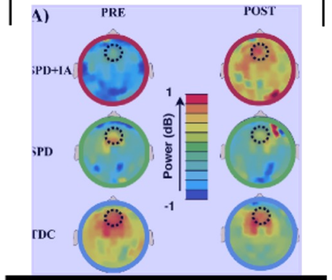

Medicine developed,
delivered, and *experienced*
in a completely new way.

2013

1st HUMAN RESULTS W/TECHNOLOGY PROTOTYPE

2016



1st HUMAN RESULTS W/COMMERCIAL CANDIDATE (SPD)

R&D

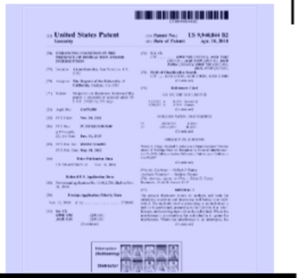
TECHNOLOGY STUDIED AS **SCREEN IN EARLY ALZHEIMER'S**

Akili and Pfizer successfully gamify Alzheimer's — FIERCE Biotech

TECHNOLOGY STUDIED AS A **MONITOR IN MULTIPLE SCLEROSIS**

Akili's game-based EVO Monitor shows promise for cognitive screening in MS: study — FirstWord HEALTHTECH

2018



1st PATENT GRANTED FOR AKILI'S UNIQUE DTX MECHANISM

2020

1st PIVOTAL TRIAL RESULTS IN SPECIFIC DISEASE POPULATION (ADHD)

PORTFOLIO EXPANSION

DEPRESSION RCT RESULTS PUBLISHED IN AJP

Akili study grabs attention with digital therapeutic for adults with MDD and new hire — BioWorld™

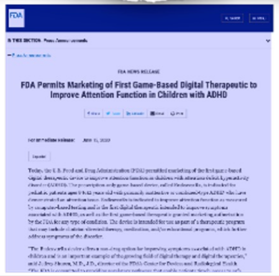
1ST TRIAL IN US TARGETING **COVID BRAIN FOG** COMMENCED

Doctors are testing a prescription video game for COVID-19 'brain fog — THE VERGE

MULTIPLE SCLEROSIS RCT RESULTS PUBLISHED IN NEUROLOGY AND THERAPY

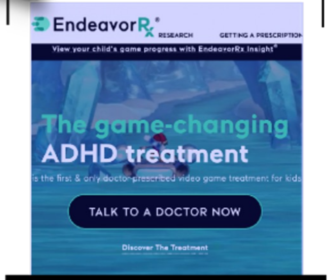
2020

In a landmark decision, FDA greenlights a video game for kids with ADHD — STAT+



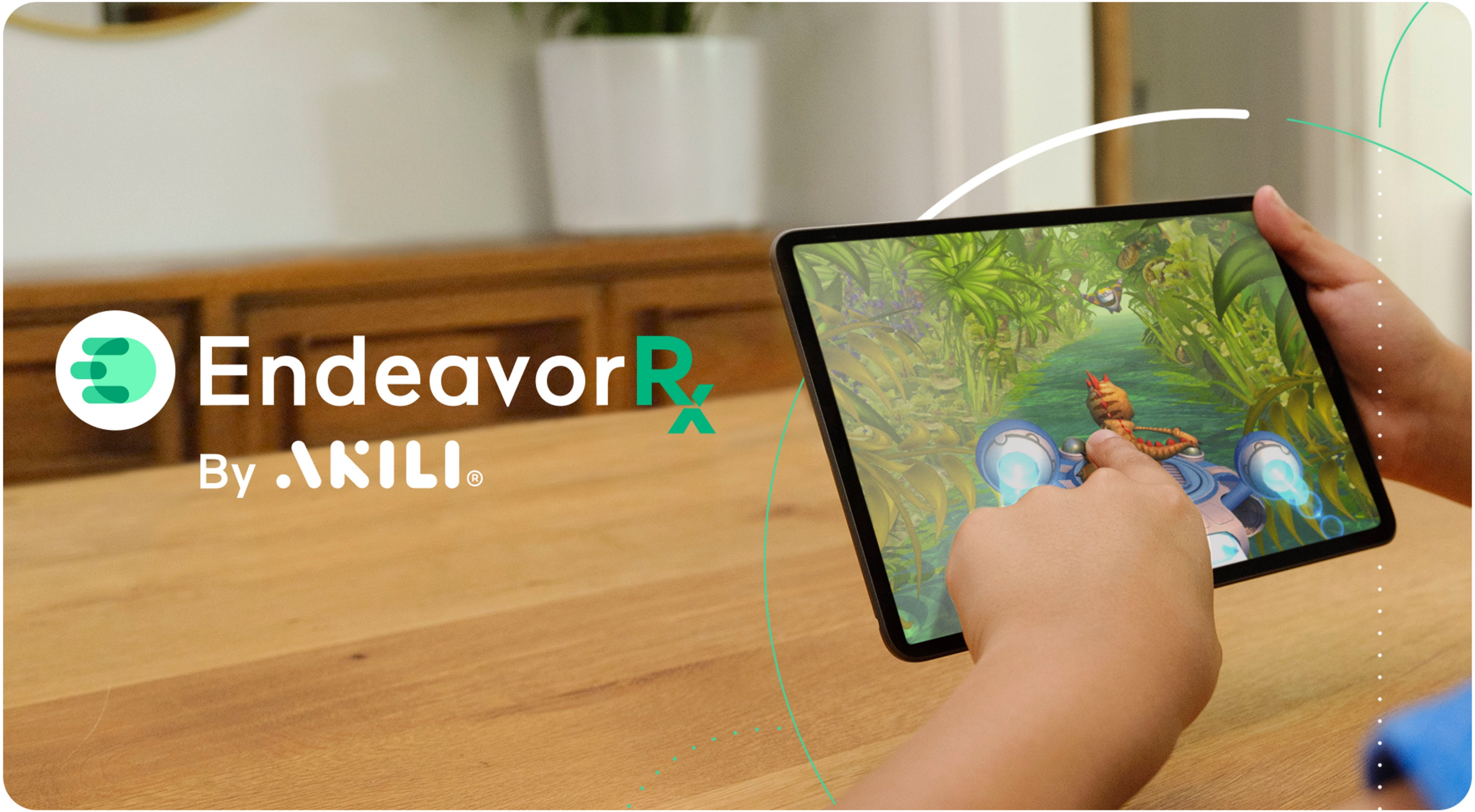
1st FDA CLEARANCE & CE MARK FOR FIRST PRODUCT FROM PLATFORM

2022



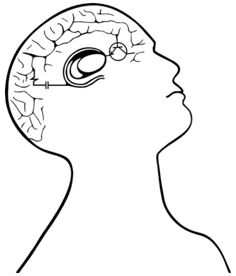
PURPOSE-BUILT DTX DISTRIBUTION AND COMMERCIAL INFRASTRUCTURE

AKILI


EndeavorRx
By AKILI®

EndearvorRx video skipped here to keep filesize manageable….

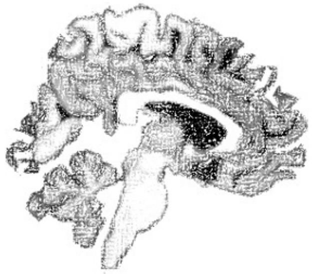
Likelihood Approximation Networks in PyMC

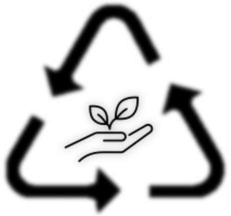


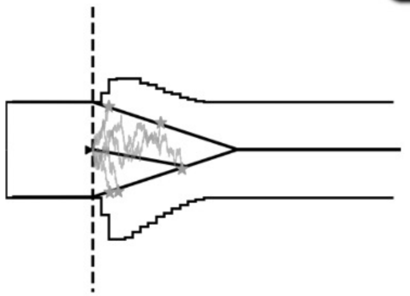
Alexander Fengler

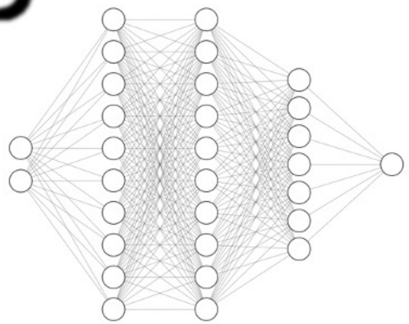
Ricardo Viera

01.12.2023









As a starting point…

Let's unpack a simplified version of the EndeavorRx® game

PyMC Labs

NeuroRacer

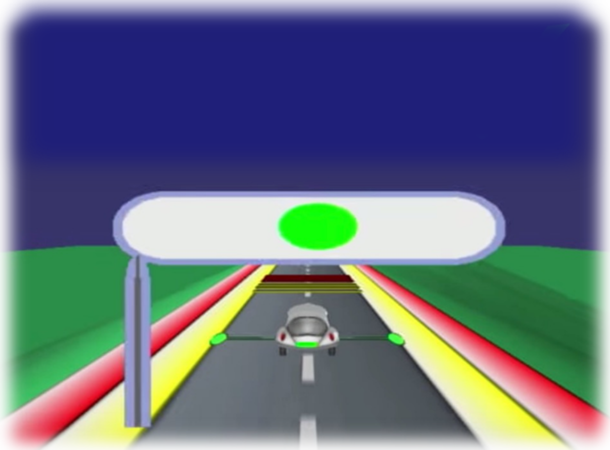
We are driving
around a circuit!

https://cociwg.org/blog/2014/5/17/exercising-the-mind-to-treat-attention-deficits —— Picture

Anguera, J. A., Boccanfuso, J., Rintoul, J. L., Al-Hashimi, O., Faraji, F., Janowich, J., ... & Gazzaley, A. (2013). Video game training enhances cognitive control in older adults. Nature, 501(7465), 97-101. —— NeuroRacer main research paper

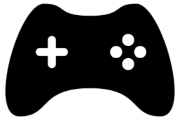
PyMC Labs

NeuroRacer

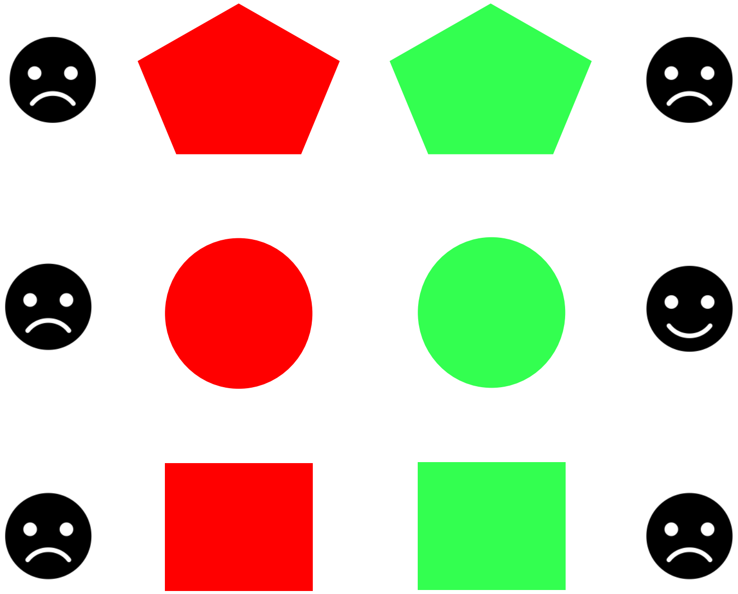


We are driving around a circuit!

Road-sign appears!

Drive! 

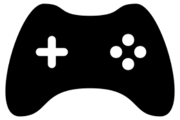
NeuroRacer



We are driving around a circuit!

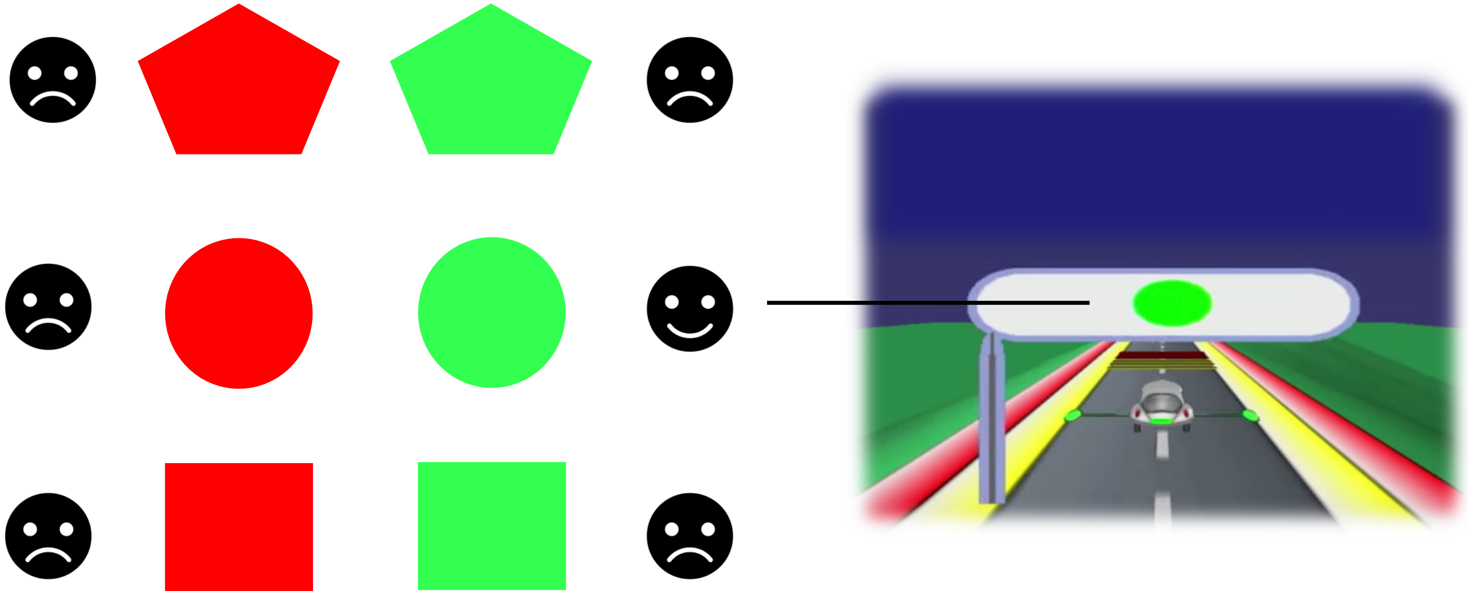
Road-sign appears!

