Cerebellum and planning depth in gameplay

Research Assistant: Tianyi Xia

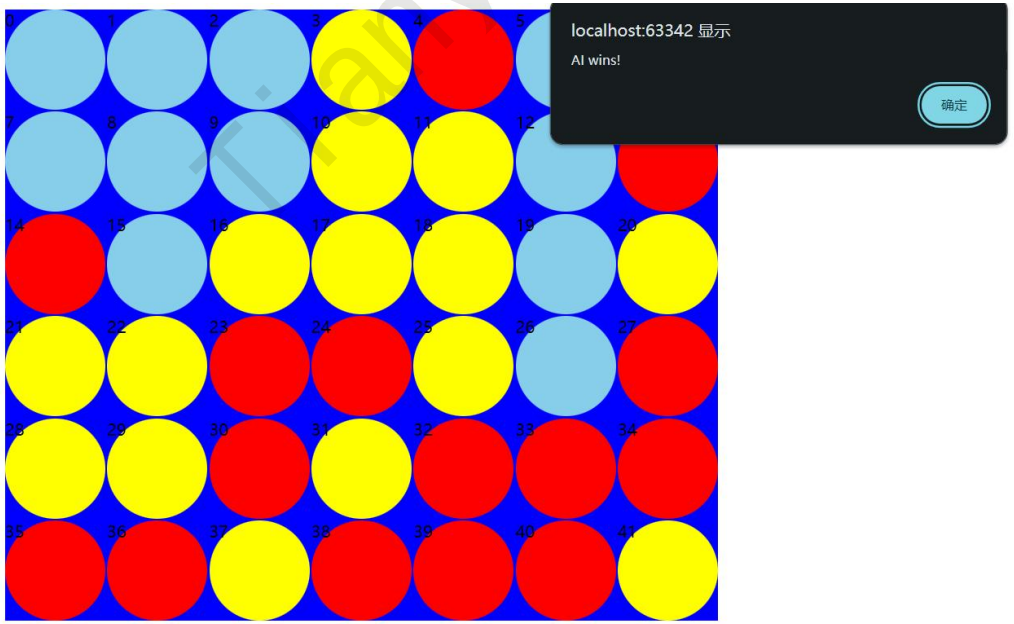
Supervisor: Sabrina Abram

Is the cerebellum involved in mentally simulating the interaction of our actions and the actions of others?

1. How do patients and healthy individuals plan their own moves in gameplay?
2. How do they predict the opponent's moves in gameplay?
3. How do these two measures develop with experience among the two groups?

Experimental design

- At first, we developed the Connect4 game. Since each player only has seven possible actions, the state space is small and there exists optimal/suboptimal moves with lower degrees of freedom in the game.



