COSMOS SUMMER SCHOOL

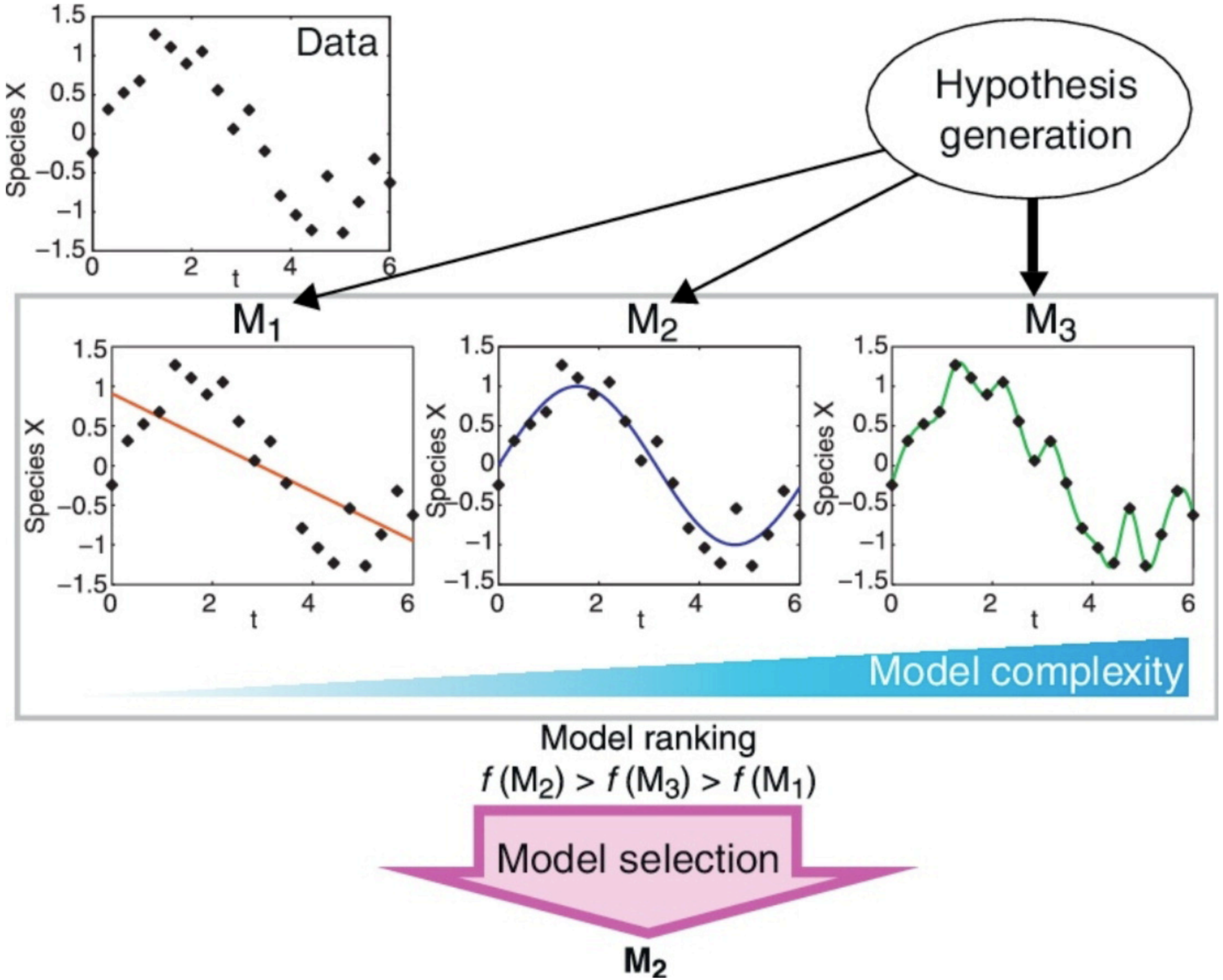
Computational summer school on modeling social and collective behavior
Tokyo Sept 29th - Oct 3rd, 2025

Tutorial 3: Model comparison and robustness

Charley Wu & Wataru Toyoka

Oct 2nd

Kirk, Thorne, & Stumpf (*Curr Opin Biotech* 2013)



1. Generate hypotheses
2. Build models for each hypothesis
3. Fit models to data
4. Determine the best model
5. Interpretation

Outline

Part 1. Model Comparison

Part 2. Robustness

Part 1. Model Comparison

What makes a good model?

essentially,
all models are wrong,
but some are useful

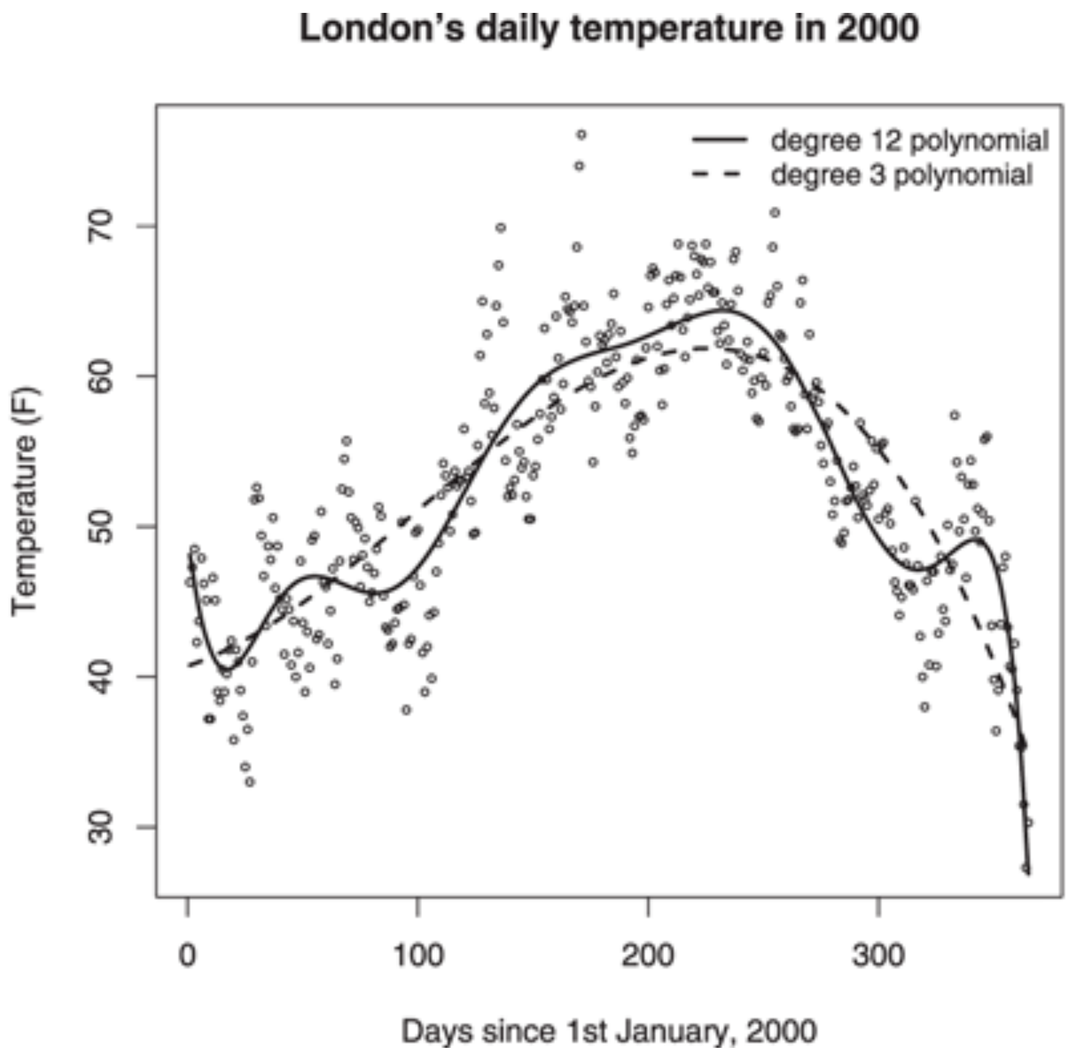
George E. P. Box

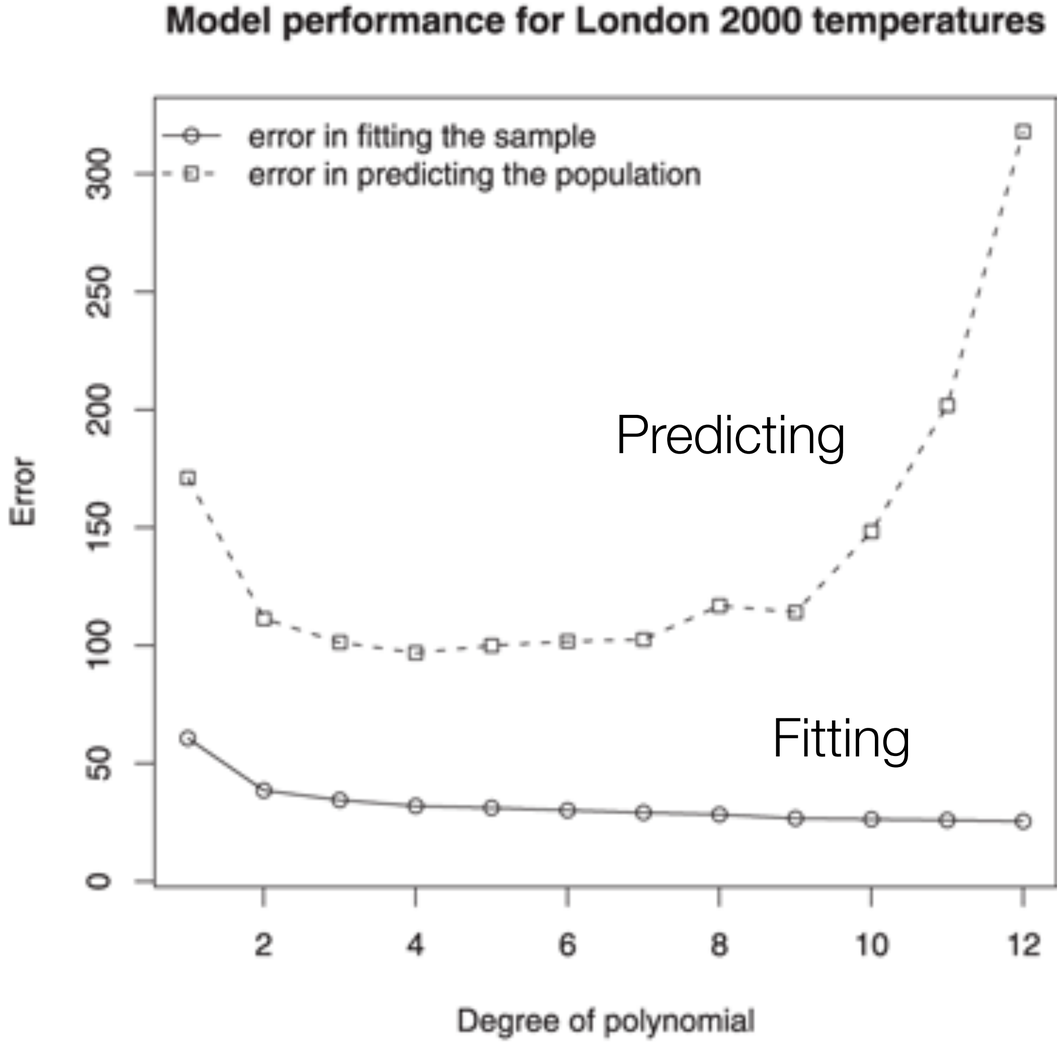
“As simple as possible, but not simpler”

… **In that empire, the art of cartography attained such perfection that […] the cartographers guilds struck a map of the empire whose size was that of the empire, and which coincided point for point with it**. The following generations, who were not so fond of the study of cartography as their forebears had been, saw that that vast map was useless, and not without some pitilessness was it, that they delivered it up to the inclemencies of sun and winters.