Drive!



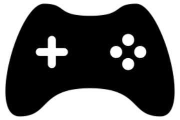
Target or no Target?

NeuroRacer



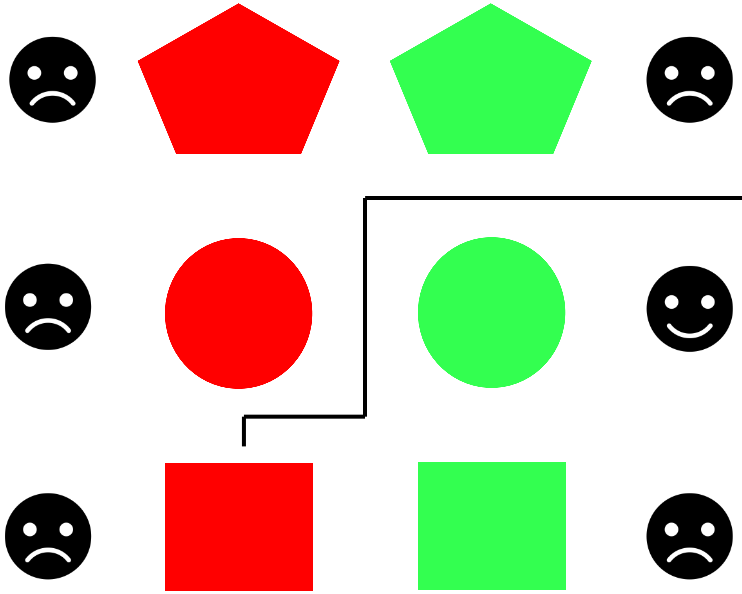
We are driving around a circuit!

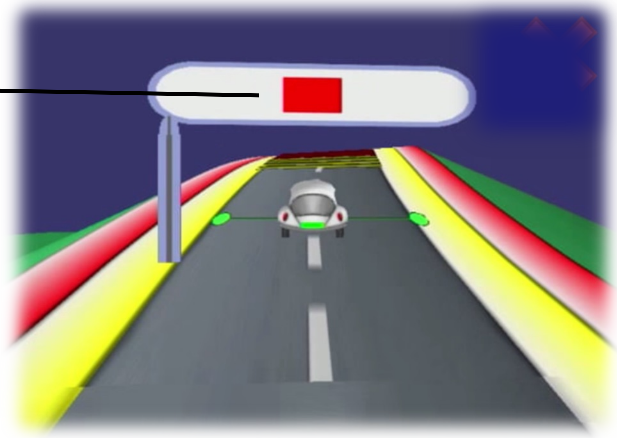
Road-sign appears!

Drive!  Press!

Target or no Target?

NeuroRacer

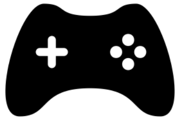




We are driving around a circuit!

Road-sign appears!

Drive!



Don't press!

Target or no Target?

NeuroRacer

Trials

Drive!

Press!

Don't press!

Target or no Target?

Leaves us with <u>four</u> <u>types</u> of <u>responses</u>!

PyMC Labs

NeuroRacer

Trials

Drive! — Press! / Don't press!

Target or no Target?

	Good Symbol	Bad Symbol
Press	CORRECT Go	FALSE Go
Don't Press	FALSE NoGo	CORRECT NoGo

Leaves us with four types of responses!

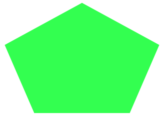
PyMC Labs

NeuroRacer

Trials

Drive! — Press! / Don't press!

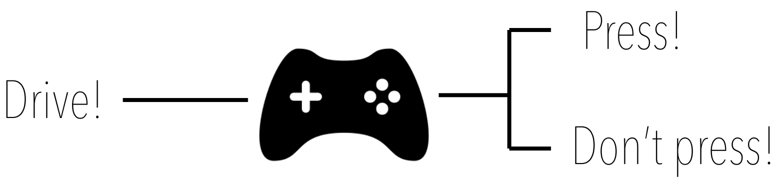
Target or no Target?



	Good Symbol	Bad Symbol	
Press	CORRECT Go	FALSE Go	observe rt/choice
Don't Press	FALSE NoGo	CORRECT NoGo	observe only 'choice'

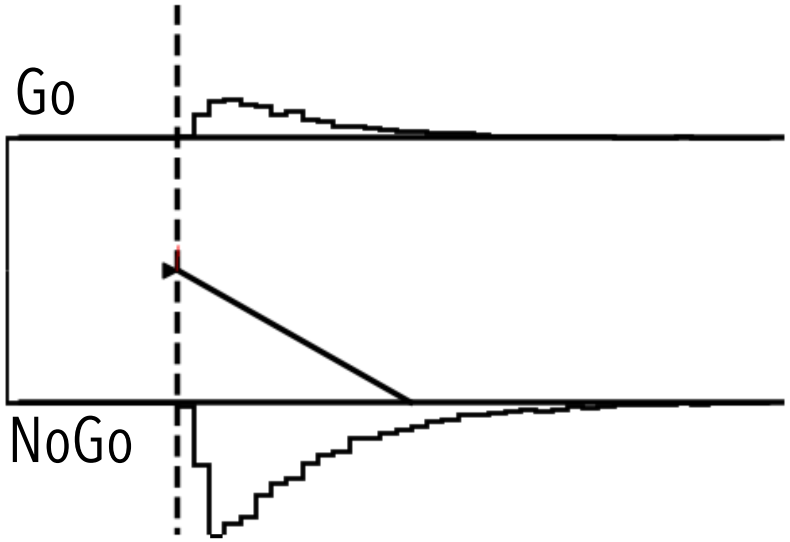
Leaves us with four types of responses!

NeuroRacer

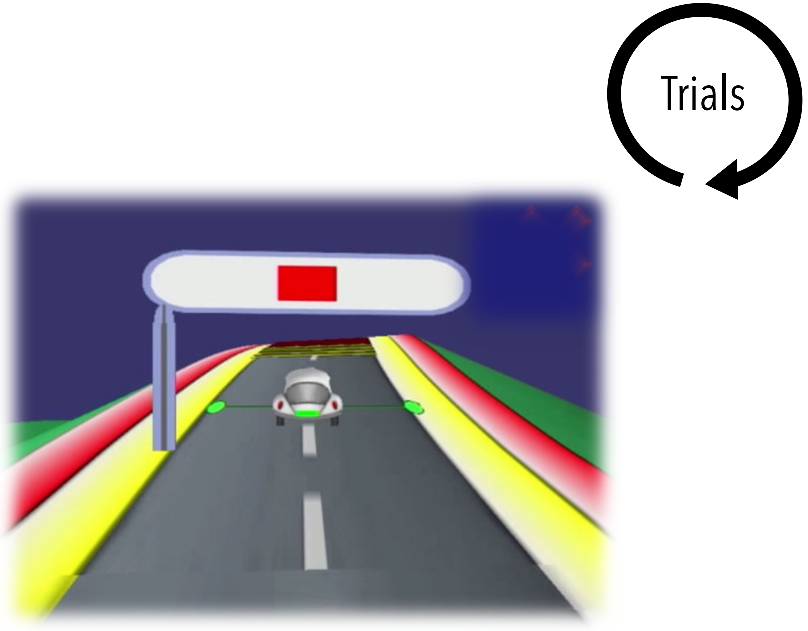


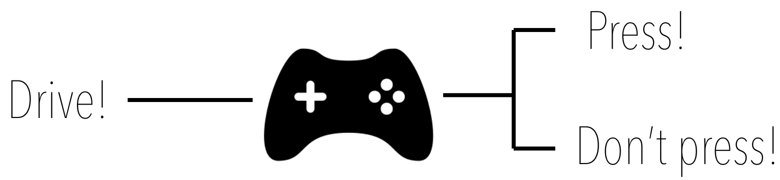
Target or no Target?

A Model



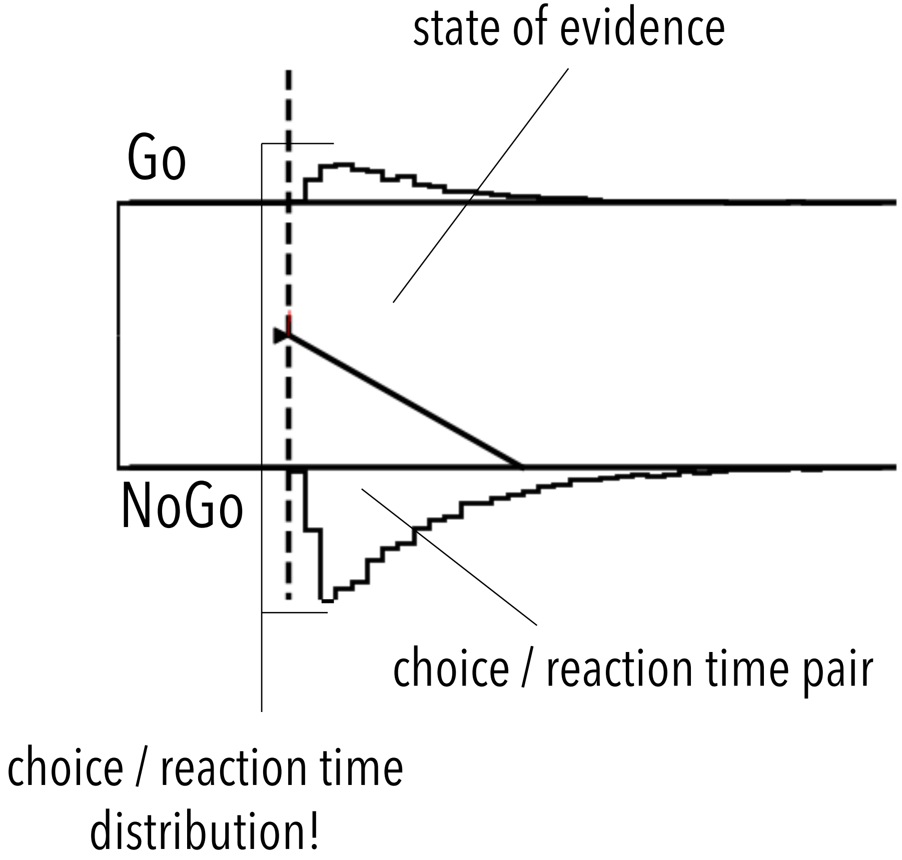
NeuroRacer





Target or no Target?

A Model



Our Model

Our Data

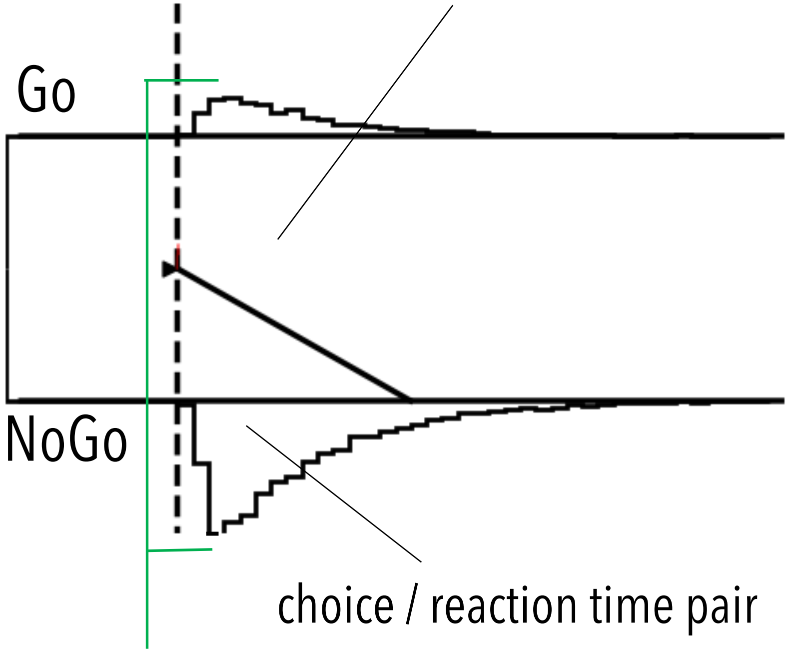
Parameters

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

Posterior

Prior

Likelihood



choice / reaction time distribution!

