

The drift-diffusion model of decision making

Ariel Zylberberg
Columbia University

Matlab code used in this tutorial:

https://github.com/arielzylberberg/CompPsychCourse_Zurich2024

Please ask many questions!
Just write them in the chat or (better!) unmute

az2368@columbia.edu

Please introduce yourself

- Your name
- What you are working on or studying
- Why are you interested in learning more about drift-diffusion models?

Indicate your attendance:
<https://tinyurl.com/3pa8vntk>

Data sets vs Data streams

Data analysts often deal with datasets,
decision-makers with data streams.

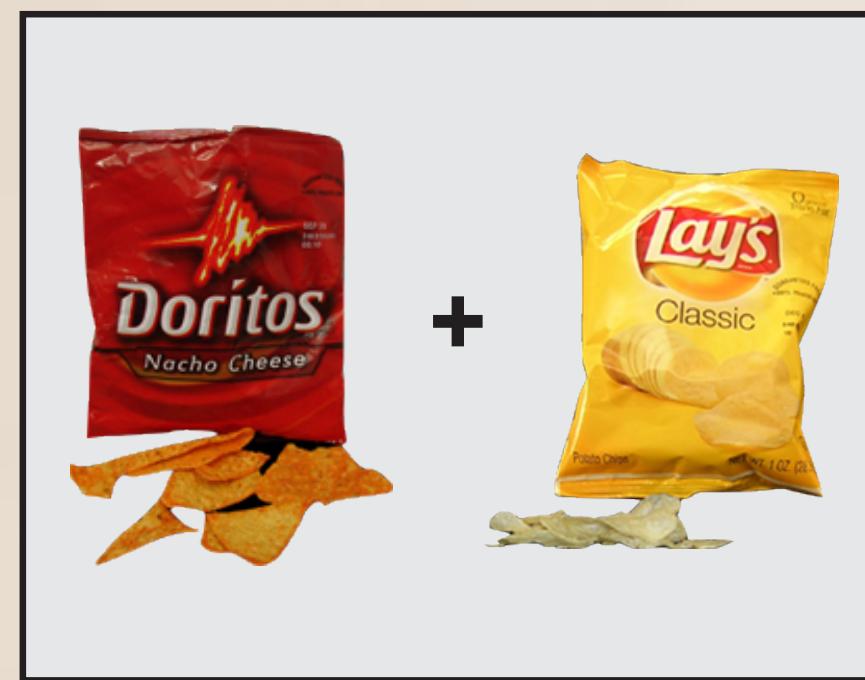


Drift-diffusion model of decision making

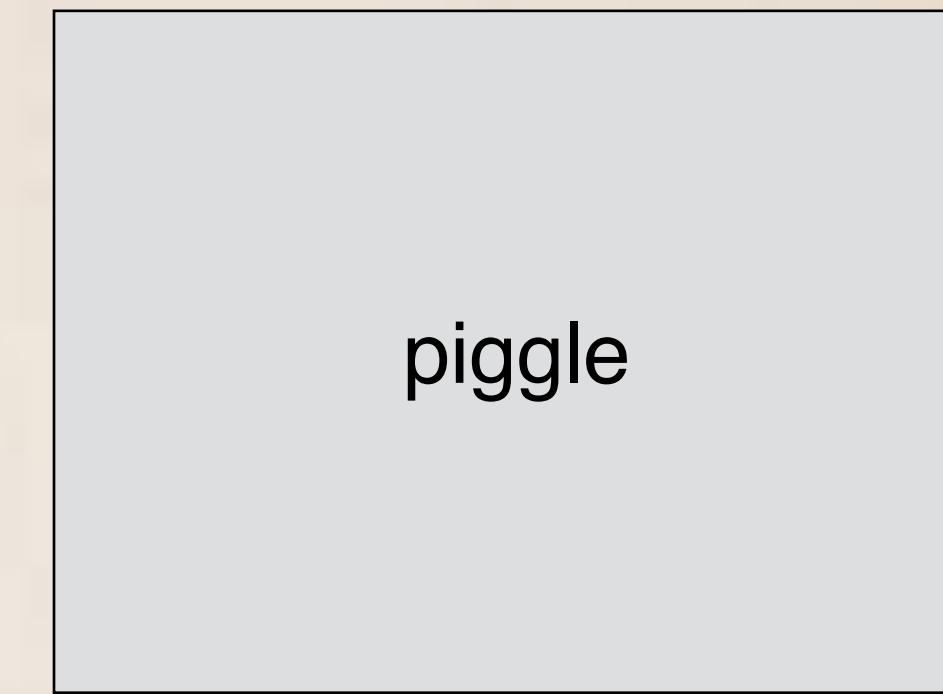
- Mathematical model used in cognitive science, neuroscience and psychology to describe the process of making a simple choice between two alternatives
- Attempts to explain choice and response time
- Supported by neurophysiology
- Extended to more complex decisions
 - Multi-alternative decisions, Confidence judgments, Changes of mind, Sequential decisions, Multi-attribute decisions, ...

DDM mostly used to model simple decisions (RTs ~ 2 s)

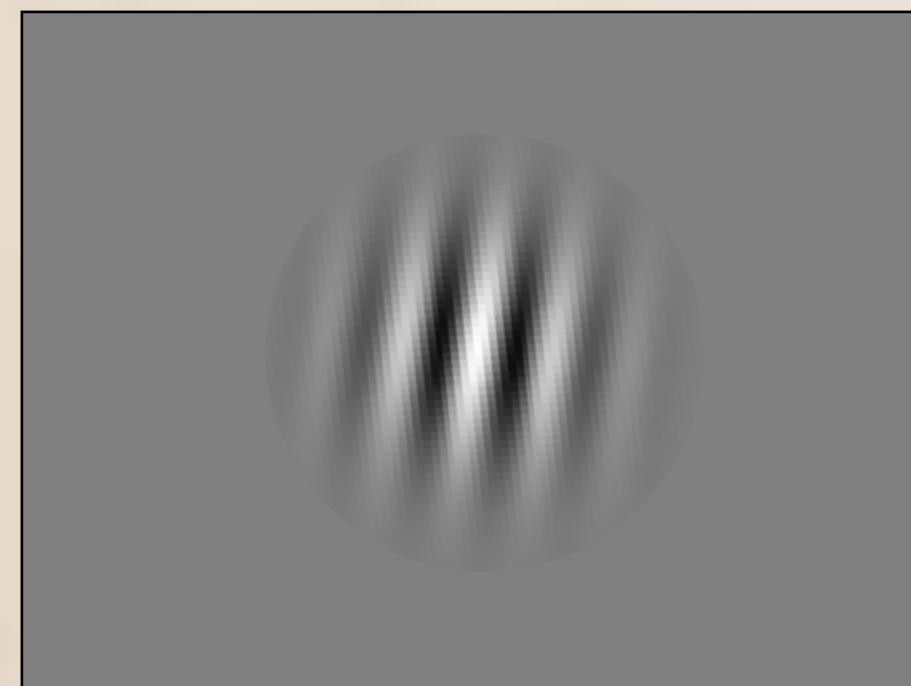
Which snack do you prefer?



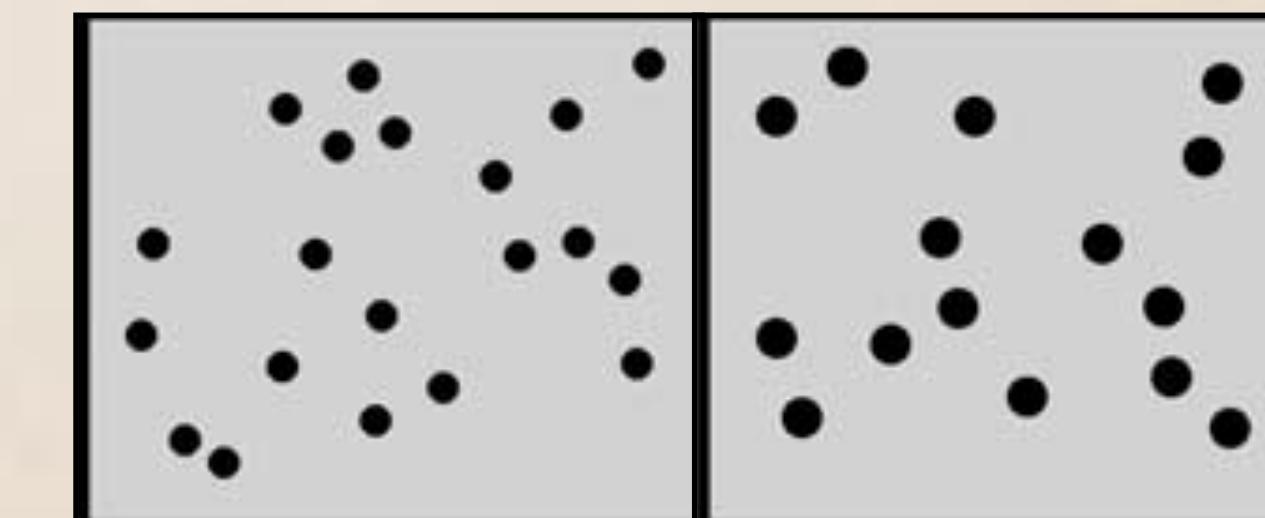
Word or non-word?



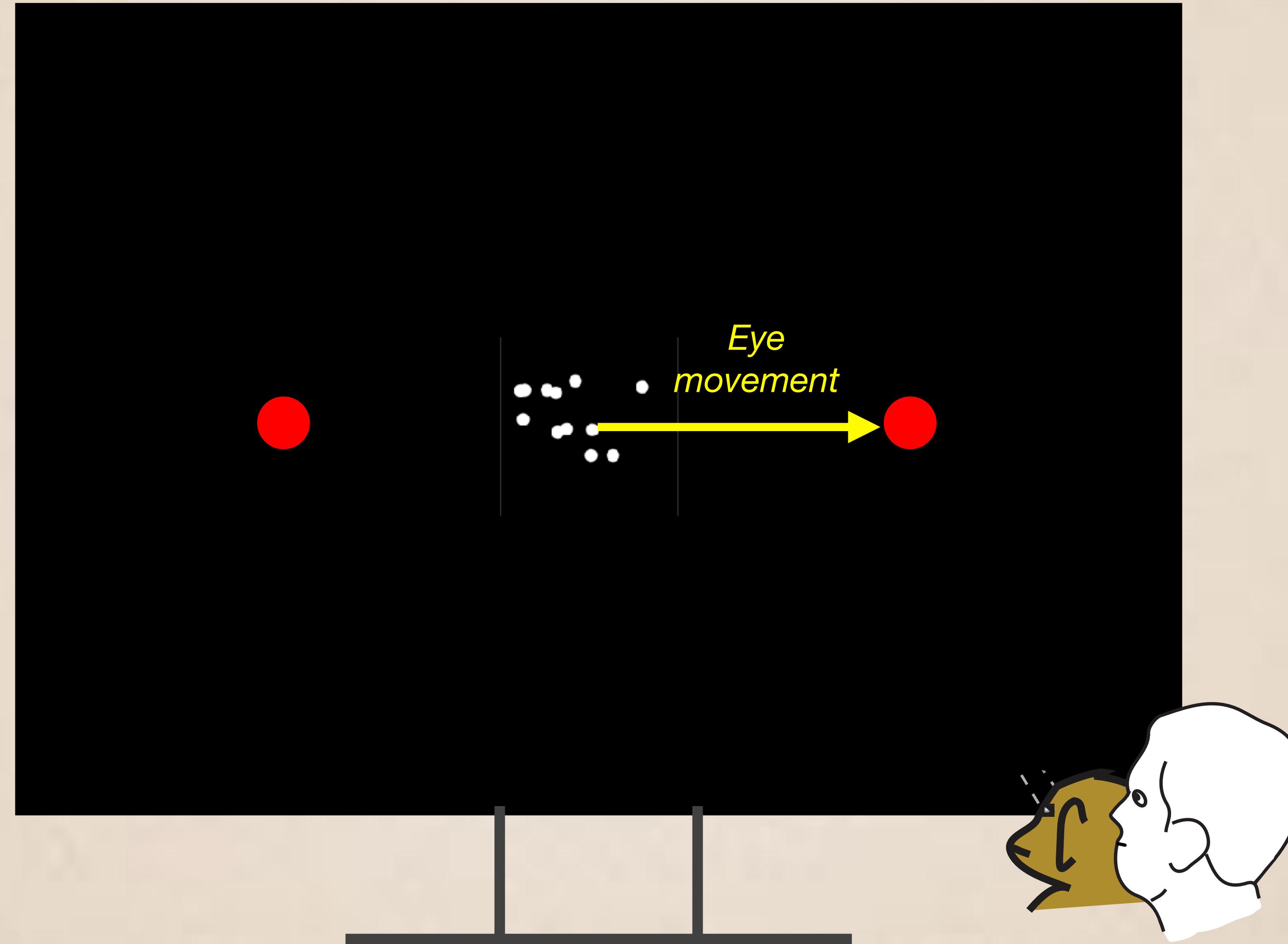
Is orientation clockwise or anti-clockwise?



Which display has more dots?

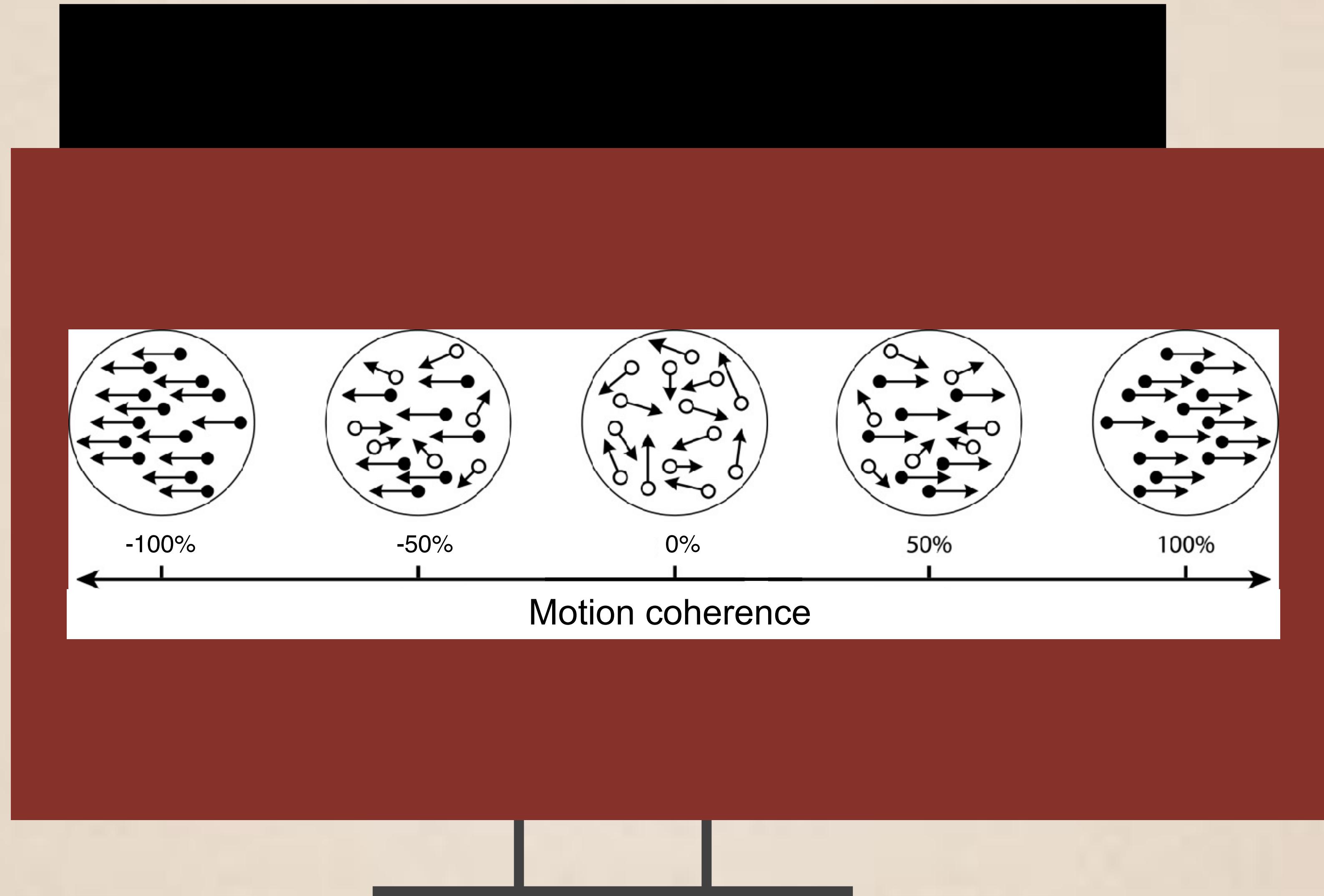


Random dot motion discrimination task

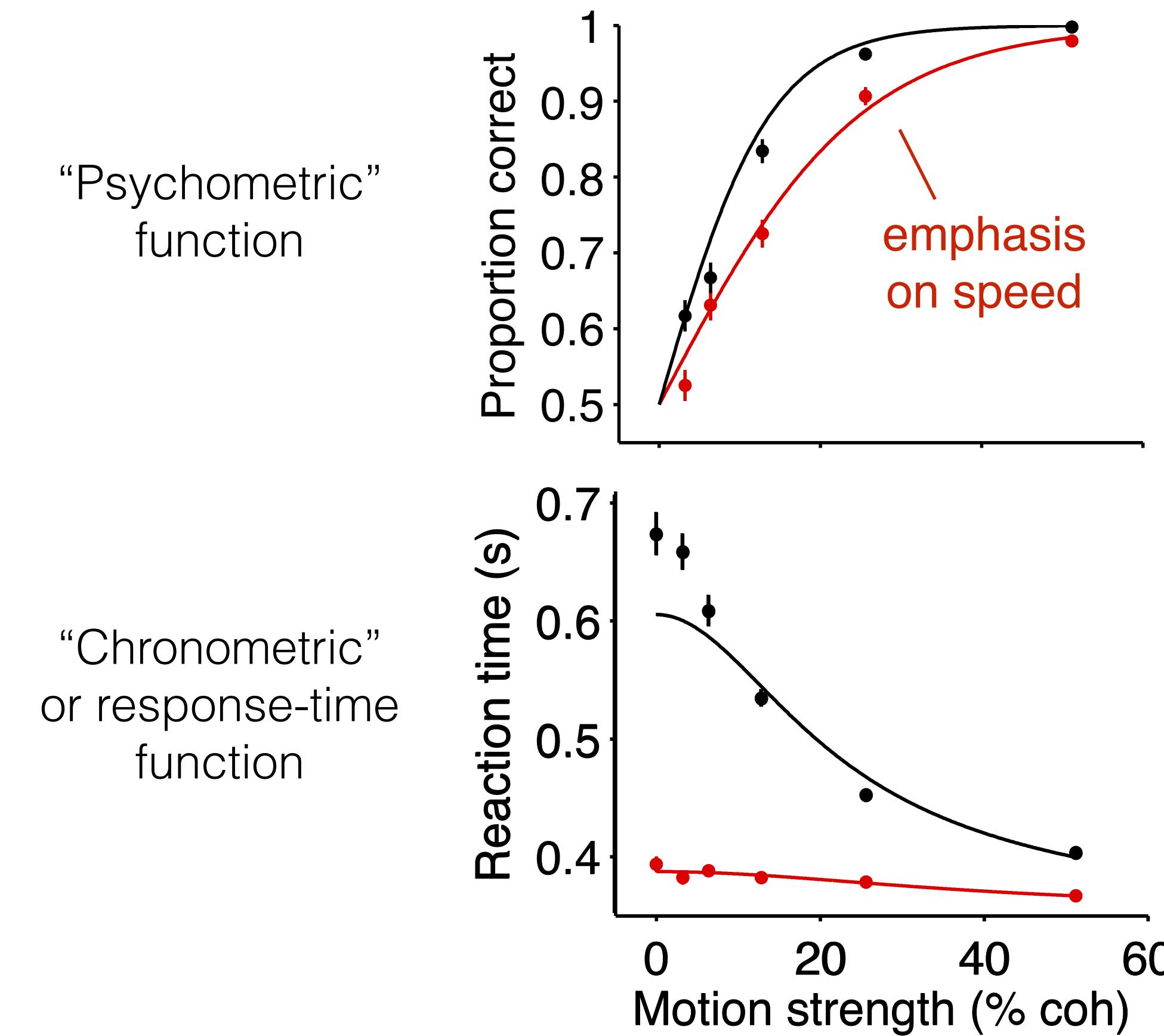


Random dot motion discrimination task

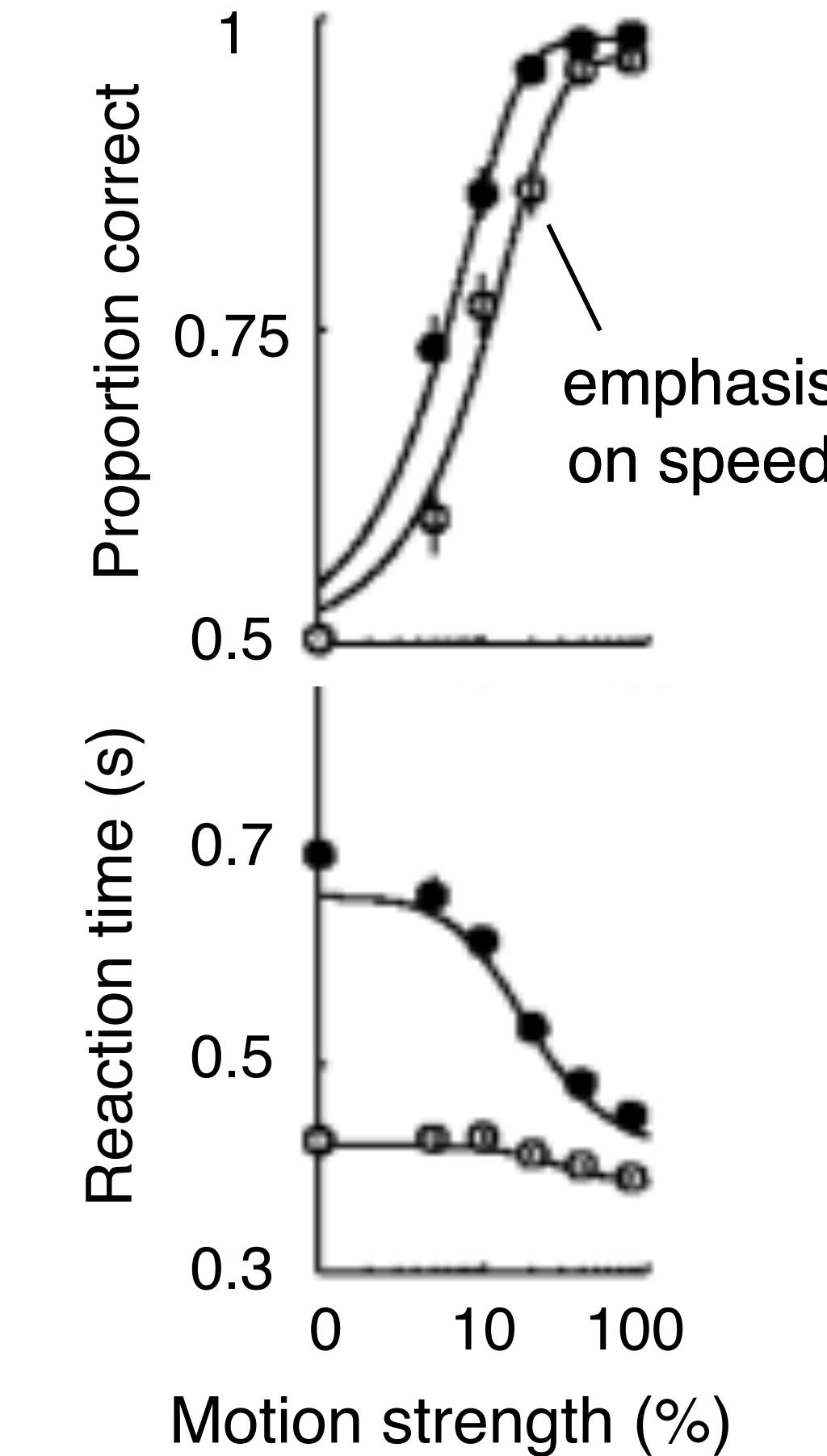
(More difficult)



Speed-accuracy tradeoff



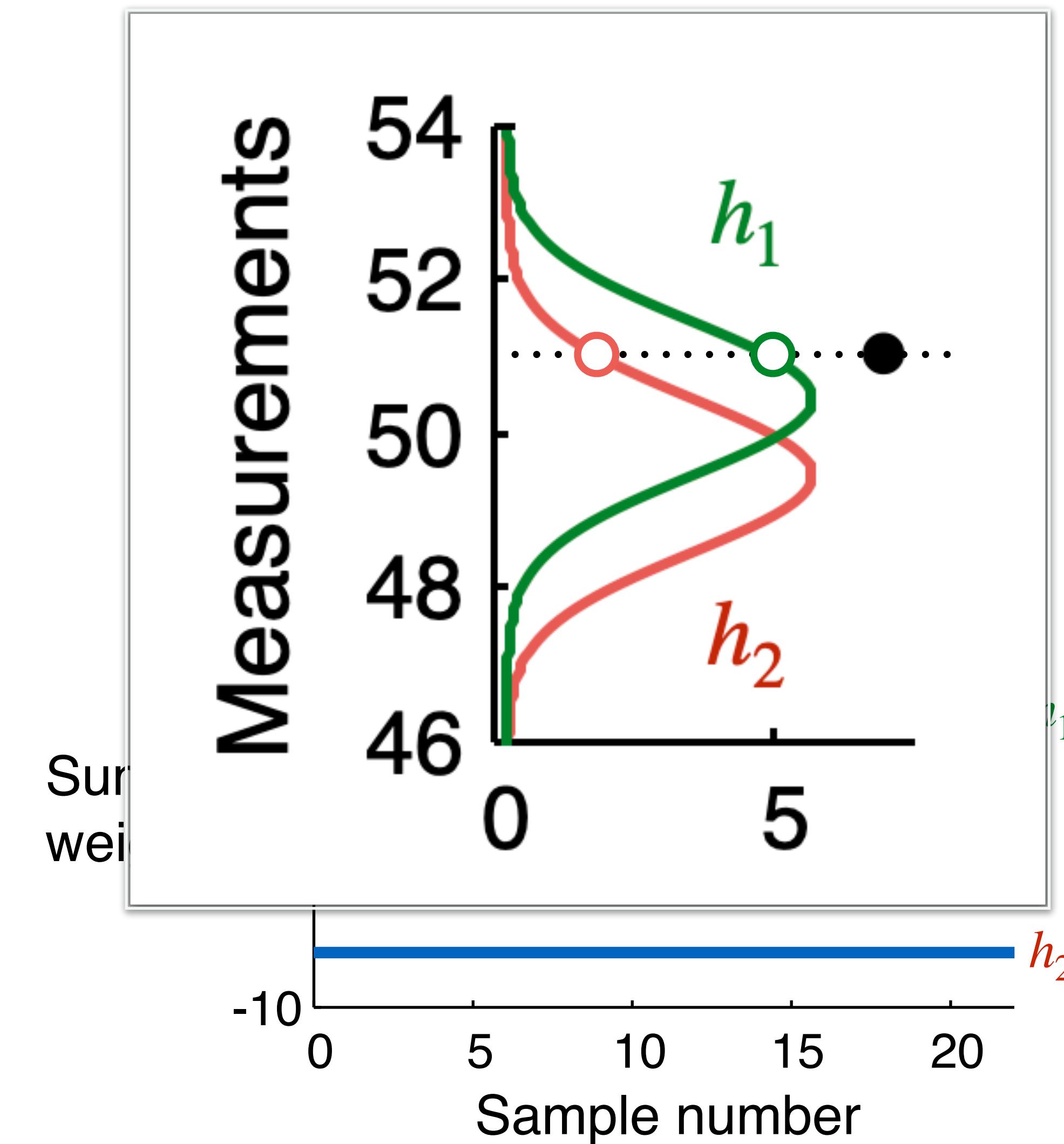
Hanks et al. eLife, 2014



Mulder et al.
Att Percept Psych
2013

Sequential probability ratio test (SPRT)

Wald & Wolfowitz (1948), A. Turing (194?)



Weight of evidence e in favor of h_1 over h_2 =

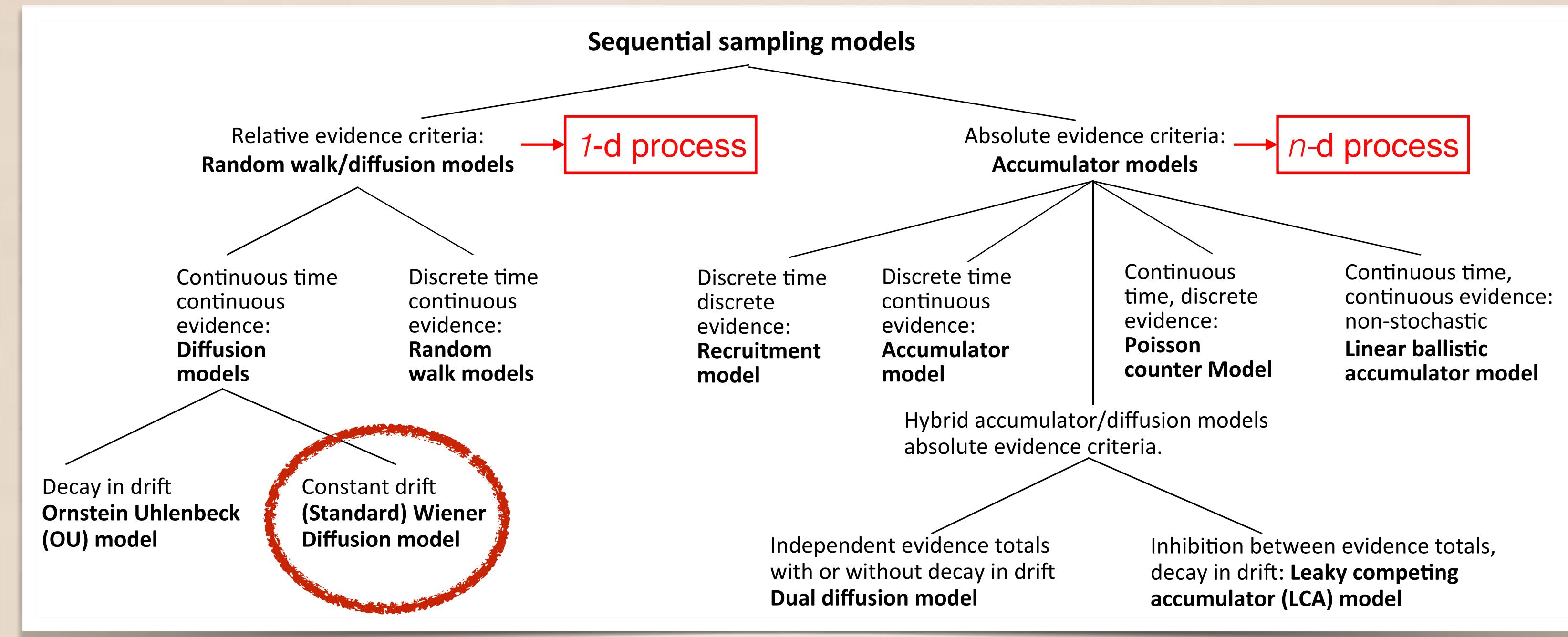
$$\log \left[\frac{\text{Probability of } e \text{ if } h_1 \text{ is true}}{\text{Probability of } e \text{ if } h_2 \text{ is true}} \right]$$

$B \approx \log \left(\frac{\text{acc}}{1 - \text{acc}} \right)$

bound height

required accuracy

Many variants of sequential sampling models for binary decisions



Ratcliff et al. TiCS 2016

- » Time (continuous or discrete)
- » Evidence samples (continuous or discrete)
- » Perfect or “leaky” accumulation
- » Other factors (with or without inhibition, ...)

Drift-diffusion model

We will simulate this in Matlab

- » Decision based on the accumulation of samples of evidence arriving sequentially over time

$$x_{i+1} = x_i + e_i$$

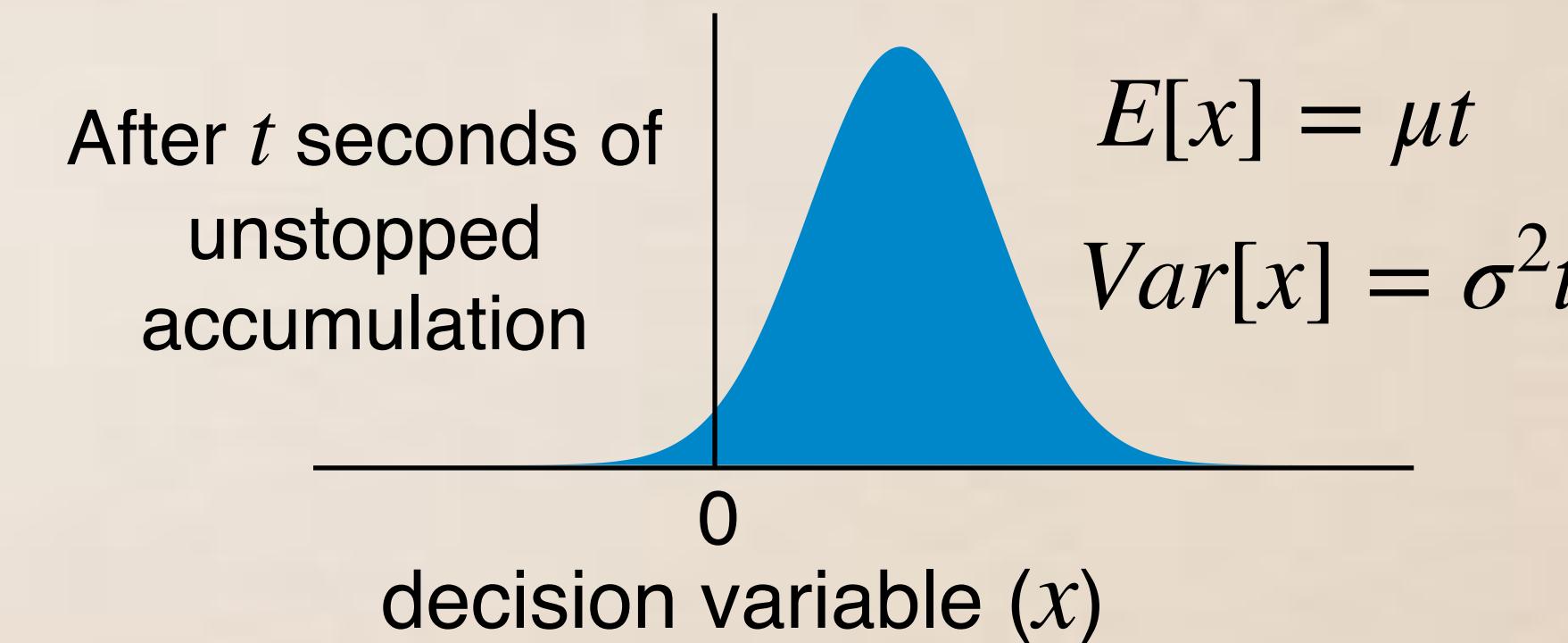
\begin{array}{c} \text{decision variable} \\ \diagup \quad \diagdown \\ x_{i+1} = x_i + e_i \\ \text{evidence} \end{array}

i : time step

- » Evidence samples e_i are independent and normally distributed

$$e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

\begin{array}{c} \text{time step} \\ \diagup \quad \diagdown \\ e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t) \\ \text{drift} \quad \text{diffusion} \\ \text{coefficient} \quad \text{coefficient} \end{array}



- » In the limit $\Delta t \rightarrow 0$, the decision process is described by a Wiener process with drift:

$$\frac{dx(t)}{dt} = \underbrace{\mu}_{\text{signal}} + \underbrace{\sigma dW(t)}_{\text{noise}}$$

\begin{array}{c} \text{decision variable} \\ \diagup \\ \frac{dx(t)}{dt} = \underbrace{\mu}_{\text{signal}} + \underbrace{\sigma dW(t)}_{\text{noise}} \end{array}

Hands on: Simulate the drift-diffusion process (no bounds for now)

[Do the Matlab poll]

Hands on: Simulate the drift-diffusion process (no bounds for now)

$$x_{i+1} = x_i + e_i$$

decision variable
evidence

$$x_0 = 0$$

i : time step

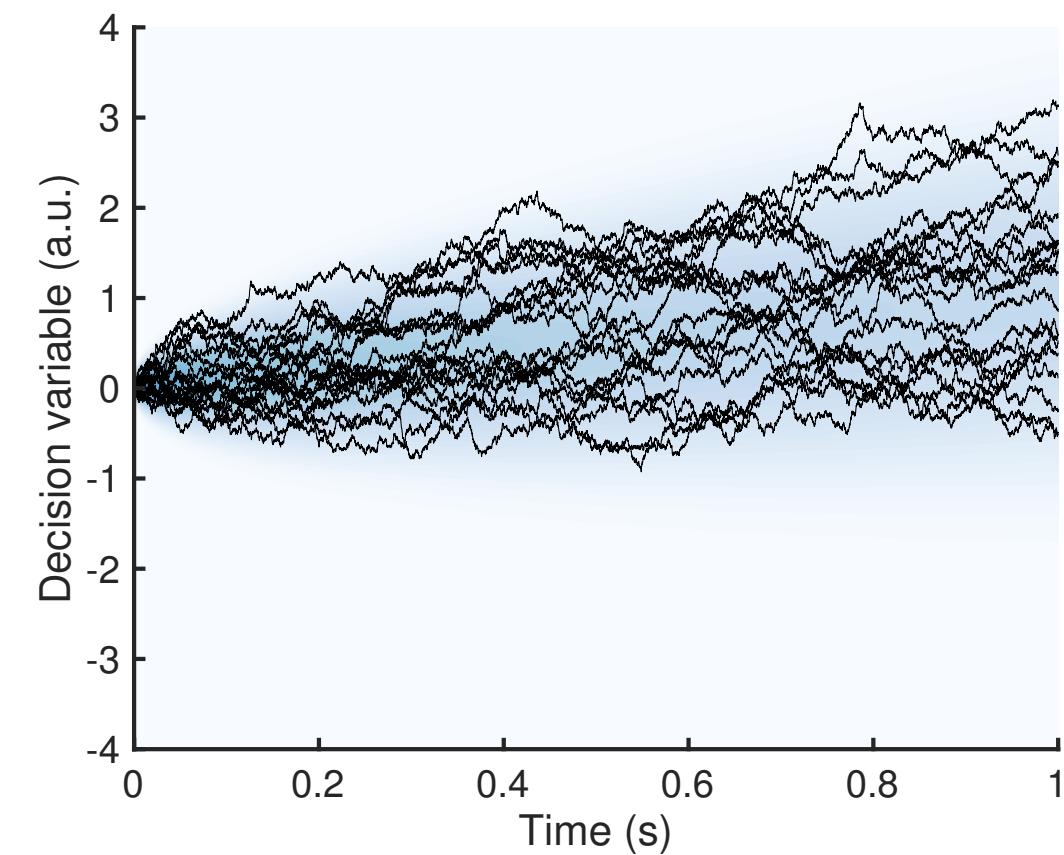
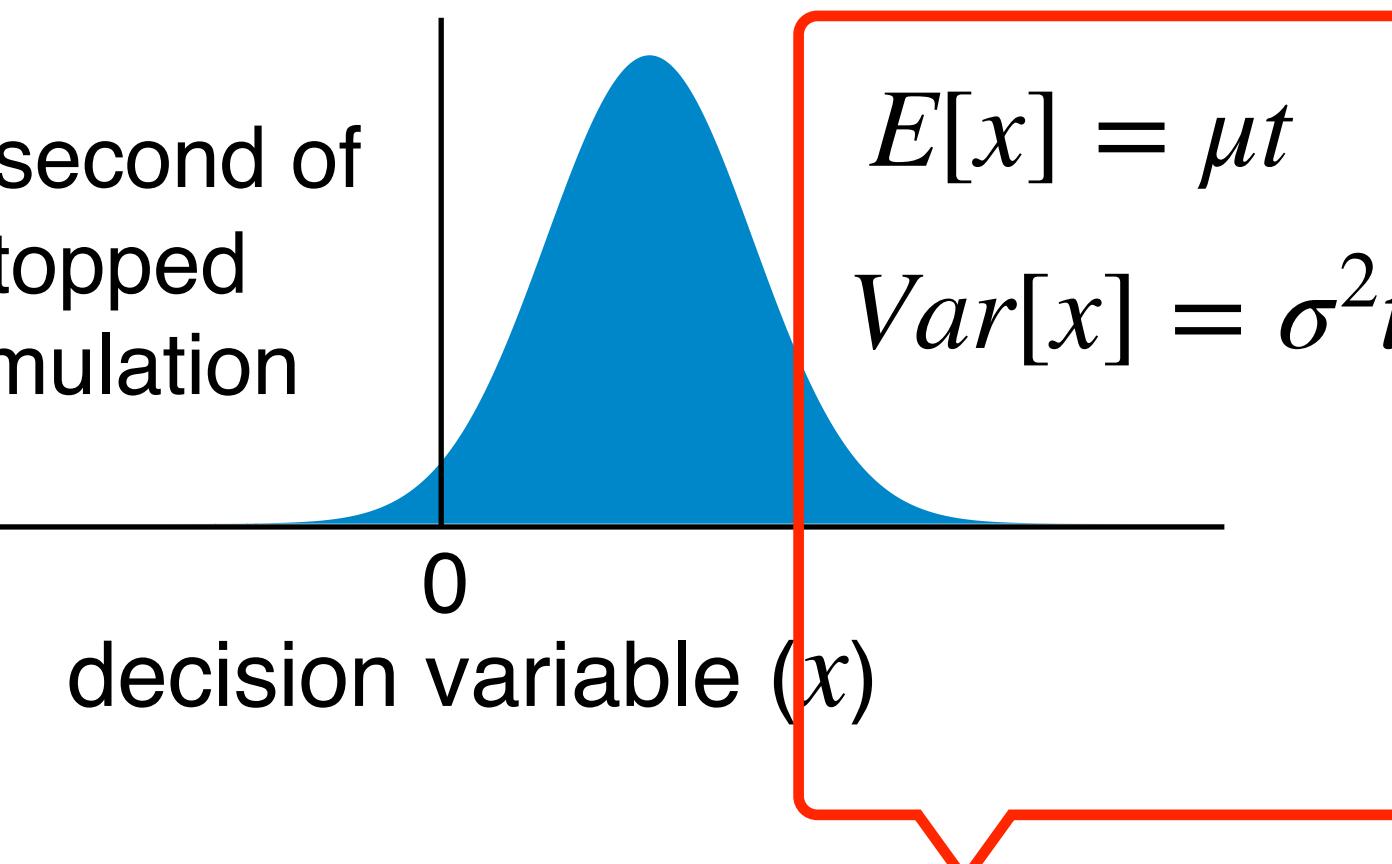
$$e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

time step
drift coefficient
diffusion coefficient

Parameters:

- NumTrials = 5000
- MaxTime = 1 second
- $\Delta t = 0.0001$ seconds
- $\mu = 1$
- $\sigma = 1$

After t second of
unstopped
accumulation



Key Matlab code:

```
% momentary evidence
e = mu * dt + sigma * sqrt(dt) * randn(NumTrials, length(T));
% accumulated evidence
dv = cumsum(e,2);
```

