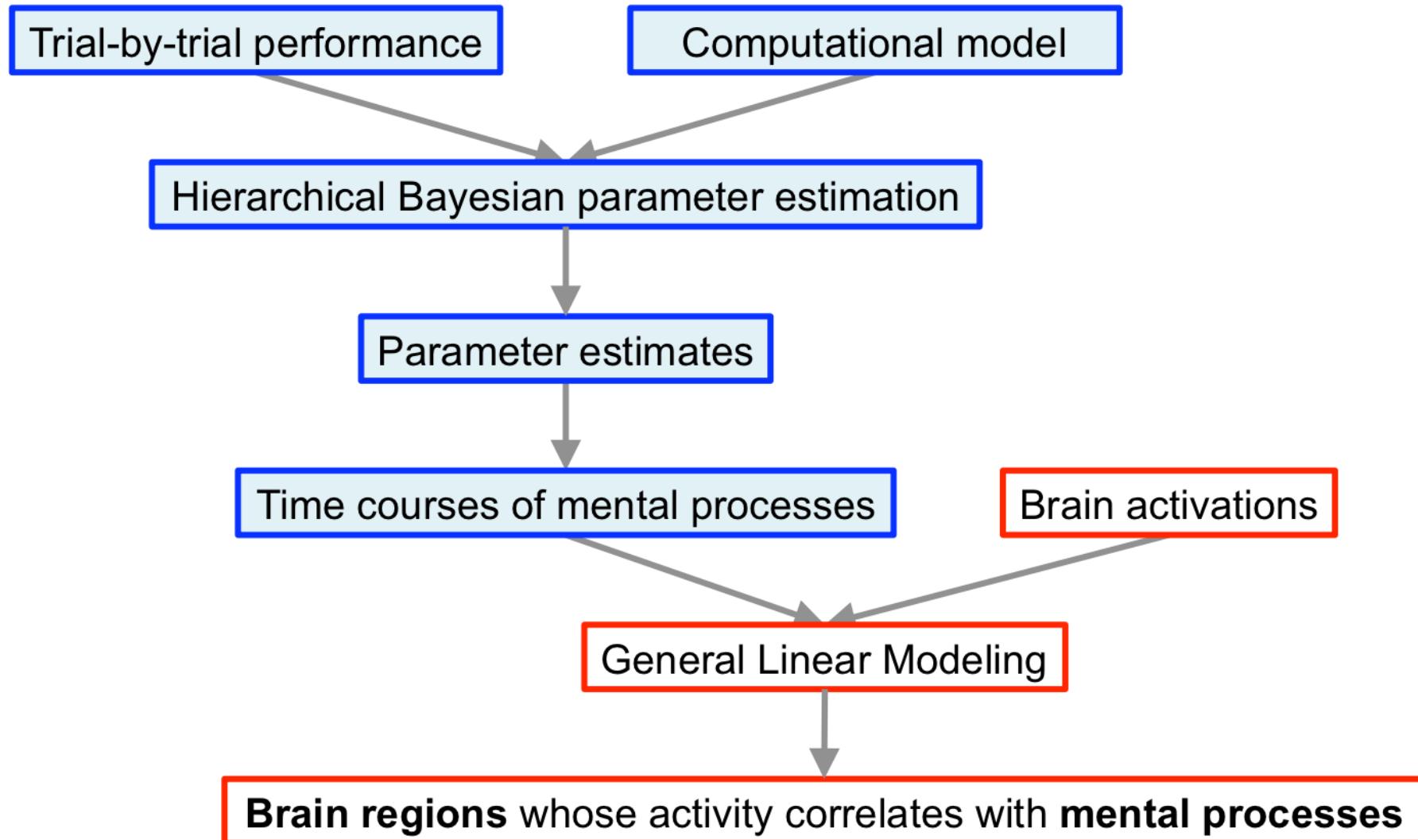
```
data_group1 = "~/Project_folder/gng_data_group1.txt" # data file for group1
data_group2 = "~/Project_folder/gng_data_group2.txt" # data file for group2

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

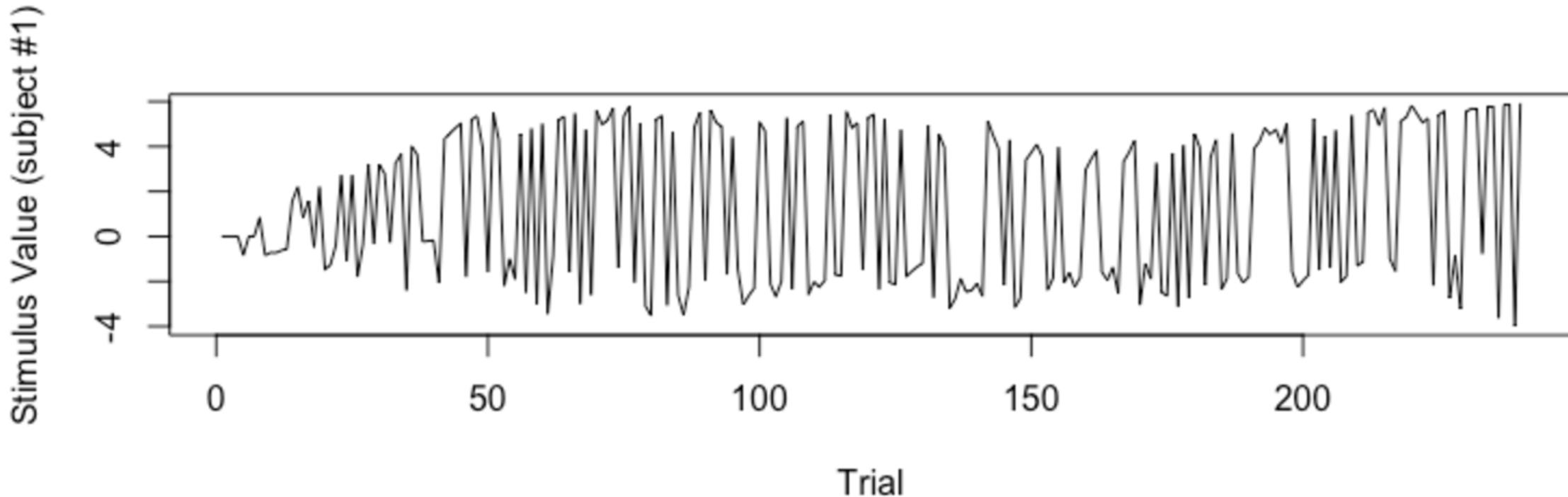
- *Trial-by-trial regressors are readily available*

```
# 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)")
```



Outline

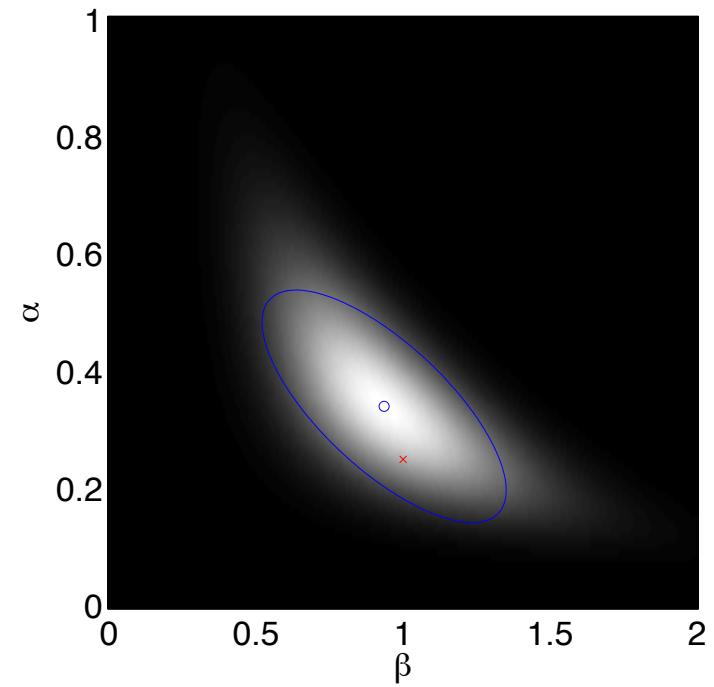
- *Understand the concepts of computational modeling*
