The drift-diffusion model of decision making

Ariel Zylberberg & Max J. Pensack
Columbia University

Matlab code used in this tutorial:

https://github.com/arielzylberberg/CompPsychCourse_Zurich2023

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

Max: mjp2143@cumc.columbia.edu
Ariel: az2368@columbia.edu

Please introduce yourself

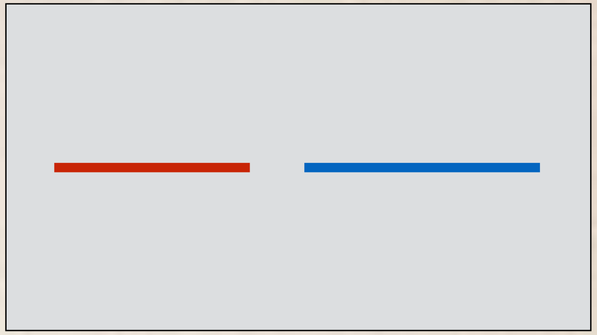
- Your name
- What you are working on or studying
- Why are you interested in learning about drift-diffusion models?

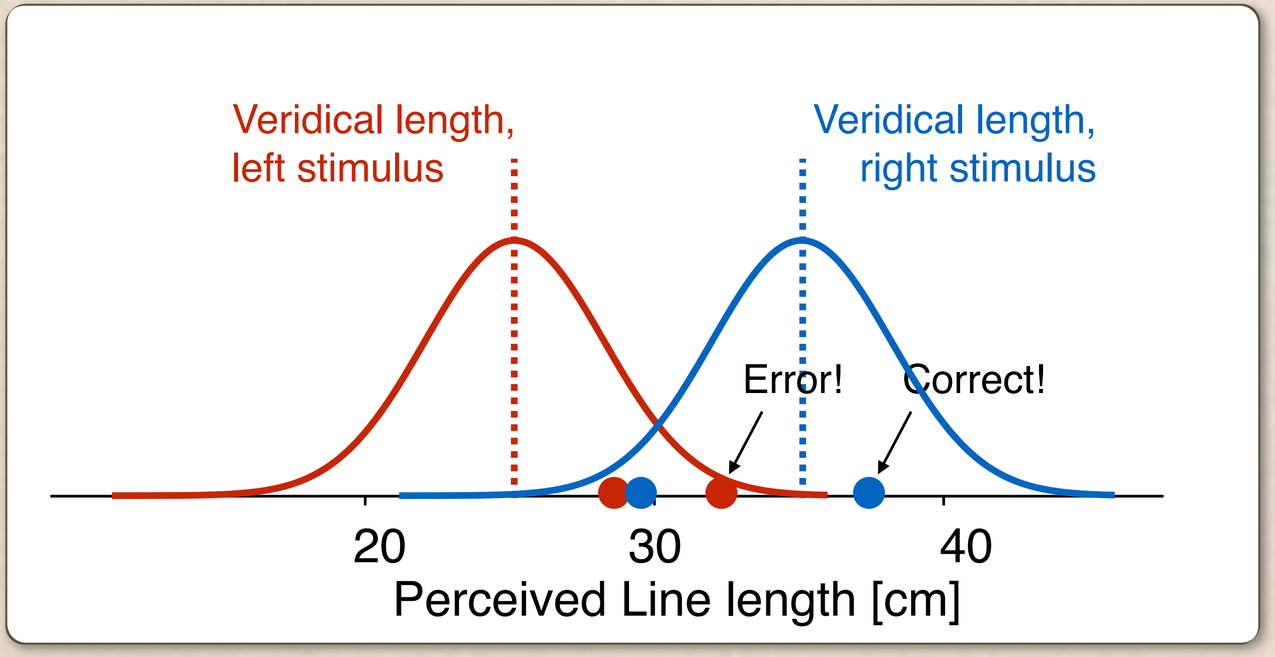
Drift-diffusion model of decision making

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

A key assumption:
Decisions are based on
signals corrupted by **noise**

Which line is longer?



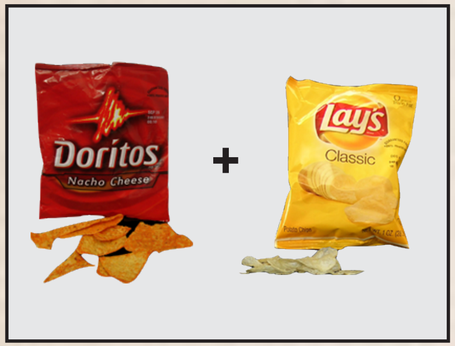


Signal-detection theory: one sample per decision

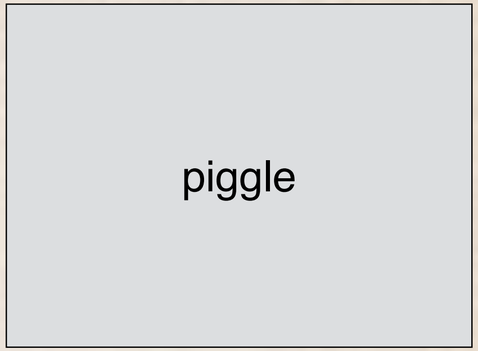
DDM: many samples per decision

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

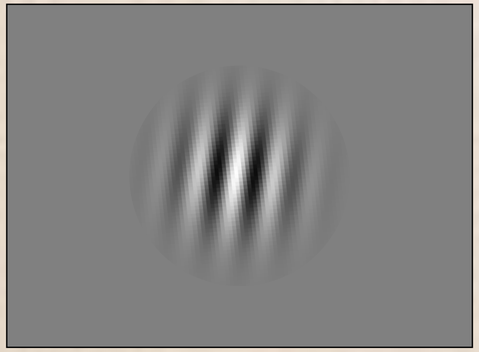
Which snack do you prefer?



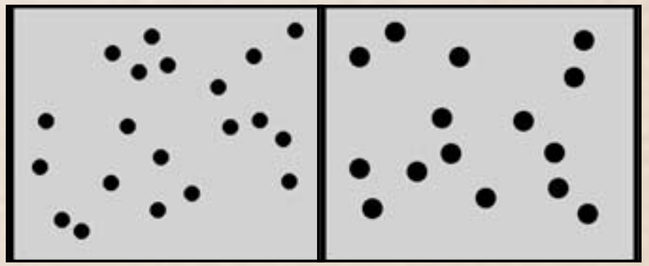
Word or non-word?



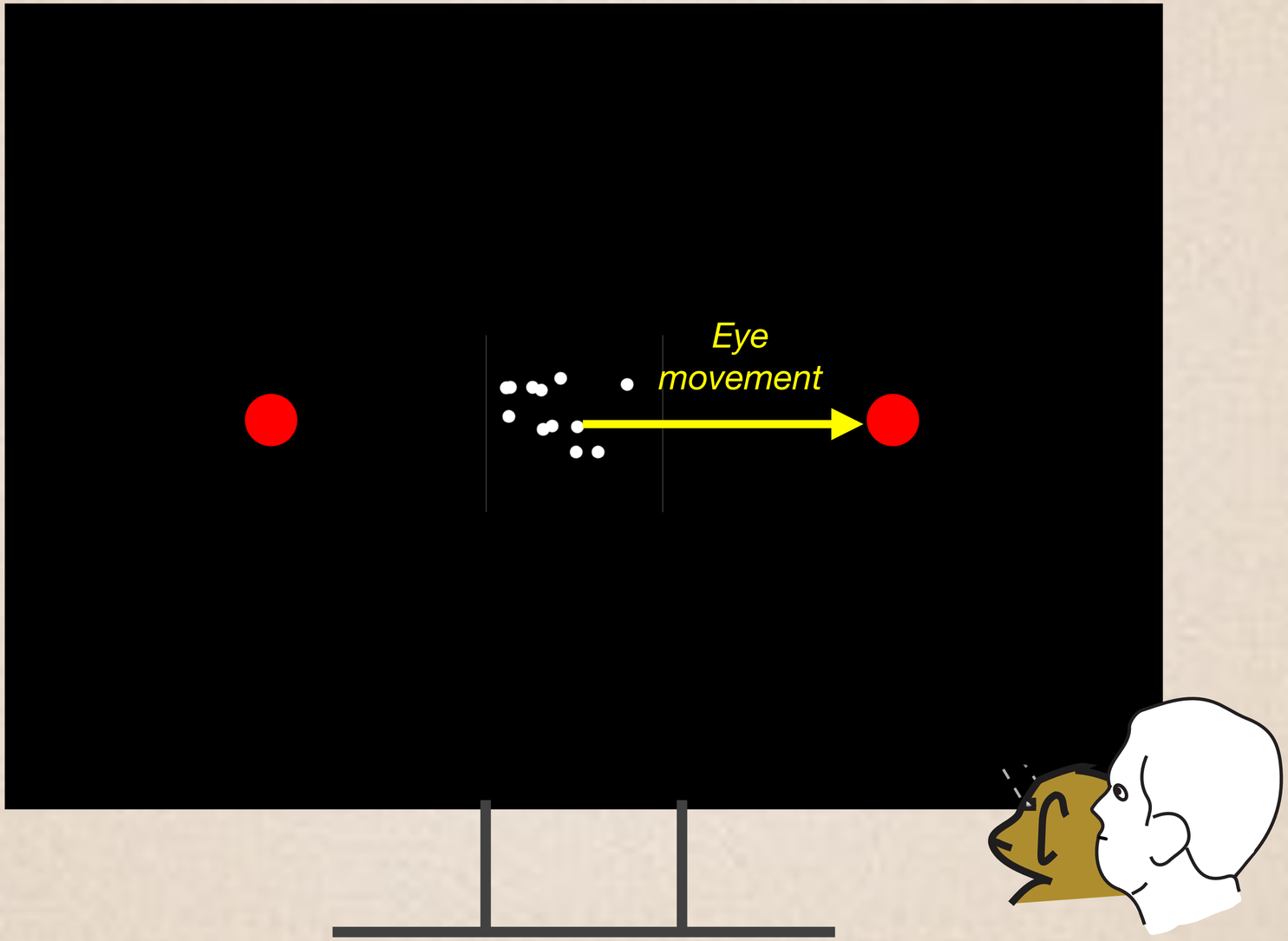
Is orientation clockwise or anti-clockwise?



Which display has more dots?



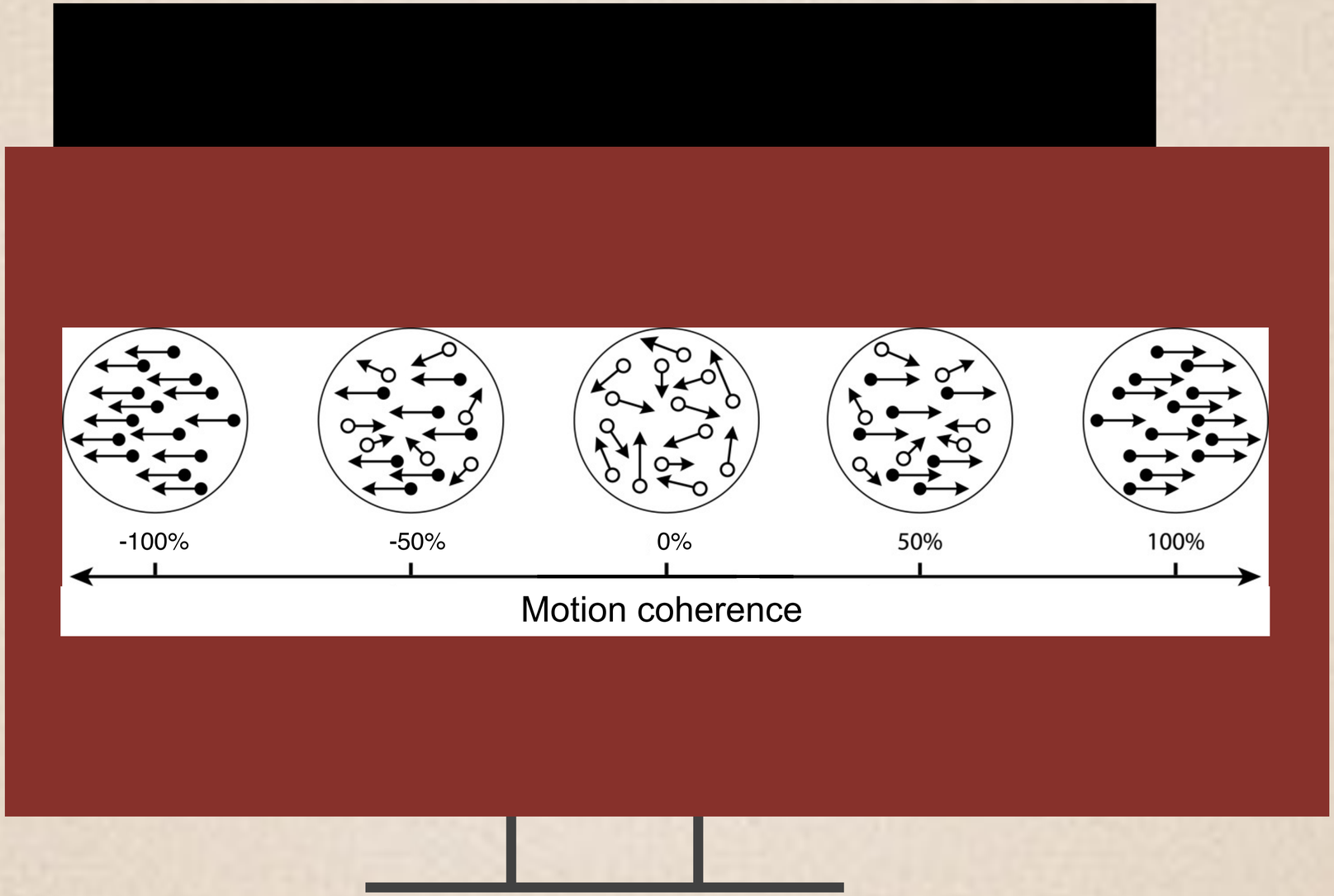
Random dot motion discrimination task



Britten, Newsome, Shadlen, …

Random dot motion discrimination task

(More difficult)

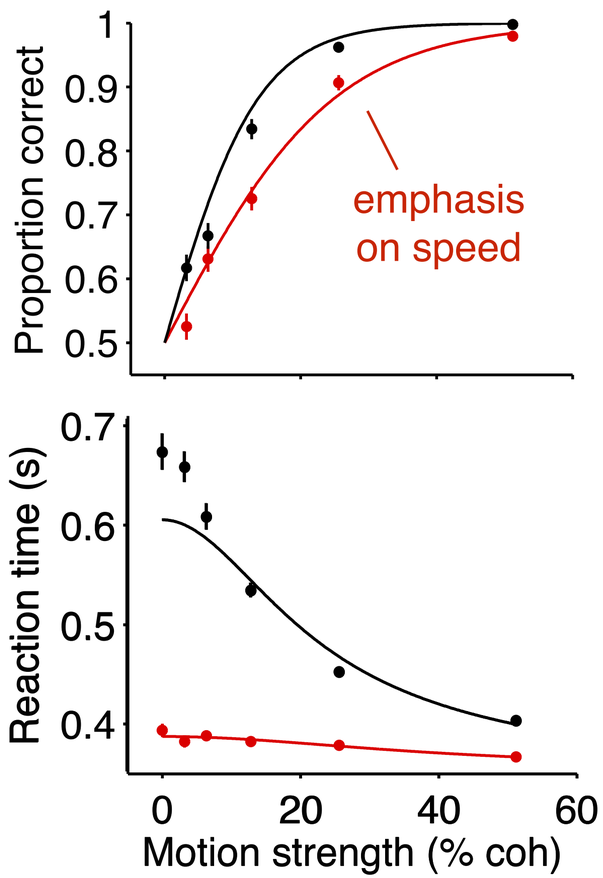


Britten, Newsome, Shadlen, …

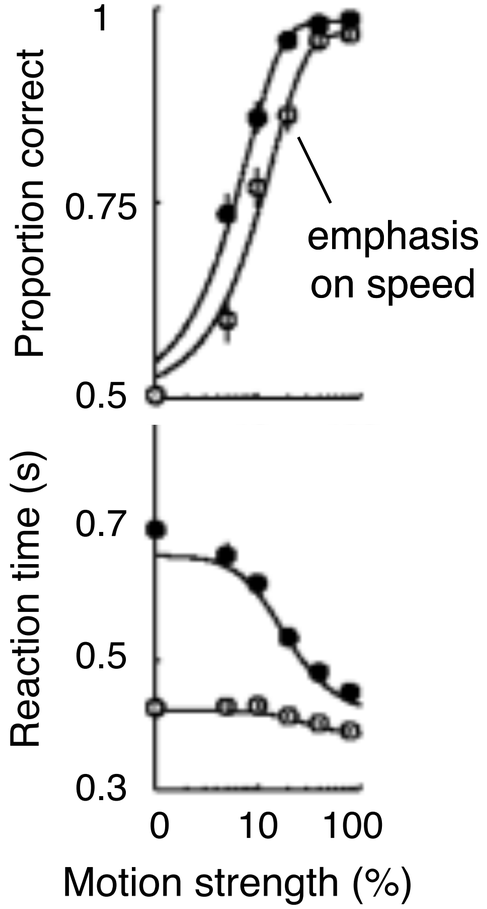
Speed-accuracy tradeoff

"Psychometric" function

"Chronometric" or response-time function



Hanks et al. eLife, 2014



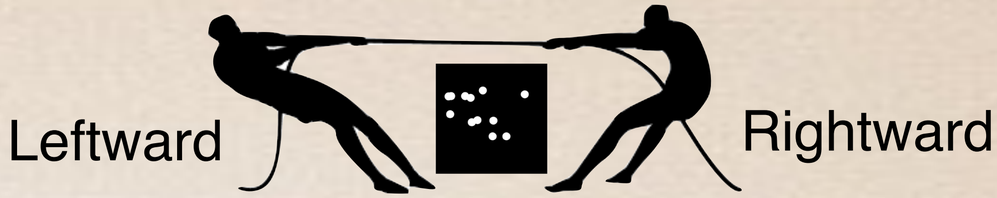
Mulder et al.
Att Percept Psych
2013

The DDM is not the only model used to account for this type of data

Nor the first…

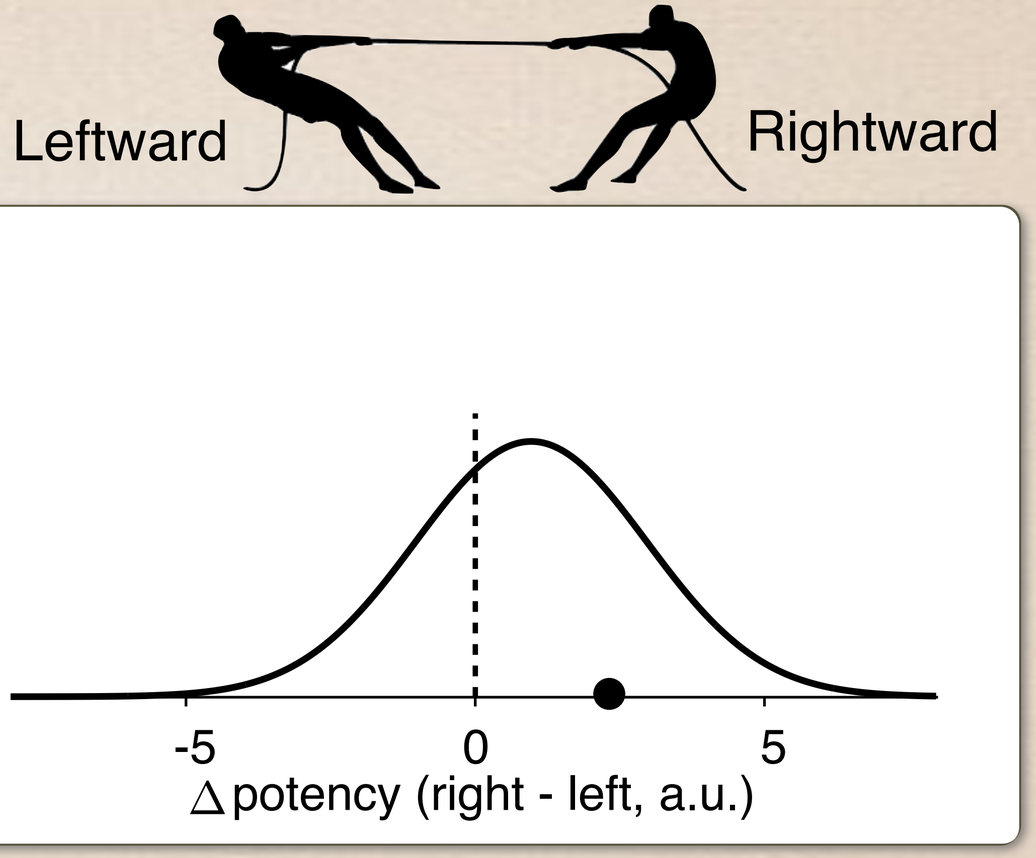
Opponent process theory of discrimination

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



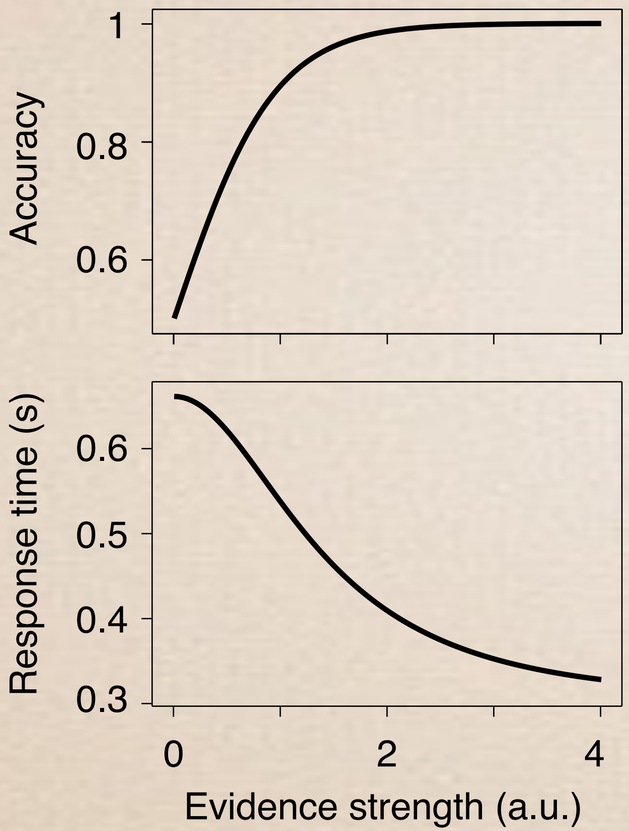
Opponent process theory of discrimination

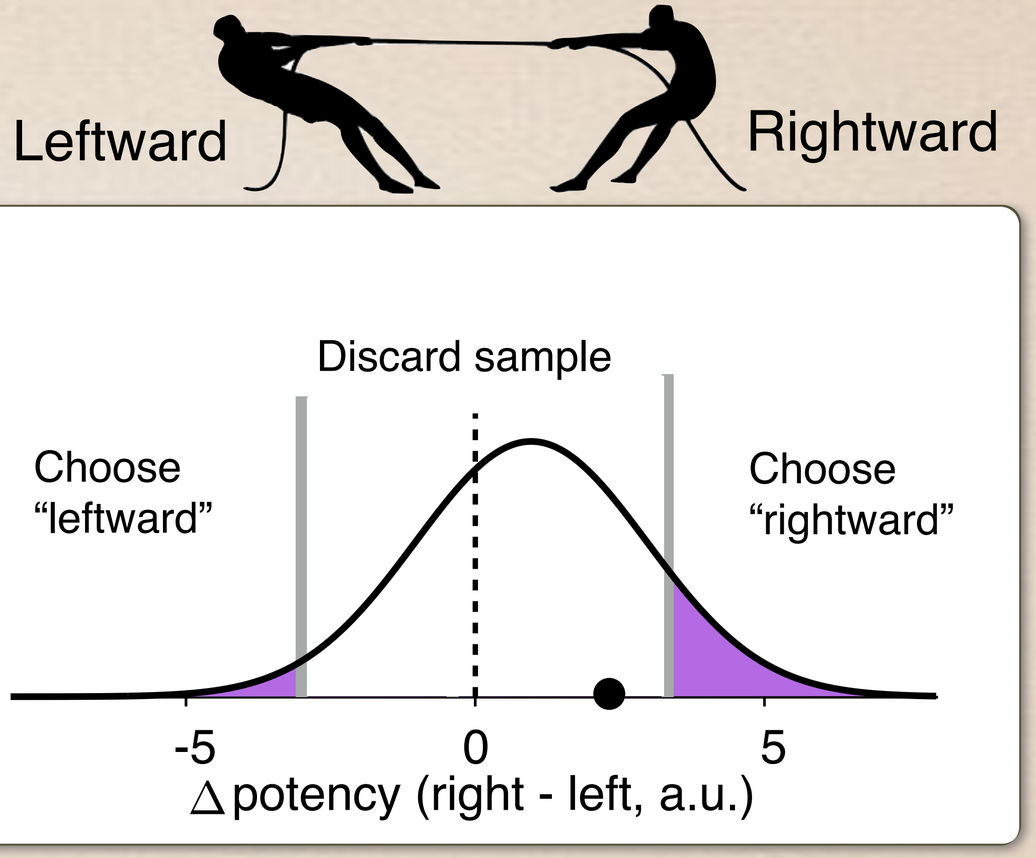
Cartwright and Festinger (1943), “A quantitative theory of decision”.



Opponent process theory of discrimination

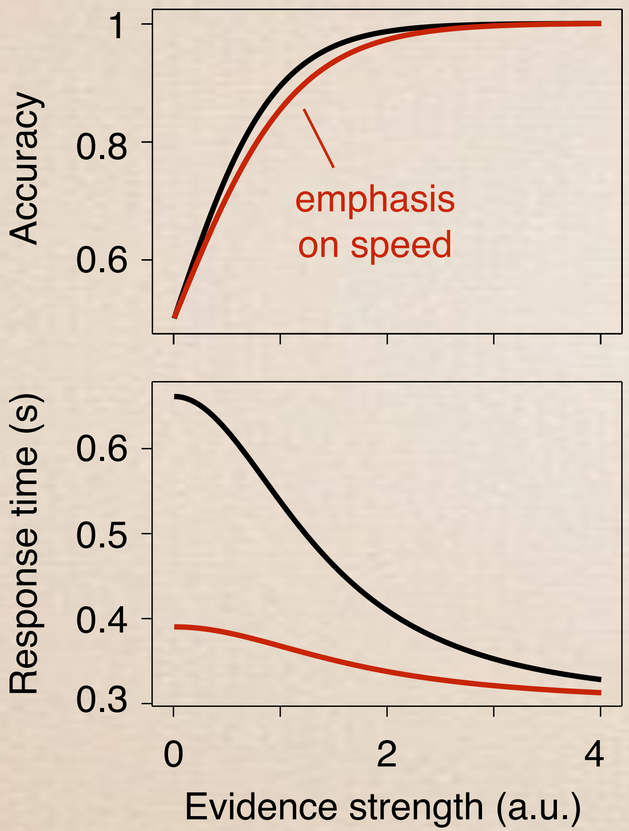
Cartwright and Festinger (1943), “A quantitative theory of decision”.

