

# Neuromechanics of Human Motion

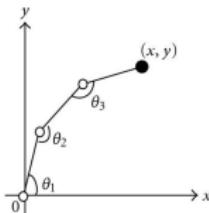
## Motor Adaptation & Learning

---

Joshua Cashaback, PhD

# Recap — Cost Functions

Redundancy: Degrees-of-Freedom (DOF) > Task Space Dimensions

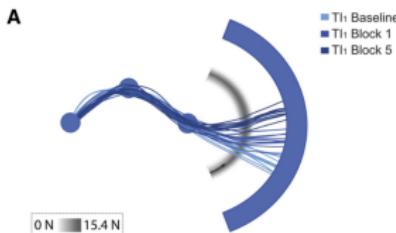


Joint Redundancy



Schematic of "full-blown" musculoskeletal model described in Kistemaker et al. (2010).

Muscle Redundancy



Work-Space Redundancy

# Recap — Cost Functions

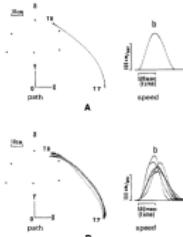
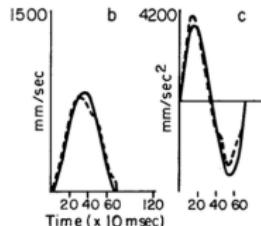
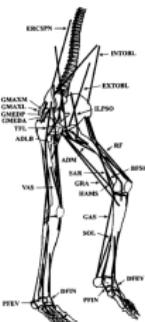
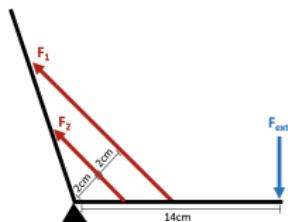
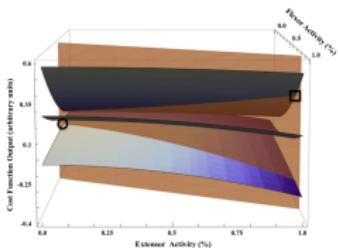


Fig. 4A and B. Large free movements between two targets (I-II = 70 cm). The movement starts at the start point (I), passes through the intermediate point (III) and ends at the end point (II) approximately in front of the body. A: Ideal trajectory predicted by the minimum torque-change principle. B: Observed hand trajectory for the seven subjects. a shows the paths and b shows the corresponding speed profiles.



# Recap — Cost Functions



Actions are the only way we can influence our world and determine whether we pass along genes

# Lecture Objectives — Learning

1. Common Experimental Tasks
2. Adaptation Model (Smith et al, 2006)
  - . Learning (and forgetting)
  - . Spontaneous recovery
  - . Savings
3. Generalization
4. Variability
5. Learning vs. Adaptation

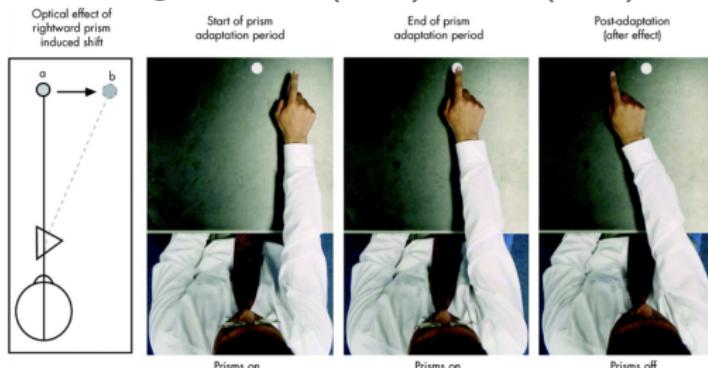
# Experimental Tasks

# Prism Adaptation

“Visuomotor Rotation”

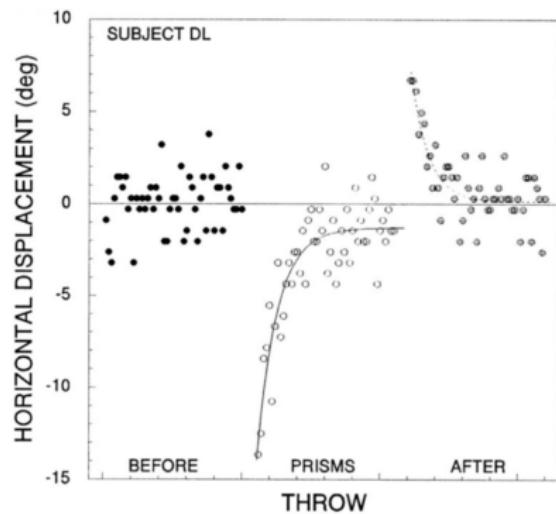


George Stratton (1896) Kohler (1951)



Rotate a cursor relative to movement (no vision of hand or arm)

