Creating ACT-R Models by Training

Andrea Stocco
University of Washington

stocco@uw.edu
@TheRealDrDre2

Confession of a recovered connectionist

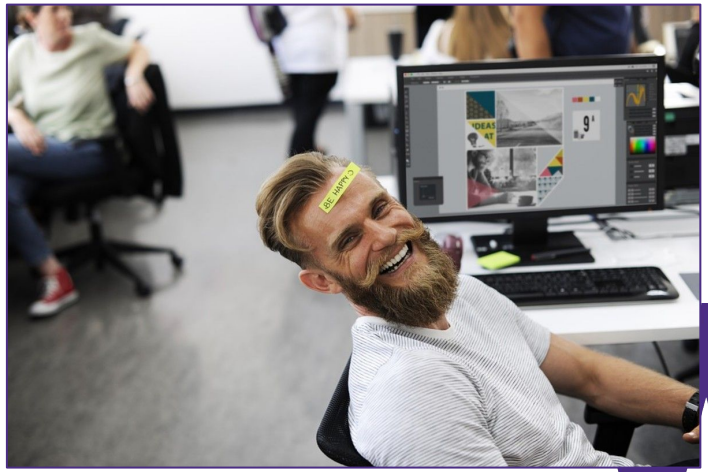
Confession of a recovered connectionist

ACT-R: You must engineer all the productions carefully --- or else!

Confession of a recovered connectionist

ACT-R: You must engineer all the productions carefully --- or else!

Neural nets: You just set the desired response, then **train** the system feedback



Why would you want to train ACT-R?

- Less time spent **building** the model
  - At least, less **user** time
- More time spent **analyzing** the model
  - Explore a greater space of possible ways to do a task
- (Maybe) More realistic neuroimaging data?

What would it look like?

Ideal workflow

- Define starting chunks (visual info, basic facts) and correct responses (“press left index” for “circle”)
- Run the task many times
  - Until model behaves accurately
- The model is adjusted based on response feedback
  - Learning
- Examine what the model has learned

How do we train ACT-R?

- Networks mostly trained with **gradient descent**
- Easy to implement in networks, not so in ACT-R…
  - Structure changes (new chunks and productions)
  - Dynamics are important
  - Tried to do some math, failed
- Solution tried here: **Reinforcement learning**
  - Learn the optimal sequence of productions for a task
- Which productions do we start with?

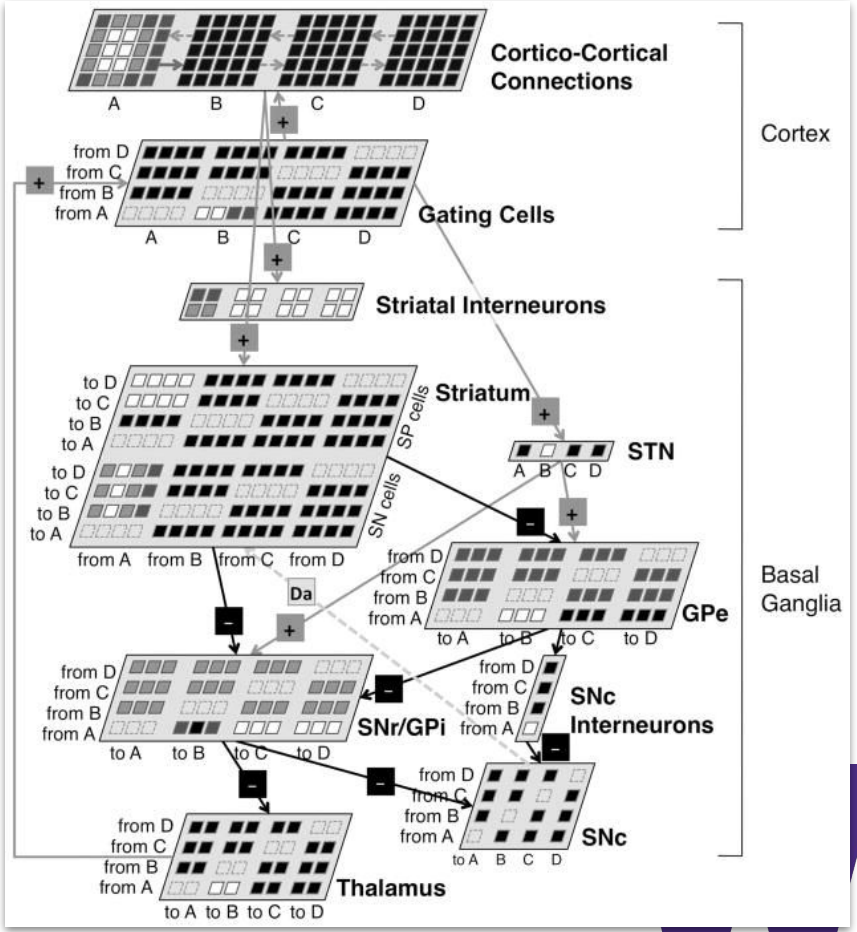
The PRIMs Solution

- Taatgen (2013) proposed that ACT-R is born with a single, large set of small productions, called **PRIMs** (for **PRIM**itive elements)
- Each PRIM basically just transfers one slot value from one buffer to another.
- PRIMs = basic **gating operations** in BG

The Basal Ganglia

Stocco et al (2010):