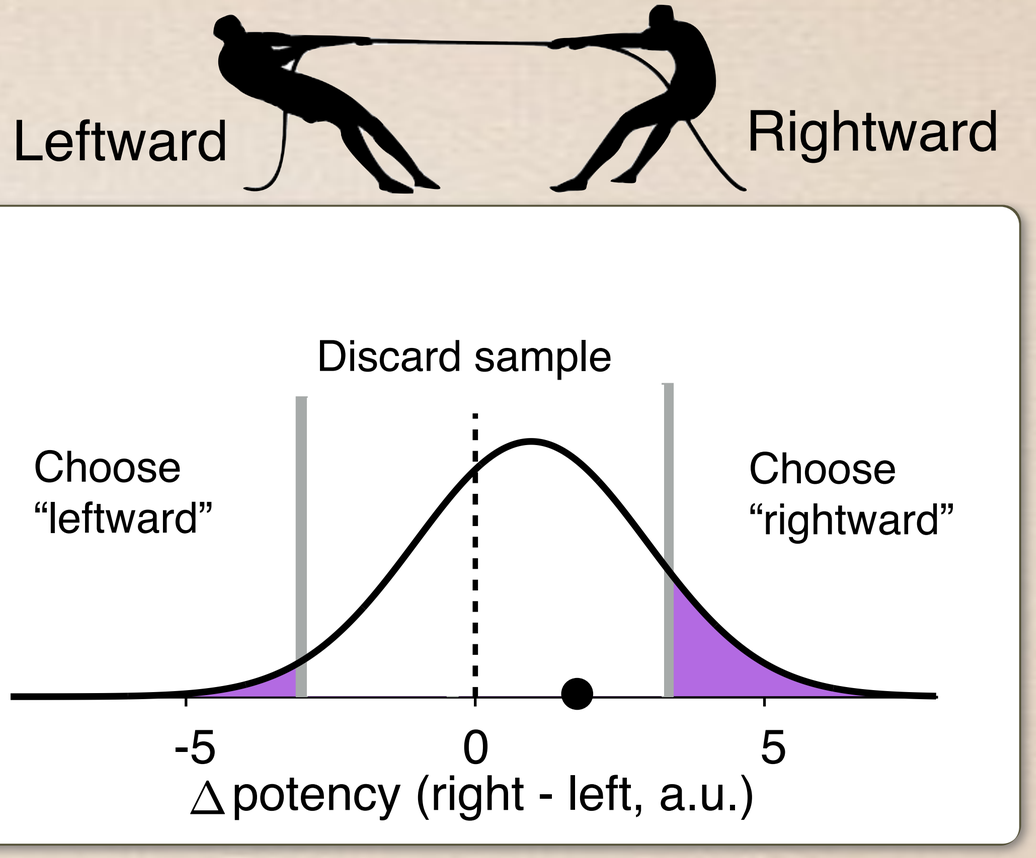




Opponent process theory of discrimination

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

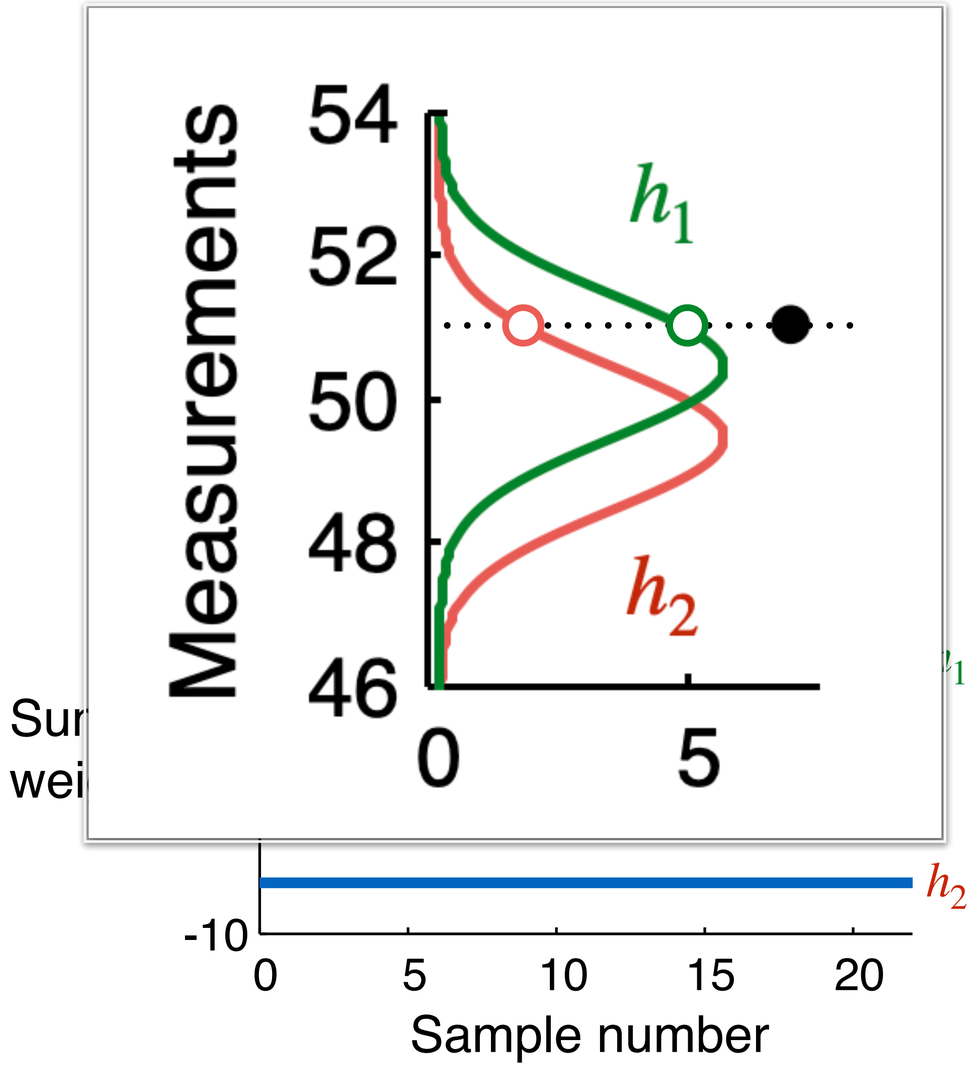




- Decisions may take a long time if high levels of accuracy are required.
- Decision rule is sub-optimal.

Sequential probability ratio test (SPRT)

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



Weight of evidence e *in favor of* h_1 *over* h_2 =

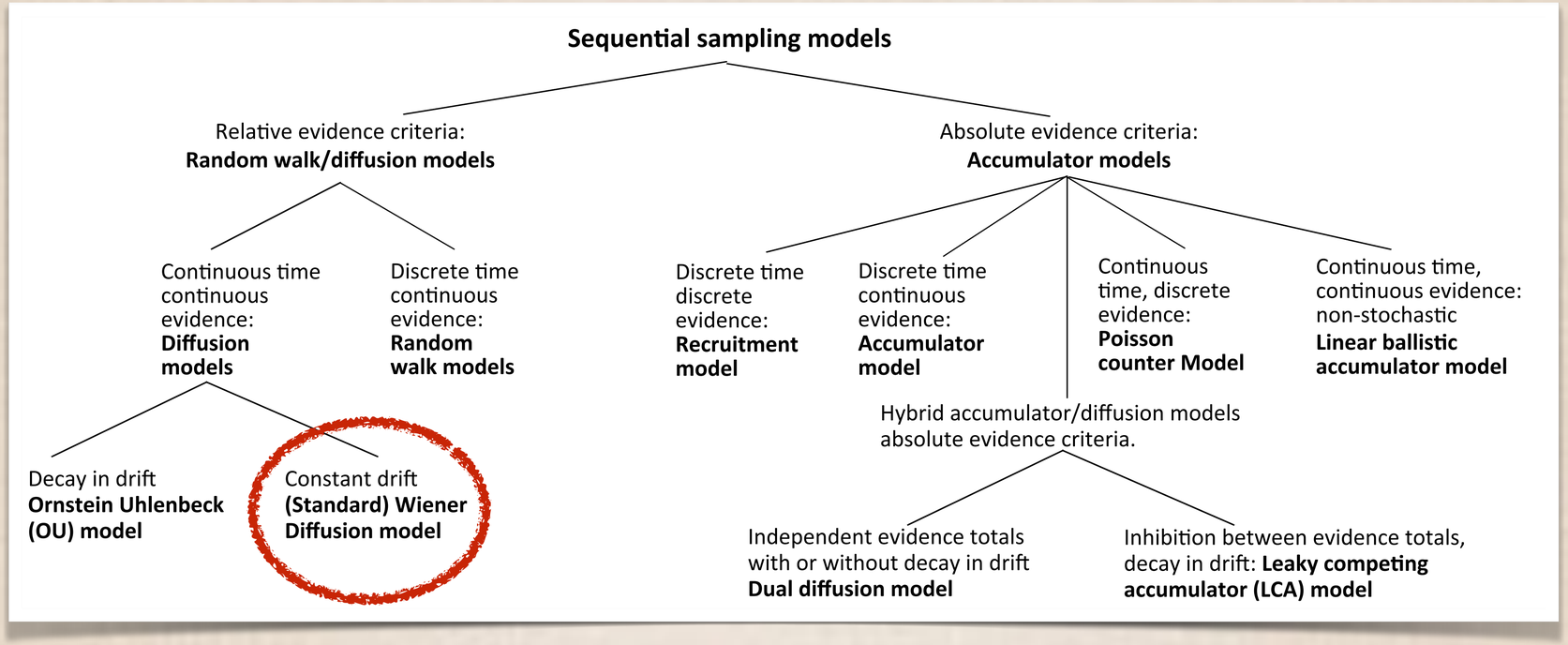
$$\text{Log}\left[\frac{\text{Probability of } e \text{ if } h_1 \text{ is true}}{\text{Probability of } e \text{ if } h_2 \text{ is true}}\right]$$

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

bound height

required accuracy

Many variants of evidence accumulation models for binary decisions



Ratcliff et al. TiCS 2016

- Time (continuous or discrete)
- Evidence samples (continuous or discrete)
- Perfect or “leaky” accumulation
- Other factors (number of accumulation processes, 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

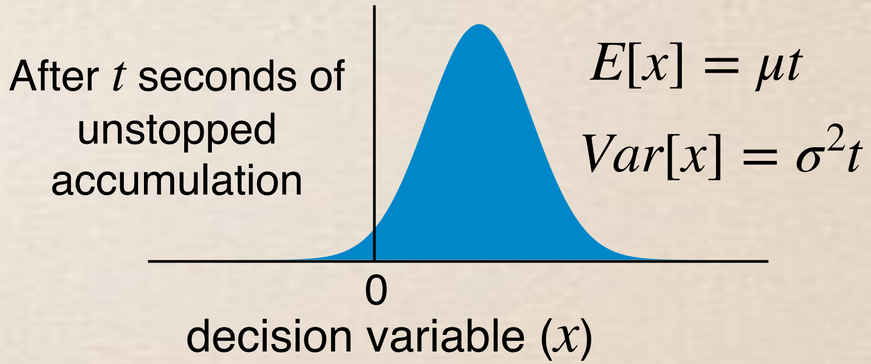
$$x_{i+1} = x_i + e_i$$

x_i: decision variable; e_i: evidence; i: time step

» Evidence samples e_i are independent and normally distributed

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

$\mu\Delta t$: drift coefficient; $\sigma^2\Delta t$: diffusion coefficient (time step)



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

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

$x(t)$: decision variable

Roger Ratcliff, 1978

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

$$x_{i+1} = x_i + e_i \qquad x_0 = 0$$

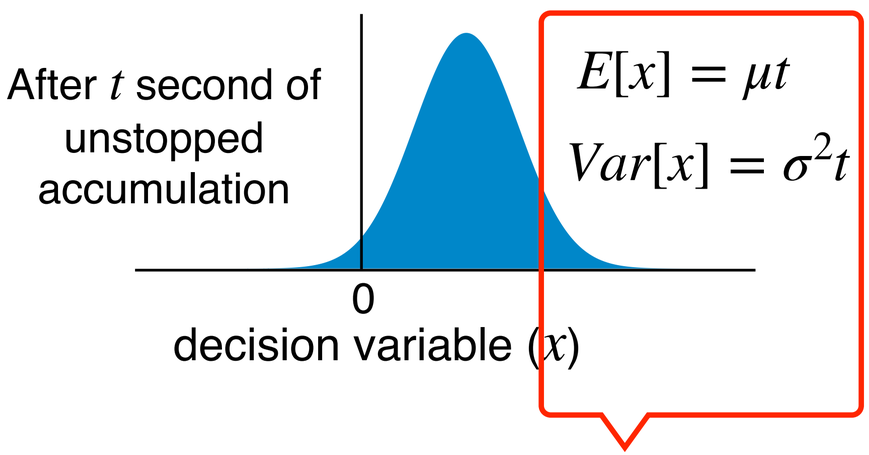
(x_i: decision variable; e_i: evidence; i: time step)

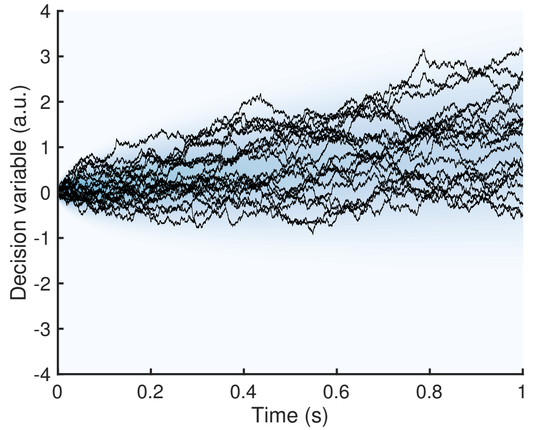
$$e_i \sim \mathcal{N}(\mu \Delta t, \sigma^2 \Delta t)$$

(Δt: time step; μ: drift coefficient; σ^2: diffusion coefficient)

Parameters:

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



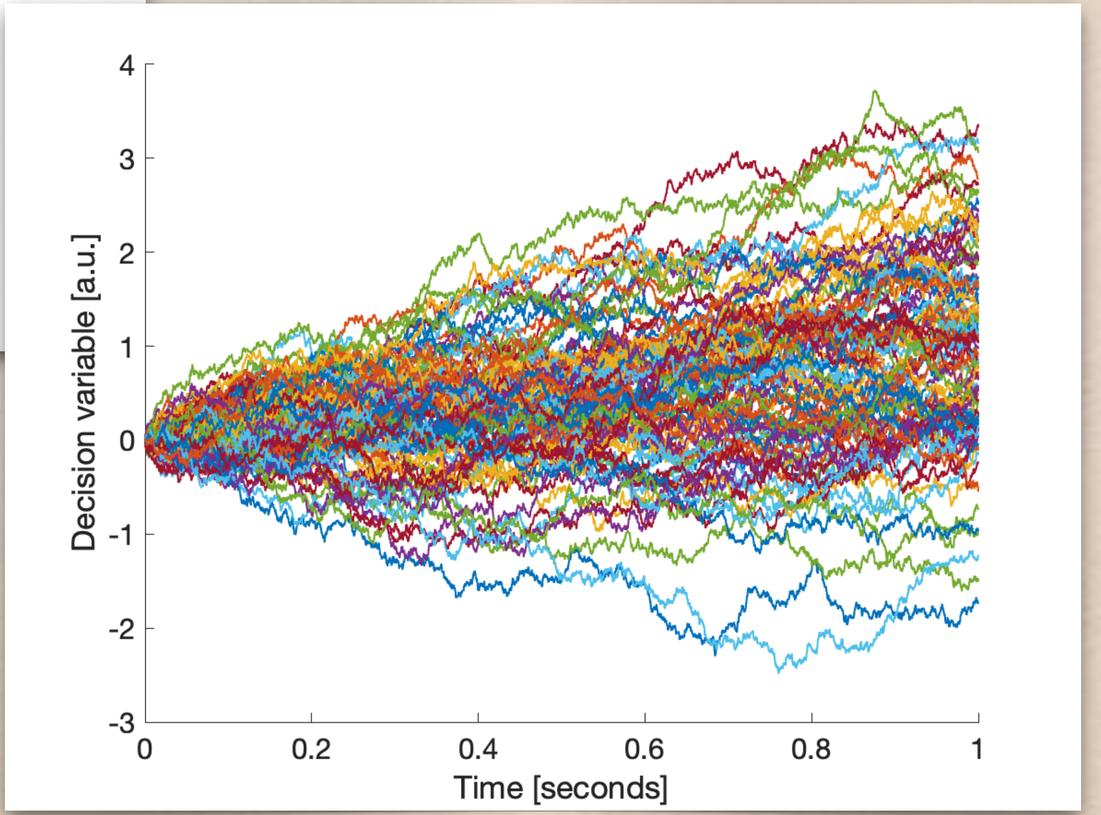


Key Matlab code:

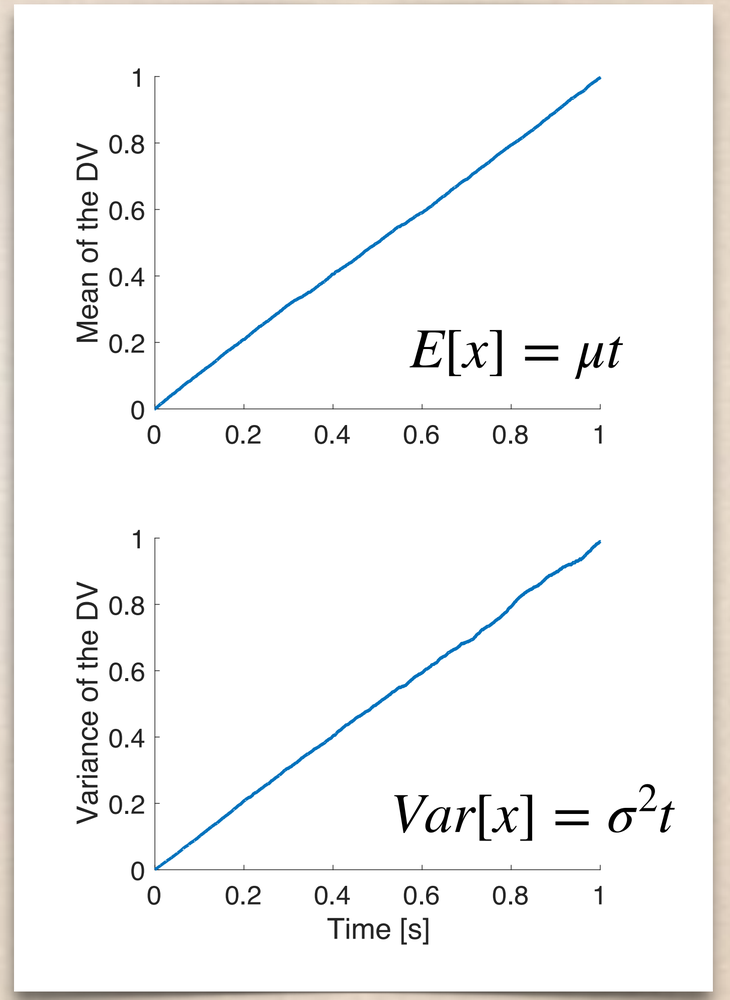
```
% momentary evidence
e = mu * dt + sigma * sqrt(dt) * randn(NumTrials, length(T));

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

Simulate diffusion paths and verify this relationship

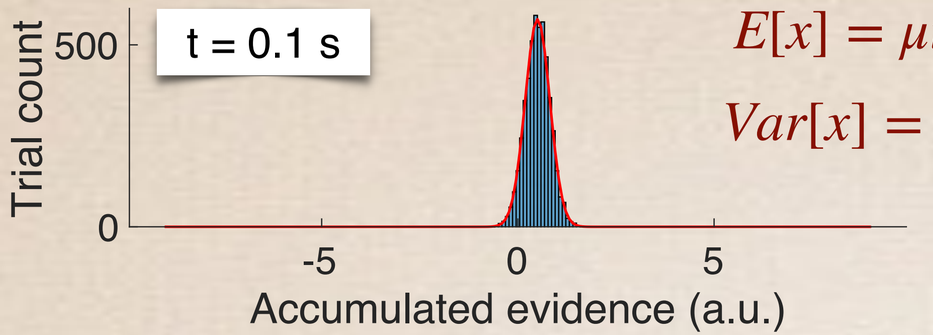
```
% parameters
NumTrials    = 5000;
MaxTime      = 1;
dt           = 0.0001;
sigma        = 1;
mu           = 1;
T = 0:dt:MaxTime;