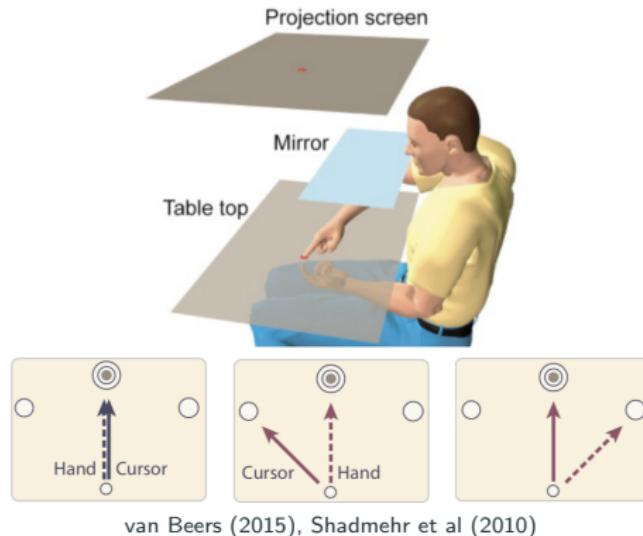
# Prism Adaptation



Martin et al. (2002) J Neurophysiol 88: 1685-1694

7

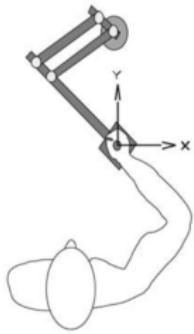
# Visuomotor Rotation



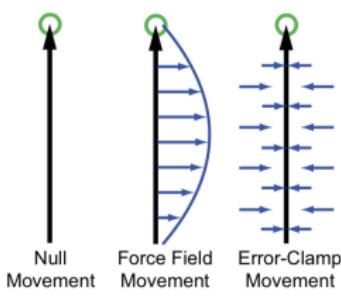
1. Virtual prism adaptation
2. Cursor rotates to create some error
3. Fast reaches or hand/arm blocked from vision

# Force-Field Adaptation

A Experimental Setup

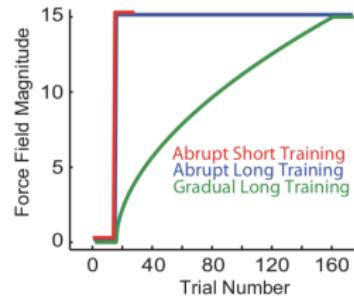


B Movement Types



$$\begin{bmatrix} f_x \\ f_y \end{bmatrix} = \begin{bmatrix} 0 & b \\ -b & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}, b = \pm 15 \text{ N/(m/s)}$$

C Training Schedule



Joiner (2013)

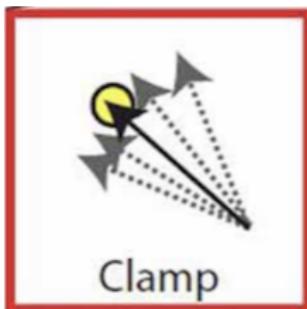
# Split-Belt Adaptation



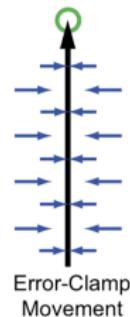
Belts move at independent speeds

$$SL_{asym} = SL_{fast} - SL_{slow}$$

# Error Clamps



visuomotor rotation clamp



force-field clamp

1. Error Clamps artificially impose zero error allowing us to probe the current state
2. Visuomotor clamp: cursor goes to target regardless of movement (record hand position)
3. Force field clamp: movement is straight (record force acting on wall and compare to optimal value)

# Error-Based Adaptation Models

# One State Model — Adaptation Rate

$$x(i+1) = x(i) + Be(i)$$

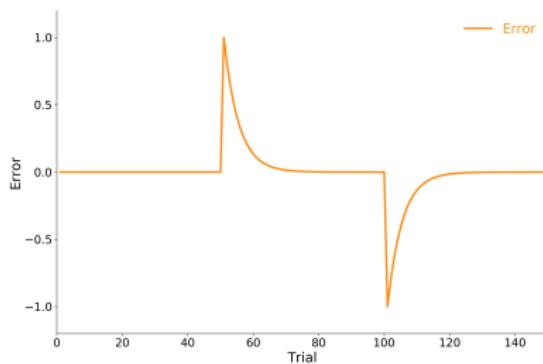
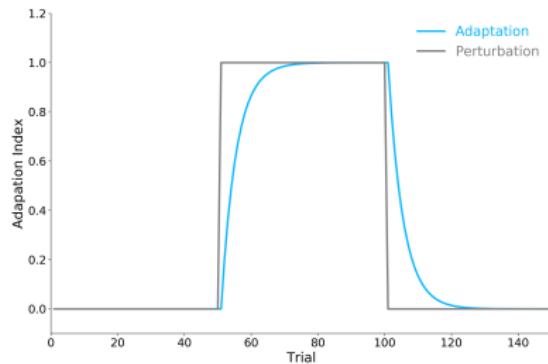
s.t.