- *Know popular and cutting-edge methods for fitting computational models / Understand the concept of Bayesian data analysis*
- *Things to know when performing MCMC sampling*
- *Use hBayesDM package to model several tasks*

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



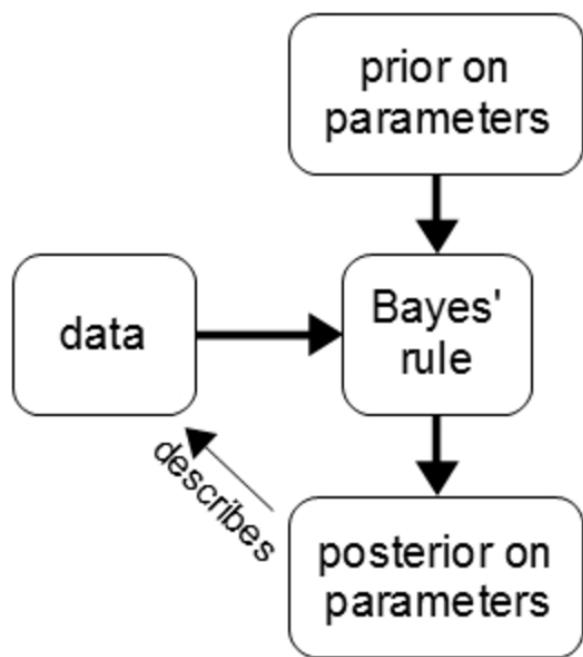
“Likelihood surface” from Daw (2011)

*Brief Introduction to
Bayesian data analysis*

Bayesian data analysis

Bayesian data analysis vs. *Bayesian model of mind*

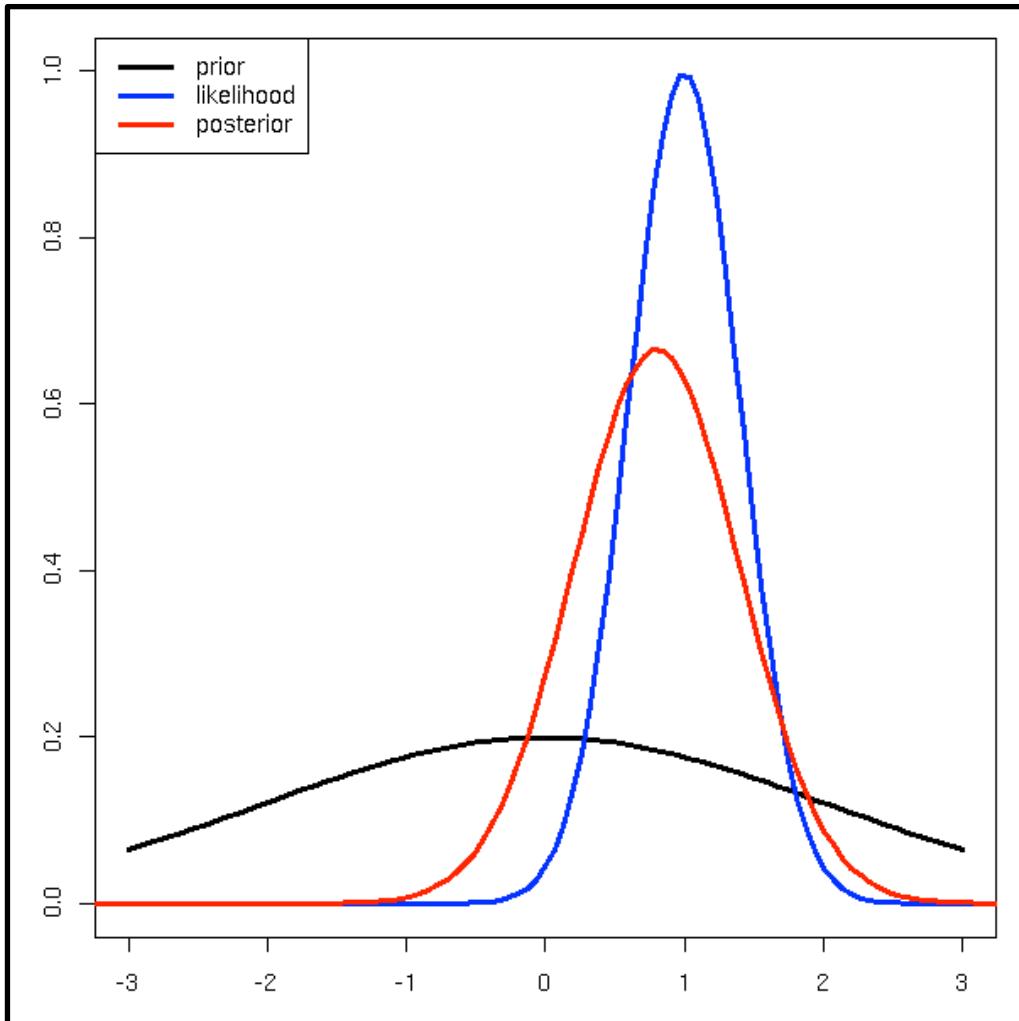