- BG just **gate information** from one cortical region to another
- BG has funnel architecture with **massive bottleneck**



Learning with PRIMs

1. Create a model with PRIMs and basic knowledge
2. Let the model run randomly until a correct response is made
3. Reward correct responses (“Reward is enough”)
4. Repeat

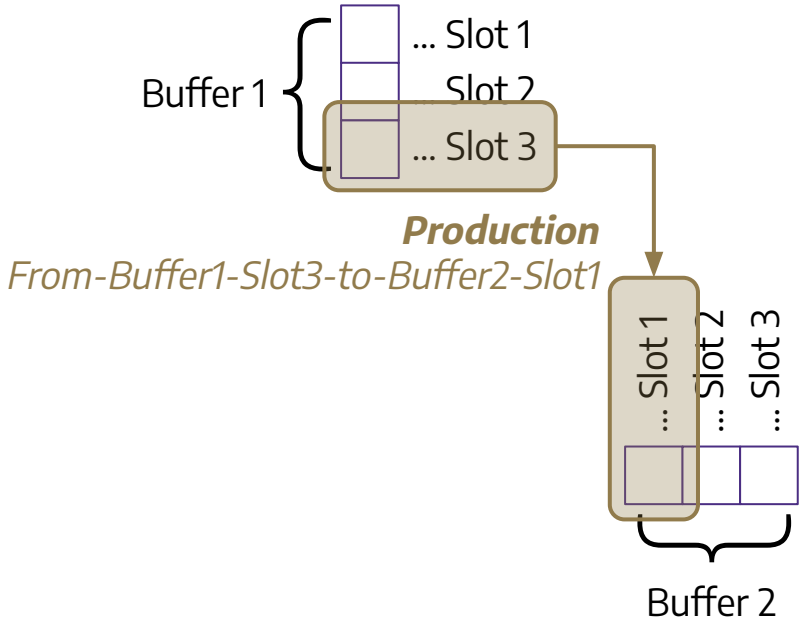
… And see if something happens

Setting chunks up for PRIMs

- Productions access values through specific **slot names**
- Creates problems of **complexity**
  - Different chunks/models = different slot names
- Forced a **unified format**: all chunks have *N* slots, all slots have meaningless names (*slot1*, *slot2*, *slot3*…)

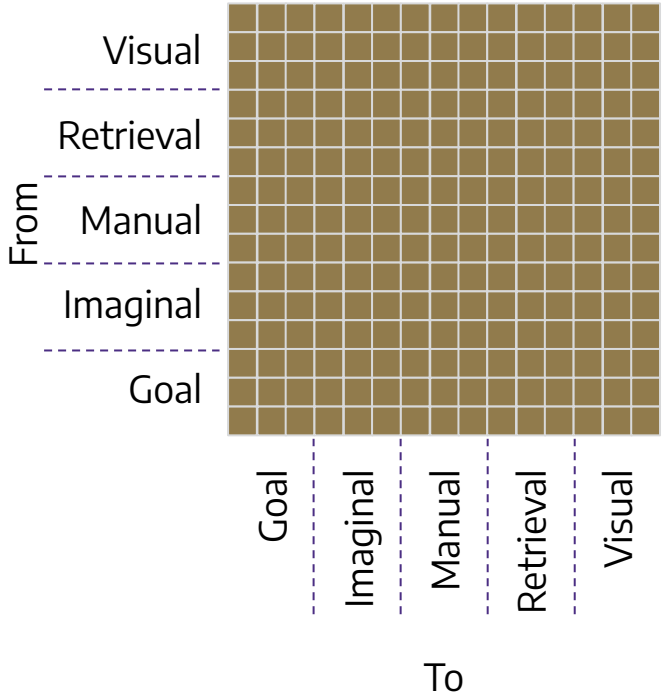
Productions

- Productions just move one slot value from one buffer to another slot value in a different buffer



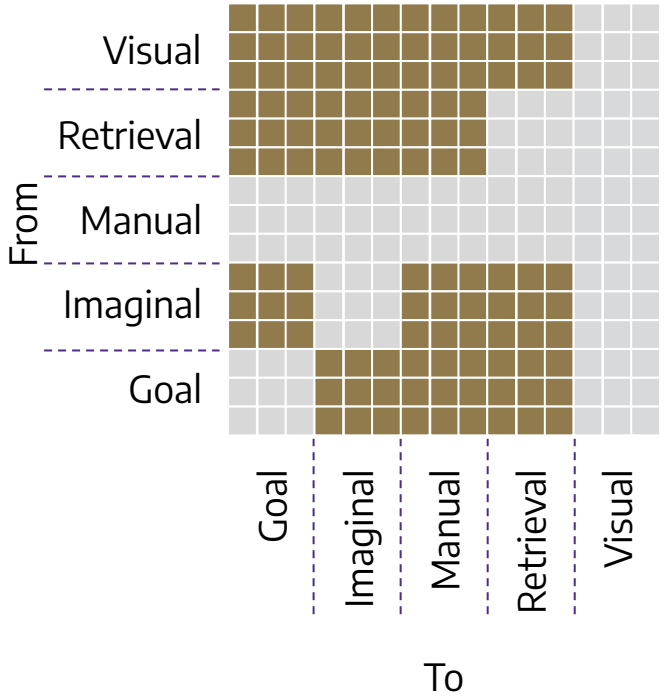
Productions

- Productions just move one slot value from one buffer to another slot value in a different buffer