% 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]');
```

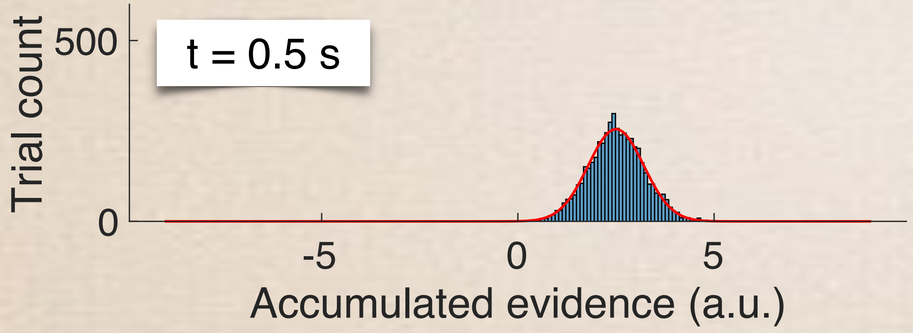


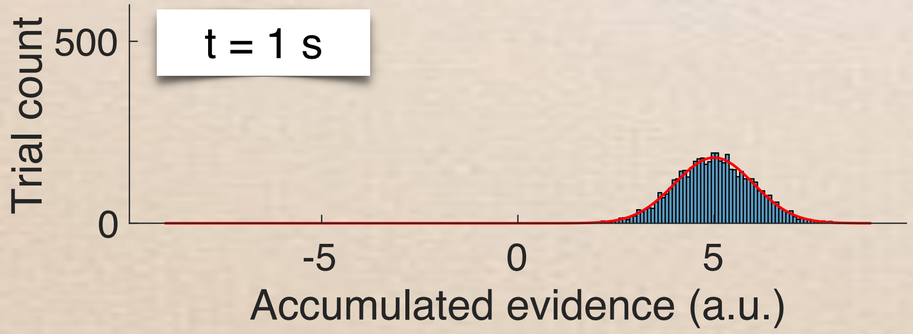
The state of the decision variable remains Gaussian over time



$$E[x] = \mu t$$

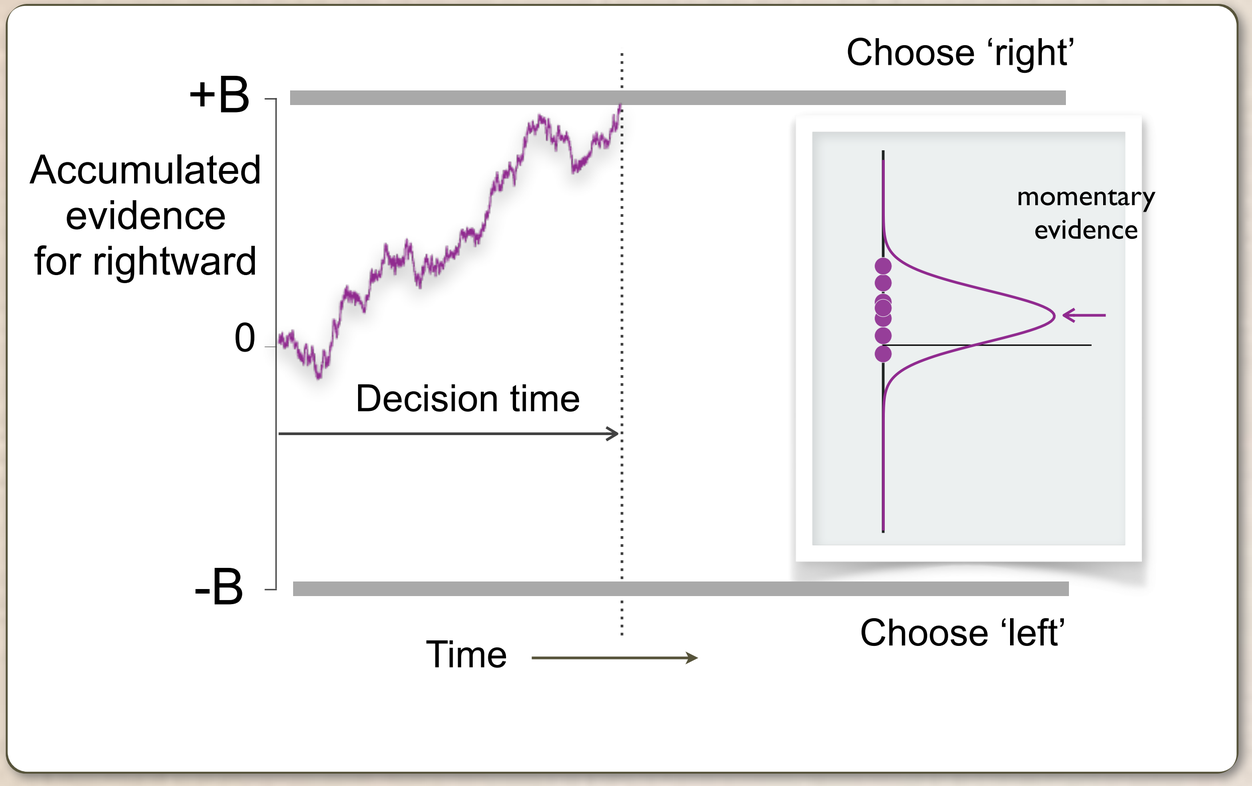
$$Var[x] = \sigma^2 t$$





No longer true when we add bounds…

The decision terminates when the accumulated evidence reaches a bound



Sample paths:

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

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

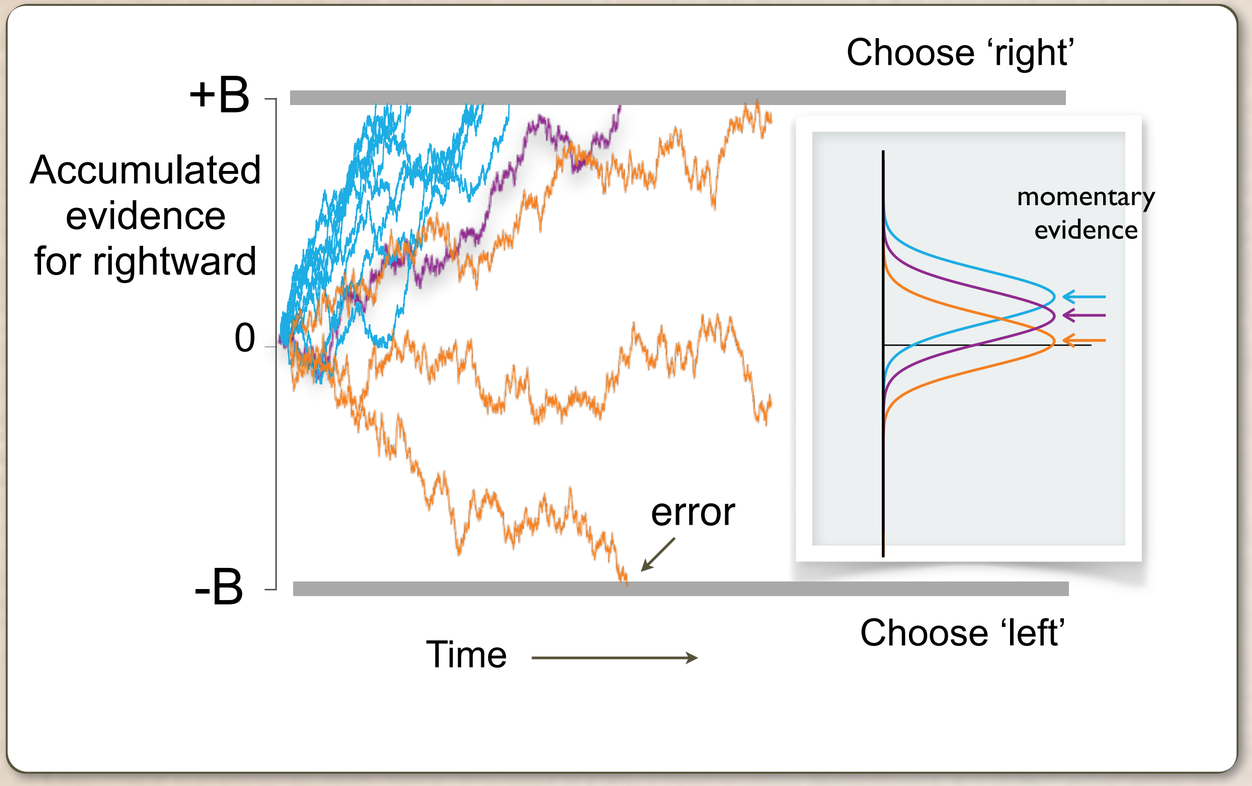
$$x_0 = 0$$

Motion task:

$\mu = \kappa \cdot$ 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:

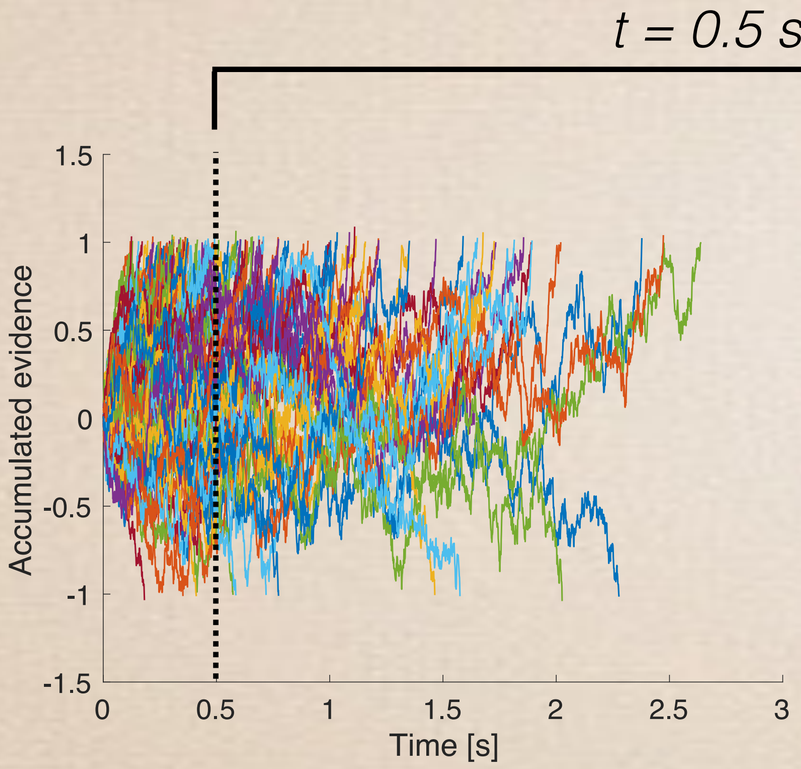
$\mu = \kappa \cdot$ Motion coherence

$|\mu|$ Increases with motion strength

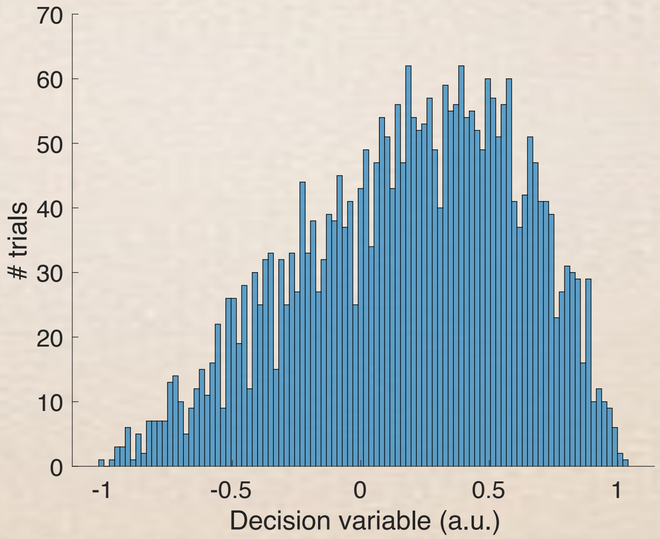
Add bounds to the simulated trajectories

Code in "02_Simulate_diffusion_with_bounds/main.m"

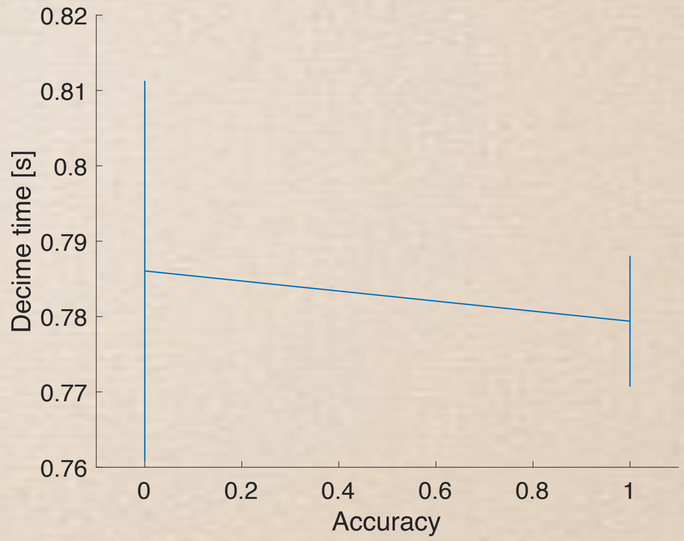
» Added symmetric bounds at: B = ± 1



Some trajectories



The state of the DV is no longer Gaussian

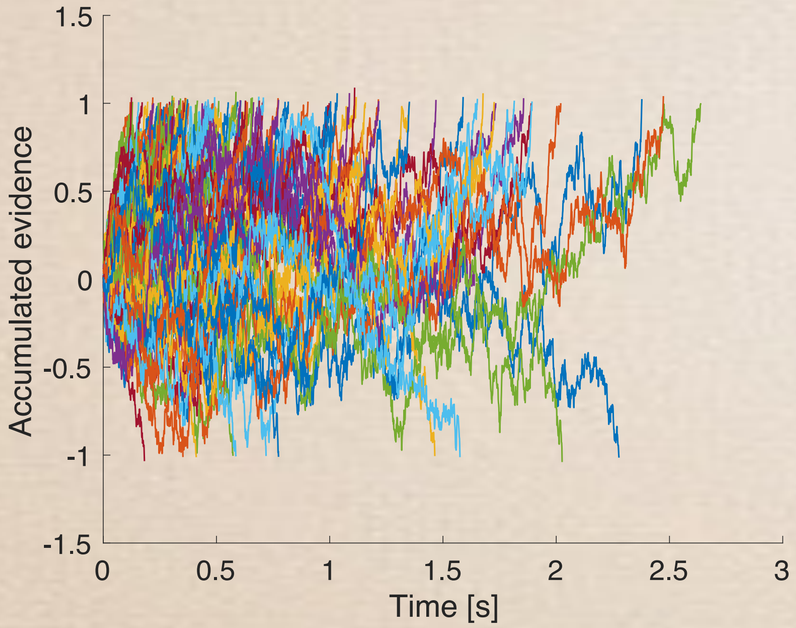


Correct and incorrect decisions have the same decision time

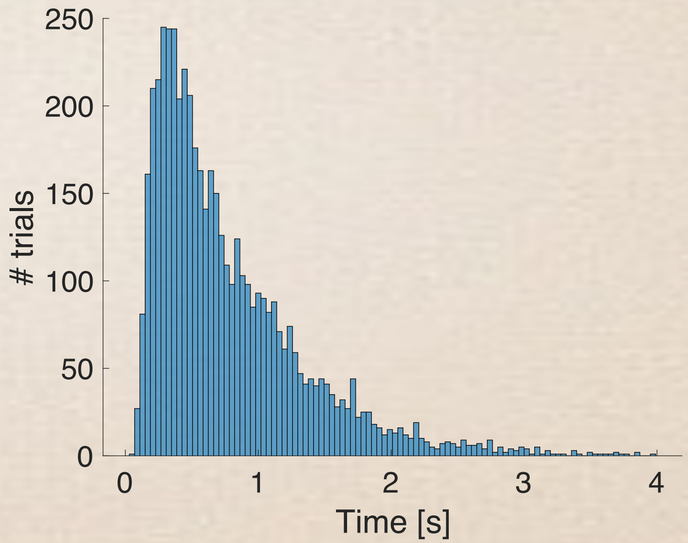
Add bounds to the simulated trajectories

Code in "02_Simulate_diffusion_with_drift/main.m"

» Added symmetric bounds at: B = ± 1

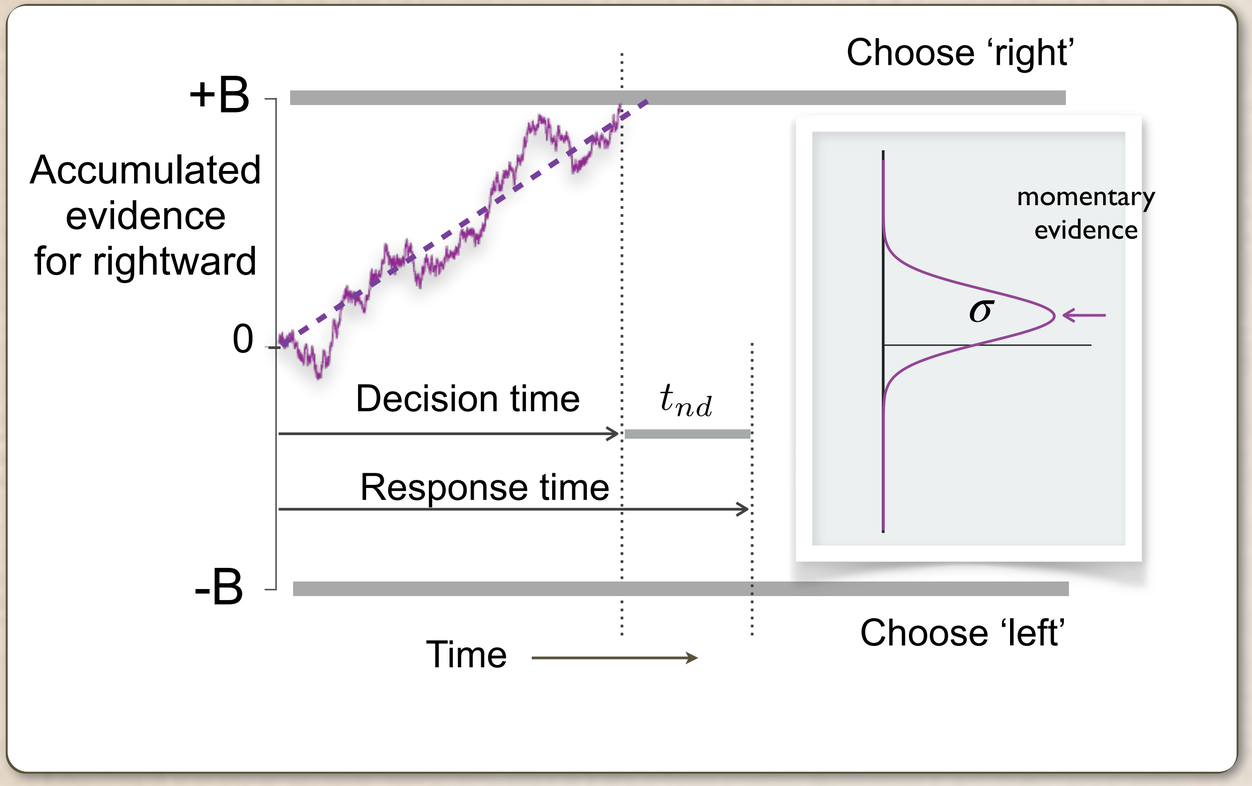


Some trajectories



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

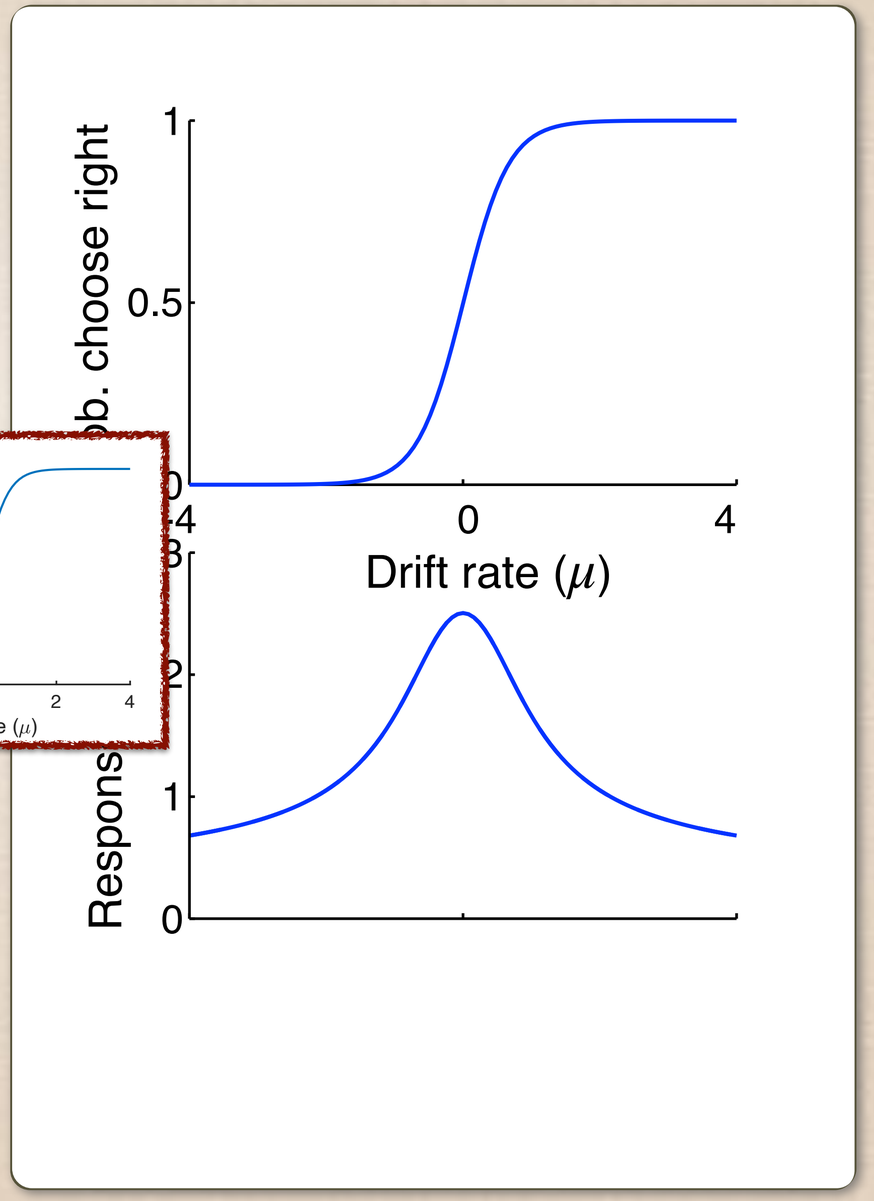
Probability of terminating at the upper bound:

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

Average response time:

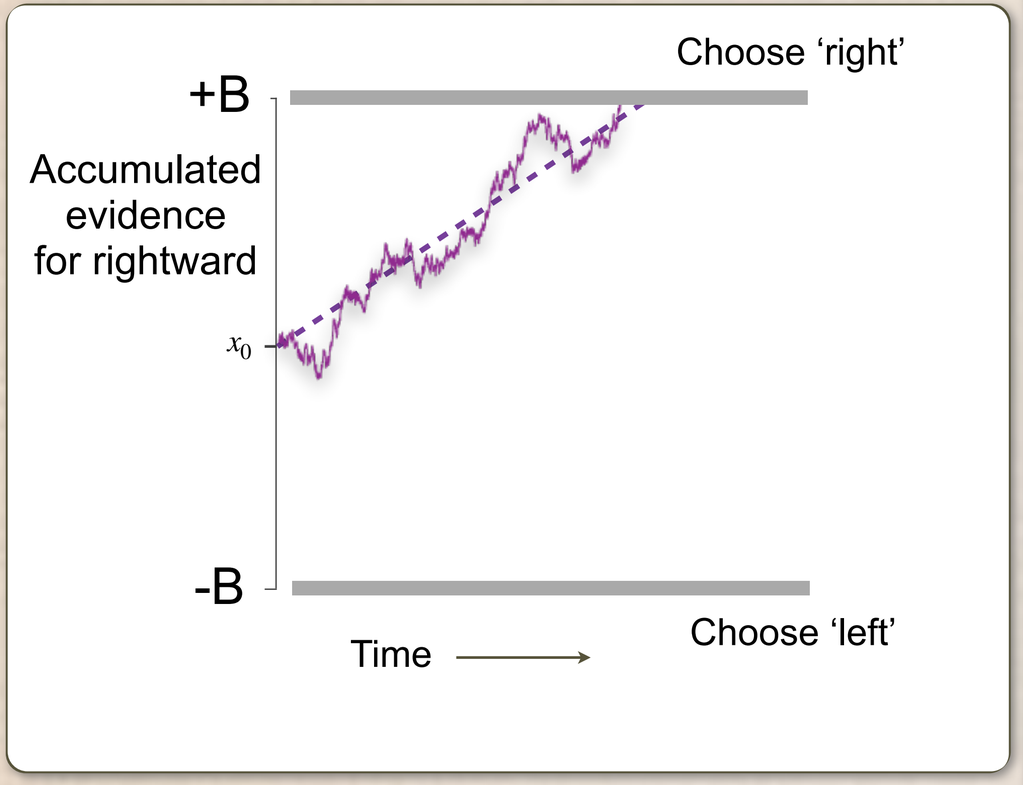
$$T = \frac{B}{\mu} \tanh(\mu B) + t_{nd}$$

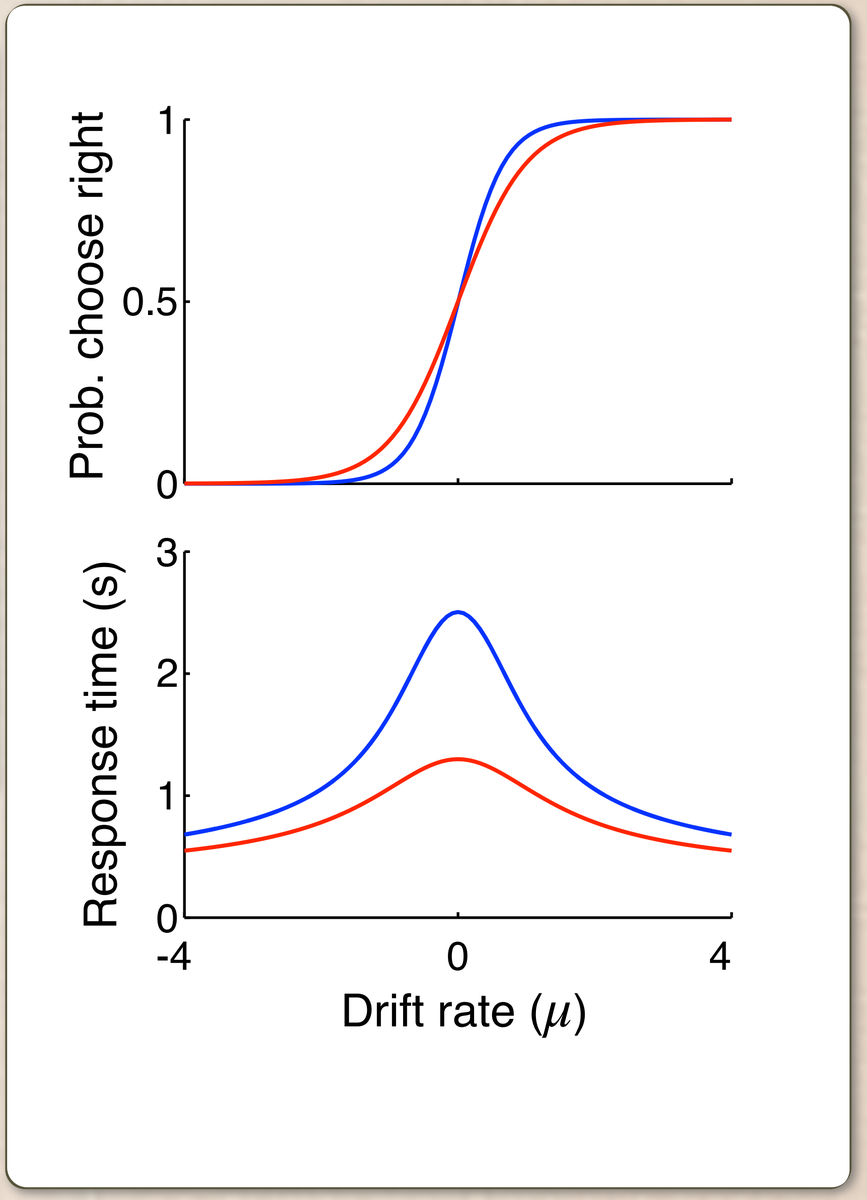
When $\mu = 0$: $T = B^2 + t_{nd}$



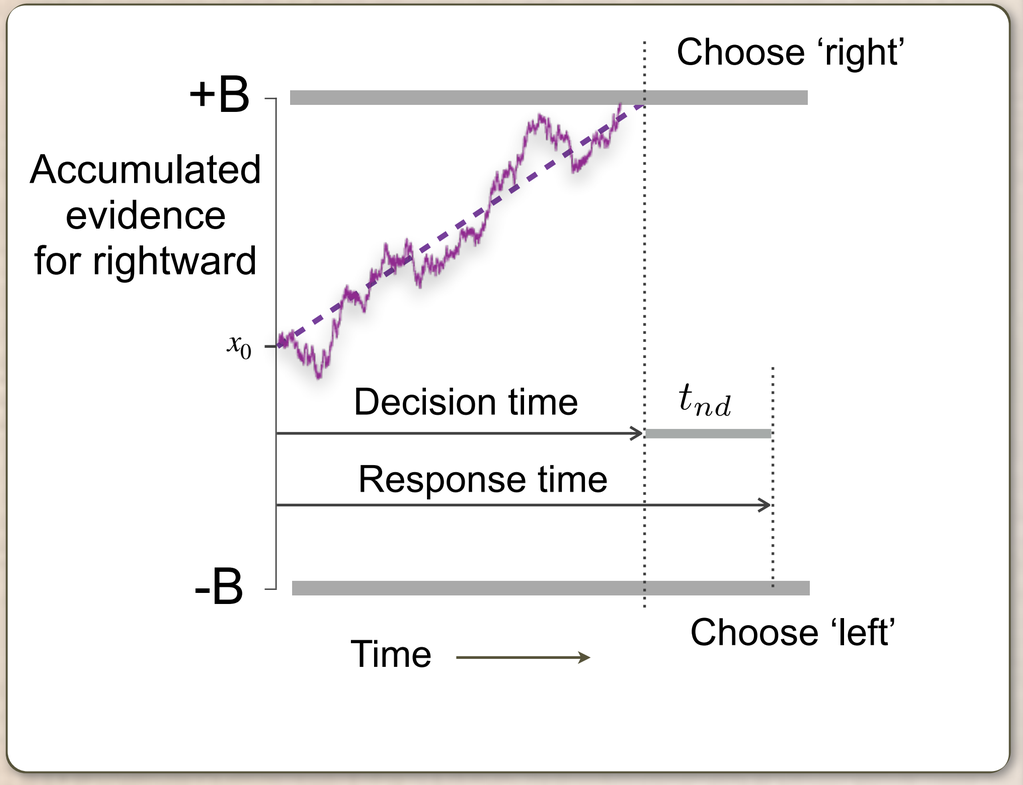
Shadlen et al. 2006, “The Speed and Accuracy of a Simple Perceptual Decision”

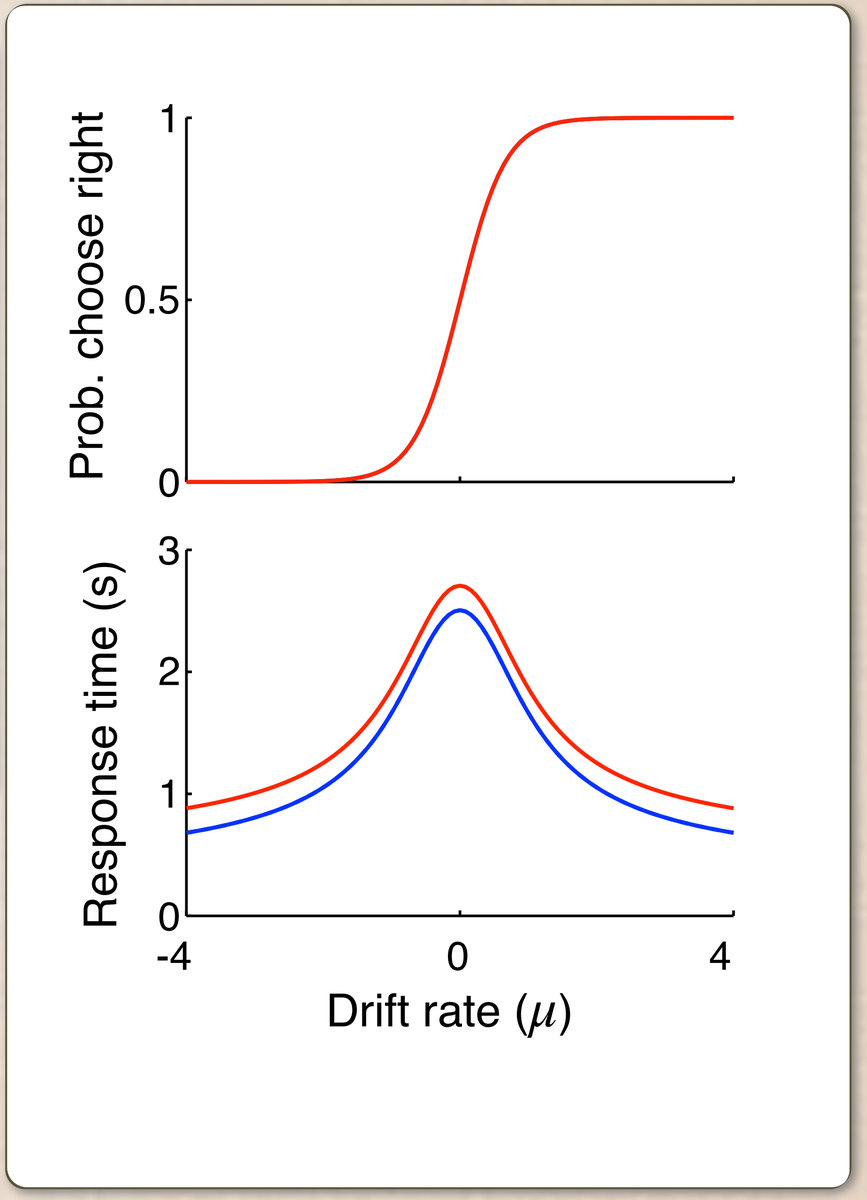
Change in bound height





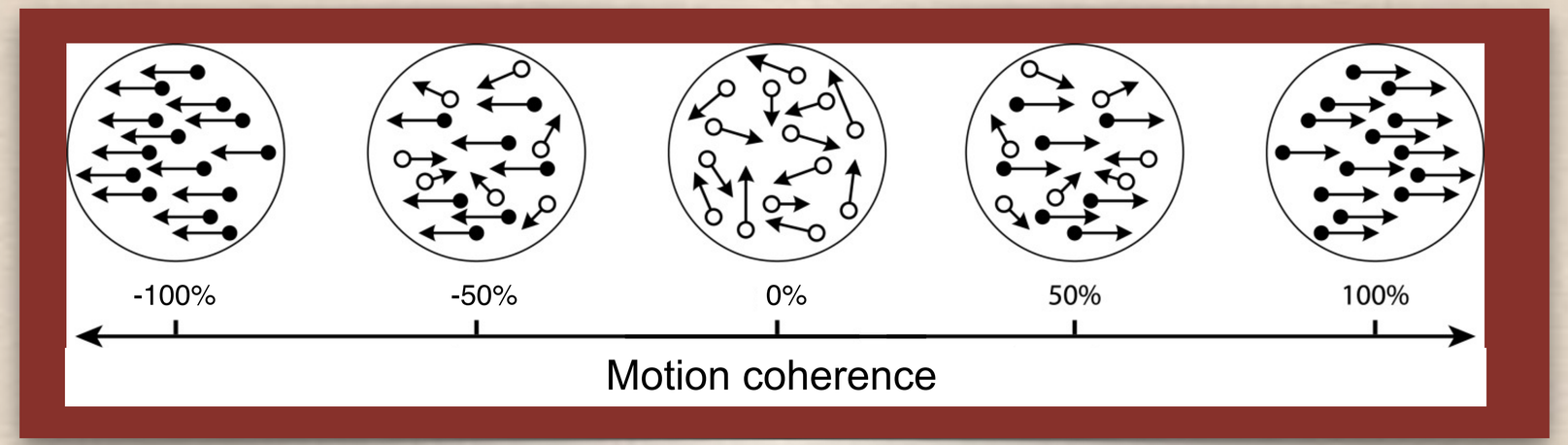
Change in non-decision time





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

$$\mu = f(C)$$

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

Motion coherence

signal-to-noise

$$\mu = \kappa \times C$$

Linear

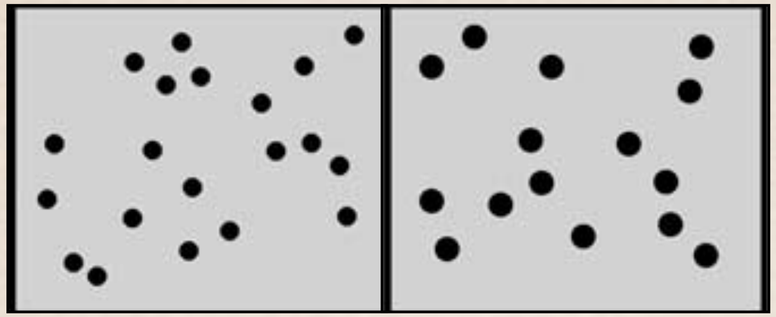
$$\sigma^2 = 1$$

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

time step

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

drift coefficient

diffusion coefficient

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

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

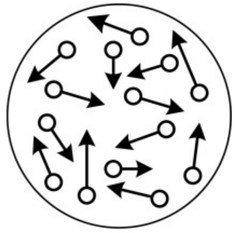
Modeling choice biases

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

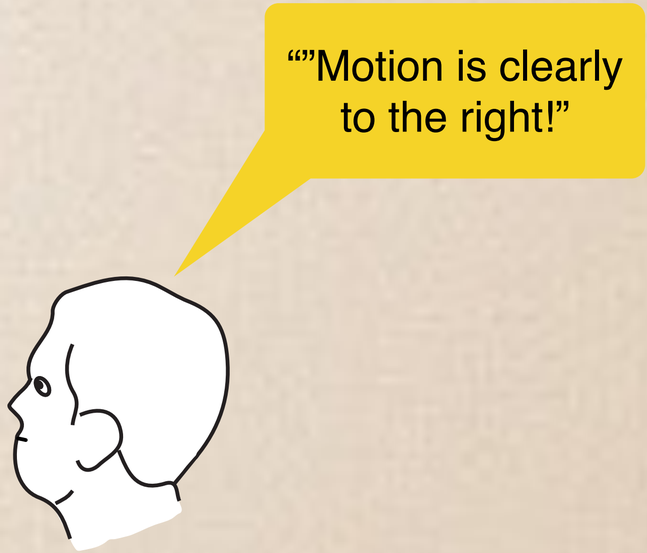
Probability of terminating at the upper bound:

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

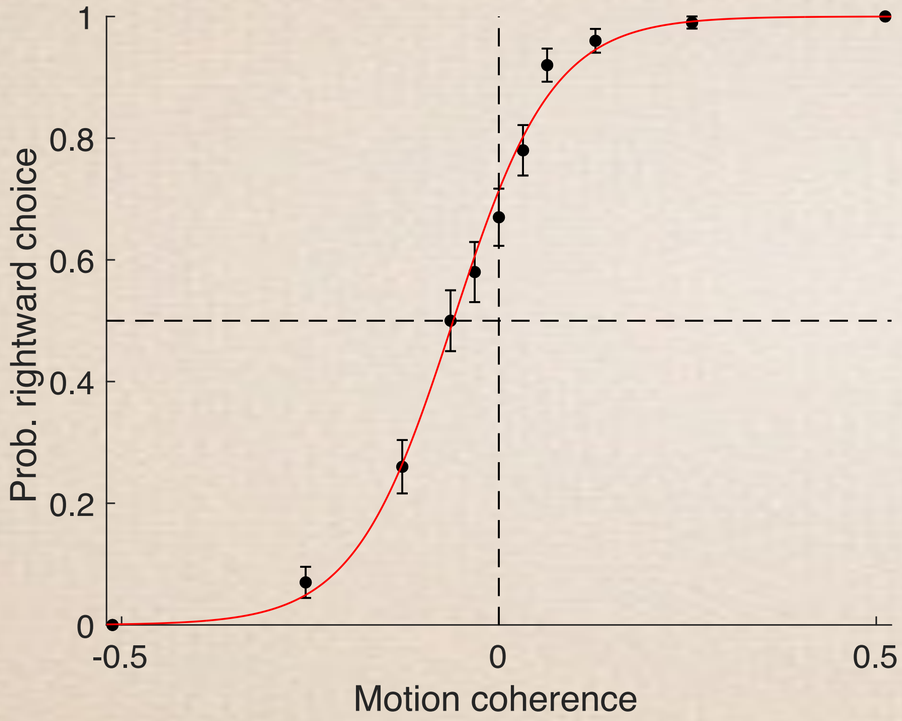
Yet, sometimes decision makers have a systematic bias favoring one of the choice alternatives



0%
Motion coherence



Modeling choice biases



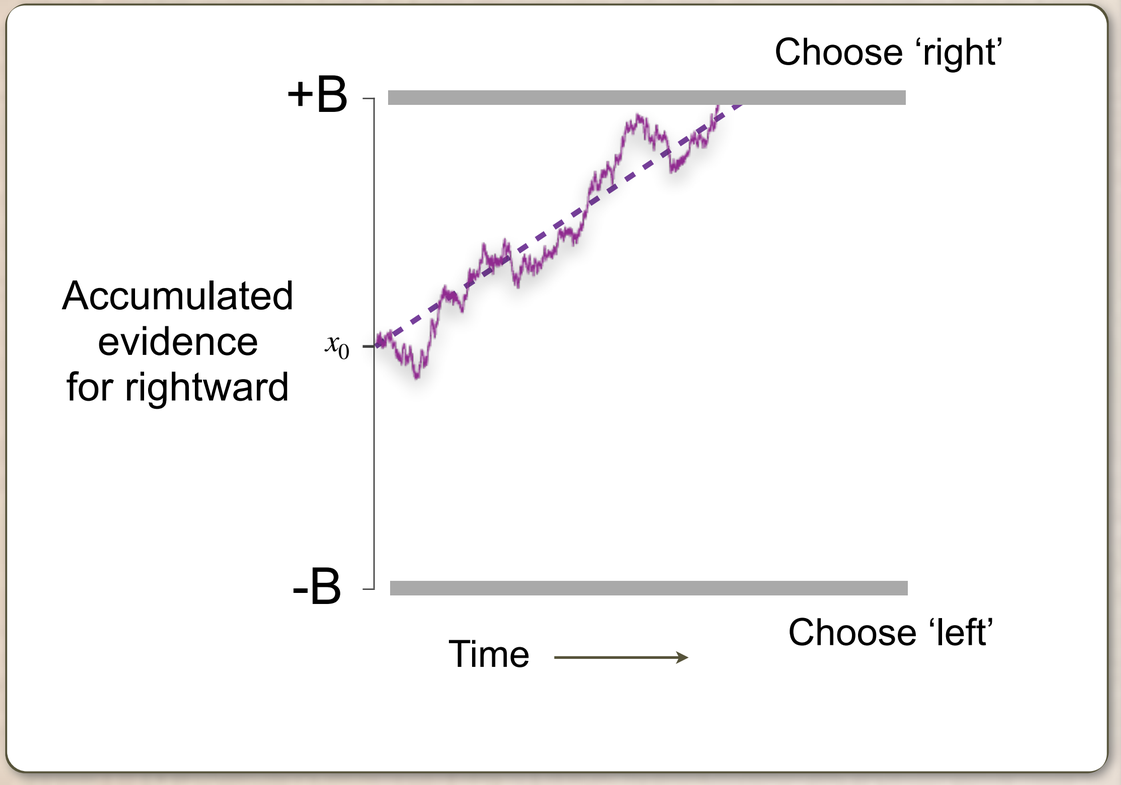
$$p_{\text{rightward}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \text{coh})}}$$

$\beta_0 \neq 0$

Bias in the DDM model:

- Offset to the starting point
- Offset to the drift rate

Offset in starting point



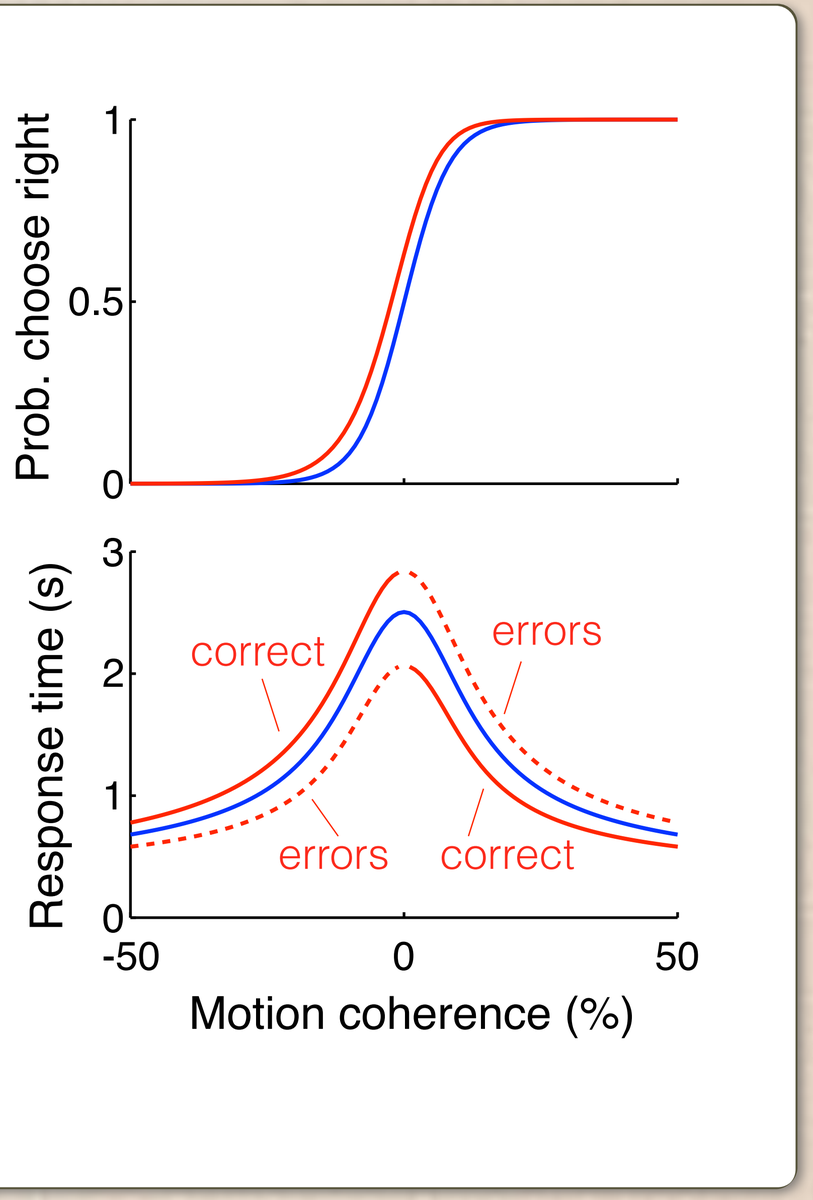
decision variable

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

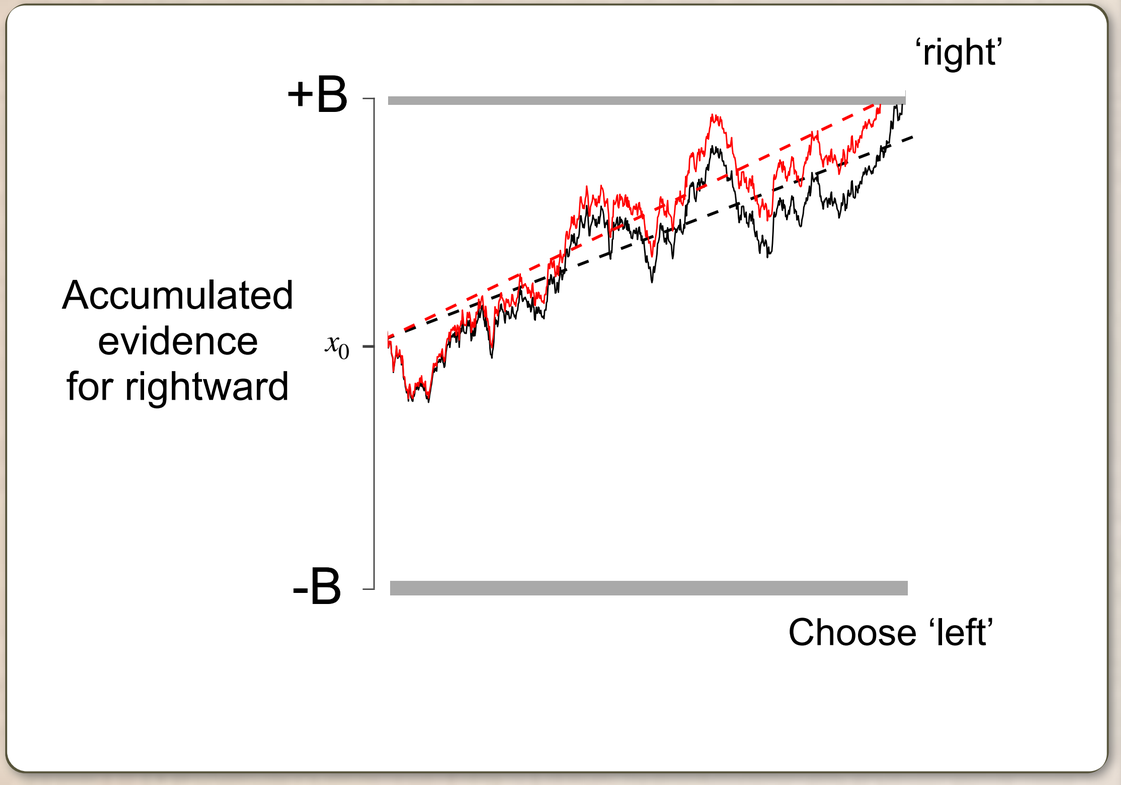
evidence

i: time step

$x_0 \neq 0$



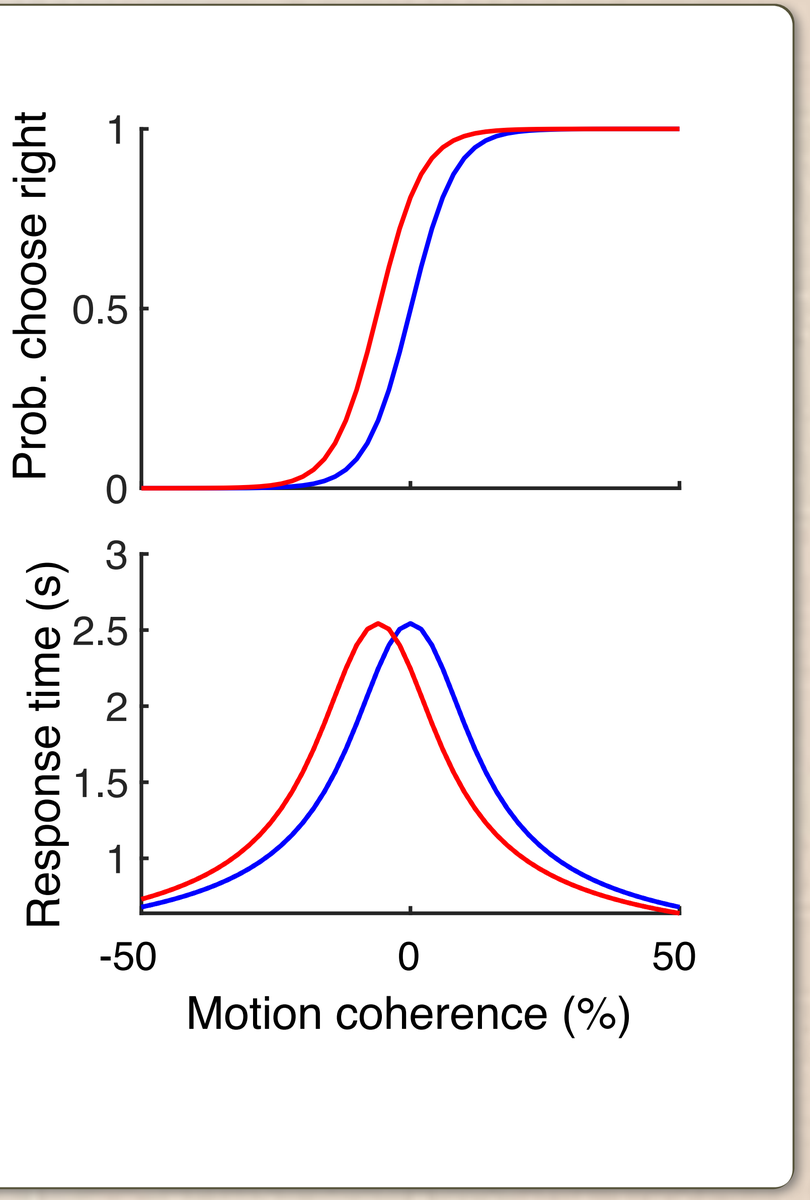
Offset in drift rate



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

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

$$x_0 = 0$$



Questions?

Please complete online questionnaire:
https://forms.gle/WV6jo2MtuGywMMBE7

Towards data fitting…

1. How to predict choice & RT from a set of model parameters?
2. What to minimize/maximize when fitting parameters?
3. What fitting algorithm to use to search over the parameter space?

1. How to predict choice & RT from a set of 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…

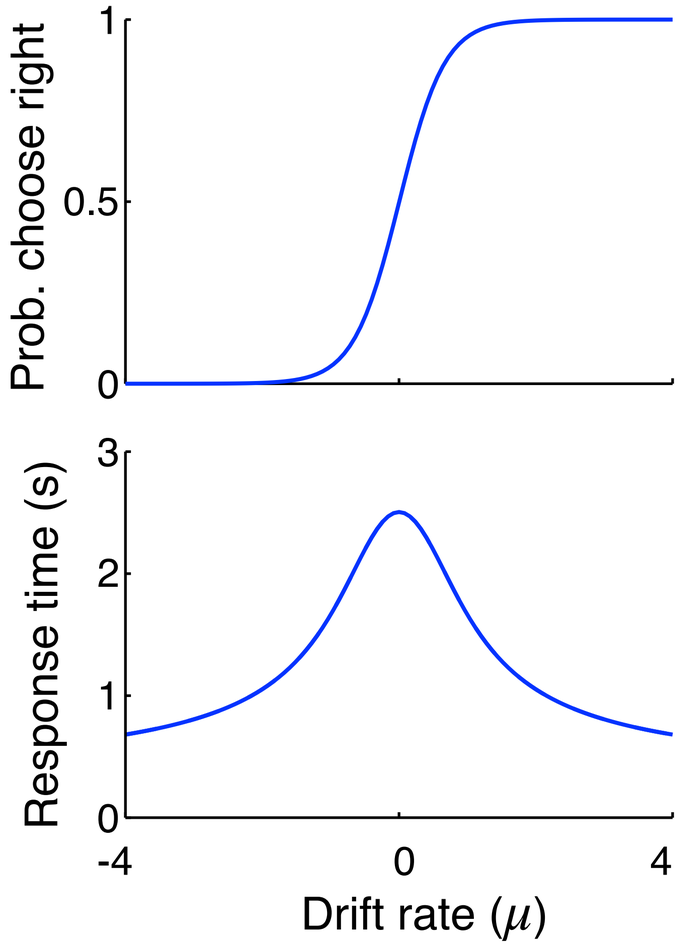
Probability of terminating at the upper bound:

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

Average response time:

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

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



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

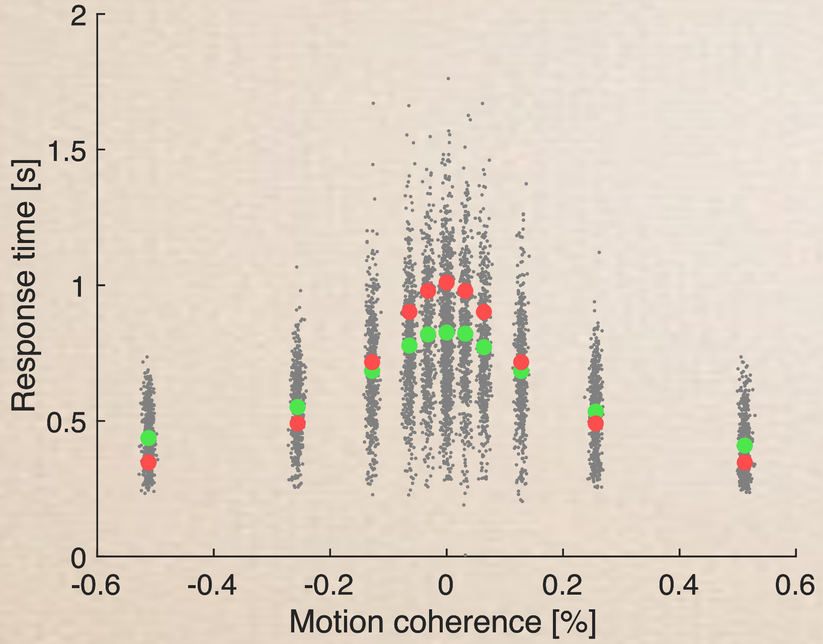
- Simulate multiple diffusion paths,
- Analytical solutions to the choice and response time functions,
- Numerical solutions (propagating over time the probability distribution for the state of the decision variable)

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



Single trials

Average per coherence

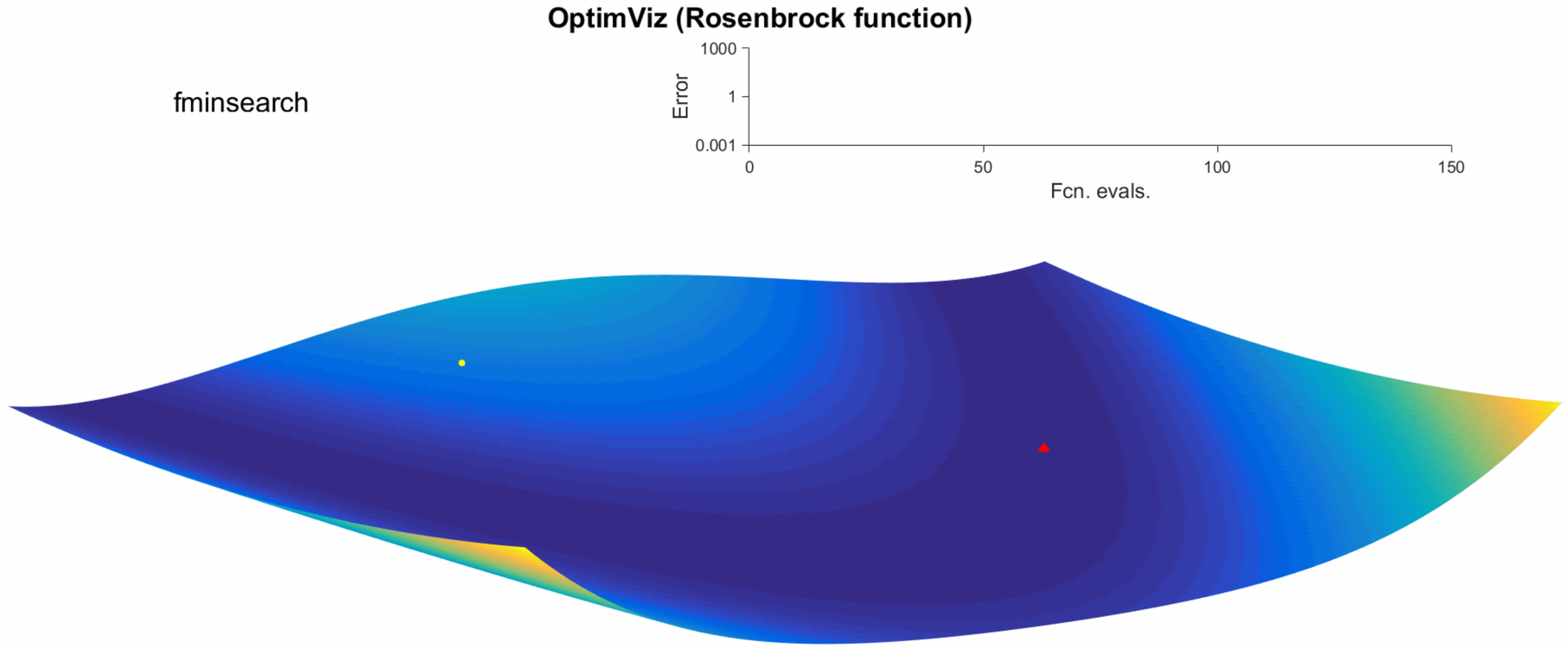
Model prediction for a set of parameters

$$\text{error} = \sum_{coh} \left(E[\text{RTdata}|\text{coh}] - E[\text{RTmodel}|\text{coh}] \right)^2$$

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)

fminsearch

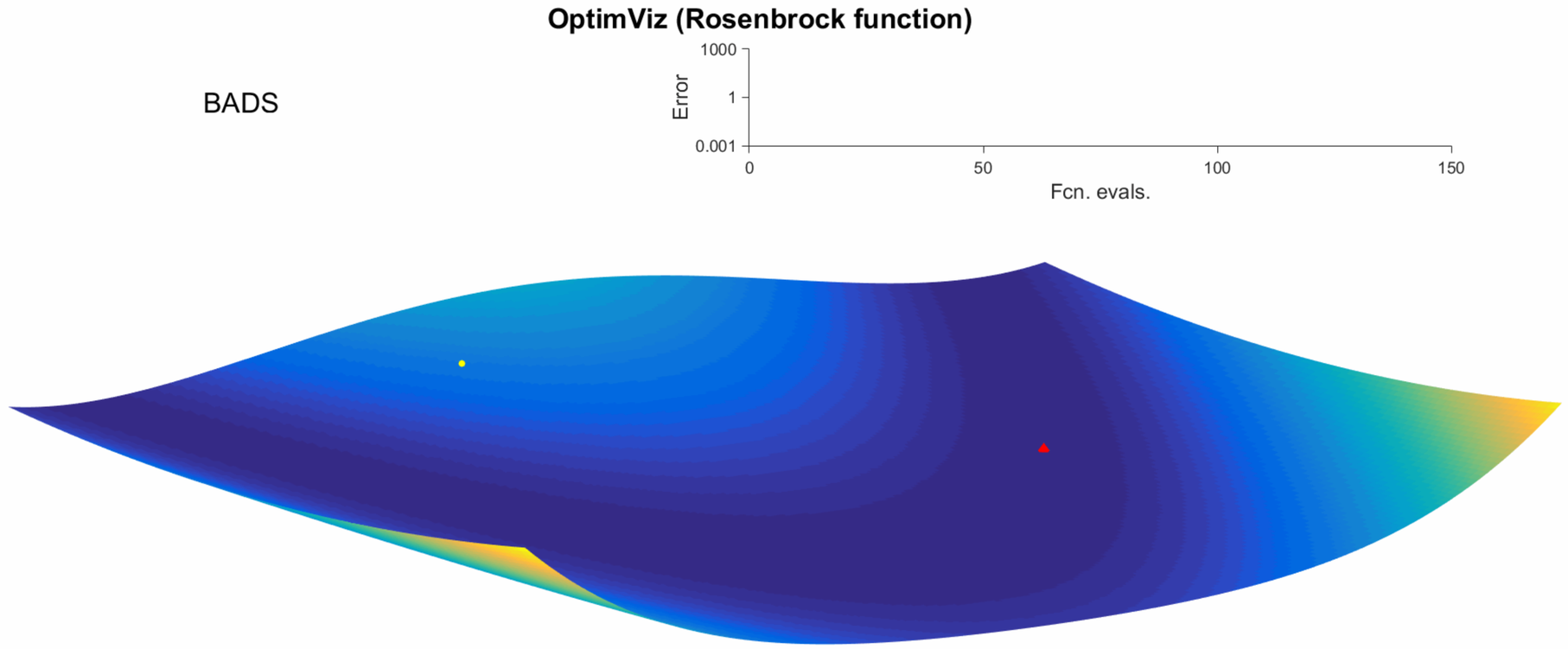


#useBADS

https://github.com/lacerbi/bads
https://github.com/lacerbi/optimviz

Slide designed by Wei Ji Ma

BADS



#useBADS

https://github.com/lacerbi/bads
https://github.com/lacerbi/optimviz

Slide designed by Wei Ji Ma

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?

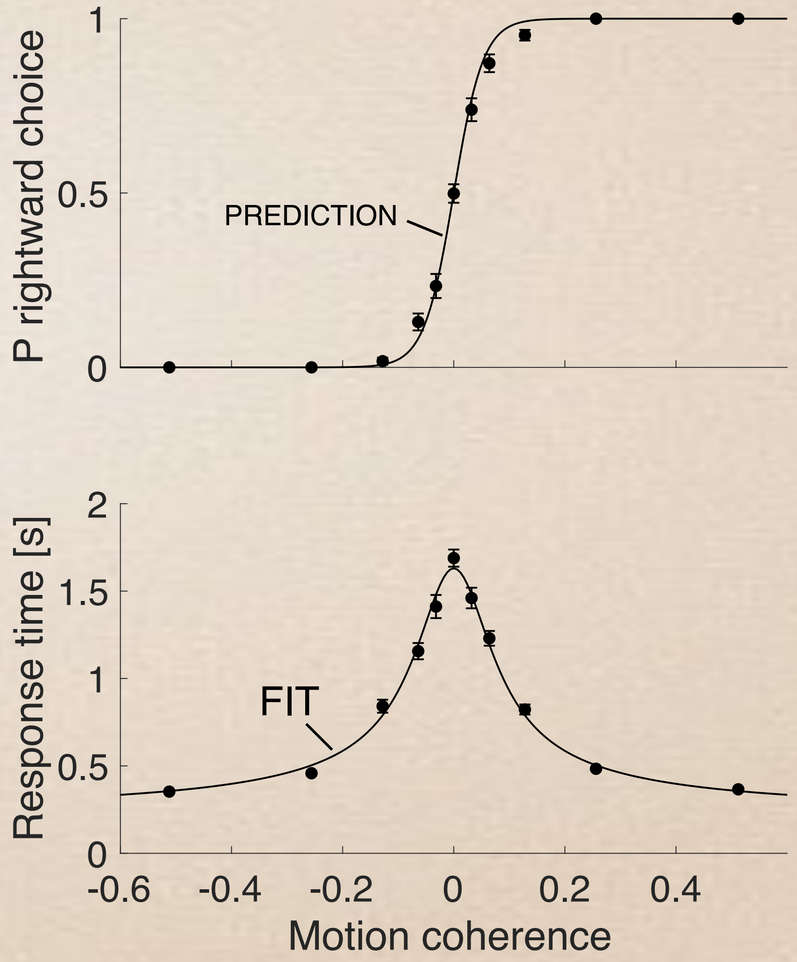
   BADS

Model fitting

Code in “03_Fit_means/main.m”

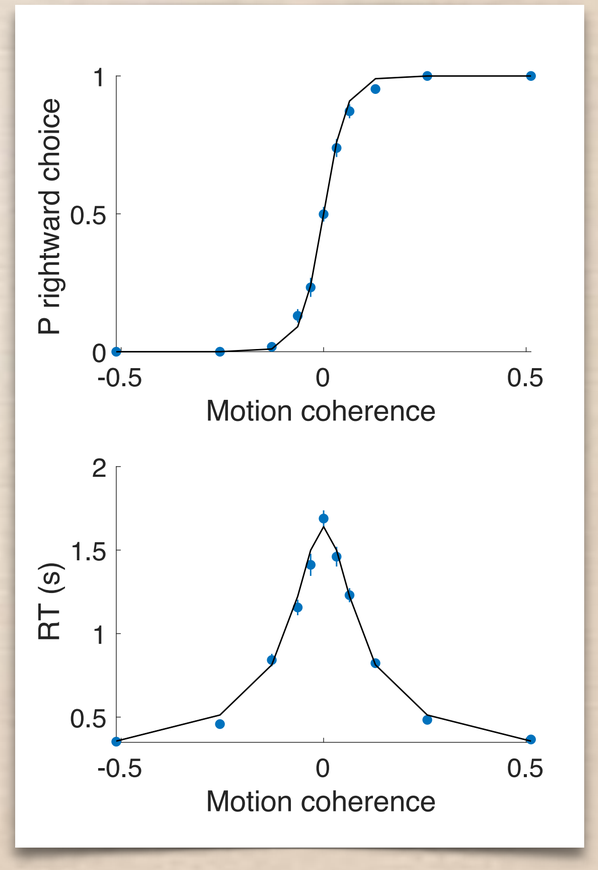
data/test_data.mat

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”