Our Model

Our Data

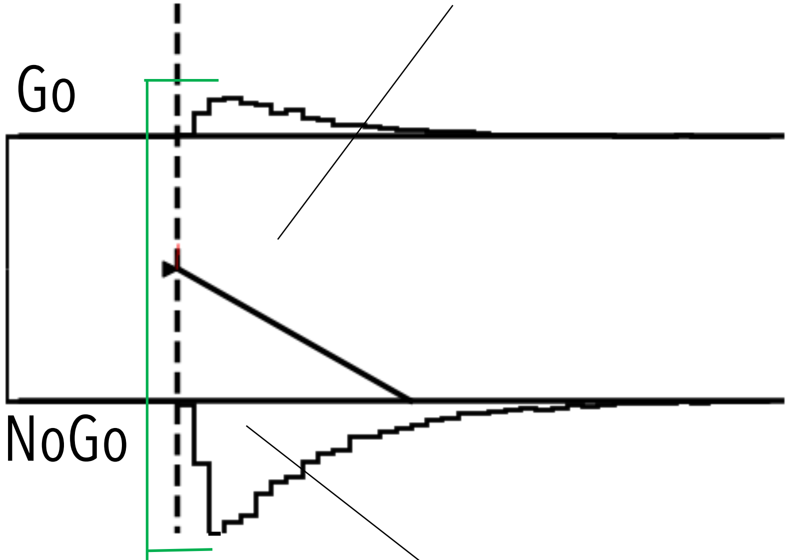
Parameters

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

Posterior

Prior

Likelihood



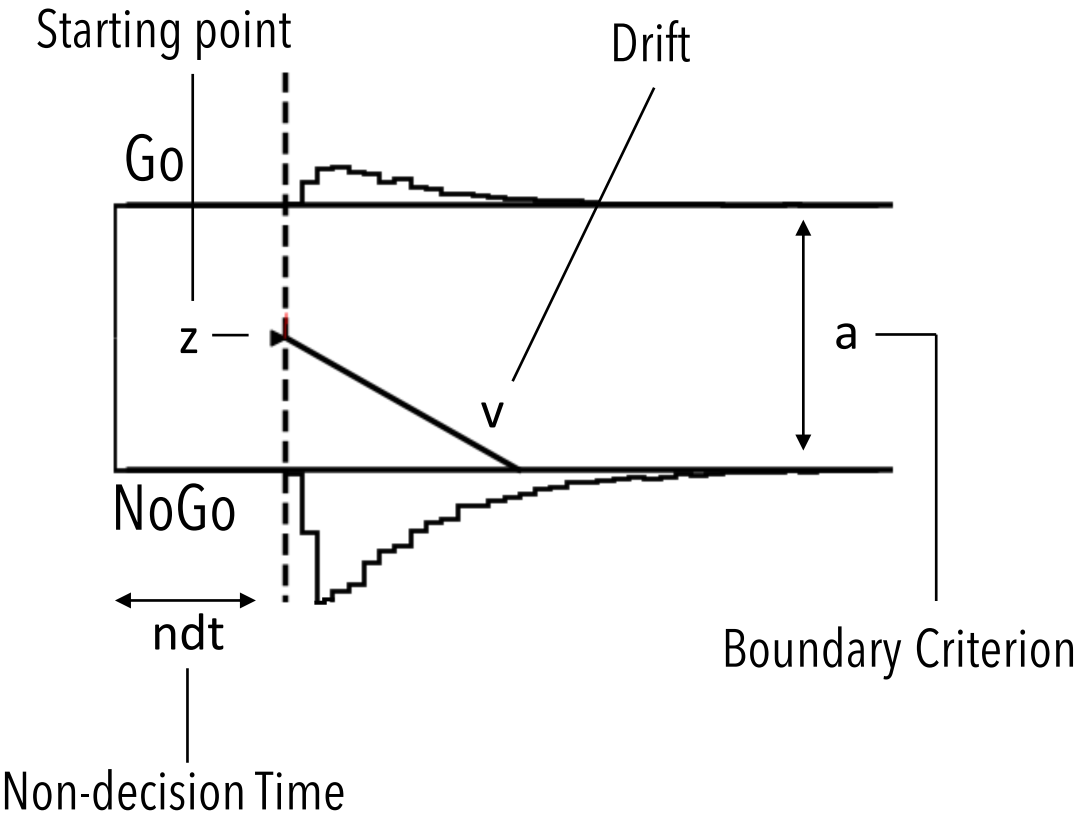
choice / reaction time distribution!

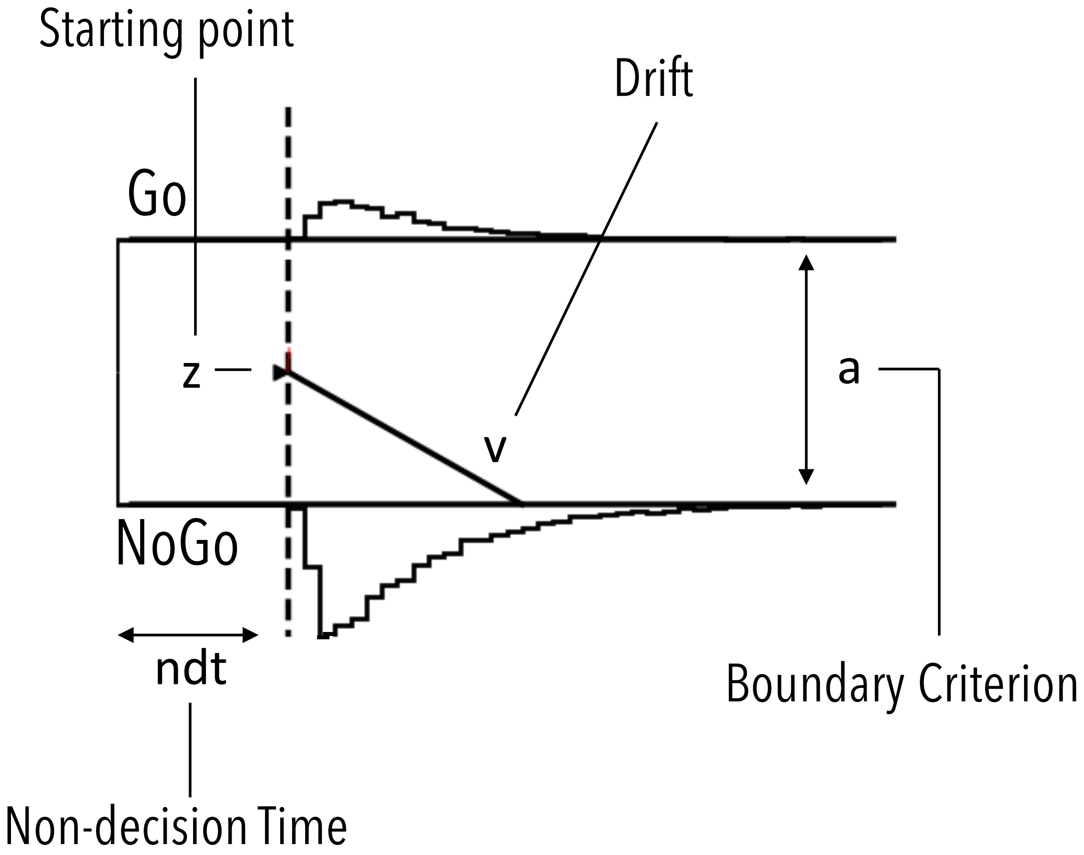
Our Model

Our Data

Parameters

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

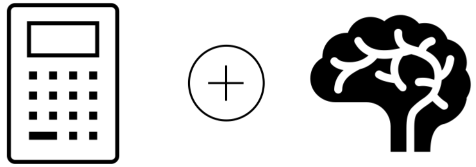




2 Important Aspects of the Model

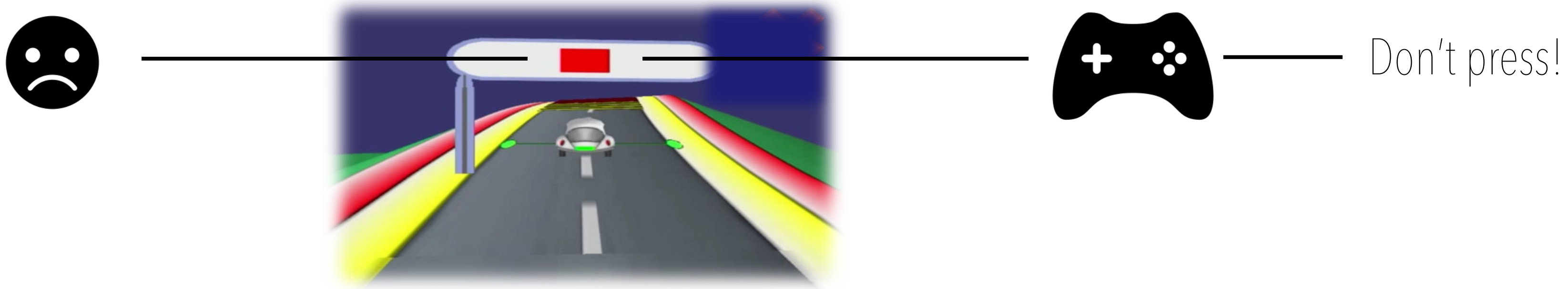
1. Parameters interpretable
2. Special case of a whole class of related models

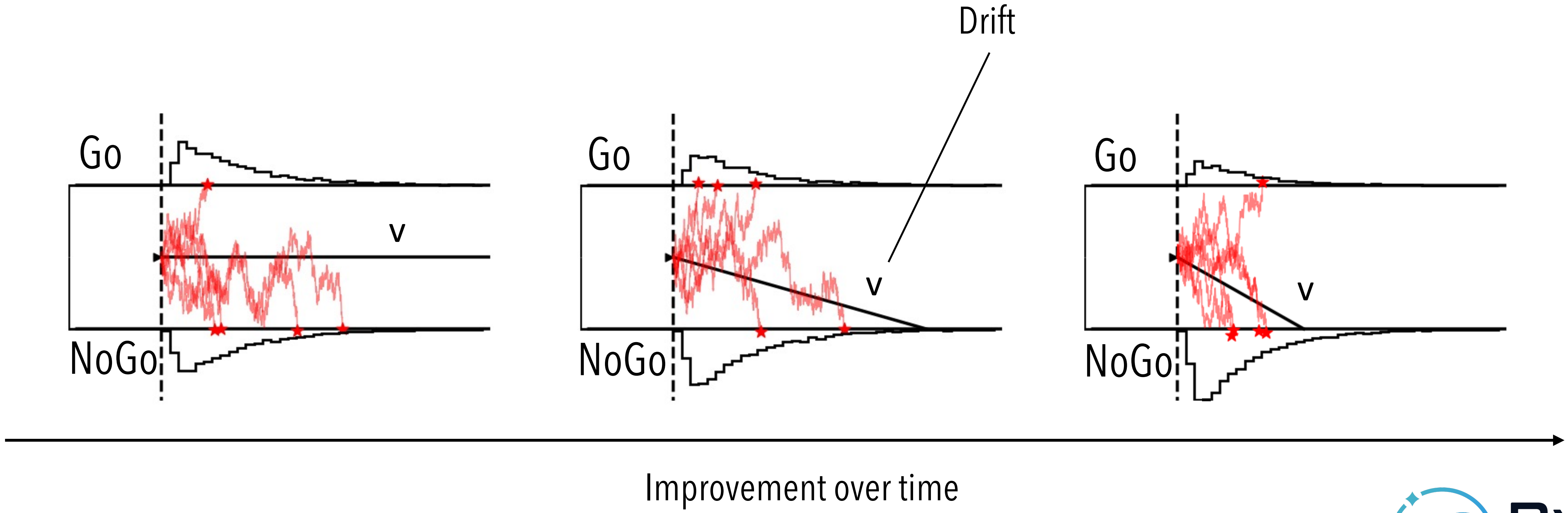
The model is abstract but designed to capture separable aspects of a cognitive process!



PyMC Labs

Speed of processing / Evidence per second

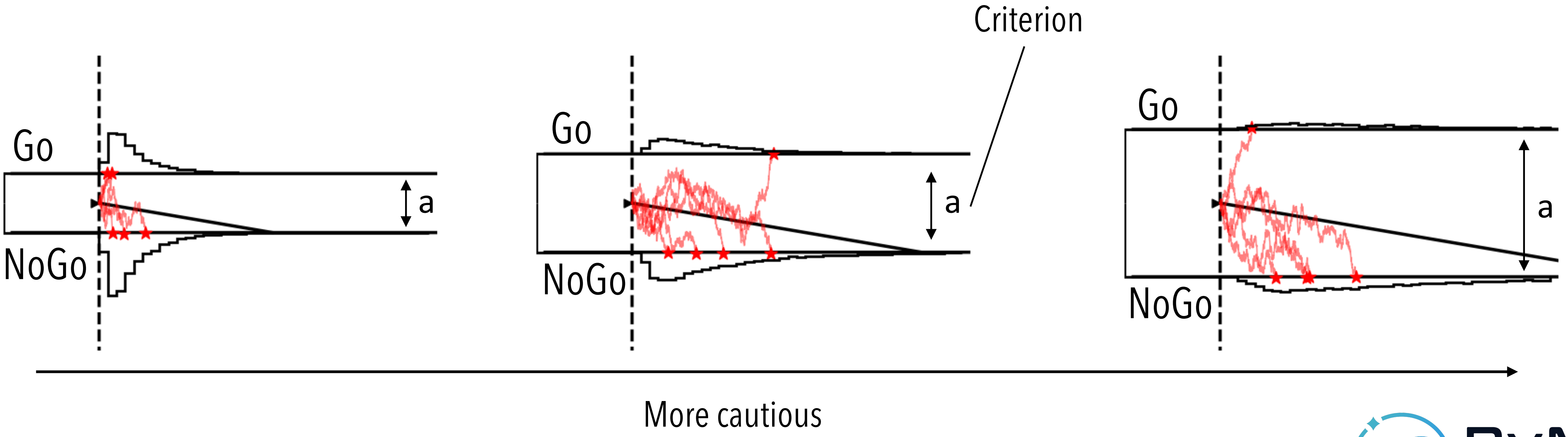




Speed accuracy trade-off

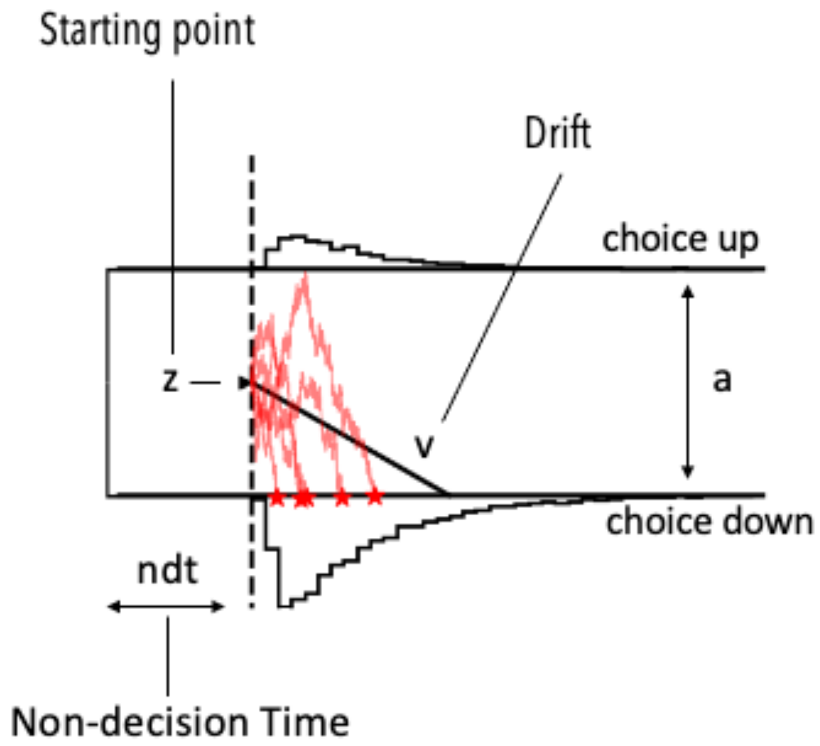
More mistakes but
shorter reaction times