Simulate diffusion paths and
verify this relationship

```

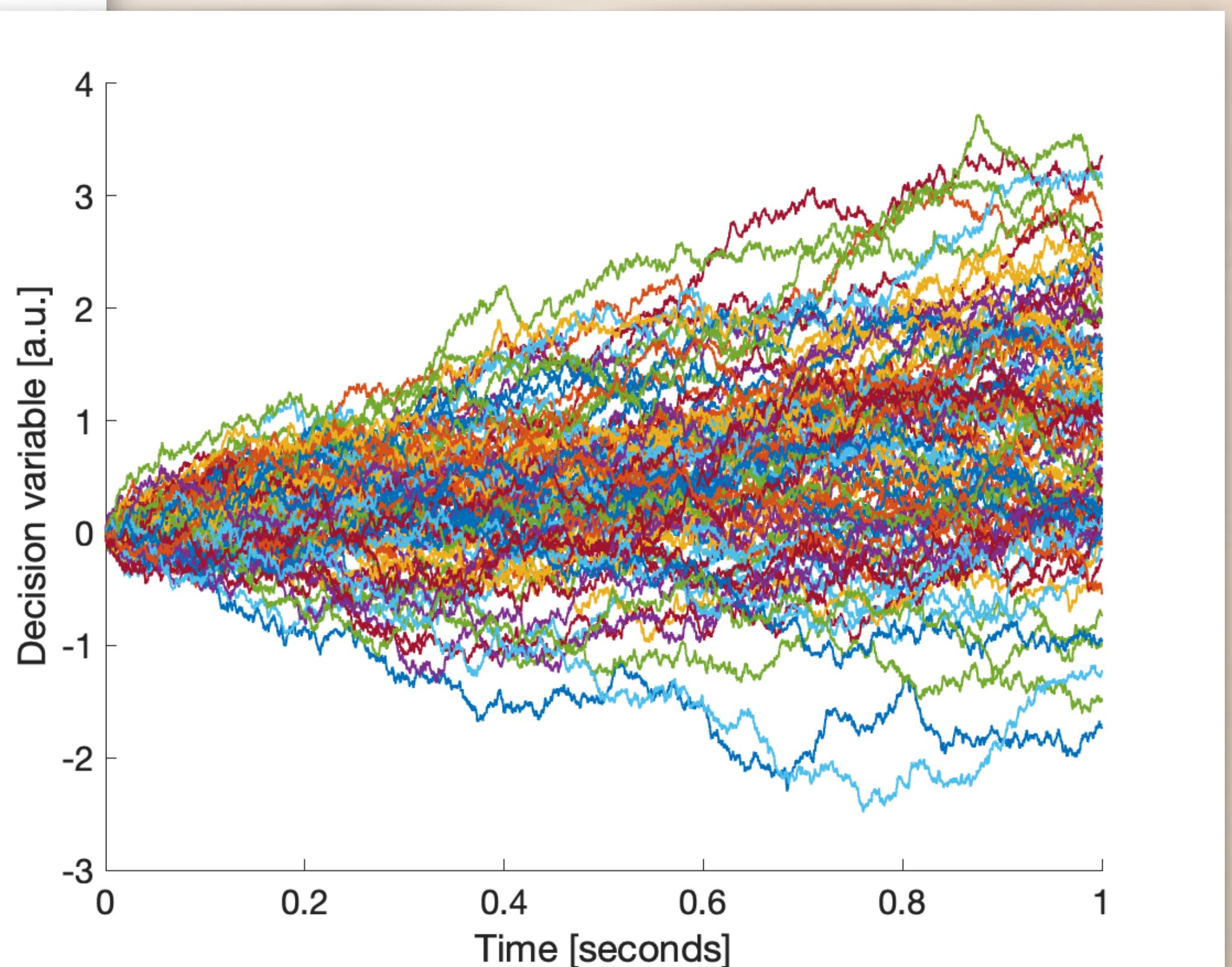
% parameters
NumTrials      = 5000;
MaxTime        = 1;
dt             = 0.0001;
sigma          = 1;
mu             = 1;
T = 0:dt:MaxTime;

% momentary evidence
e = mu * dt + sigma * sqrt(dt) * randn(NumTrials, length(T));

% accumulated evidence
dv = cumsum(e,2);

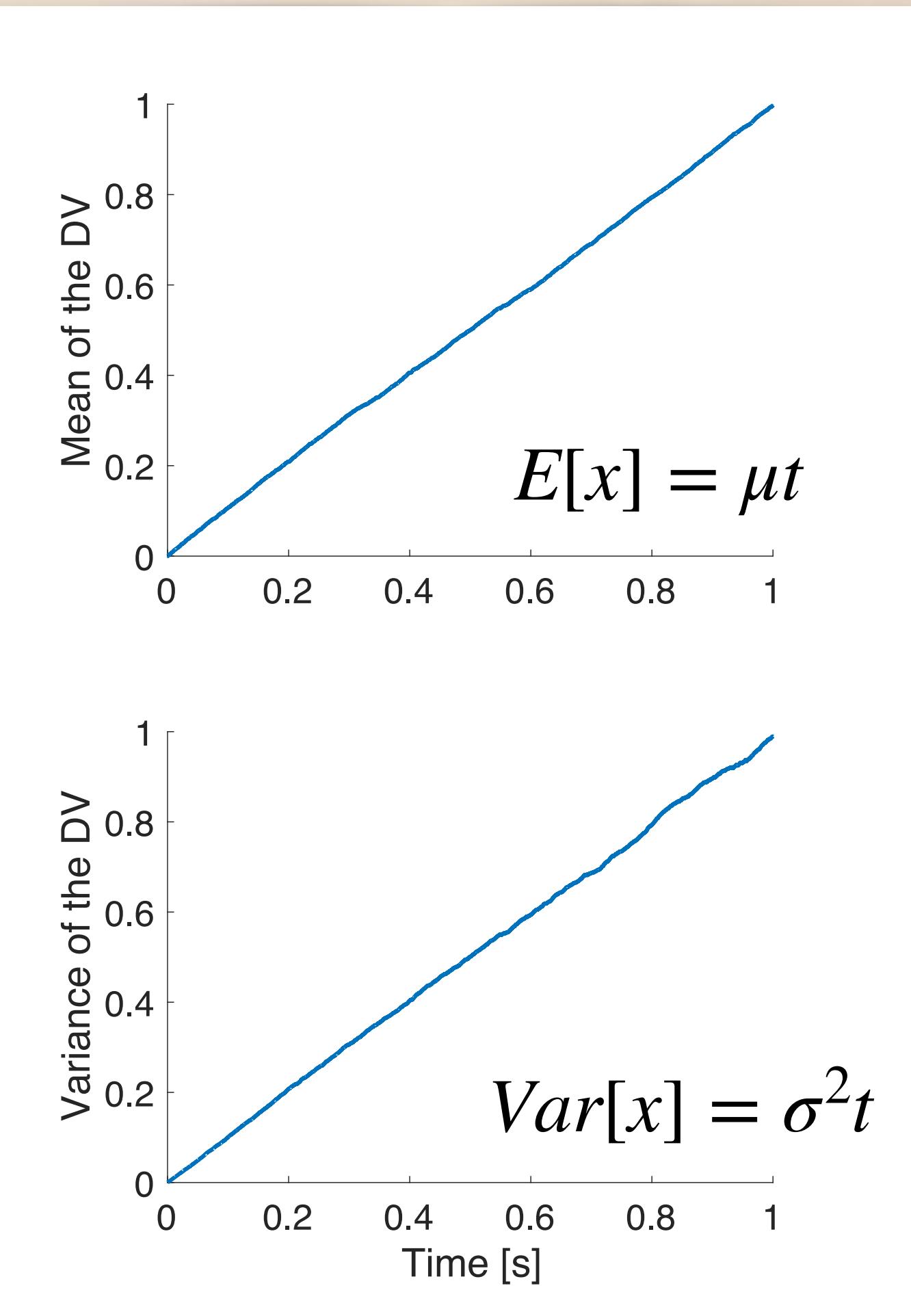
% plot
figure();
plot(T, dv(1:100,:)');
xlabel('Time [seconds]');
ylabel('Decision variable [a.u.]');

```

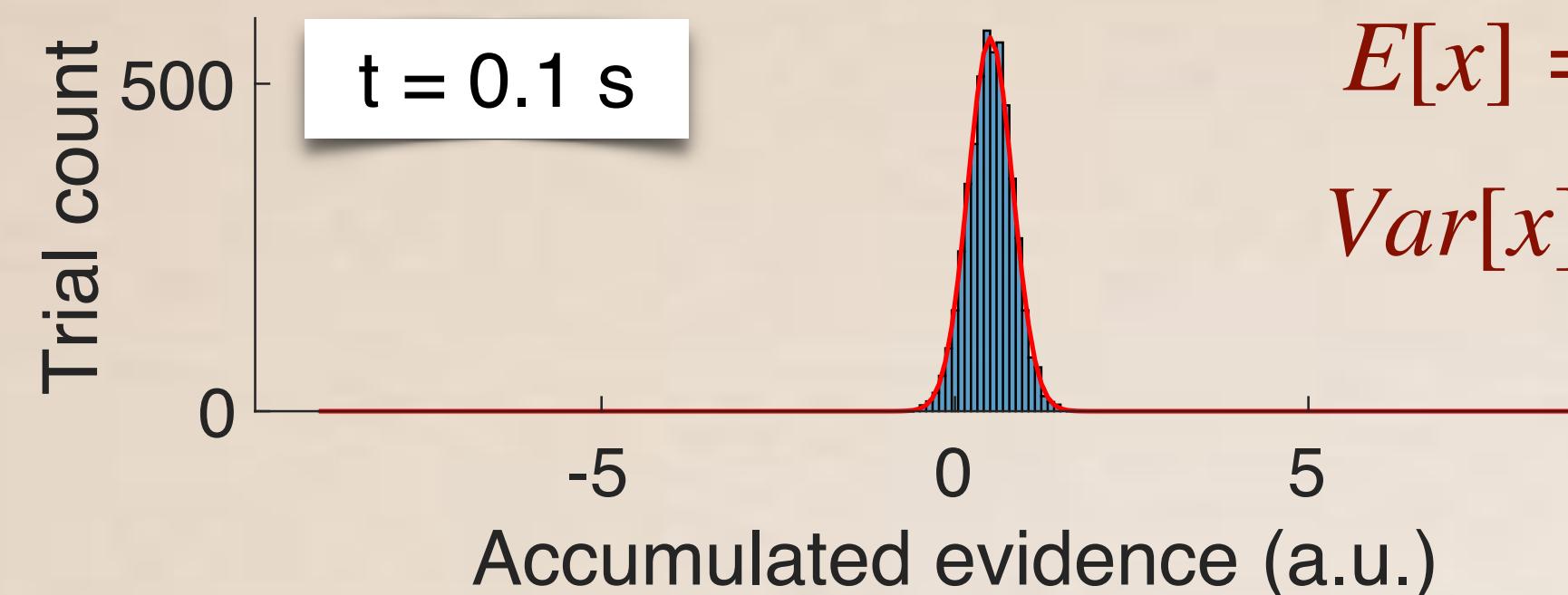


```
% mean and variance of accumulated evidence
figure();
subplot(2,1,1);
plot(T, mean(dv));
ylabel('Mean of the DV');

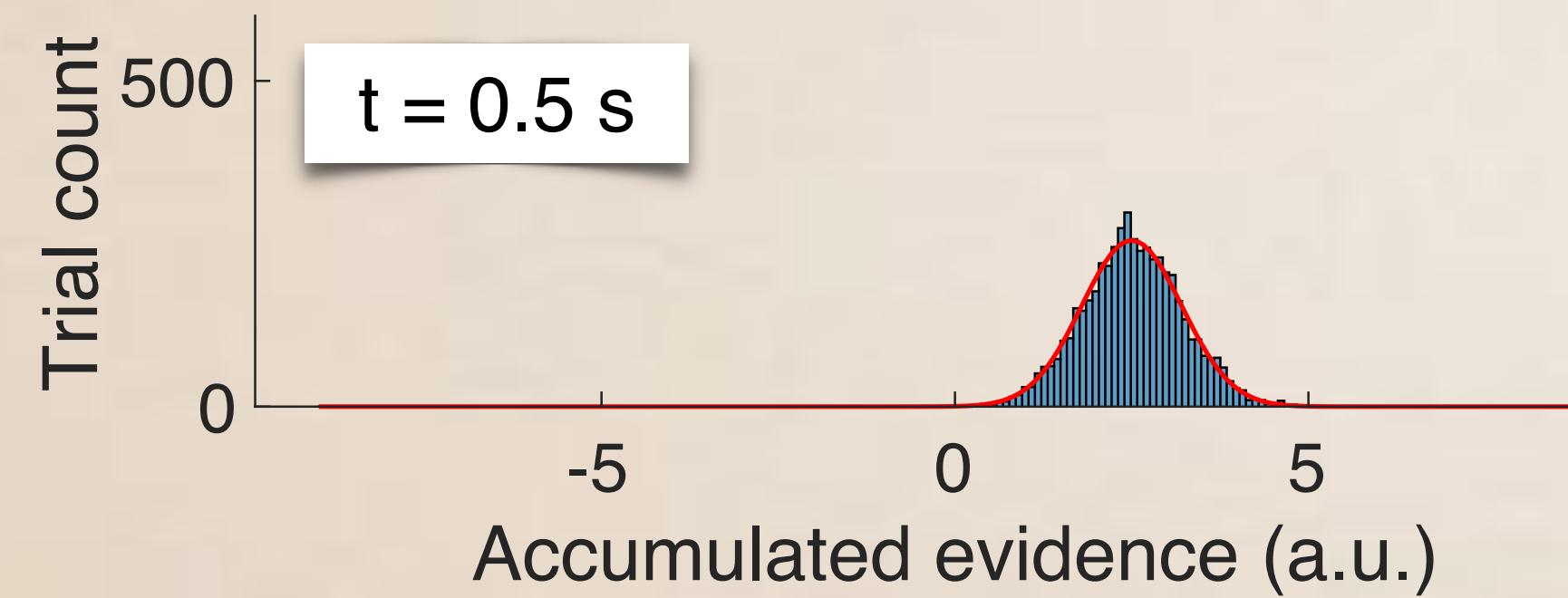
subplot(2,1,2);
plot(T, var(dv));
ylabel('Variance of the DV');
xlabel('Time [s]');
```



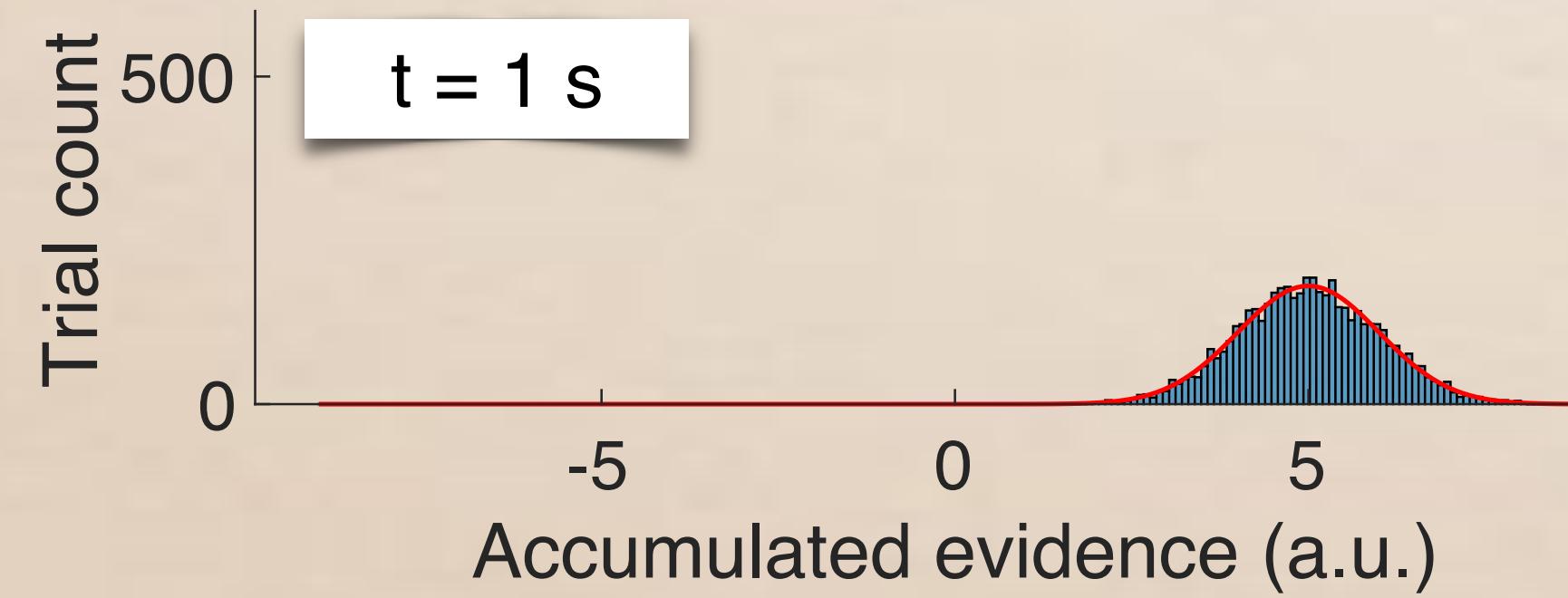
The state of the decision variable remains Gaussian over time



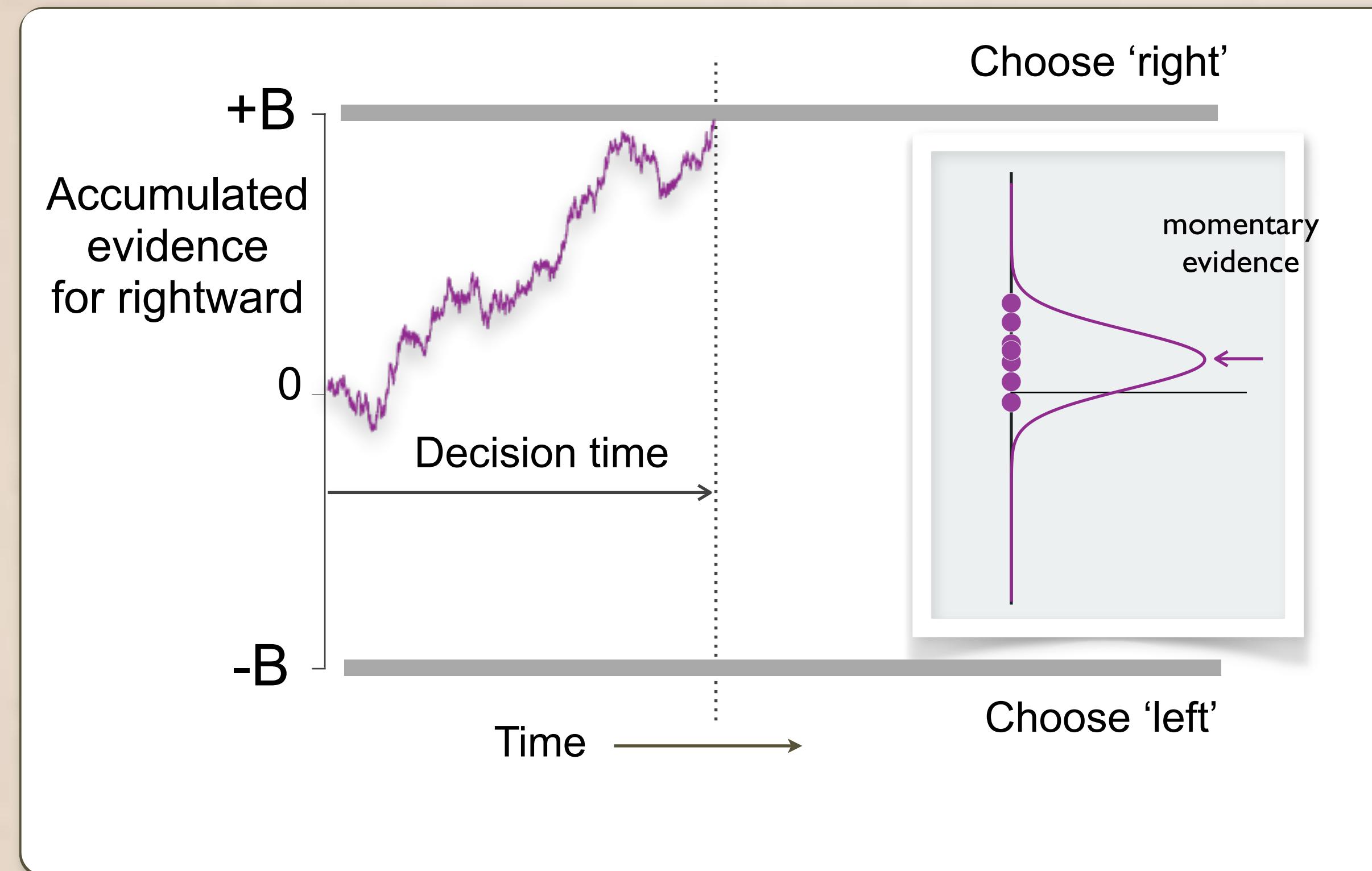
$$E[x] = \mu t$$
$$\text{Var}[x] = \sigma^2 t$$



No longer true when
we add bounds...



The decision terminates when the accumulated evidence reaches a bound



Sample paths:

$$e_i = \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

$$x_{i+1} = x_i + e_i$$

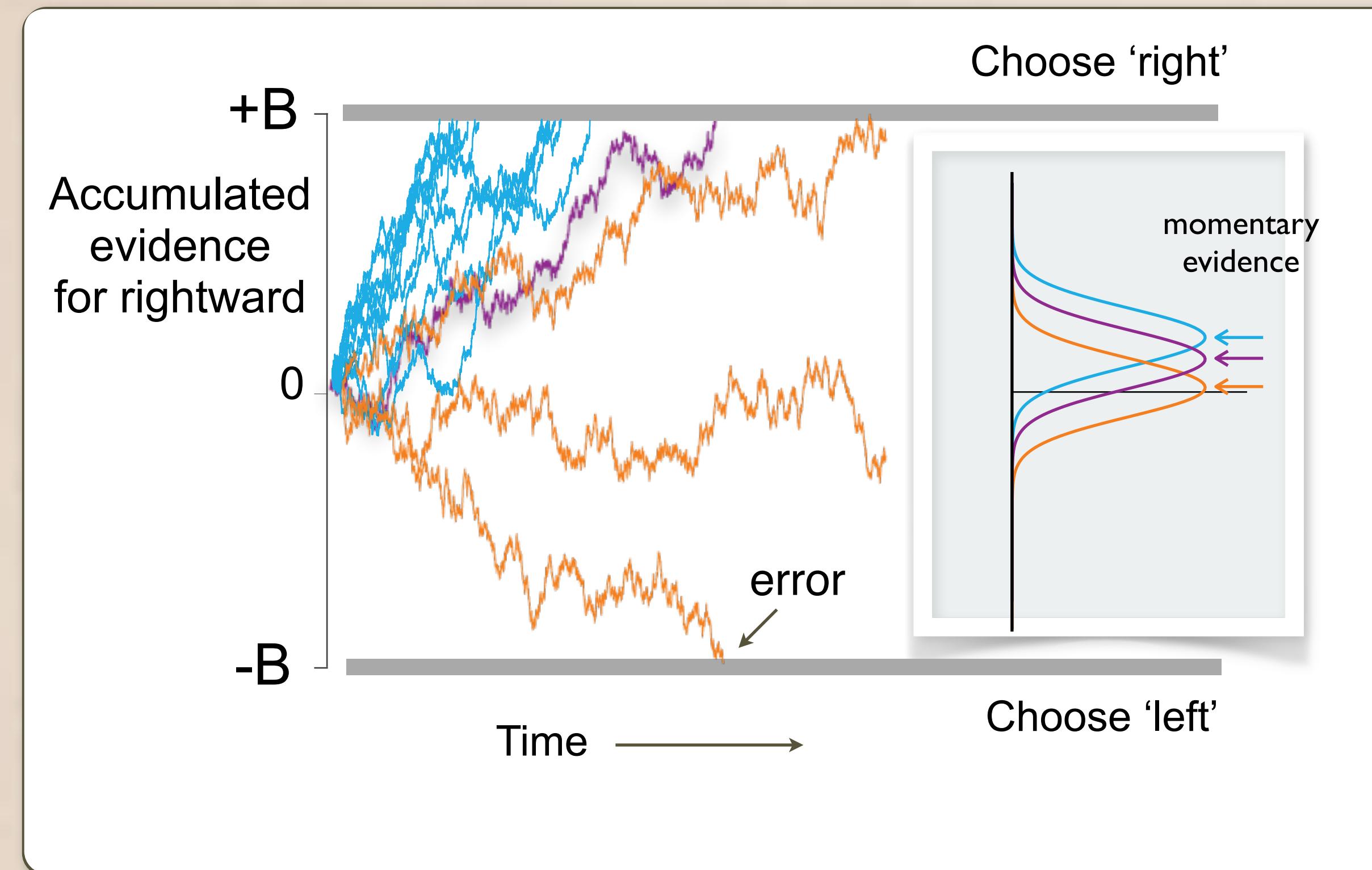
$$x_0 = 0$$

Motion task:

$$\mu = \kappa \cdot \frac{\text{Motion coherence}}{\text{Motion coherence}}$$

$|\mu|$ Increases with motion strength

The decision terminates when the accumulated evidence reaches a bound



Sample paths:

$$e_i = \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

$$x_{i+1} = x_i + e_i$$

$$x_0 = 0$$

Motion task:

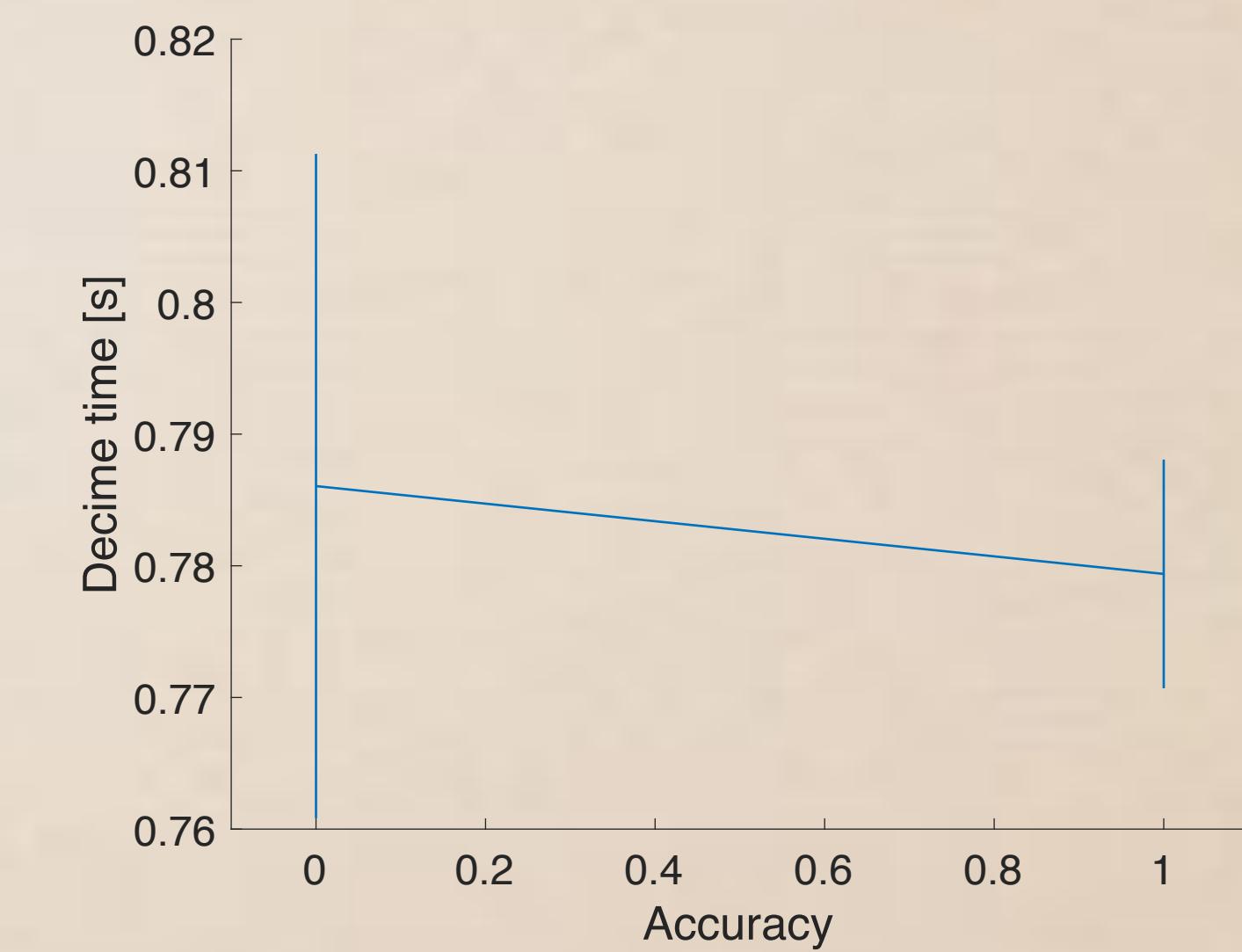
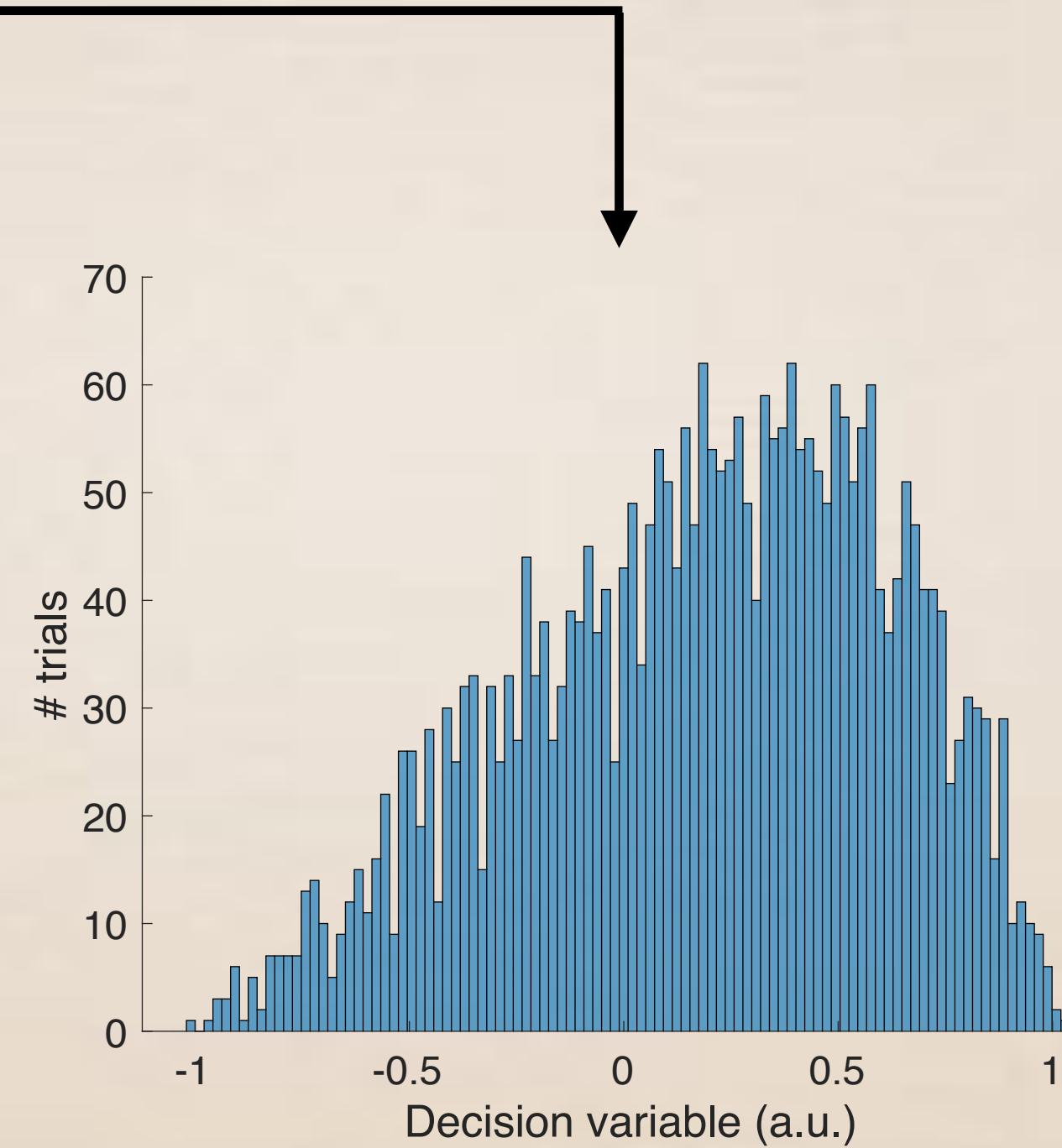
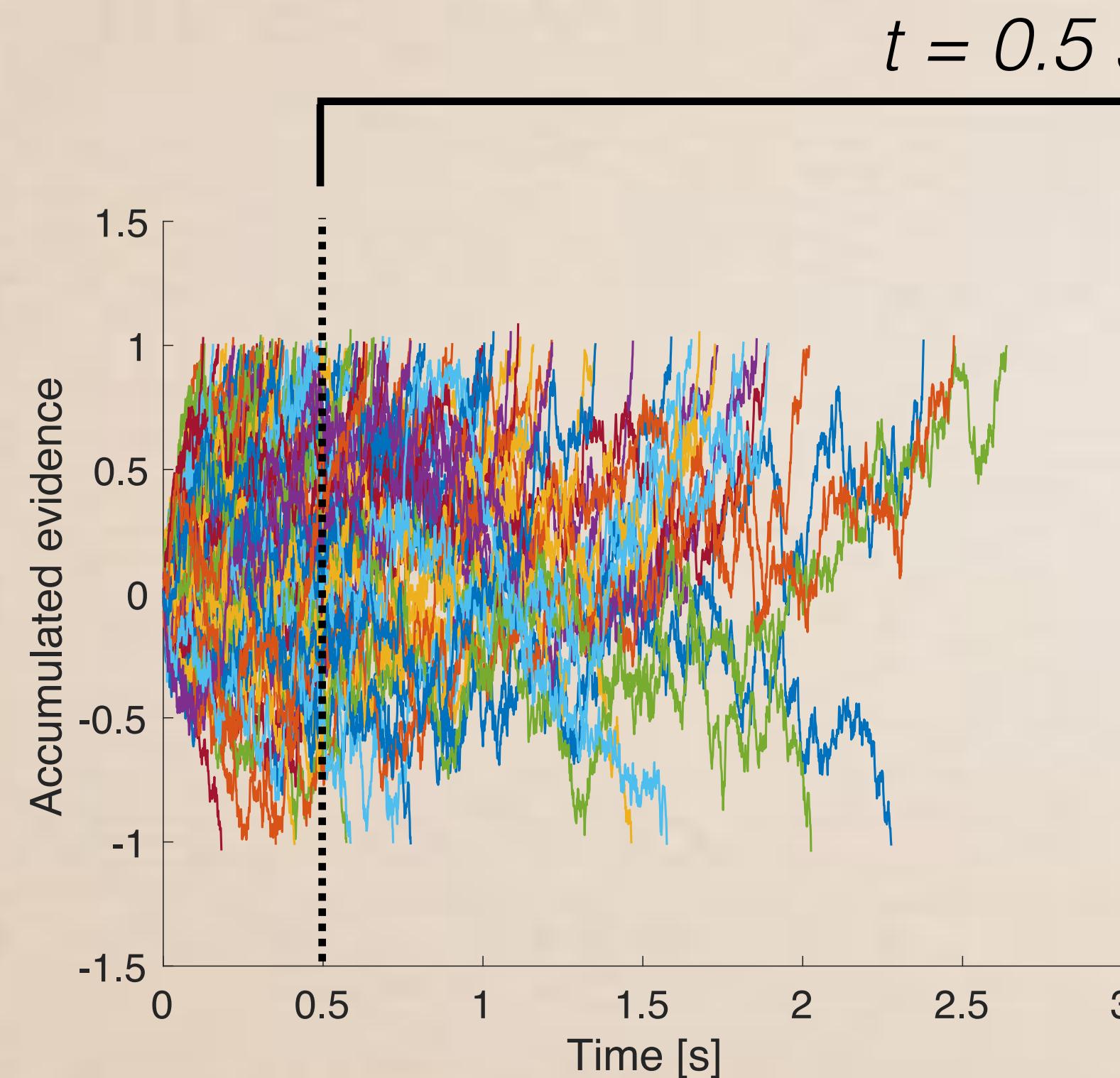
$$\mu = \kappa \cdot \frac{\text{Motion coherence}}{\text{coherence}}$$

$|\mu|$ Increases with motion strength

Add bounds to the simulated trajectories

Code in “02_Simulate_diffusion_with_bounds/main.m”

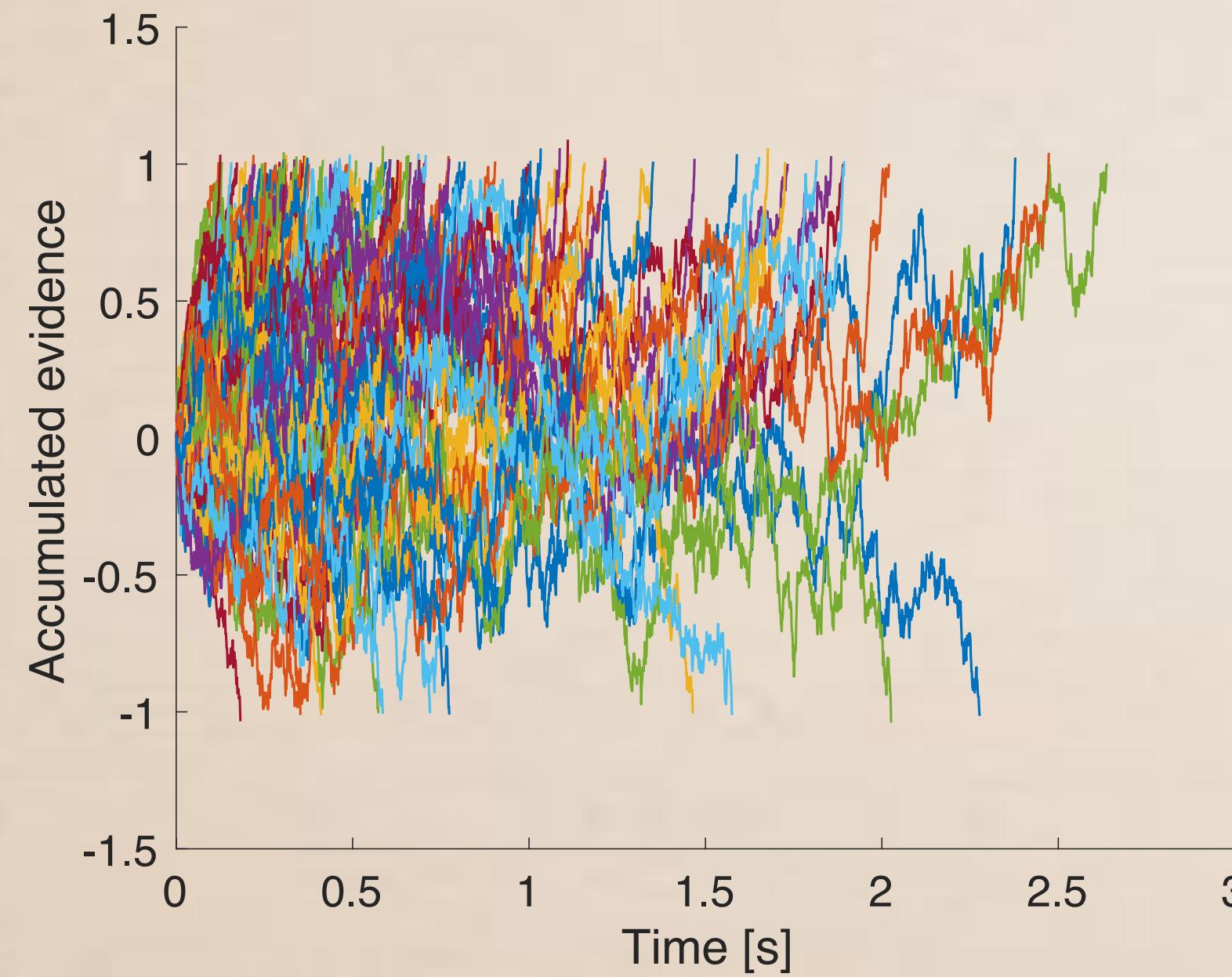
- » Added symmetric bounds at: $B = \pm 1$



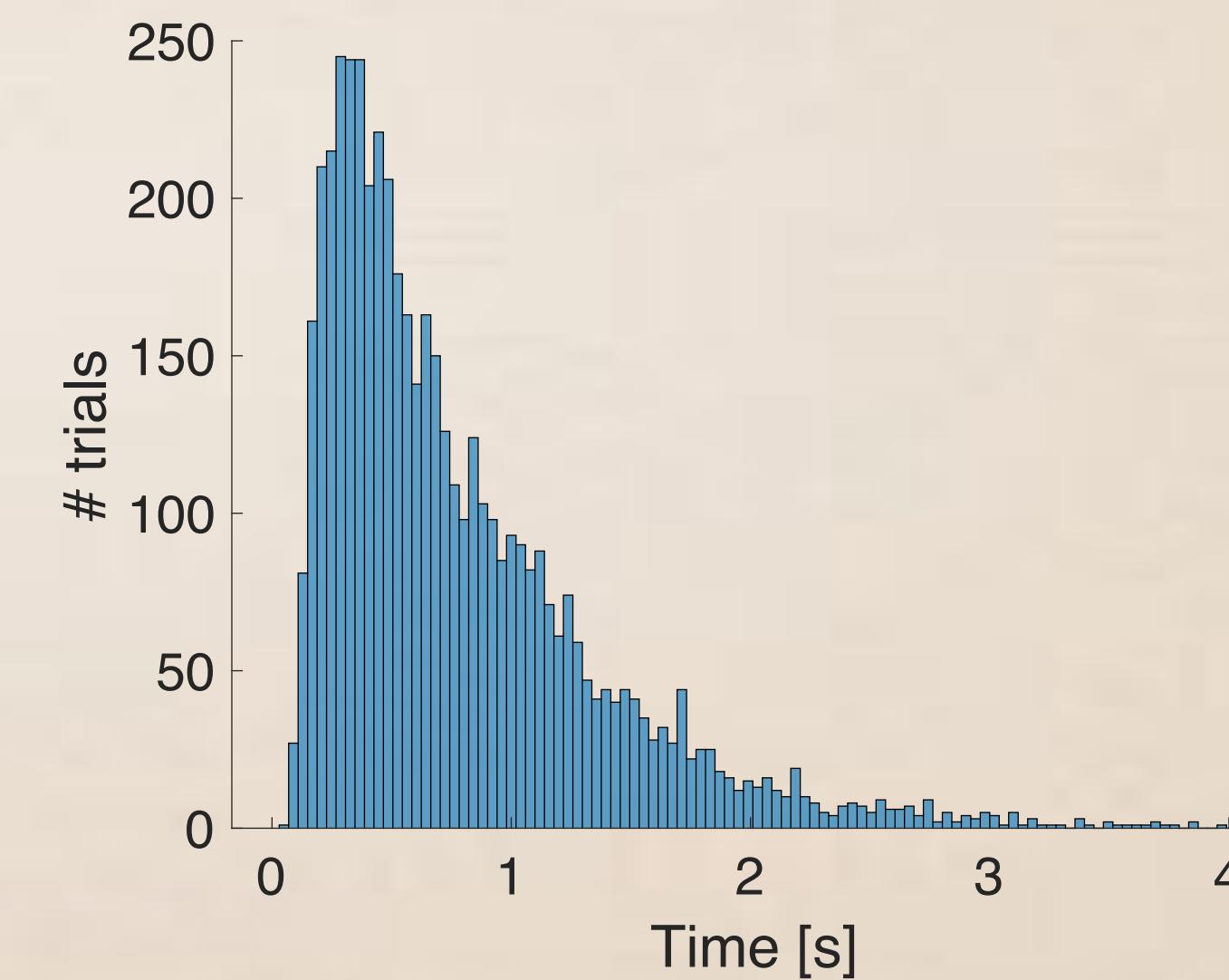
Add bounds to the simulated trajectories

Code in “02_Simulate_diffusion_with_drift/main.m”

- » Added symmetric bounds at: $B = \pm 1$

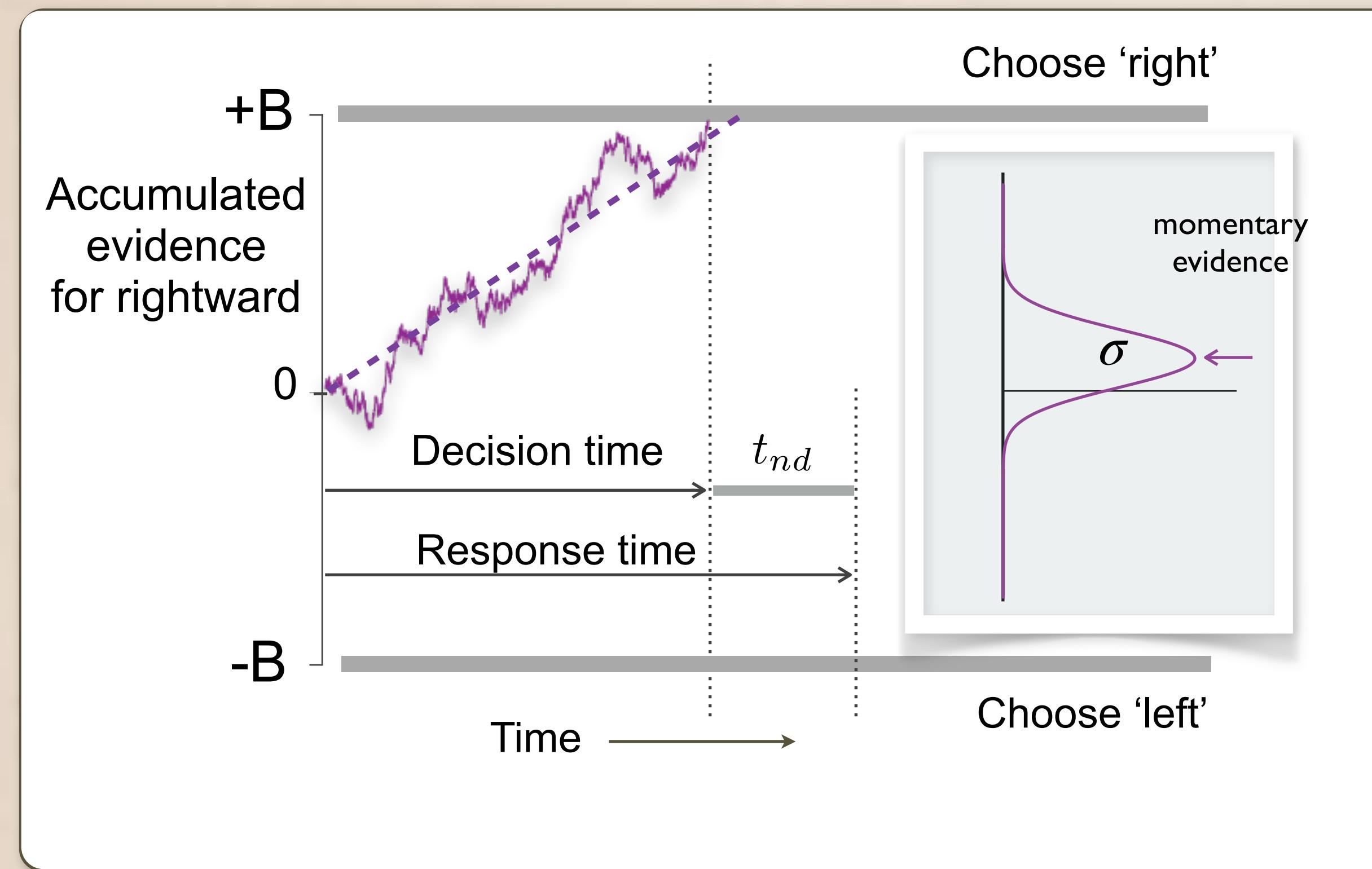


Some trajectories



*Response time
distributions have long
tails*

DDM, simplest version



Parameters in
the 'simple' version:

- » Drift rate, μ $\times 2$
- » Bound height, B $\times 2$
- » Non-decision time, t_{nd}
- » Diffusion coefficient, $\sigma = 1$ $\times 2$