$$e(i) = t(i) - x(i)$$

$$x(1) = 0$$

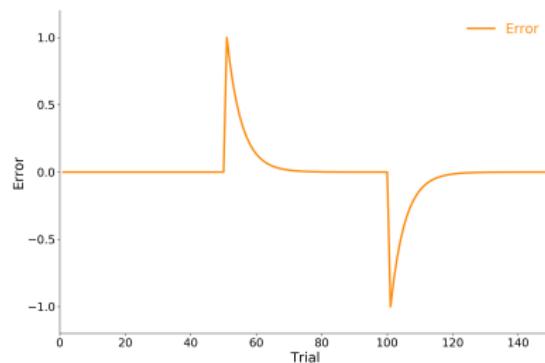
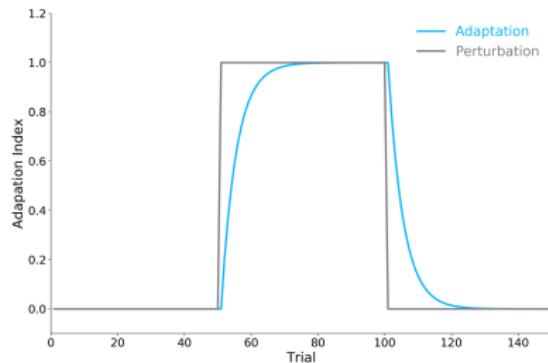
1.  $x(i)$ : movement direction or force in movement  $i$
2.  $e(i)$ : movement error or force error in movement  $i$
3.  $B$ : adaptation rate (proportion of error correction)
4.  $t(i)$ : target (e.g., target position or optimal force)

# One State Model — Adaptation Rate

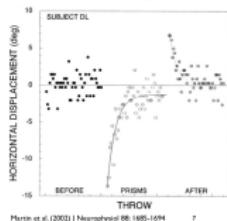


$$B = 0.2$$

# One State Model — Adaptation Rate



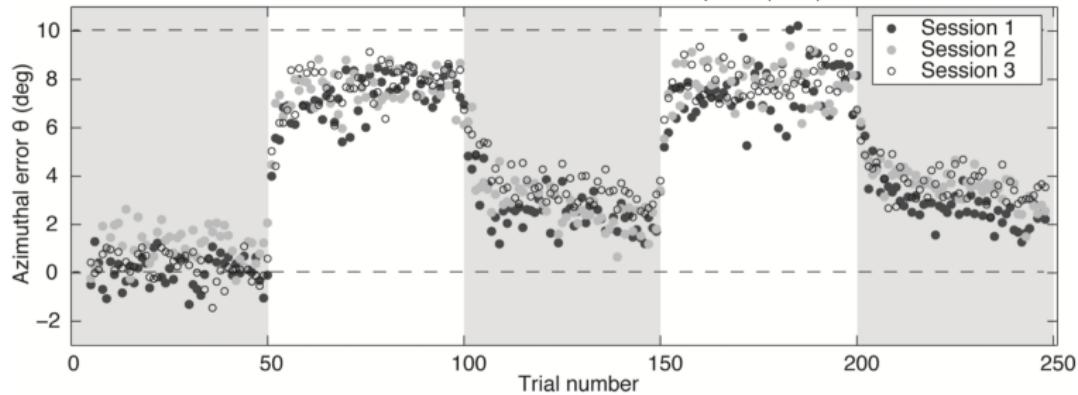
$$B = 0.2$$



Herrin et al. (2003) | *J Neurophysiol* 88:1685-1694

# Adaptation Not Complete

van der Kooij et al. (2015) PLoS ONE 10: e0117901



# One State Model — Retention Rate

$$x(i+1) = Ax(i) + Be(i)$$

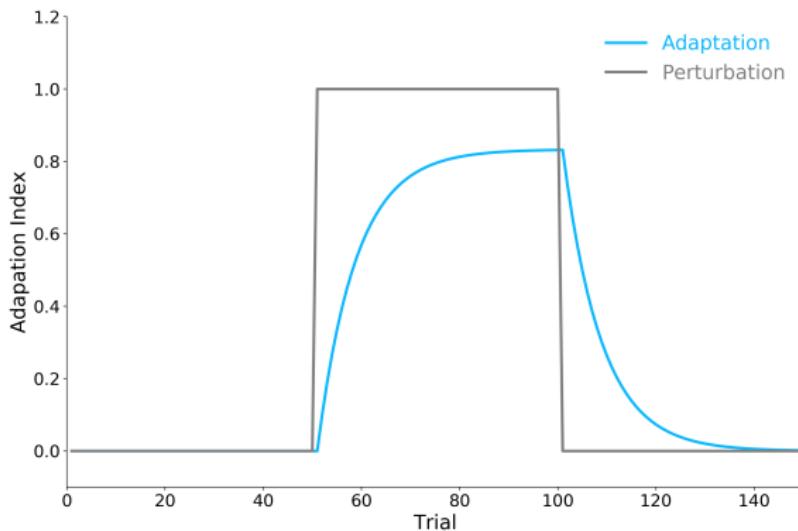
s.t.