Jorge Luis Borges, **On Exactitude in Science**

“As simple as possible, but not simpler”





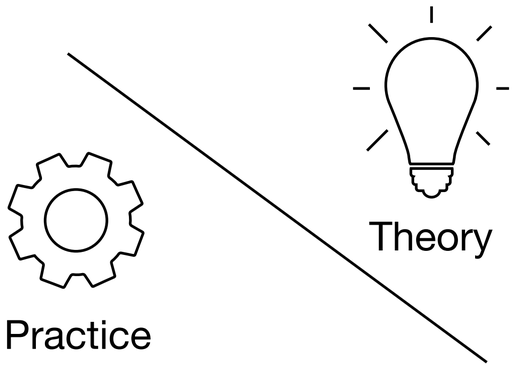
Gigerener & Brighton (*TopiCS*, 2009)

Goodness of Fit

Simplicity

Fit

Goodness of Fit Measures

	**Maximum Likelihood** $P(D \mid m, \hat{\theta})$	**Bayesian Model Selection** $\frac{P(D \mid m_1)}{P(D \mid m_2)}$
Penalizing for parameters	Akaike's Information Criterion (AIC)	Bayesian Information Criterion (BIC)
Prediction error/ Bayesian Occam's Razor	Cross-validation loss	Model evidence using Markov Chain Monte Carlo (MCMC)

Maximum likelihood estimation (MLE)

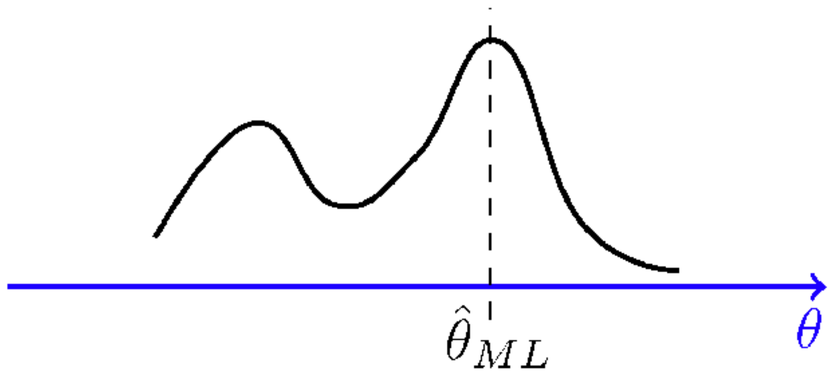
vs.

Bayesian model selection

- **Goal**: Quantify the goodness fit for a single set of parameter values $\hat{\theta}$ that provides the best fit to the data:

$$\arg\max_{\hat{\theta}} P(D \mid m, \hat{\theta})$$

- Overfitting is avoided by penalizing for the number of parameters (e.g., AIC) or using cross-validation to test predictive power



- **Goal**: quantify how well a given model m captures the data using the *marginal likelihood*:

$$P(D \mid m) = \int P(D \mid m, \theta) P(\theta \mid m) d\theta$$

- This integrates over all possible parameter values, allowing for a natural penalization of more complex models (i.e., Bayesian Occam's Razor)
  - You don't only test the model at it's best, but also at it's worse
- Intractable in most settings, so approximated using BIC or through MCMC sampling

Goodness of Fit Measures

	Maximum Likelihood $P(D \mid m, \hat{\theta})$	**Bayesian Model Selection** $\frac{P(D \mid m_1)}{P(D \mid m_2)}$
Penalizing for parameters	Akaike's Information Criterion (AIC)	Bayesian Information Criterion (BIC)

Akaike's Information Criterion (AIC)

$$AIC = \underbrace{-2\log P(D \mid \hat{\theta})}_{\text{Fit}} + \underbrace{2k}_{\text{Complexity}}$$

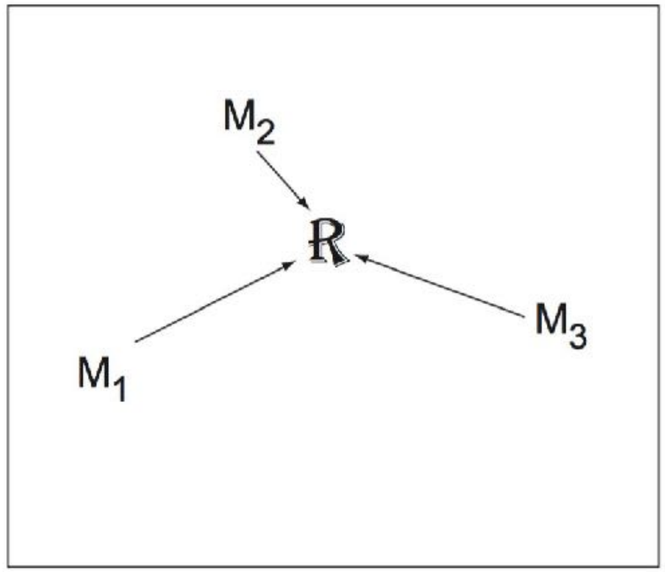
1. Perform MLE and compute 2x the negative Log Likelihood (aka *deviance*)

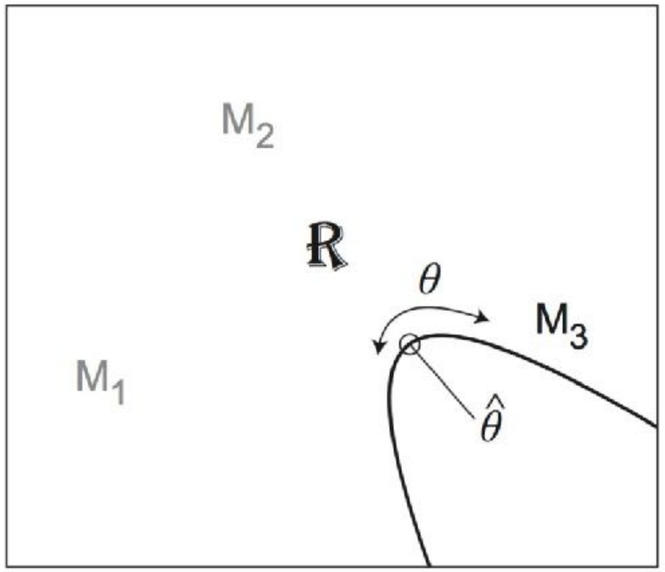
2. Penalize by adding an additional loss that is 2x the number of parameters k

Akaike's Information Criterion (AIC)

A measure of the relative information lost by a given model that is trying to capture some objective reality R(x)

$$KL = \int R(x)\log R(x)dx - \int R(x)\log P(x\,|\,\theta)dx$$

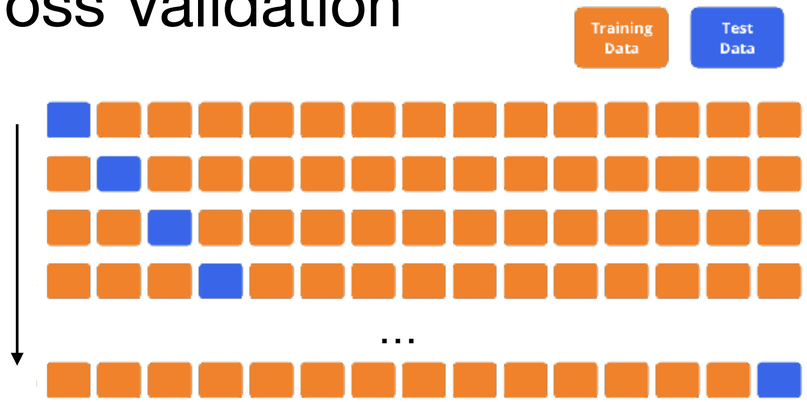




Akaike's Information Criterion (AIC)

Practice

Asymptotically, AIC is equivalent to Leave-One-Out-Cross Validation (Stone, 1977)



- for linear regression and mixed-effects regression
- in the limit of infinite data

… yet for it's simplicity, AIC is commonly used for non-linear models and certainly always short of infinite data

In practice, AIC can be considered the most lax of the goodness of fit measures we introduce, and is more prone to preferring an overfit model

Bayesian Information Criterion (BIC)

$$BIC = \underbrace{-2\log P(D \mid \hat{\theta})}_{\text{Fit}} + \underbrace{k \log n}_{\text{Complexity}}$$

1. Perform MLE and compute 2x the negative Log Likelihood (aka *deviance*)

2. Penalize by adding an additional loss that is **the number of parameters k times the log of the number of data points n**

Bayesian Information Criterion (BIC)

Bayesian model selection sometimes relies on Bayes Factors (BFs) to quantify the evidence of one model m_1 over another m_2

$$BF_{1,2} = \frac{P(D \mid m_1)}{P(D \mid m_2)}$$

- BF = 1; no evidence for either model
- BF >> 1; evidence for model 1
- BF << 1; evidence for model 2

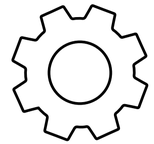
$$P(D \mid m) = \int P(D \mid \theta, m) P(\theta \mid m) d\theta$$

$$P(D \mid m) \approx BIC$$

BIC approximates the marginal likelihood using the MLE and by making some assumptions about the prior (Schwartz, 1975)

$$BF_{1,2} = \exp\left(-\frac{1}{2}(BIC_1 - BIC_2)\right)$$

Bayesian Information Criterion (BIC)



Practice

Bayesian interpretation is not without controversy (see Lewandowsky & Farrell, 2010 for a discussion) and the assumptions are hardly ever met or even unpacked