```
%% add functions to path
addpath(genpath('../functions_addtopath/'));

%% figure defaults
set(0,'defaultAxesFontSize',18);
set(0, 'DefaultLineLineWidth', 1);

%% load data
load ../data/test_data.mat

%% run one set of parameters
kappa = 15;          % signal-to-noise parameters (\mu = kappa x coh)
ndt_m = 0.2;         % non-decision time, mean [s]
ndt_s = 0;           % non-decision time, standard deviation [s]
B0    = 1.2;         % bound height
coh0  = 0;           % bias (in units of coherence)
y0    = 0;           % starting-point bias, as a fraction of bound height
ndt_m_delta = 0;     % difference in non-decision time between right and left choices
theta = [kappa,ndt_m,ndt_s,B0,coh0,y0,ndt_m_delta]; % model parameters

params = struct('plot_flag',1);

% data struct
D = struct('rt',rt,...           % response time
           'coh', coh,...        % motion coherence
           'choice',choice,...   % choice [0,1]
           'c',c);               % correct [0,1]

% eval the model parameters and plot
[~,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
```

Before we saw the analytical expressions for the expectation of the choice and response time:

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

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:

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

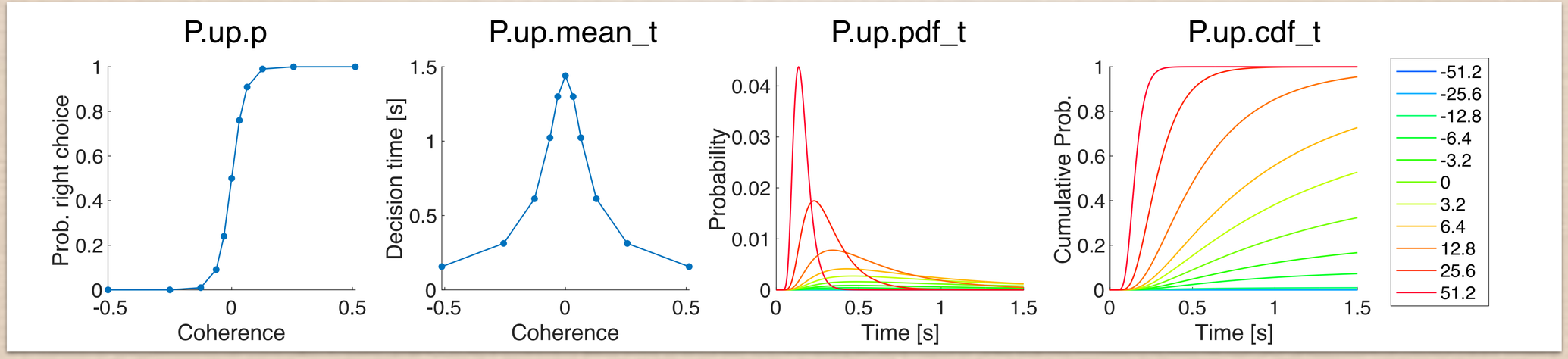
, where

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

Ratcliff 1978

“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
```



“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…

```
%% fit parameters, sum of squared errors

% kappa, ndt_mu, ndt_sigma, B0, coh0, y0, ndt_m_delta
tl = [5,  0.1, 0 ,.5 ,0,0,0]; % lower limit
th = [40, 0.5, 0 ,3  ,0,0,0]; % upper limit
tg = [15, 0.2, 0 ,1  ,0,0,0]; % guess

params = struct('plot_flag',1,'optim_method',2); % sum of squared errors of RT

fn_fit = @(theta) (wrapper_analytic_DDM(theta,D,params));

MaxFunEvals = 200; % For the tutorial only, so it does not take too long
options = optimset('Display','final','TolFun',.01,'FunValCheck','on',...
    'MaxFunEvals',MaxFunEvals);
ptl = tl;
pth = th;
[theta_SE,fval,exitflag,output] = bads(@(theta) fn_fit(theta),tg,tl,th,ptl,pth,options);

```

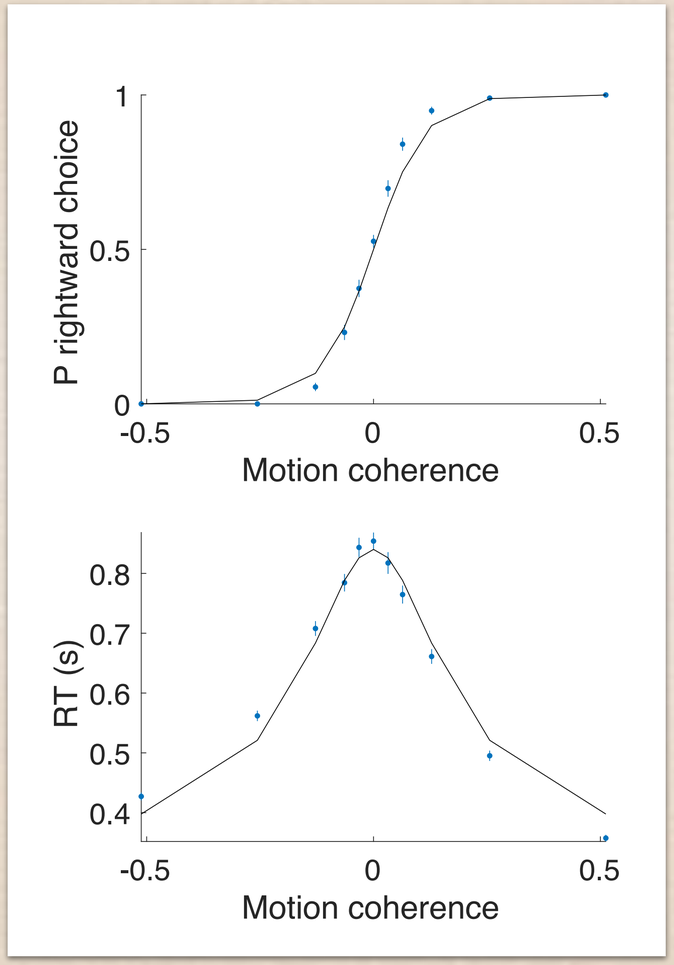
Matlab exercise:

Make a copy of the file “03_Fit_means/main.m”

Replace the “test_data” by the data of Roitman and Shadlen 2002, that is in “../data/roitman_data.mat”

Fit the model to the Roitman dataset and interpret the results

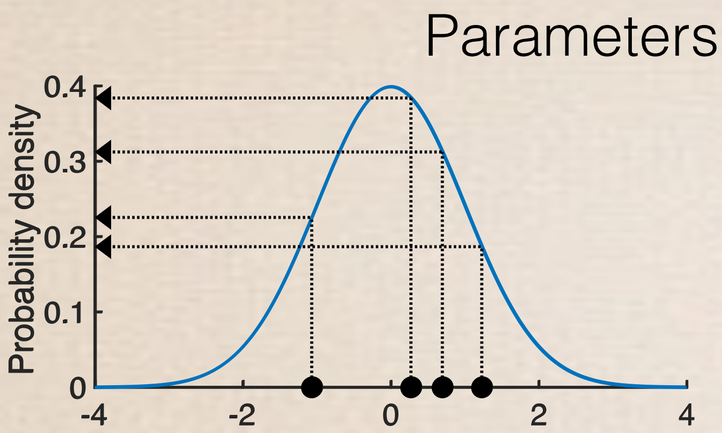
You probably obtained something like this:



Why not fit summary statistics (mean RTs) using the sum of squared errors?

- 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^*

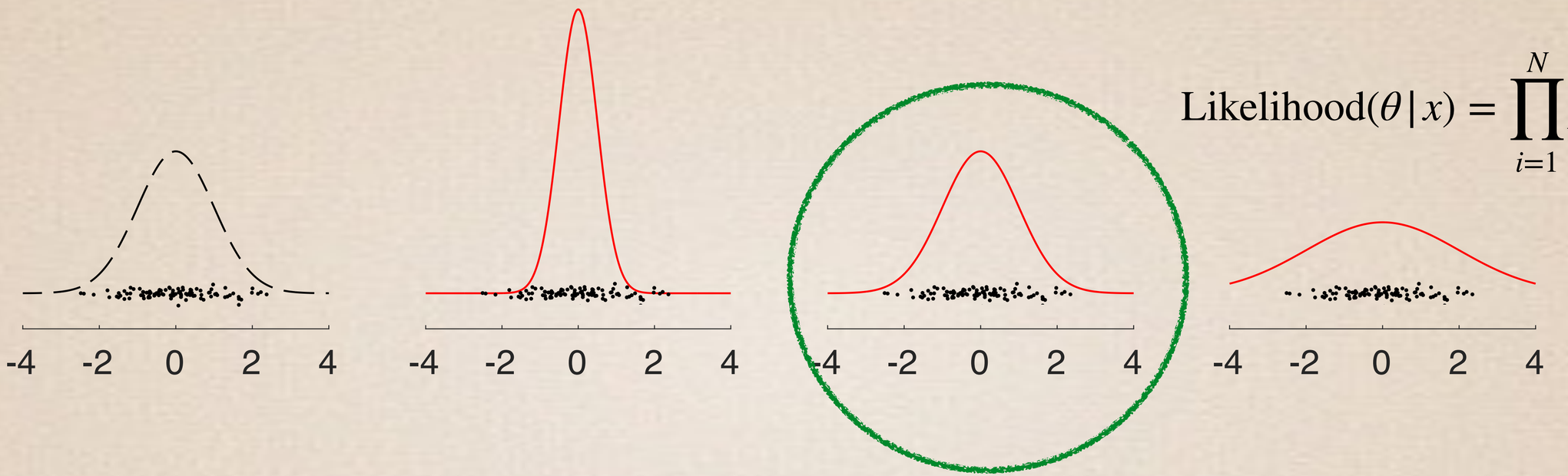
data points

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

parameters

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

Better approach: Maximum Likelihood



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

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

$$LL = \sum_{i} log(p(x^{(i)} \mid \theta))$$

data points

parameters

Exercise on maximum-likelihood

```
%% 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);
```

Exercise:

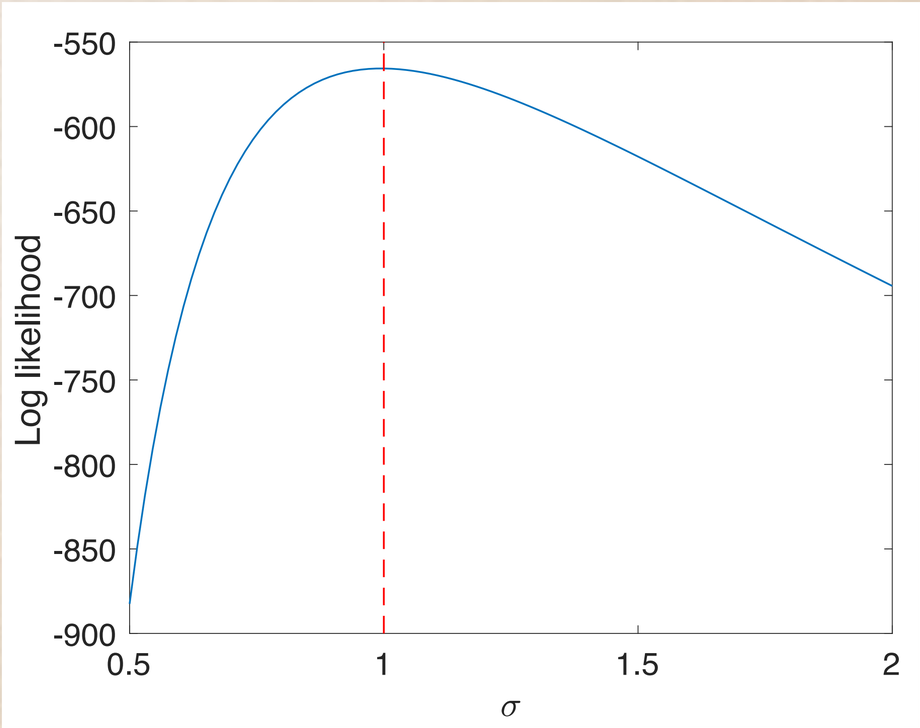
- Assume you know that the observations come from a normal distribution with mean $\mu = 0$
- But we do not know σ
- Compute the log-likelihood for each possible value of σ
- Find the value of σ that maximizes the log-likelihood
- Key Matlab command: p = normpdf(data, 0, sigma(i))

```
%% 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));
```

data points

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

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 coherence

We need to be able to evaluate this p.d.f.

For ‘flat’ bounds, there are analytic solutions to the choice and response time distributions

Probability of terminating at the lower bound at time t:

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

, where

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

Ratcliff 1978, Feller 1968

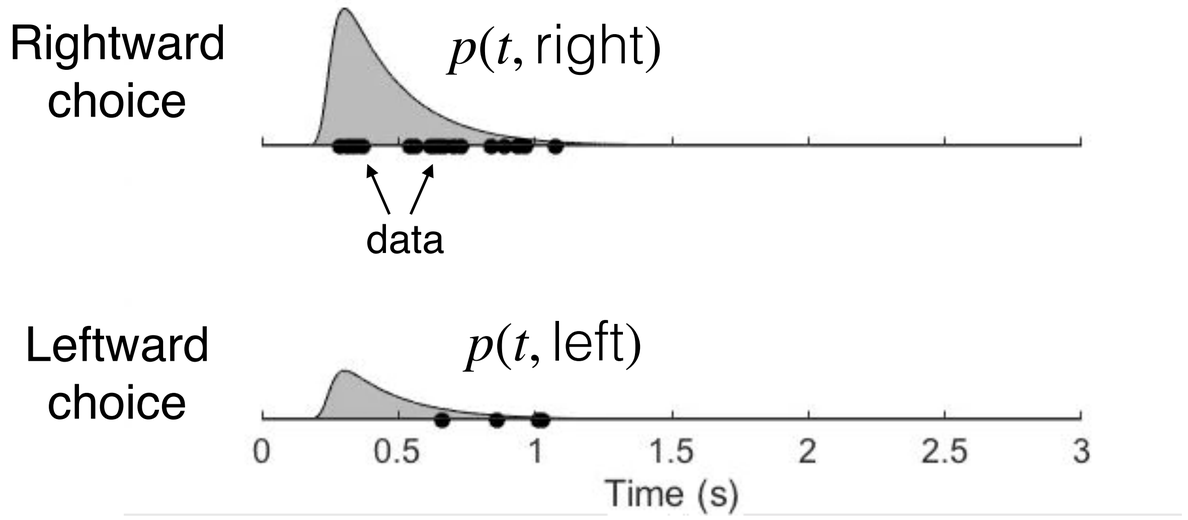
Maximum Likelihood for choice & RT data

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

model parameters

motion coherence

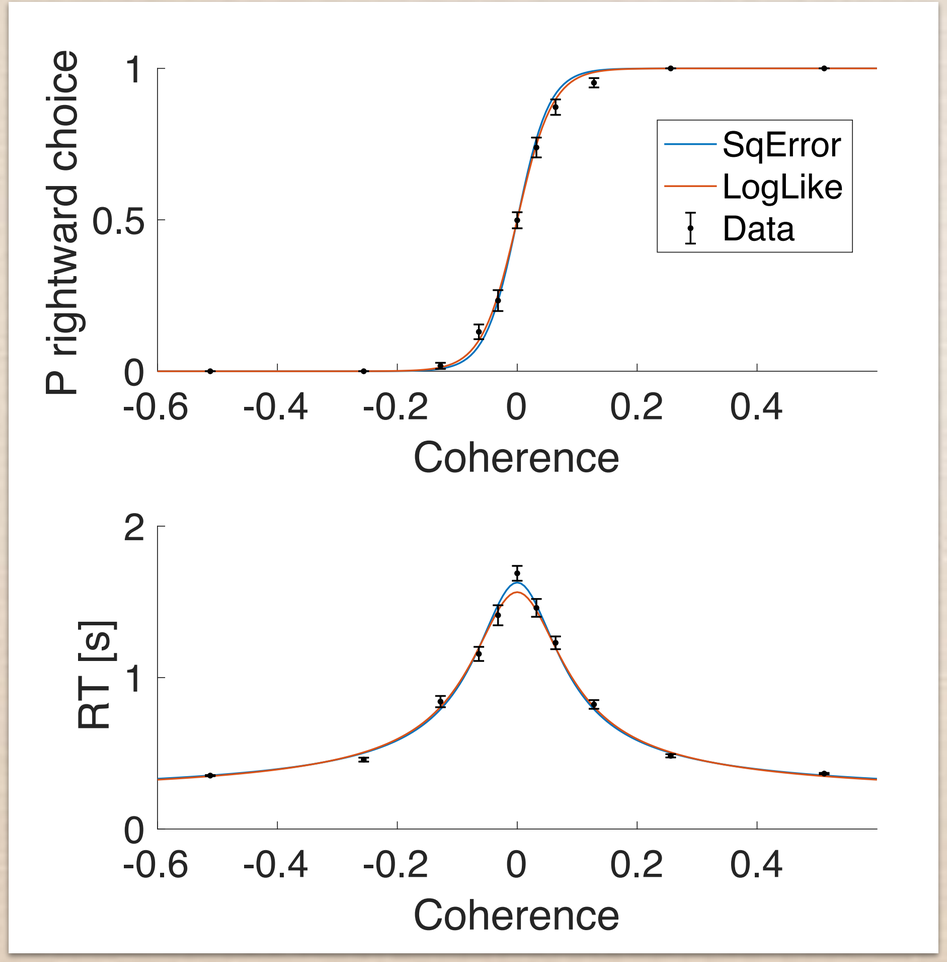
Motion coherence = 6.4%



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

“03_Fit_means/main.m”.

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



Toolboxes for fitting the DDM with flat bounds

1. HDDM

   http://ski.clps.brown.edu/hddm_docs/

2. DMAT

   https://ppw.kuleuven.be/okp/software/dmat/

3. fast-dm

   https://www.psychologie.uni-heidelberg.de/ae/meth/fast-dm/

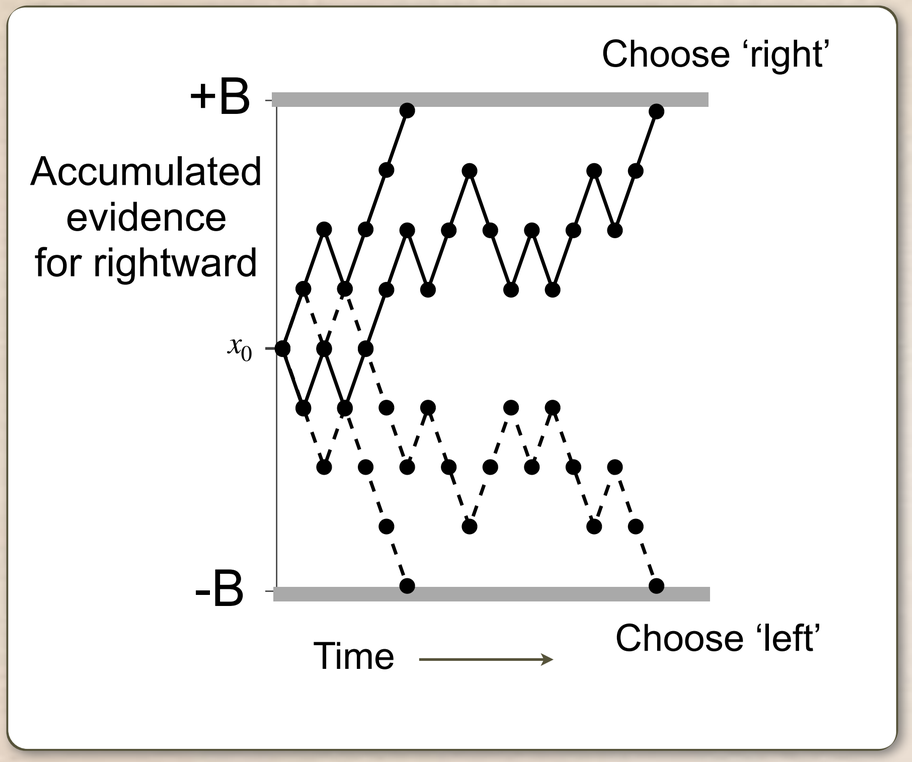
4. EZ

   https://www.ejwagenmakers.com/2007/EZ.pdf

Maximum likelihood tries to capture the full distribution of RTs, not just the means.