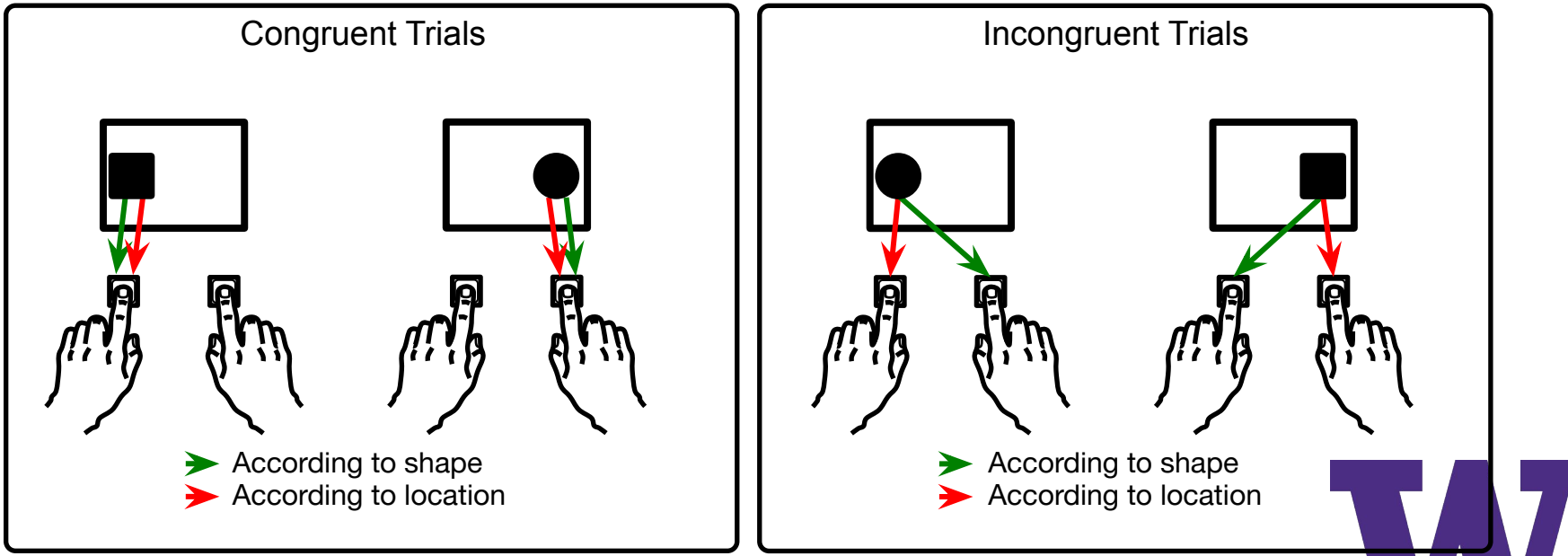


Productions

- Productions just move one slot value from one buffer to another slot value in a different buffer
- Not all productions are meaningful
  - Copying a value onto itself
  - Copying to Visual
  - Copying from Manual



Example: Simon task



Why the Simon task?

- Basic lab task
- Similar to other common interference tasks
  - Stroop, Flanker, Picture/Word
- Many ACT-R models have been written

Example: Simon task

N = 3 slots; Rules in DM, Stimuli are placed in Visual

Rules

Slot1	Slot2	Slot3
Circle	Left	Rule
Square	Right	Rule

Stimuli

Slot1	Slot2	Slot3
Circle	Left	Black
Circle	Right	Black
Square	Left	Black
Square	Right	Black

The “model”

```
(define-model simon-train
  (sgp :er t
       :esc t
       :ul t
       :egs 0.5
       :alpha 0.05
       )

  (chunk-type memory slot1 slot2 slot3)

  (add-dm (rule1 isa memory slot1 rule
            slot2 circle slot3 left)
          (rule2 isa memory
            slot2 rule slot2 square slot3 right)
          (congruent1 isa memory slot1 circle
             slot2 left slot3 black)
          (congruent2 isa memory slot1 square
            slot2 right slot3 black)
          (incongruent1 isa memory slot1 circle
            slot2 right slot3 black)
          (incongruent2 isa memory slot1 square
             slot2 left slot3 black)
)

(create-productions-from-template)
```

The “model”