Analytical solution to the choice and response time functions

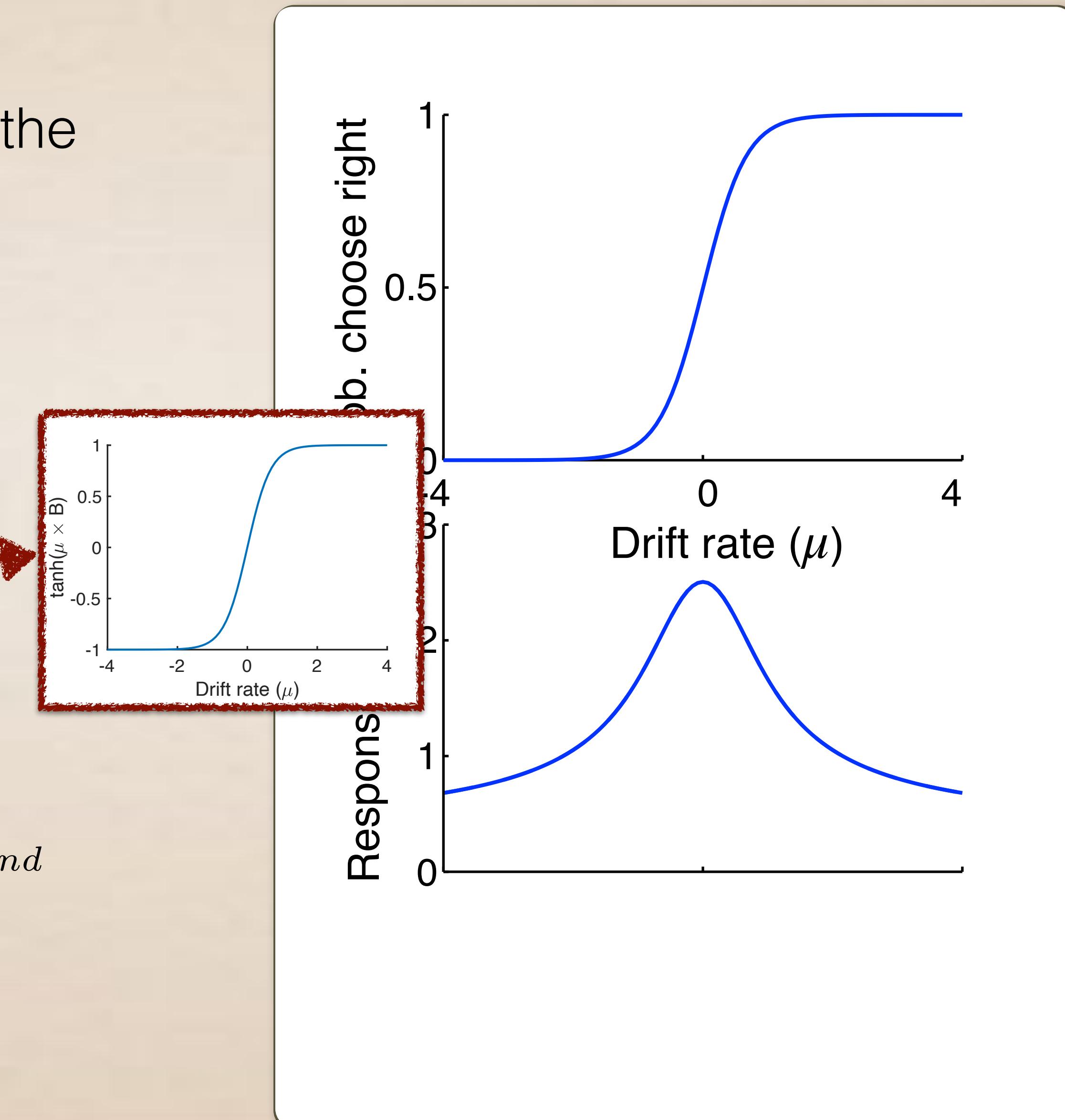
Probability of terminating at the upper bound:

$$p_+ = \frac{1}{1 + e^{-2\mu B}}$$

Average response time:

$$T = \frac{B}{\mu} \tanh(\mu B) + t_{nd}$$

When $\mu = 0$: $T = B^2 + t_{nd}$



Effect of each model parameter on choice and RT

» Play with it yourself: <https://rb.gy/f6aoxr>

Drift Diffusion Model (DDM), Choice and Response Time Functions

Psychometric Function:

$$P(\text{right}) = \frac{1}{1 + \exp(-2 \cdot \mu \cdot B)}$$

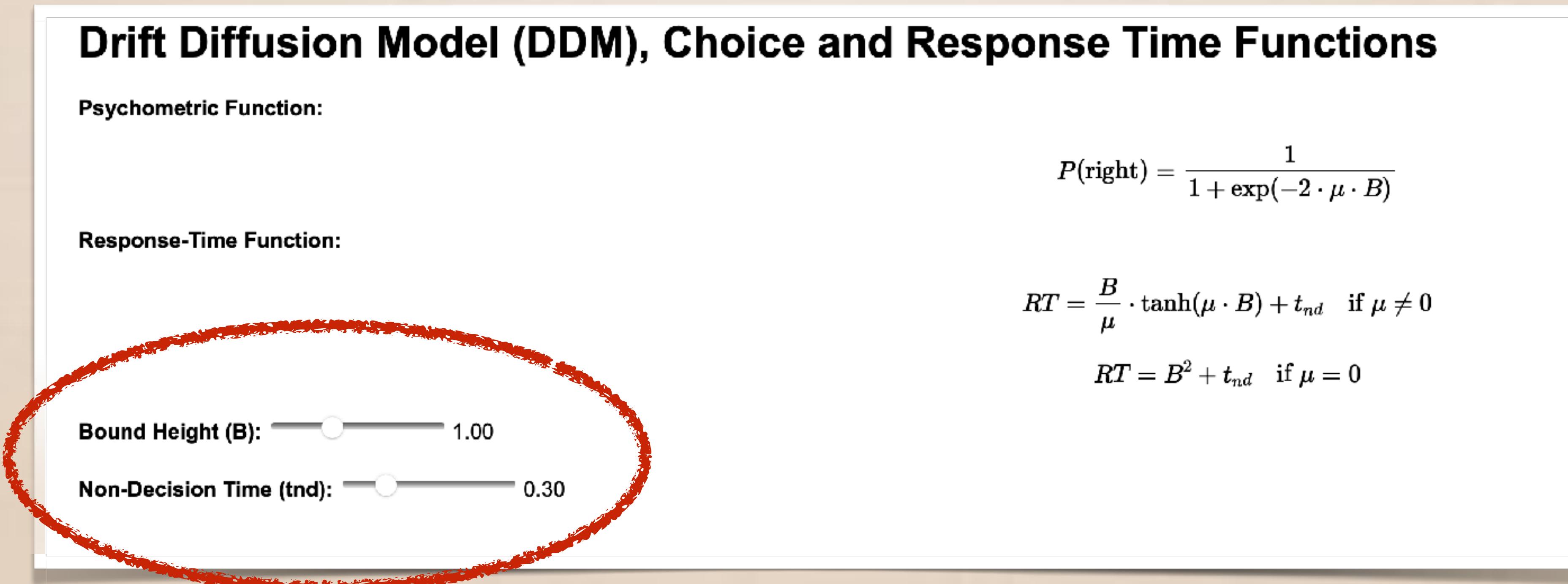
Response-Time Function:

$$RT = \frac{B}{\mu} \cdot \tanh(\mu \cdot B) + t_{nd} \quad \text{if } \mu \neq 0$$

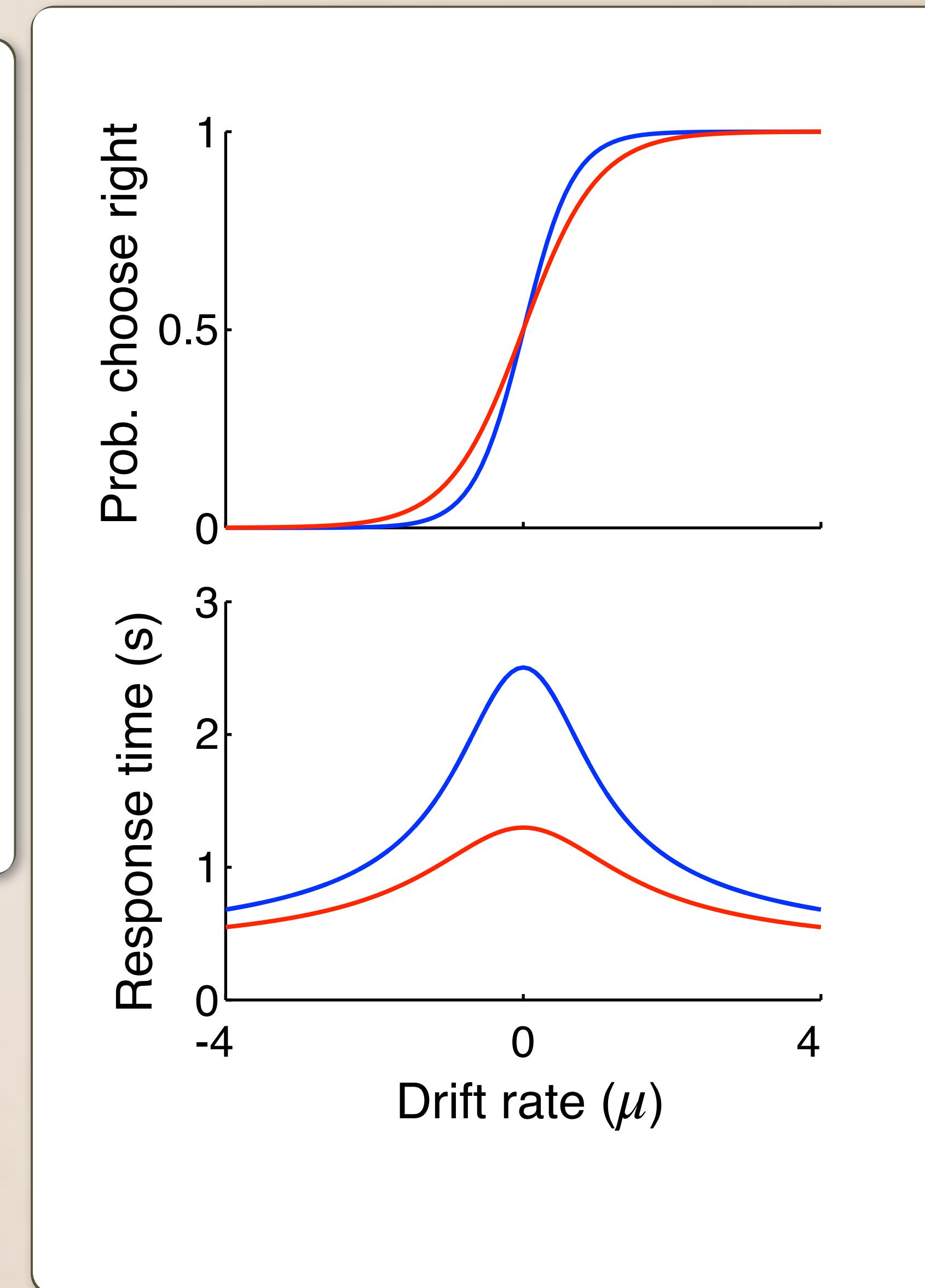
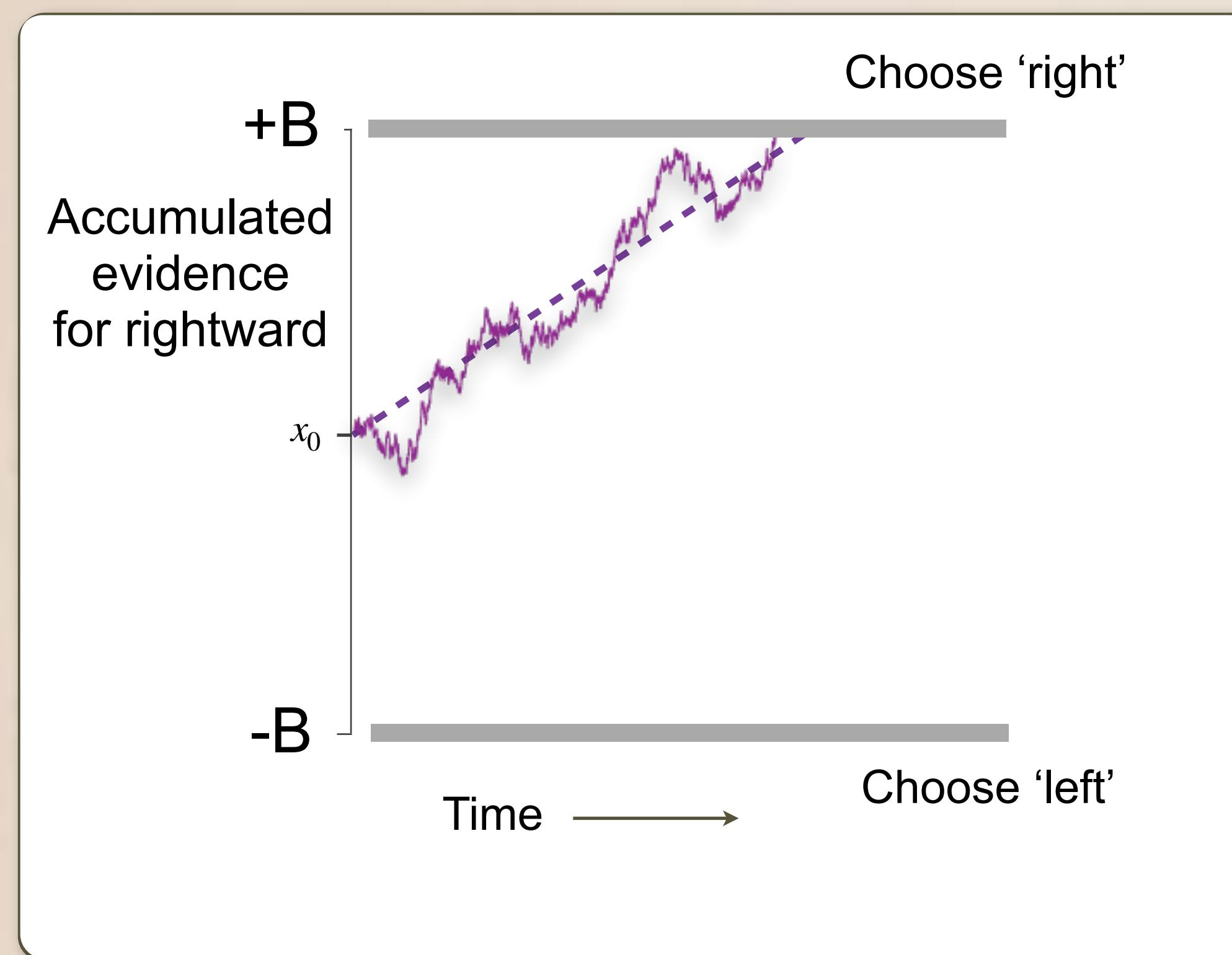
$$RT = B^2 + t_{nd} \quad \text{if } \mu = 0$$

Bound Height (B):

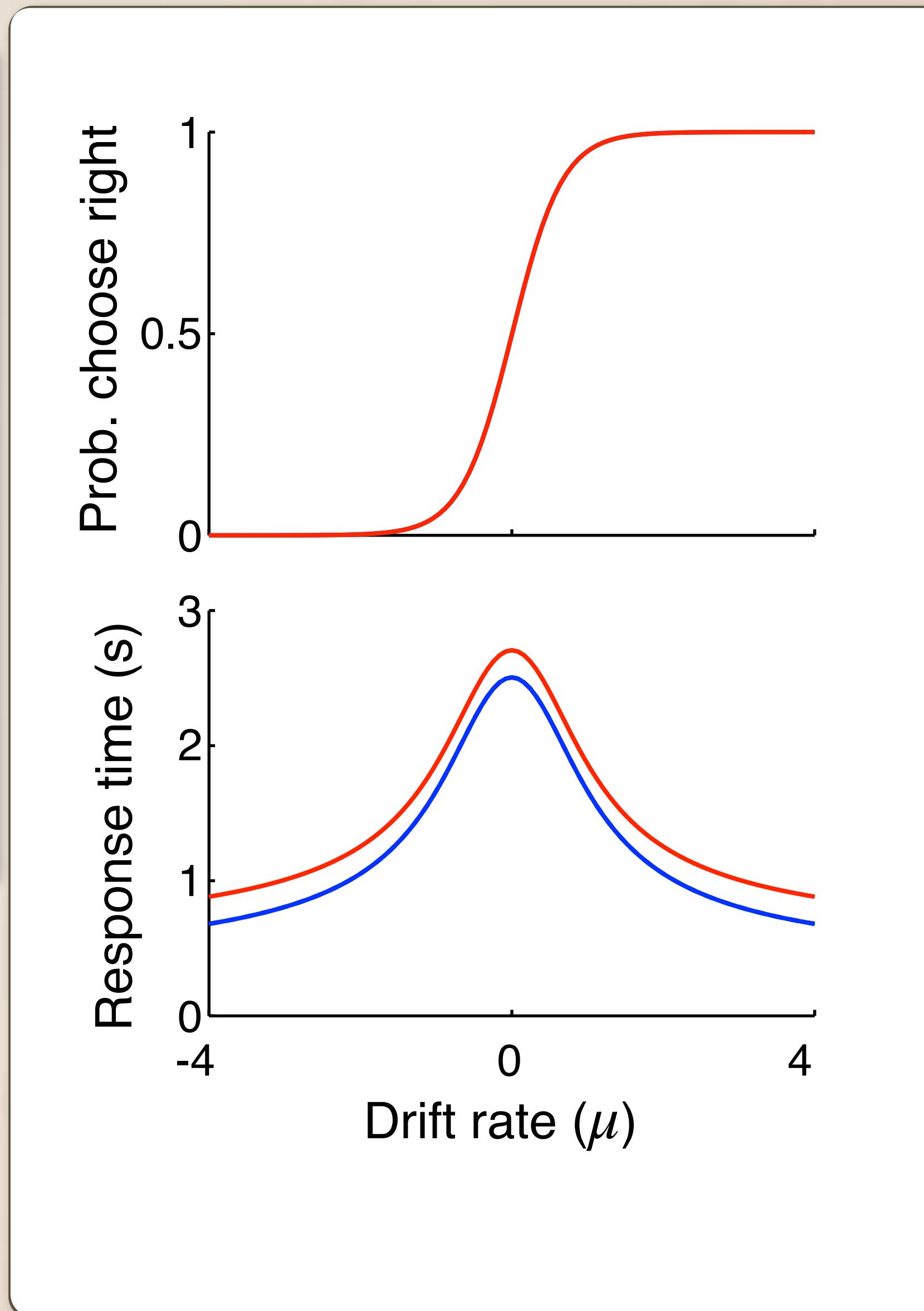
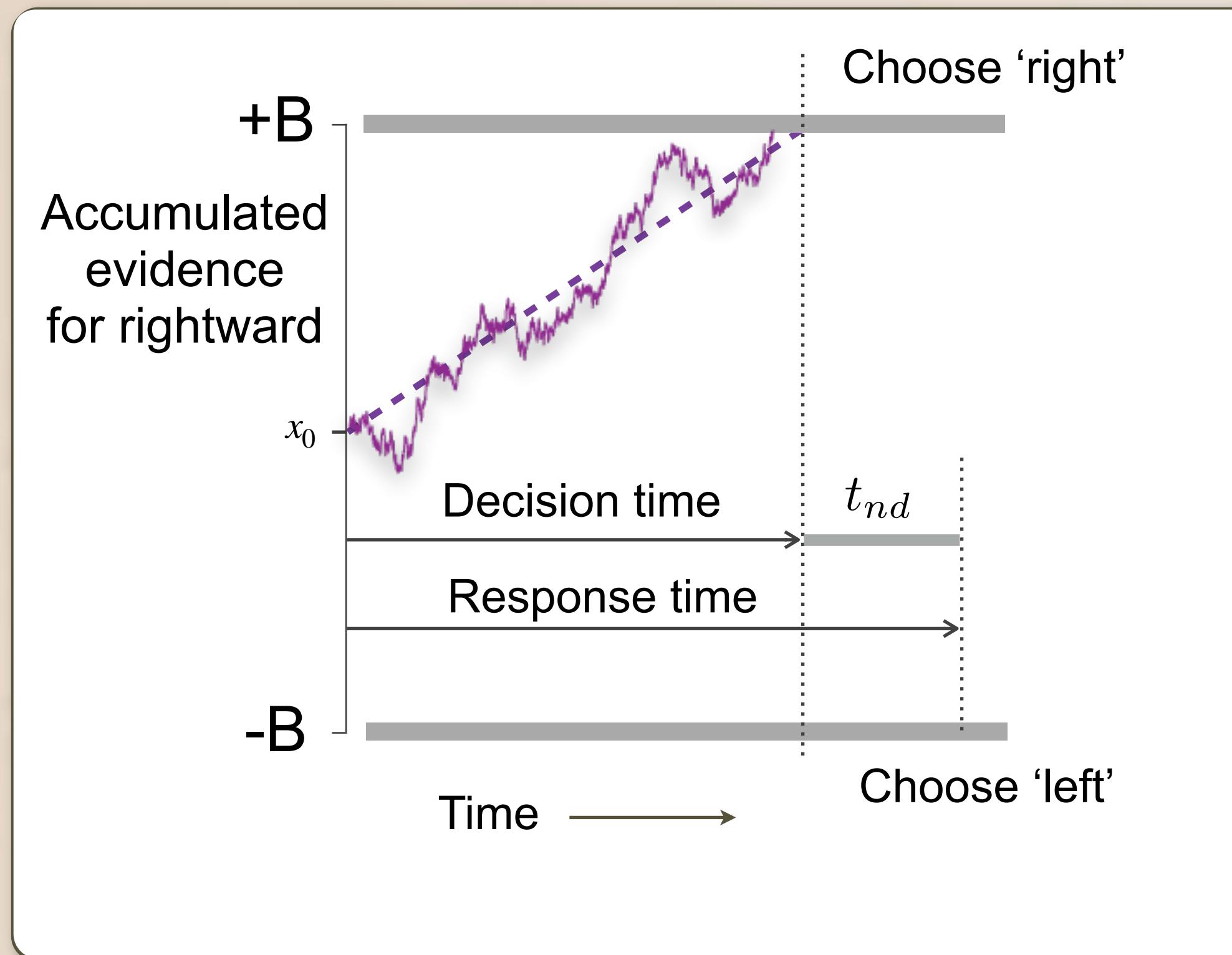
Non-Decision Time (tnd):



Effect of a change in bound height

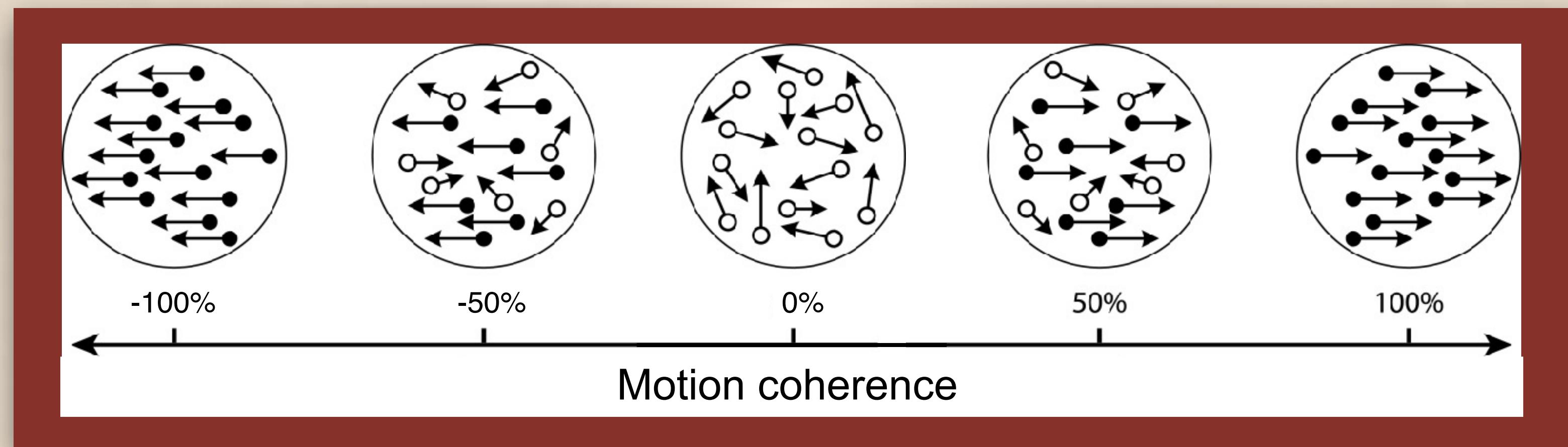


Effect of a change in non-decision time



How to model an actual task?

Need to link parameters of the task to those of the model



Momentary evidence in the DDM

$$e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

time step
drift coefficient diffusion coefficient

Make the drift and diffusion coefficients a function of the coherence

$$\mu = f(C)$$

signal-to-noise
Motion coherence
Linear

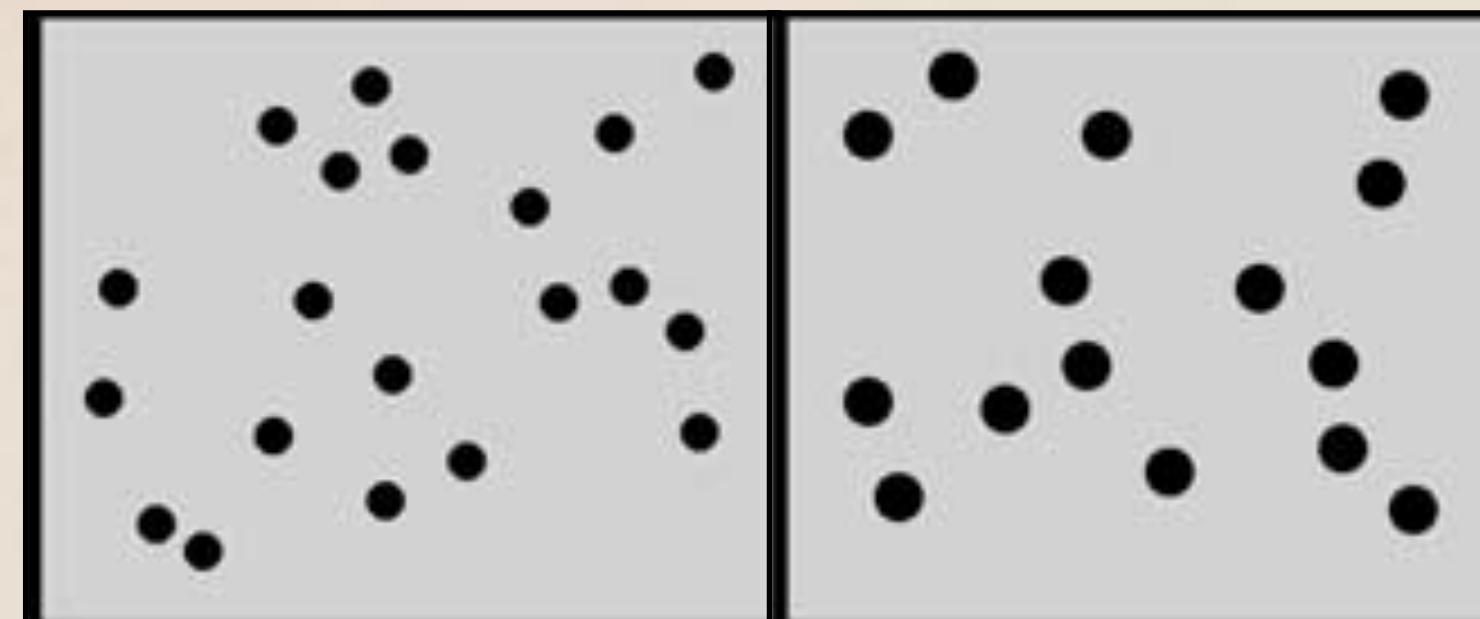
$$\sigma^2 = g(C)$$

Constant

How to model an actual task?

Need to link parameters of the task to those of the model

Which display has more dots?



Momentary evidence in the
DDM

$$e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

time step

drift coefficient diffusion coefficient

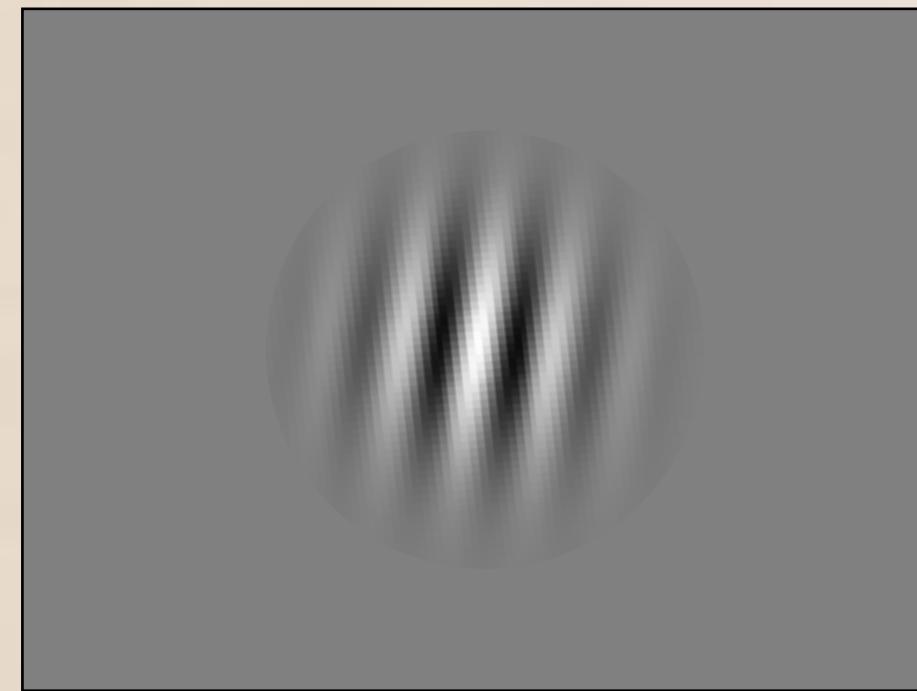
$$\mu = \kappa \times (\text{NumDotsRight} - \text{NumDotsLeft})$$

$$\sigma^2 = 1 + \alpha(\text{NumDotsRight} + \text{NumDotsLeft})$$

How to model an actual task?

Need to link parameters of the task to those of the model

Is orientation clockwise or
anti-clockwise from vertical?



$$\mu = ?$$

Which snack do you prefer?



$$\mu = ?$$

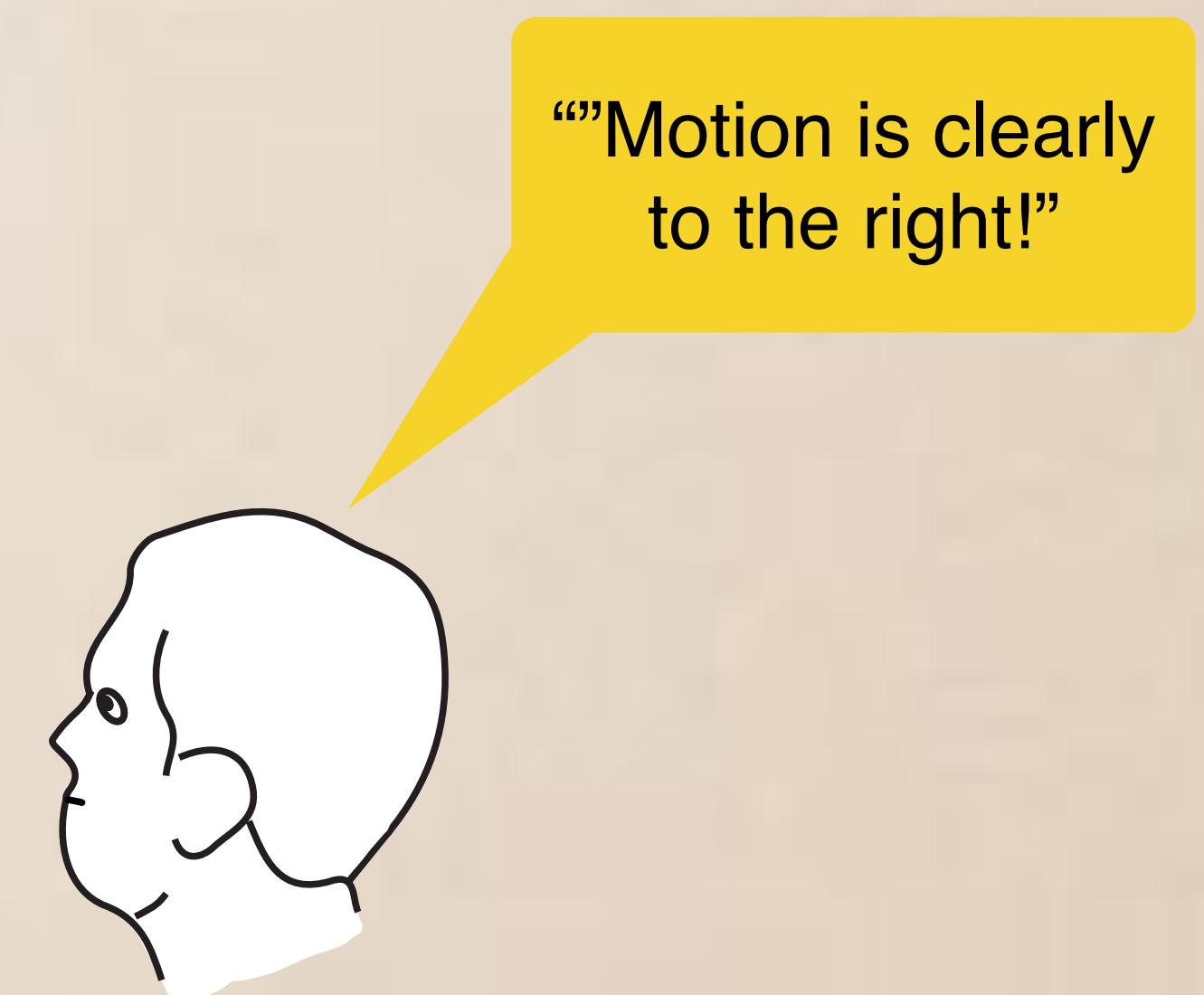
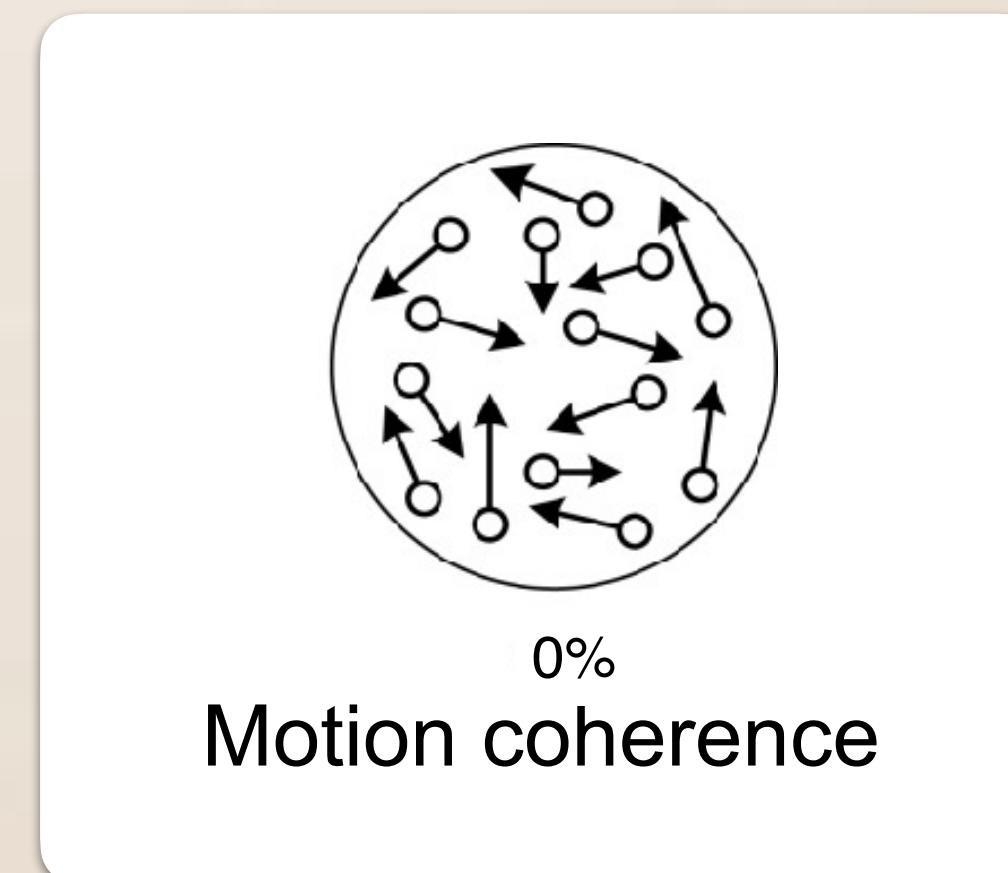
Modeling choice biases

In the “simple” DDM, when the momentary evidence is just noise ($\mu = 0$), both choice options are equally likely

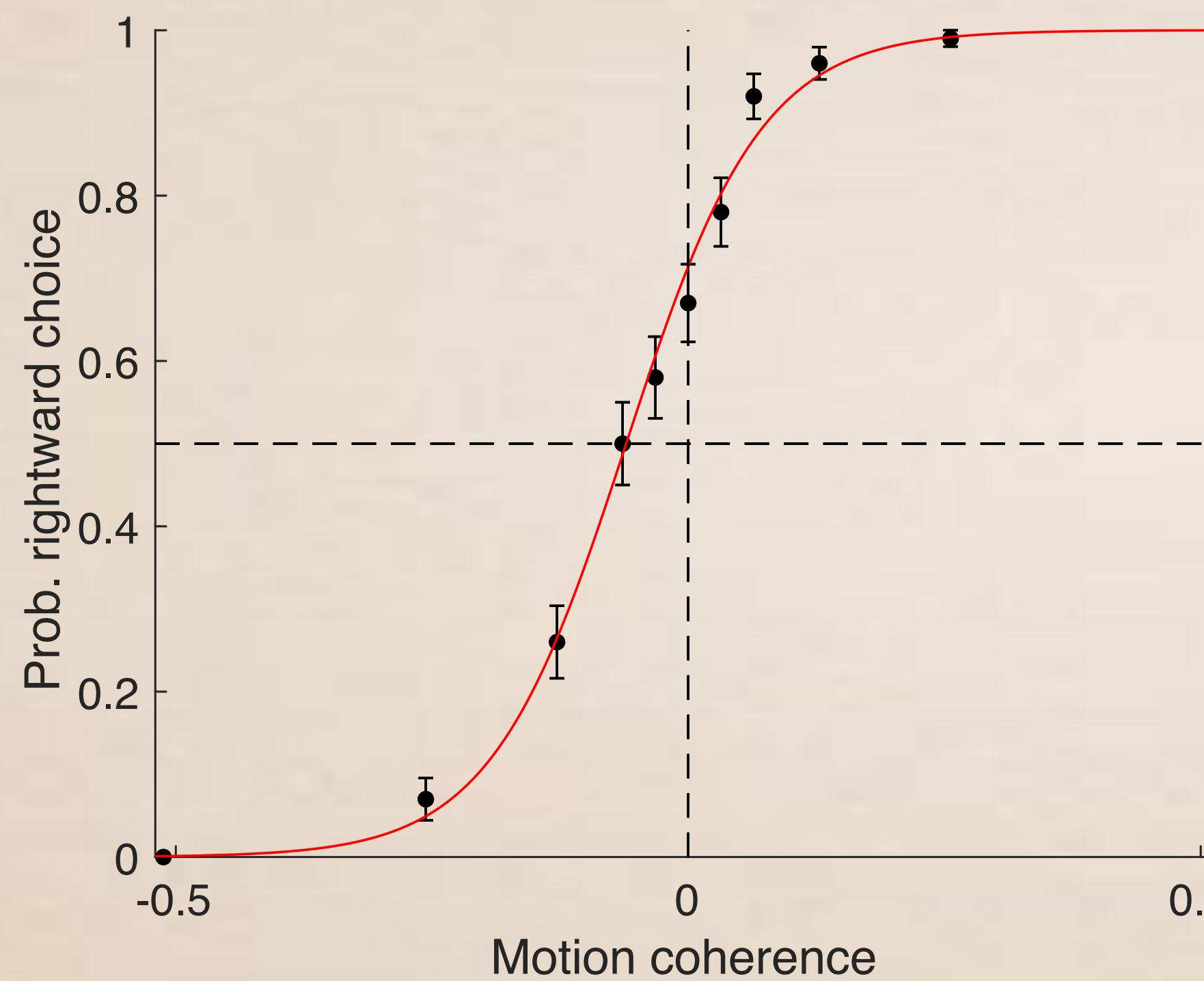
Probability of terminating at the upper bound:

$$p_+ = \frac{1}{1 + e^{-2\mu B}} \bigg|_{\mu=0} = \frac{1}{2}$$

Yet, sometimes decision makers have a systematic bias favoring one of the choice alternatives



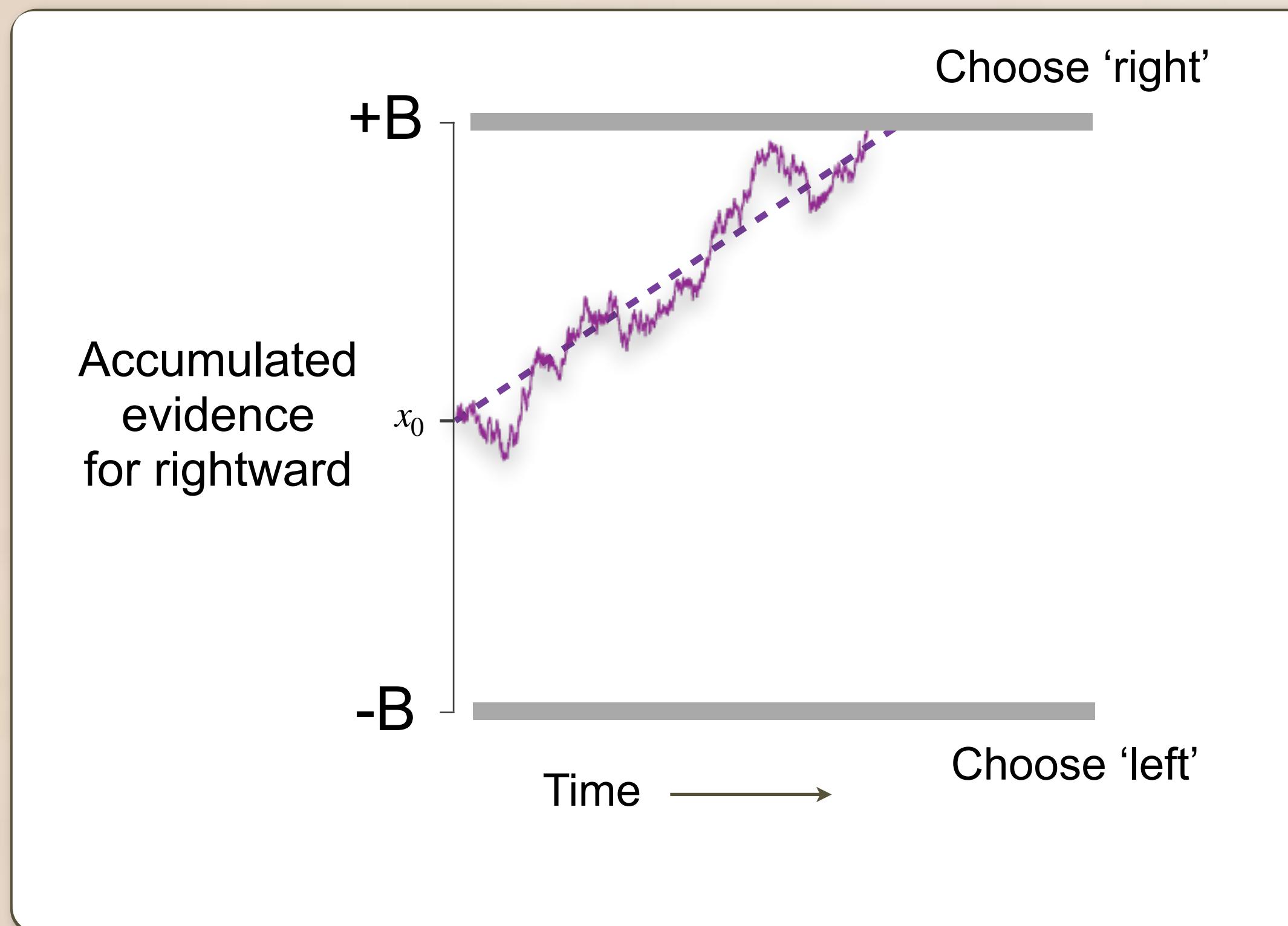
Modeling choice biases



$$p_{\text{rightward}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \text{coh})}}$$

- Bias in the DDM model:
- Offset to the starting point
 - Offset to the drift rate