$$e(i) = t(i) - x(i)$$

$$x(1) = 0$$

1.  $A$ : retention rate (or forgetting what you previously learning)

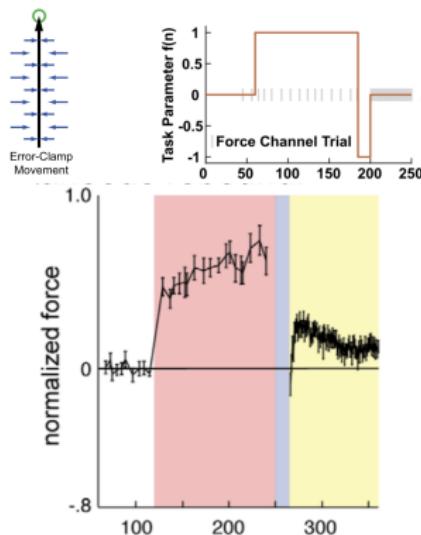
# One State Model — Retention Rate



$$A = 0.98, B = 0.1$$

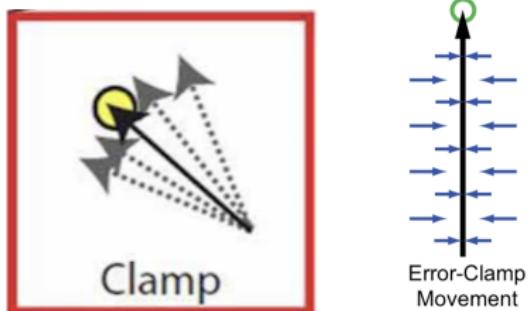
We can capture incomplete learning by including a retention rate

# Spontaneous Recovery



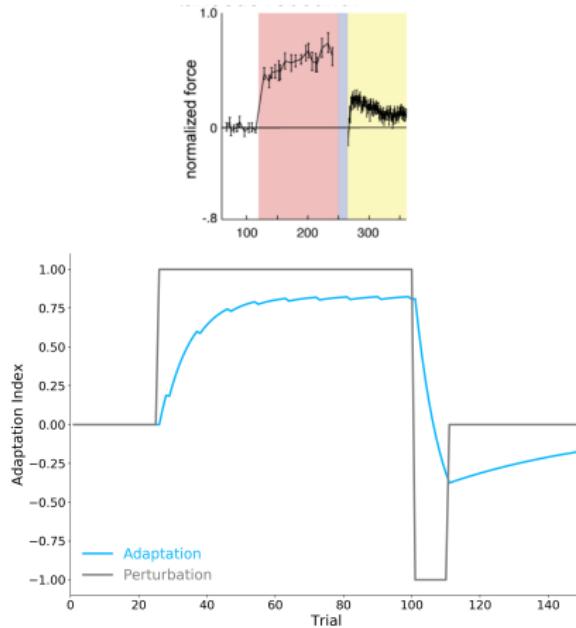
1. Clamp trials in a force-field task
2. Baseline (intermittent clamps), CW force-field (intermittent clamps), CCW force-field, clamps
3. Notice in the last phase the spontaneous recovery (rebound) to pre-CCW levels

# Modelling Clamp Trials



1. Clamp trials make the error to be zero.
2. The idea is to probe the state of the system when there is no error signal [ $t = x(i)$ ].
3.  $e(i) = t(i) - x(i) = 0$

# One State Model — Spontaneous Recovery



One state model cannot capture spontaneous recovery

# Two State Model

$$x_f(i+1) = A_f x_f(i) + B_f e(i)$$

$$x_s(i+1) = A_s x_s(i) + B_s e(i)$$

$$x_{net}(i) = x_f(i) + x_s(i)$$

s.t.