Less mistakes, but
longer reaction times



Very successful modeling paradigm

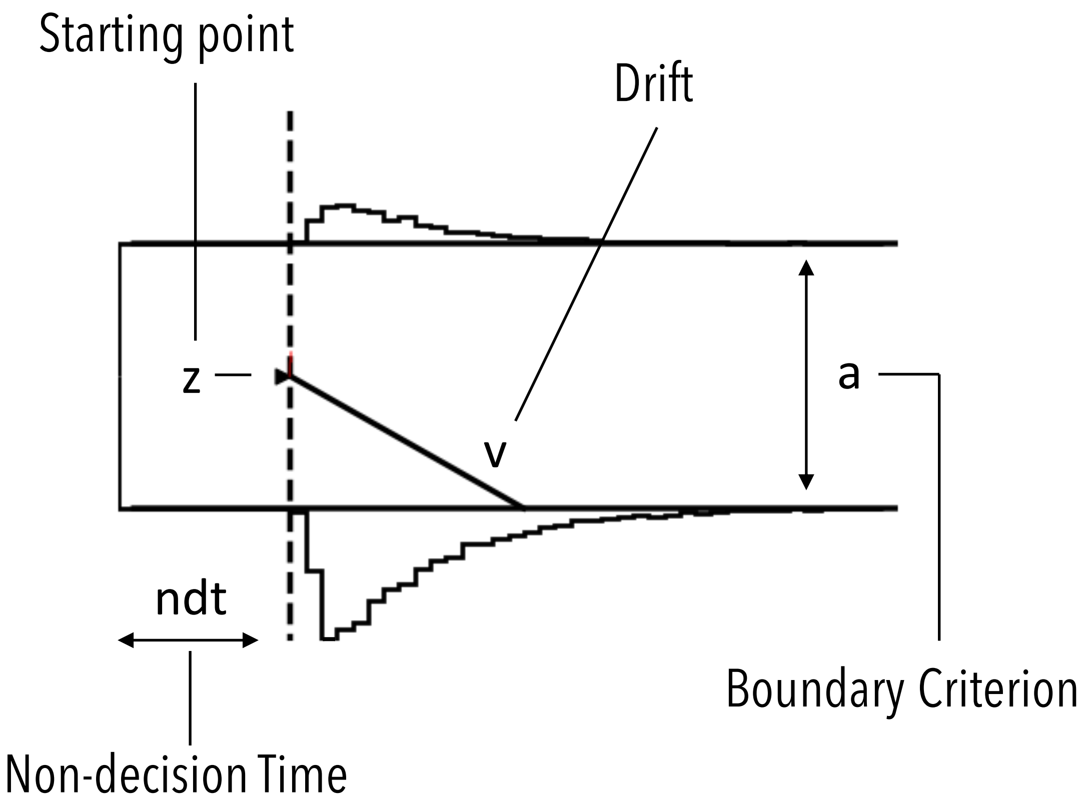
Widely applied with 1000s of publications across many different experiment modalities!



But it is does not capture all aspects of the task which are of interest to us!

PyMC Labs

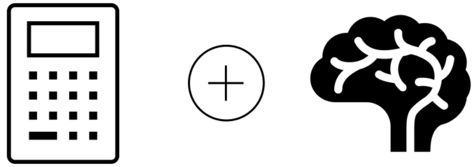
Our Model



Our Model

1. Parameters interpretable
2. Special case of a whole class of related models

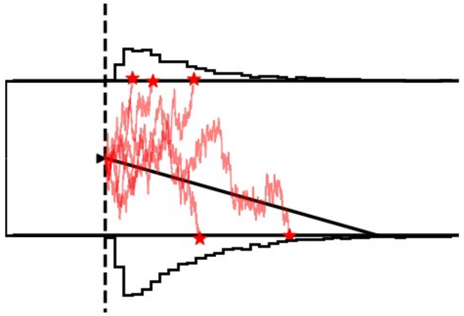
The model is abstract but designed to capture separable aspects of a cognitive process!



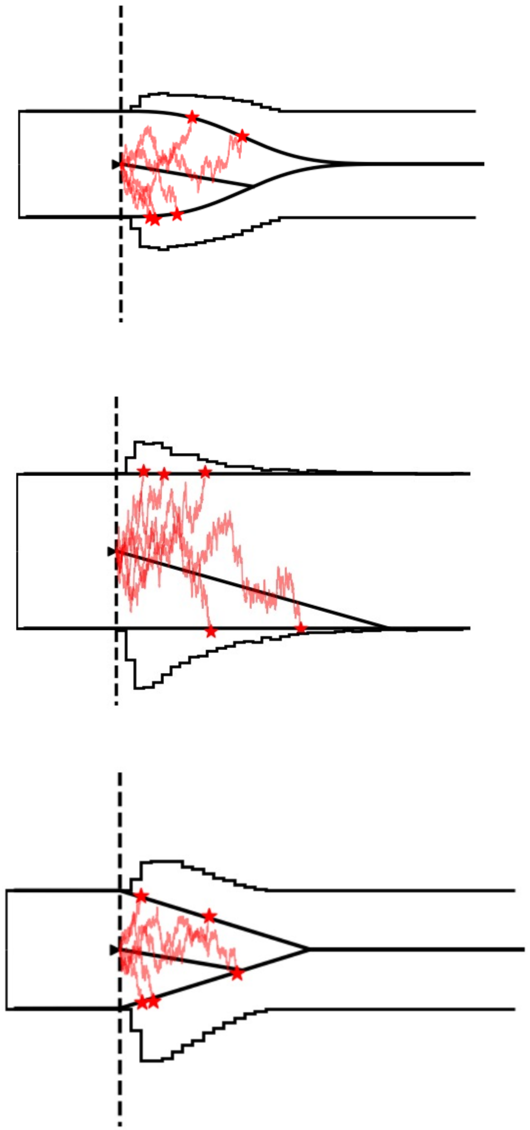
There is a deadline to the response here:

Players might want to enforce a choice by compromising accuracy towards the end of the acceptable reaction time window!

Our model implies a constant evidence threshold over time…



Our model implies a constant evidence threshold over time…



There is a deadline to the response here:

Players might want to enforce a choice by compromising accuracy towards the end of the acceptable reaction time window!

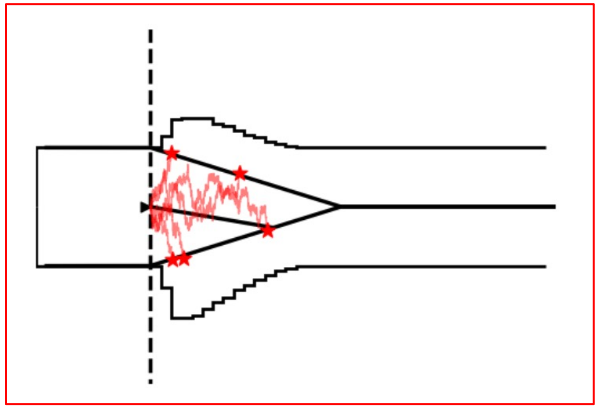
These models might be better suited to model some aspects of the game!

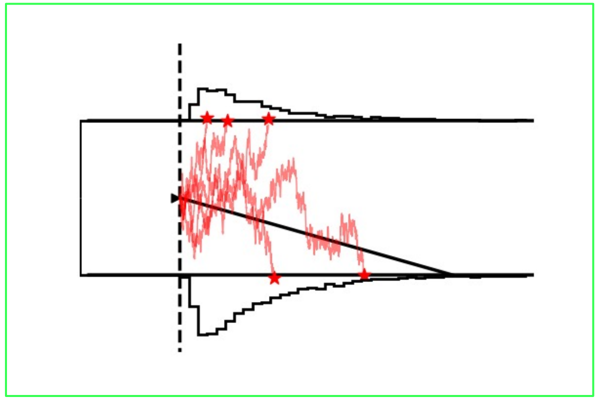
We want to do inference with these model variants!

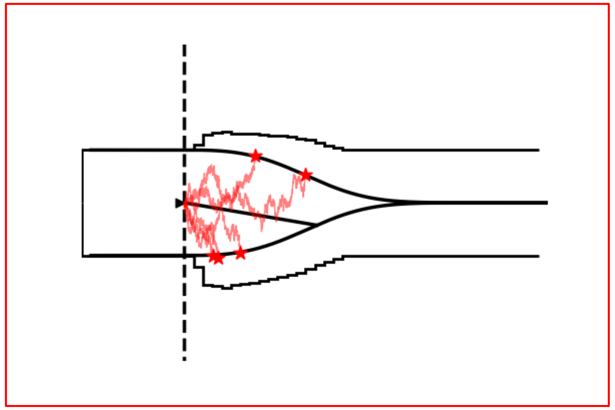
But there is a fundamental problem…

Derivation of closed-form likelihoods is a lot harder!

Without likelihoods, no Bayes' Rule…







Simulation is easy however!

We want to do inference with these model variants!

Inference from access to simulators?

Field with a long history.

Many recent advances!

Approximate Bayesian Computation (ABC)!
[These days: Simulation Based Inference (SBI)]

Marjoram, P., Molitor, J., Plagnol, V., & Tavaré, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, *100*(26), 15324-15328. — Traditional ABC

Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, *117*(48), 30055-30062. — Overview, modern approaches

We want to do inference with these model variants!

Inference from access to simulators?

Field with a long history.

Many recent advances!

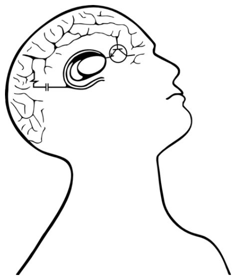
Approximate Bayesian Computation (ABC)!
[These days: Simulation Based Inference (SBI)]

We will use one recent technique based on Neural Networks
(The PyMC workflow allows other techniques to be substituted in)

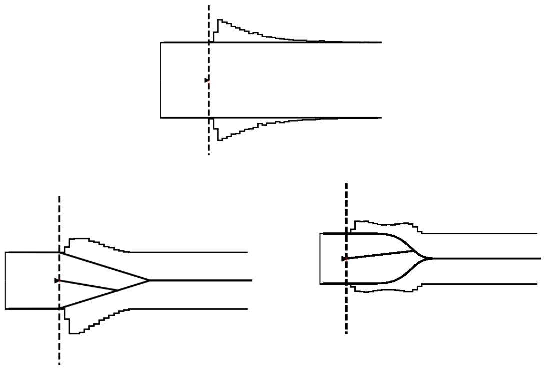
Marjoram, P., Molitor, J., Plagnol, V., & Tavaré, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, *100*(26), 15324-15328. — Traditional ABC

Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, *117*(48), 30055-30062. — Overview, modern approaches

Fengler, A., Govindarajan, L. N., Chen, T., & Frank, M. J. (2021). Likelihood approximation networks (LANs) for fast inference of simulation models in cognitive neuroscience. *Elife*, *10*, e65074. — Our approach



We want to do inference with these model variants!

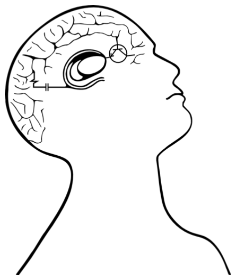


Run simulations

|

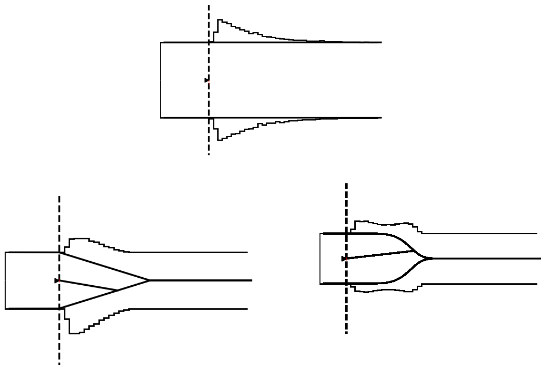
Costly Once

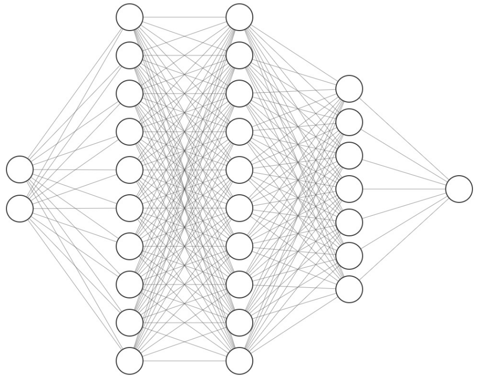
https://elifesciences.org/articles/65074



We want to do inference with these model variants!