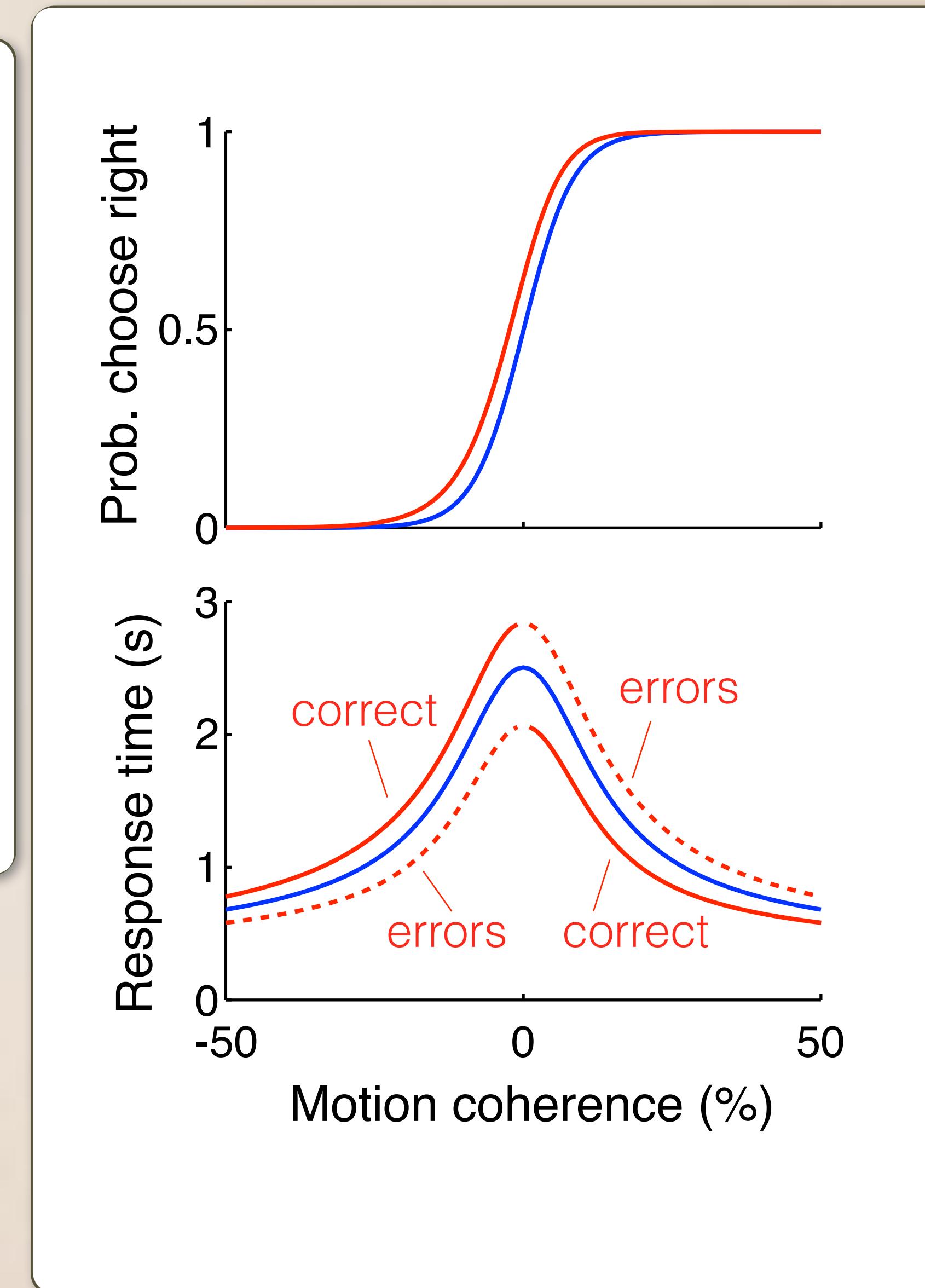
Offset in starting point



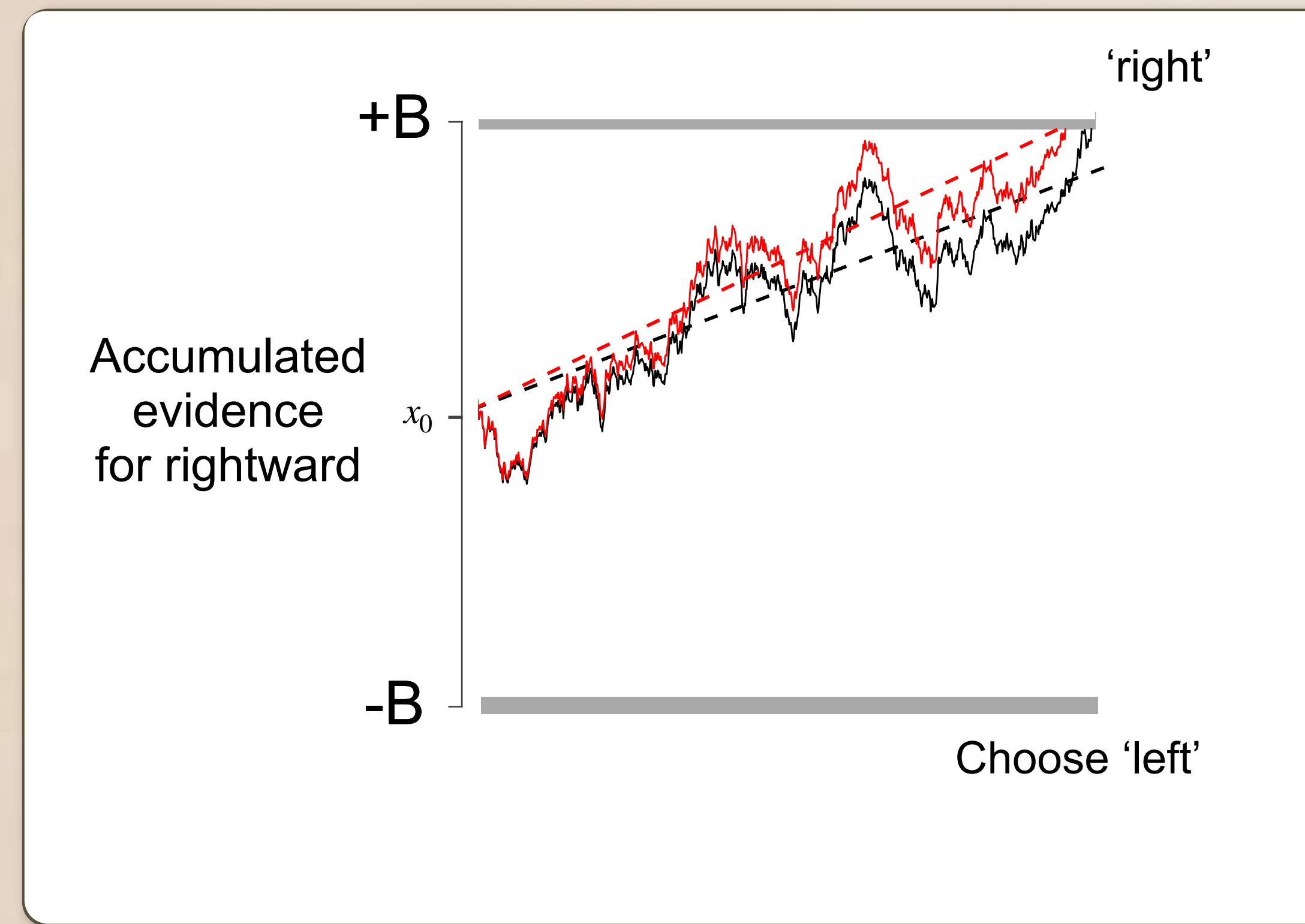
$$x_{i+1} = x_i + e_i$$

decision variable
 $x_0 \neq 0$
evidence

i : time step



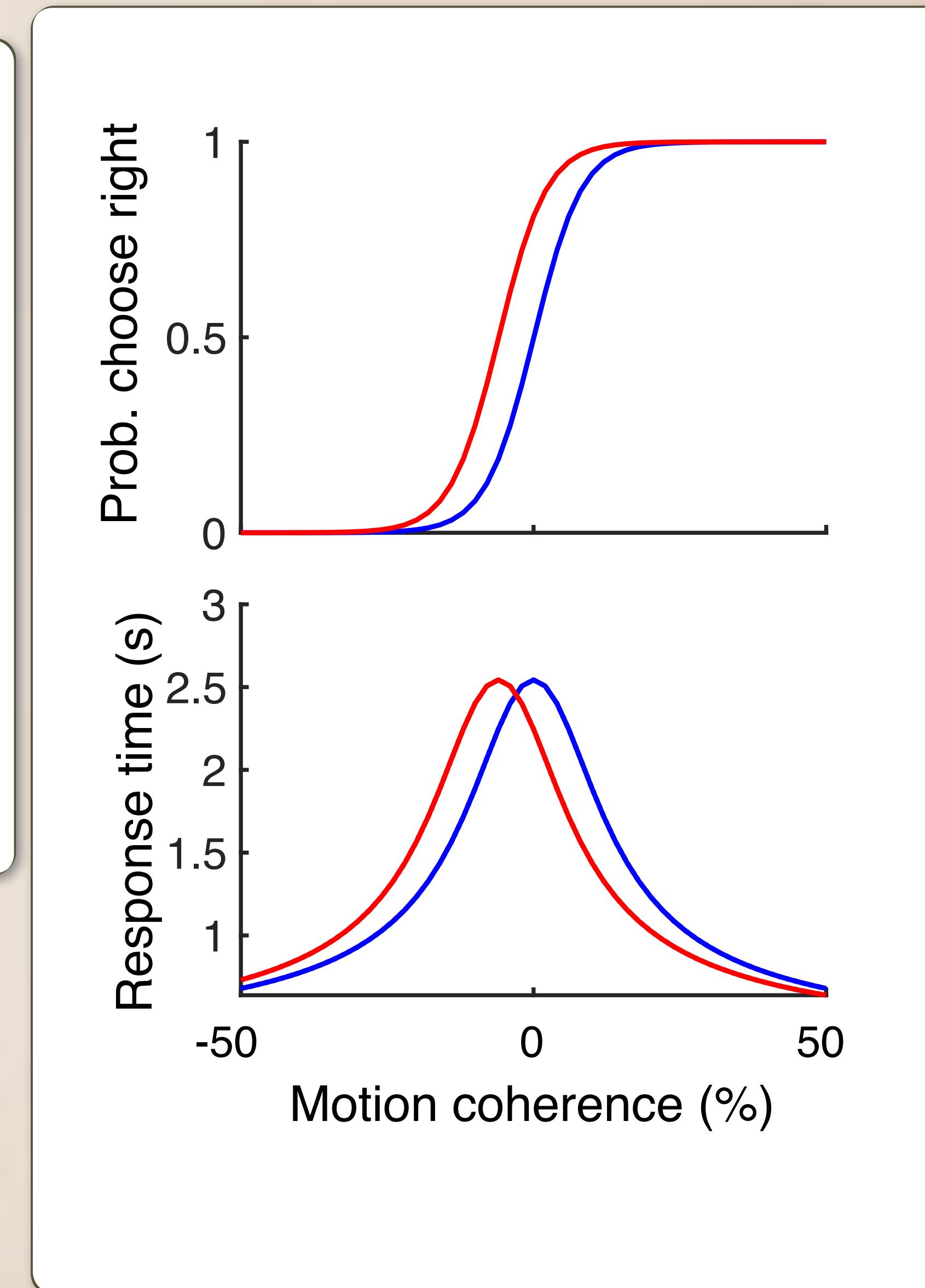
Offset in drift rate



$$e_i = \mathcal{N}(\mu \Delta t + \mu_0 \Delta t, \sigma^2 \Delta t)$$

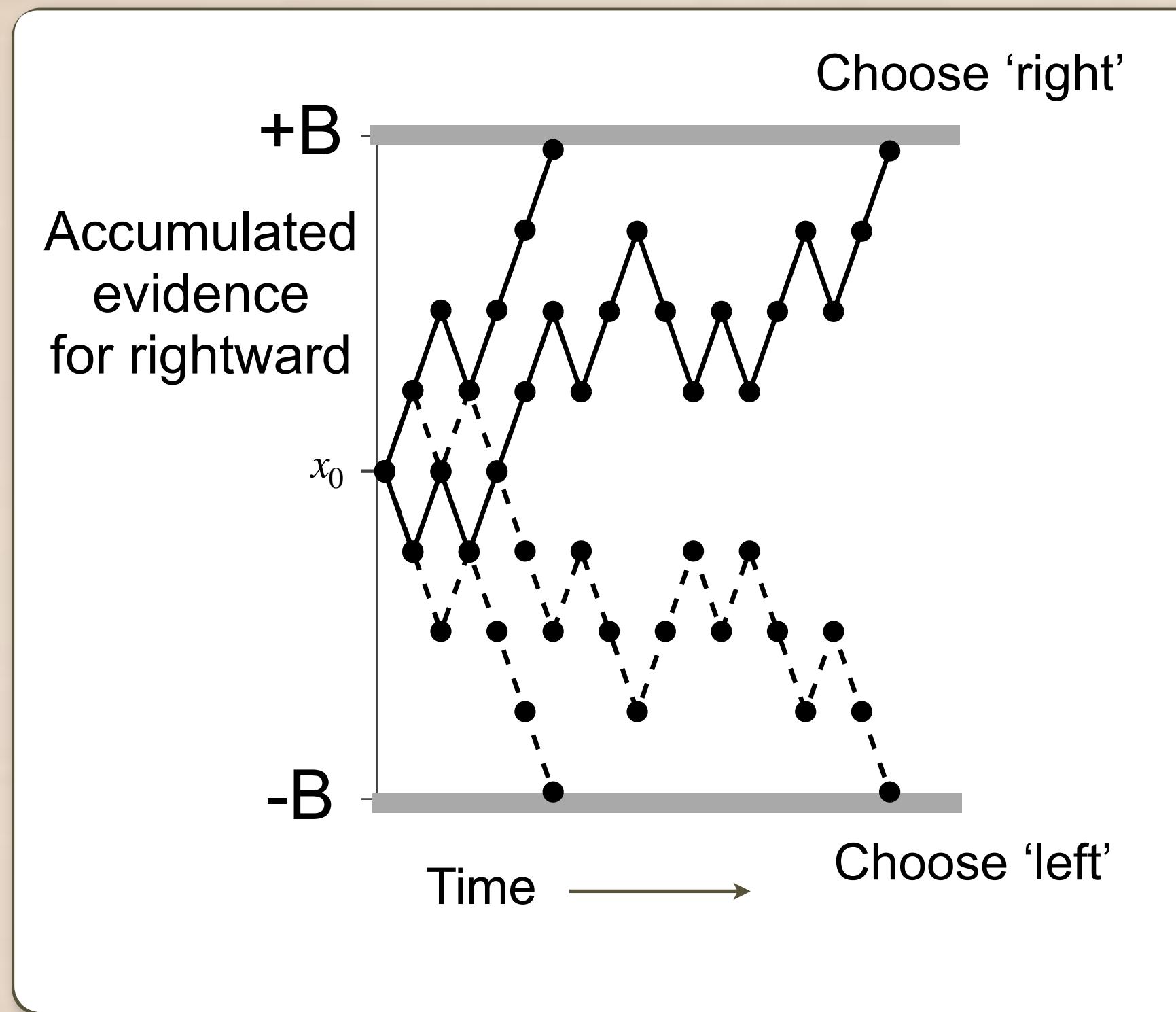
$$x_{i+1} = x_i + e_i$$

$$x_0 = 0$$



Please complete the online questionnaire:
<https://forms.gle/wiGDEZHir5yQjzBa9>

A DDM with ‘flat’ bounds leads to identical RT distributions for correct and incorrect choices



↑ with probability p

↓ with probability $q = 1 - p$

$$\Pr = qp\overline{ppp}q\overline{ppq}\overline{ppp}ppqpp$$

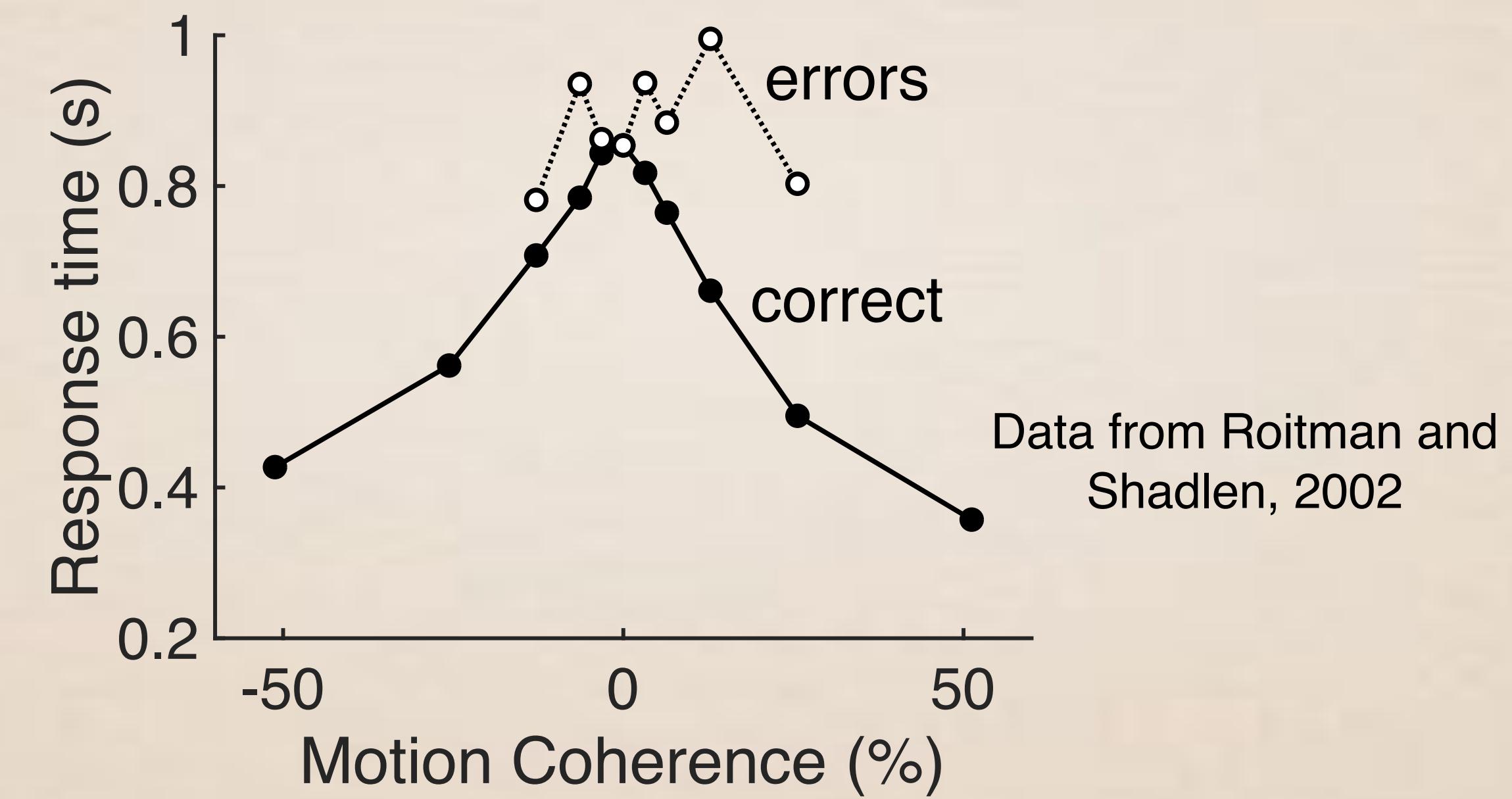
$$\Pr(\text{mirror}) = \overline{q}\overline{p}\overline{ppp}q\overline{q}\overline{ppq}\overline{q}\overline{ppp}q\overline{q}\overline{ppq}q$$

$$\frac{\Pr}{\Pr(\text{mirror})} = \frac{pppp}{qqqq} = \left(\frac{p}{1-p}\right)^4$$

After normalization:

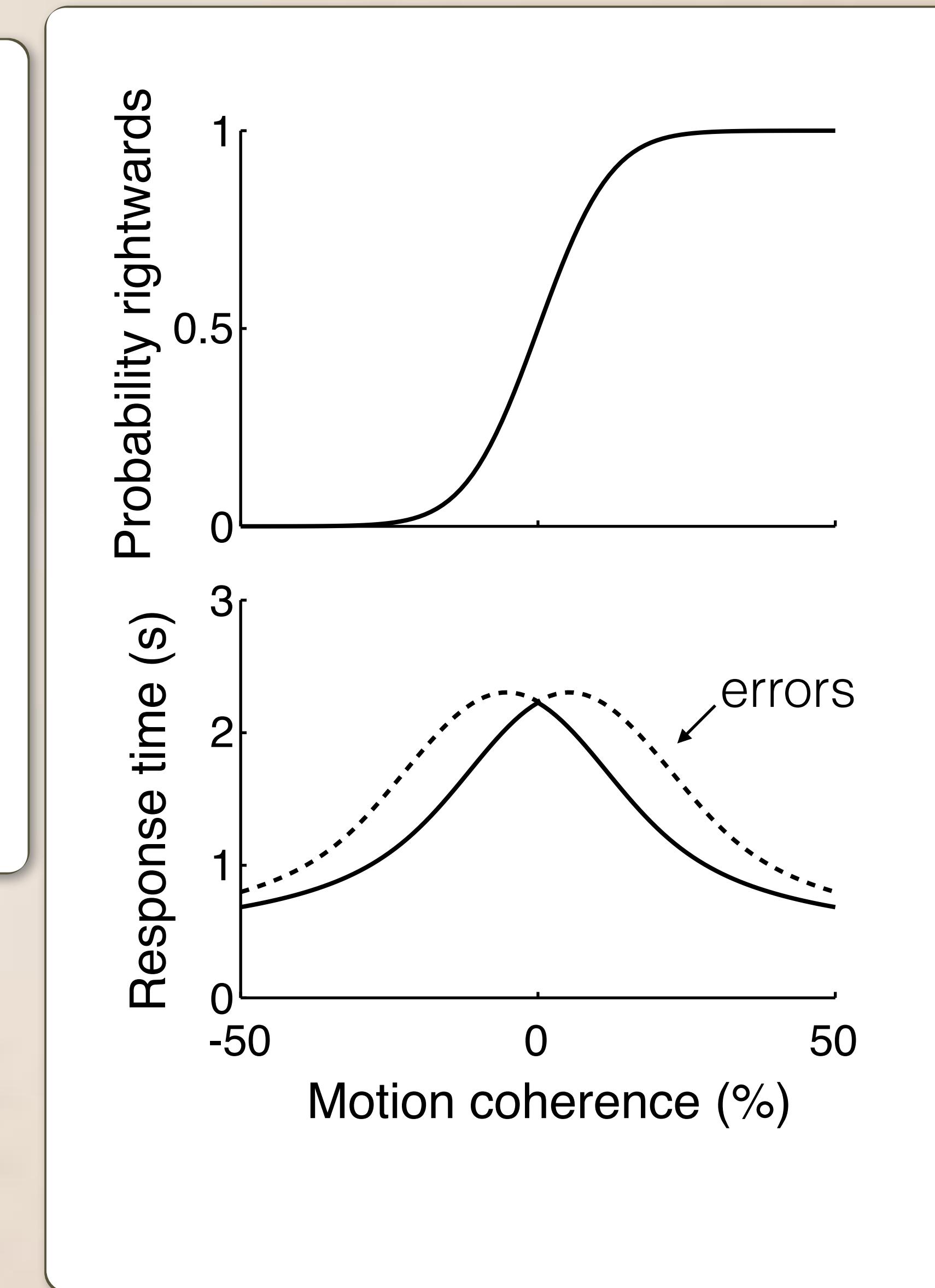
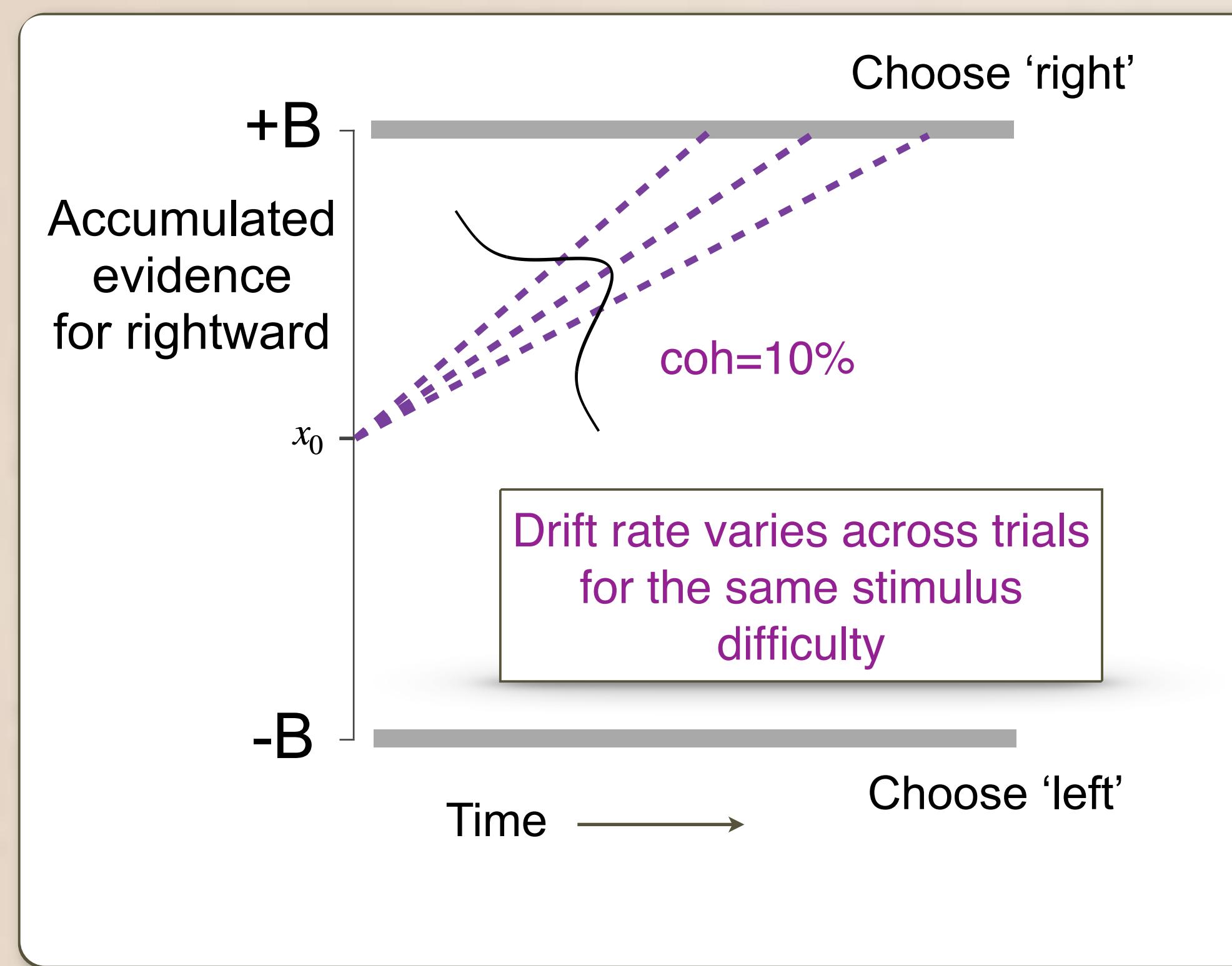
$$P(t \mid \text{right}) = P(t \mid \text{left})$$

Not what is seen in data...

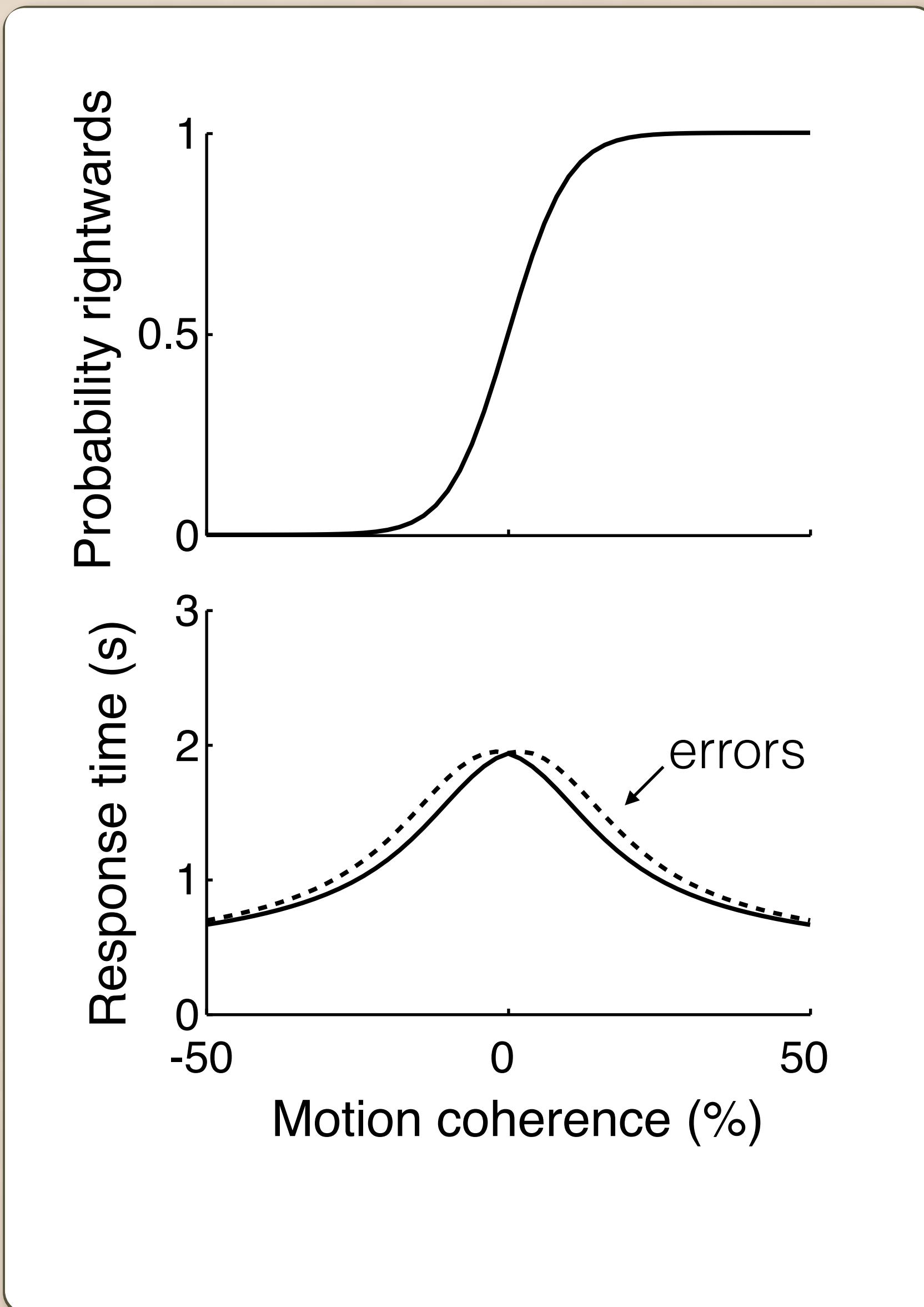
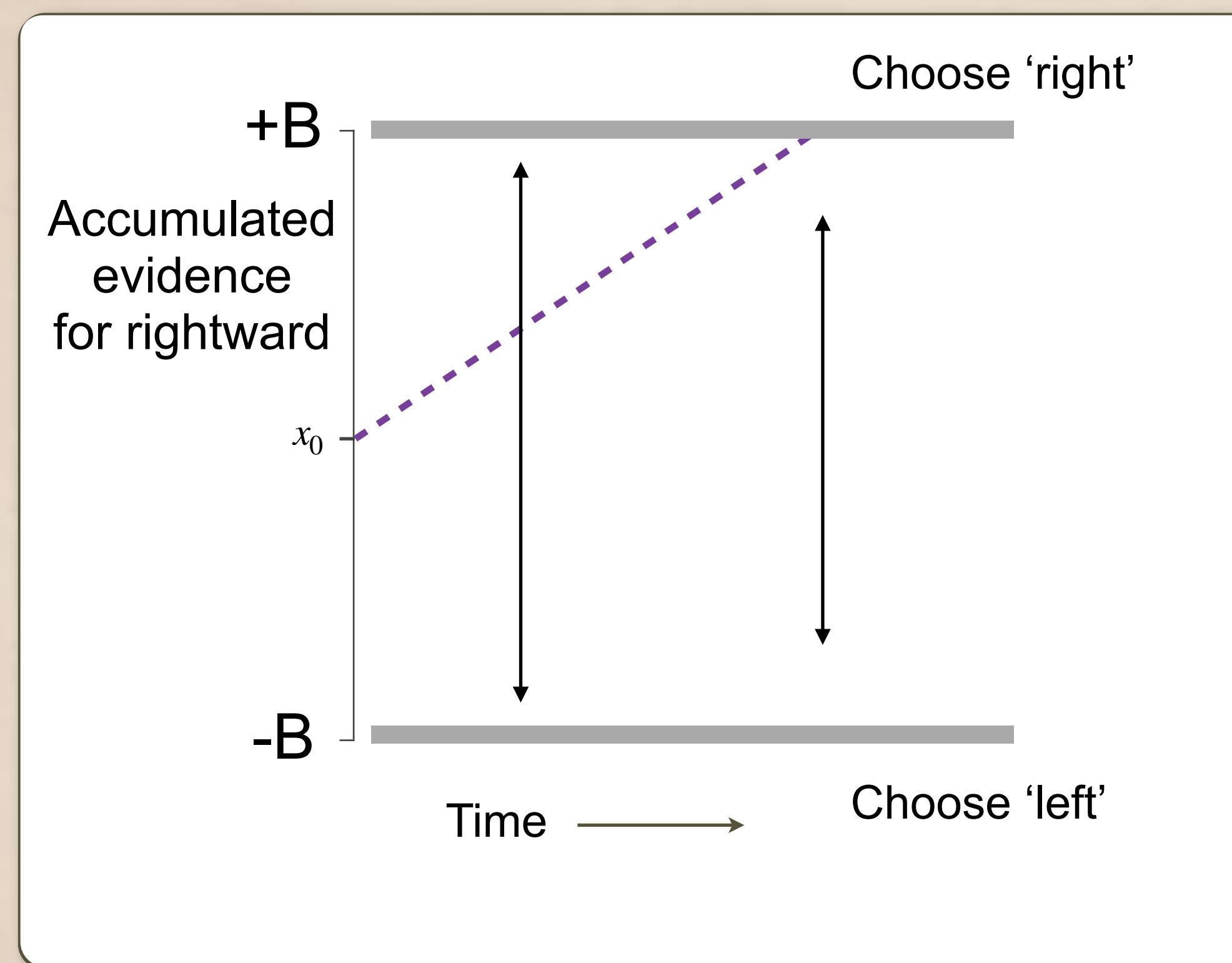


How to “patch” the simple model to make errors slower than correct responses?

Inter-trial variability in drift-rate across trials

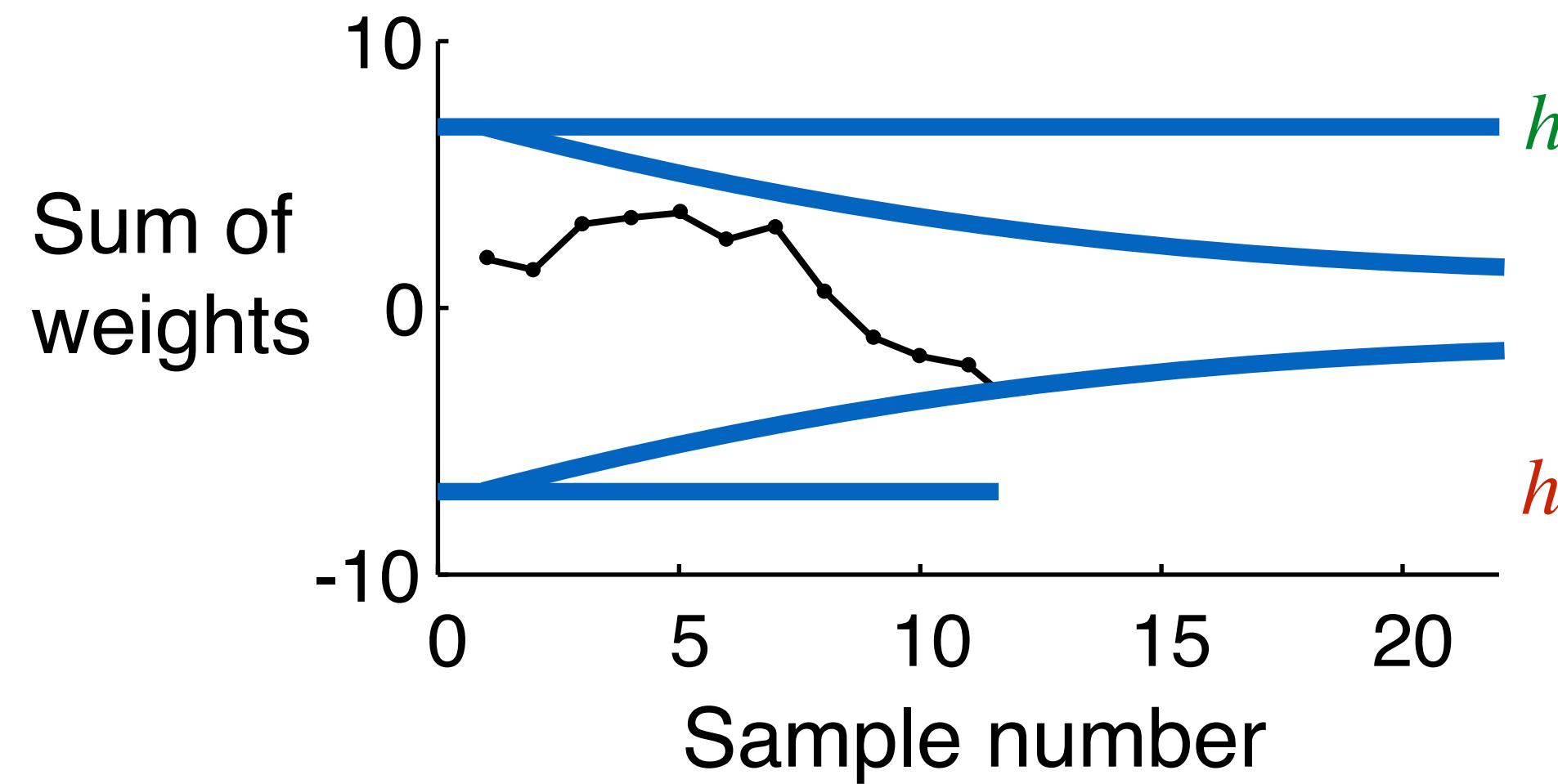
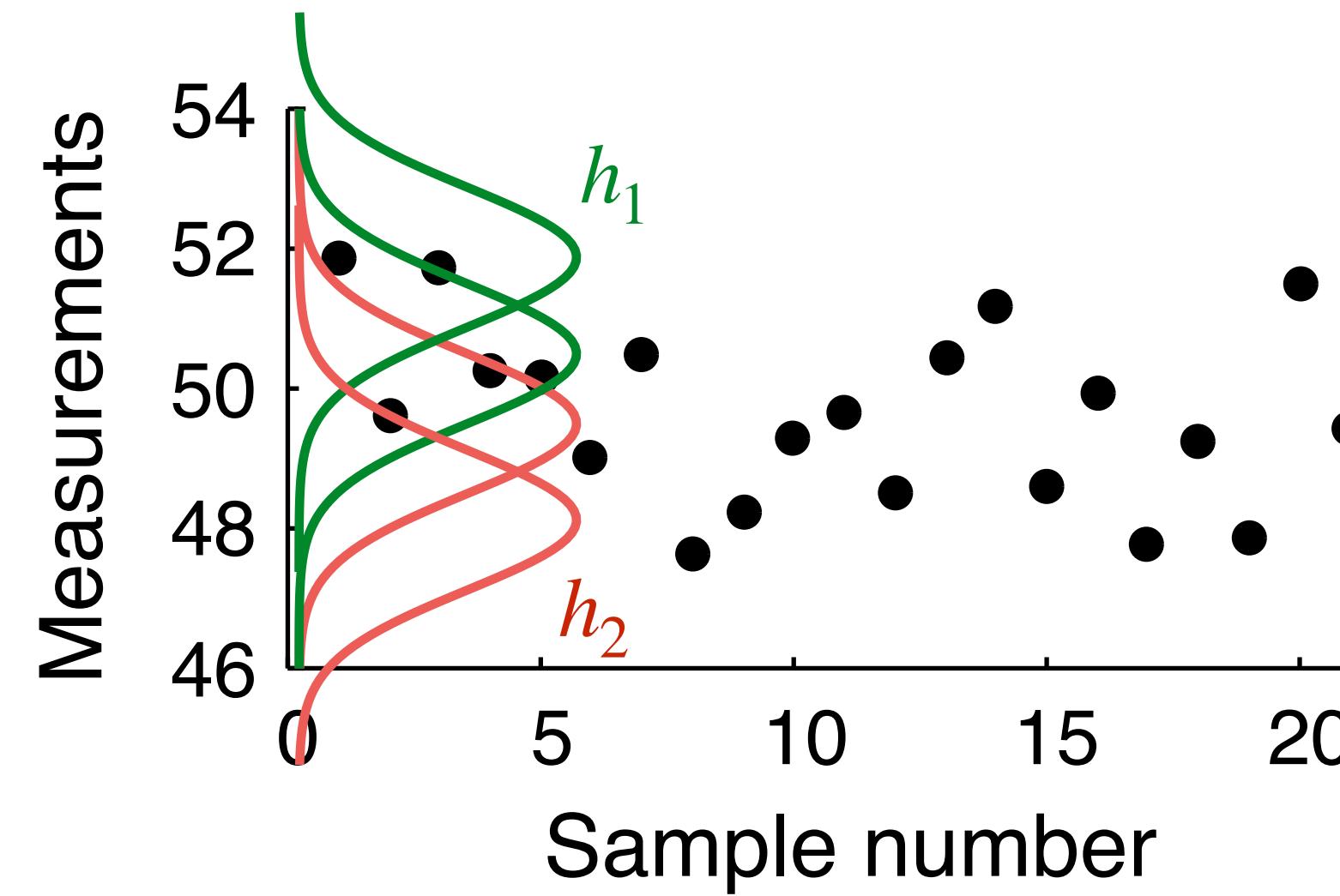


Collapsing bounds



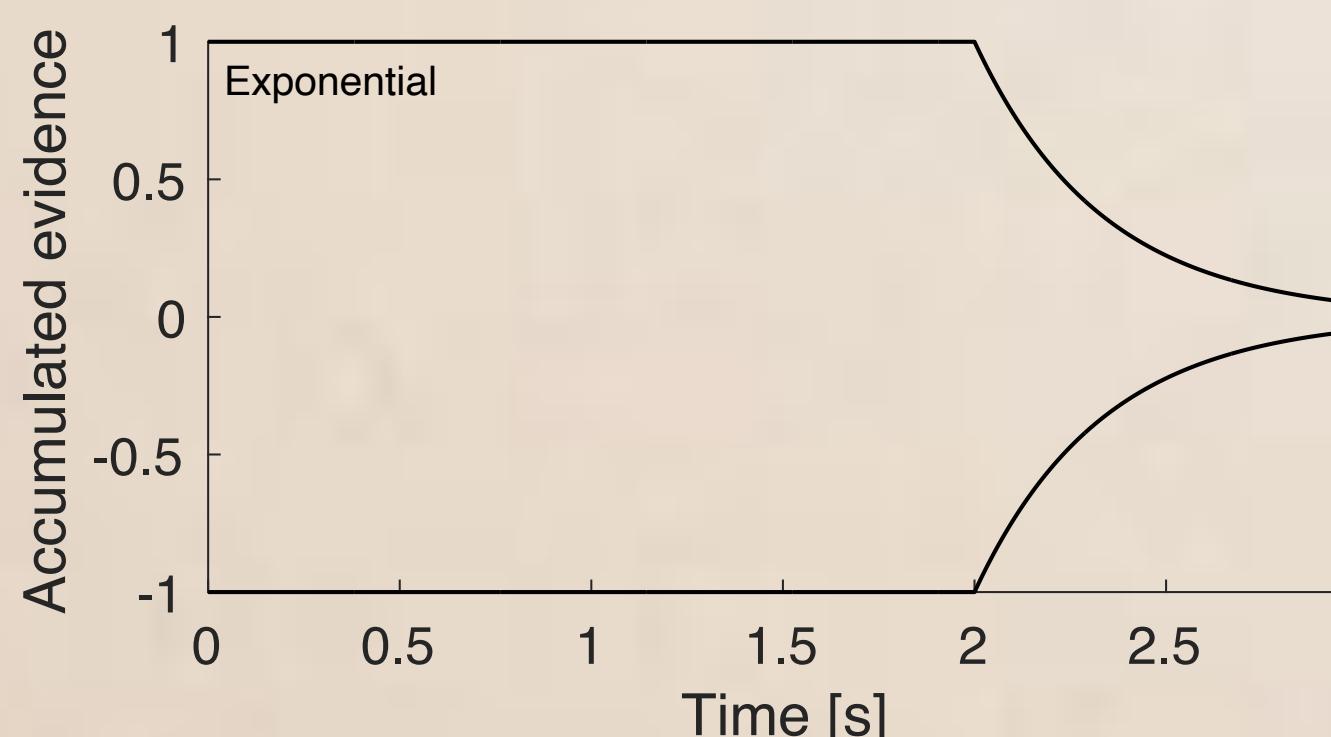
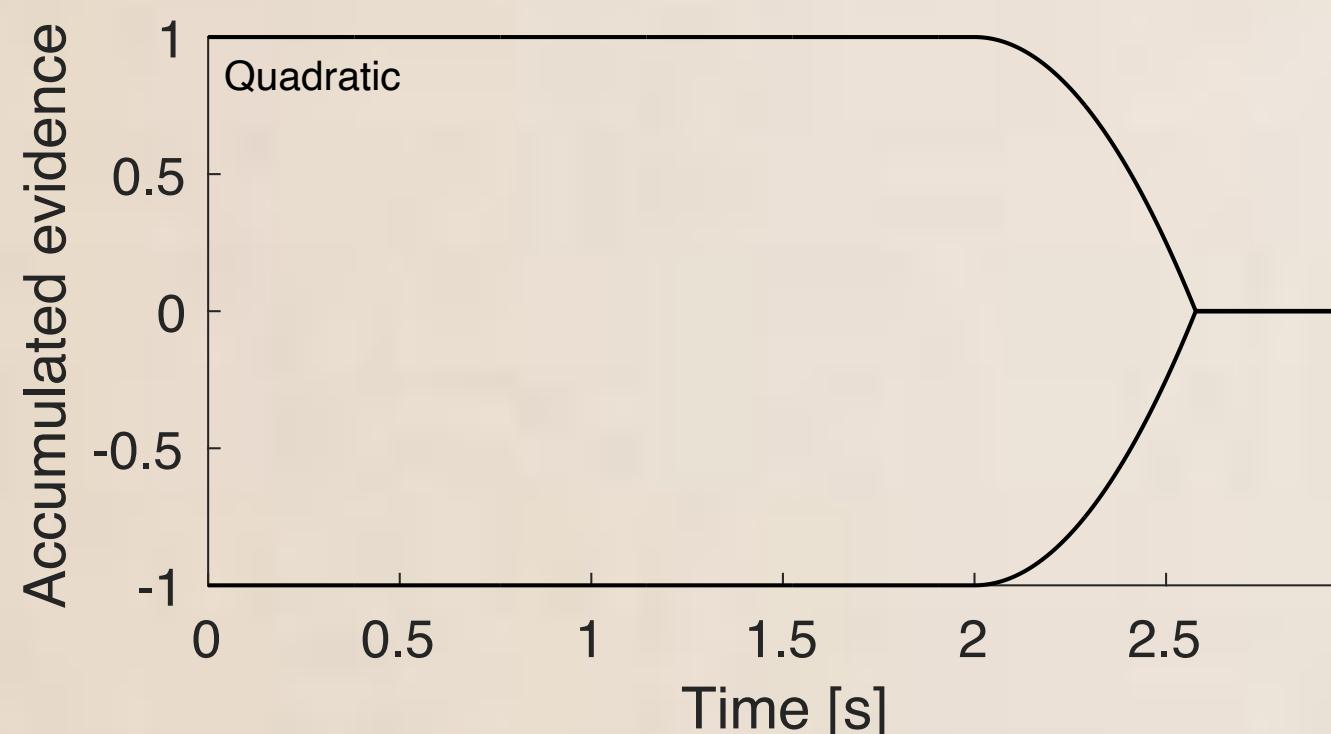
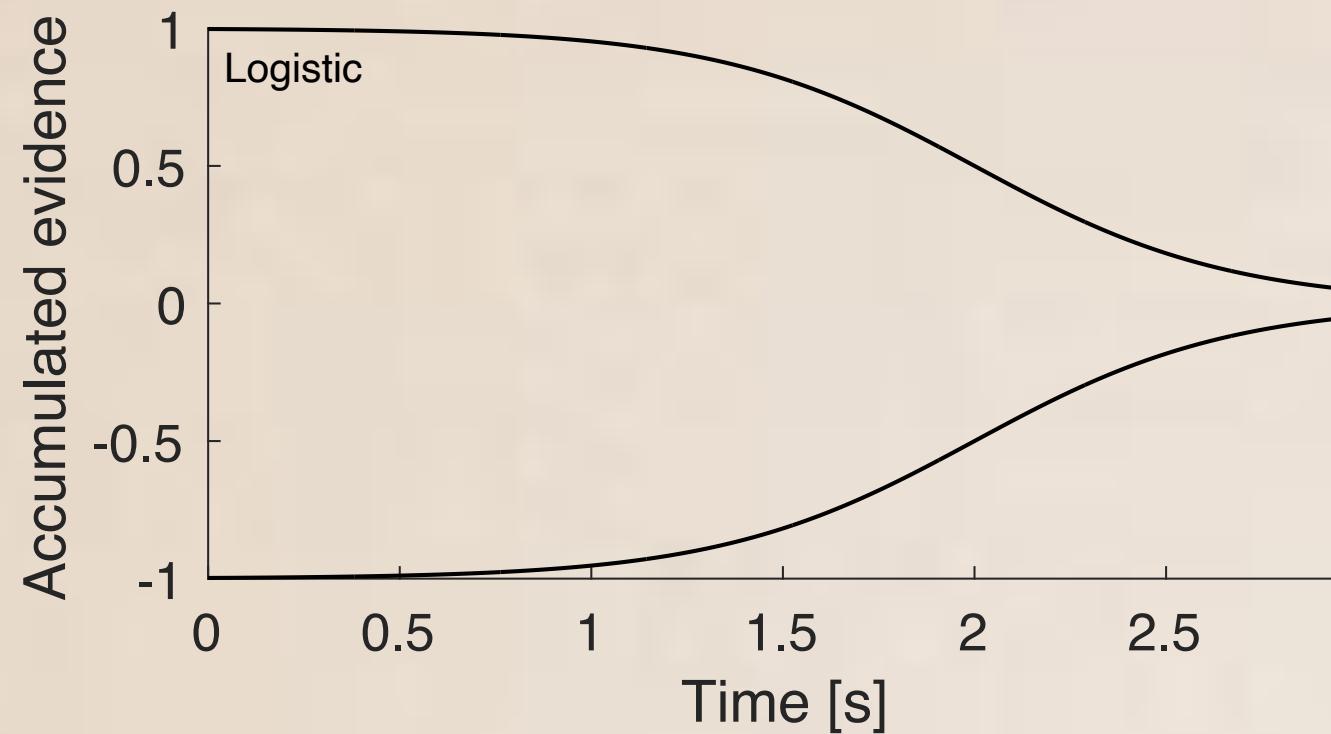
The patch is not really a patch...