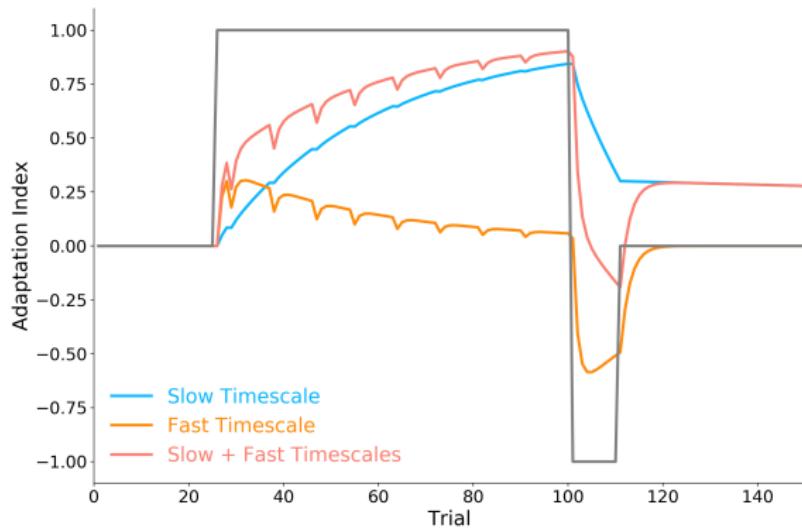
$$e(i) = t(i) - x_{net}(i)$$

$$A_s \geq A_f; B_f \geq B_s$$

$$x_f(1) = 0, x_s(1) = 0$$

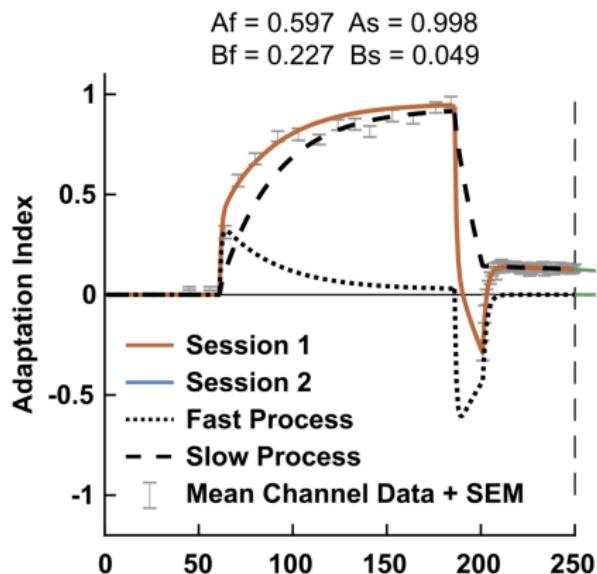
Smith et al (2006). PLoS biology, 4(6).

# Two State Model — Spontaneous Recovery



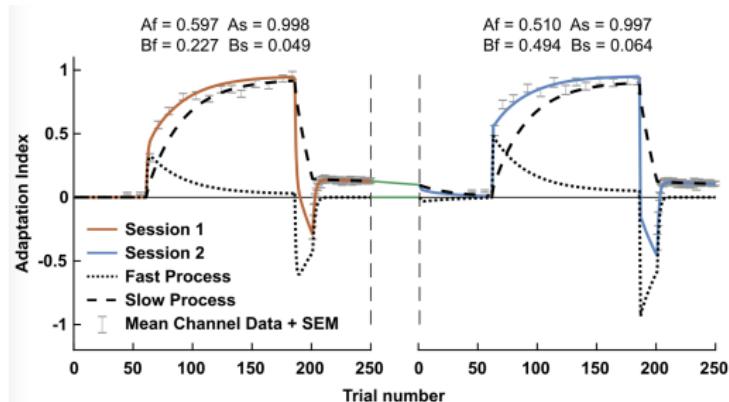
Two-state model can capture spontaneous recovery ('rebound')

# Two State Model — Spontaneous Recovery



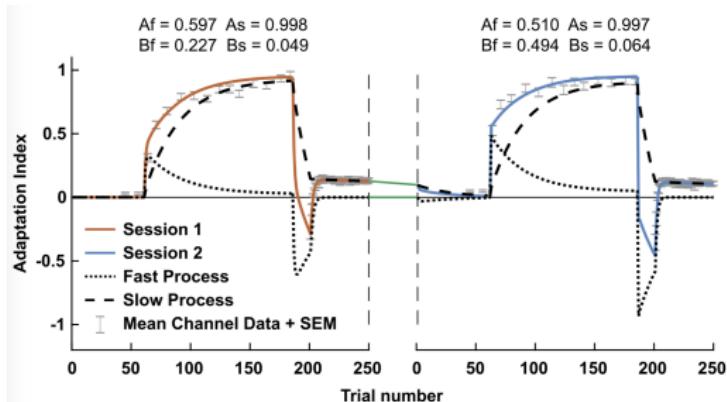
Two state model can capture spontaneous recovery ('rebound')

# Two State Model — Savings



1. Savings: significantly faster learning the second time you perform the same task
2. The two-state model cannot capture 'savings' when there is a sufficiently large number of washout trials.

# How to Capture Savings?

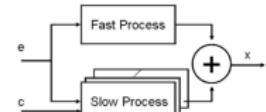


## Capturing Savings

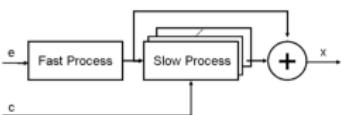
1. Multiple slow states
2. History of errors
3. Explicit (and Implicit) processes?