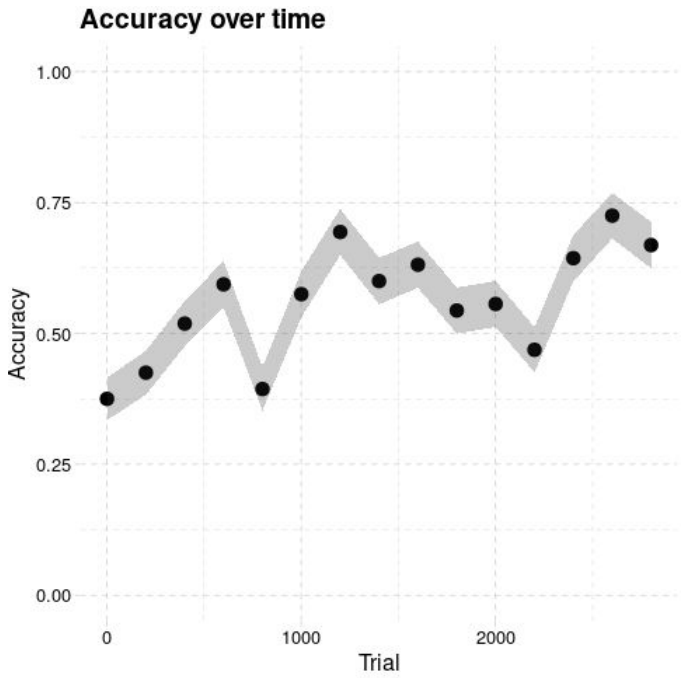
```
(define-model simon-train
  (sgp :er t
       :esc t
       :ul t
       :egs 0.5
       :alpha 0.05
       )

  (chunk-type memory slot1 slot2 slot3)

  (add-dm (rule1 isa memory slot1 rule
            slot2 circle slot3 left)
          (rule2 isa memory
            slot2 rule slot2 square slot3 right)
          (congruent1 isa memory slot1 circle
            slot2 left slot3 black)
          (congruent2 isa memory slot1 square
            slot2 right slot3 black)
          (incongruent1 isa memory slot1 circle
            slot2 right slot3 black)
          (incongruent2 isa memory slot1 square
            slot2 left slot3 black)
)

(create-productions-from-template)
```

...

...

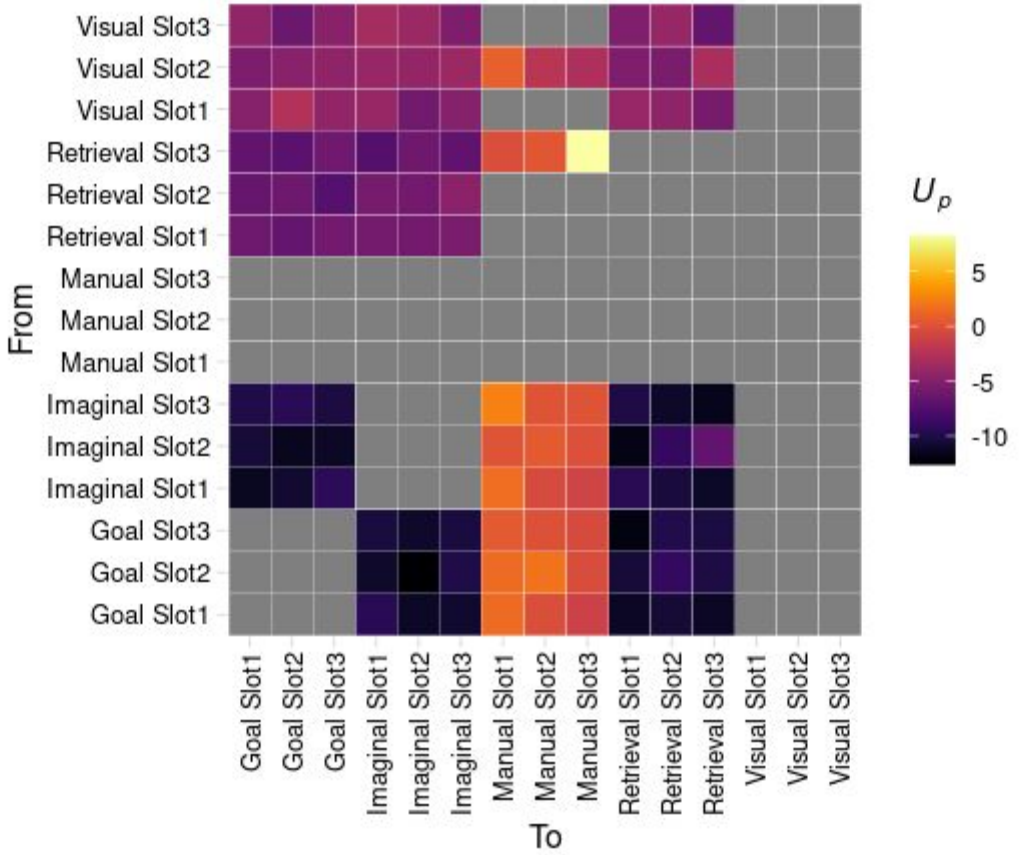
And then... We let it run

Does it work?



The model (slowly) learns!

Does it work?