Collapsing bounds maximize reward-rate when difficulty varies across trials



Frazier & Yu, NiPS 2008
Drugowitsch et al., JNeuro 2012

Some bound shapes...



$$B(t) = \frac{B_0}{1 + e^{a(t-d)}}$$

$$B(t) = \begin{cases} B_0, & \text{if } t < d \\ B_0 - a(t - d)^2 & \text{otherwise} \end{cases}$$

$$B(t) = \begin{cases} B_0, & \text{if } t < d \\ B_0 e^{-a*(t-d)}, & \text{otherwise} \end{cases}$$

See `expand_bounds.m`

DATA FITTING

The DDM has free parameters... we usually want to fit them to data

» Why?

- We want to know if the DDM can underlie the pattern of choices and RTs
- We want to compare the DDM to alternative models
- Hypothesis are often formulated in terms of specific DDM parameters

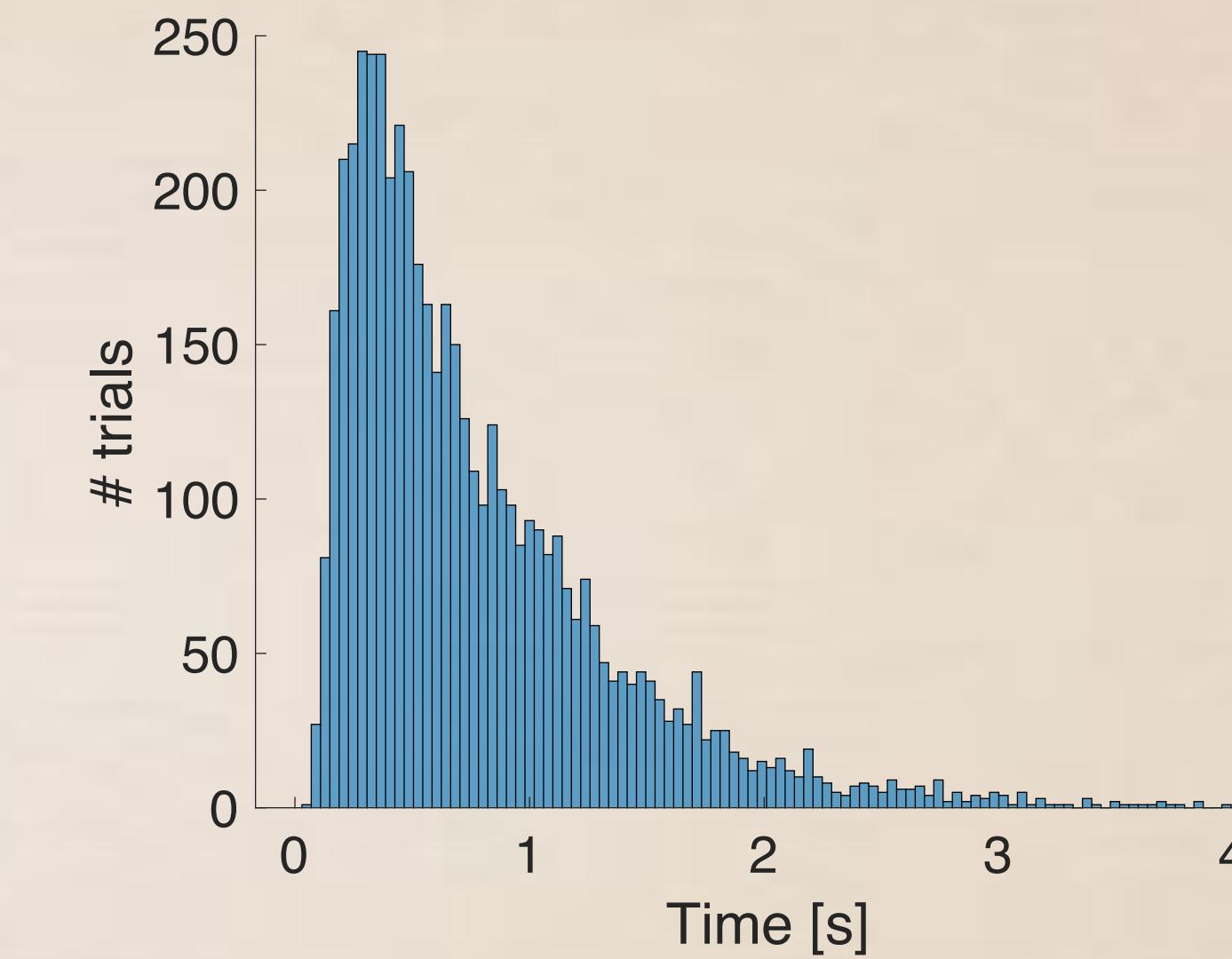
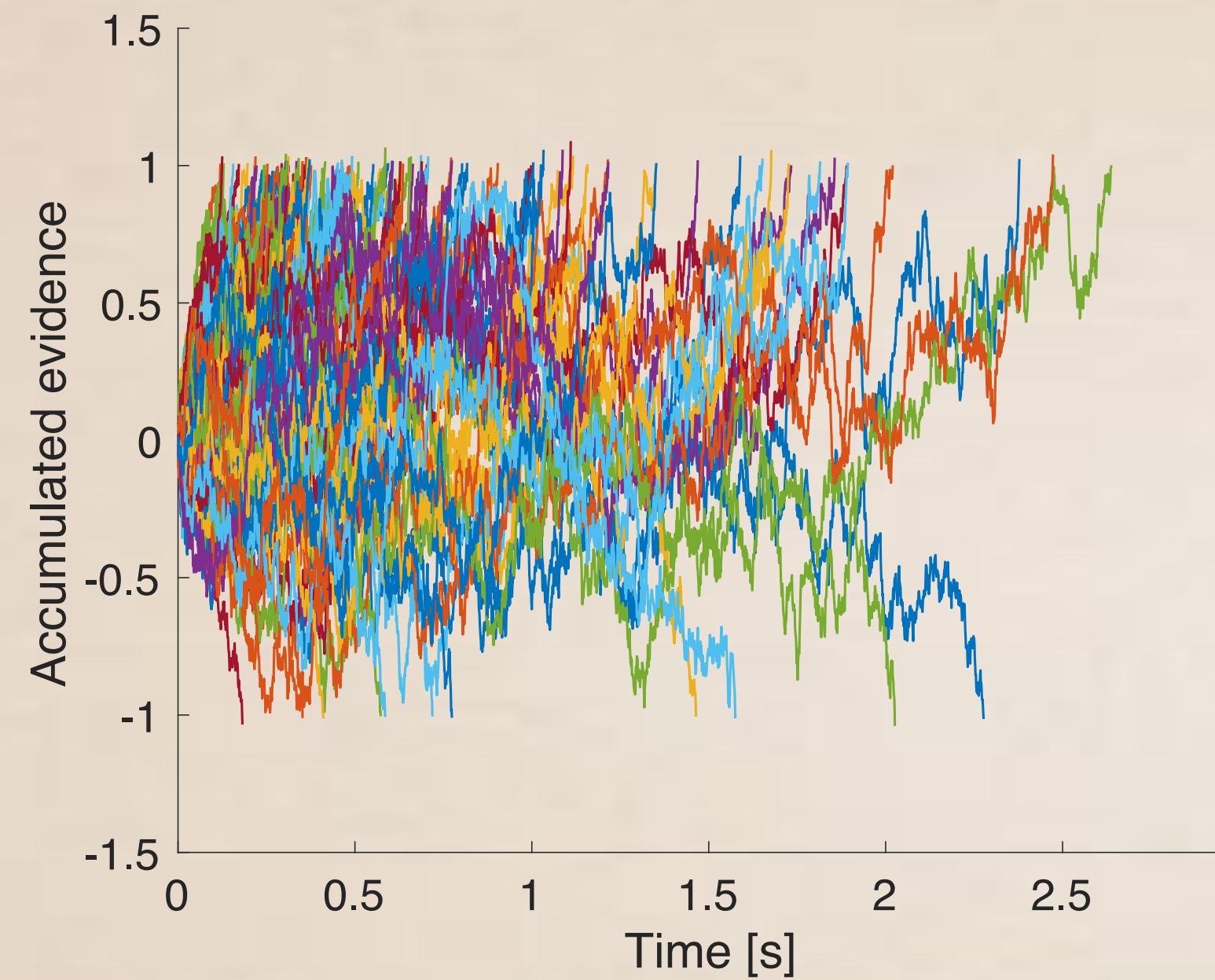
Towards data fitting...

1. Obtain the joint probability of choice & RT given a set of model parameters, $p(\text{choice}, \text{RT} \mid \text{model parameters})$
2. Decide what to minimize/maximize
3. Choose a fitting algorithm to search over the parameter space

1. How to obtain $p(\text{choice}, RT \mid \text{model parameters})$

- » **Simulate multiple diffusion paths,**

Simulate multiple diffusion paths



- *Easy to perform*
- *Results are stochastic*
- *Fitting can be slow*
- *May incorrectly estimate the likelihood of rare events*

1. How to obtain $p(\text{choice}, RT \mid \text{model parameters})$

- » Simulate multiple diffusion paths,
- » **Analytical solutions to the choice and response time functions,**

Example: analytical solution to the expectation of choice and response time

We saw this slide already...

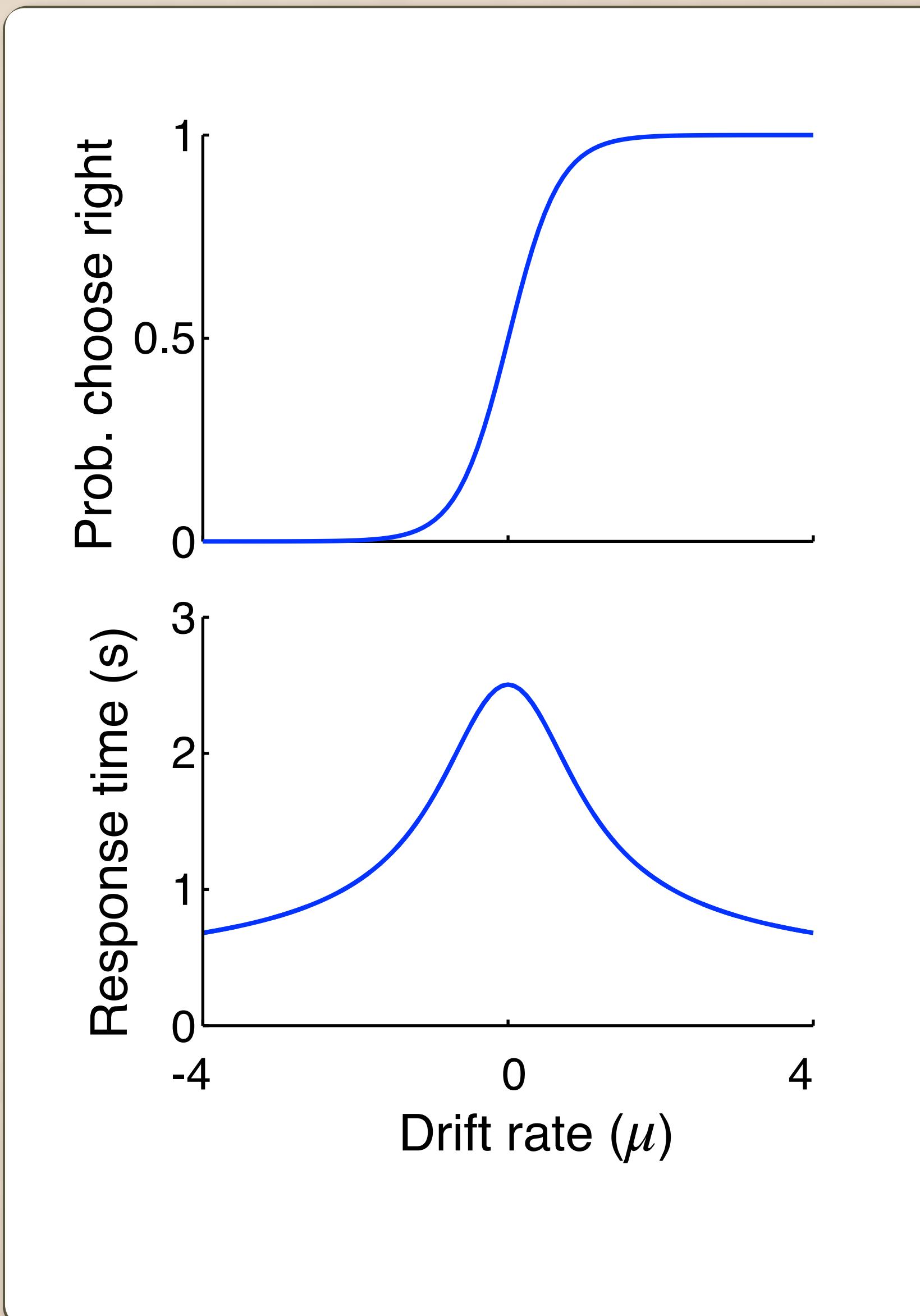
Probability of terminating at the upper bound:

$$p_+ = \frac{1}{1 + e^{-2\mu B}}$$

Average response time:

$$T = \frac{B}{\mu} \tanh(\mu B) + t_{nd}$$

When $\mu = 0$: $T = B^2 + t_{nd}$



For ‘flat’ bounds, there are analytical solutions to the choice and decision time distributions

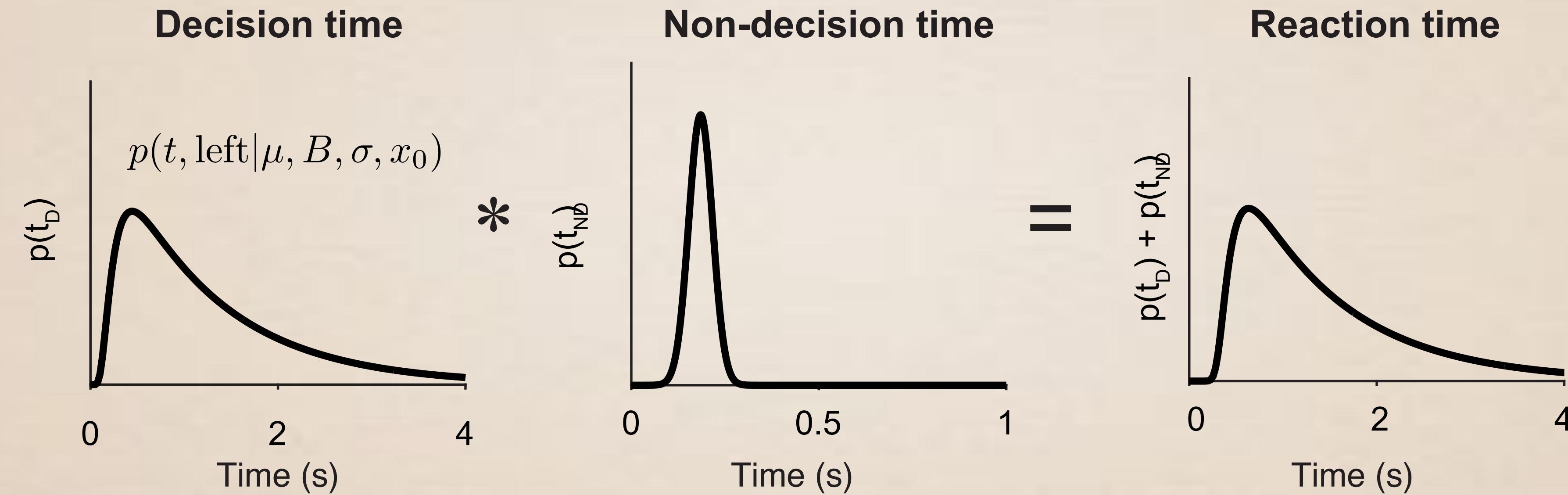
Probability of terminating at the lower bound at time t :

$$p(t, \text{left} | \mu, B, \sigma, x_0) = p_-(\mu, B, \sigma, x_0) - \frac{\pi\sigma^2}{4B^2} e^{-\frac{(B+x_0)\mu}{\sigma^2}} \times \sum_{k=1}^{\infty} \frac{2k \sin\left(\frac{k\pi(B+x_0)}{2B}\right) e^{-\frac{1}{2}\left(\frac{\mu^2}{\sigma^2} + \frac{\pi^2 k^2 \sigma^2}{4B^2}\right)t}}{\left(\frac{\mu^2}{\sigma^2} + \frac{\pi^2 k^2 \sigma^2}{4B^2}\right)}$$

, where

$$p_-(\mu, B, \sigma, x_0) = \frac{e^{-(4\mu B/\sigma^2)} - e^{-(2\mu(B+x_0)/\sigma^2)}}{e^{-(4\mu B/\sigma^2)} - 1}$$

From decision time to response time



1. How to obtain $p(\text{choice}, RT \mid \text{model parameters})$

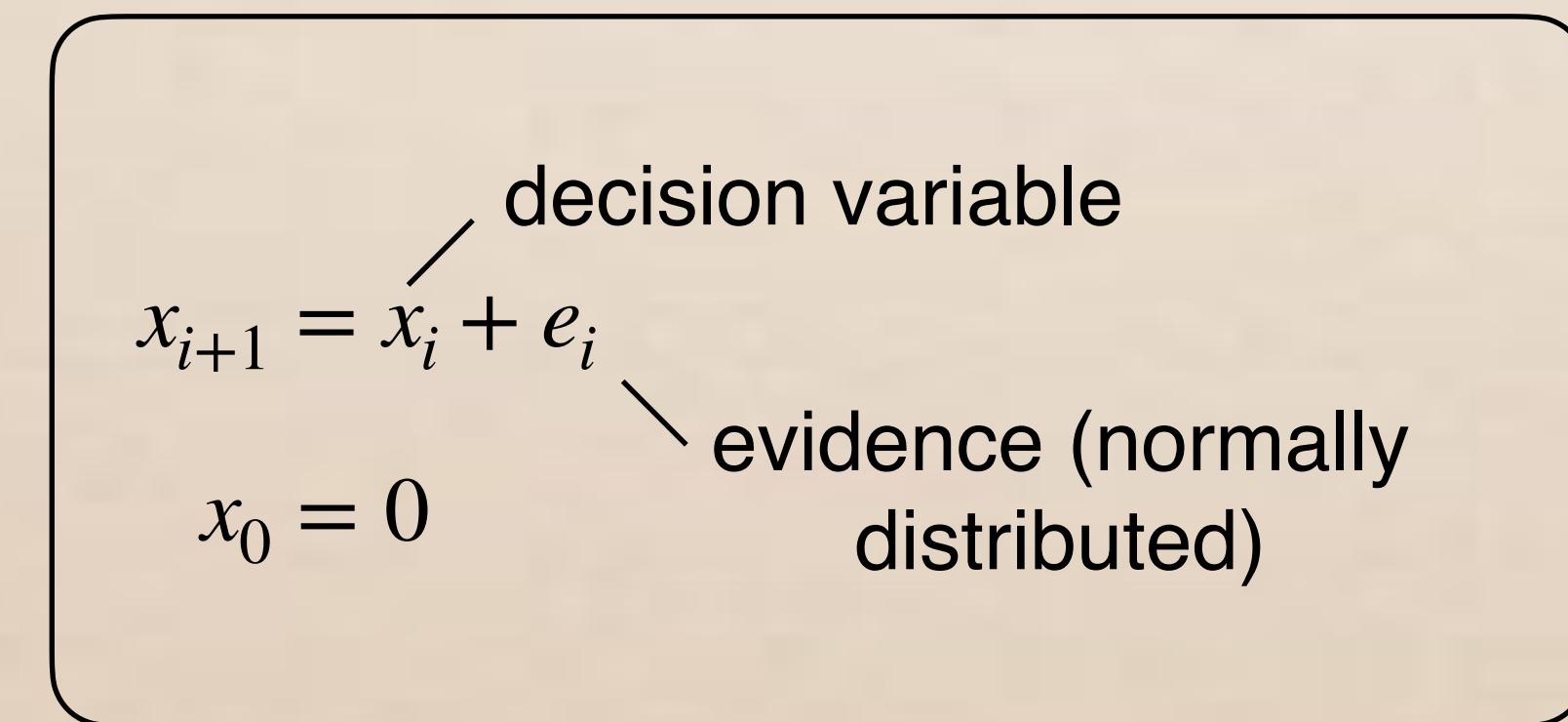
- » Simulate multiple diffusion paths,
- » Analytical solutions to the choice and response time functions,
- » **Numerical solutions (propagating over time the probability distribution for the state of the decision variable)**

Fokker-Planck equation

Partial differential equation that describes the time evolution of the probability density function of the decision variable x :

$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial^2 x}$$

- Think about prob. distributions instead of single particles
- The eq. relates the derivative over time (t) to that over accumulated evidence (x)


$$x_{i+1} = x_i + e_i$$

decision variable

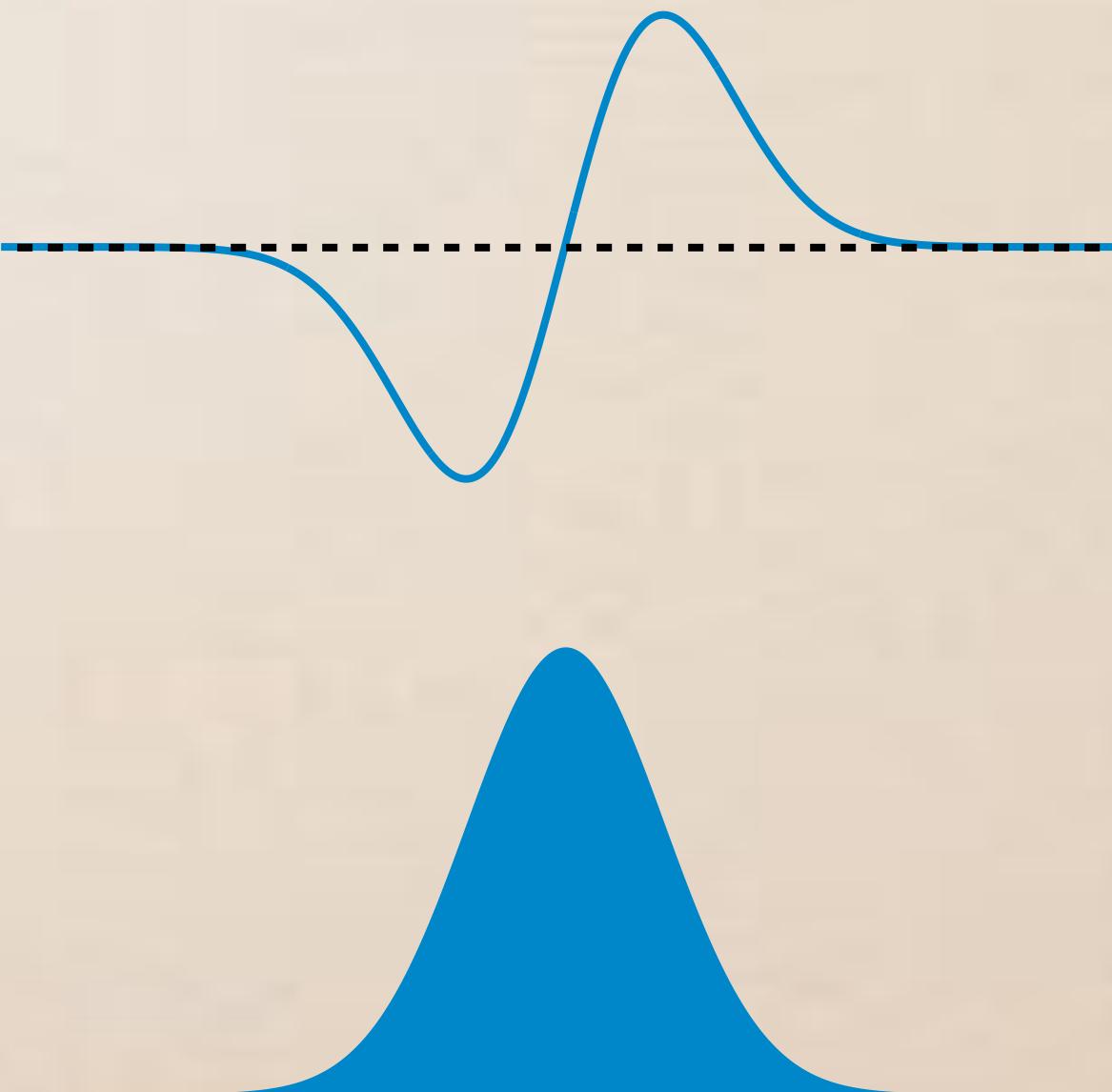
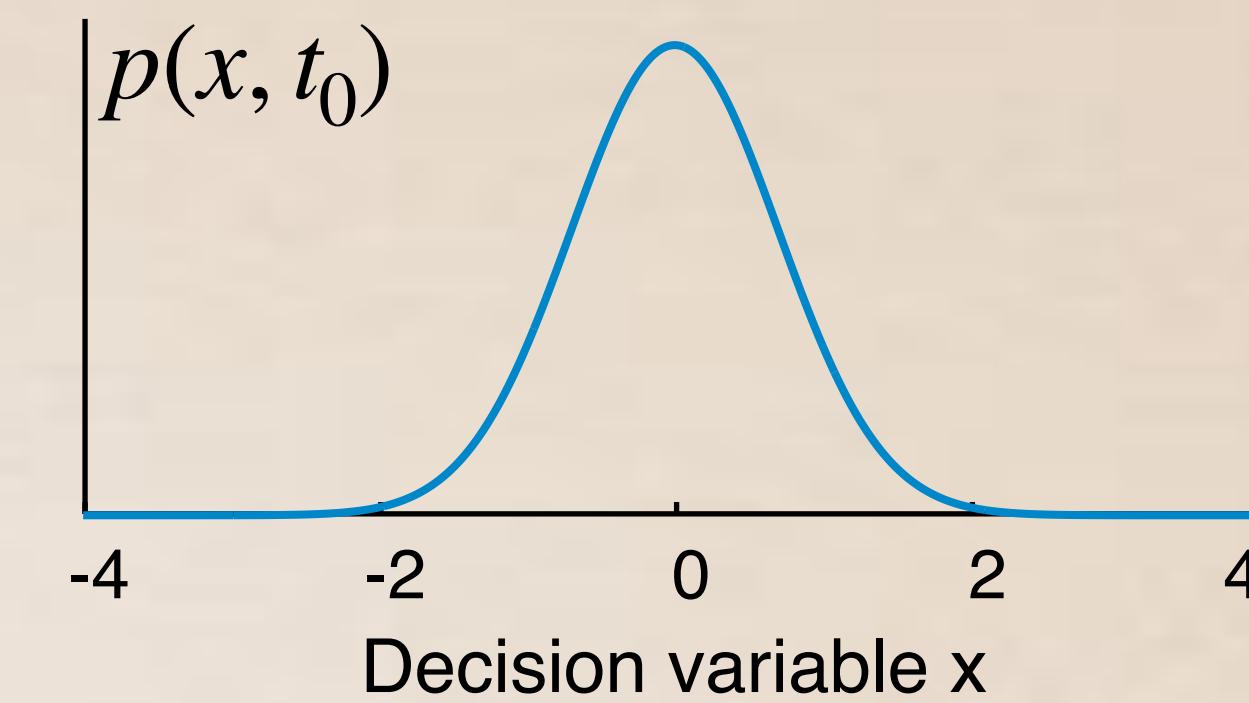
$$x_0 = 0$$

evidence (normally distributed)

Describe the
same process?

Fokker-Planck intuition

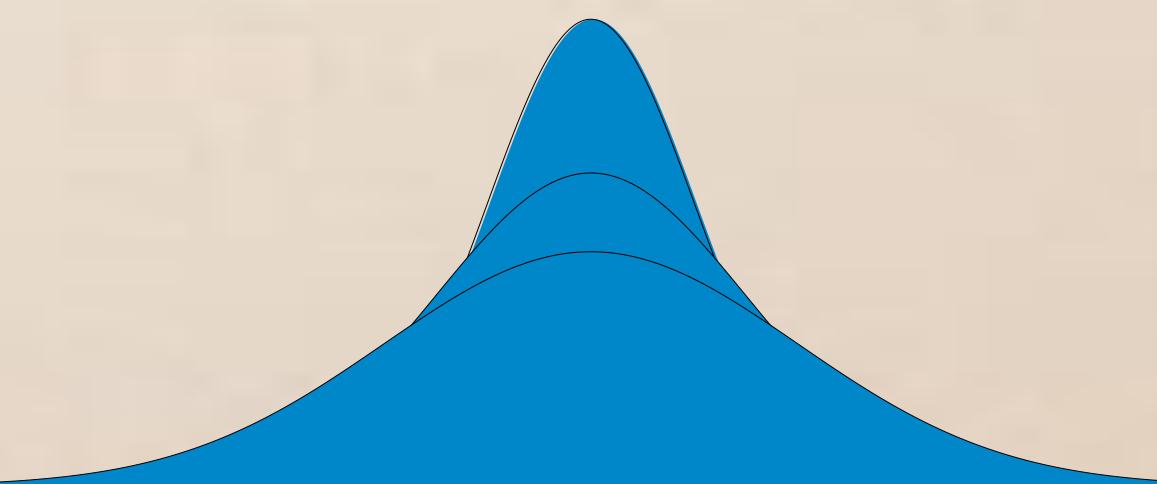
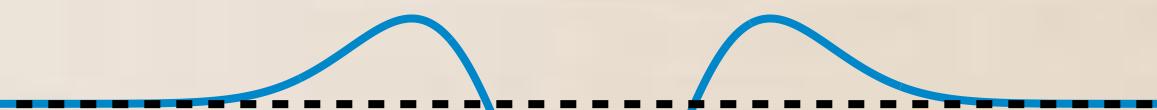
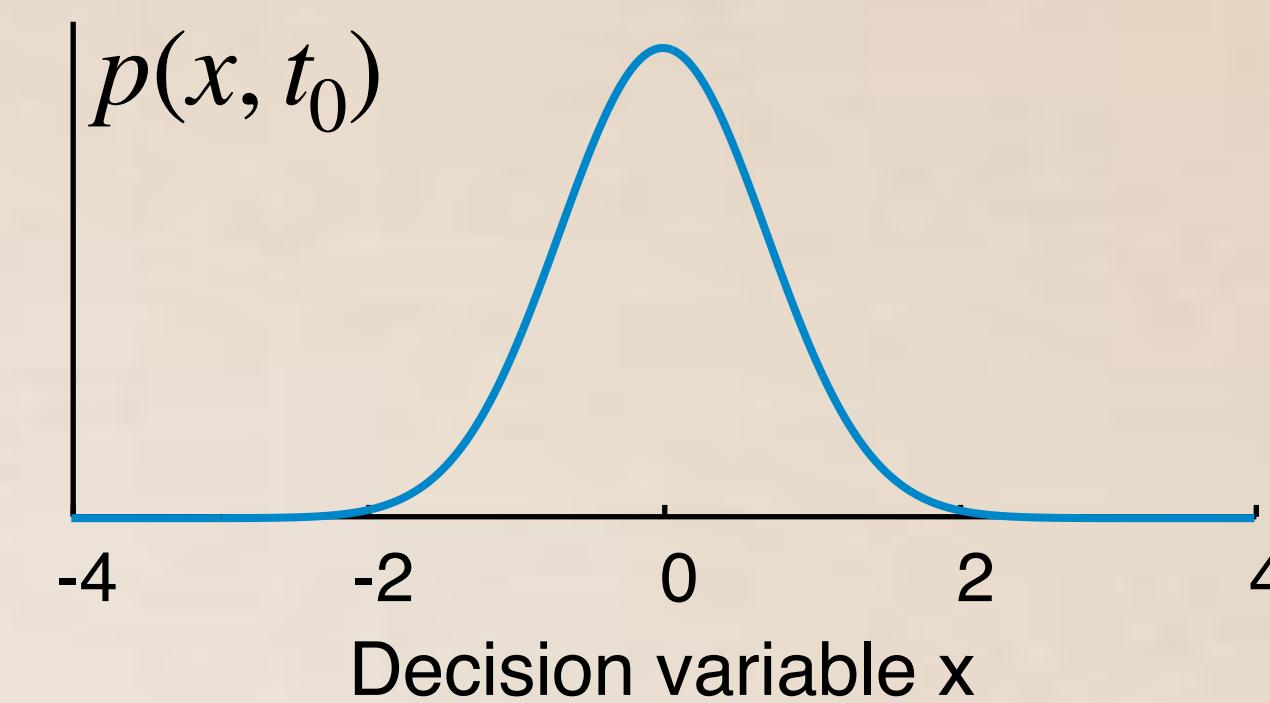
$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial x^2}$$



Fokker-Planck intuition

Fokker Planck equation:

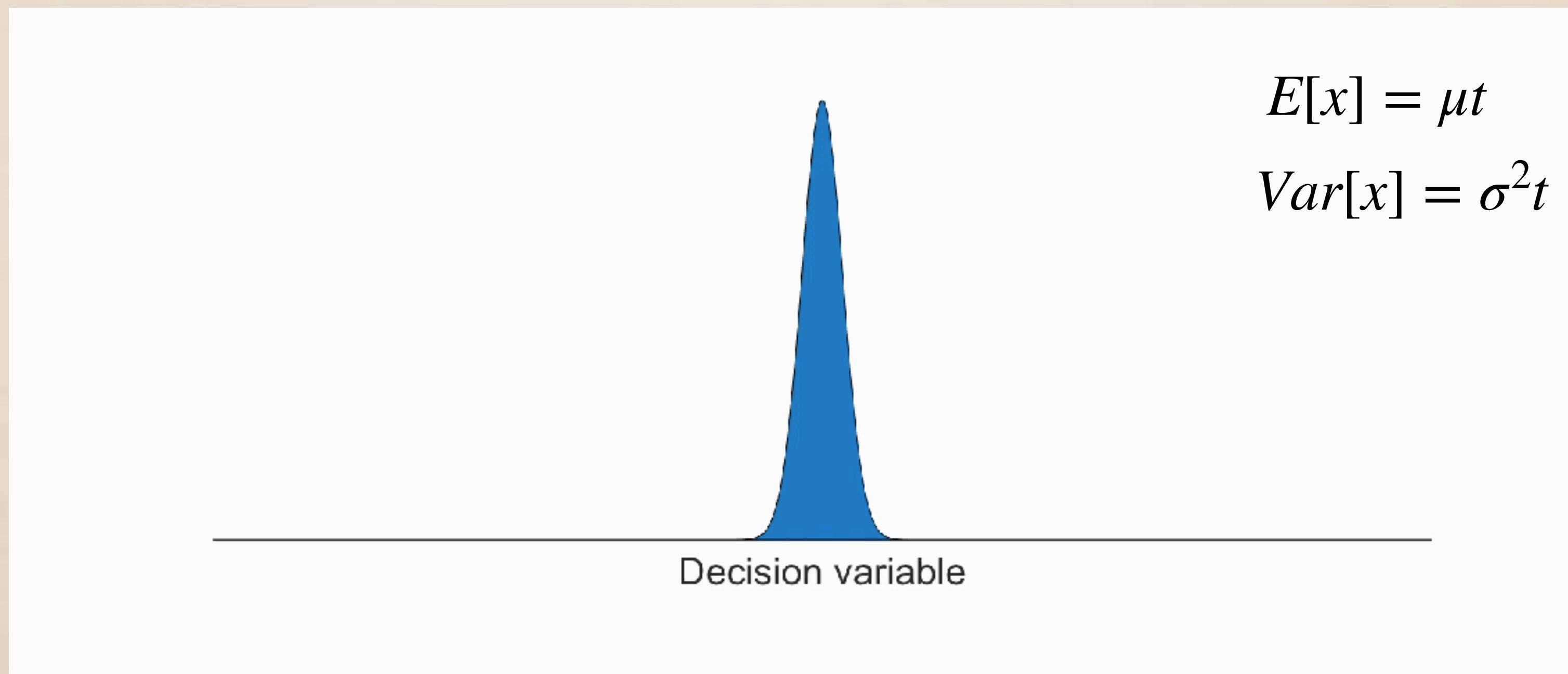
$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial^2 x}$$



The two terms together

Fokker Planck equation:

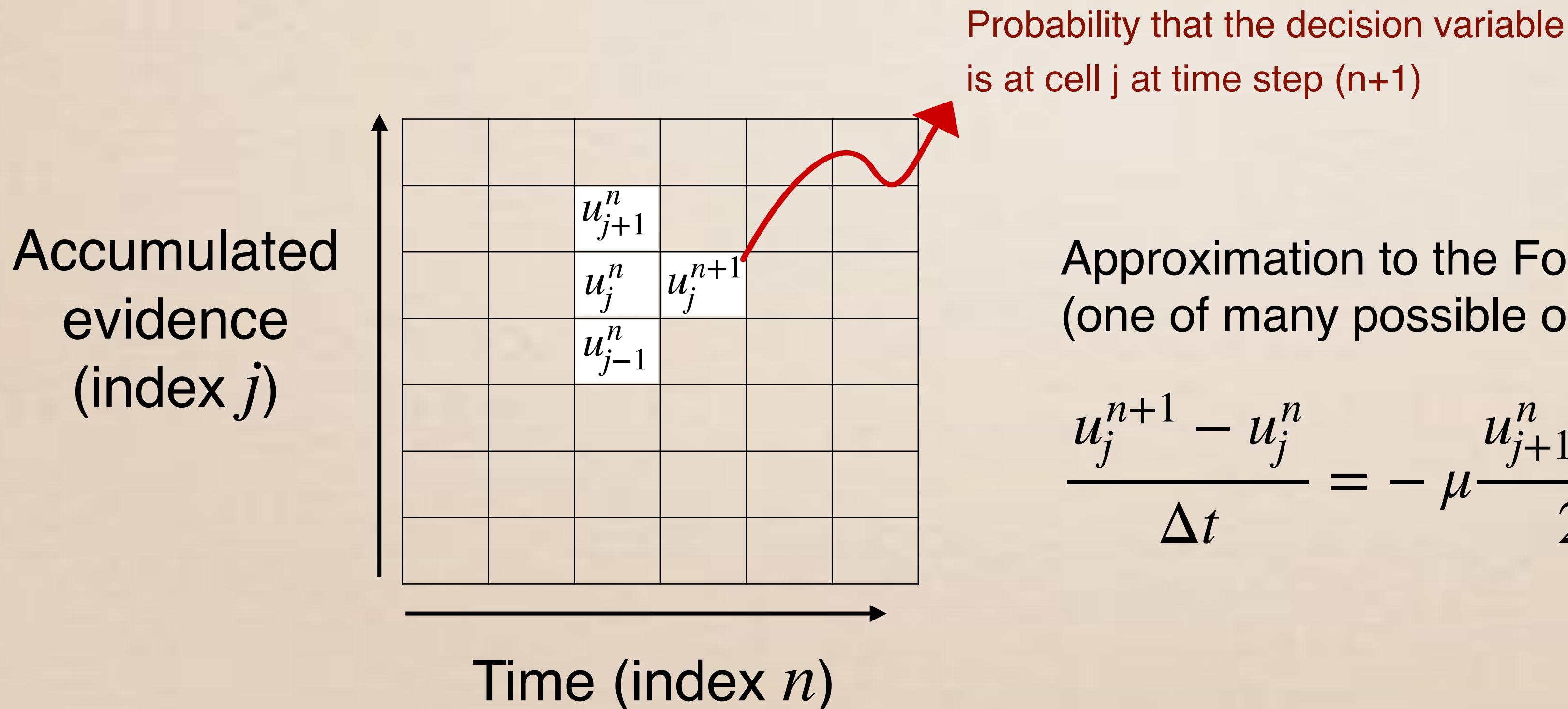
$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial^2 x}$$



Numerical approximations to the FP equation

Finite difference methods: approximate a differential equation by a difference equation

$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial^2 x}$$



Numerical approximations to the FP equation

$$\begin{pmatrix} U^{n+1} \end{pmatrix} = \begin{pmatrix} \text{TRI-DIAGONAL} \\ \text{MATRIX} \end{pmatrix} \begin{pmatrix} U^n \end{pmatrix} \quad , \text{ where } U^n = \begin{pmatrix} u_1^n \\ u_2^n \\ \vdots \\ u_j^n \\ \vdots \end{pmatrix}$$

Can be implemented with a few lines of Matlab code:

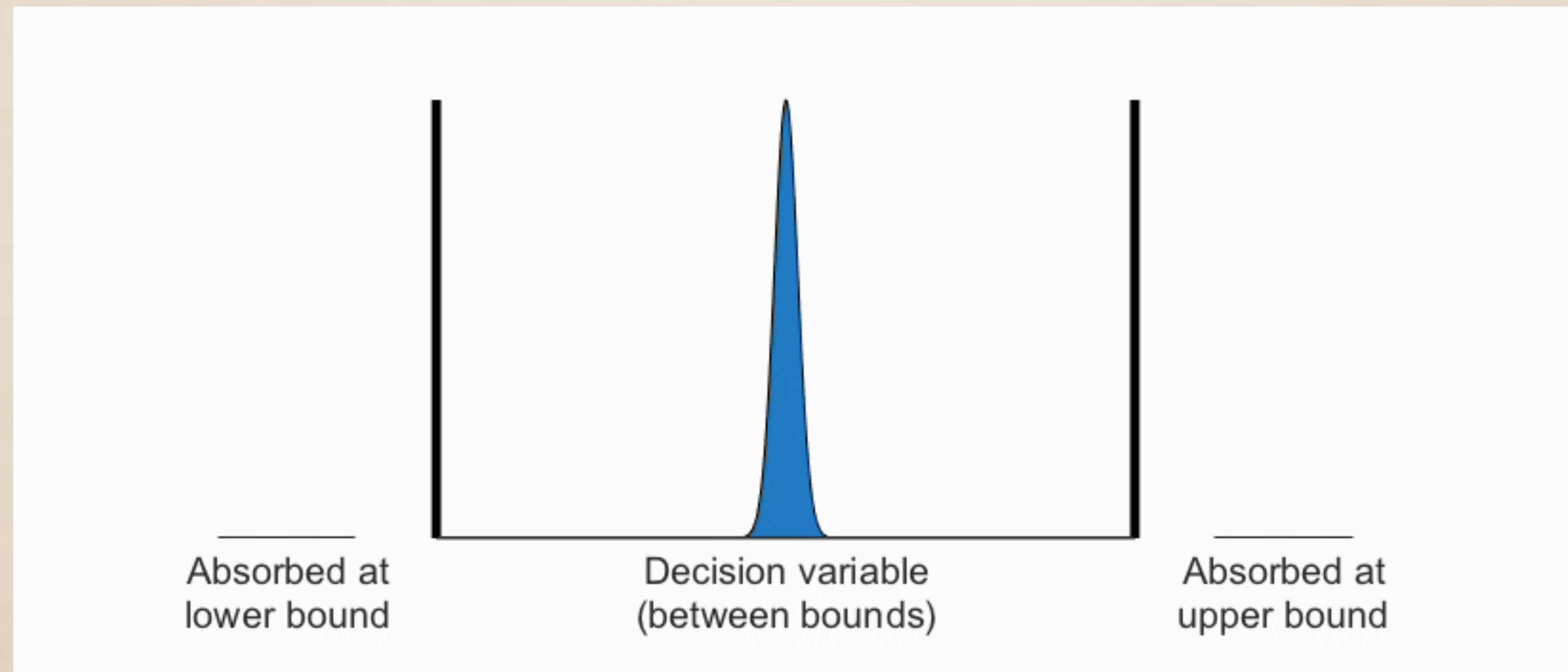
```
1 % prepare tri-diagonal matrix
2 temp = repmat([(-a/(2*dx)+D/(dx^2)), -2*D/(dx^2), (a/(2*dx)+D/dx^2)], nx, 1);
3 A = spdiags(temp, -1:1, nx, nx);
4
5 for i = 2:nt % iterate over time
6     u = dt*A*u + u; % update u
7     U(:, i) = u; % save
8 end
```

Probability distribution for
the state of the decision
variable at time step n

With bounds

Fokker Planck equation:

$$\frac{\partial p(x, t)}{\partial t} = -\mu \frac{\partial p(x, t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x, t)}{\partial^2 x}$$