<http://doingbayesiandataanalysis.blogspot.hk/2011/10/bayesian-models-of-mind-psychometric.html>



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

A diagram illustrating the Bayesian formula. At the top, three boxes are labeled "Posterior", "Likelihood", and "Prior". Below them, the posterior probability $P(\Theta_i | D_i)$ is shown in a red-bordered box. To its right is the formula $P(\Theta_i | D_i) = \frac{P(D_i | \Theta_i)P(\Theta_i)}{P(D_i)}$, also enclosed in a red border. A red arrow points from the "Prior" box to the term $P(\Theta_i)$ in the numerator. Another red arrow points from the "Likelihood" box to the term $P(D_i | \Theta_i)$. A red arrow points from the "Evidence" box down to the term $P(D_i)$ in the denominator. The entire fraction is also enclosed in a red border.

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



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

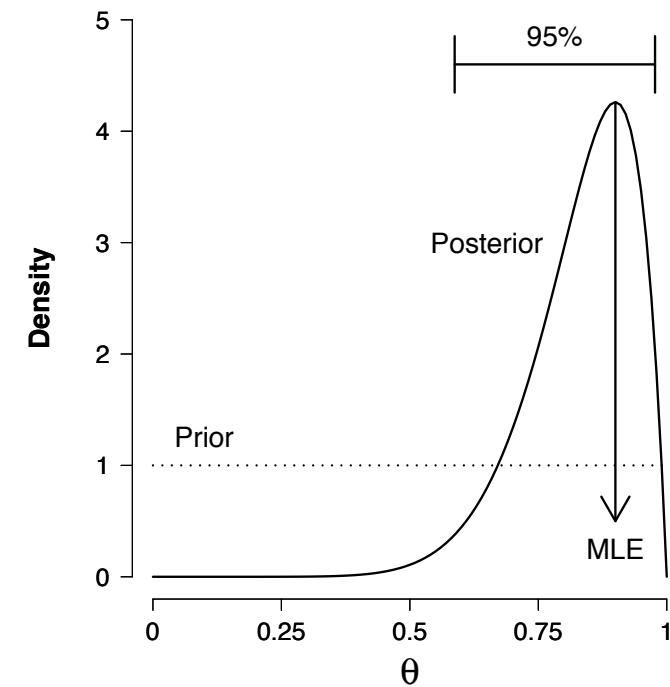
Relationship between MLE and Bayesian data analysis

An example from Lee & Wagenmakers (2012)

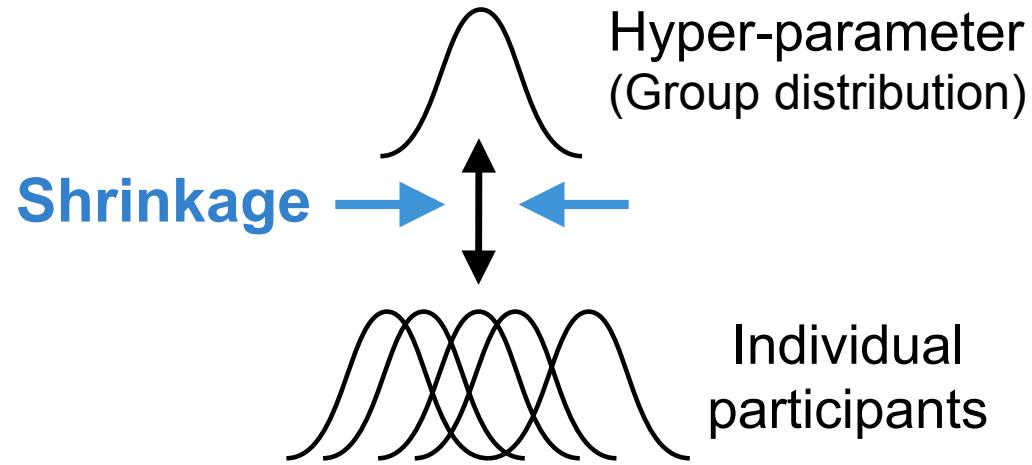