Inspecting the model

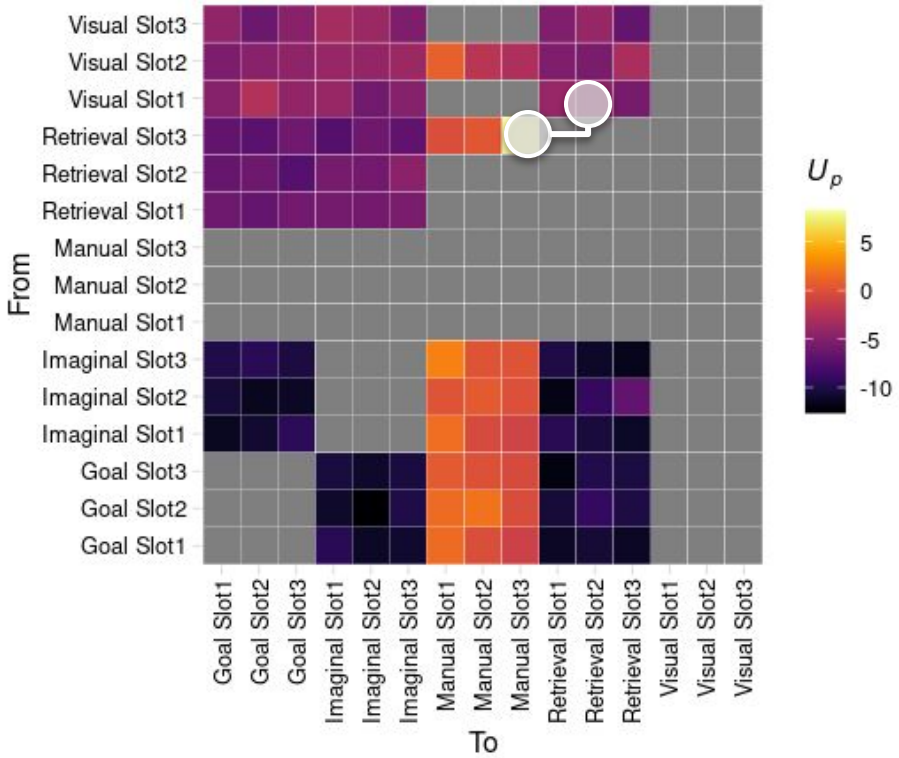
Interesting results 1: Where does interference come from?

Two modeling families

- Declarative interference
  - (Altmann 2001, van Maanen 2009)
- Procedural interference
  - (Lovett 2005, Stocco 2017)

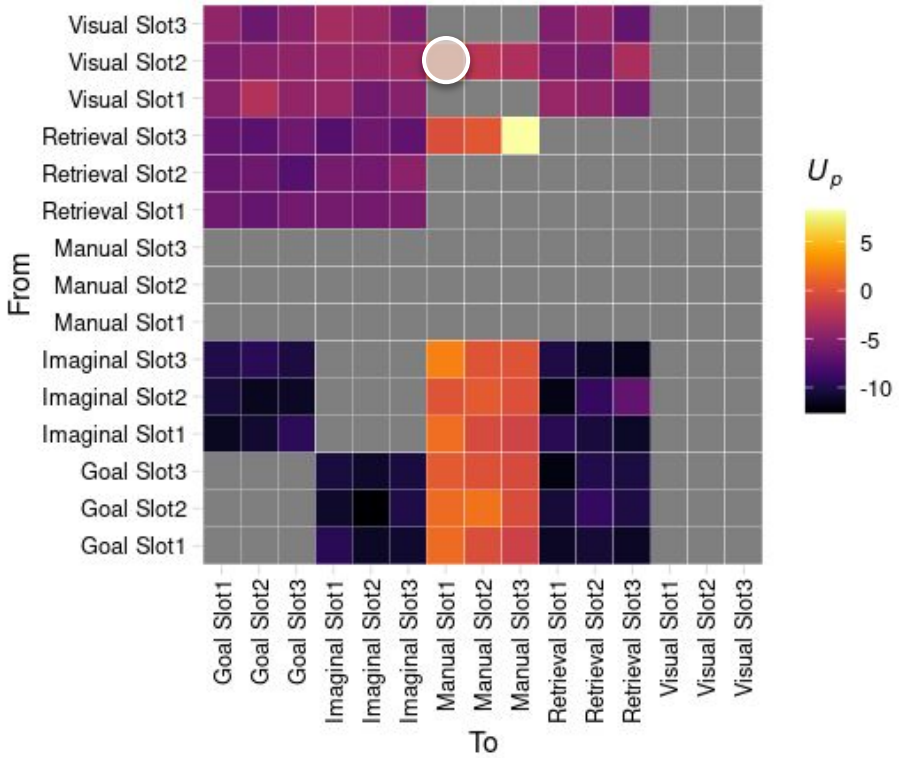
Interesting results 1: Where does interference come from?

- Retrieve response associated with the shape
- Use the retrieved response in the manual module



Interesting results 1: Where does interference come from?

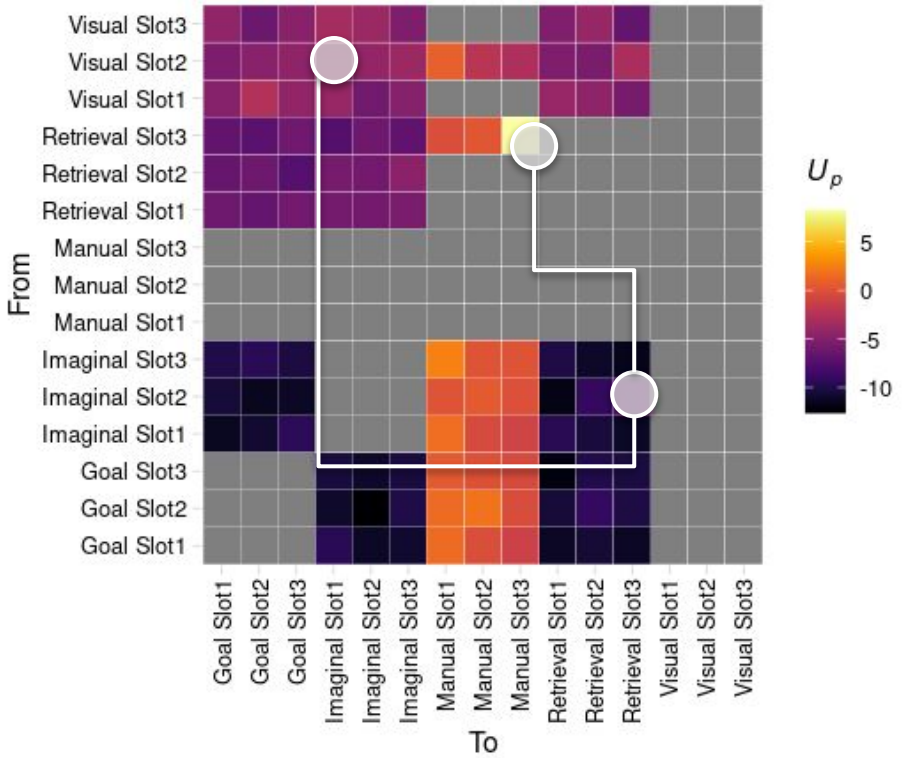
- Retrieve response associated with the shape
- Use the retrieved response in the manual module
- Use the location of the stimulus as response