# Savings — Multiple Slow States

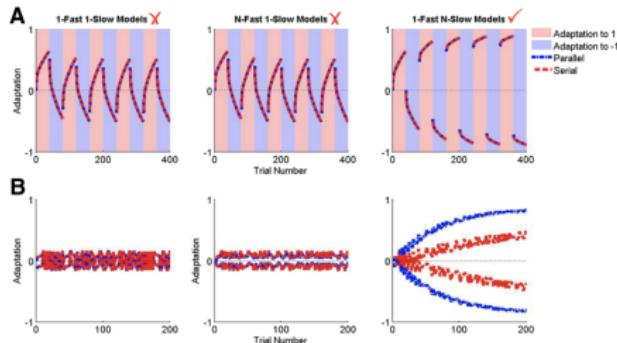
## A Parallel Architecture



## B Serial Architecture



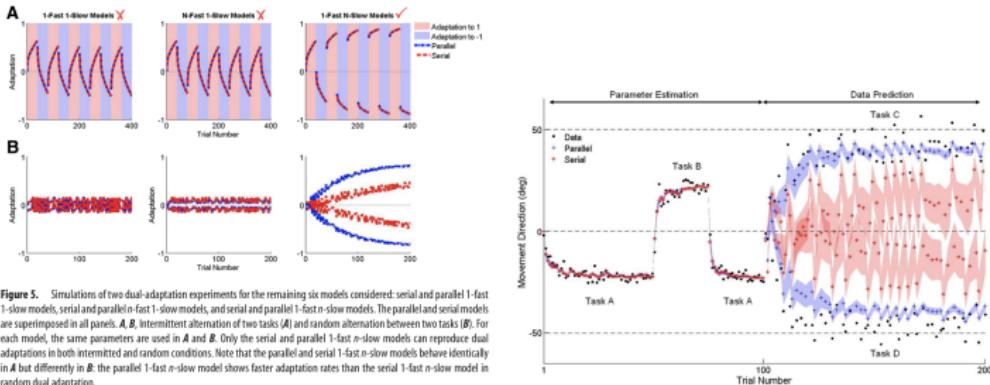
**Figure 2.** The two possible architectures of the 1-fast  $n$ -slow model: parallel (**A**) and serial (**B**).  $e$  is a motor error,  $c$  is a contextual cue, and  $x$  is a motor output. Multiple boxes in the slow process represent internal states switched by the contextual cue input.



**Figure 5.** Simulations of two dual-adaptation experiments for the remaining six models considered: serial and parallel 1-fast 1-slow models, serial and parallel  $n$ -fast 1-slow models, and serial and parallel 1-fast  $n$ -slow models. The parallel and serial models are superimposed in all panels. **A**, **B**, Intermittent alternation of two tasks (**A**) and random alternation between two tasks (**B**). For each model, the same parameters are used in **A** and **B**. Only the serial and parallel 1-fast  $n$ -slow models can reproduce dual adaptations in both intermittent and random conditions. Note that the parallel and serial 1-fast  $n$ -slow models behave identically in **A** but differently in **B**: the parallel 1-fast  $n$ -slow model shows faster adaptation rates than the serial 1-fast  $n$ -slow model in random dual adaptation.

Lee & Schweighofer (2009) JNeurosci, 29(33), 10396-10404.

# Savings — Multiple Slow States



**Figure 5.** Simulations of two dual-adaptation experiments for the remaining six models considered: serial and parallel 1-fast 1-slow models, serial and parallel n-fast 1-slow models, and serial and random alternation between two tasks (**A**). Only the serial and parallel 1-fast n-slow models can reproduce dual adaptations in both intermittent and random conditions. Note that the parallel and serial 1-fast n-slow models behave identically in **A** but differently in **B**: the parallel 1-fast n-slow model shows faster adaptation rates than the serial 1-fast n-slow model in random model adaptation.

1. Data - black dots; parallel model = blue; serial model = red
2. Data significantly closer to parallel model

# Savings — History of Errors

Typical learning model

$$x(i+1) = Ax(i) + Be(i)$$

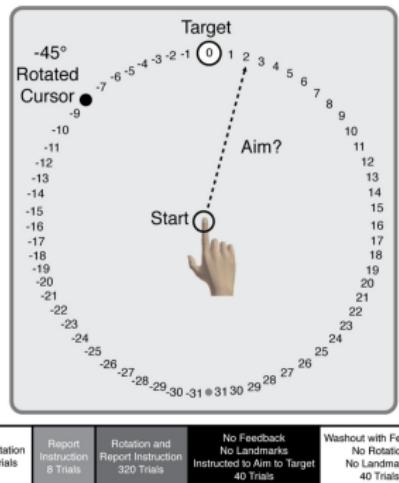
Learning rate sensitive to error size

$$x(i+1) = Ax(i) + B(e(i))e(i)$$

1. Error sensitivity decreases in a random environment
2. The nervous system stores a history of error
3. Will remember large and infrequent errors (cache): can explain savings

Herzfeld et al (2014). Science, 345(6202), 1349-1353.