LAN





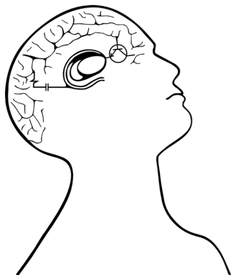
Run simulations

|

Costly Once

Train Neural Network
to Represent Approximate
Likelihood

https://elifesciences.org/articles/65074

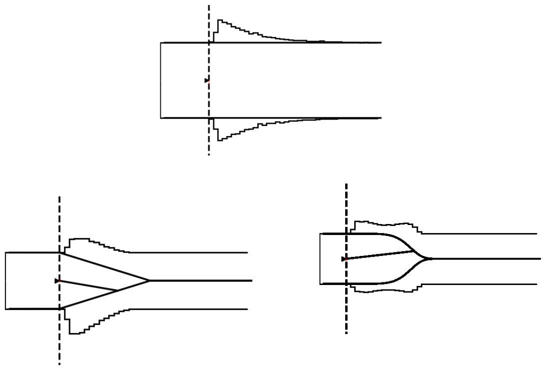


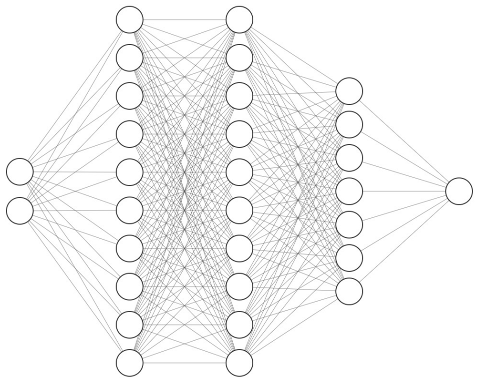
We want to do inference with these model variants!

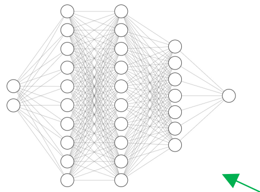
ROBERT J. & NANCY D. CARNEY
INSTITUTE FOR BRAIN SCIENCE
BROWN UNIVERSITY

COGNITIVE LINGUISTIC & PSYCHOLOGICAL SCIENCES · BROWN UNIVERSITY

LAN







$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

Run simulations
|
Costly Once

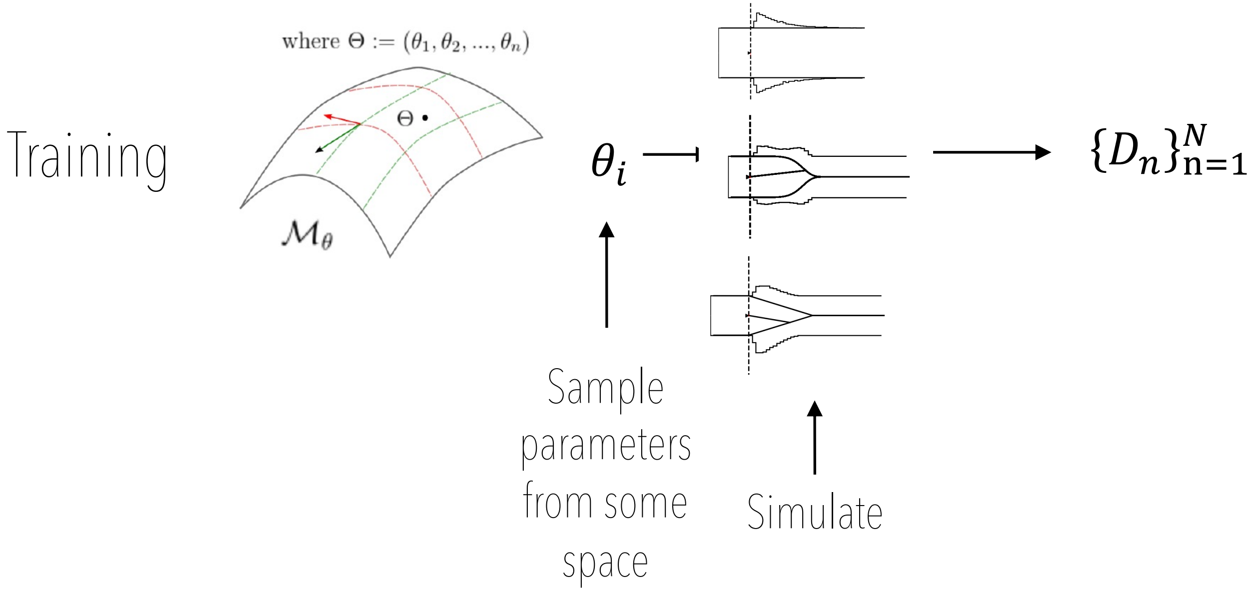
Train Neural Network
to Represent Approximate
Likelihood

(Re)Use for Inference
|
Cheap

https://elifesciences.org/articles/65074

PyMC Labs

Training



https://elifesciences.org/articles/65074

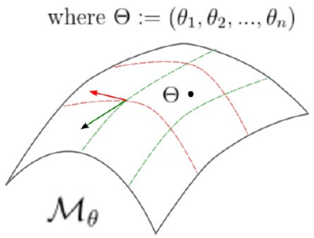
Training

Label

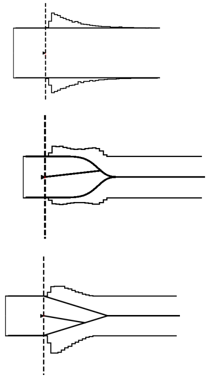
$log p_M^{Hist}(D_n; \theta_i)$

Construct and **evaluate**
Histogram!

Training



θ_i



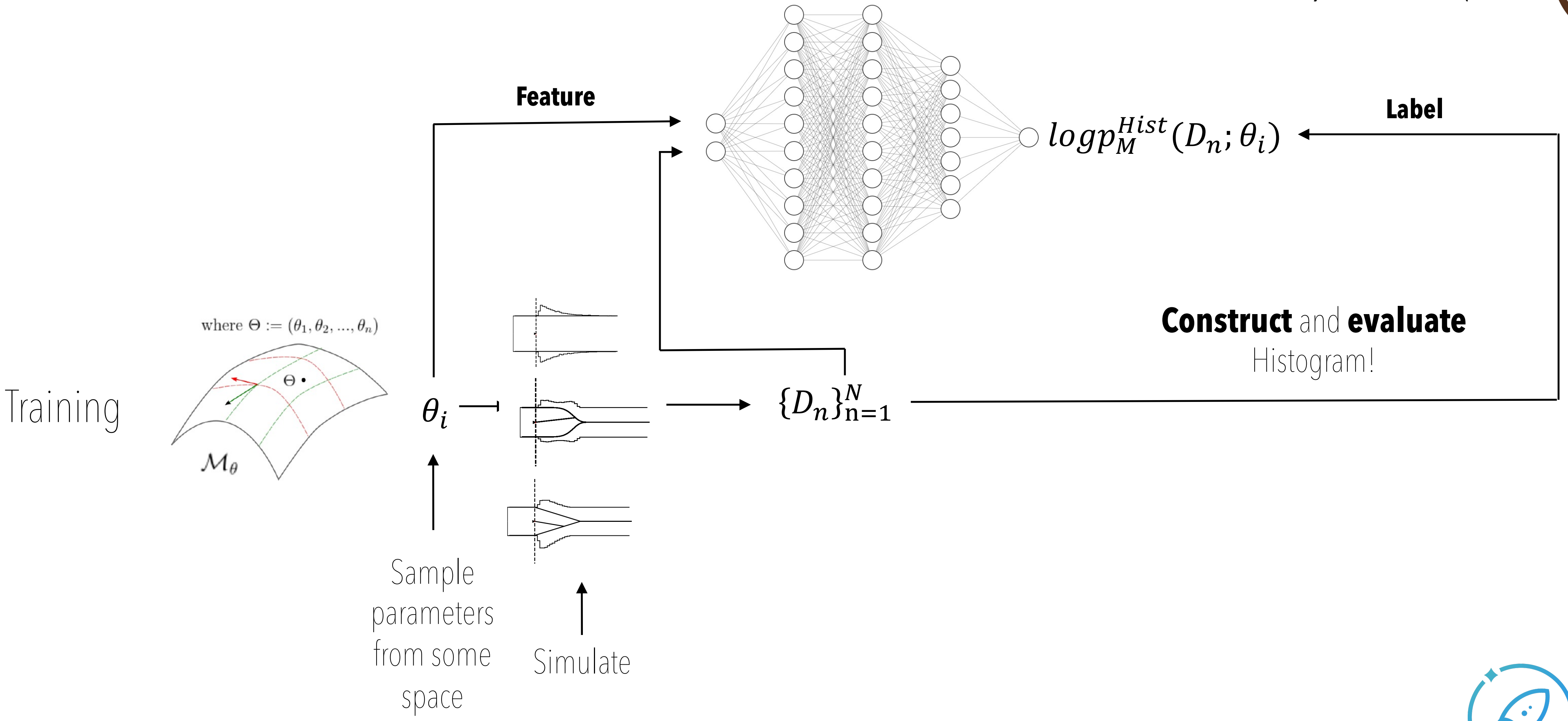
$\{D_n\}_{n=1}^N$

Sample parameters from some space

Simulate

https://elifesciences.org/articles/65074

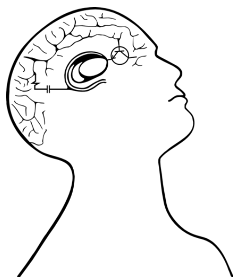
Training



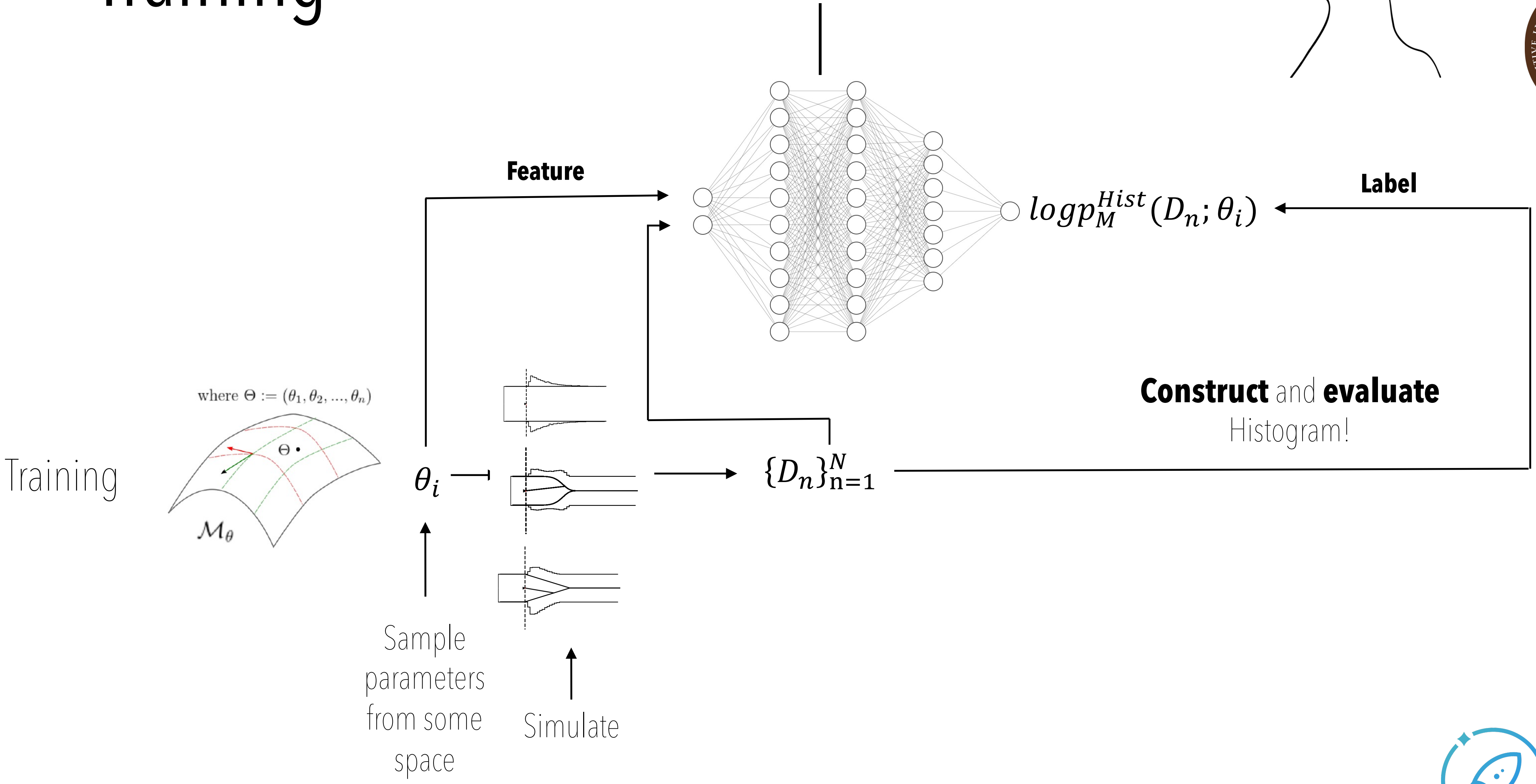
https://elifesciences.org/articles/65074

Training

We made this previously available
through a separate toolbox: HDDM



https://direct.mit.edu/jocn/article-abstract/34/10/1780/112585/



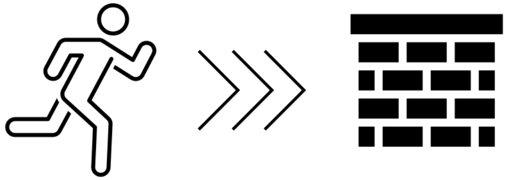
https://elifesciences.org/articles/65074

Training

We made this previously available
through a separate toolbox: HDDM

https://direct.mit.edu/jocn/article-abstract/34/10/1780/112585/

In our joint work with Akili we ran into
the limitations of this toolbox



It relies on an outdated backend
which compromises forward
compatibility and performance!