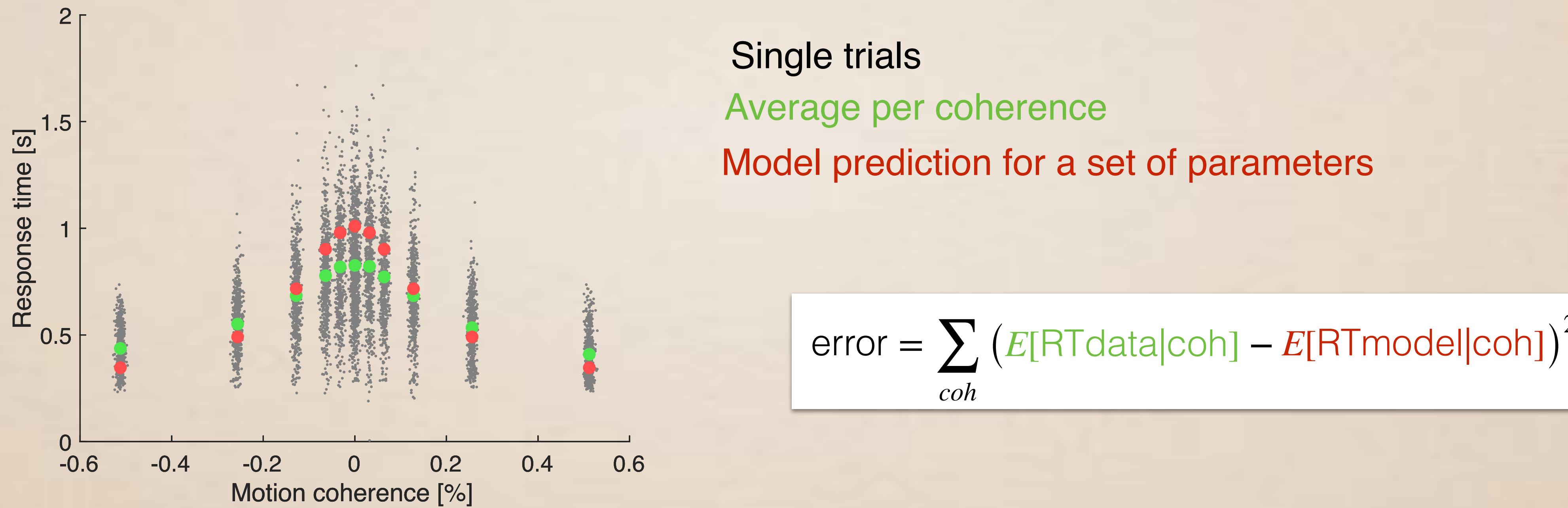
So... we calculate or approximate $p(\text{choice}, RT \mid \text{model parameters})$ either through simulations, analytical or numerical solutions

2. What to minimize/maximize when fitting parameters?

One (simple but suboptimal) option:
Minimize the sum of squared errors of mean RTs

2. What to minimize/maximize when fitting parameters?

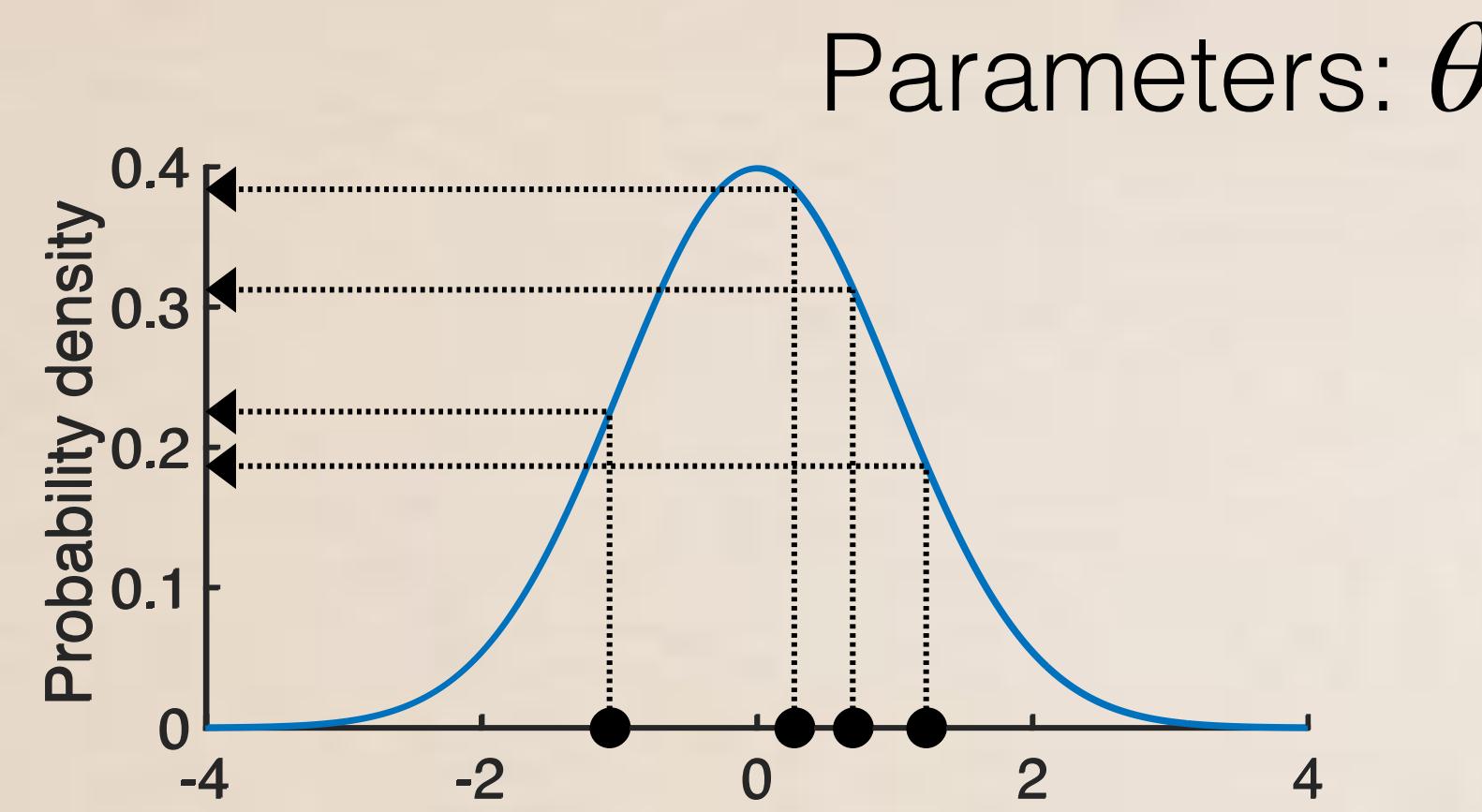
One (simple but suboptimal) option:
Minimize the sum of squared errors of mean RTs



Limitations of the sum of squares

- » There can be a big mismatch in the RT distributions between model and data
- » Not straightforward to combine choice and response time data
- » The sum of squared errors is suboptimal (except for linear models with normally-distributed errors)

Better approach: Maximum Likelihood



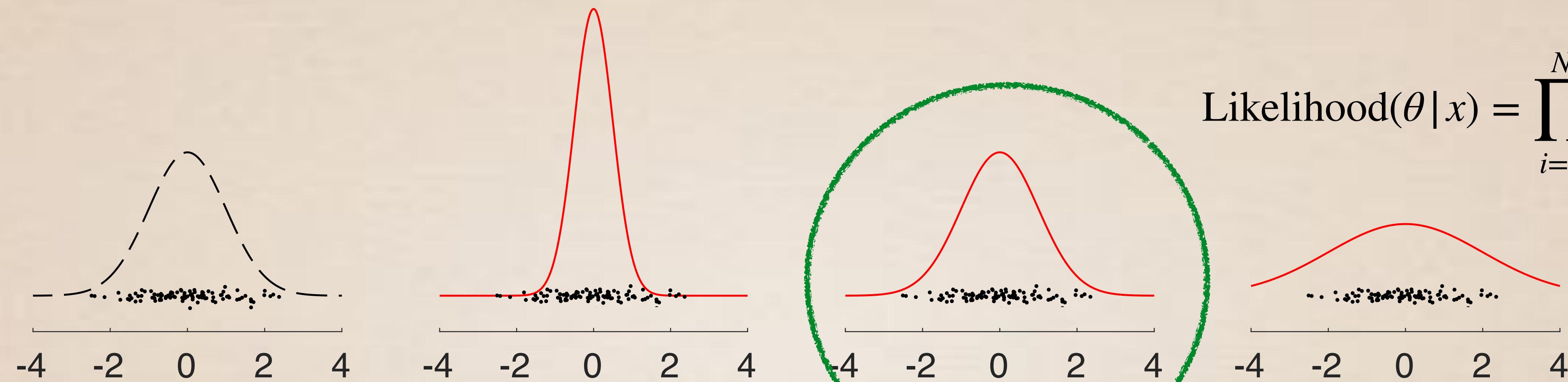
Independent and identically distributed observations (iid), coming from a distribution with unknown parameters θ^*

$$\text{Likelihood}(\theta | x) = \prod_{i=1}^N p(x^{(i)} | \theta)$$

data points
parameters

$$\text{Likelihood}(\theta | x) = 0.22 \times 0.18 \times 0.38 \times 0.31$$

Better approach: Maximum Likelihood



$$\text{Likelihood}(\theta | x) = \prod_{i=1}^N p(x^{(i)} | \theta)$$

It is more convenient to work with the logarithm of the likelihood for numerical reasons

$$LL = \sum_i \log(p(x^{(i)} | \theta))$$

data points
parameters

Exercise on maximum-likelihood

Exercise:

- Assume you know that the observations come from a normal distribution with mean $\mu = 0$
- But you do not know σ
- Compute the log-likelihood for each possible value of σ
- Find the value of σ that maximizes the log-likelihood
- Key Matlab command: `p = normpdf(data, 0, sigma(i))`

```
%% true parameters
mu_true = 0; % mean of the Normal generative distribution
sigma_true = 1; % standard deviation of the Normal generative distrib.

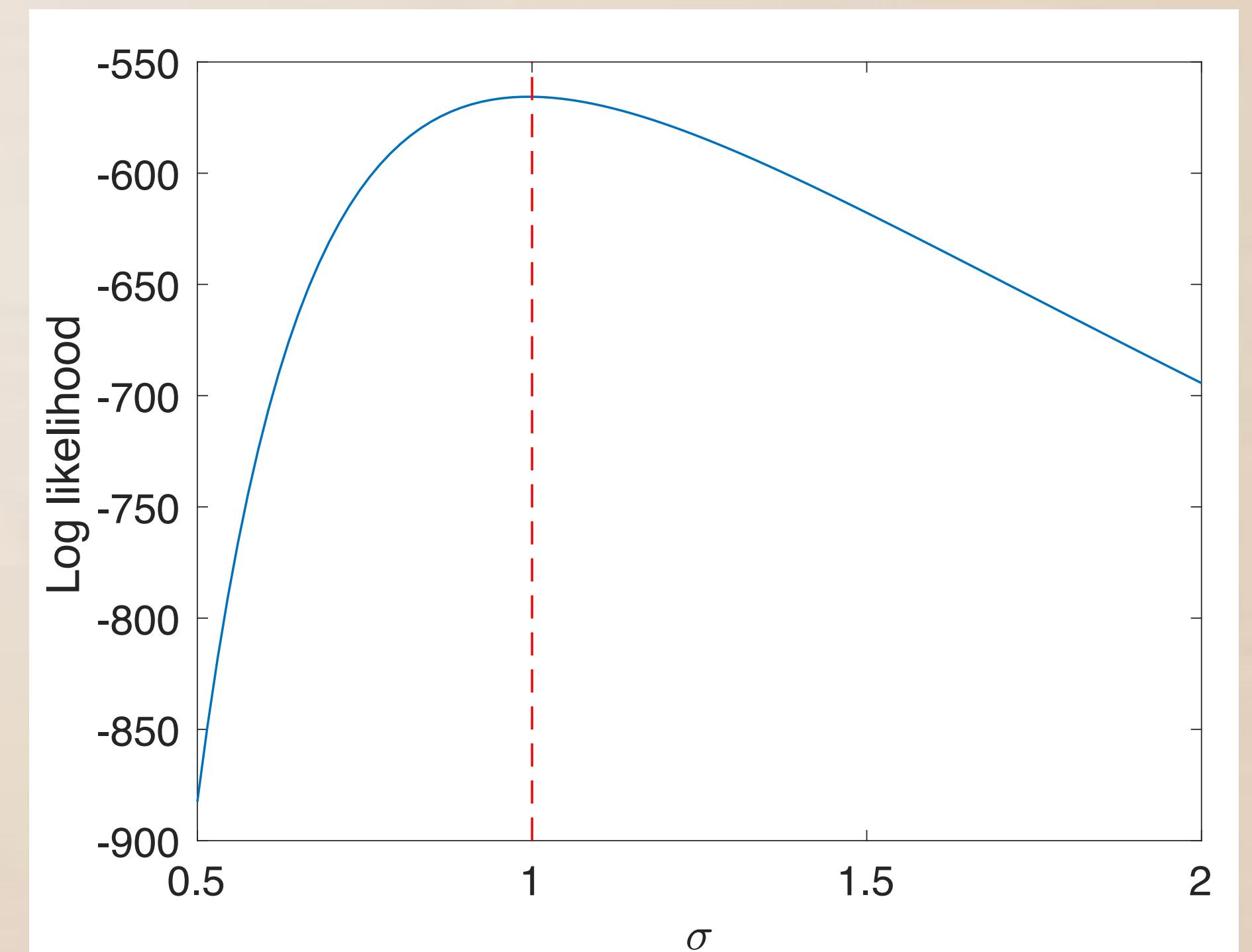
%% sample data points from a normal distribution (mu_true, sigma_true)
N = 400;
data = randn(N,1)*sigma_true + mu_true;

%% range of values of the standard deviation
sigma = linspace(0.5,2);

%% compute the log-likelihood for each sigma
LL = zeros(size(sigma));
for i=1:length(sigma)
    ps = normpdf(data, 0 , sigma(i));
    LL(i) = sum(log(ps));
end
I = find(LL==max(LL));
sigma_best = sigma(I(1));
```

$$LL = \sum_i \log(p(x^{(i)} | \theta))$$

data points
model parameters



(Assume $\mu = 0$)

Maximum Likelihood for choice & RT data

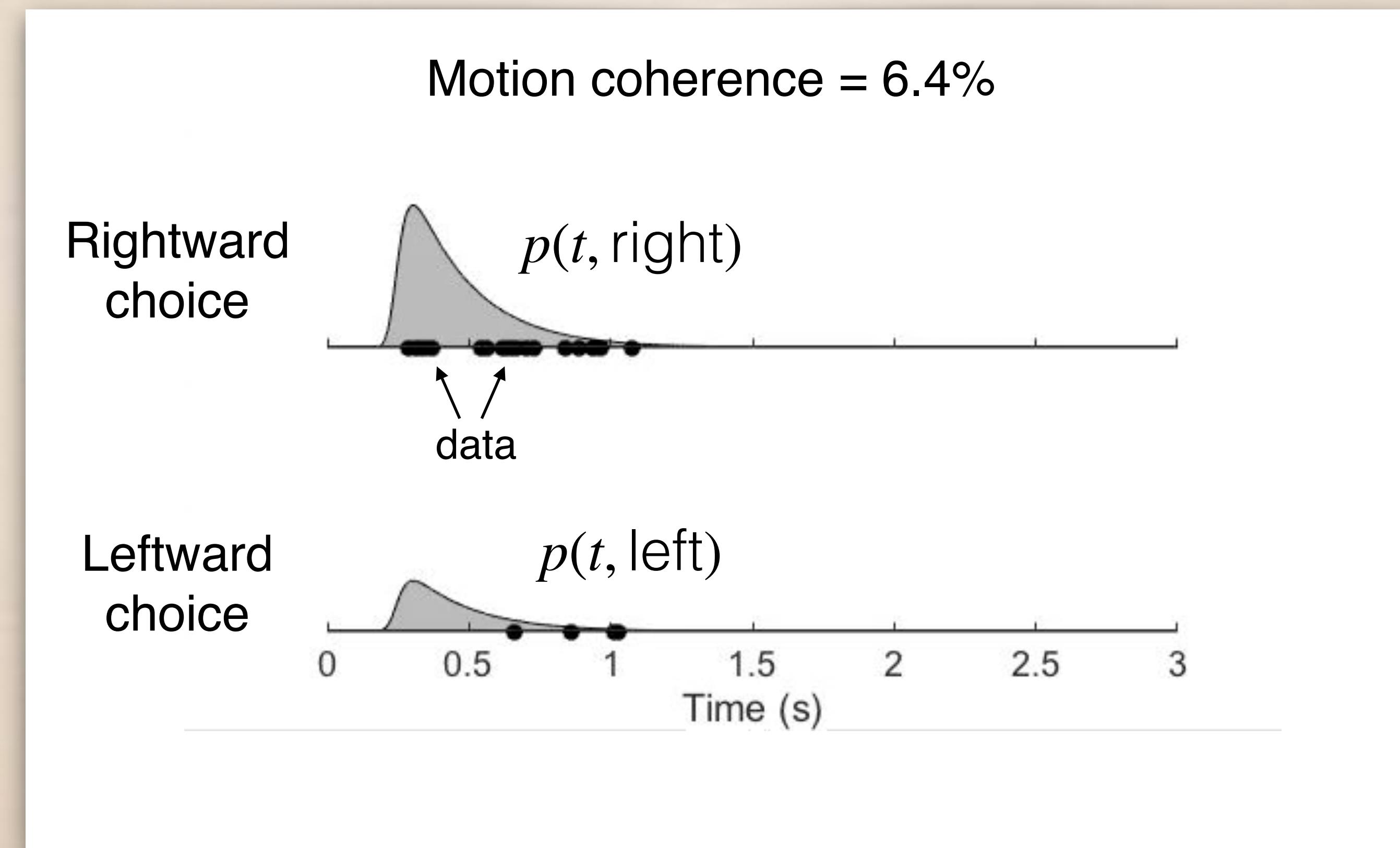
$$\hat{\theta}_{\text{mle}} = \arg \max_{\theta} \left(\sum_{i=1}^{n_{tr}} \log \left(p(\text{RT}^{(i)}, \text{choice}^{(i)} | c^{(i)}, \theta) \right) \right)$$

model parameters
motion
We need to be able to coherence evaluate this p.d.f.

Maximum Likelihood for choice & RT data

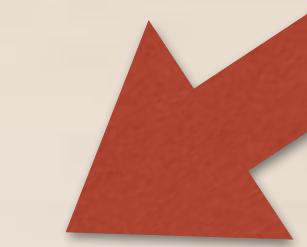
$$\hat{\theta}_{\text{mle}} = \arg \max_{\theta} \left(\sum_{i=1}^{n_{tr}} \log \left(p(\text{RT}^{(i)}, \text{choice}^{(i)} | c^{(i)}, \theta) \right) \right)$$

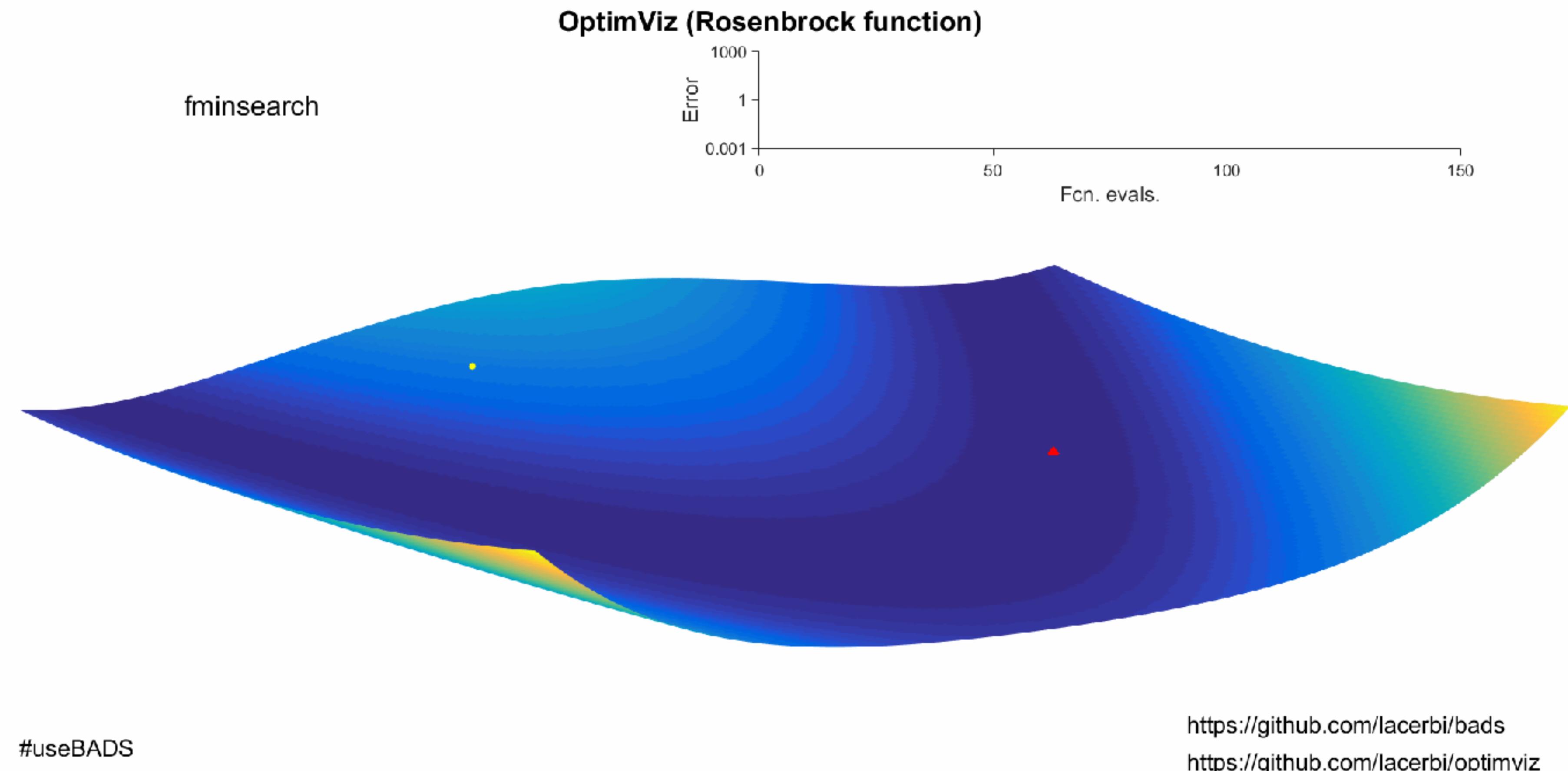
model parameters
motion coherence

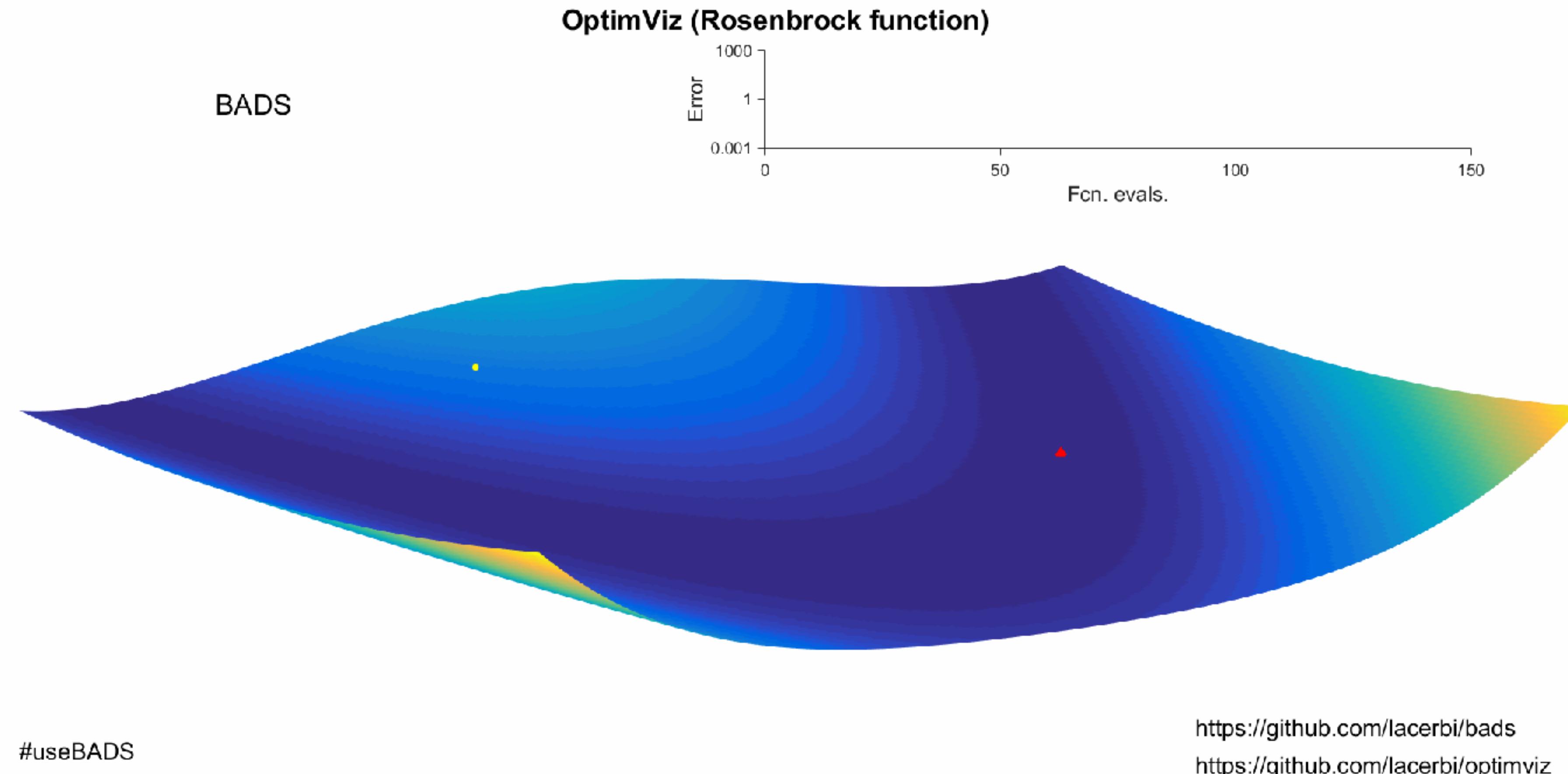


3. What fitting algorithm to use to search over the parameter space?

- » fminsearch (Nelder–Mead simplex method)
- » fmincon
- » Bayesian Adaptive Direct Search (BADS)
 - Luigi Acerbi & Wei Ji Ma (<https://github.com/lacerbi/bads>)







Let's try fitting a DDM!

1. How to predict choice & RT from a set of model parameters?

Using analytic solutions to the choice and RT functions

2. What to minimize/maximize when fitting parameters?

Minimize the sum of squared errors of the mean RTs

3. What fitting algorithm to use to search over the parameter space?

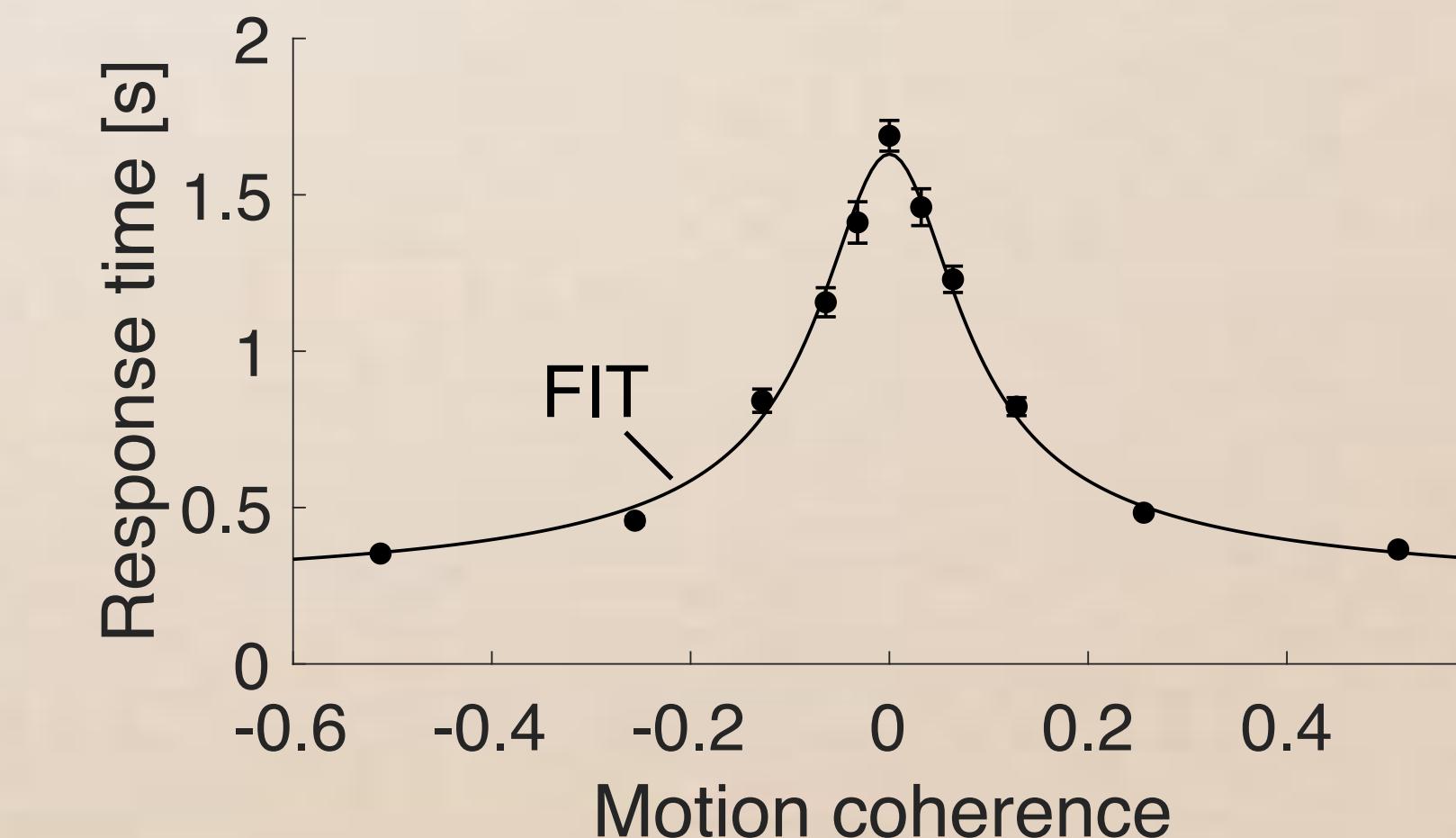
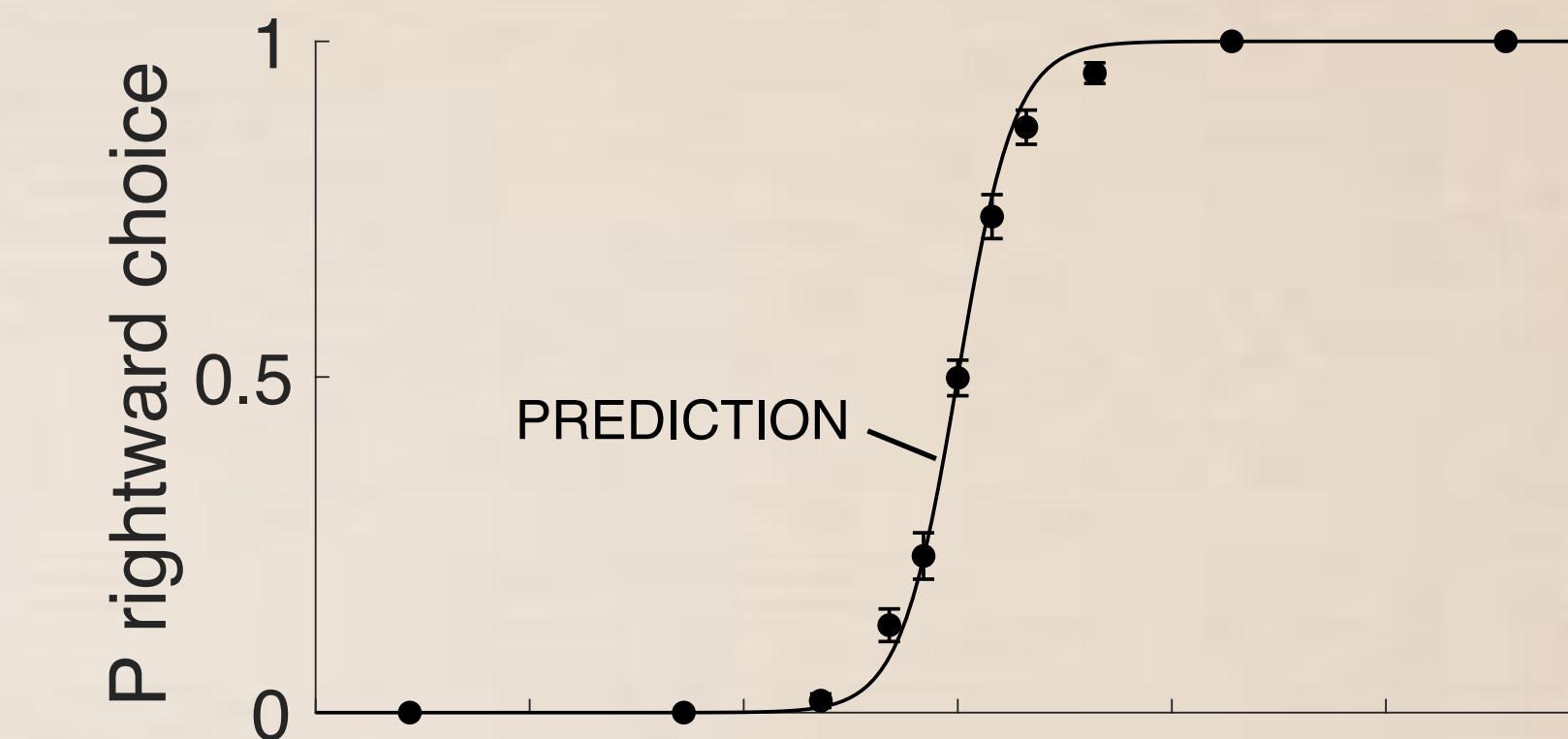
BADS

Model fitting

Code in “03_Fit_means/main.m”

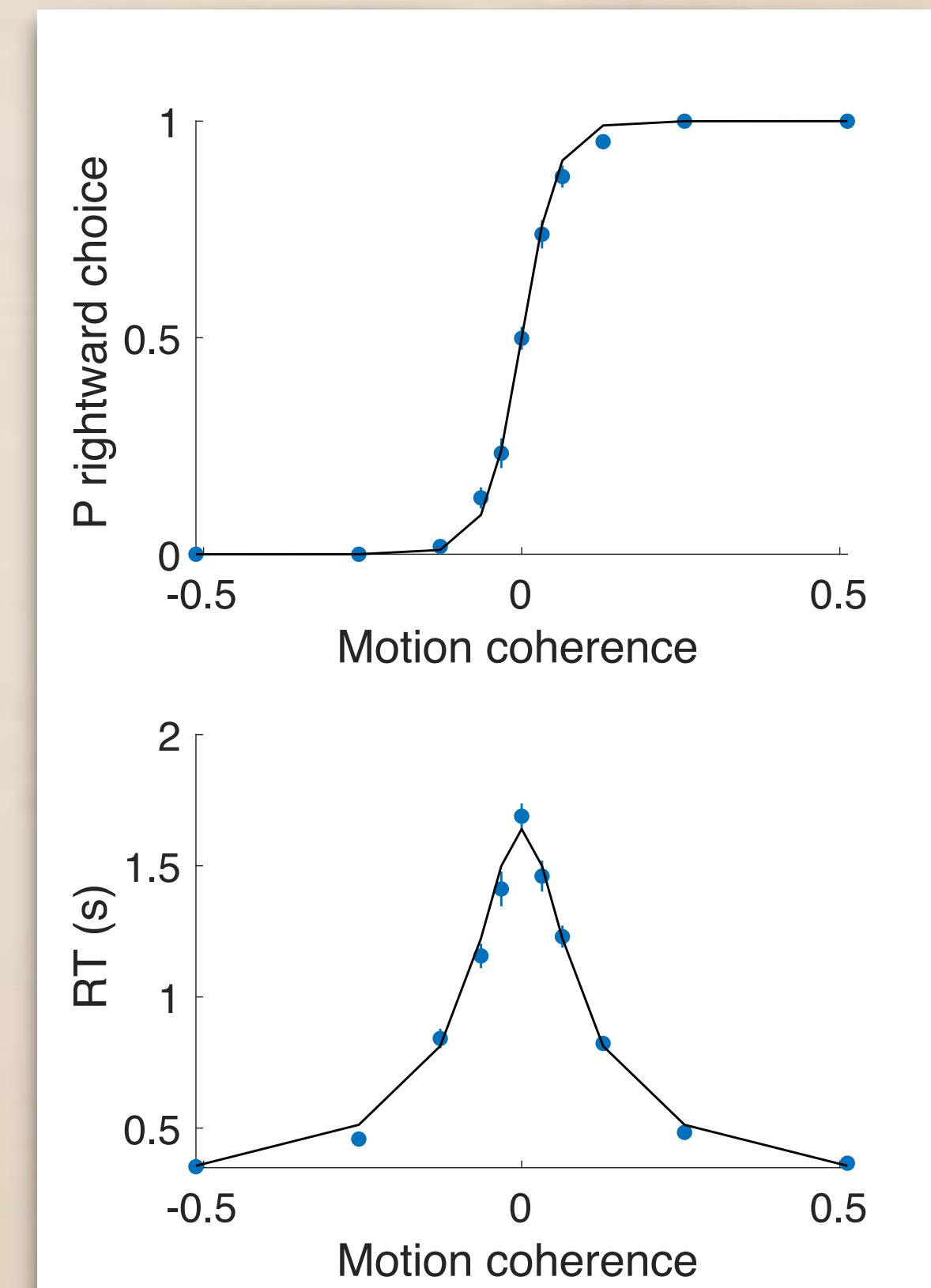
data/test_data.mat

Motion Coherence (%)	Choice	RT (s)	Correct
-3.2	0	2.32	1
-12.8	0	0.76	1
0.0	0	1.31	0
51.2	1	0.39	1
-51.2	0	0.38	1
-25.6	0	0.55	1
-12.8	0	0.52	1
0.0	0	1.40	0
51.2	1	0.31	1
-3.2	0	2.05	1



Code in “03_Fit_means/main.m”

```
1 %% add functions to path
2 addpath(genpath('..../functions_addtopath/'));
3
4 %% figure defaults
5 set(0,'defaultAxesFontSize',18);
6 set(0, 'DefaultLineLineWidth', 1);
7
8 %% load data
9 load ../data/test_data.mat
10
11 %% run one set of parameters
12 kappa = 15; % signal-to-noise parameters (\mu = kappa x coh)
13 ndt_m = 0.2; % non-decision time, mean [s]
14 ndt_s = 0; % non-decision time, standard deviation [s]
15 B0 = 1.2; % bound height
16 coh0 = 0; % bias (in units of coherence)
17 y0 = 0; % starting-point bias, as a fraction of bound height
18 ndt_m_delta = 0; % difference in non-decision time between right and left choices
19 theta = [kappa,ndt_m,ndt_s,B0,coh0,y0,ndt_m_delta]; % model parameters
20
21 params = struct('plot_flag',1);
22
23 % data struct
24 D = struct('rt',rt,... % response time
25 % motion coherence
26 'coh', coh,... % choice [0,1]
27 'choice',choice,... % correct [0,1]
28 'c',c);
29
30 % eval the model parameters and plot
31 [~,P] = wrapper_analytic_DDM(theta,D,params);
```



“wrapper_analytic_DDM.m” function:

```
88 %% solve the DDM> model
89 Bup = B0;
90 drift = kappa * unique(coh + coh0);
91
92 yp = y0a/B0; % as a proportion of the bound height
93 P = analytic_DDM(drift,t,Bup,yp);
94
95 % P: struct with the model output:
96 % P.up.p: probability of a rightward choice for each unique coherence [ncoh]
97 % P.up.mean_t: mean decision time for each coherence for rightward choices [ncoh]
98 % P.up.pdf_t: probability of reaching the upper bound at each time step for
99 % each unique coherence [ncoh x ntimes]
100 % P.up.cdf_t: Same as P.up.pdf_t, but cumulative over time
101 % P.lo: same as P.up for leftward choices
102 % P.drift: drift-rates for each unique coherence
103 % P.Bup: upper bound
104 % P.Blo: lower bound
105 % P.t: vector of times
106
```

The “analytic_DDM.m” function implements the analytical solutions to the choice and response time distributions for a DDM with constant bounds:

Probability of terminating at the lower bound at time t :

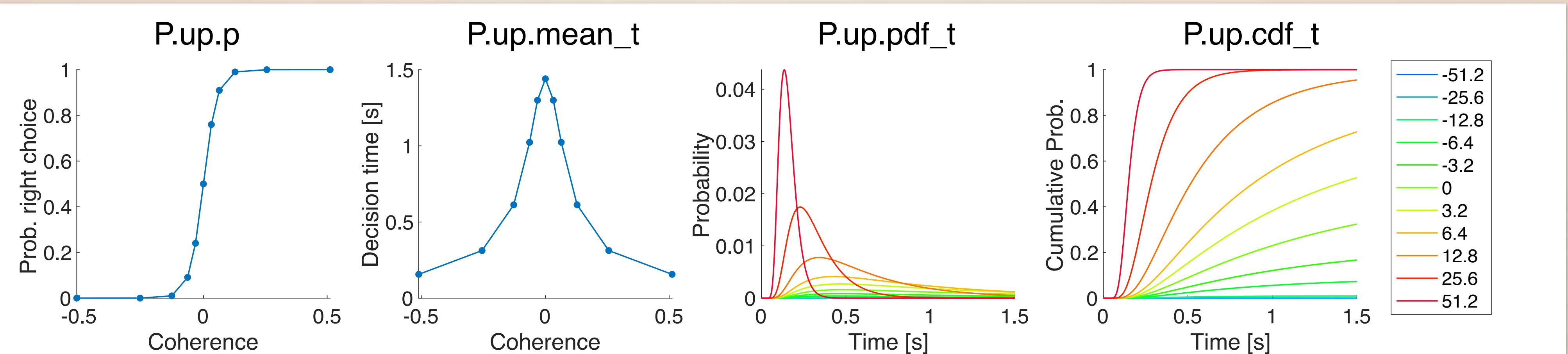
$$p(t, \text{left} | \mu, B, \sigma, x_0) = p_-(\mu, B, \sigma, x_0) - \frac{\pi\sigma^2}{4B^2} e^{-\frac{(B+x_0)\mu}{\sigma^2}} \times \sum_{k=1}^{\infty} \frac{2k \sin\left(\frac{k\pi(B+x_0)}{2B}\right) e^{-\frac{1}{2}\left(\frac{\mu^2}{\sigma^2} + \frac{\pi^2 k^2 \sigma^2}{4B^2}\right)t}}{\left(\frac{\mu^2}{\sigma^2} + \frac{\pi^2 k^2 \sigma^2}{4B^2}\right)}$$

, where

$$p_-(\mu, B, \sigma, x_0) = \frac{e^{-(4\mu B/\sigma^2)} - e^{-(2\mu(B+x_0)/\sigma^2)}}{e^{-(4\mu B/\sigma^2)} - 1}$$

“wrapper_analytic_DDM.m” function:

```
88 %% solve the DDM> model
89 Bup = B0;
90 drift = kappa * unique(coh + coh0);
91
92 yp = y0a/B0; % as a proportion of the bound height
93 P = analytic_DDM(drift,t,Bup,yp);
94
95 % P: struct with the model output:
96 % P.up.p: probability of a rightward choice for each unique coherence [ncoh]
97 % P.up.mean_t: mean decision time for each coherence for rightward choices [ncoh]
98 % P.up.pdf_t: probability of reaching the upper bound at each time step for
99 % each unique coherence [ncoh x ntimes]
100 % P.up.cdf_t: Same as P.up.pdf_t, but cumulative over time
101 % P.lo: same as P.up for leftward choices
102 % P.drift: drift-rates for each unique coherence
103 % P.Bup: upper bound
104 % P.Blo: lower bound
105 % P.t: vector of times
106
```



“wrapper_analytic_DDM.m” function:

```
107  %% calc. the criteria to minimize (MSE, likelihood, etc.)  
108  if calc_optim_criteria  
109  
110      if isfield(params,'optim_method')  
111          optim_method = params.optim_method;  
112      else  
113          optim_method = 3;  
114      end  
115  
116      switch optim_method  
117          case 1 % fit choice and RT full distribution  
118              [err,pPred] = logl_choiceRT_1d(P,choice,rt,coh,ndt_m,ndt_s);  
119  
120  
121          case 2 % sum of squares errors of the mean RTs, correct trials only  
122  
123              [~,mRT,eRT] = curva_media(rt,coh,c==1,0);  
124              udrift = P.drift;  
125              mean_dt = nan(size(udrift));  
126              mean_dt(udrift>0) = P.up.mean_t(udrift>0);  
127              mean_dt(udrift<0) = P.lo.mean_t(udrift<0);  
128              mean_dt(udrift==0) = 0.5 * (P.lo.mean_t(udrift==0) + P.up.mean_t(udrift==0)); % assumes equal  
129              % number of right and left  
130  
131              ndtm = nan(size(udrift));  
132              ndtm(udrift>0) = ndt_m + ndt_mu_delta;  
133              ndtm(udrift<0) = ndt_m;  
134              ndtm(udrift==0) = ndt_m + ndt_mu_delta/2; % assume equal number of right and left  
135  
136              err = sum((mean_dt + ndtm - mRT).^2);  
137
```

Code in “03_Fit_means/main.m”. Now the fitting...

```
32 %% fit parameters, sum of squared errors
33
34 % kappa, ndt_mu, ndt_sigma, B0, coh0, y0, ndt_m_delta
35 tl = [5, 0.1, 0 ,.5 ,0,0,0]; % lower limit
36 th = [40, 0.5, 0 ,3 ,0,0,0]; % upper limit
37 tg = [15, 0.2, 0 ,1 ,0,0,0]; % guess
38
39 params = struct('plot_flag',1,'optim_method',2); % sum of squared errors of RT
40
41 fn_fit = @(theta) (wrapper_analytic_DDM(theta,D,params));
42
43 MaxFunEvals = 200; % For the tutorial only, so it does not take too long
44 options = optimset('Display','final','TolFun',.01,'FunValCheck','on',...
45 'MaxFunEvals',MaxFunEvals);
46 ptl = tl;
47 pth = th;
48 [theta_SE,fval,exitflag,output] = bads(@(theta) fn_fit(theta),tg,tl,th,ptl,pth,options);
49
```

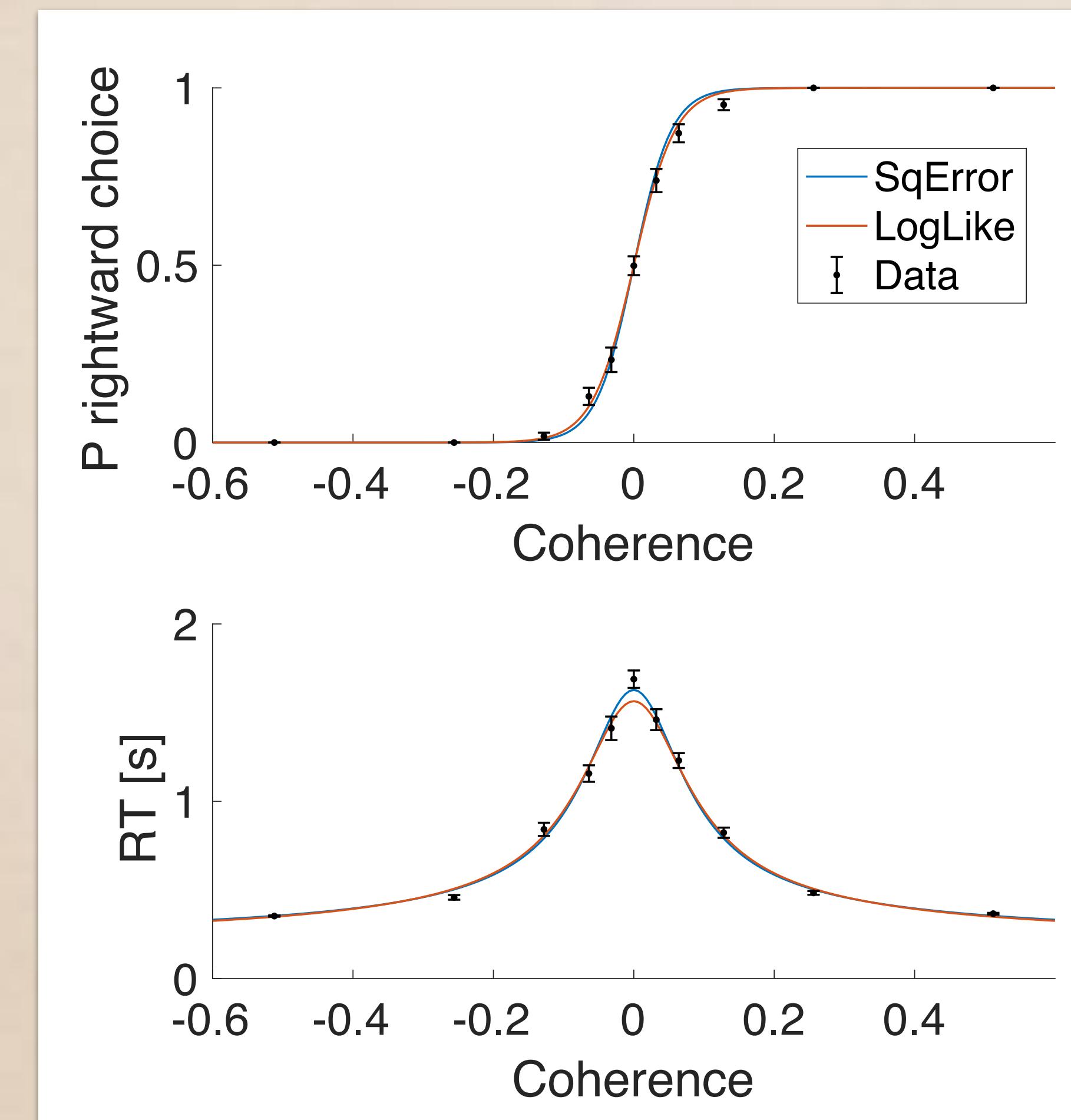
Let's run it in Matlab...