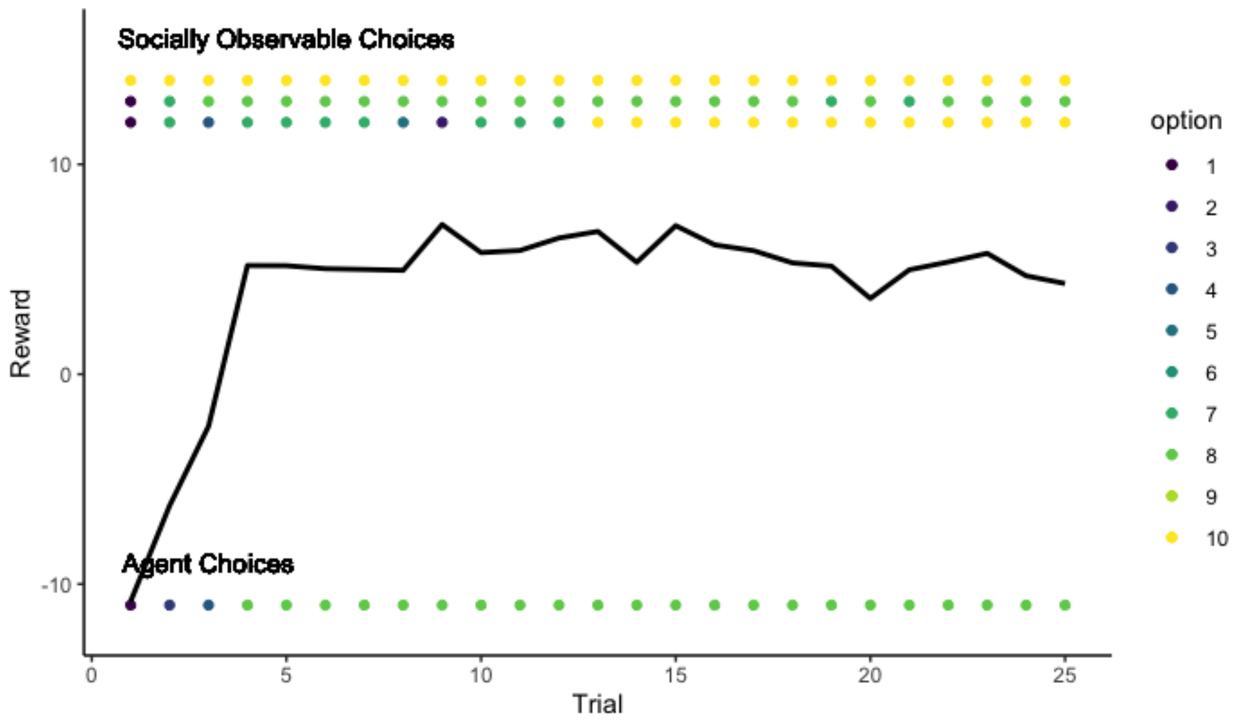
In practice, BIC is generally more strict than AIC in penalizing for complexity and less likely to prefer an overfit model

$$BIC = -2\log P(D\,|\,\hat{\theta}) + k\log n \qquad AIC = -2\log P(D\,|\,\hat{\theta}) + 2k$$

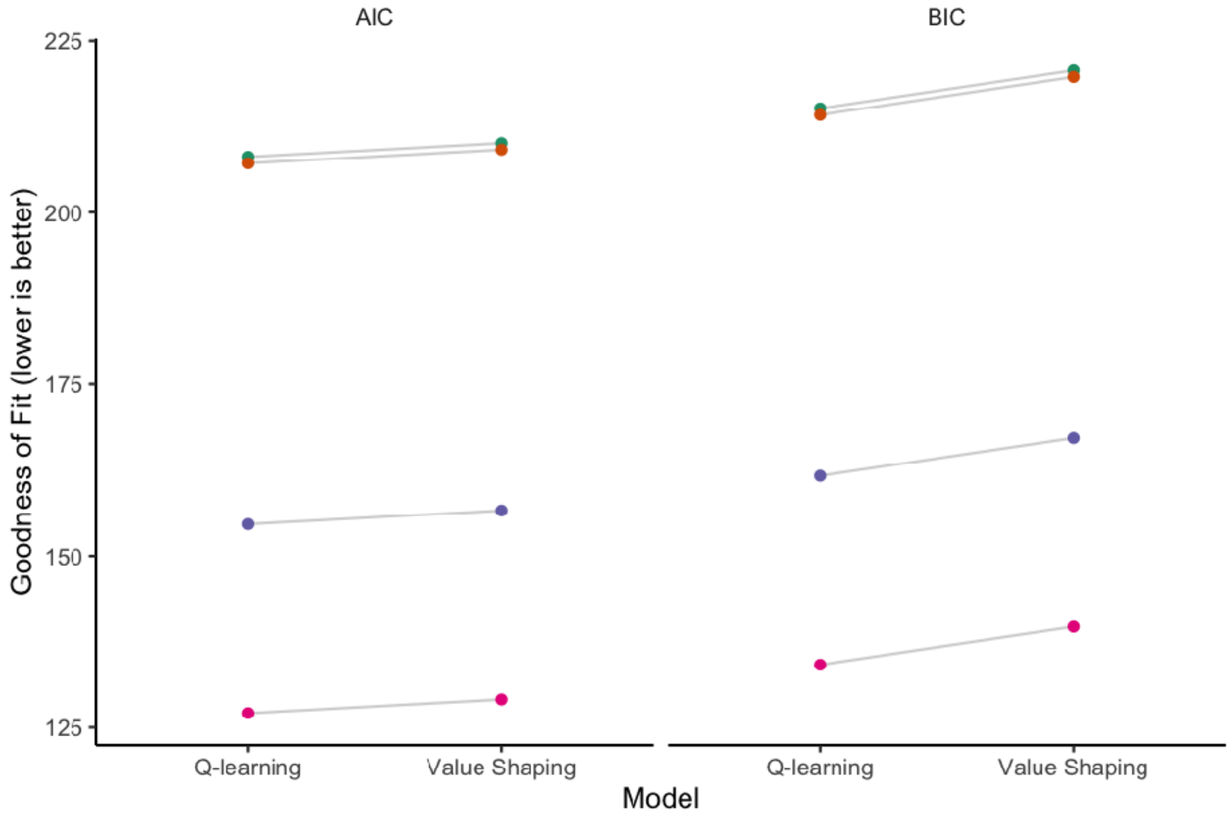
Note how $\log(n) > 2$ *when there are at least 8 data points*

AIC vs. BIC

Simulated data from a Q-learning agent



Q-learning vs. Value shaping



Goodness of Fit Measures

	Maximum Likelihood $P(D \mid m, \hat{\theta})$	**Bayesian Model Selection** $\frac{P(D \mid m_1)}{P(D \mid m_2)}$
Prediction error/ Bayesian Occam's Razor	Cross-validation loss	Model evidence using Markov Chain Monte Carlo (MCMC)

Cross Validation

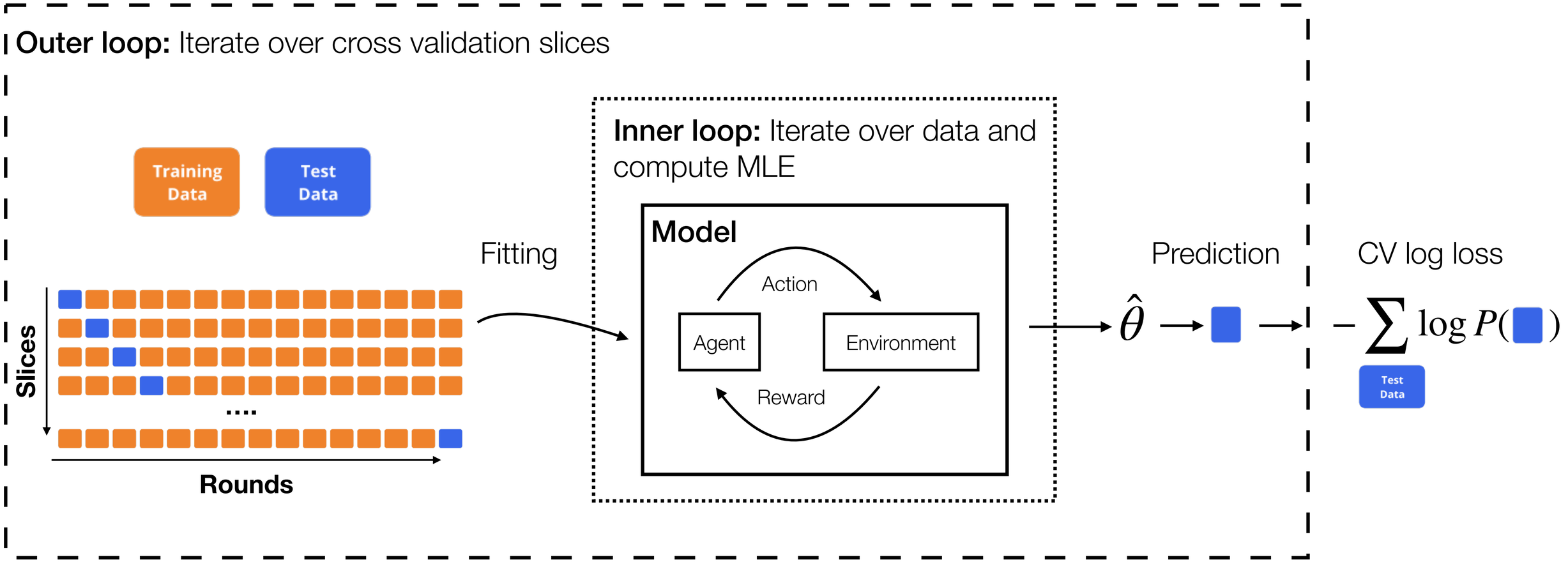
Rather than penalizing for complexity posthoc, we can actively test the predictive accuracy of a model through cross validation