Interesting results 1: Where does interference come from?

- Retrieve response associated with the shape
- Use the retrieved response in the manual module
- Use the location of the stimulus as response

- Put shape in WM
- Use WM to retrieve response
- Use the retrieved response in the manual module



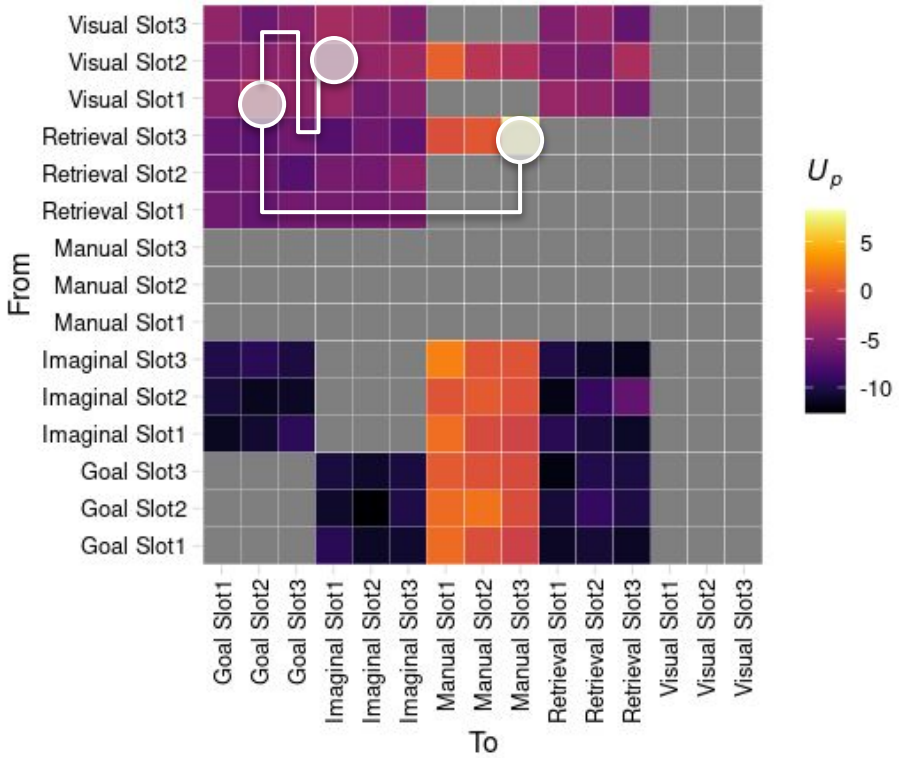
Interesting results 1: Where does interference come from?

- Retrieve response associated with the shape
- Use the retrieved response in the manual module

- Use the location of the stimulus as response

- Put shape in WM
- Use WM to retrieve response
- Use the retrieved response in the manual module

- Put shape in WM
- Then retrieve associated response
- Then put response in manual