Training

We made this previously available
through a separate toolbox: HDDM

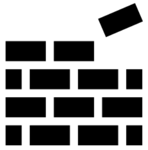
https://direct.mit.edu/jocn/article-abstract/34/10/1780/112585/

In our joint work with Akili we ran into
the limitations of this toolbox

It relies on an outdated backend
which compromises forward
compatibility and performance!

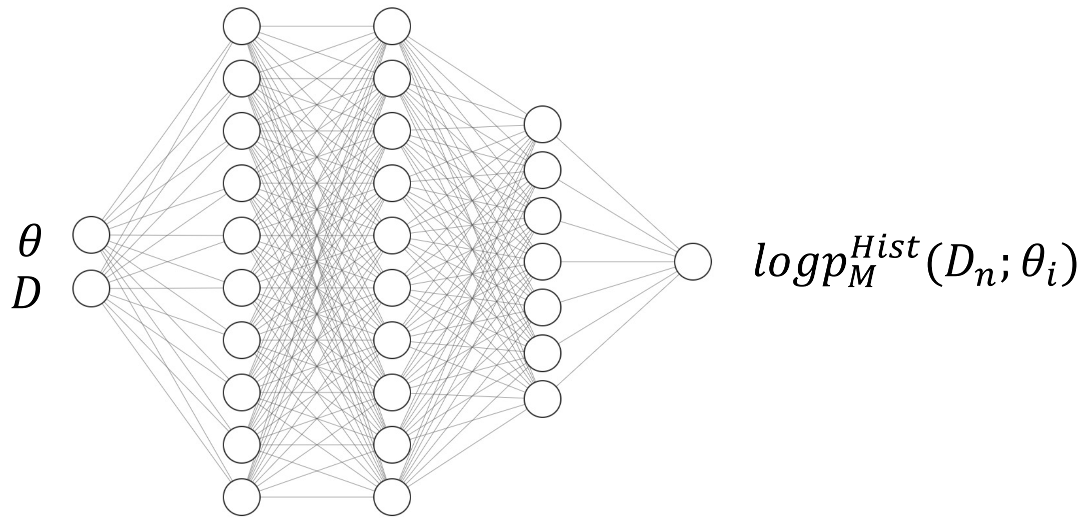
Break roadblocks by relying on
modern backend!

PyMC



Properties inherited from Neural Networks

LAN



1. Differentiable with respect to inputs

$$\nabla_\theta log p_M^{Hist}(D_n; \theta_i)$$

2. Speed via batching across datapoints

https://elifesciences.org/articles/65074

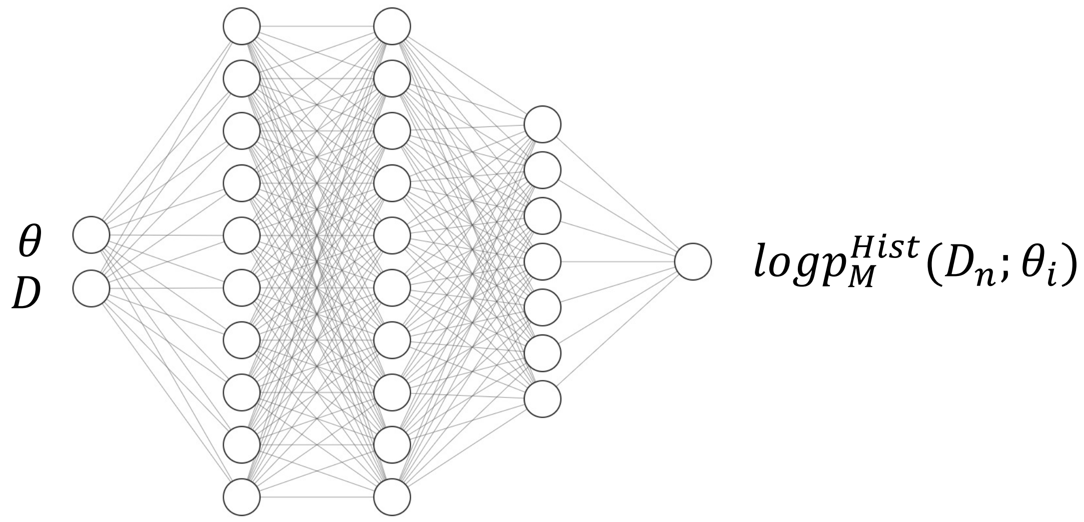
Properties inherited from Neural Networks

$$\nabla_\theta log p_M^{Hist}(D_n;\theta_i)$$

PyTensor

JAX

LAN



https://elifesciences.org/articles/65074

Properties inherited from Neural Networks

$$\nabla_\theta logp_M^{Hist}(D_n; \theta_i)$$

PyTensor

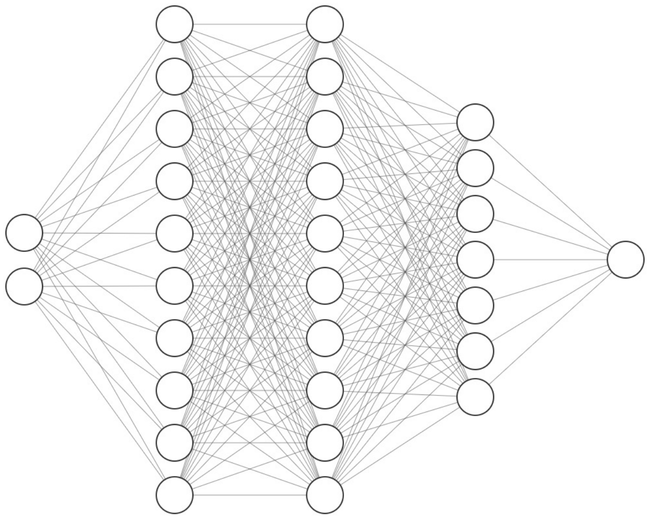
PyMC

Hamiltonian Monte Carlo

LAN

θ

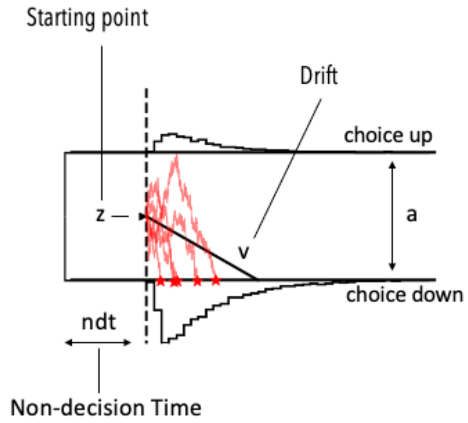
D

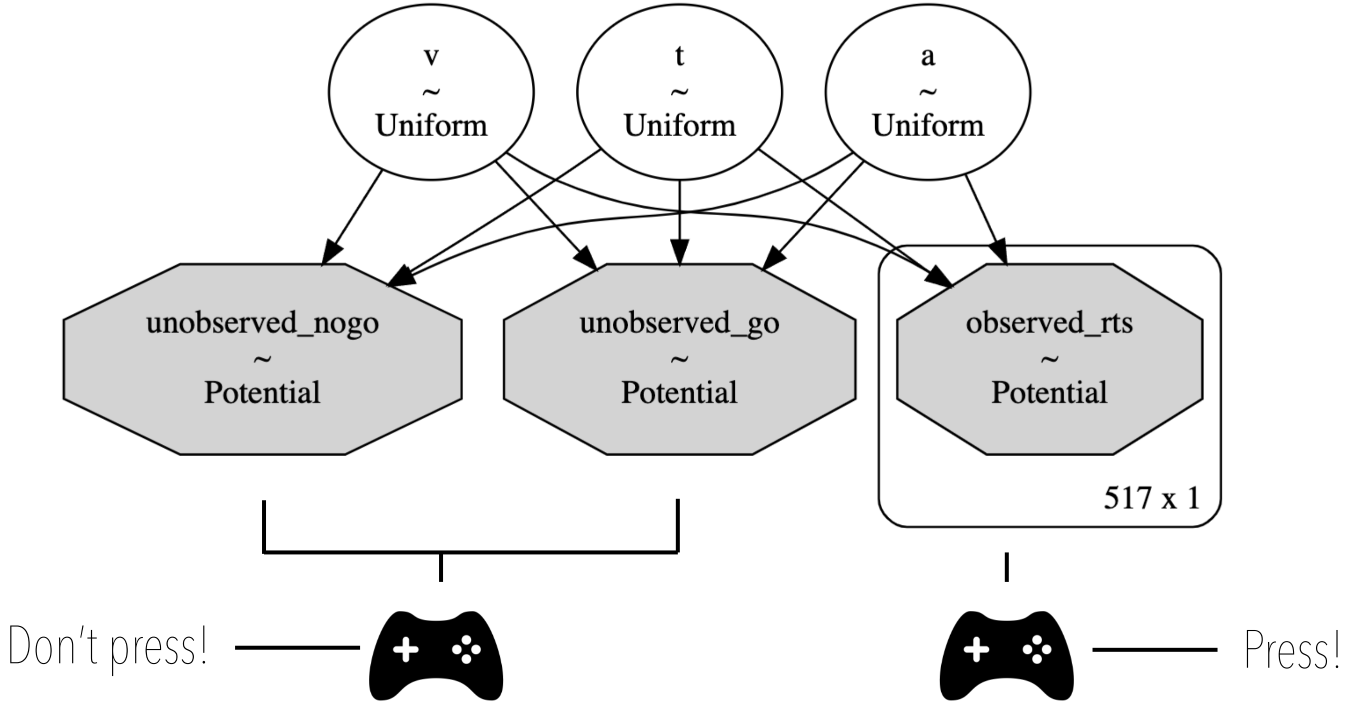


$logp_M^{Hist}(D_n; \theta_i)$

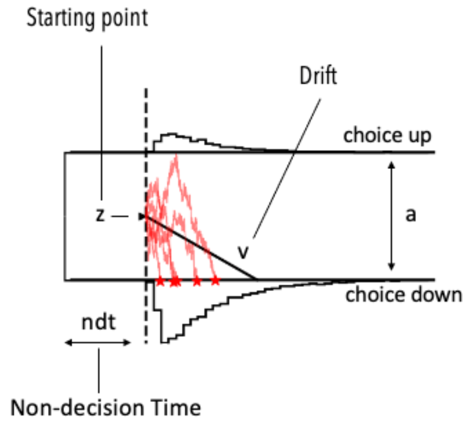
https://elifesciences.org/articles/65074