Compare fits using sum of squared errors and maximum-likelihood:

“03_Fit_means/main.m”.

39

```
params = struct('plot_flag',1,'optim_method',2);
```



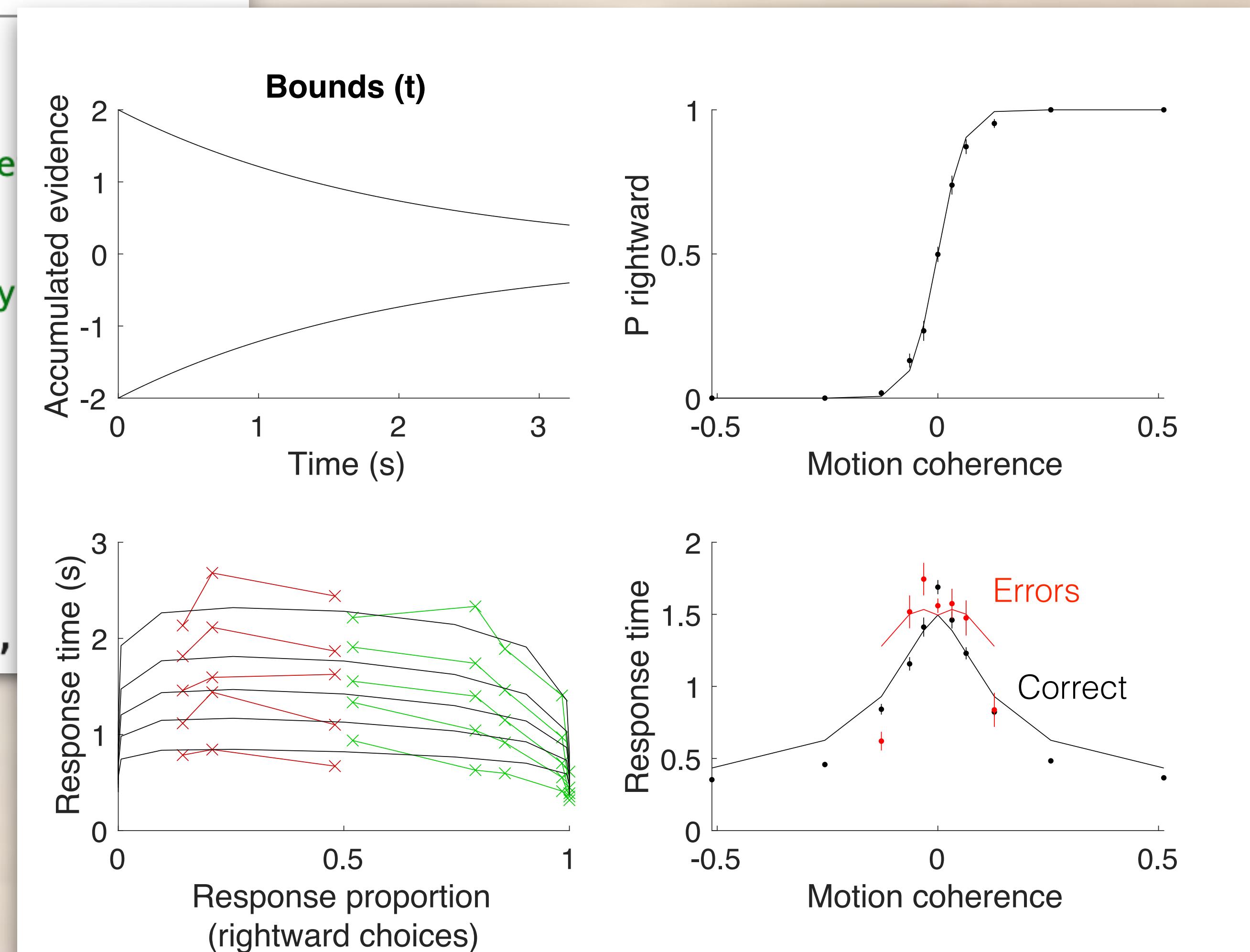
Let's fit a model with collapsing bounds

1. How to predict choice & RT from a set of model parameters?
~~Using analytic solutions to the choice and RT functions~~
Using numerical approximations to the Fokker Planck equation
2. What to minimize/maximize when fitting parameters?
~~Minimize the sum of squared errors~~
Maximize the likelihood of the parameters given the single-trial choice and RT data
3. What fitting algorithm to use to search over the parameter space?
BADS

Go to “04_Fit_single_trials/main.m”...

“04_Fit_single_trials/main.m”

```
1 addpath(genpath('..../functions_addtopath/'))  
2  
3 load('..../data/test_data.mat','rt','coh','c','choice');  
4  
5 D = struct('rt',rt,'coh', coh,'choice',choice,'c',c);  
6  
7 %% one example run  
8 kappa = 15; % signal-to-noise  
9 ndt_m = 0.2; % non-decision time, mean [s]  
10 ndt_s = 0.01; % non-decision time, standard de  
11 B0 = 2; % bound height  
12 a = 0.5; % bound collapse parameter  
13 d = 0; % bound parameter, time of decay  
14 y0 = 0; % bias, on starting point  
15 coh0 = 0; % bias, on coherence  
16  
17 theta = [kappa,ndt_m,ndt_s,B0,a,d,coh0,y0];  
18  
19 pars = struct('plot_flag',true);  
20  
21 [err,P] = wrapper_DTB_parametricbound(theta,D,  
22
```



“wrapper_DTB_parametricbound.m”

```
60  %% set the time vector
61  if ~isempty(params) && isfield(params,'t')
62      t = params.t;
63  else
64      dt = 0.0005;
65      t = 0:dt:10;
66  end
67
68  %% bounds
69  if ~isempty(params) && isfield(params,'USfunc')
70      USfunc = params.USfunc;
71  else
72      USfunc = 'Exponential';
73  end
74  [Bup,Blo] = expand_bounds(t,B0,a,d,USfunc);
75
76  %% discretize space and initialize the decision variable
77  y = linspace(min(Blo)-0.3,max(Bup)+0.3,750)';
78
79  y0a = clip(y0a,Blo(1),Bup(1));
80
81  y0 = zeros(size(y));
82  y0(findclose(y,y0a)) = 1;
83  y0 = y0/sum(y0);
84
```

Discretize time

Time-varying bounds

Discretize the possible DVs

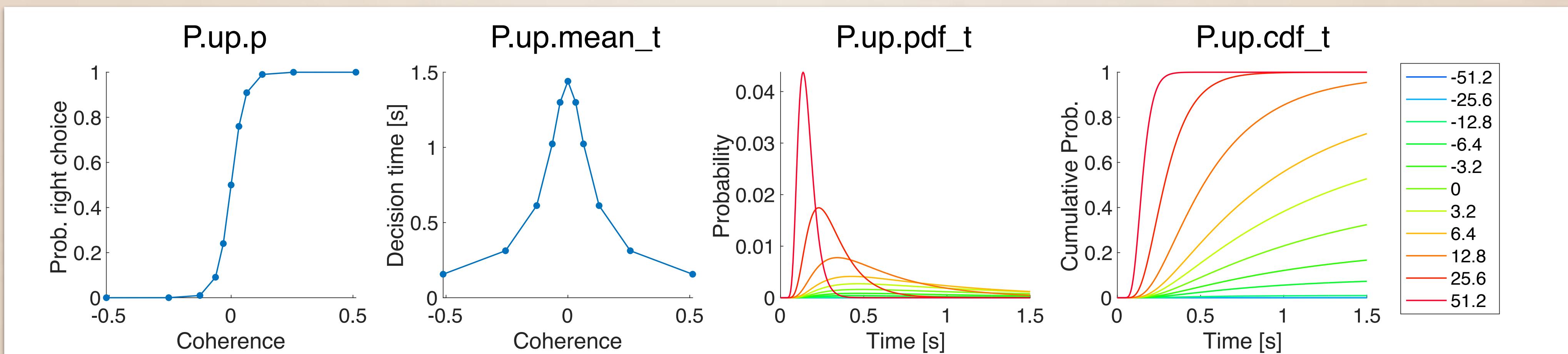
““wrapper_DTB_parametricbound.m””

```
92 %% numerical solution using Chang-Cooper's method
93 drift = kappa * unique(coh + coh0);
94 P = dtb_fp_cc_vec(drift,t,Bup,Blo,y,y0,notabs_flag); ←
95
96 %%
97 if calc_optim_criteria
98
99 %% likelihood
100 err = logl_choiceRT_1d(P,choice,rt,coh,ndt_m,ndt_s); ←
101
```

Numerical solution to the Fokker-Planck equation (Chang & Cooper 1970)

Computes the likelihood of the parameters given the single-trial choice and RT

Struct “P”:



“main.m”

Fitting using BADS just as before...

```
23 %% fitting
24
25 % kappa, ndt_mu, ndt_sigma, B0, a, d, coh0, y0
26 tl = [5, 0.1, .01 ,0.5 , -1, -3,0,0];
27 th = [40, 0.7, .15 ,4 , 4 ,4,0,0];
28 tg = [15, 0.2, .02 ,1 , 0.1 ,1,0,0];
29
30 pars = struct('plot_flag',true,'USfunc','Logistic');
31
32 fn_fit = @(theta) (wrapper_DTB_parametricbound(theta,D,pars));
33
34 MaxFunEvals = 100; % For the tutorial only, so it does not take too long
35 options = optimset('Display','final','TolFun',.01,'FunValCheck','on',...
36 'MaxFunEvals',MaxFunEvals);
37
38
39 ptl = tl;
40 pth = th;
41 [theta, fval, exitflag, output] = bads(@(theta) fn_fit(theta),tg,tl,th,ptl,pth,options);
42
43 % run the DTB with the best fit
44 [~,P] = fn_fit(theta);
45
```

Matlab exercise: fit the data from Roitman & Shadlen JNeurosci 2002

Adapt the code in “04_Fit_single_trials/main.m” to fit the data from Roitman and Shadlen, Journal of Neuroscience 2002.

The data is in “/data/roitman_data.mat”

You should have obtained something like this...

Best fitting parameters:

$$\kappa = 12.77$$

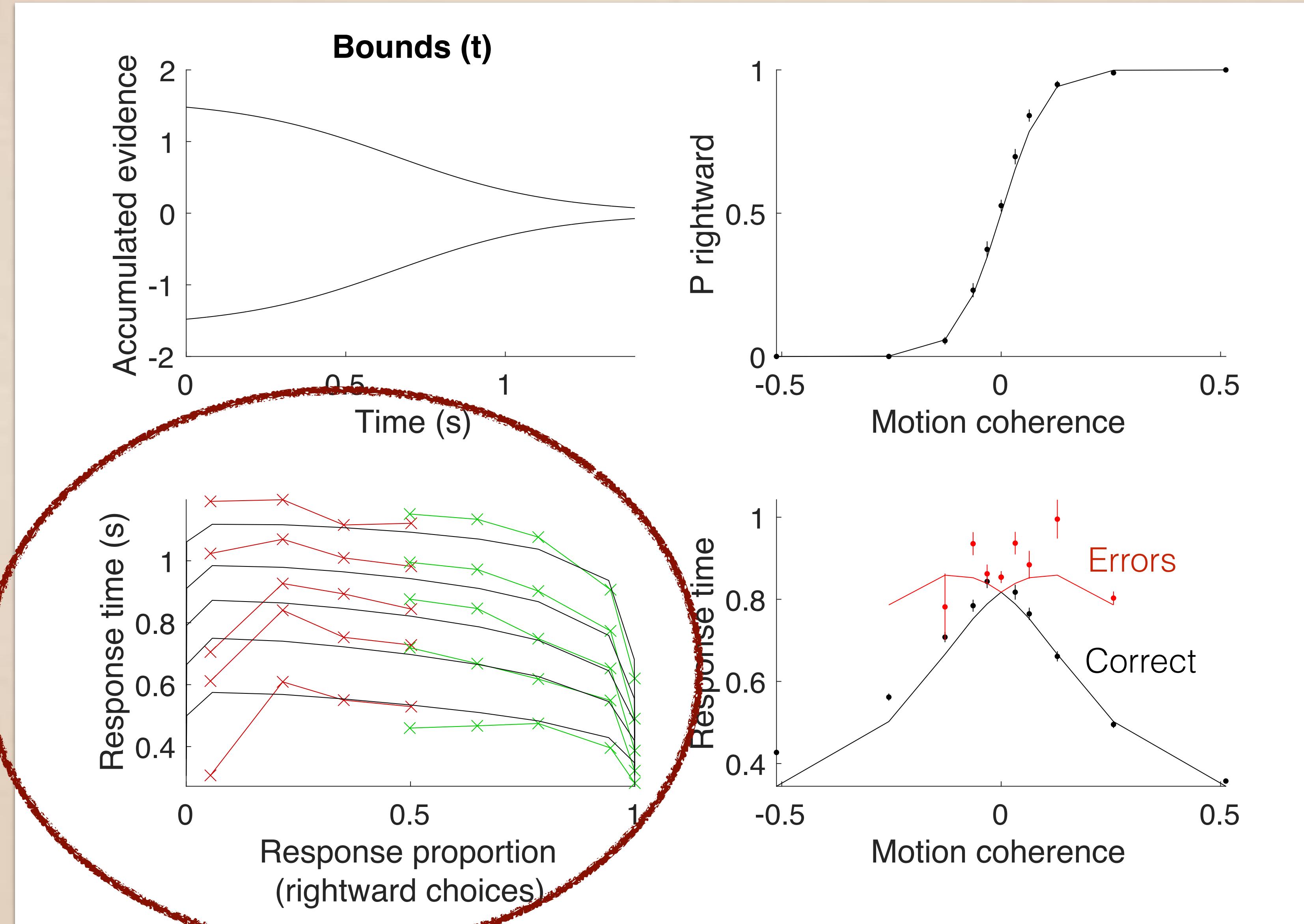
$$\mu_{nd} = 0.13$$

$$\sigma_{nd} = 0.13$$

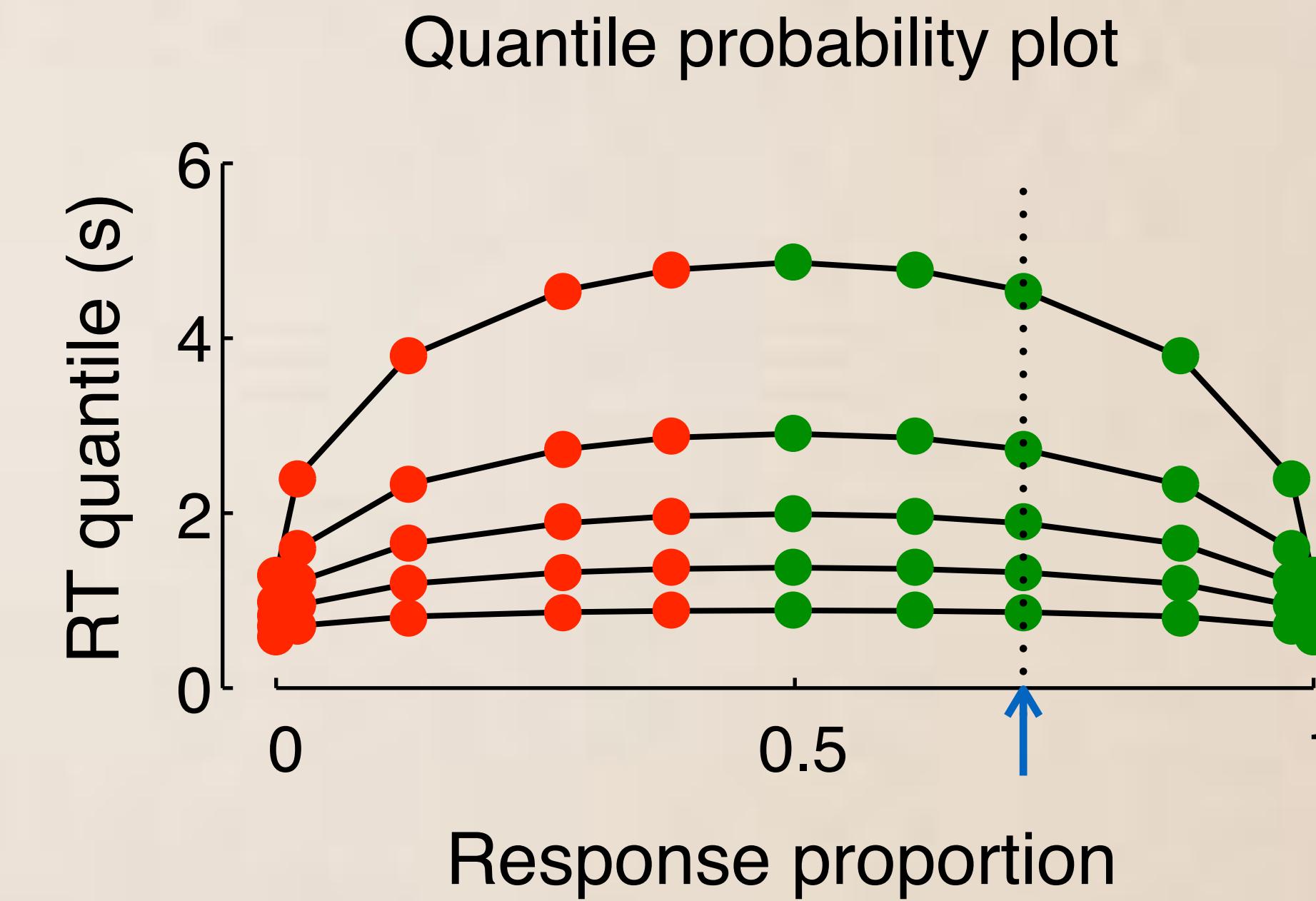
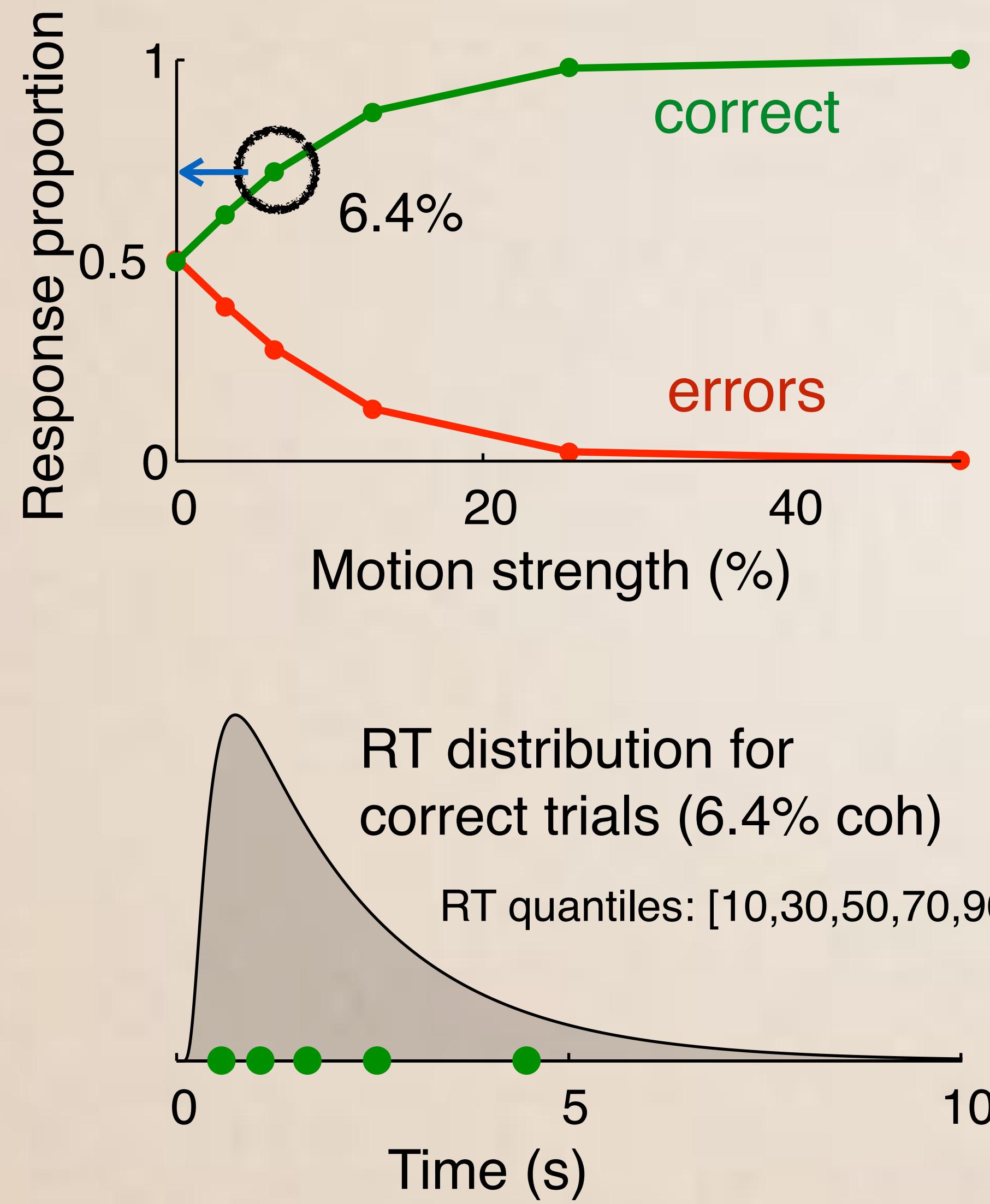
$$B_0 = 1.59$$

$$a = 4$$

$$d = 0.66$$



Quantile probability plot



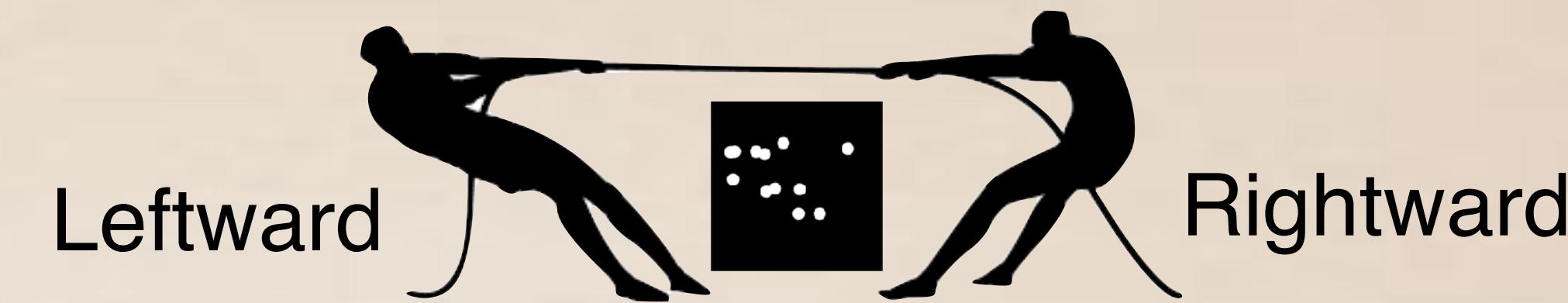
We have been focusing on the DDM...

The DDM is not the only model attempting to
explain choice-RT data...

Not either the first one

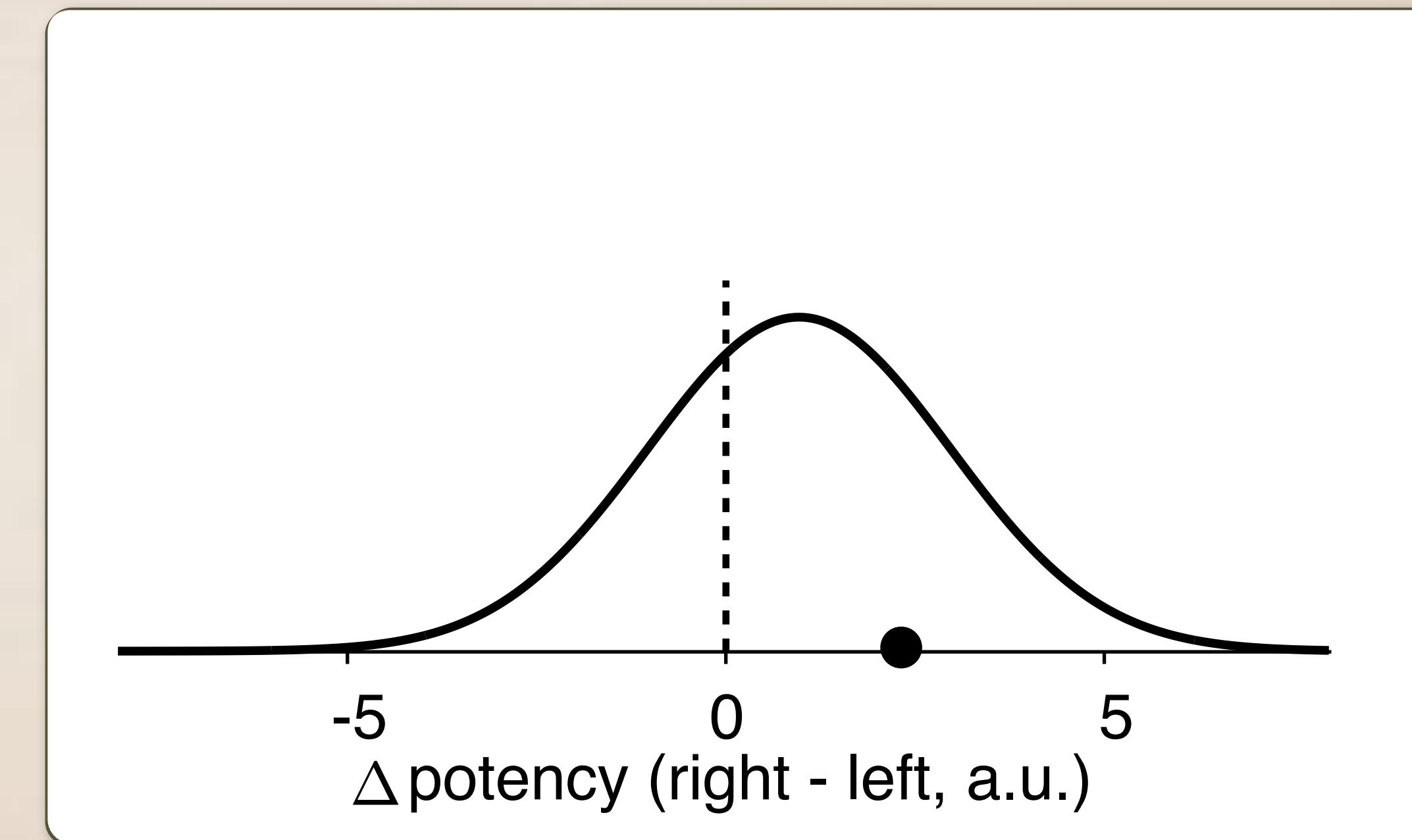
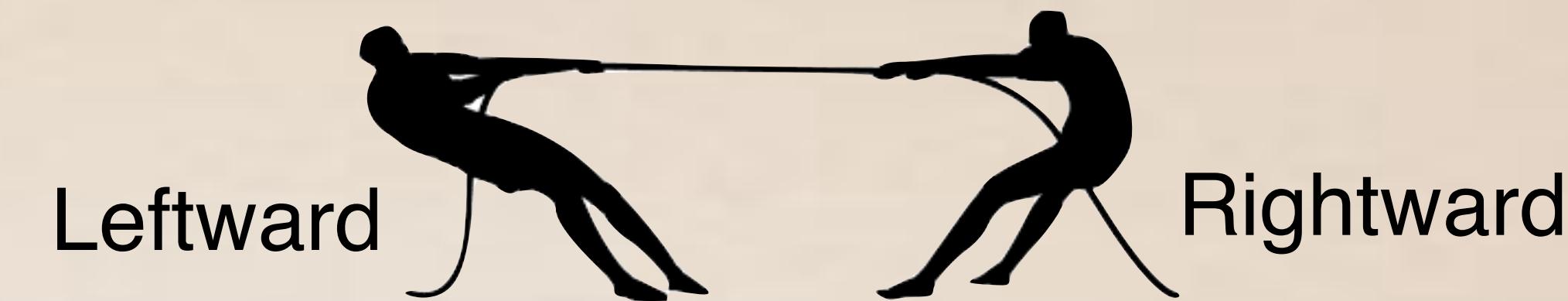
Opponent process theory of discrimination

Cartwright and Festinger (1943), “A quantitative theory of decision”.



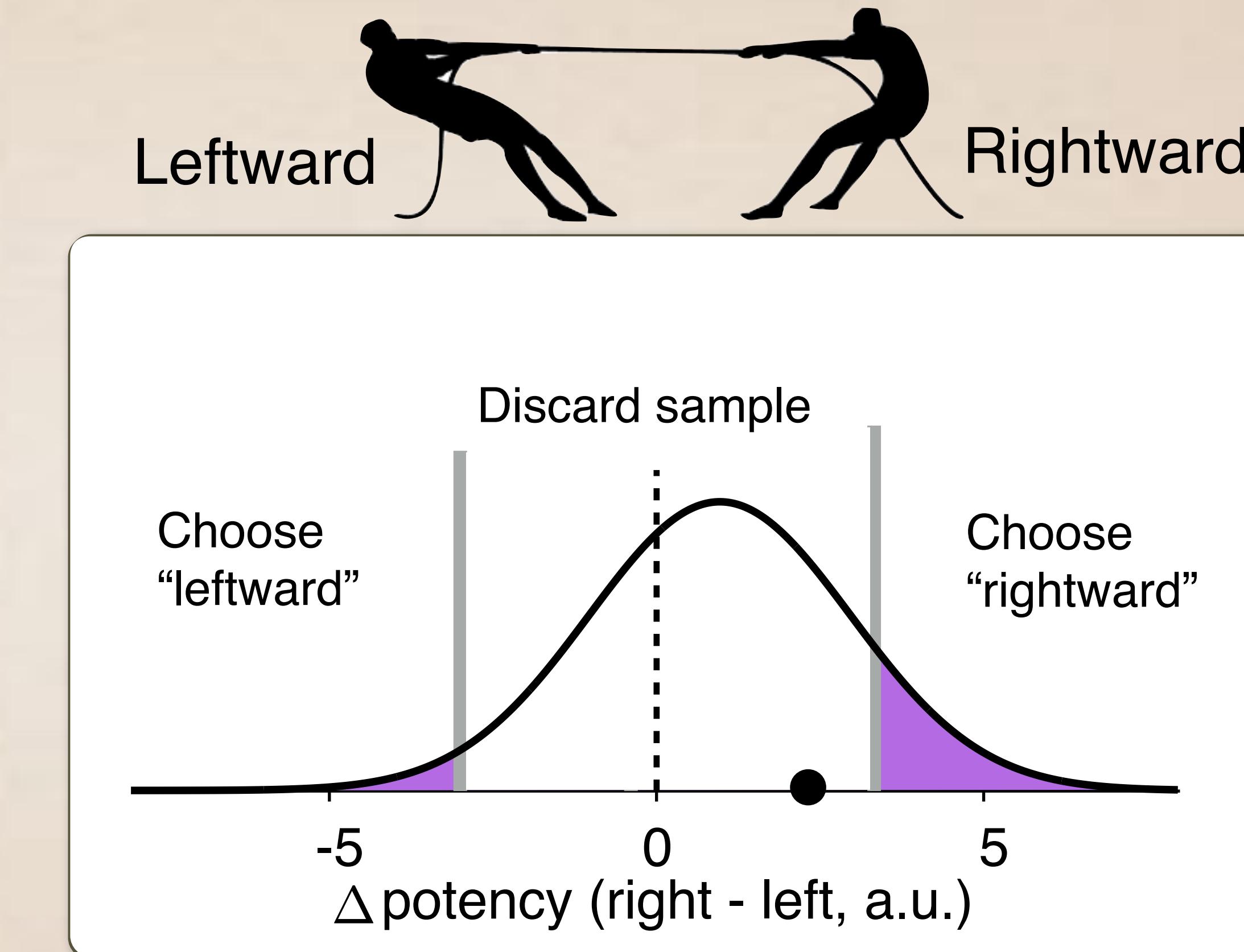
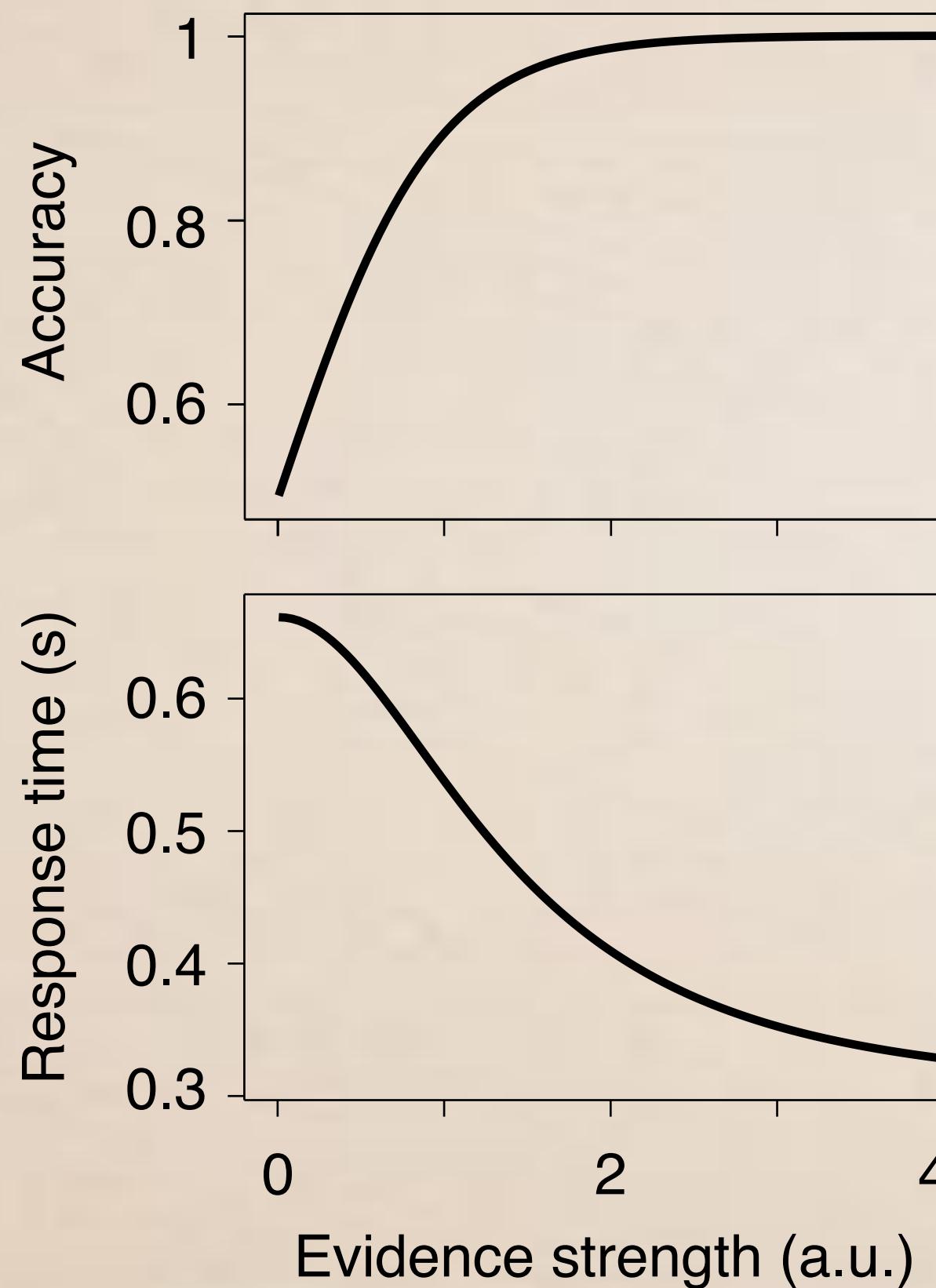
Opponent process theory of discrimination

Cartwright and Festinger (1943), “A quantitative theory of decision”.



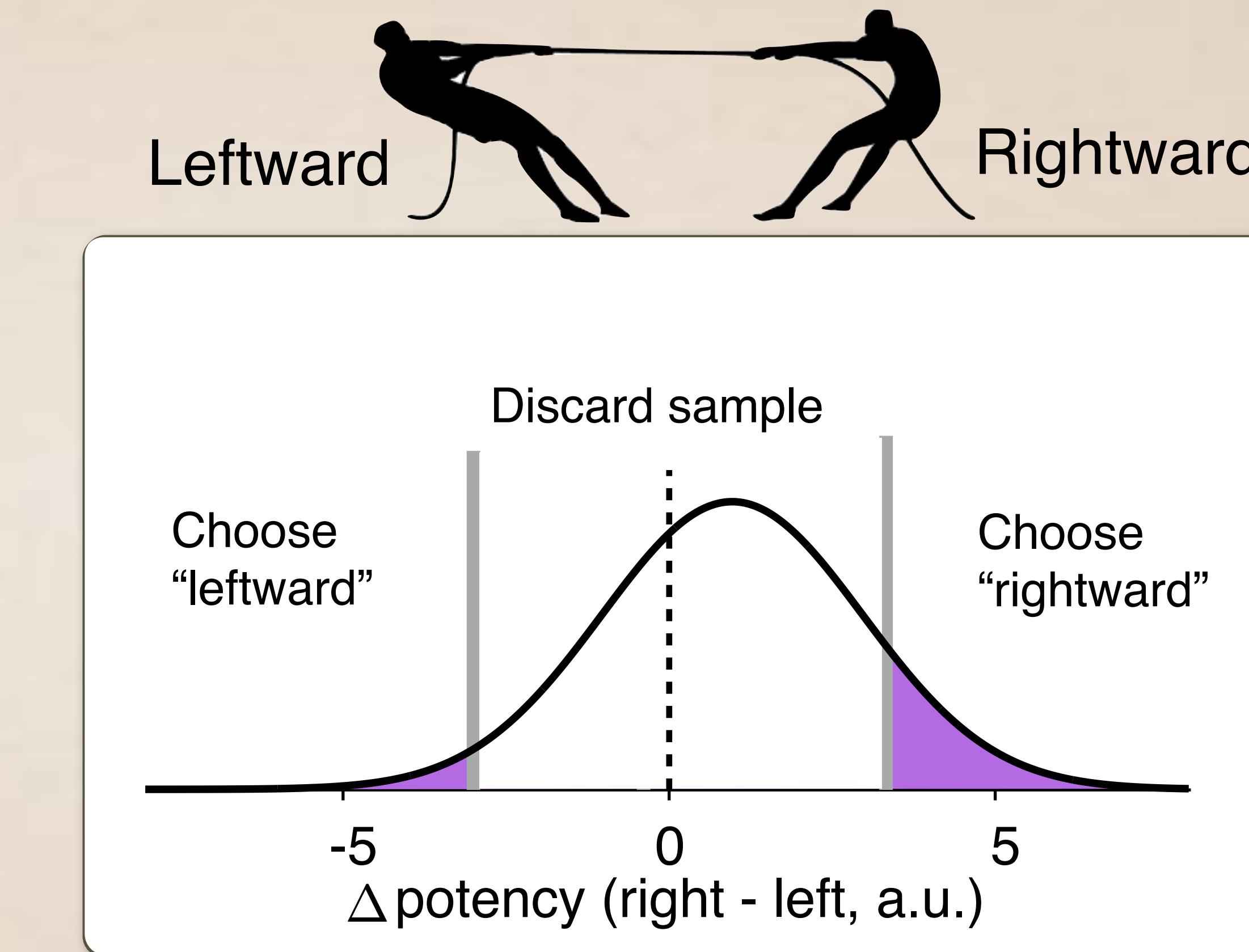
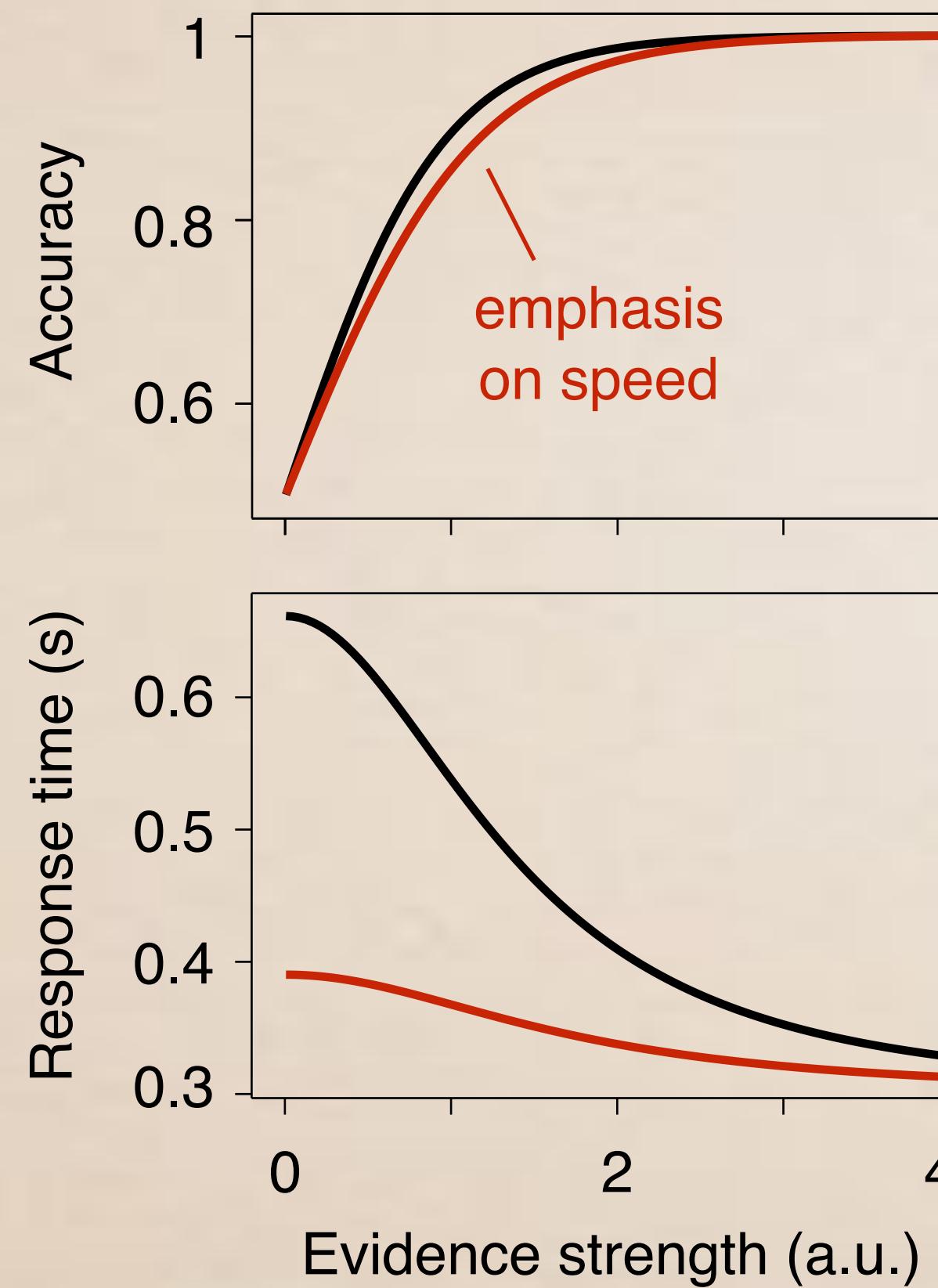
Opponent process theory of discrimination

Cartwright and Festinger (1943), “A quantitative theory of decision”.