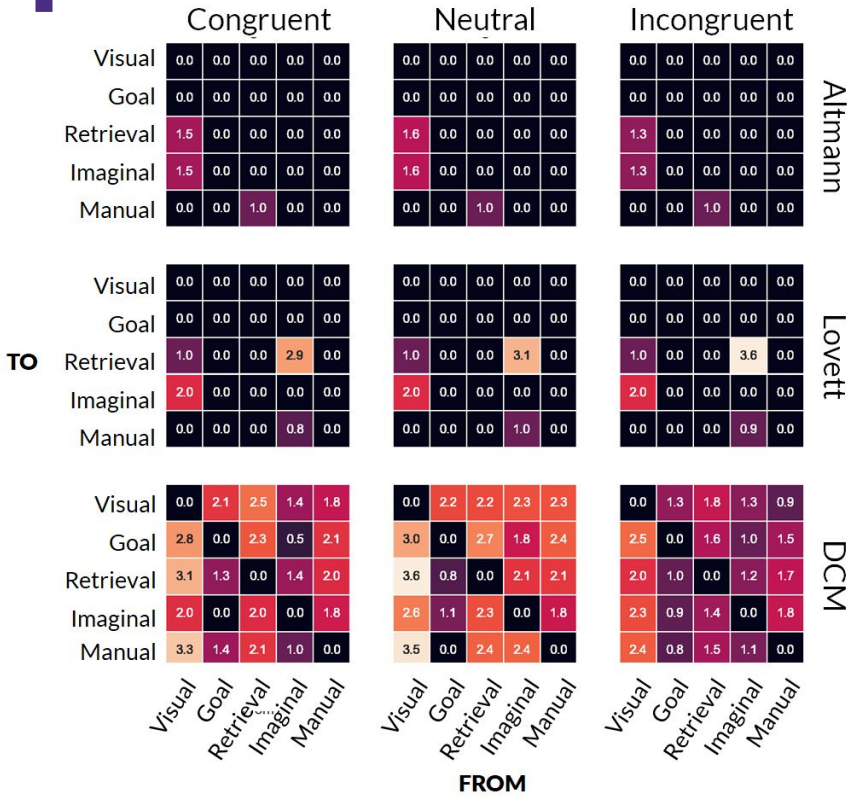


Interesting results 2: Functional connectivity predictions

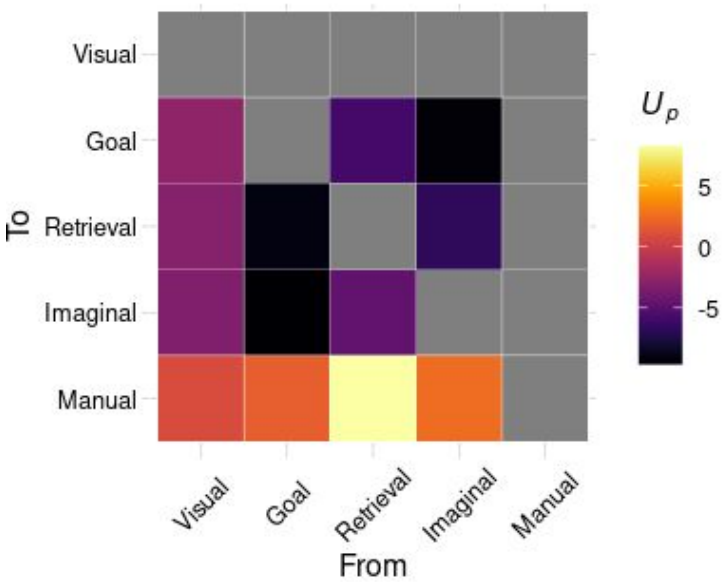
- Ketola et al. 2020 proposed using **functional connectivity** between ACT-R regions to evaluate models
- Used Altmann 2001 and Lovett 2005 Stroop models

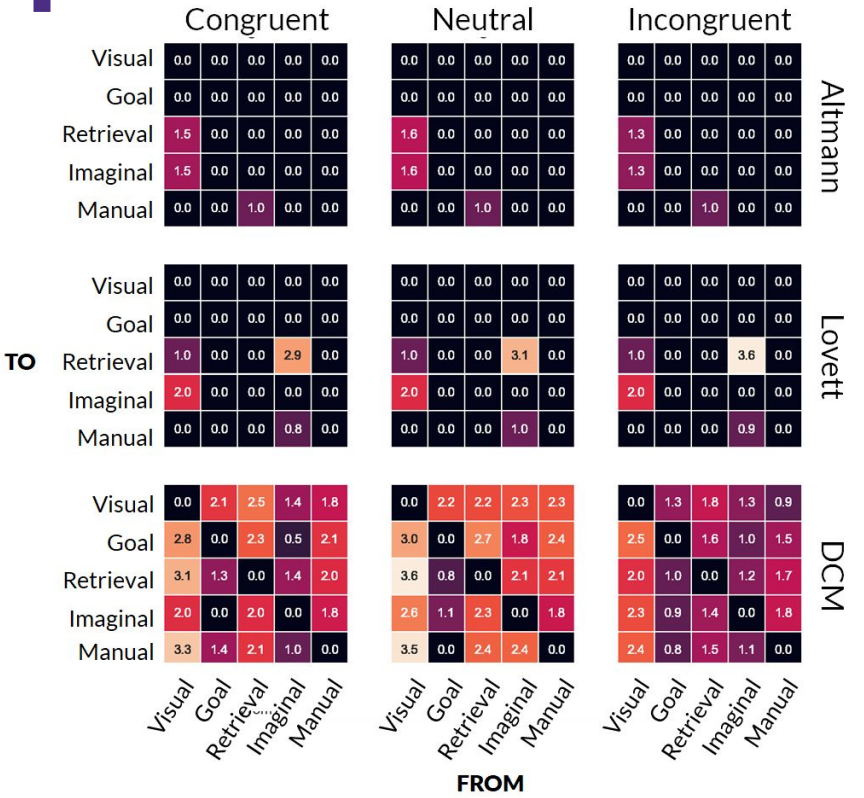
Interesting results 2: Functional connectivity predictions

- Ketola et al. 2020 proposed using **functional connectivity** between ACT-R regions to evaluate models
- Used Altmann 2001 and Lovett 2005 Stroop models
- **Very sparse compared to real connectivity**