- *A test with 10 questions*
- *Parameter of interest: your ability ($0 < \theta < 1$)*
- *Data: 9 correct answers out of 10 ($k=9$, $n=10$)*
- *Likelihood function: Bernoulli*

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

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



Hierarchical Bayesian analysis



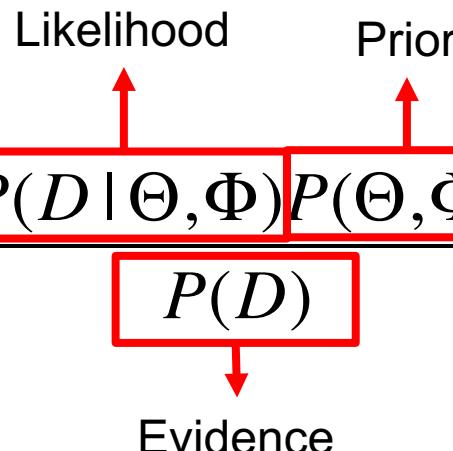
- Similarities & differences across participants
- Small amount of data from each subject

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

Posterior

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

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

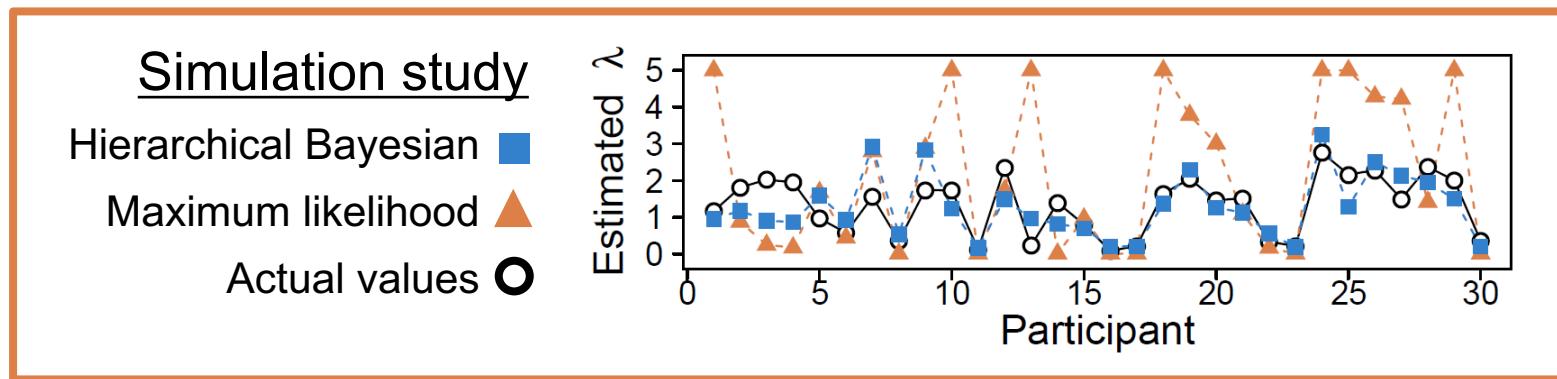


Evidence

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

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

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

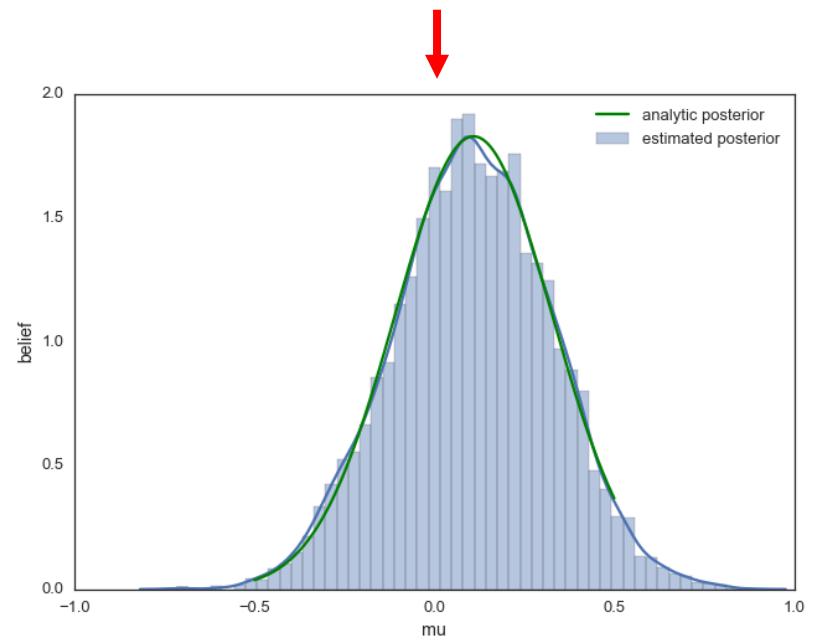
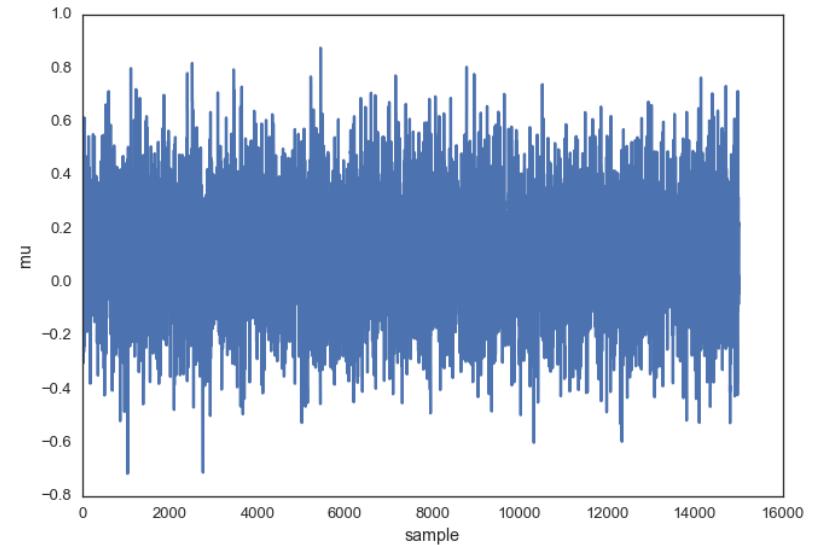
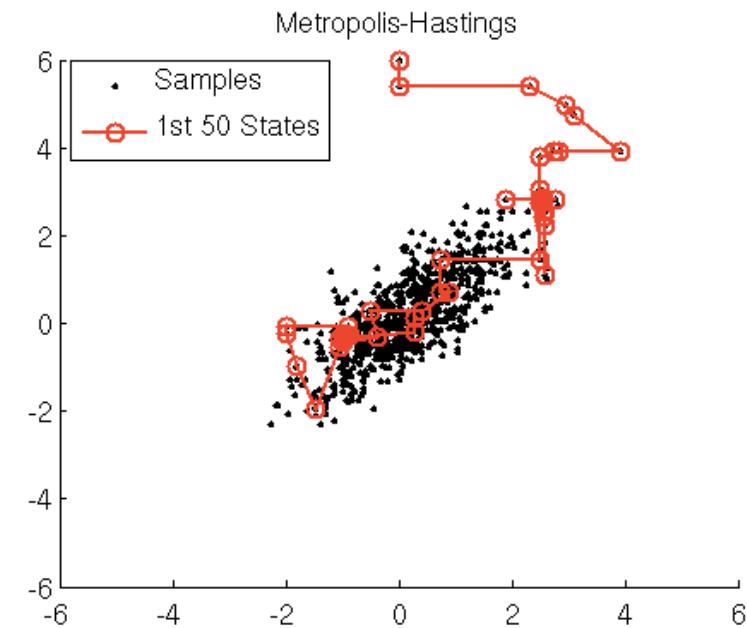
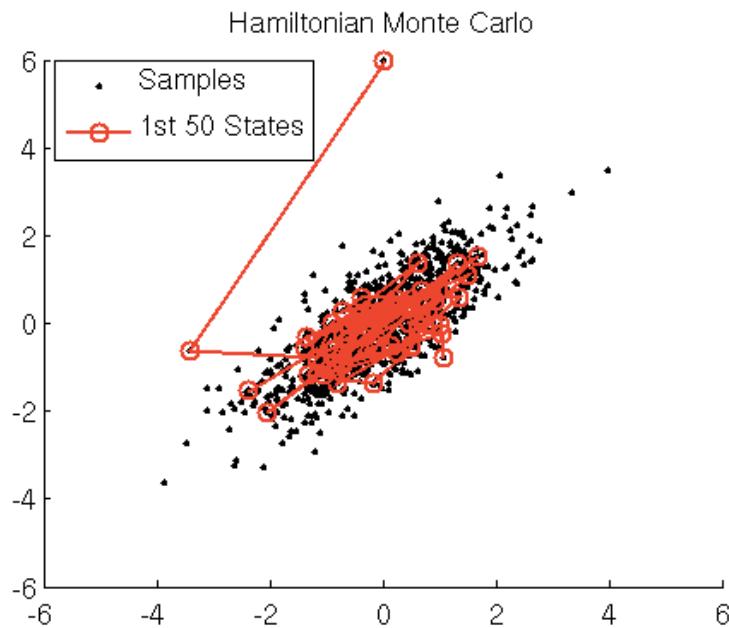