1. Iteratively split the data into training and test sets

2. Estimate MLE on the training set, and then predict out-of-sample on the test set

3. Goodness of fit is the summed negative log likelihood of all out-of sample predictions:

Cross validation

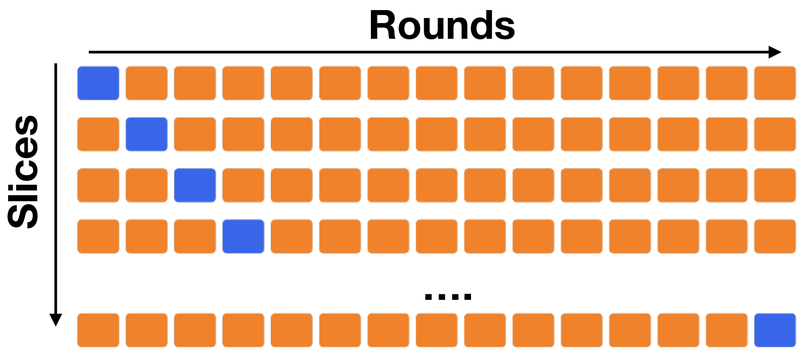


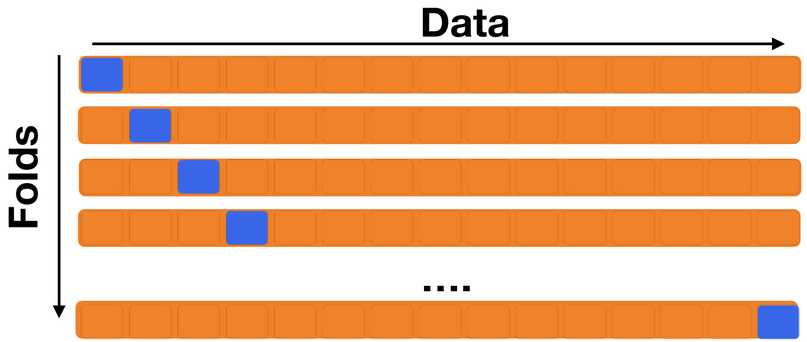
Variants of Cross validation

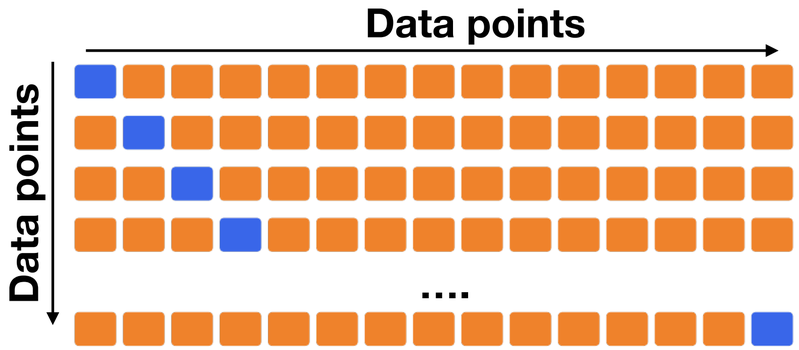
Leave-one-round-out cross validation: Use the natural distinction between independent rounds or blocks in an experiment

k-Fold cross validation: when there is no natural structure in the data, we can break it into *k* equally sized slices

Leave-one-out-cross validation: most extreme case, where we iteratively leave a single data point out of the training set







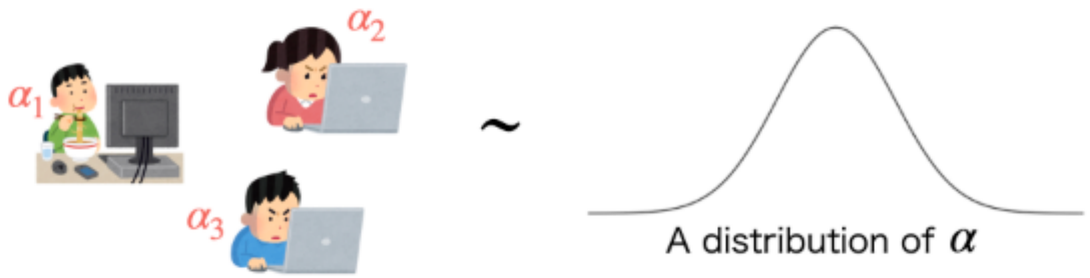

Let’s get distributional!

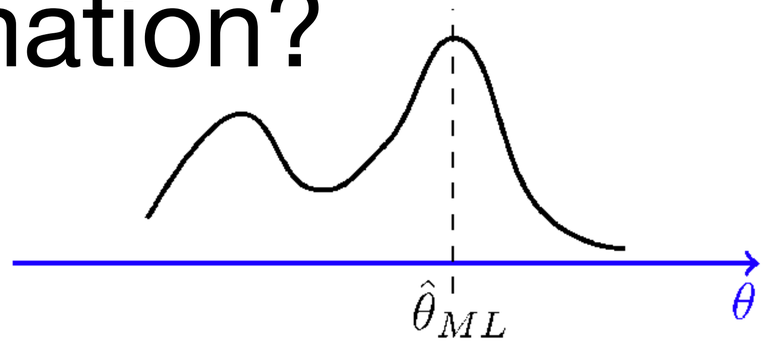
Why Bayesian model estimation?

1. Not just a point estimate, but an entire **probability distribution over parameters**

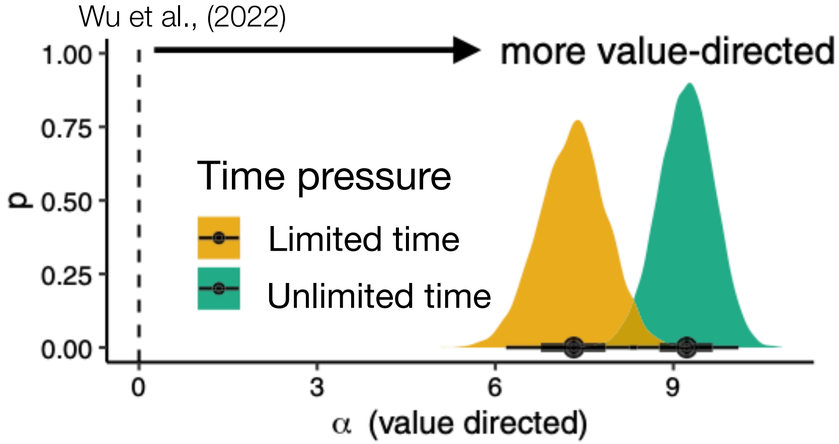
2. Rather than only assuming participants are independent samples, we can model **hierarchical relationships**

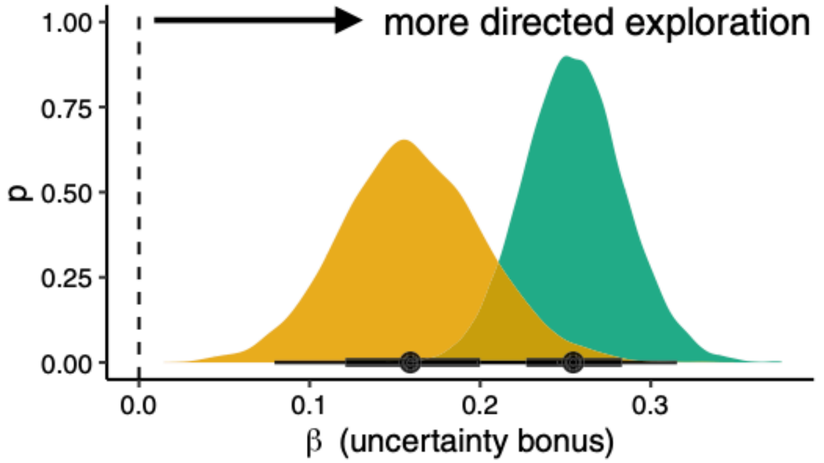
3. Naturally avoid overfitting through **Bayesian Occam's Razor**, since we evaluate the model across the entire range of parameters





Wu et al., (2022)





Posterior distribution over parameters

- Previously, we only used MLE to provide a point estimate of the best parameters $\hat{\theta}$
- Here, we want to estimate the full distribution of parameters suggested by the data and our choice of model:

$$P(\theta \mid D, m) \propto P(D \mid \theta, m) P(\theta, m)$$

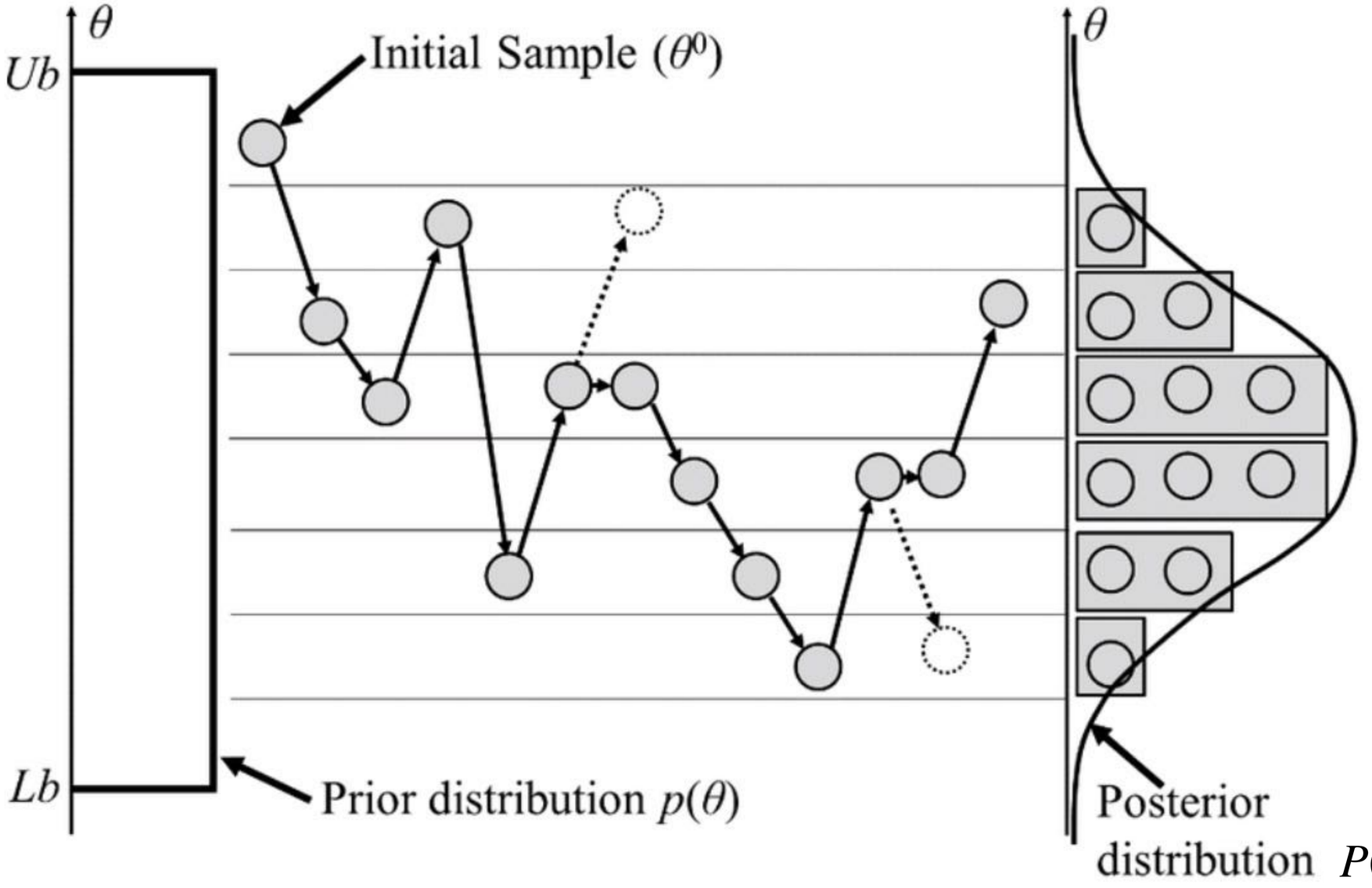
- $P(\theta \mid D, m)$ is the **posterior** distribution, which we compute using Bayes' rule combining:
  - The **likelihood** $P(D \mid \theta, m)$ of the data given a specific model and set of parameters
  - A **prior** $P(\theta, m)$ over parameters, capturing our initial guess before we see the data