Interesting results 2: Functional connectivity predictions



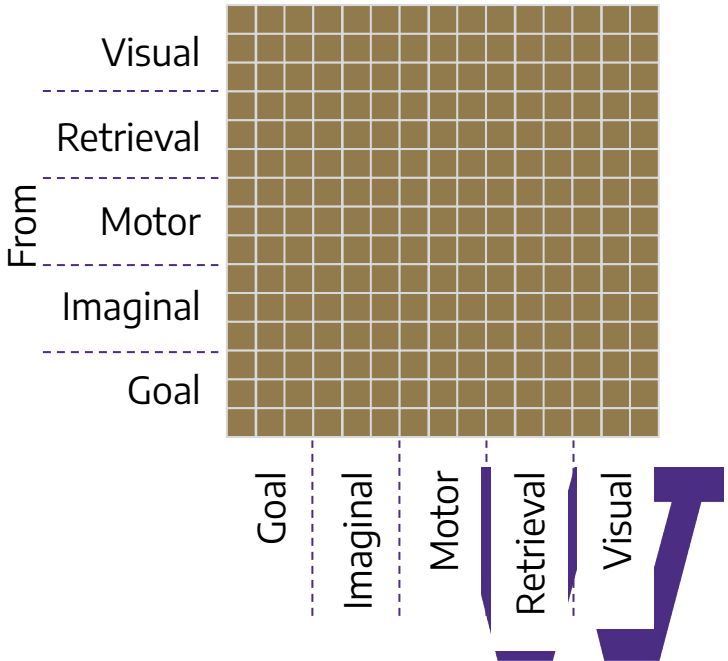


Why would you want to train ACT-R?

- Less time spent **building** the model ✅
  - At least, less **user** time
- More time spent **analyzing** the model ✅
  - Explore a greater space of possible ways to do a task
- (Maybe) More realistic neuroimaging data? ✅

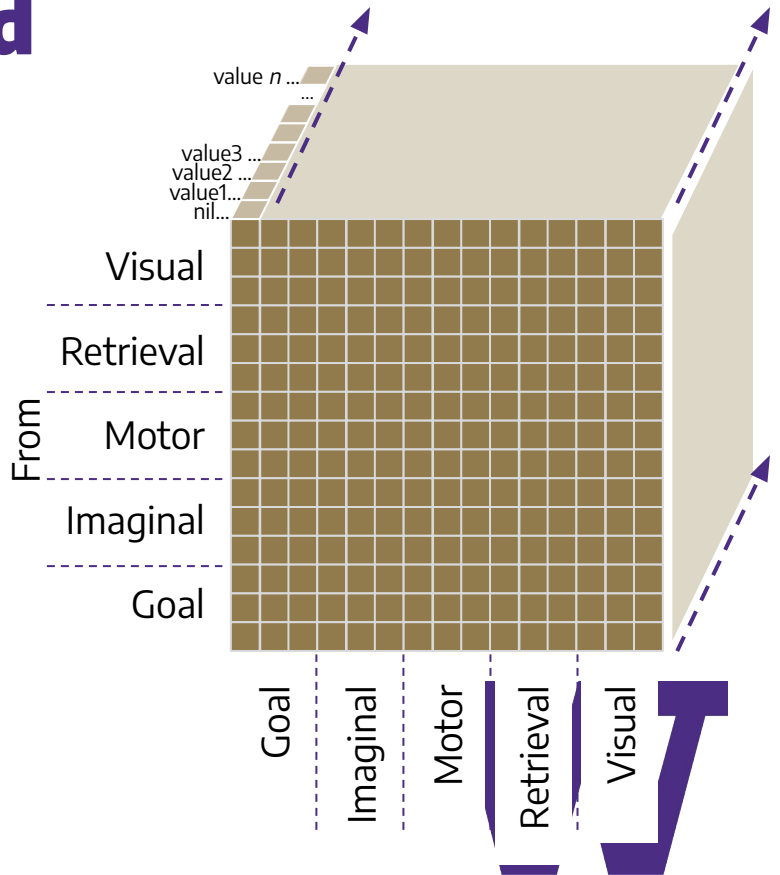
Caveat #1: Realistic tasks

- The Simon task is basic
- To deal with realistic tasks, we need **more complex** productions:
  - Test multiple conditions
  - More retrieval cues
- Possible solution: Expand productions to include **values**



Caveat #1: The road ahead

- The Simon task is basic
- To deal with realistic tasks, we need **more complex** productions:
  - Test multiple conditions
  - More retrieval cues
- Possible solution: Expand productions to include **values**



Caveat #2: Handling memory

- So far, I have avoided the biggest simplification
- The Simon example **avoids** creating new memories
  - All buffers are initialized with empty chunks
  - No strict harvesting on imaginal and goal
- Why?
  - During learning, the model produces a lot of **garbage memories**

Garbage memories

- Many unsuccessful trials create **useless** knowledge

```
(GOAL-CHUNK0-4
   SLOT1  CIRCLE
   SLOT2  RULE
   SLOT3  CIRCLE)
```

- Garbage that is accidentally retrieved is more likely to be retrieved again, even when $R(t)$ is negative!

Final Thoughts

- Not sure how much it will scale up
  - but so far, fingers crossed!
- Problems with memory are tantalizing
  - You want to forget a lot early on… Reminds me of debate about childhood amnesia
  - Debate on whether memory is affected by reward
- Need to dance around the procedural model a lot
  - Might want to try out some redesigns
  - We might not need “productions”

Thanks!

Special thanks to the UW crew:

- Holly Hake
- Teddy Haile
- Jim Treyens
- Cher Yang