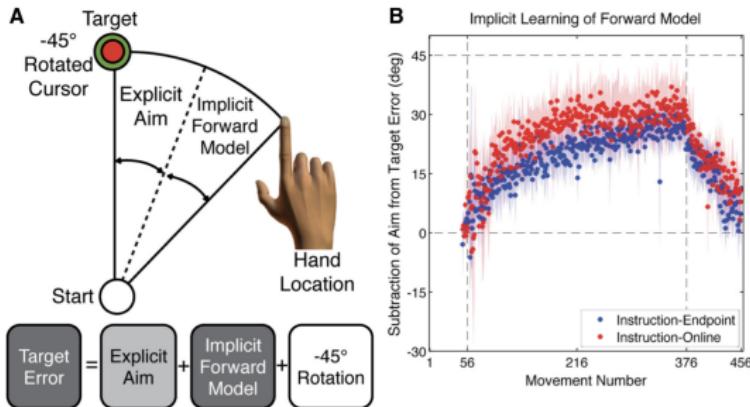
# Implicit vs. Explicit



Taylor et al (2014) JNeurosci, 34(8), 3023-3032.

1. Participants reported their aim
2. Reported aim = explicit (cognitive) strategy

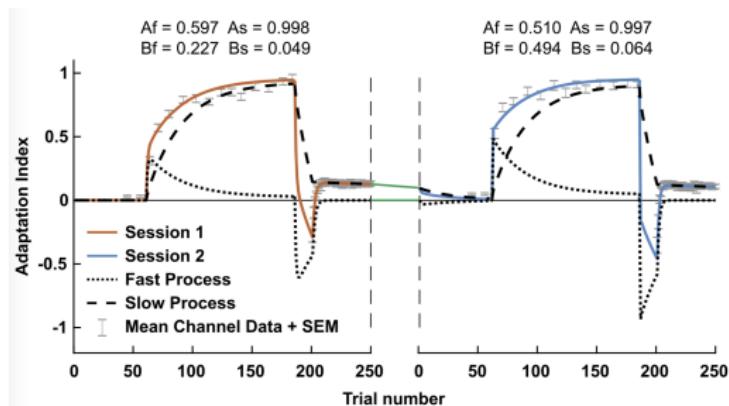
# Implicit vs. Explicit



Taylor et al (2014) JNeurosci, 34(8), 3023-3032.

1. Implicit = Target error - explicit - (-45°)
2. Implicit maps onto slow process
3. Explicit strategy maps onto fast process

# Explicit (and Implicit) Contributions to Savings

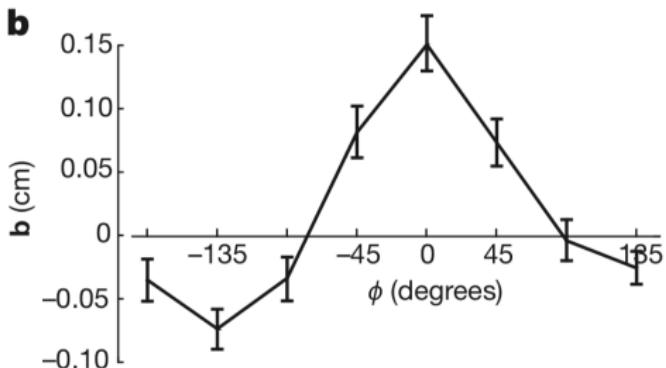
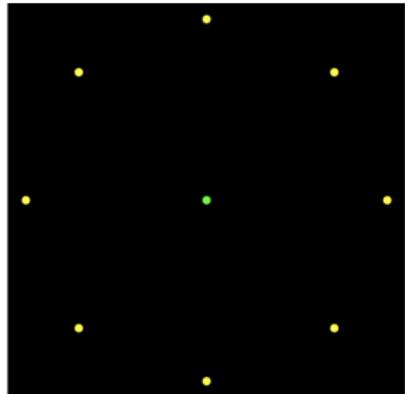


1. Well accepted that explicit processes involved with savings
2. Mixed results on whether implicit processes also involved

# Other Considerations of Learning

1. Generalization
2. Variability
3. Learning vs. Adaptation

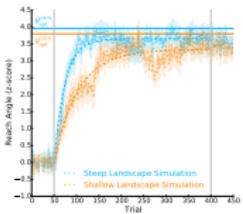
# Generalization



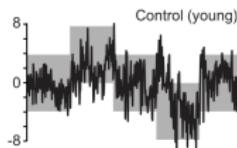
Thoroughman & Shadmehr (2000) Nature 407: 742-747

1. How does force-field learning when reaching to some target transfer when reaching to the current target?
2. Two state model version (Tanaka 2012, Neural Comput 24: 939-966)

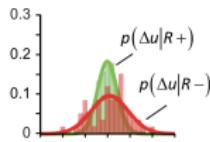
# Variability



Cashaback et al. (2019)



Pekny et al. (2015)