Markov Chain Monte Carlo

- **Problem**: We want to model a probability distribution that is difficult to compute analytically
- **Solution**: acquire random samples that approximate this distribution

- *Markov Chain*
  - sequential process, where each random sample is used as a stepping stone to generate the next sample
  - Special property: Markov Chain has as it's equilibrium distribution the target distribution we are trying to approximate
- *Monte Carlo*
  - Law of large numbers —> enough randomly drawn samples will approximate the underlying distribution

Metropolis-Hastings MCMC

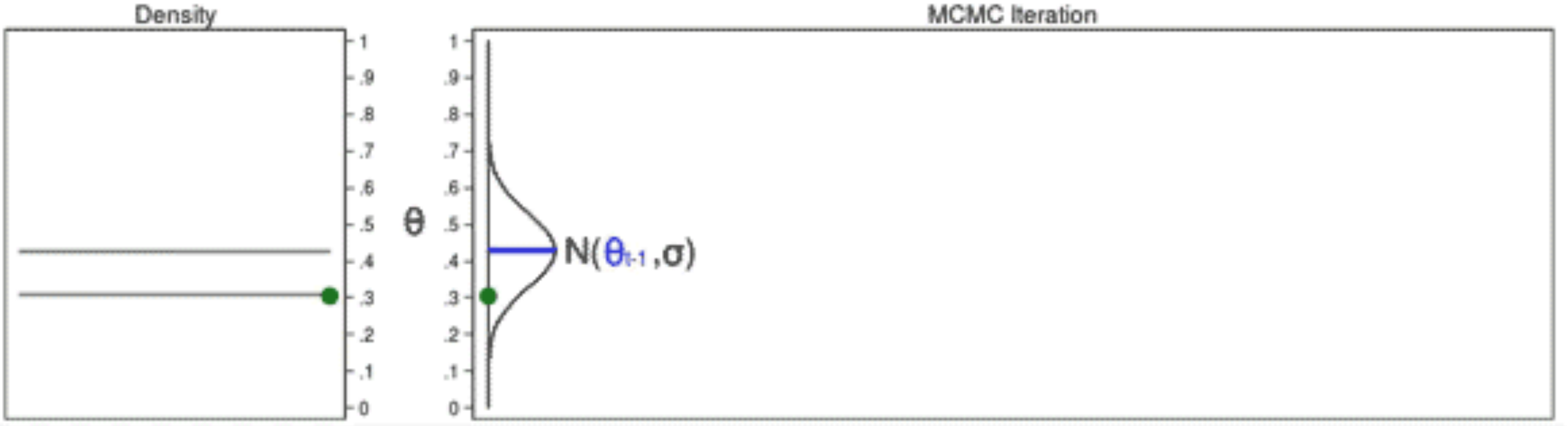


Psuedocode

1. Sample θ^i from $P(\theta^i | \theta^{i-1})$
2. Compute likelihood of data given these parameters $P(D | \theta^i)$
3. Accept the sample with probability proportional to how much of an improvement $P(D | \theta^i)$ is over $P(D | \theta^{i-1})$

The final collection of samples approximates the posterior parameter estimate $P(\theta | D)$

Lee, Sung, & Choi (2015)



Step 1: $r(\theta_{new}, \theta_{t-1}) = \frac{\text{Posterior}(\theta_{new})}{\text{Posterior}(\theta_{t-1})} = \frac{\text{Beta}(1,1,0.306) \times \text{Binomial}(10,4,0.306)}{\text{Beta}(1,1,0.429) \times \text{Binomial}(10,4,0.429)} = 0.834$

Step 2: Acceptance probability $\alpha(\theta_{new}, \theta_{t-1}) = \min\{r(\theta_{new}, \theta_{t-1}), 1\} = \min\{0.834, 1\} = 0.834$

Step 3: Draw $u \sim \text{Uniform}(0,1) = 0.617$

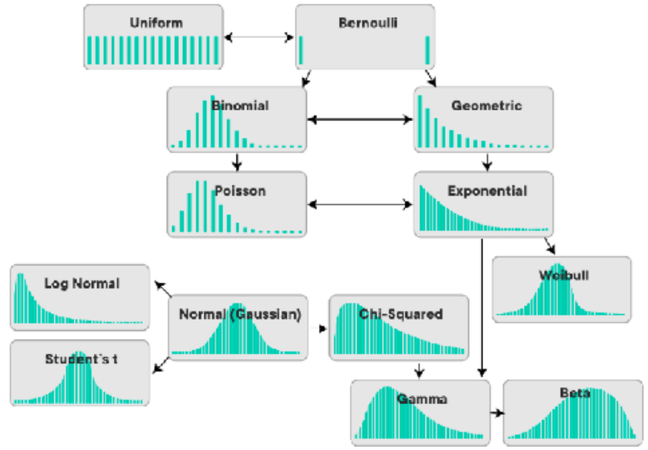
Step 4: If $u < \alpha(\theta_{new}, \theta_{t-1}) \rightarrow$ If $0.617 < 0.834$ Then $\theta_t = \theta_{new} = 0.306$

Otherwise $\theta_t = \theta_{t-1} = 0.429$

MCMC Samplers

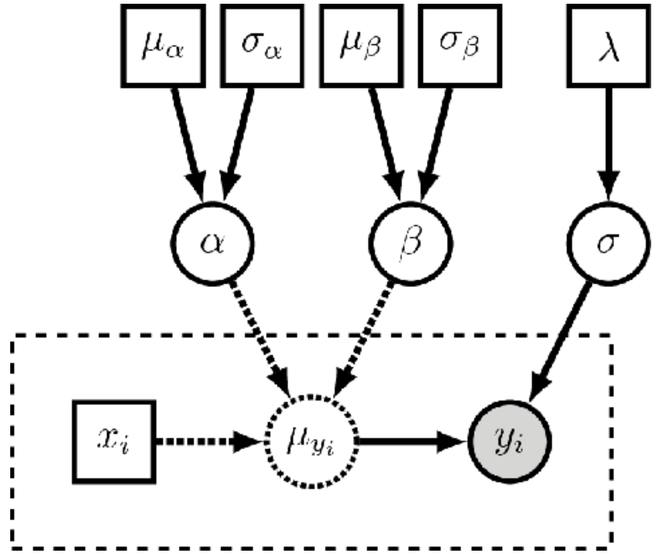
STAN

```
data {
  int<lower=0> N;                    // N >= 0
  int<lower=0,upper=1> y[N];         // y[n] in { 0, 1 }
}
parameters {
  real<lower=0,upper=1> theta;       // theta in [0, 1]
}
model {
  theta ~ beta(1,1);                 // prior
  y ~ bernoulli(theta);              // likelihood
}
```

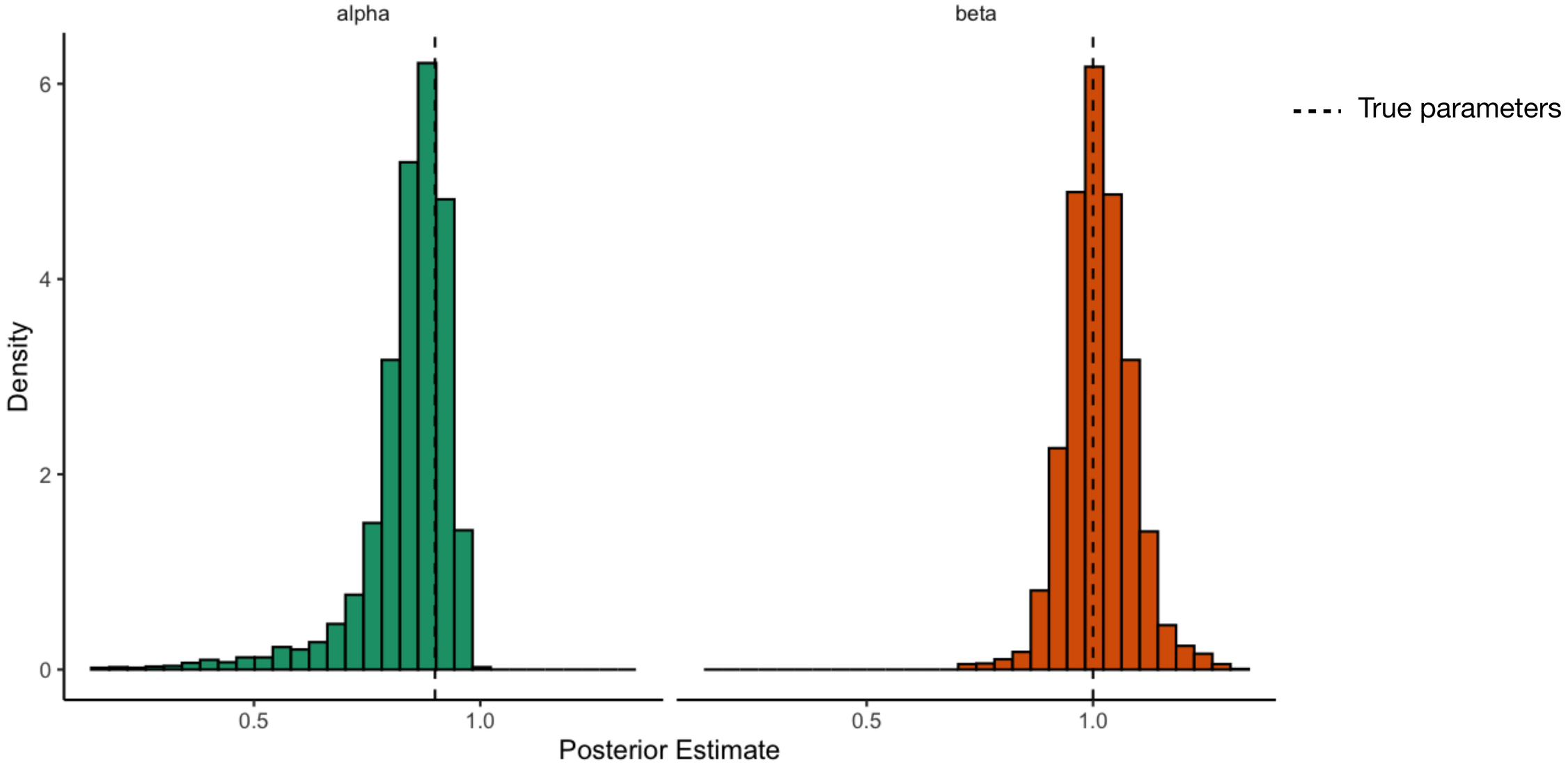


PyMC3

```
with pm.Model() as hierarchical_model_centered:
    # hyperpriors for group nodes
    mu_a = pm.Normal('mu_a', mu=0., sd=100**2)
    sigma_a = pm.HalfCauchy('sigma_a', 5)
    mu_b = pm.Normal('mu_b', mu=0., sd=100**2)
    sigma_b = pm.HalfCauchy('sigma_b', 5)

    # intercept about each county
    a = pm.Normal('a', mu=mu_a, sd=sigma_a, shape=n_counties)
    b = pm.Normal('b', mu=mu_b, sd=sigma_b, shape=n_counties)

    # error
    eps = pm.HalfCauchy('eps', 5)

    # regression
    radon_est = a[county_idx] + b[county_idx] * Dat.floor.values

    # likelihood
    radon_like = pm.Normal('radon_like', mu=radon_est, sd=eps, observed=Dat.log_radon)
```