Experimental design

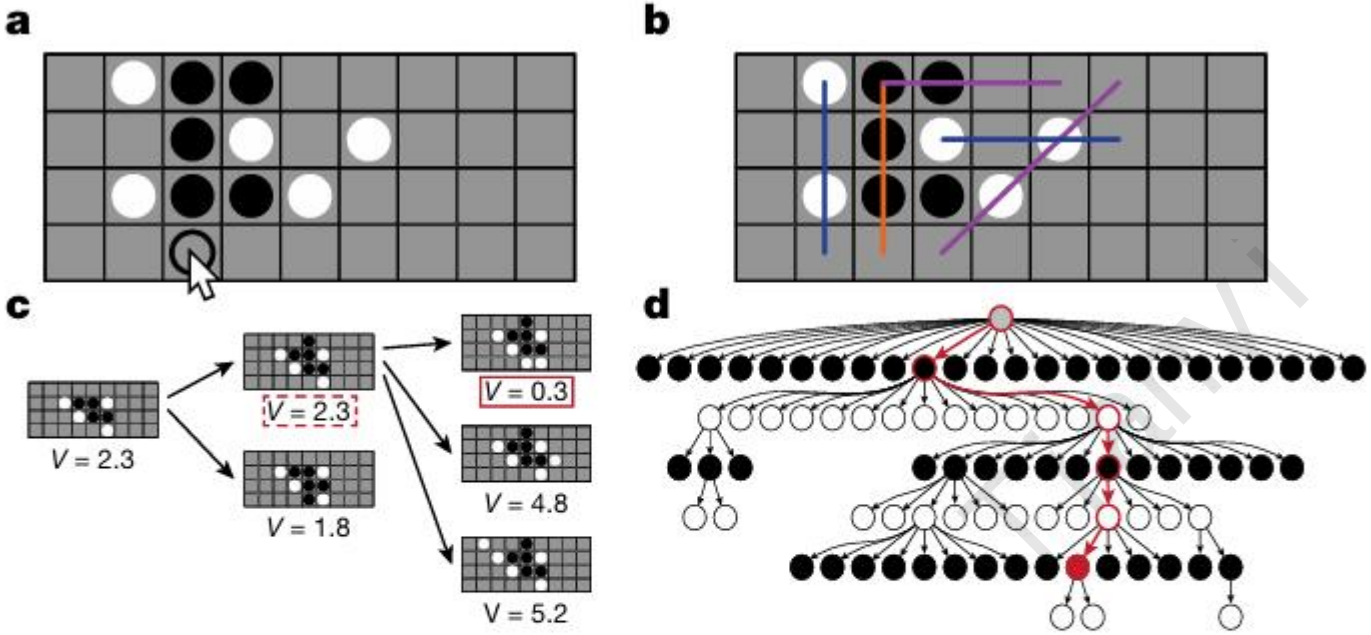


Fig. 1 | Task and computational model. a, Example board position in the four-in-a-row game. Two players, black and white, alternate placing pieces on the board, and the first player to achieve four-in-a-row wins the game. In this position, black is about to win by moving on the third square in the bottom row (open circle, mouse cursor). **b**, The features used in the heuristic function. Features with identical colours are constrained to have identical weights. The model also includes a central tendency feature and a four-in-a-row feature.

- We next turned to the four-in-a-row game developed by Wei Ji Ma's group. The experiment would consist of 60 games at the same difficulty level (a level where both older controls and patients have some success). Every 4 games they would complete 10 2AFC trials (5 predicting the opponent, 5 making their choice), where we test their knowledge of the opponent and validate the accuracy of our model's prediction.

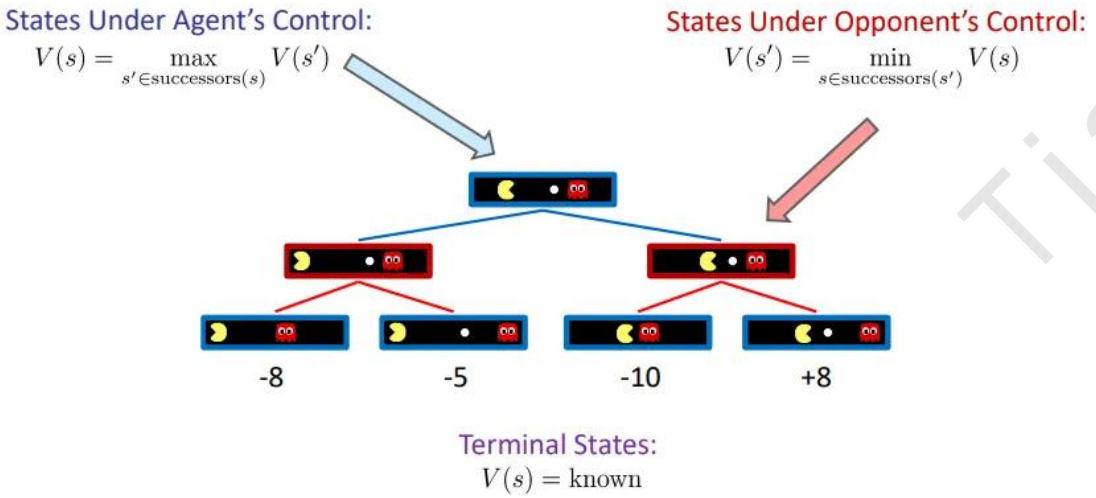
- Based on the demo they provided, I added more elements, logics and modified into the experiment that we need using JavaScript and HTML, with Firebase for the data storage.