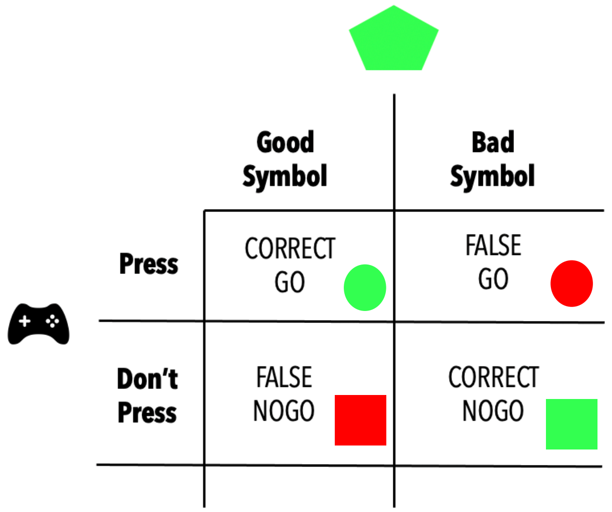
Let's look at all this through PyMC

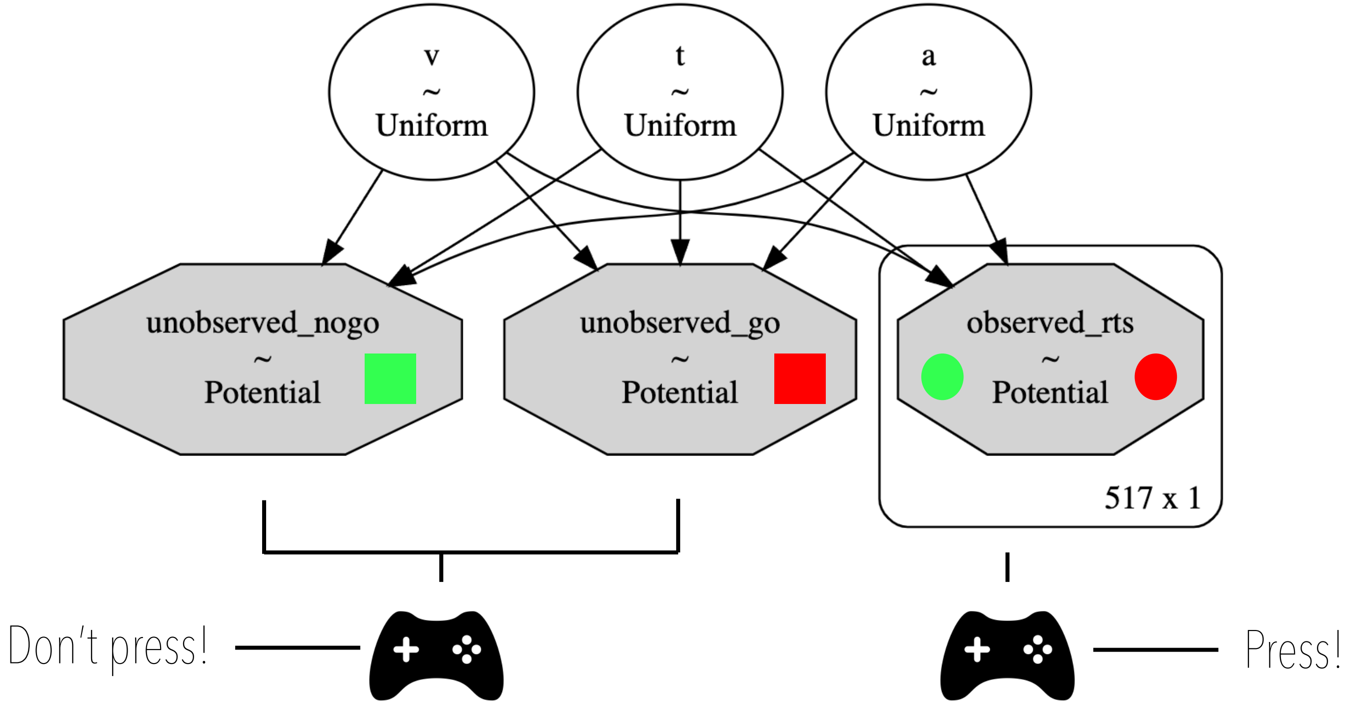




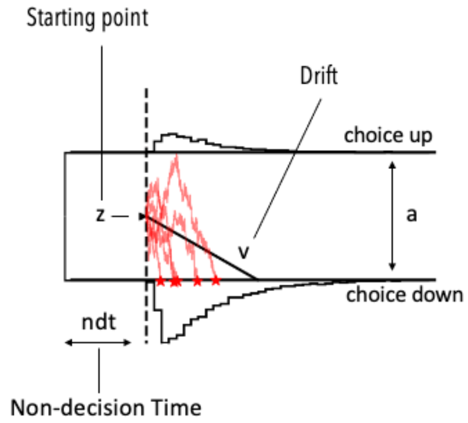
Let's look at all this through PyMC

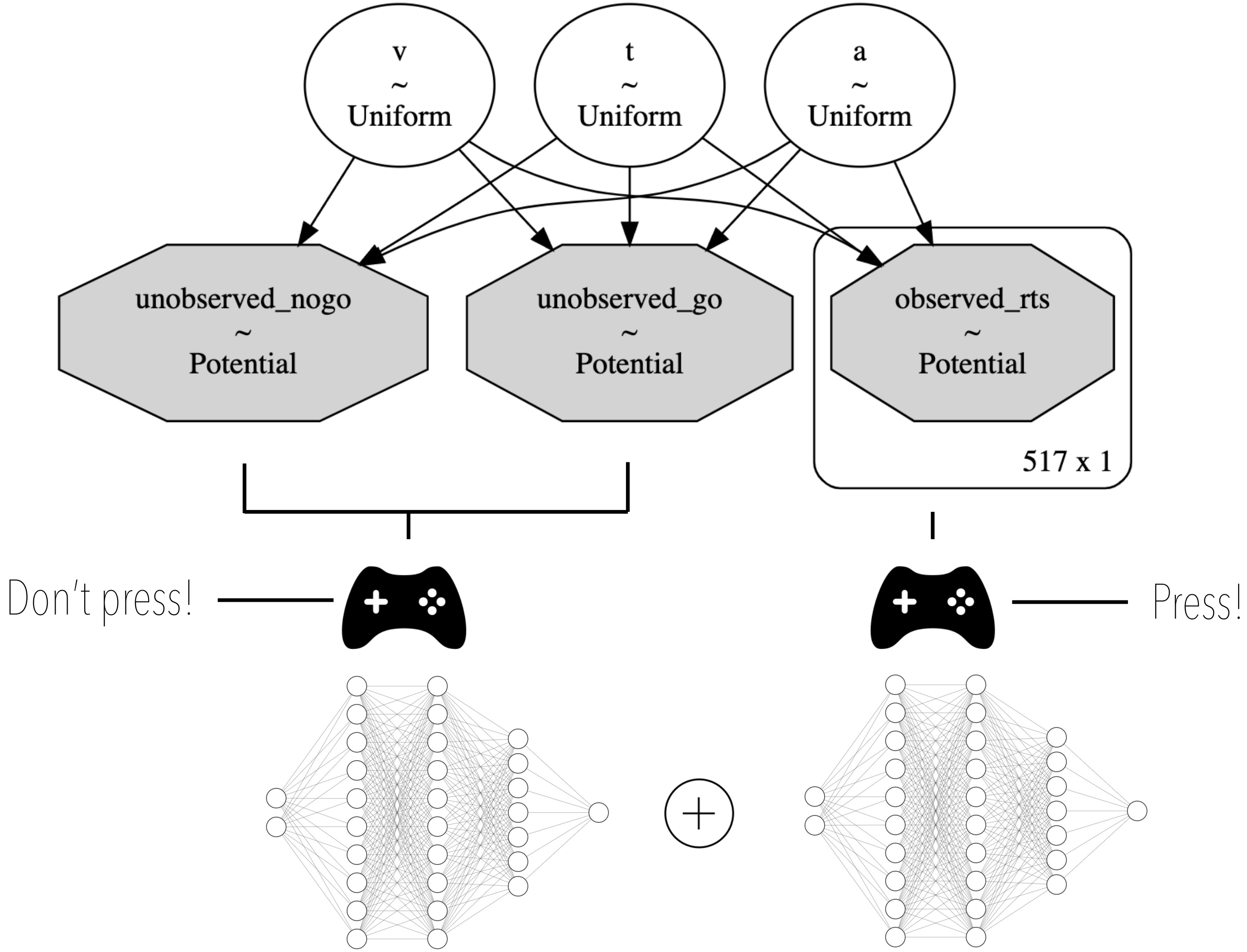




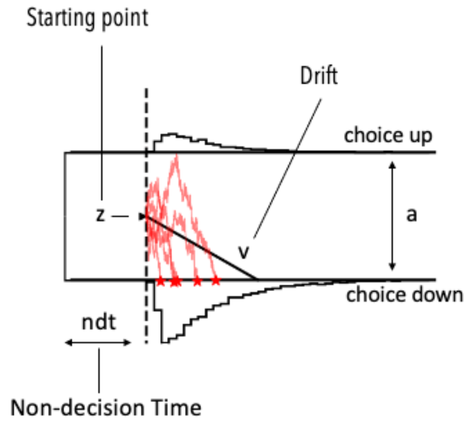


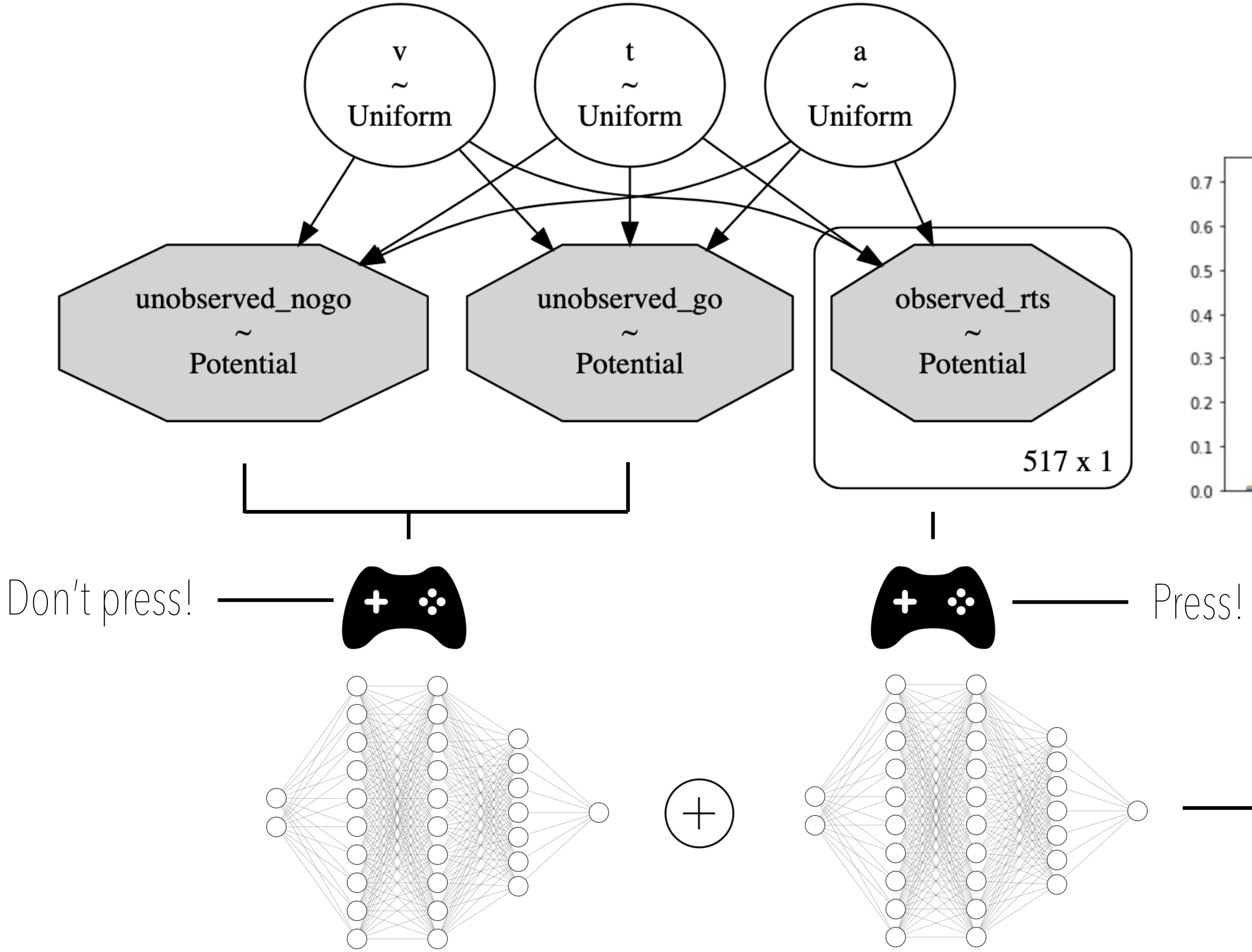
Let's look at all this through PyMC

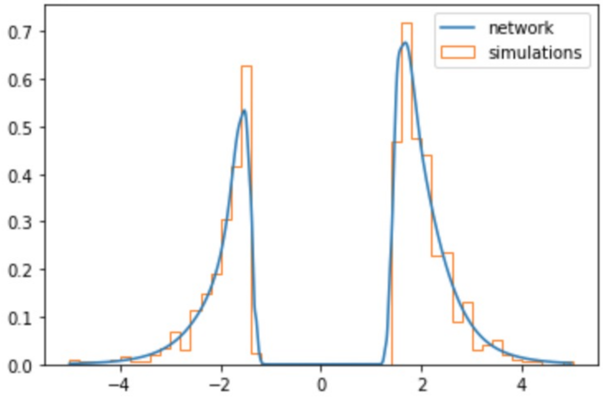




Let's look at all this through PyMC



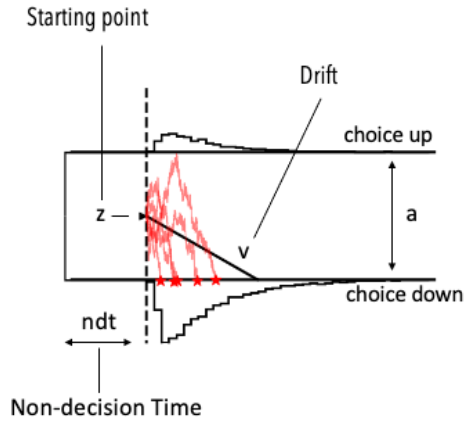


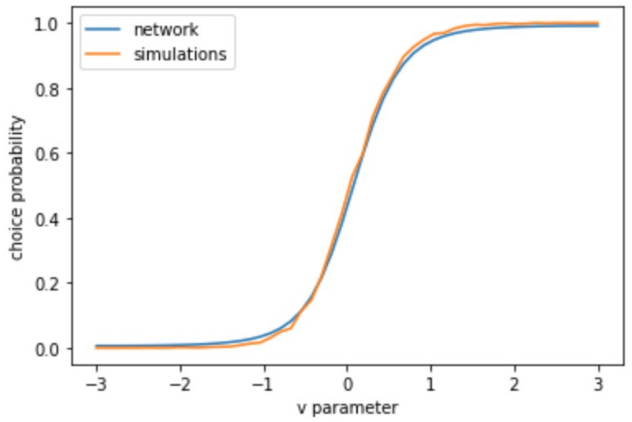


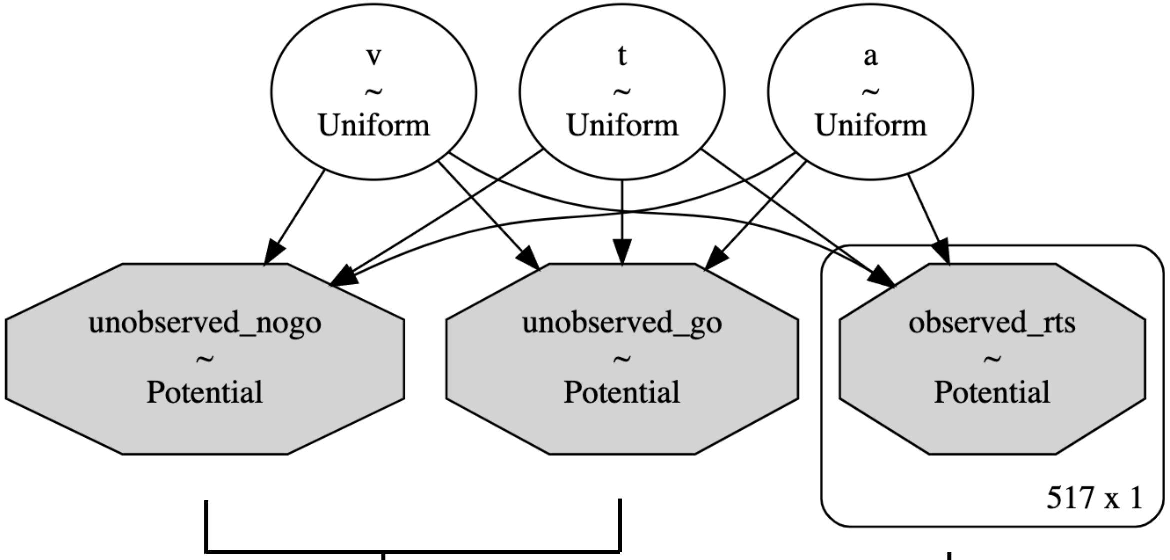
Network for choice reaction time pairs

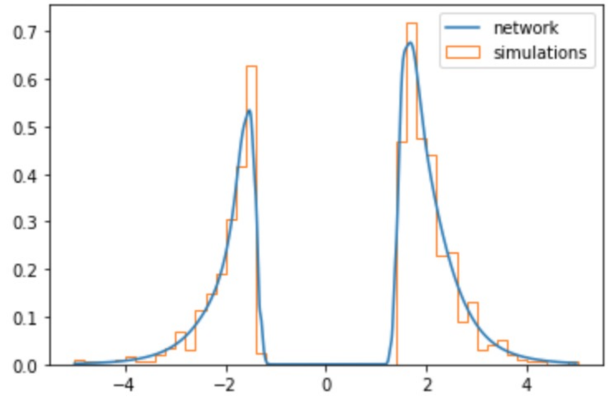
PyMC Labs

Let's look at all this through PyMC



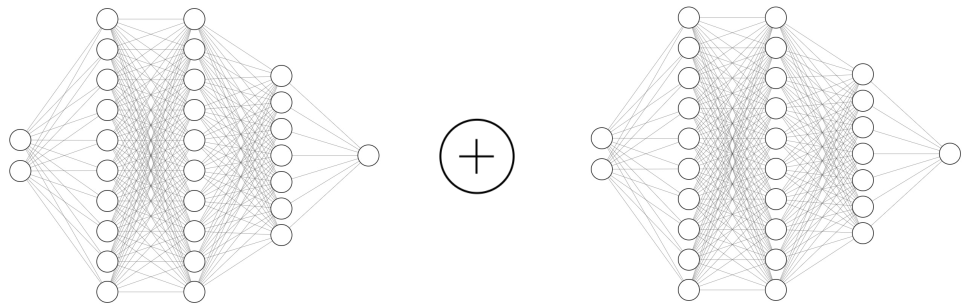






Don't press!

Press!



Network for choice probabilities (integral over choice-wise reaction time distributions)

Network for choice reaction time pairs

Code PyMC

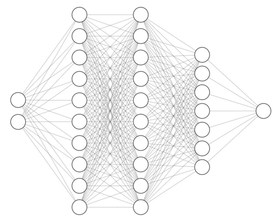
```python
with pm.Model() as m_ddm_gonogo:
    # Priors
    v = pm.Uniform("v", 0.000, 3.0)
    a = pm.Uniform("a", 0.3, 2.5)
    z = at.constant(0.5)
    t = pm.Uniform("t", 0.0, 2.0)

```

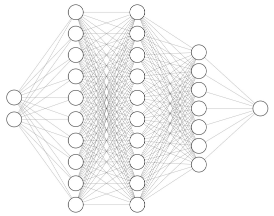
Specify priors as per usual

```python
    neg_choice_sum_go = at.constant(np.sum(obs_ddm_go['choices'] == -1))
    neg_choice_sum_nogo = at.constant(np.sum(obs_ddm_nogo['choices'] == -1))

    in_go = at.zeros((np.sum(obs_ddm_go["choices"] == 1), 6))
    in_nogo = at.zeros((np.sum(obs_ddm_nogo["choices"] == 1), 6))

    # subset to choice == 1
    # go trials
    in_go = at.set_subtensor(in_go[:, :-2], at.stack([v, a, z, t]))
    in_go = at.set_subtensor(in_go[:, -2], obs_ddm_go["rts"][obs_ddm_go["choices"] == 1])
    in_go = at.set_subtensor(in_go[:, -1], obs_ddm_go["choices"][obs_ddm_go["choices"] == 1])