Posterior over parameters



Bayesian model comparison

Information Criteria

AIC – Akaike information criterion

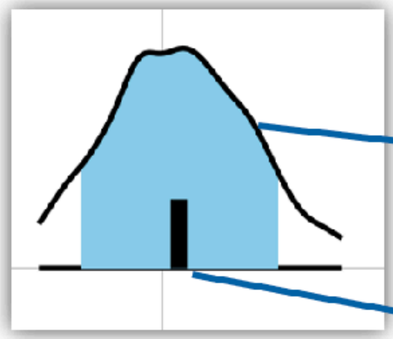
DIC – Deviance Information Criterion

WAIC – Widely Applicable Information Criterion

(Watanabe–Akaike information criterion)

finding the model that has the highest out-of-sample predictive accuracy

approximation to LOO



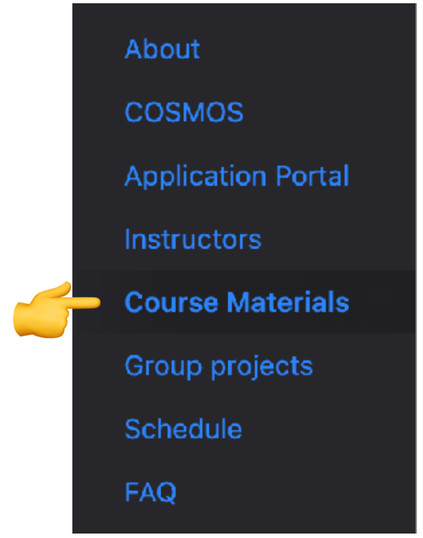
WAIC: using entire posterior distribution

AIC/DIC: using point estimation

52

Lei Zhang

For further study, see COSMOS 2023 Tutorial 4:



Part 1 Summary

	Maximum Likelihood $P(D \mid m, \hat{\theta})$	**Bayesian Model Selection** $\frac{P(D \mid m_1)}{P(D \mid m_2)}$
Penalizing for parameters	Akaike's Information Criterion (AIC)	Bayesian Information Criterion (BIC)
Prediction error/ Bayesian Occam's Razor	Cross-validation loss	Model evidence using Markov Chain Monte Carlo (MCMC)

Notebook

https://cosmos-konstanz.github.io/notebooks/tutorial-3-model-comparisons.html#model-fitting-exercise

Model fitting exercise

[meteor.csv](#)

..1	trial	round	agent	reward	choice
1	1	1	1	1.394896	6
2	1	1	2	6.316758	8
3	1	1	3	8.294503	9
4	1	1	4	2.370241	6
5	2	1	1	10.749758	10
6	2	1	2	4.841205	8
7	2	1	3	1.433119	6
8	2	1	4	6.110761	8
9	3	1	1	10.770239	10

[comet.csv](#)

..1	trial	round	agent	reward	choice
1	1	1	1	-8.802858	1
2	1	1	2	7.816797	9
3	1	1	3	9.492214	9
4	1	1	4	-5.848119	3
5	2	1	1	9.153988	9
6	2	1	2	-7.123128	2
7	2	1	3	8.075340	9
8	2	1	4	8.093050	9
9	3	1	1	1.326370	6

Self-contained model-fitting code

```
#1. Load the data
meteorDF <- read_csv('meteor.csv')   #Make sure the file path is correct,
cometDF <- read_csv('comet.csv')

k <- length(unique(c(cometDF$choice,meteorDF$choice))) #number of arms; wh

#softmax function
softmax <- function(beta, Qvec){
  p <- exp(beta*Qvec)
  p <- p/sum(p) #normalize to sum to 1
  return(p)
}

#2. Likelihood functions for different models

#Asocial Q-learning
asocialLikelihood <- function(params, data, Q0=0){ #We assume that prior v
  names(params) <- c('alpha', 'beta') #name parameter vec
  nLL <- 0 #Initialize negative log likelihood
  rounds <- max(data$round)
  trials <- max(data$trial)
  for (r in 1:rounds){ #loop through rounds
    Qvec <- rep(Q0,k) #reset Q-values each new round
    for (t in 1:trials){ #loop through trials
      p <- softmax(params['beta'], Qvec) #compute softmax policy
      trueAction <- subset(data, trial==t & round == r)$choice
      negativeloglikelihood <- -log(p[trueAction]) #compute negative log l
      nLL <- nLL + negativeloglikelihood #update running count
      Qvec[trueAction] <- Qvec[trueAction] + params['alpha']*(subset(data,
    }
  }
  return(nLL)
}
```

```
#3. Fit the model to each participant, by minimizing the nLL

# Here we show examples using the asocial learner, decision-biasing,

#Asocial learner
init <- c(1,1) #initial guesses for alpha and beta
lower <- c(0,-Inf) #lower and upper limits. We use very liberal bounds
upper <- c(1,Inf)

MLE <- optim(par=init, fn = asocialLikelihood,lower = lower, upper=upp
MLE$value #nLL of the MLE
MLE$par #Parameter estimates of the MLE

#DecisionBiasing
init <- c(1,1,1,1) #initial guesses for alpha, beta, gamma, and theta
lower <- c(0,-Inf,0, 1) #lower and upper limits. We use very liberal
upper <- c(1,Inf,1, Inf)

MLE <- optim(par=init, fn = dbLikelihood,lower = lower, upper=upper, n
MLE$value #nLL of the MLE
MLE$par #Parameter estimates of the MLE

#value shaping
init <- c(1,1,1) #initial guesses for alpha, beta, and eta
lower <- c(0,-Inf,0) #lower and upper limits. We use very liberal bou
upper <- c(1,Inf,Inf)

MLE <- optim(par=init, fn = vsLikelihood,lower = lower, upper=upper, n
MLE$value #nLL of the MLE
MLE$par #Parameter estimates of the MLE
```

Which model best explains each dataset?

Part 2. Robustness

(5 minute break)

Robustness checks

1. **Model recovery**

   - Can the data actually differentiate between the models we are considering? Could there be model mimicry, where the wrong model can mistakenly win?

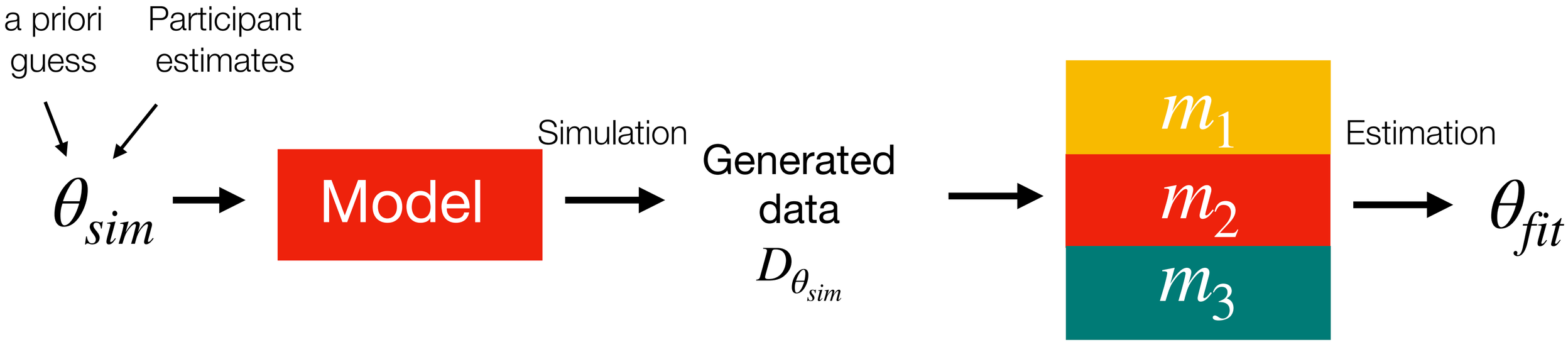
2. **Parameter recovery**

   - Are the parameters of the model capturing distinct phenomenon? Can changes in one parameter be acommodated by changes in another parameter (i.e., misspecification)?

3. **Simulated data**

   - Can the model generate realistic participant behavior? Is it capturing the mechanisms that matter for performance, rather than simply fitting the noise?

Model recovery



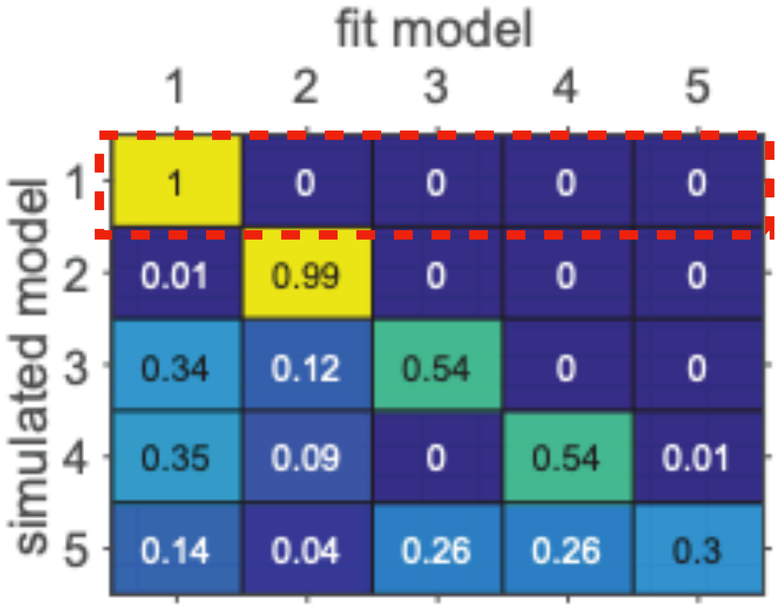
1. Use models to simulate data, parameterized with θ_{sim} either an *a priori* guess or from participant estimates

2. Use the same model estimation procedure on the simulated data to estimate θ_{fit} for each model under consideration

3. How often does the correct model provide the best fit?

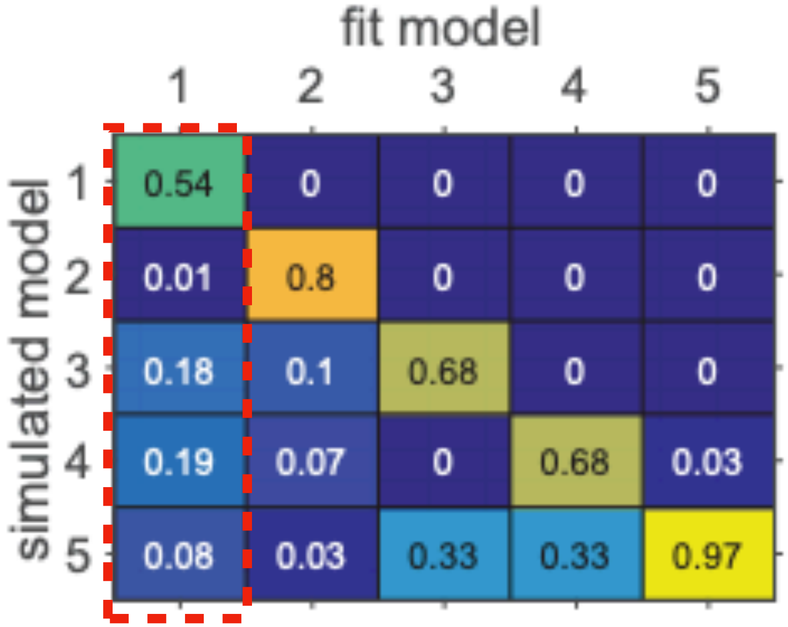
Model recovery

Confusion matrix p(fit|sim)



Which alternative models mimic a given simulation model?