$$P(\Theta_i | D_i) = \frac{P(D_i | \Theta_i)P(\Theta_i)}{P(D_i)} = \frac{P(D_i | \Theta_i)P(\Theta_i)}{\int P(D_i | \Theta_{i'})P(\Theta_{i'})d\Theta_{i'}}$$

1. *Analytical solutions (often don't exist)*
2. *Grid approximation*
3. *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

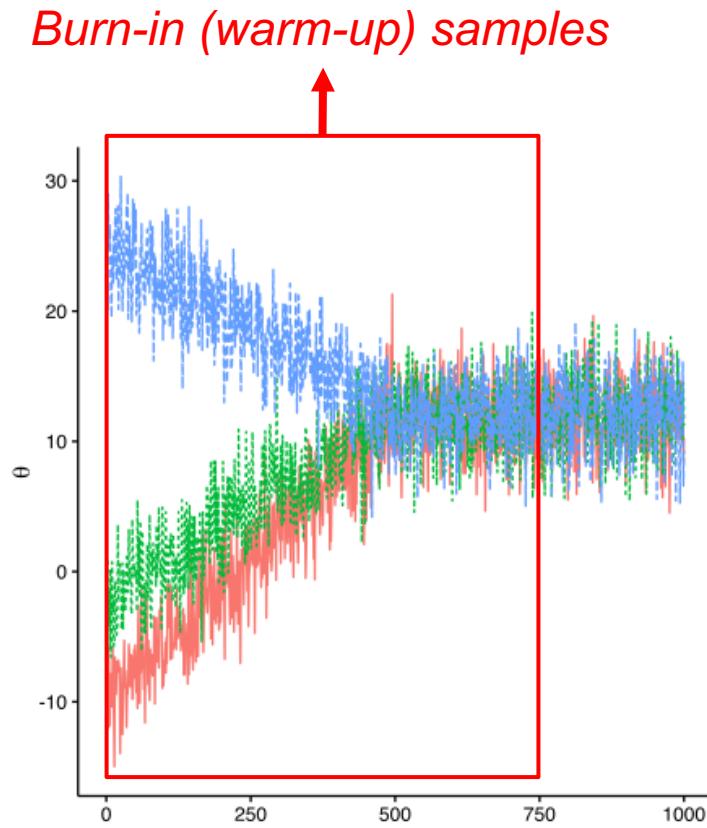
1. *Program your own!?*
2. *User-friendly packages*
 - *WinBUGS/OpenBUGS*
 - *JAGS*
 - *PyMC3 (in Python)*
 - *Stan*
3. *hBayesDM utilizes Stan (<http://mc-stan.org/>), which is well-maintained by a team led by Andrew Gelman.*

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!”