    # nogo trials
    in_nogo = at.set_subtensor(in_nogo[:, :-2], at.stack([(-1) * v, a, z, t]))
    in_nogo = at.set_subtensor(in_nogo[:, -2], obs_ddm_nogo["rts"][obs_ddm_nogo["choices"] == 1])
    in_nogo = at.set_subtensor(in_nogo[:, -1], obs_ddm_nogo["choices"][obs_ddm_nogo["choices"] == 1])

    # combine go and nogo trials
    in_ = at.concatenate([in_go, in_nogo])
```

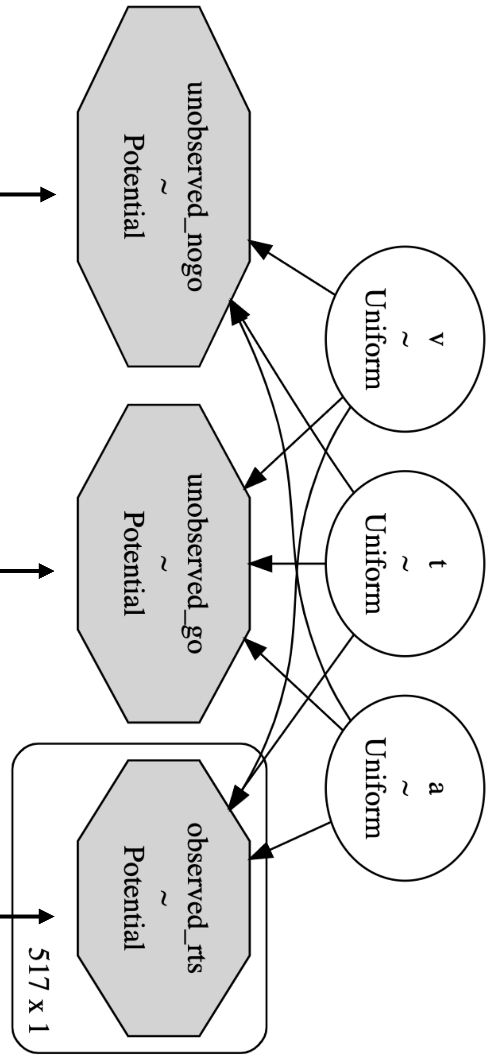
Some data prep…
Let's skip this detail

Code PyMC

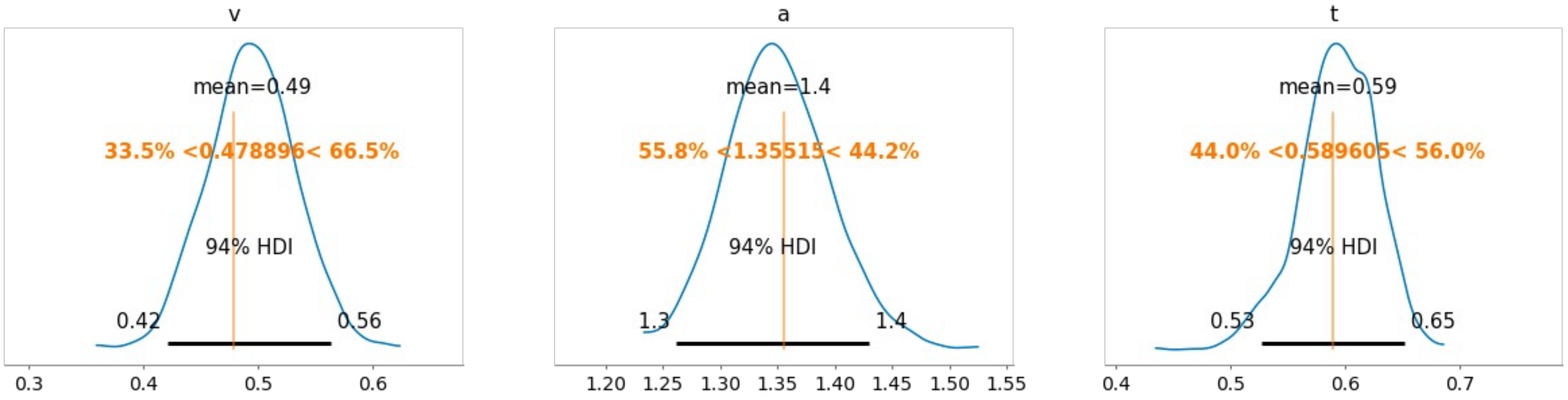




```python
# LAN
hid0 = at.tanh(at.dot(in_, weights[0]) + biases[0])
hid1 = at.tanh(at.dot(hid0, weights[1]) + biases[1])
hid2 = at.tanh(at.dot(hid1, weights[2]) + biases[2])
out = at.dot(hid2, weights[3]) + biases[3]
pm.Potential("observed_rts", out)

# CPN
in_cpn  = at.stack([at.stack([v, a, z, t]), at.stack([(-1) * v, a, z, t])])
hid0_cpn = at.tanh(at.dot(in_cpn, weights_cpn[0]) + biases_cpn[0])
hid1_cpn = at.tanh(at.dot(hid0_cpn, weights_cpn[1]) + biases_cpn[1])
hid2_cpn = at.tanh(at.dot(hid1_cpn, weights_cpn[2]) + biases_cpn[2])
preout_cpn = at.dot(hid2_cpn, weights_cpn[3]) + biases_cpn[3]
out_cpn = (-1) * at.log( 1 + at.exp(preout_cpn)) # this takes log(1-p) (lower bound)

# go trials
pm.Potential('unobserved_go', out_cpn[0, 0] * neg_choice_sum_go)

# nogo trials
pm.Potential('unobserved_nogo', out_cpn[1, 0] * neg_choice_sum_nogo)
```



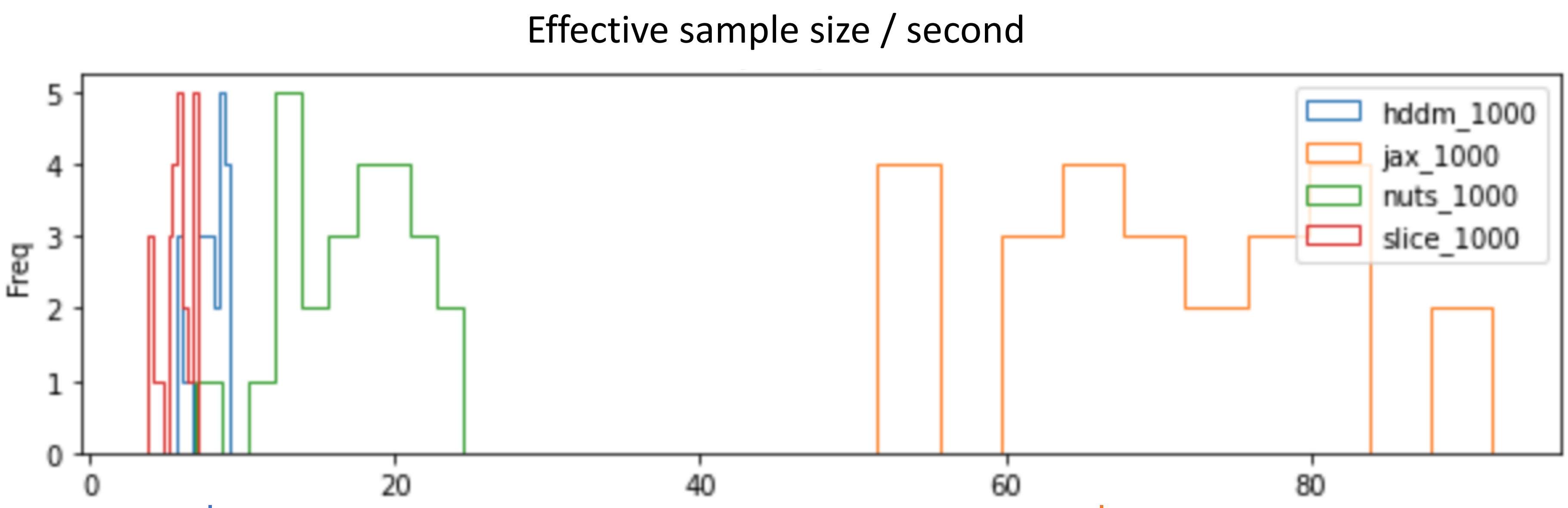
Proof of concept: (Parameter Recovery)



Just a <u>representative example</u> here

Works well on current set of test cases!

Proof of concept: (Speed)



Software stack of a previous project
Beating it with ~10x speed improvement!

Our approach with PyMC, through JAX

AKILI

PyMC Labs

https://www.pymc-labs.io/newsletter/