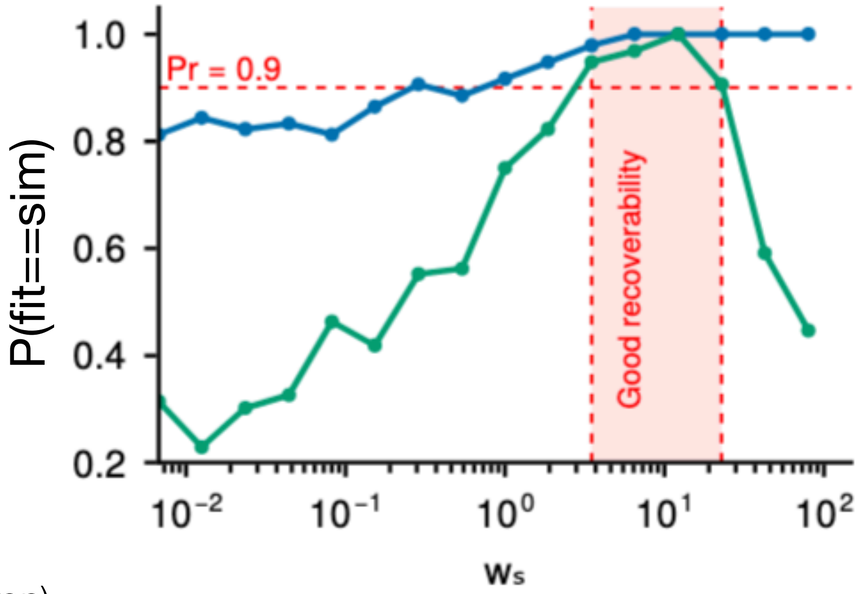
Inversion matrix p(sim|fit)

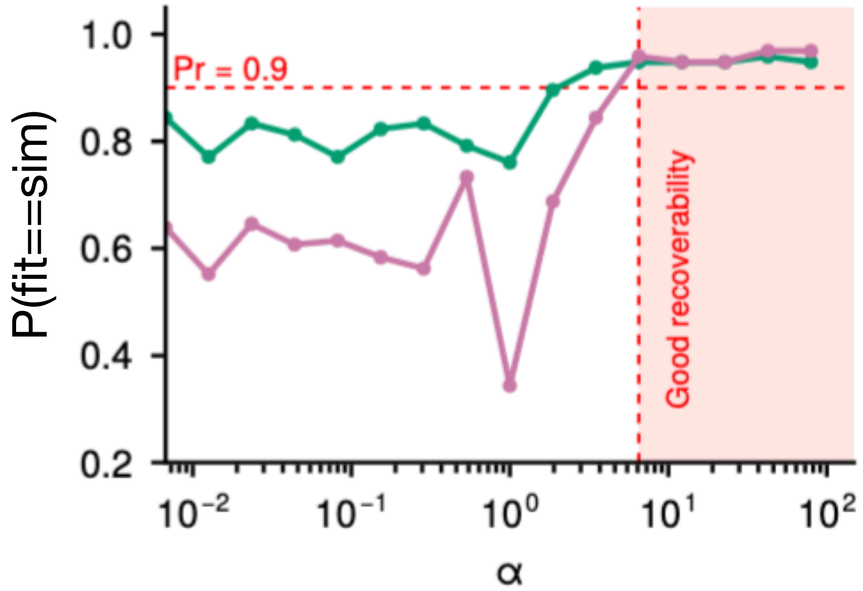


If a given model wins a model competition, how likely is it to actually be the true generative model?

Improving model recovery for nested models

- For nested models, not all ranges of parameters will yield distinct behavior
- You can run model recovery over different parameter values of each new component of the model to find ranges with good recovery
- Use these ranges as parameter bounds for estimation





Ten, Sakaki, Breit, Murayama & Wu (*in prep*)

Parameter Recovery

Goal: Determine if parameters are distinct and behaviorally specific

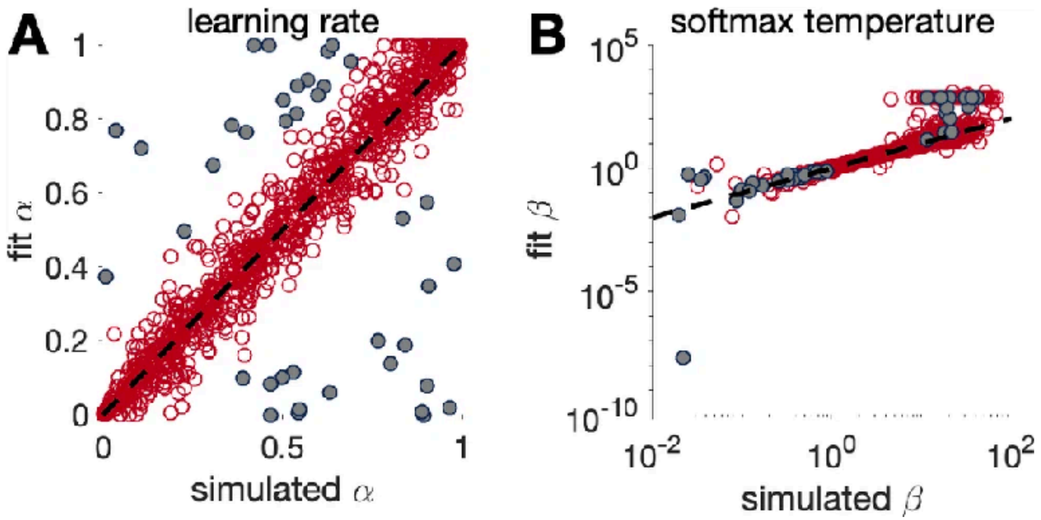
1. Use either participant parameter estimates or some prior guess to simulate data (x-axis)

2. Run model fitting to estimate new parameters on simulated data (y-axis)

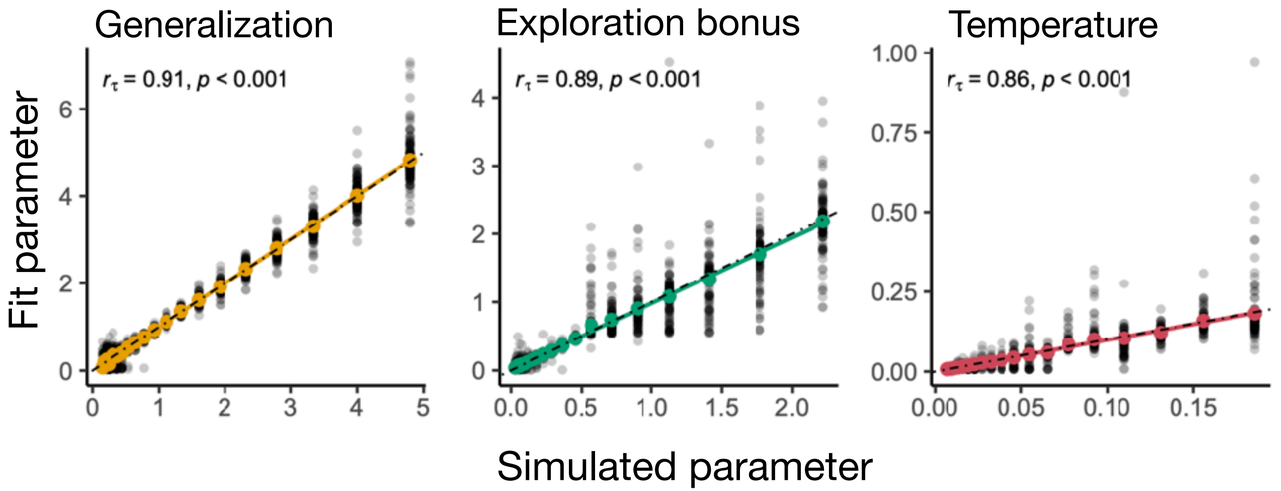
3. Do the fit parameters correspond to the simulated parameters?