But there is an issue with the “simple” version of the DDM we analyzed so far…

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



↗ with probability p

↘ with probability $q = 1 - p$

Pr = $qppppqpp$

Pr(mirror) = $pqqqpqqq$

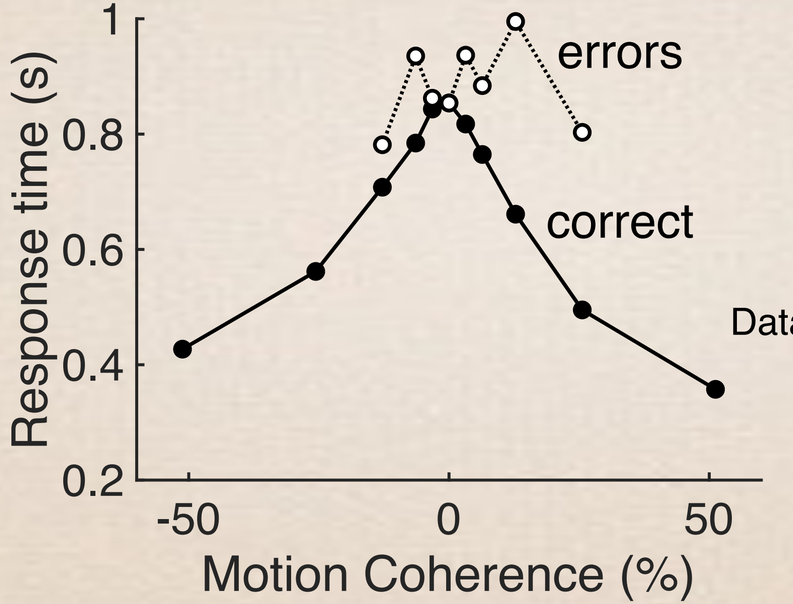
$$\frac{\Pr}{\Pr(\text{mirror})} = \frac{pppp}{qqqq} = \left(\frac{p}{1-p}\right)^{\gamma} \quad \gamma = 4$$

After normalization:

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

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

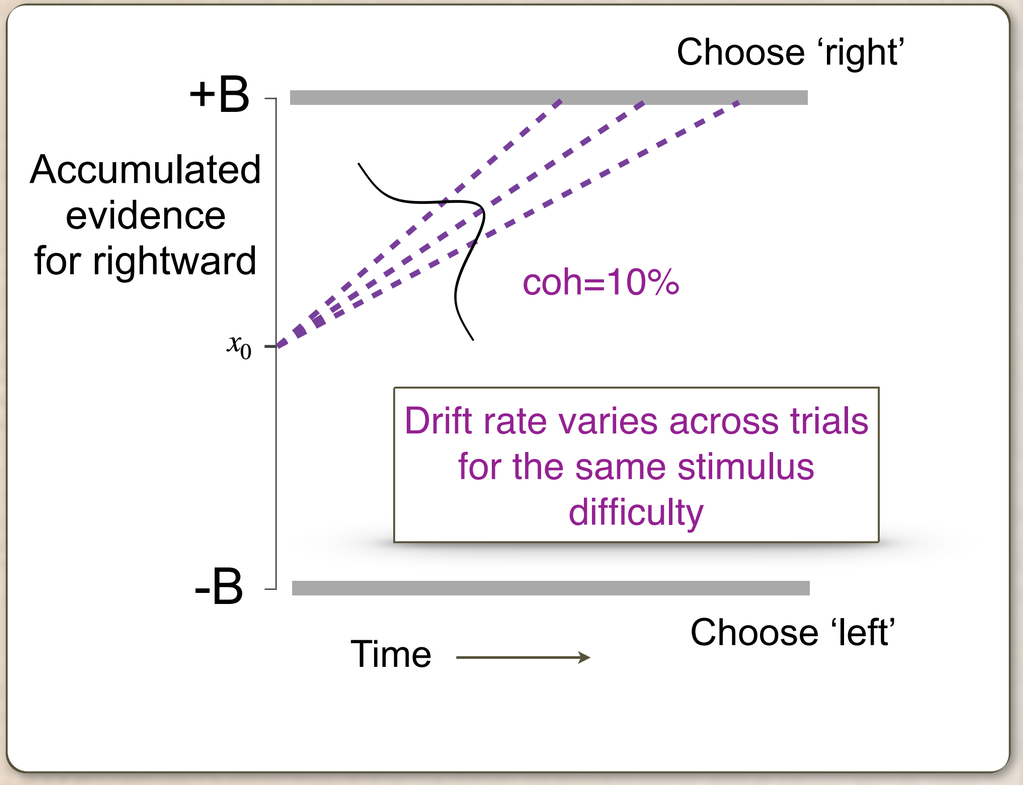
Not what is seen in data…



Data from Roitman and Shadlen, 2002

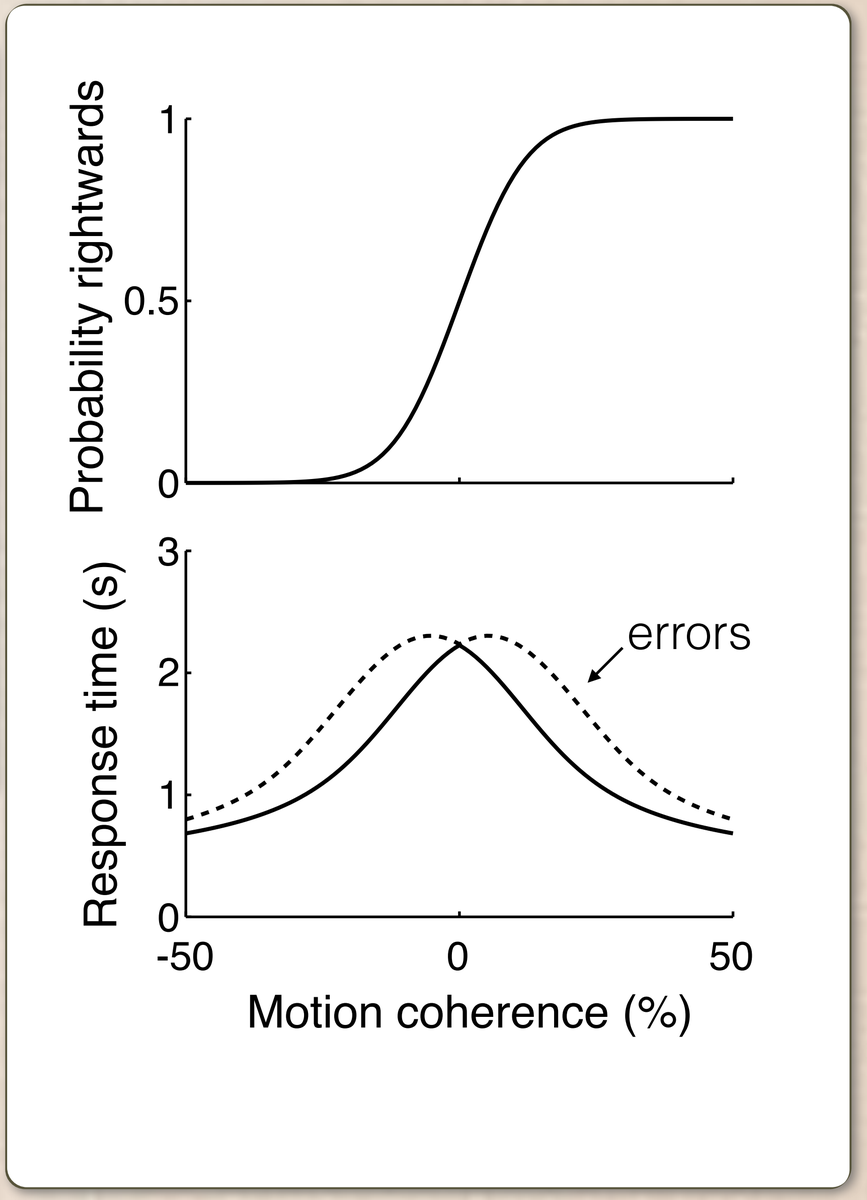
How to “patch” the simple model to make errors slower than correct responses?

Inter-trial variability in drift-rate across trials



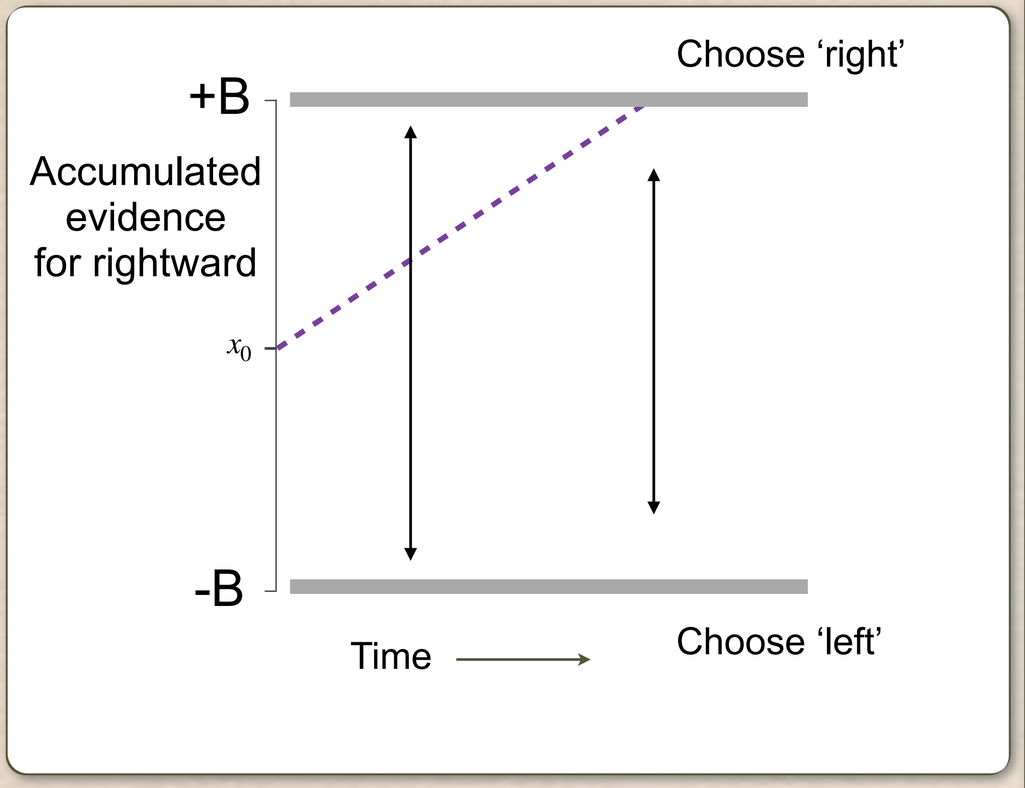
Drift-rates closer to zero → Slower responses

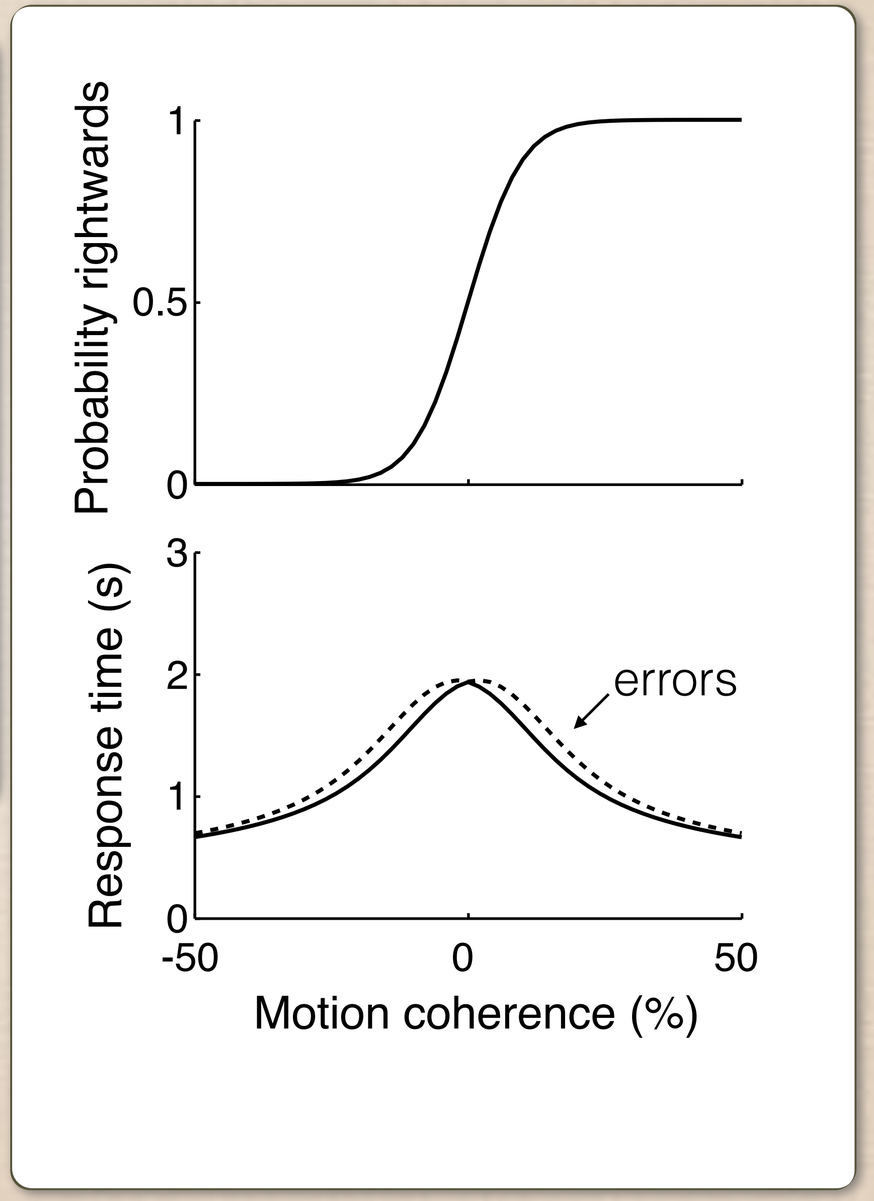
Drift-rates closer to zero → Higher error rates



Ratcliff & McKoon (2008)

Collapsing bounds

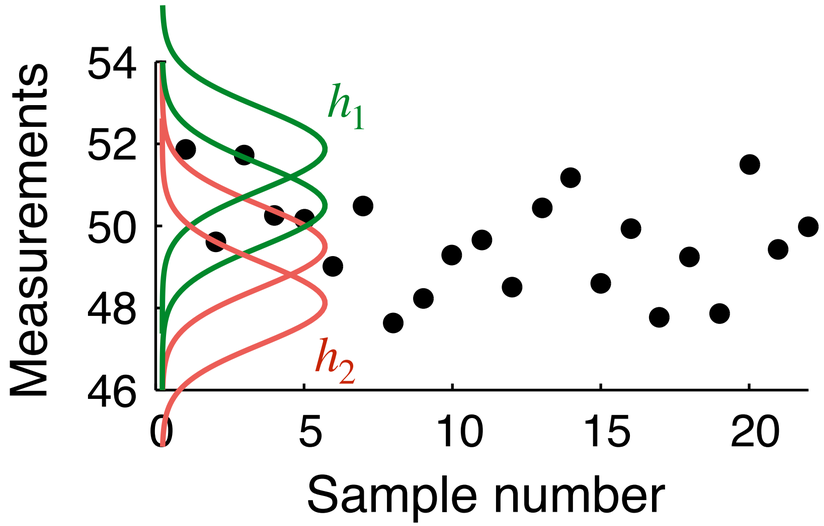


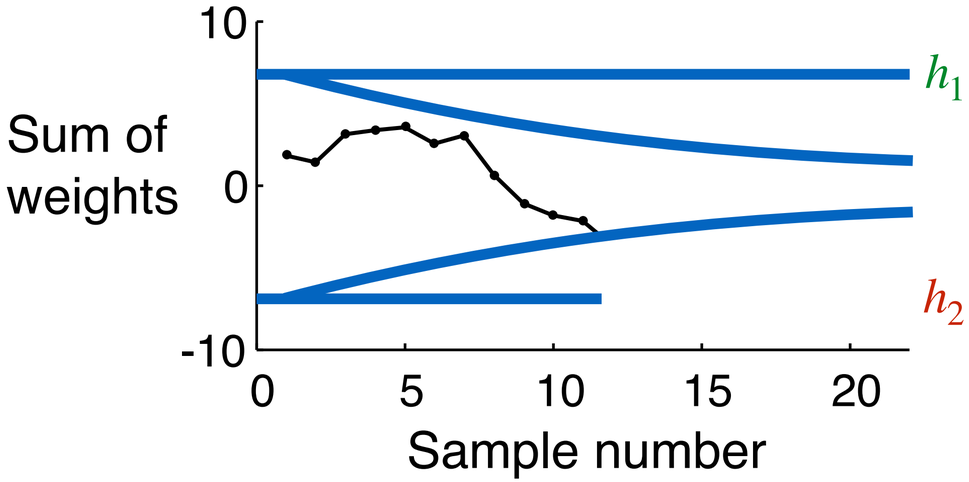


MN Shadlen et al. (2006)

The patch is not really a patch…

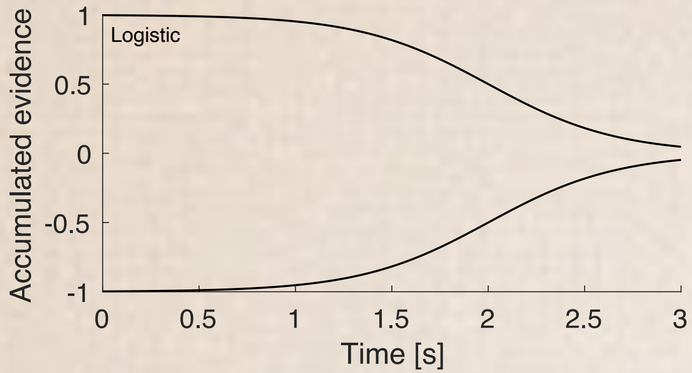
Collapsing bounds maximize reward-rate when difficulty varies across trials



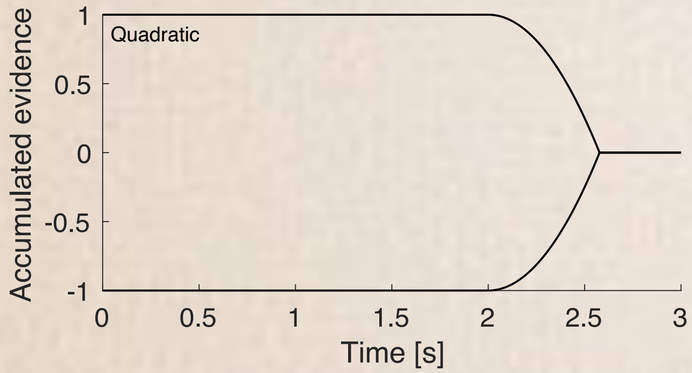


Frazier & Yu, NiPS 2008
Drugowitsch et al., JNeuro 2012

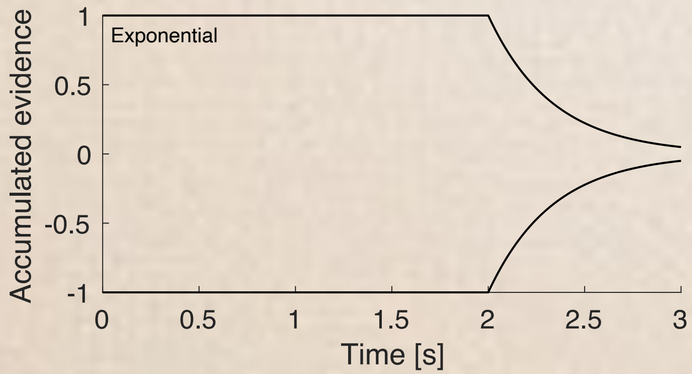
Some bound shapes…



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



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



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

See expand_bounds.m

Issue: collapsing bounds do not usually admit analytical solutions to the choice and response time distributions

We have seen this slide before…

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

- Simulations of diffusion paths,
- Analytical solutions to the choice and response time functions, for averages or full distributions,
- Numerical solutions (propagate over time a probability distribution for the state of the decision variable)

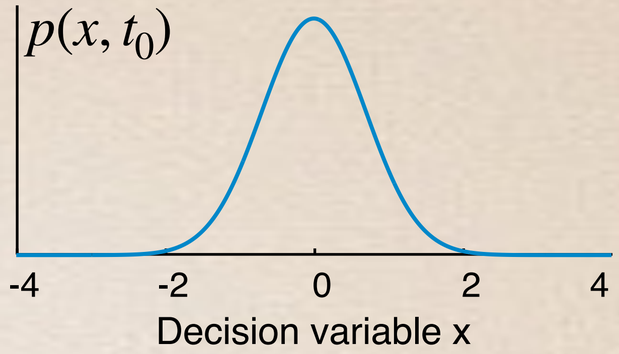
Fokker-Planck equation

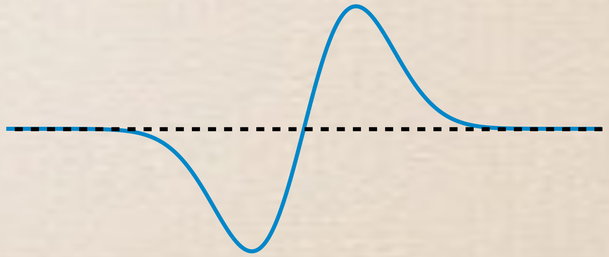
Partial differential equation that describes the time evolution of the probability density function of the decision variable x:

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

Fokker-Planck intuition

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

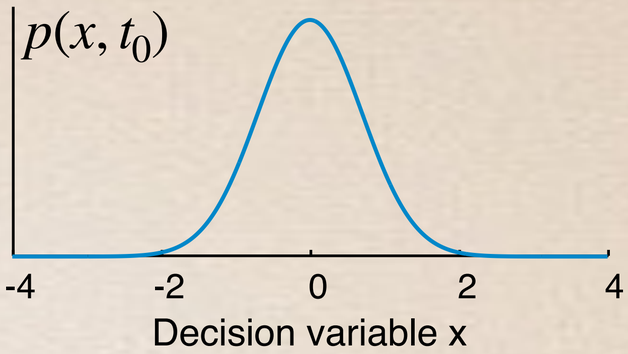


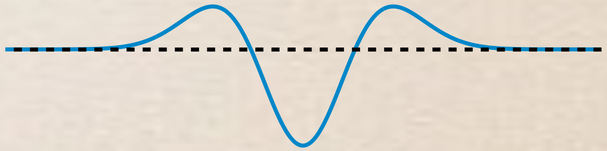


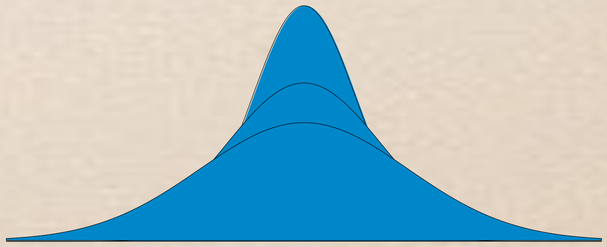
Fokker-Planck intuition

Fokker Planck equation:

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



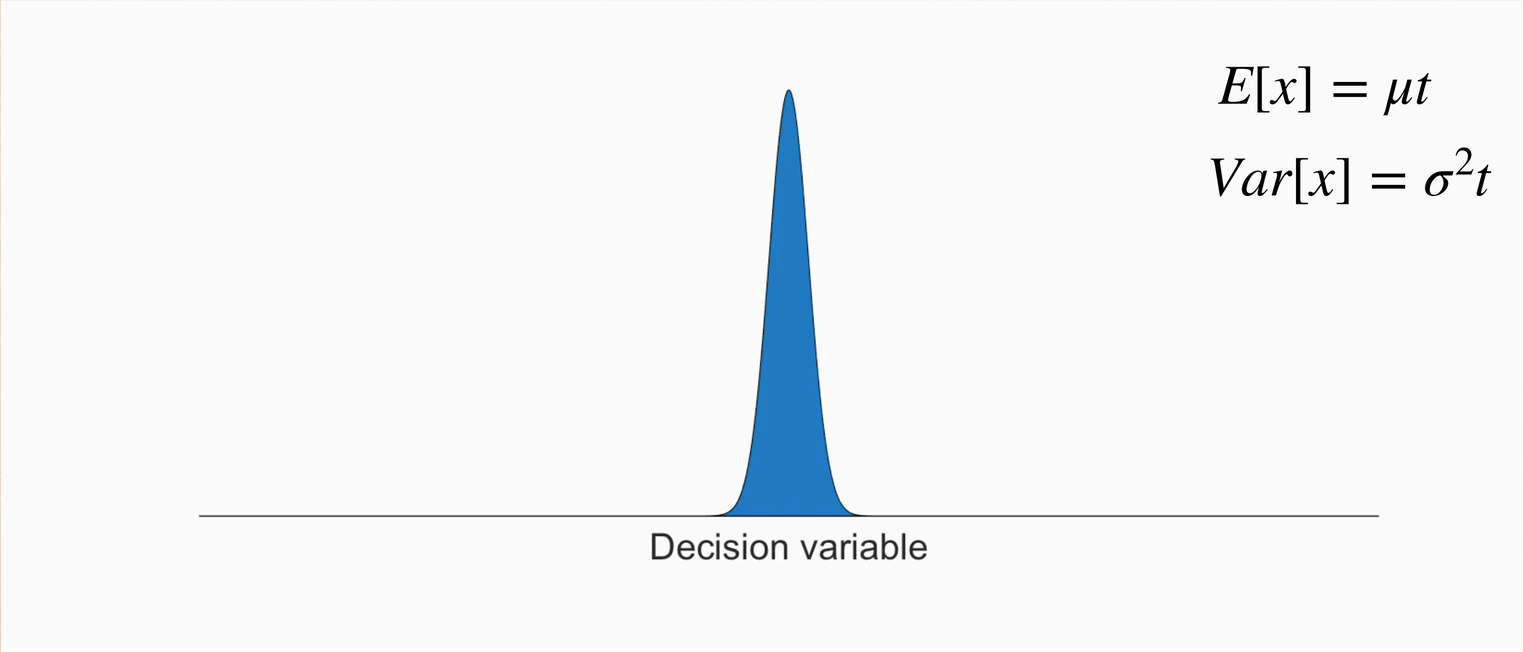




The two terms together

Fokker Planck equation:

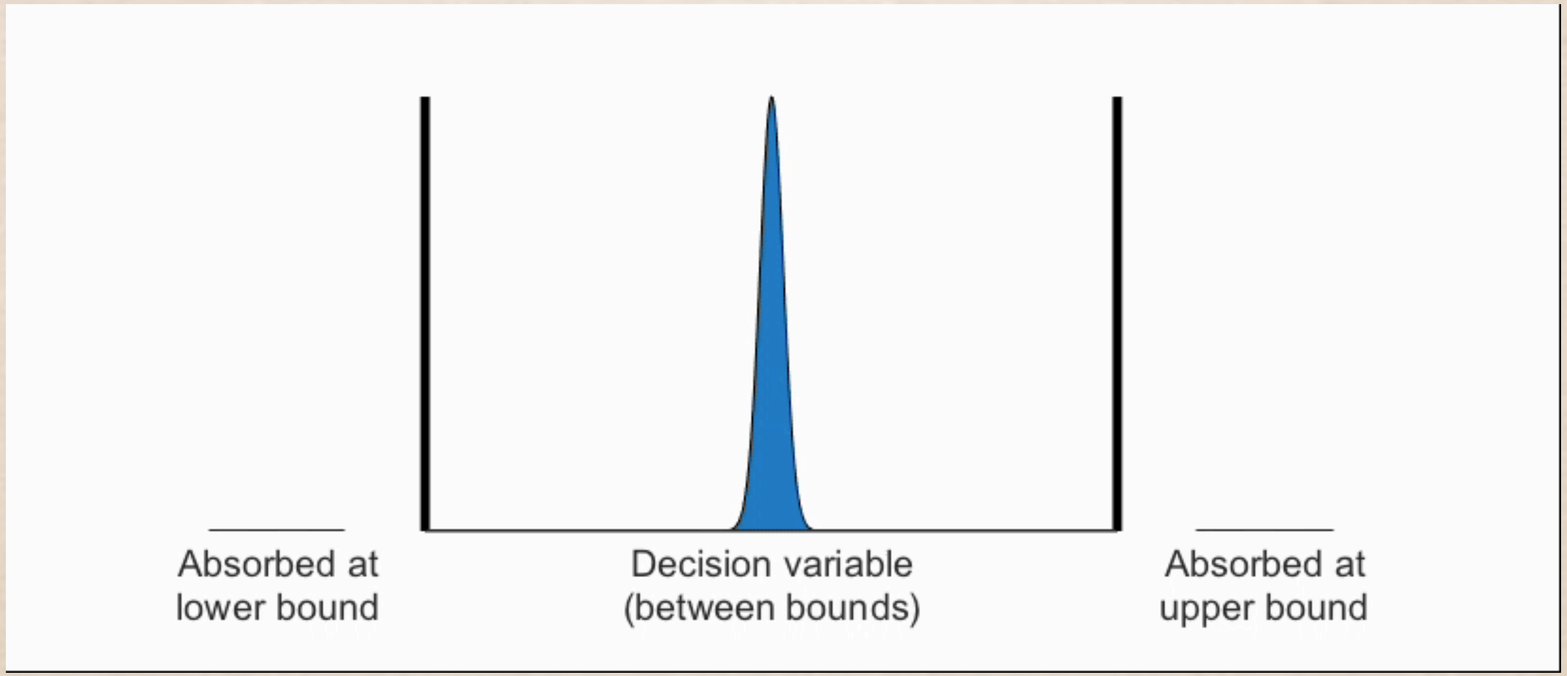
$$\frac{\partial p(x,t)}{\partial t} = -\mu \frac{\partial p(x,t)}{\partial x} + \frac{\sigma^2}{2} \frac{\partial^2 p(x,t)}{\partial^2 x}$$