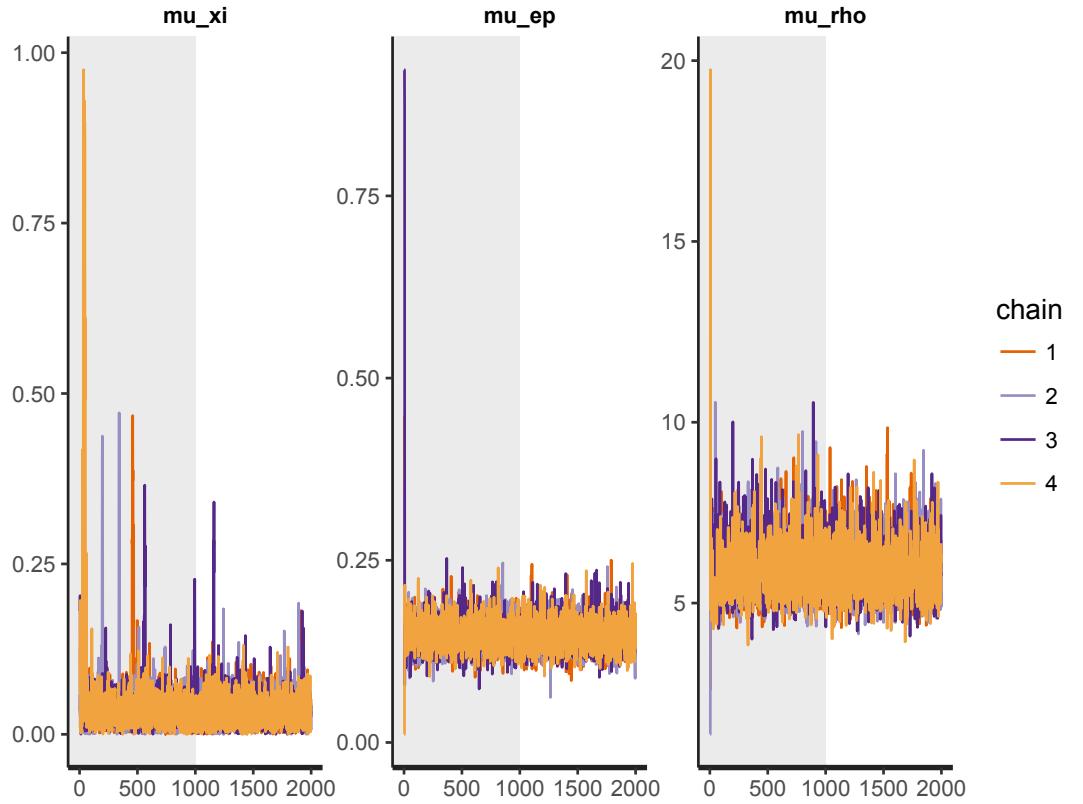


Check the convergence!

- 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

Things to know when performing MCMC sampling

“Beware: *hBayesDM* can be dangerous!”



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

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

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!*