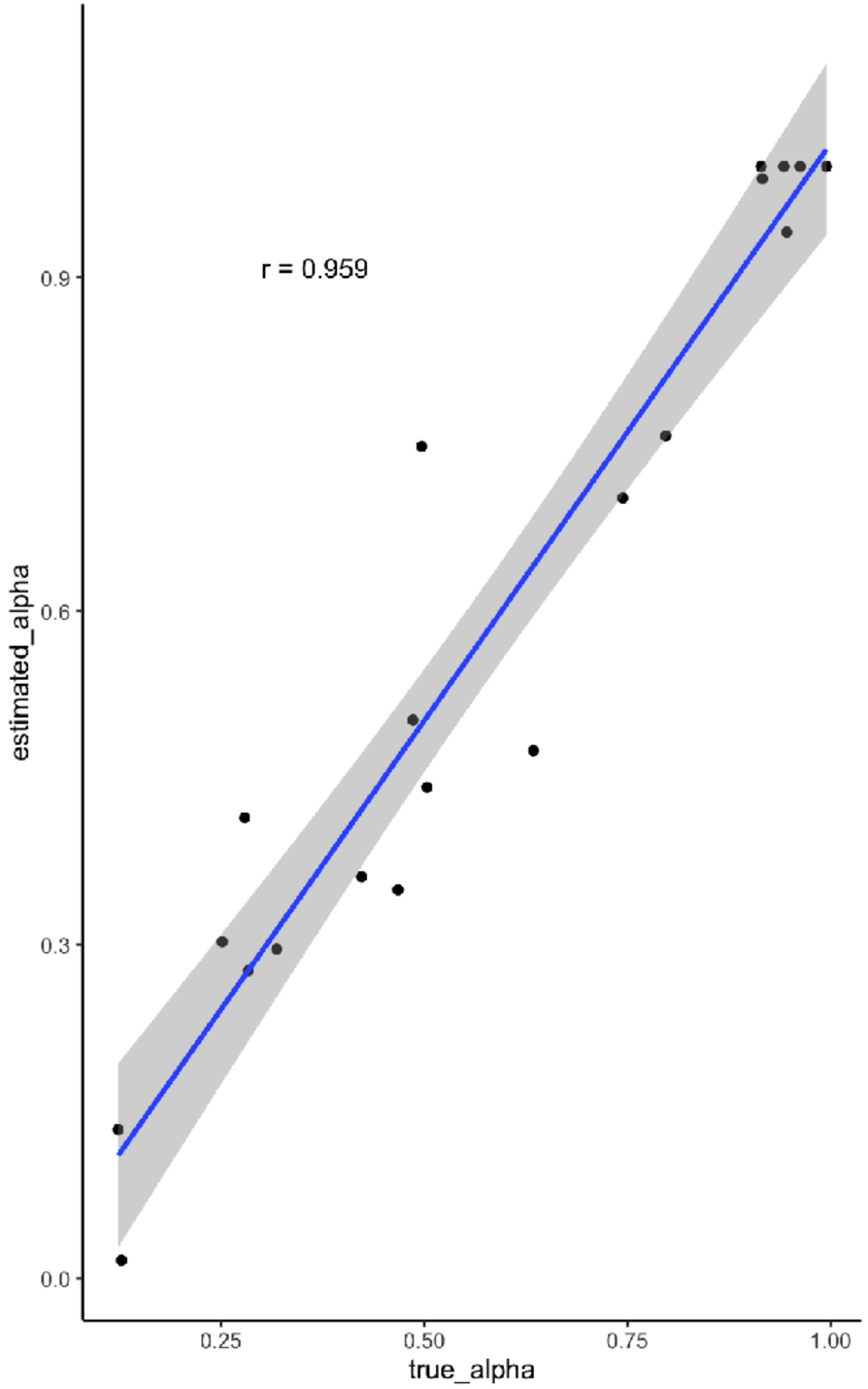
[Bonus] Counterfactual parameter recovery: Systematically vary simulating parameters across a range of plausible values. Does the entire hypothesis space recover?

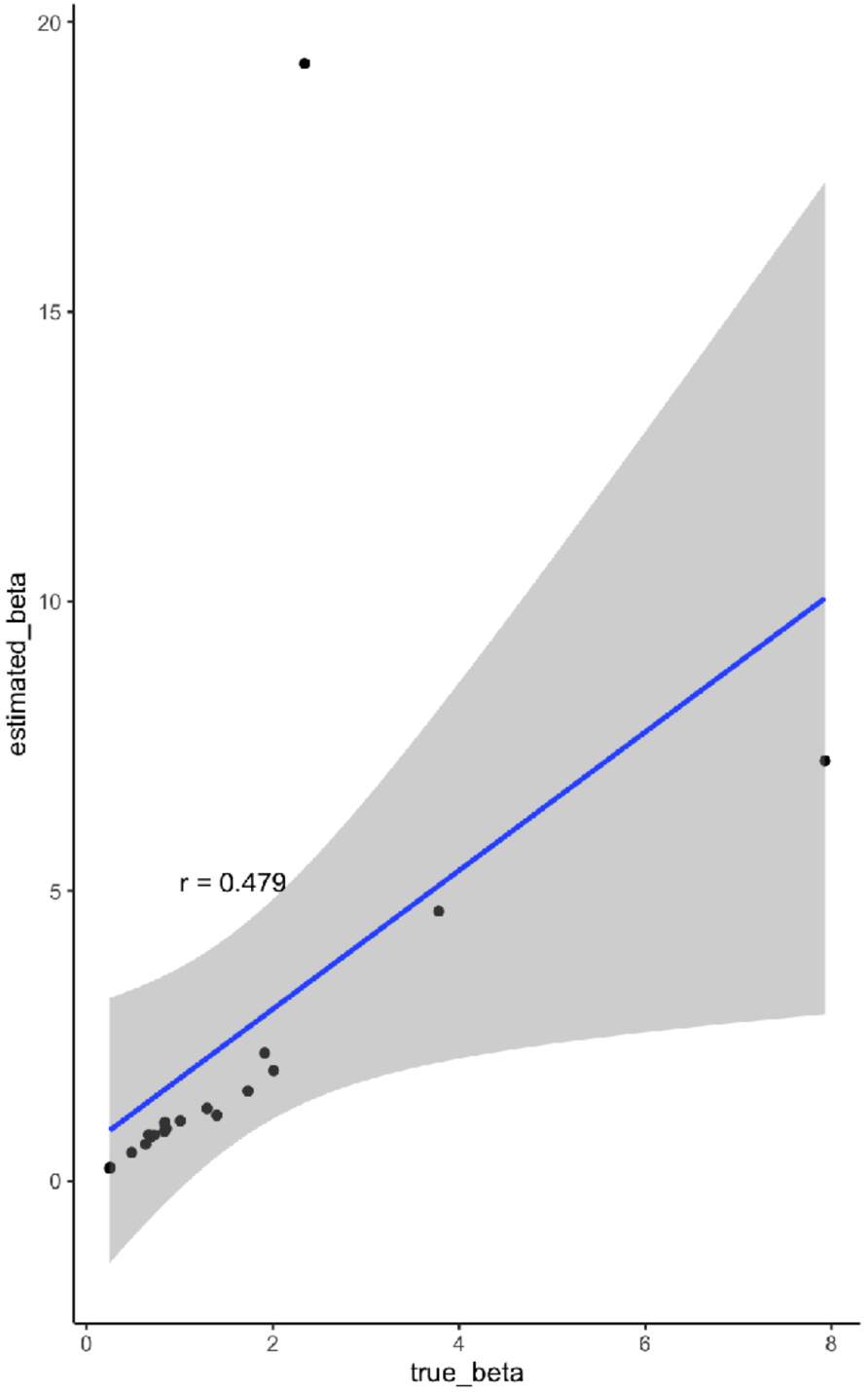
Wilson & Collins (*eLife* 2019)



Giron et al., (2023)



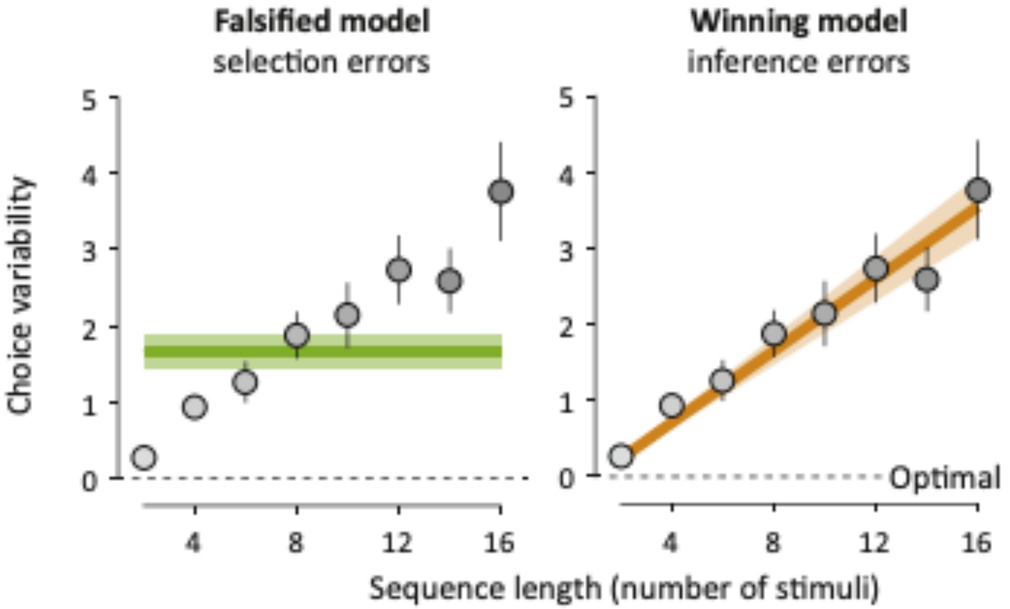

r = 0.959
estimated_alpha
0.0
0.3
0.6
0.9
0.25
0.50
0.75
1.00
true_alpha


r = 0.479
estimated_beta
0
5
10
15
20
0
2
4
6
8
true_beta

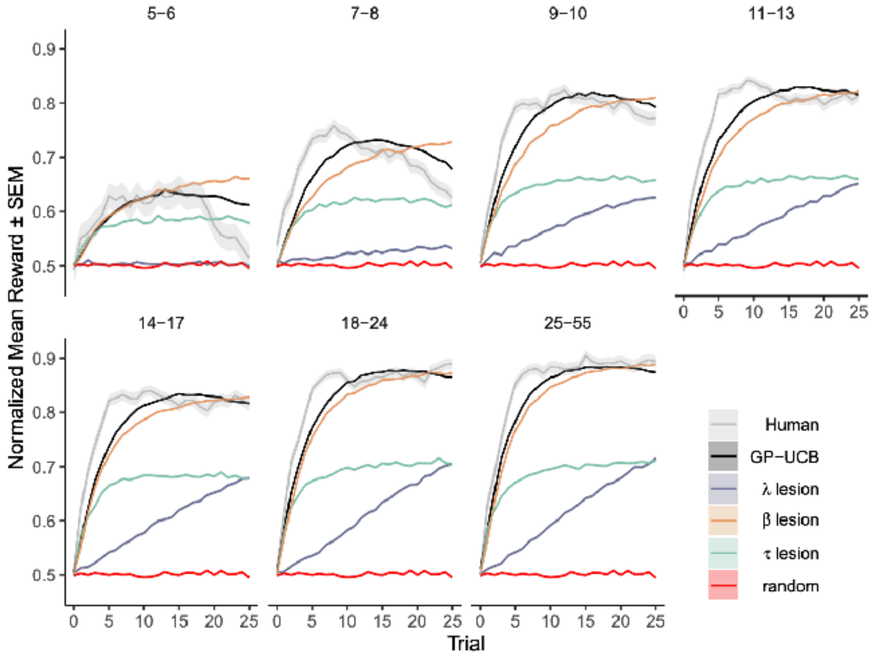
Simulated Performance

- Goodness of fits don't always tell the full story
- Sometimes you need to check that models can reproduce important patterns of human behavior
  - Can also be used to probe hidden components of the model, such as value representations
- Compare simulated model performance to human performance
  - Can the model replicate differences across experimental manipulations or from different populations

Palminteri, Wyart, & Koechlin (*TICS* 2017)



Giron et al., (2023)



General Recipe for Cognitive Modeling

Not fixed, step by step instructions…

General Recipe

1. What are your hypotheses? Turn them into models
2. How will you estimate the model parameters and perform model comparison?
3. Is your modeling framework robust? If not, rethink your task, the models, and/or your modeling framework.
4. [Collect data]
5. Analyze and interpret results
6. Test if recoverability still works with participant parameters

What can you justify?

Recommended Readings