With bounds

Fokker Planck equation:

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

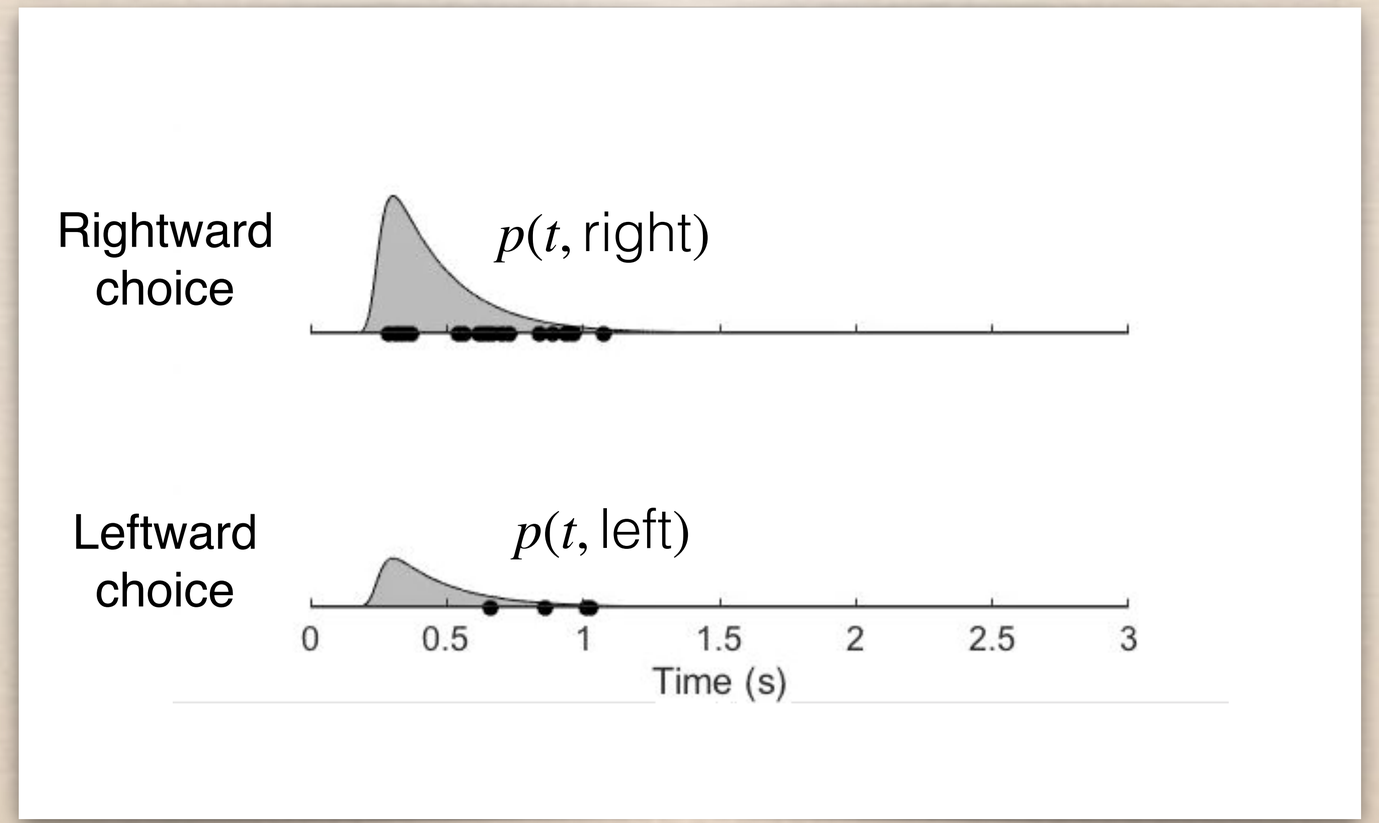


Model fitting

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

model parameters

motion coherence



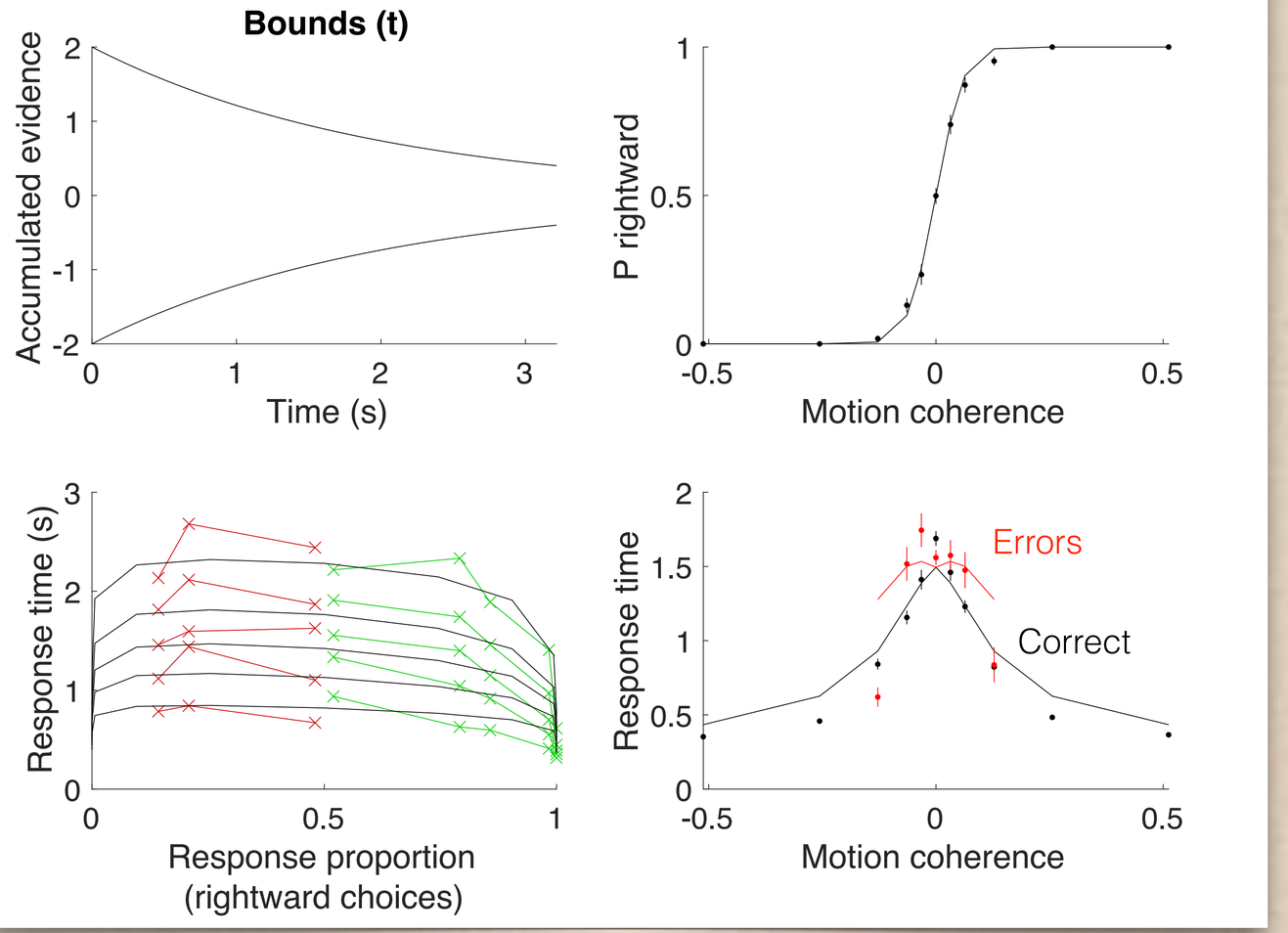
Let's try it!

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”

```
addpath(genpath('../functions_addtopath/'))

load('../data/test_data.mat','rt','coh','c','choice');

D = struct('rt',rt,'coh', coh,'choice',choice,'c',c);

%% one example run
kappa = 15;    % signal-to-noise
ndt_m = 0.2;   % non-decision time, mean [s]
ndt_s = 0.01; % non-decision time, standard de
B0    = 2;     % bound height
a     = 0.5;   % bound collapse parameter
d     = 0;     % bound parameter, time of decay
y0    = 0;     % bias, on starting point
coh0  = 0;     % bias, on coherence

theta = [kappa,ndt_m,ndt_s,B0,a,d,coh0,y0];

pars = struct('plot_flag',true);

[err,P] = wrapper_DTB_parametricbound(theta,D,
```



“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

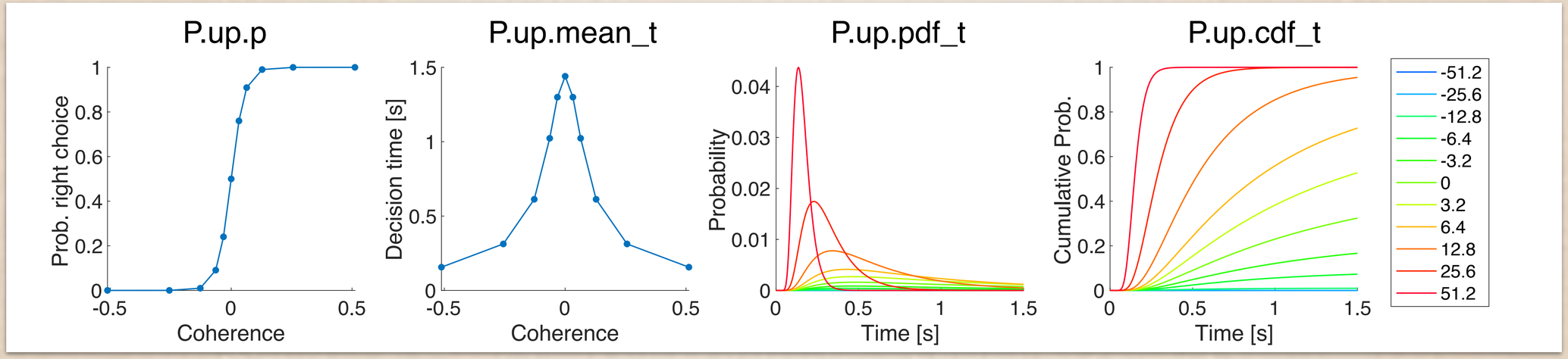
“"wrapper_DTB_parametricbound.m"”

```
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…

```
%% fitting

% kappa, ndt_mu, ndt_sigma, B0, a, d, coh0, y0
tl = [5,  0.1, .01 ,0.5  , -1, -3,0,0];
th = [40, 0.7, .15 ,4    , 4 ,4,0,0];
tg = [15, 0.2, .02 ,1    , 0.1 ,1,0,0];

pars = struct('plot_flag',true,'USfunc','Logistic');

fn_fit = @(theta) (wrapper_DTB_parametricbound(theta,D,pars));

MaxFunEvals = 100; % For the tutorial only, so it does not take too long
options = optimset('Display','final','TolFun',.01,'FunValCheck','on',...
    'MaxFunEvals',MaxFunEvals);


ptl = tl;
pth = th;
[theta, fval, exitflag, output] = bads(@(theta) fn_fit(theta),tg,tl,th,ptl,pth,options);