Opponent process theory of discrimination

Cartwright and Festinger (1943), “A quantitative theory of decision”.



» No accumulation of evidence!

Can we distinguish between data
generated by the DDM and by
alternative (e.g., Extrema) models?

Compare fit parameters across conditions/datasets

- » In "data/sim_dat_extrema.mat" there are two behavioral datasets (sim(1) and sim(2)).
- » For example, these may correspond to two sets of participants performing the same task (e.g., patients and controls).
- » Exercise:
 - Fit the DDM to these two data sets by editing a copy of the "main.m" file.
 - Compare and interpret the parameter changes between the two data sets.

Fits of the DDM to the 1st dataset

Best fitting parameters:

$$\kappa = 24.34$$

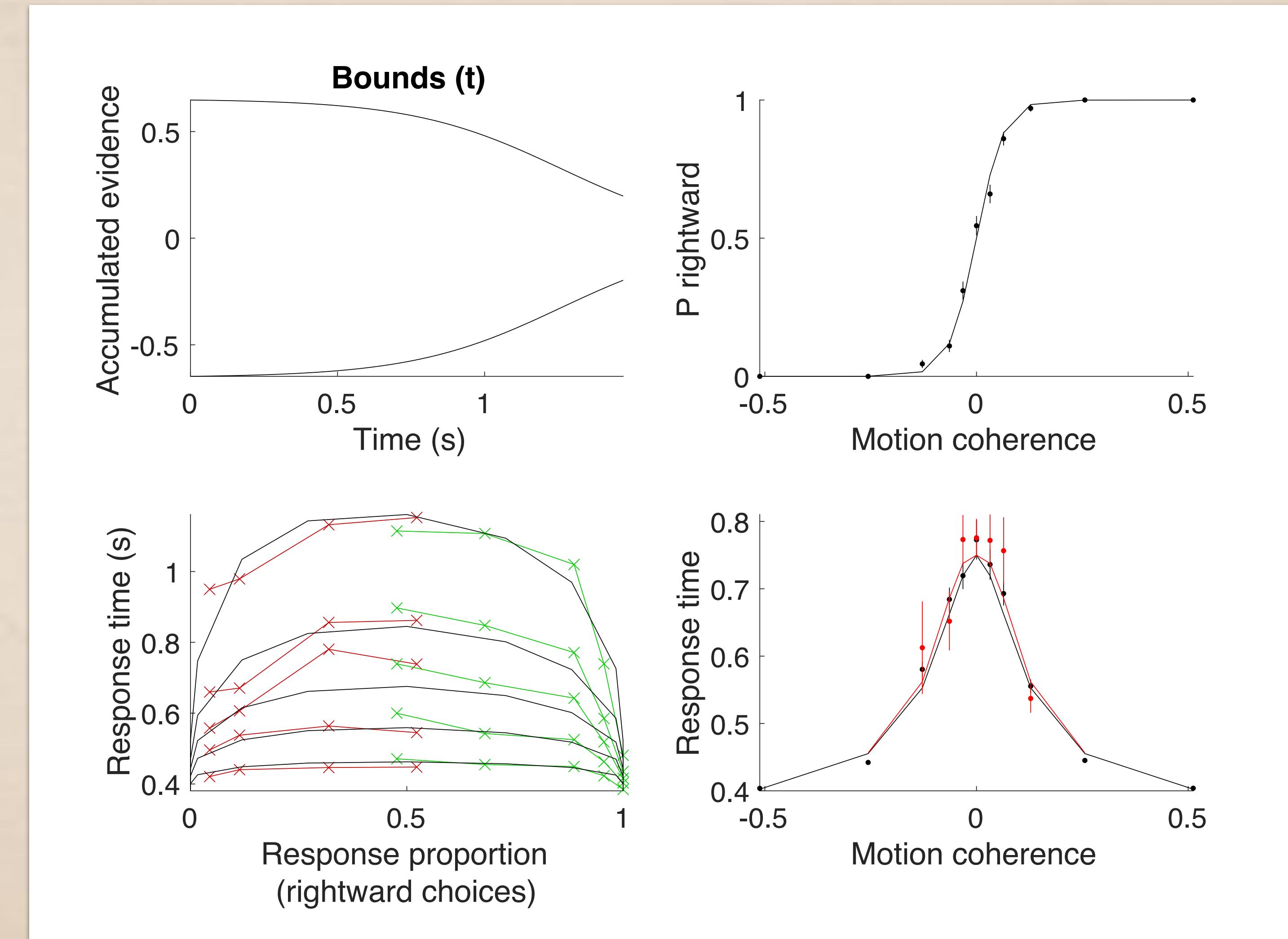
$$\mu_{nd} = 0.35$$

$$\sigma_{nd} = 0.01$$

$$B_0 = 0.65$$

$$a = 3.95$$

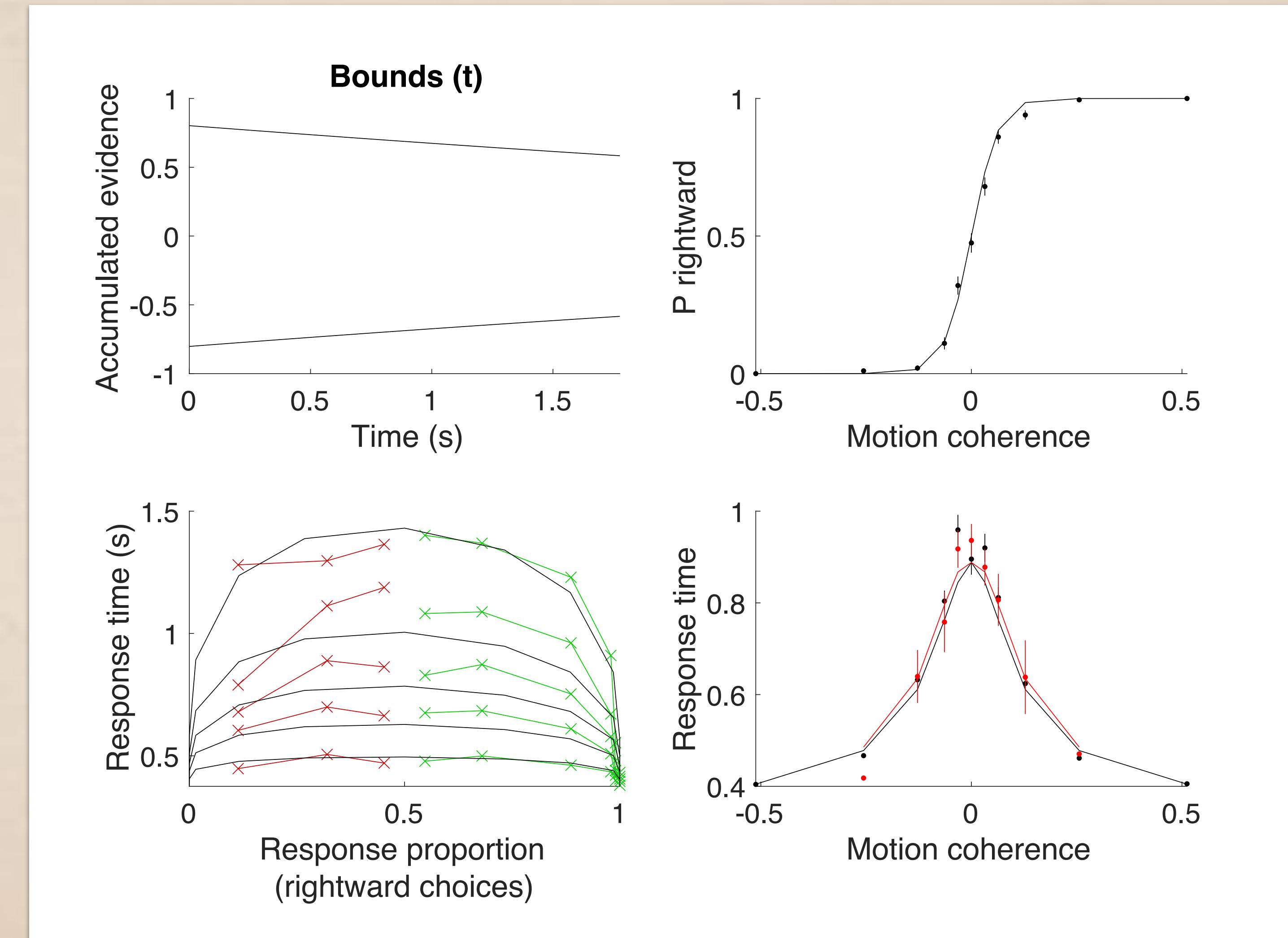
$$d = 1.26$$



Fits of the DDM to the 2nd dataset

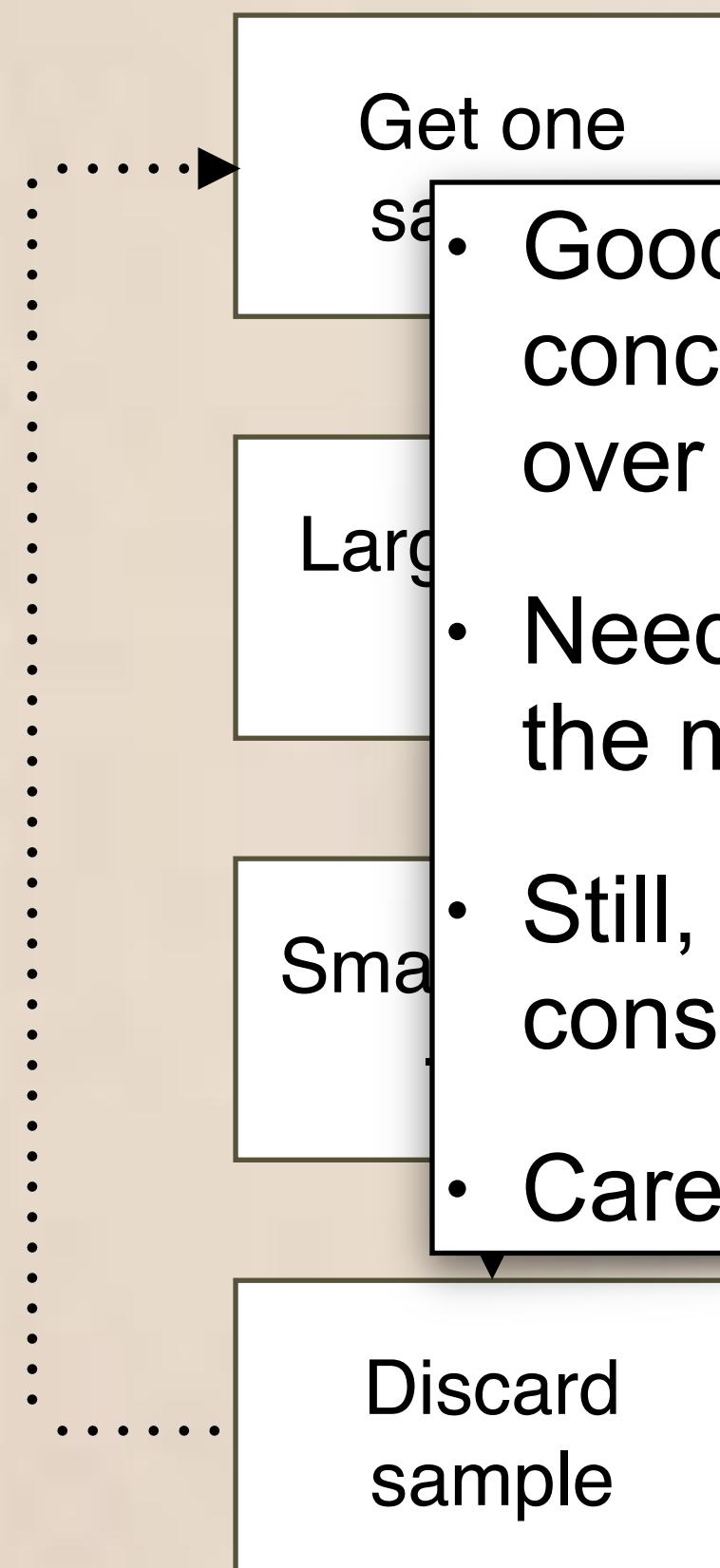
Best fitting parameters:

κ = 24.34	21
μ_{nd} = 0.35	0.33
σ_{nd} = 0.01	0.01
B_0 = 0.65	2.43
a = 3.95	0.25
d = 1.26	-2.82



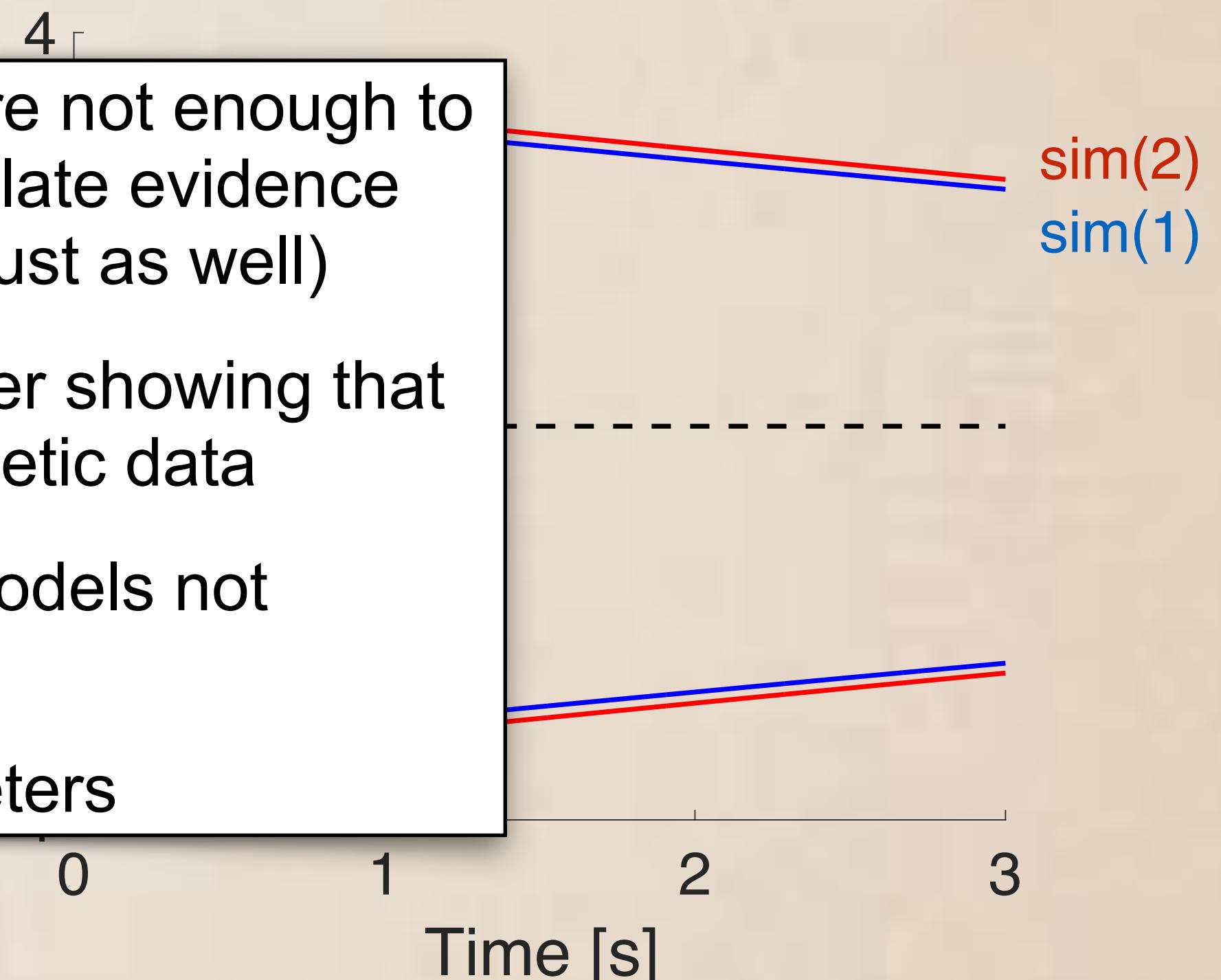
The data was generated with the ‘extrema’ model; only B_0 changed across conditions

Cartwright and Festinger 1943



- Good fits from a drift-diffusion model are not enough to conclude that decision makers accumulate evidence over time (alternative models could fit just as well)
- Need for formal model comparison, after showing that the models are distinguishable in synthetic data
- Still, there will be a blindspot (due to models not considered)
- Careful when interpreting fitted parameters

True difference between conditions:

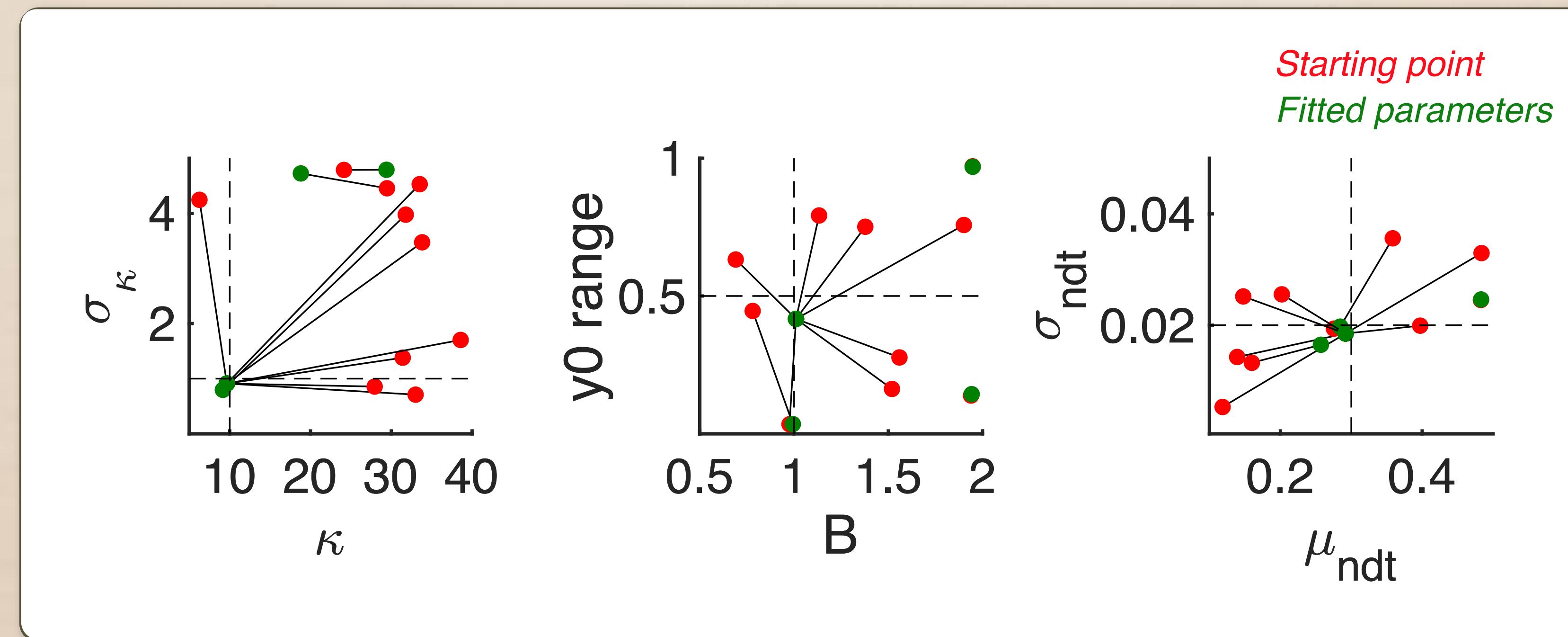


Validating the model fitting method (are parameters identifiable?)

- » Fit model to experimental data
- » Simulate data with the fitted parameters
- » Fit the simulated data and see if the optimization procedure recovers the parameters used to generate the data

Validating the model fitting method (are parameters identifiable?)

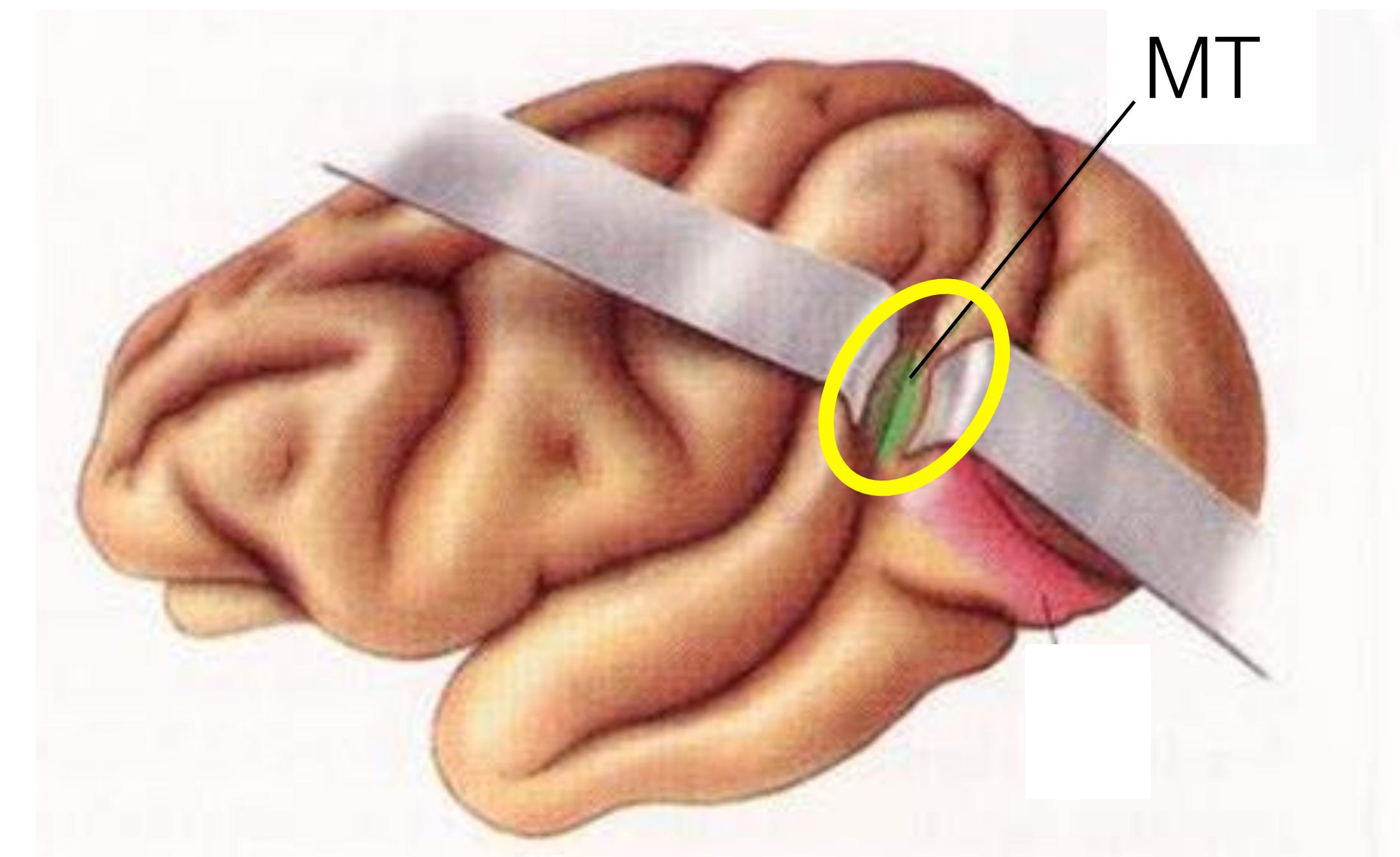
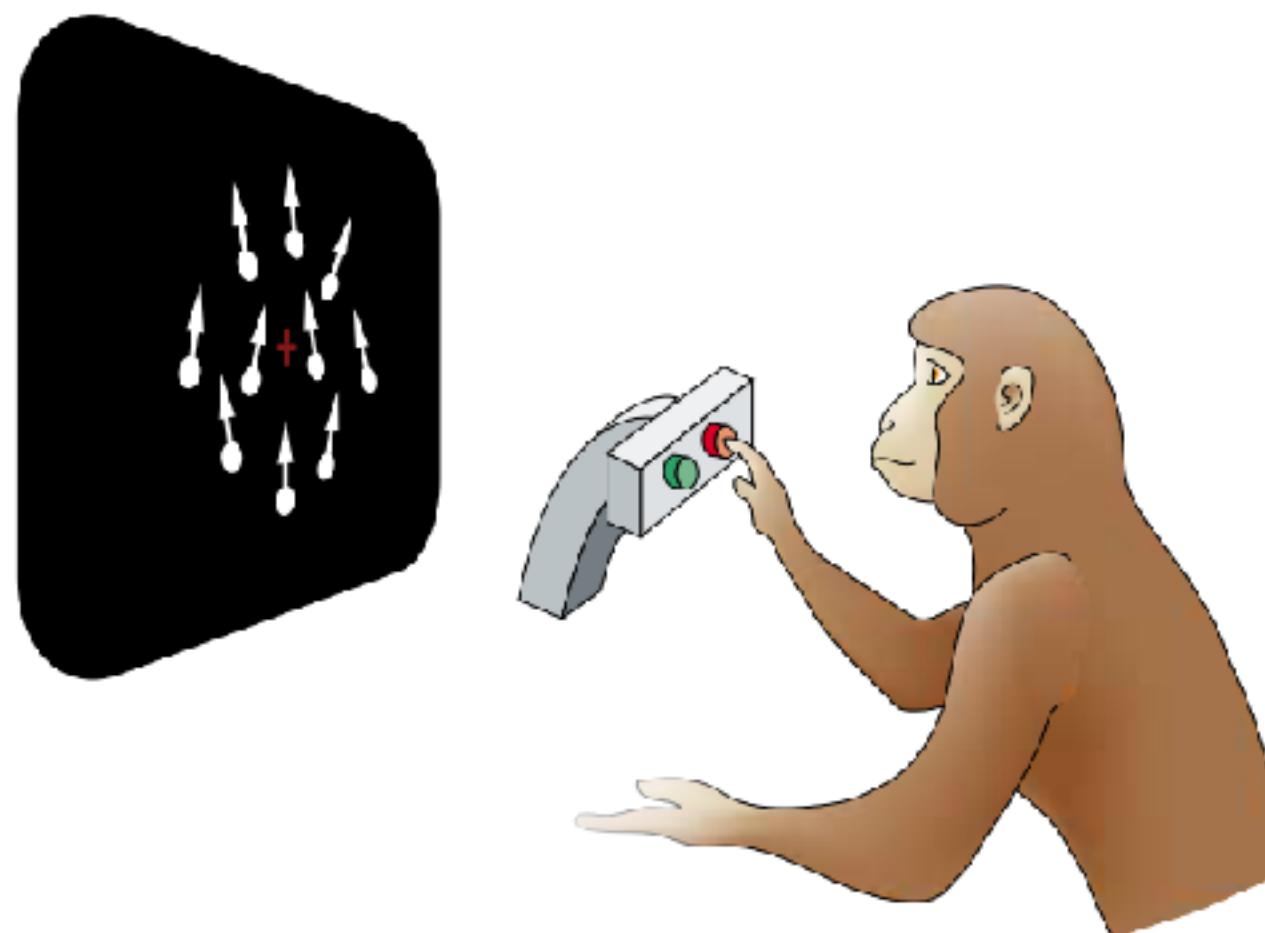
- » Fit model to experimental data
- » Simulate data with the fitted parameters
- » Fit the simulated data and see if the optimization procedure recovers the parameters used to generate the data



Questions so far?

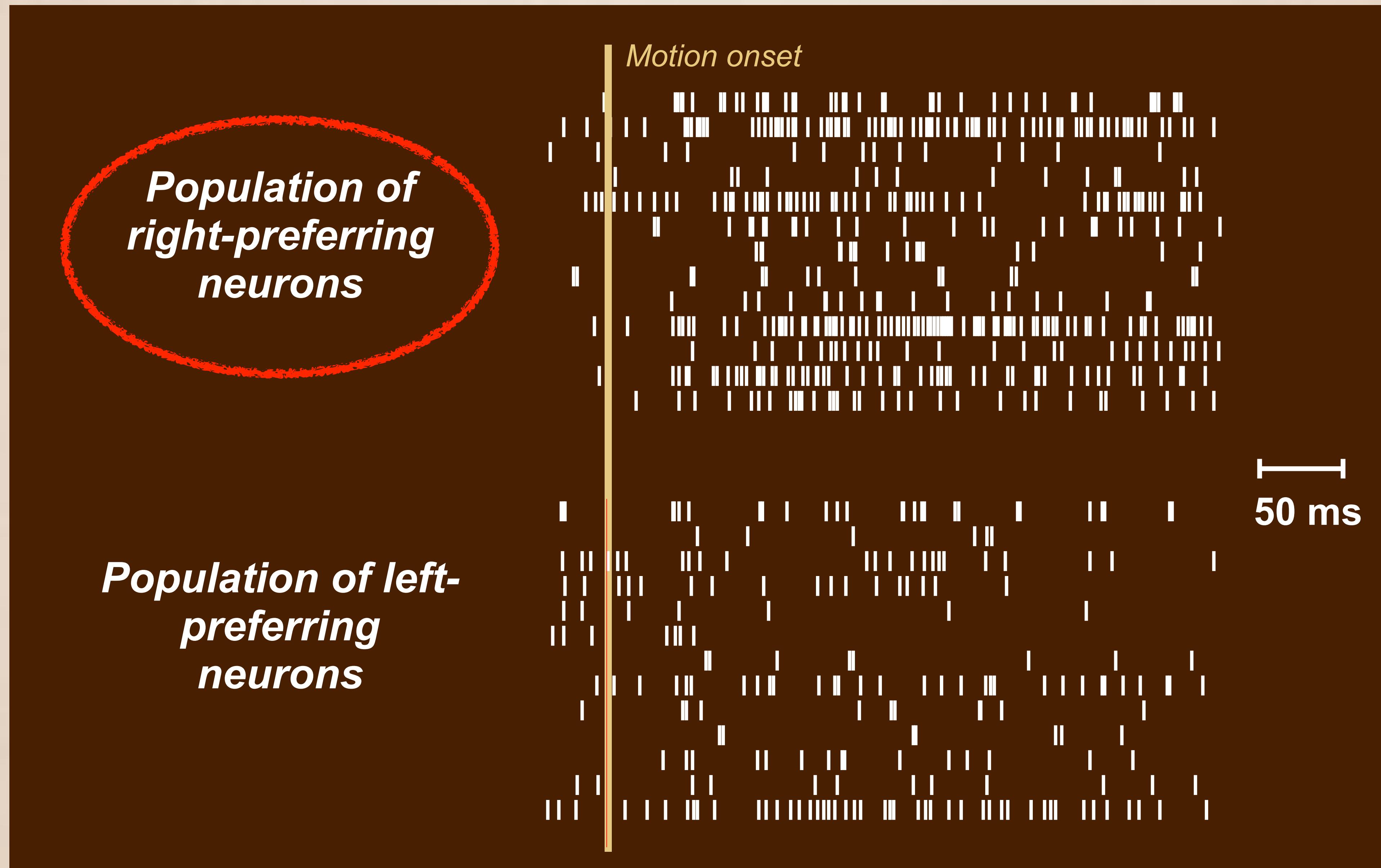
Neurophysiology

The RDM task in macaques



Albright, Movshon, Newsome, Britten, ...

Is the motion rightward or leftward?



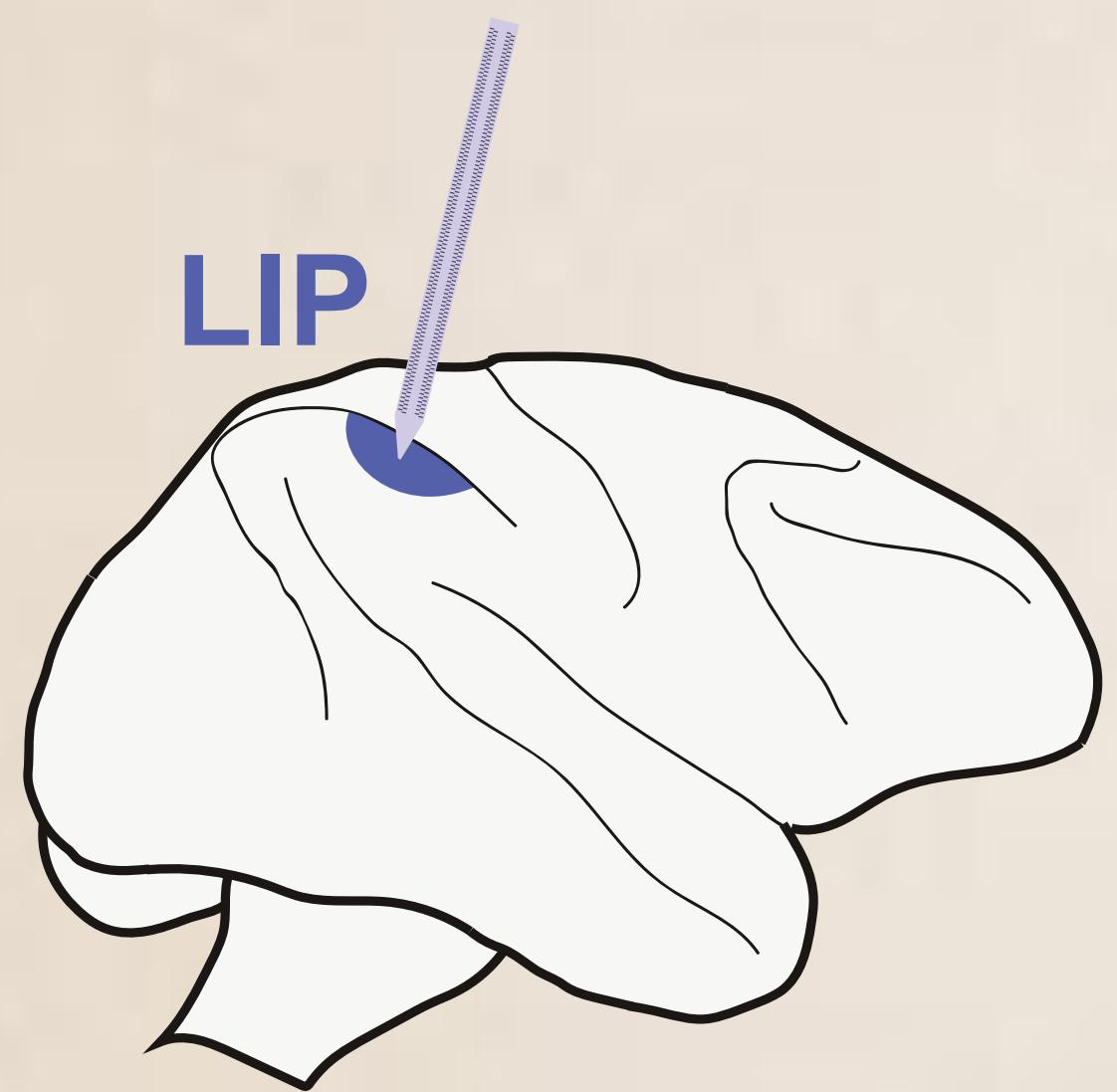
Is the motion rightward or leftward?



Decisions based on evidence streams

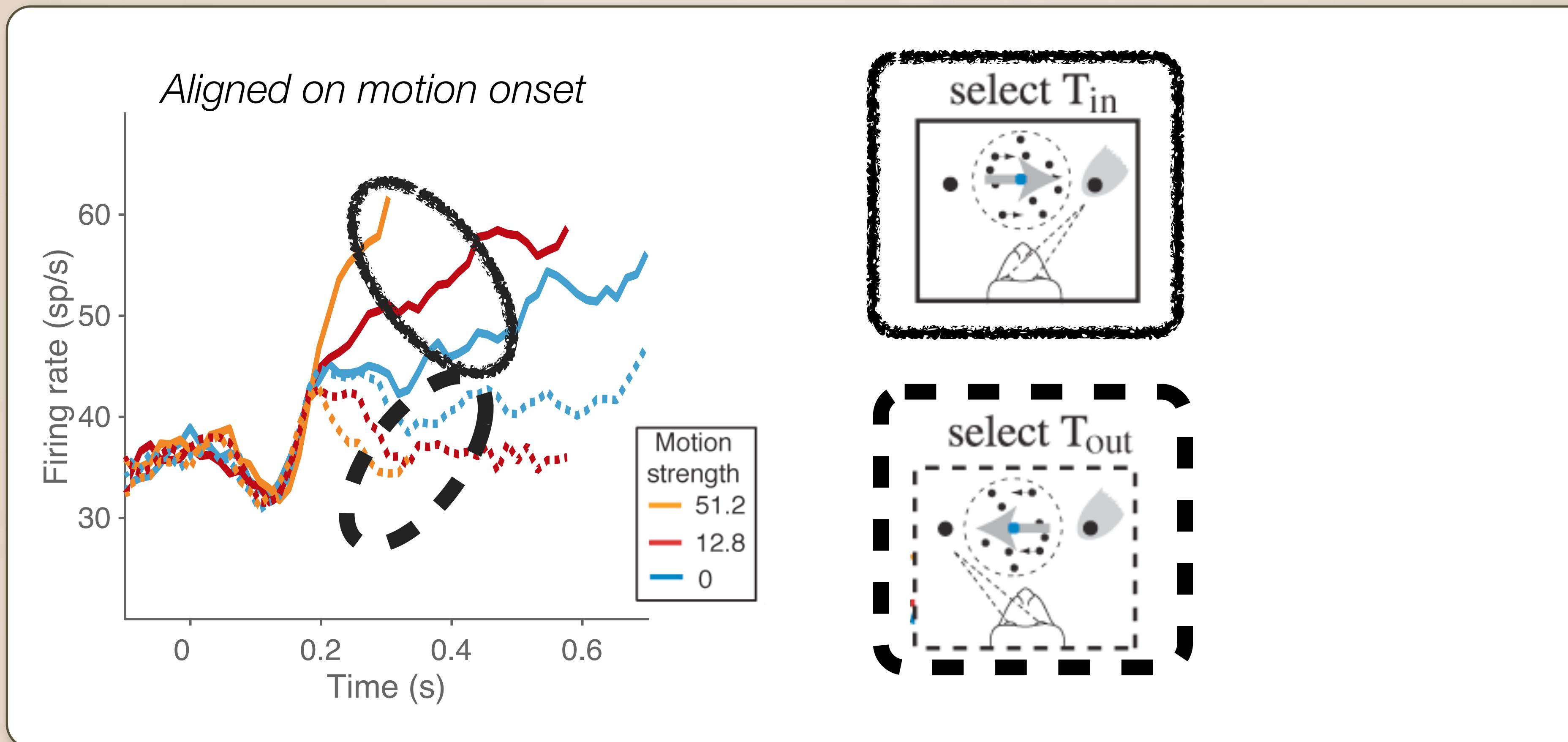
*Population of
right-preferring
neurons*

*Population of left-
preferring
neurons*

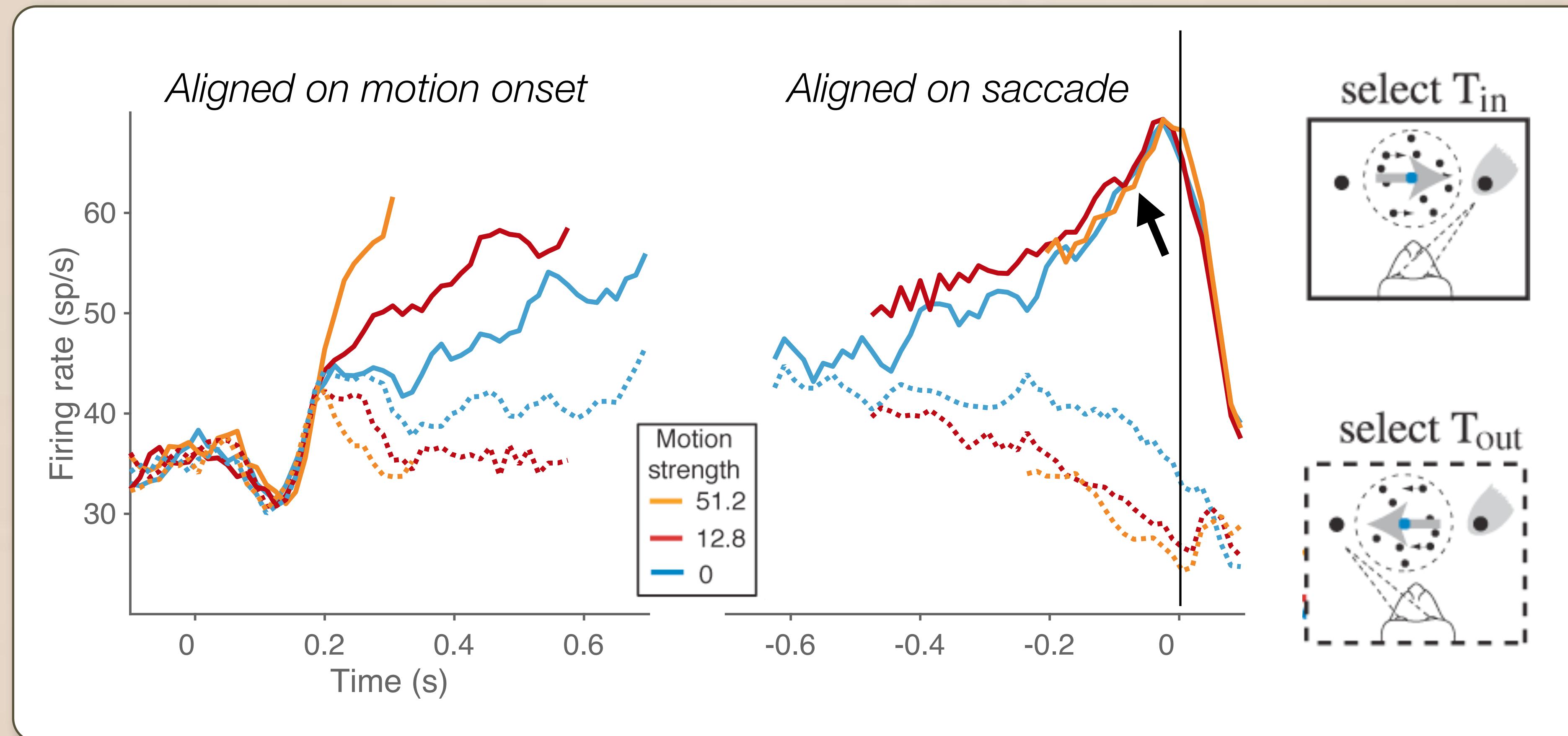


Mike Shadlen

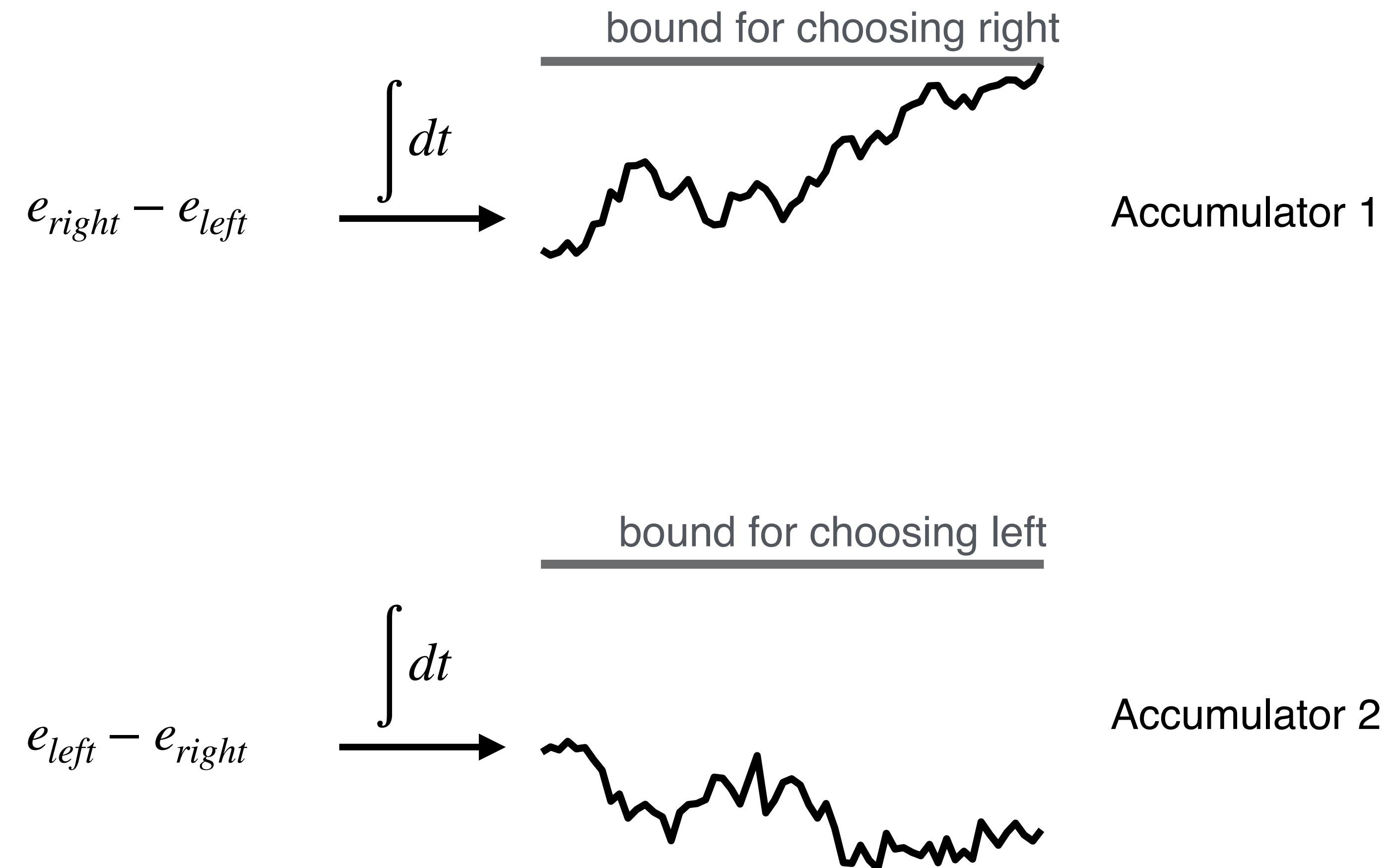
Average LIP activity in the random-dot motion task



Average LIP activity in the random-dot motion task



Competing races



e.g., Usher and McClelland 2001, Kiani & Shadlen 2014

Done, thank you!