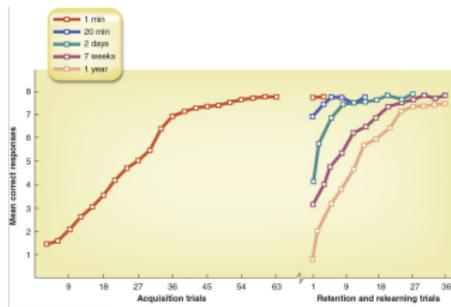
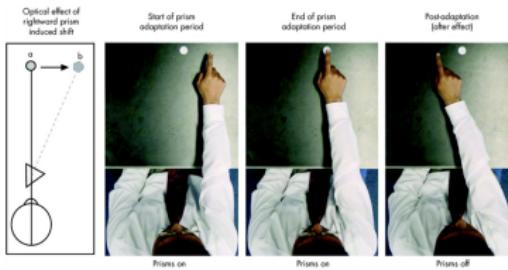


1. Variability may or may not benefit error-based learning.
  - . He et al. (2016). Wu et al (2014).
2. Seems necessary to explore in reinforcement-based learning.
  - . Wu et al (2014).

# Learning vs. Adaptation

## Adaptation

A set of processes associated with modulating the sensorimotor system for relatively temporary gains in task performance



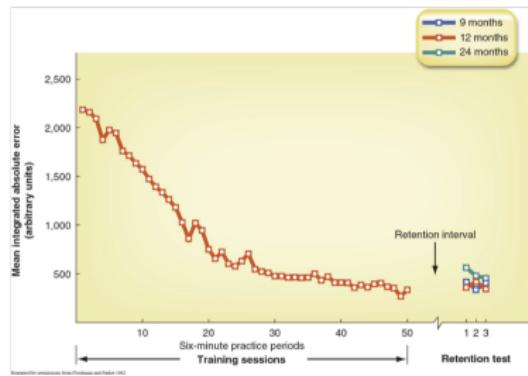
Often associated with discrete, lab-based tasks.

Acquired quickly and more fleeting

# Learning vs. Adaptation

## Learning

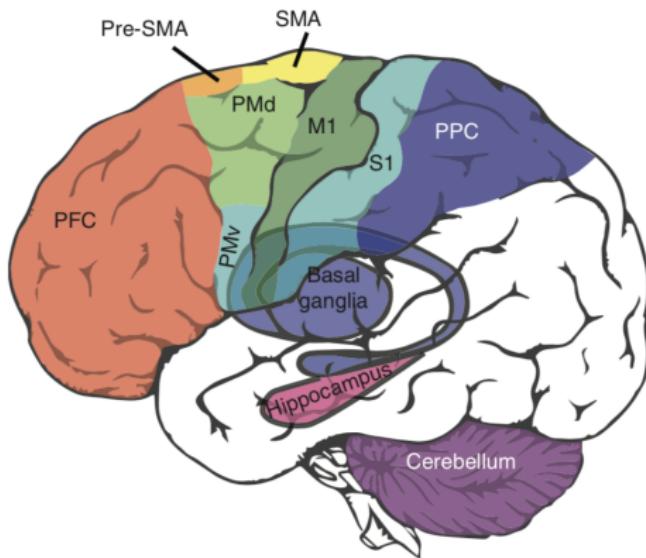
A set of processes associated with practice or experience causing relatively permanent gains to produce skilled performance.



Often associated with continuous, skill-based, real-world tasks.

Can take a very long time to acquire

# Neural Basis of Learning



Krakauer et al (2019) Comprehensive Physiology 9 (2), 613-663

# Two State Fitting — Sample Python Code

```
from pylab import *
import numpy as np
from scipy.optimize import minimize
# p = perturbation vector, eclamp = clamp trial vector
# data = fake data | Xnet(i) from forward model with known parameters
def find_coeff(IC, data, p, eclamp):
    As, Bs, Af, Bf = IC[0], IC[1], IC[2], IC[3]
    # two_state() below is the two state model function
    E, XS, XF, X = two_state(As, Bs, Af, Bf, p, eclamp)
    if Bf <= Bs or Af >= As or Bf <= 0.0 or Bf >= 1.0 or Bs <= 0.0 or Bs >= 1.0 or A
        mindiff = 1000.0
    else:
        mindiff = np.sum(abs(data-X) ** 1.0) * 1.0
    print(mindiff, As, Bs, Af, Bf)
    axis([0,150,-0.5,1])
    plt.ion()
    plot(data, linewidth = 5.0, color = 'blue')
    plot(X, linewidth = 2.0, color = 'orange')
    show()
    plt.pause(0.0001)
    plt.clf()
    return mindiff
As0, Bs0, Af0, Bf0 = 0.994, 0.025, 0.521, 0.268 # initial guess
IC = [As0, Bs0, Af0, Bf0]
results = minimize(find_coeff, IC, args = (data, p, eclamp), method = 'nelder-mead')
As1, Bs1, Af1, Bf1 = results.x
Neuromechanics - BMEG 467/667
```

# Summary

1. Understand multiple states of adaptation
2. Two state model can capture many features of learning
3. Many factors to consider (e.g., generalization, variability, etc.)
4. Model Fitting

Questions???

# Assignment

see handout

# Next Class

Presentations!

# Acknowledgements

Robert van Beers (CoSMo lecture)

Michael Carter