% run the DTB with the best fit
[~,P] = fn_fit(theta);
```

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

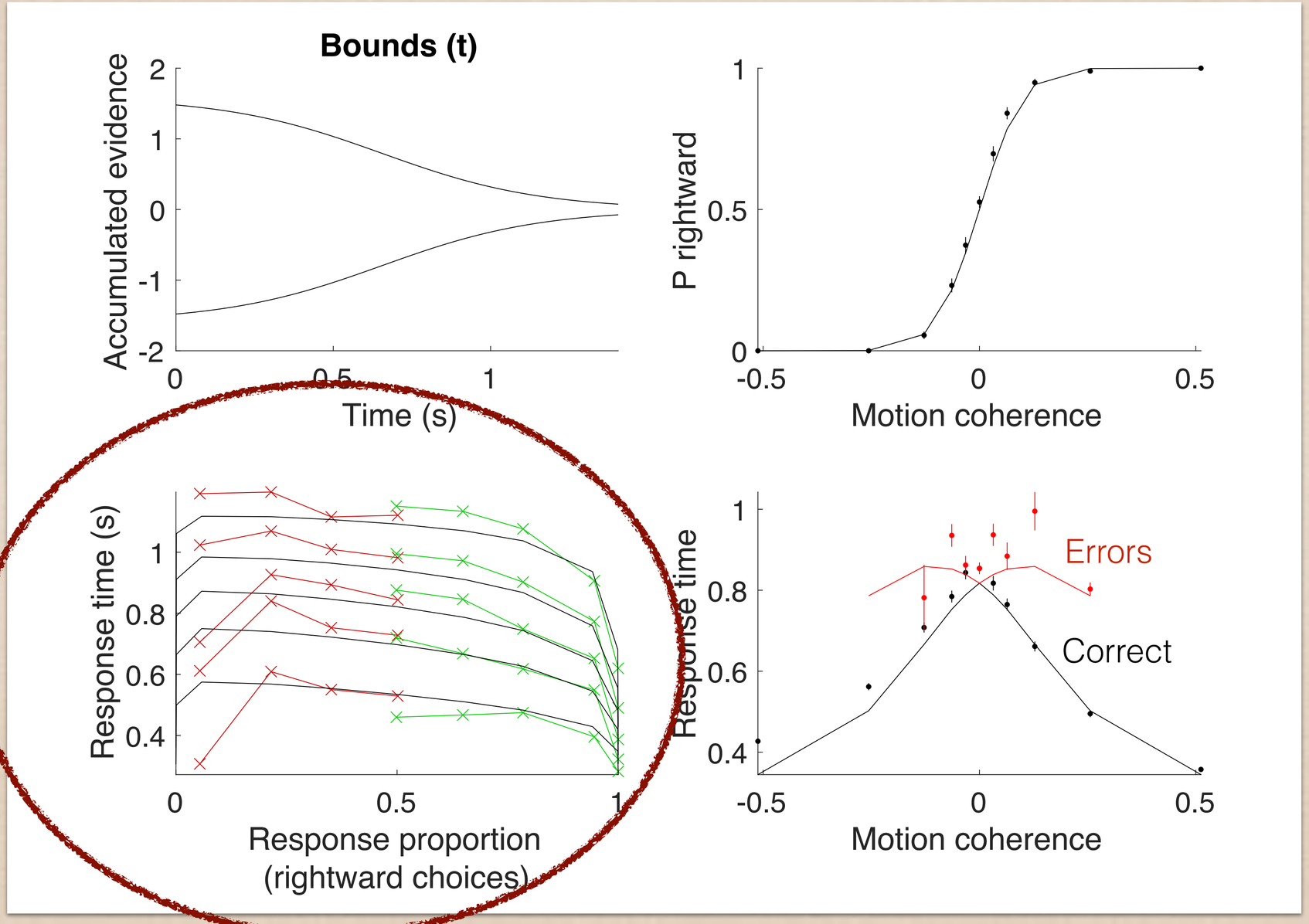
Adapt the code in “04_Fit_single_trials/main.m” to fit the data from Roitman and Shadlen, Journal of Neuroscience 2002.

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

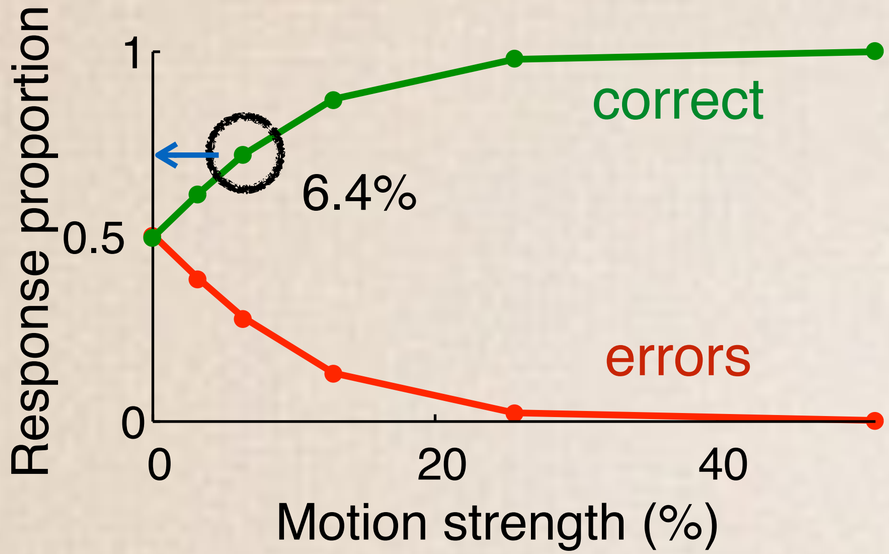
You should have obtained something like this…

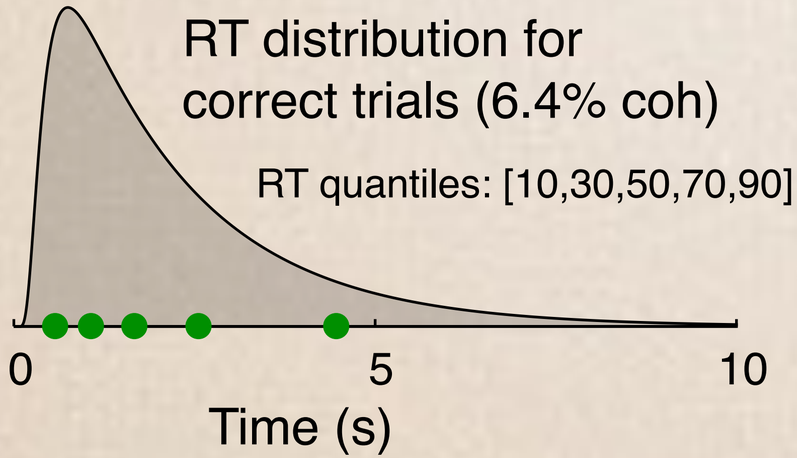
Best fitting parameters:

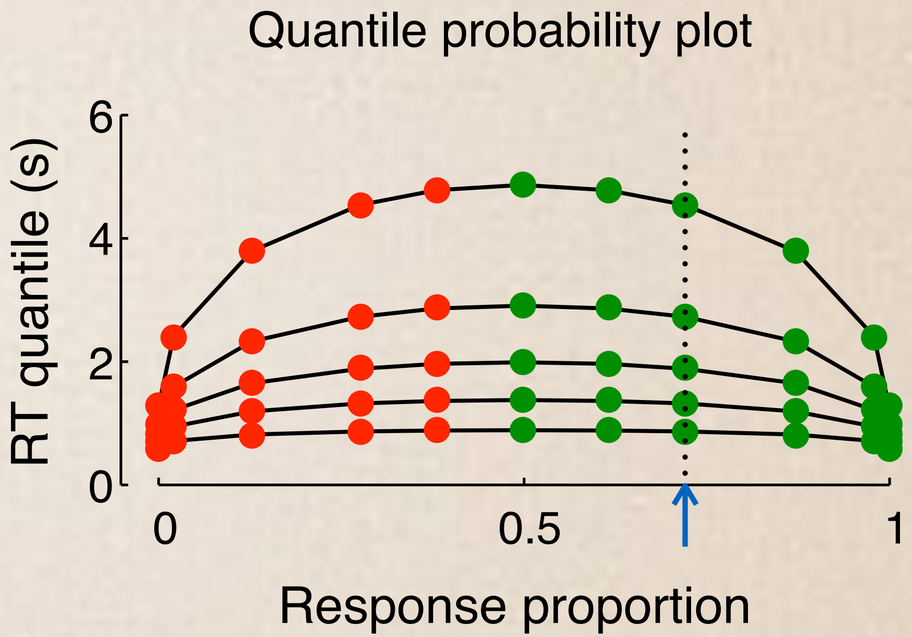
$$\kappa = 12.77$$
$$\mu_{nd} = 0.13$$
$$\sigma_{nd} = 0.13$$
$$B_0 = 1.59$$
$$a = 4$$
$$d = 0.66$$



Quantile probability plot







Compare fit parameters across
conditions/datasets
(e.g., clinical vs. non-clinical participants)

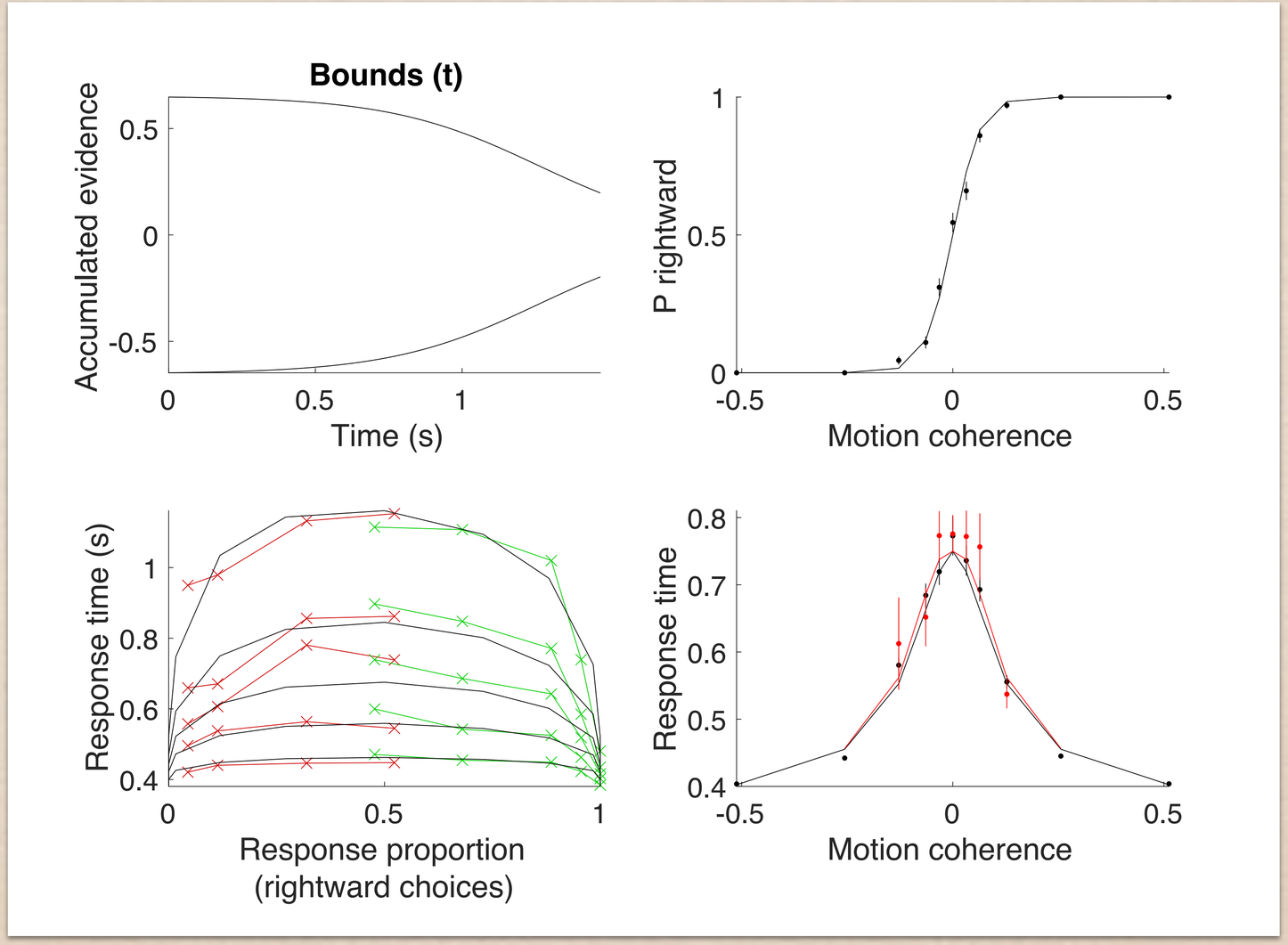
Compare fit parameters across conditions/datasets

- In "data/sim_dat_extrema.mat" there are two behavioral datasests (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:

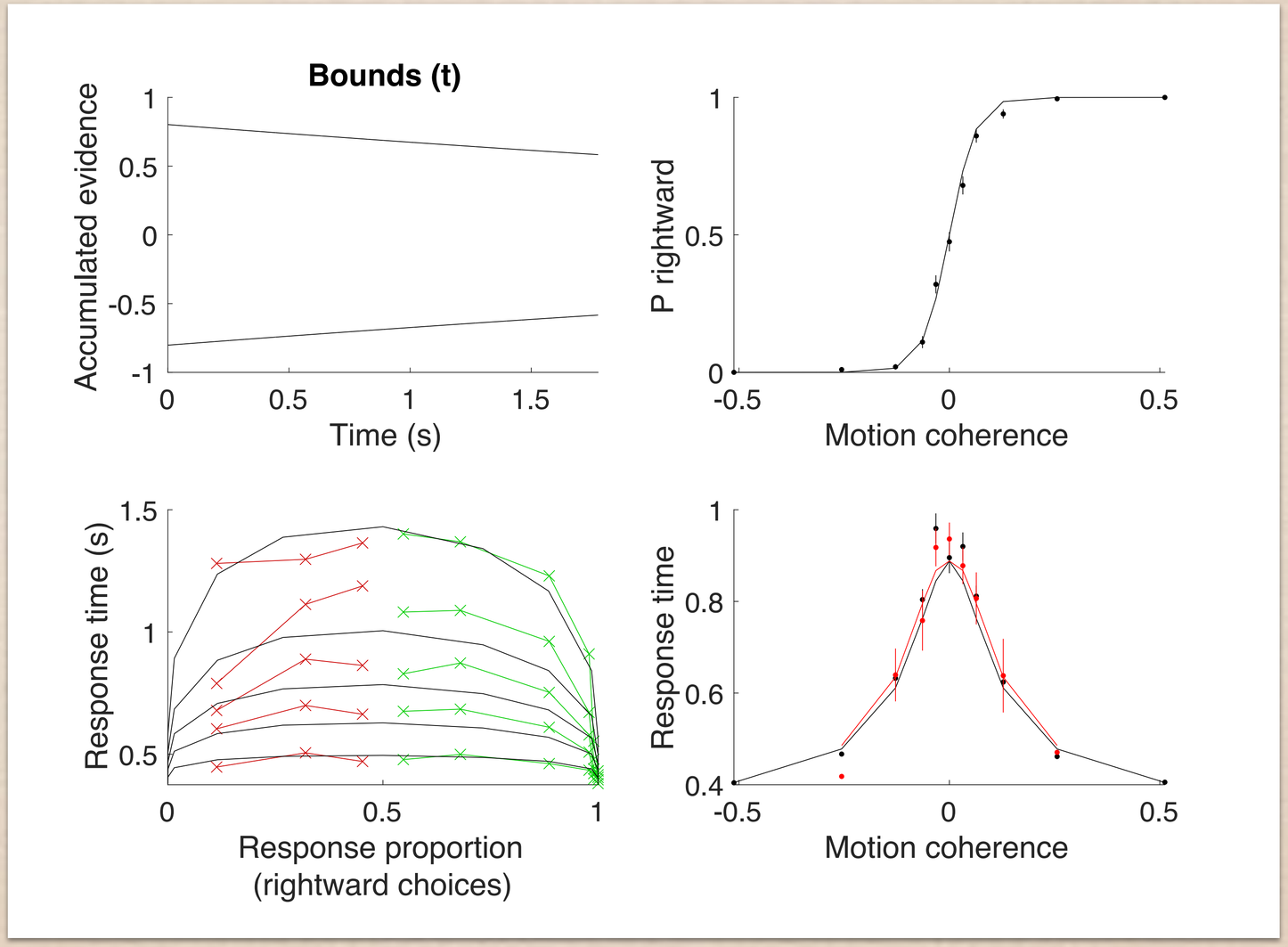
$$\kappa = 24.34$$
$$\mu_{nd} = 0.35$$
$$\sigma_{nd} = 0.01$$
$$B_0 = 0.65$$
$$a = 3.95$$
$$d = 1.26$$



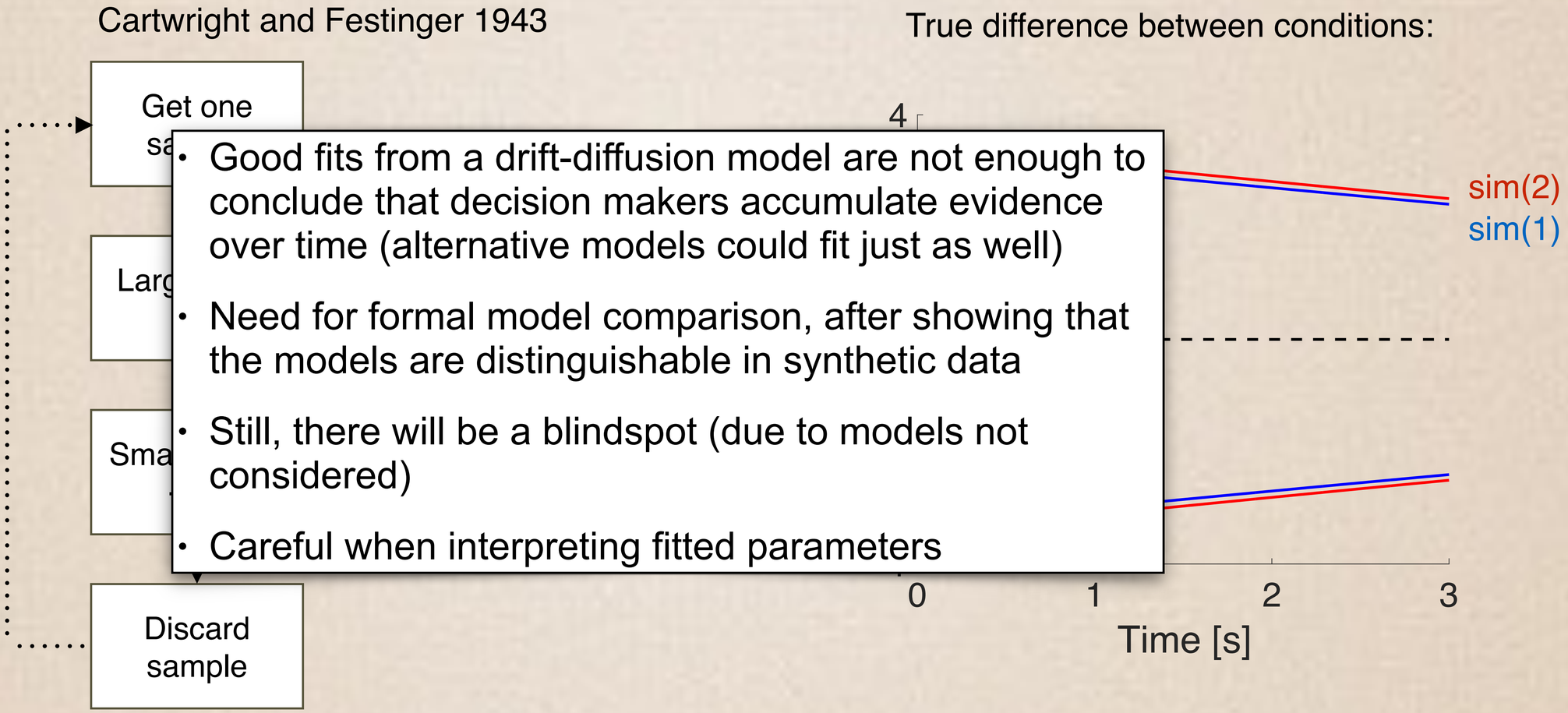
Fits of the DDM to the 2nd dataset

Best fitting parameters:

$\kappa = 24.34$	21
$\mu_{nd} = 0.35$	0.33
$\sigma_{nd} = 0.01$	0.01
$B_0 = 0.65$	2.43
$a = 3.95$	0.25
$d = 1.26$	-2.82



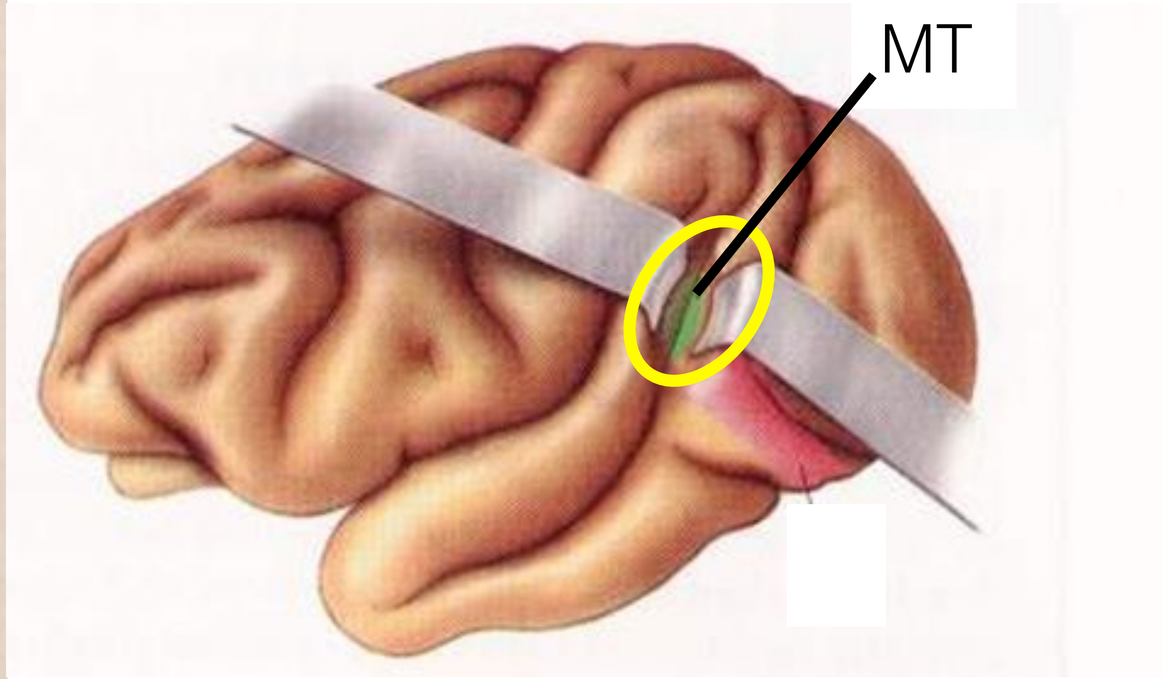
The data was generated with the ‘extrema’ model; only B_0 changed across conditions



- 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

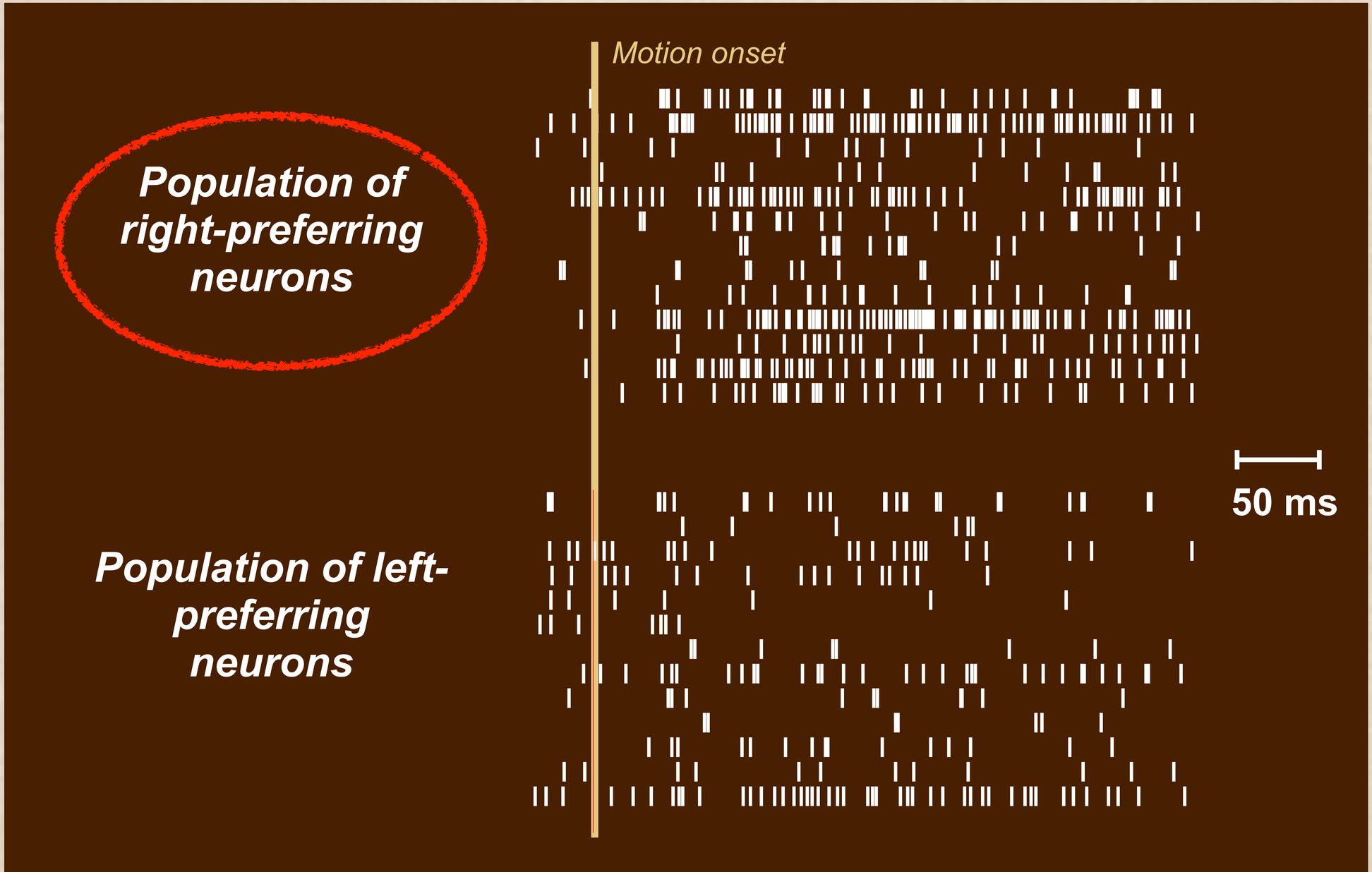
Neurophysiology
(Max)

Direction-selective neurons in the visual cortex

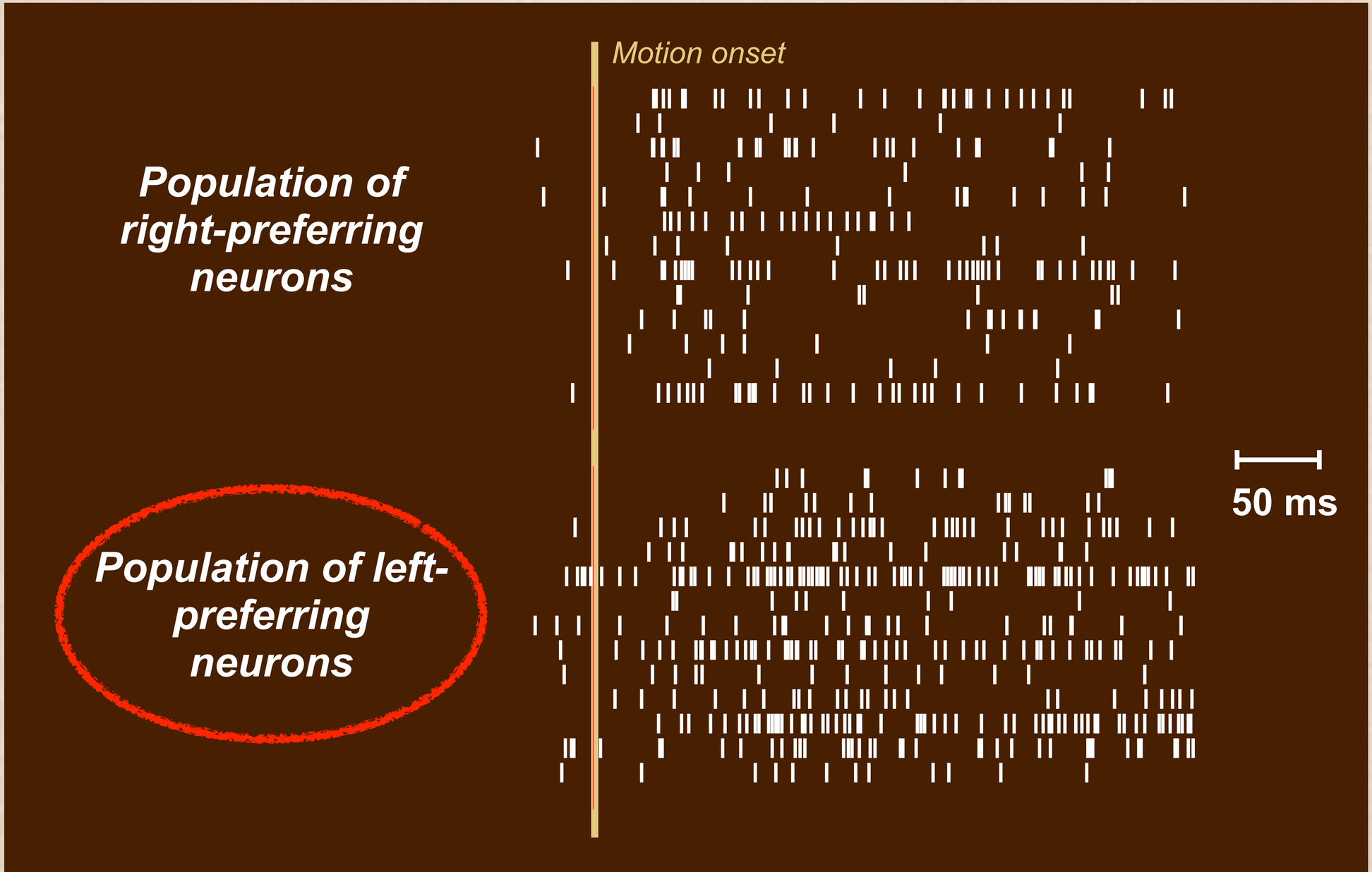


Albright, Movshon, Newsome, Britten, …

Is the motion rightward or leftward?



Is the motion rightward or leftward?

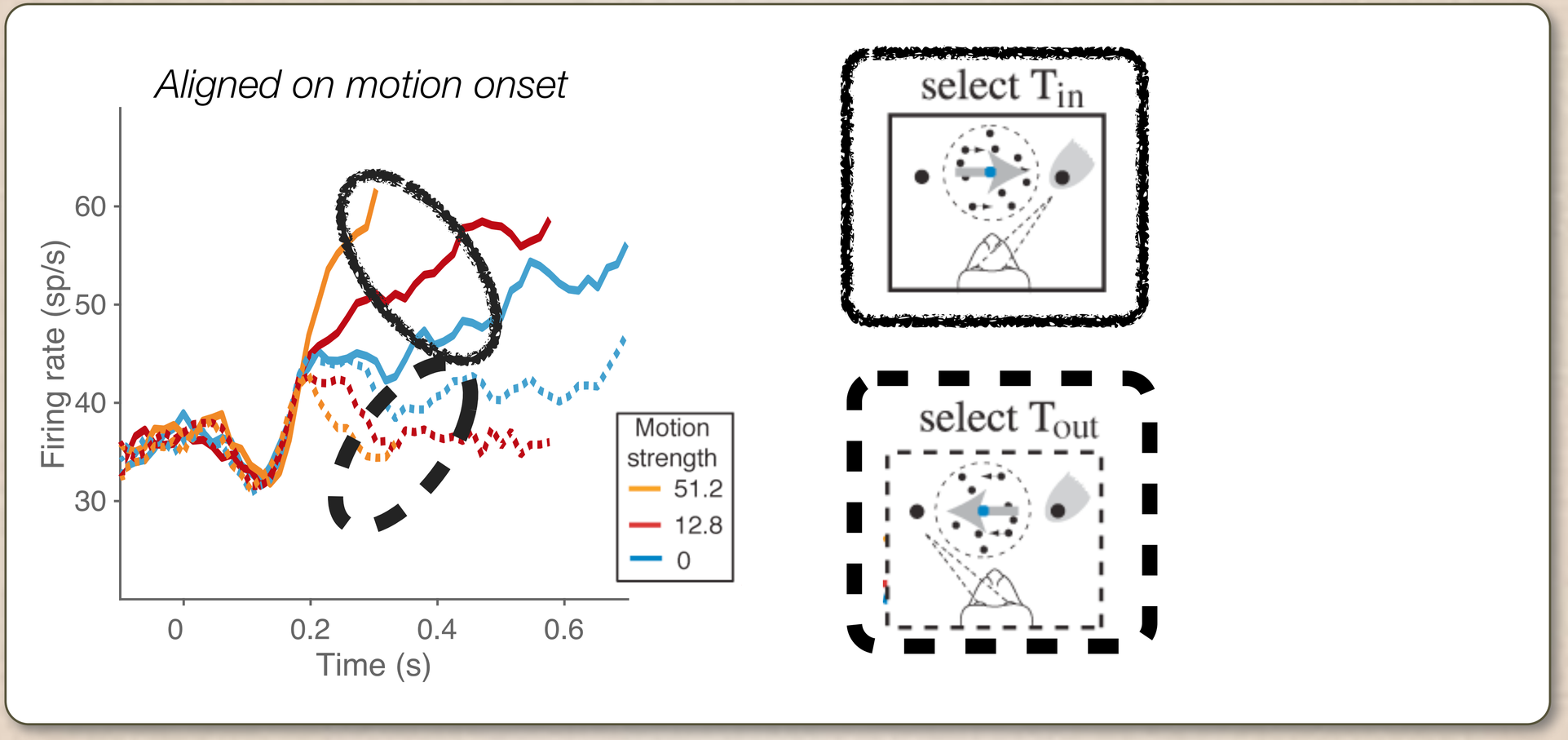


Decisions based on evidence streams

Population of right-preferring neurons

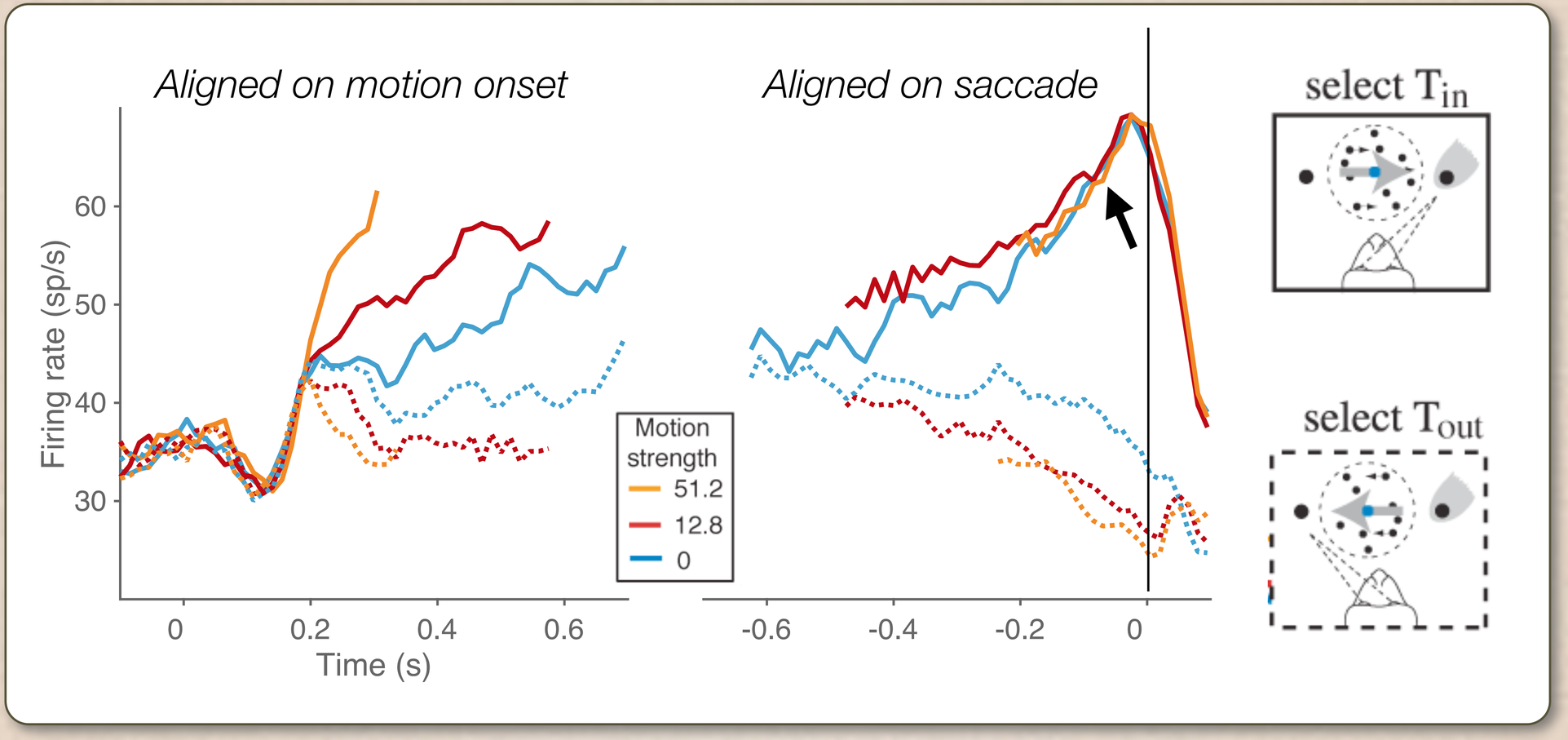
Population of left-preferring neurons

Average LIP activity in the random-dot motion task



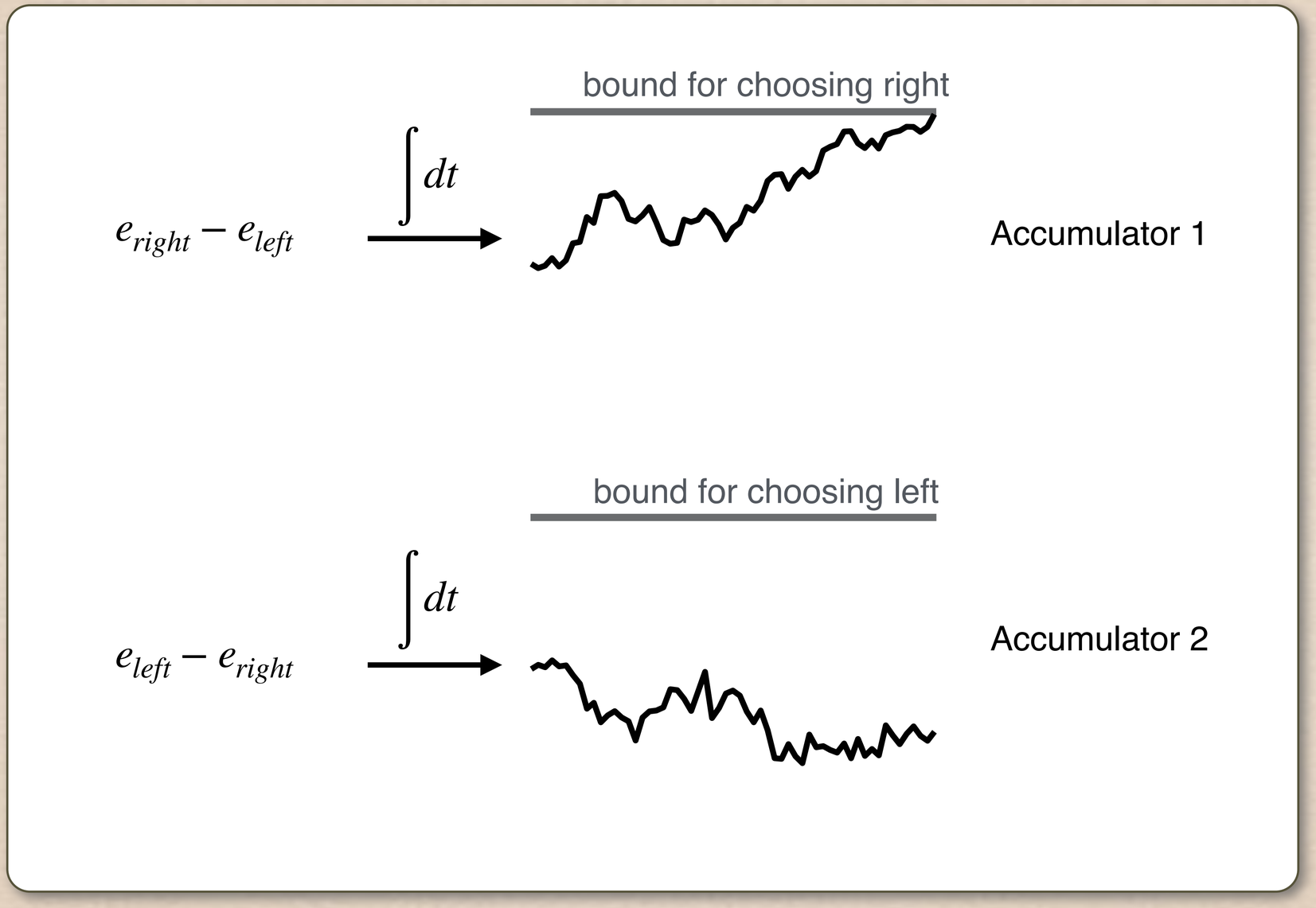
Roitman & Shadlen, 2002 J. Neurosci.

Average LIP activity in the random-dot motion task



Roitman & Shadlen, 2002 J. Neurosci.

Competing races



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