https://test1-2c630.web.app/

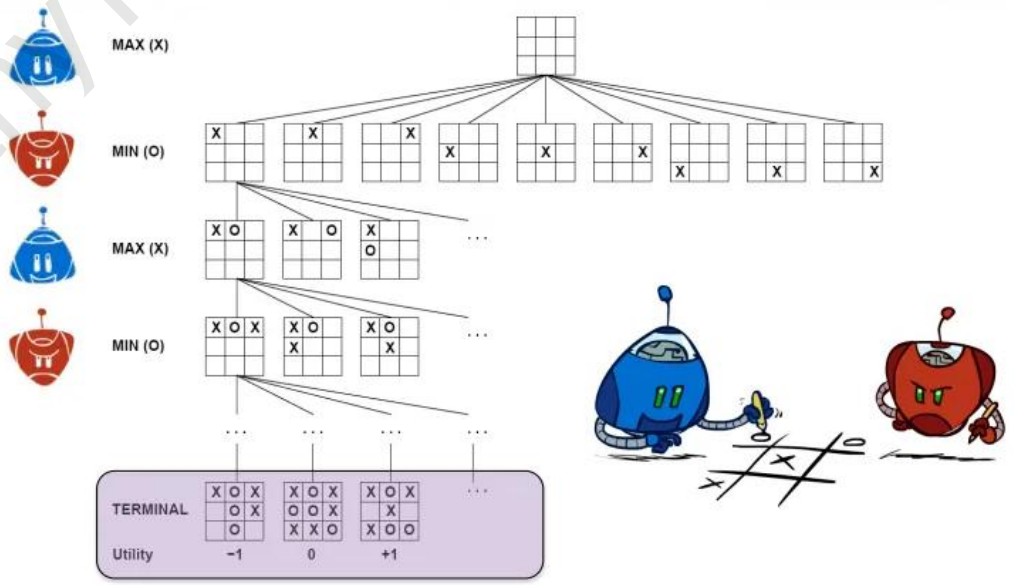
Computational cognitive model: The search tree with Best First Search (Minimax Algorithm)

- The model explores a decision tree of possible continuations.

Minimax Values:

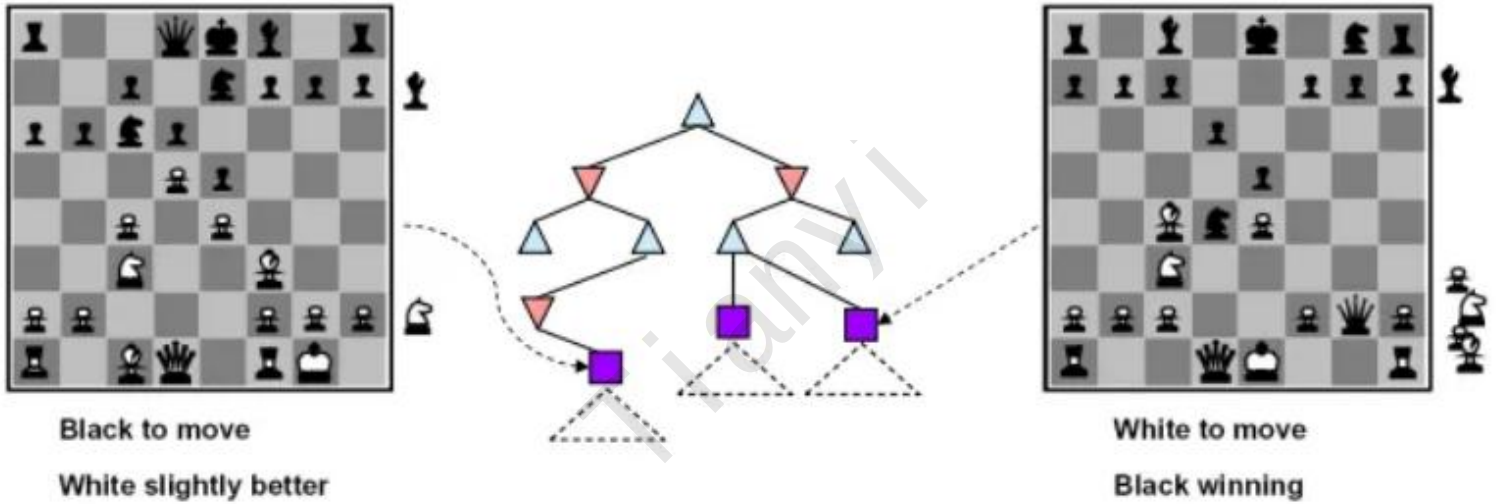


Tic-Tac-Toe Game Tree:



Evaluation Functions

- Evaluation functions score non-terminals in depth-limited search



- Ideal function: returns the actual minimax value of the position
- In practice: typically weighted linear sum of features:

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

  - E.g. $f_1(s)$ = (num white queens – num black queens), etc.
- Or a more complex nonlinear function (e.g., NN) trained by self-play RL

Evaluation Functions

We define F to be the set of all such features (one for each type, orientation and board location), and associate a weight w to each feature in this set. The feature weight depends only on its type, and not on the orientation or location. Finally, we write the value function as:

$$V_F(s) = w_{\text{centre}} V_{\text{centre}}(s) + c_{\text{black}} \sum_{i \in F} w_i f_i(s, \text{black}) - c_{\text{white}} \sum_{i \in F} w_i f_i(s, \text{white}) + \mathcal{N}(0,1) \quad (3)$$

where $c_{\text{black}} = C$ and $c_{\text{white}} = 1$ whenever black is to move in state s, and $c_{\text{black}} = 1$ and $c_{\text{white}} = C$ when it is white's move. The final term $\mathcal{N}(0,1)$ represents additive Gaussian noise with mean zero and unit variance.

First term:

Detailed model specification. Value function. The value function consists of two terms, the first of which measures whose pieces are closer to the board centre:

$$V_{\text{centre}}(s) = \sum_{\mathbf{x} \in \text{Pieces(s,black)}} \frac{1}{\|\mathbf{x} - \mathbf{x}_{\text{centre}}\|} - \sum_{\mathbf{x} \in \text{Pieces(s,white)}} \frac{1}{\|\mathbf{x} - \mathbf{x}_{\text{centre}}\|} \quad (2)$$

where Pieces(s, p) enumerates the locations of all pieces that player p owns, $\mathbf{x}_{\text{center}}$ denotes the coordinate of the board centre, and $\|\cdot\|$ is the Euclidean distance.

Evaluation Functions

We define F to be the set of all such features (one for each type, orientation and board location), and associate a weight w to each feature in this set. The feature weight depends only on its type, and not on the orientation or location. Finally, we write the value function as:

$$V_F(s) = w_{\text{centre}} V_{\text{centre}}(s) + c_{\text{black}} \sum_{i \in F} w_i f_i(s, \text{black}) - c_{\text{white}} \sum_{i \in F} w_i f_i(s, \text{white}) + \mathcal{N}(0,1) \quad (3)$$

where $c_{\text{black}} = C$ and $c_{\text{white}} = 1$ whenever black is to move in state s, and $c_{\text{black}} = 1$ and $c_{\text{white}} = C$ when it is white's move. The final term $\mathcal{N}(0,1)$ represents additive Gaussian noise with mean zero and unit variance.

Second term:

The second term counts how often particular patterns occur on the board (horizontally, vertically or diagonally). A feature is a binary function $f_{t,x,y,o}(s)$ that returns 1 if a pattern of type t occurs at location (x, y) with orientation o, and 0 otherwise. We use the following four patterns. (1) Connected two-in-a-row: two adjacent pieces with enough empty squares around them to complete four-in-a-row.

(2) Unconnected two-in-a-row: two non-adjacent pieces that lie on a line of four contiguous squares, with the remaining two squares empty. (3) Three-in-a-row: three pieces that lie on a line of four contiguous squares, with the remaining square empty. This pattern represents an immediate winning threat. (4) Four-in-a-row: four pieces in a row. This pattern appears only in board states where a player has already won the game.

Evaluation Functions

We define F to be the set of all such features (one for each type, orientation and board location), and associate a weight w to each feature in this set. The feature weight depends only on its type, and not on the orientation or location. Finally, we write the value function as:

$$V_F(s) = w_{\text{centre}} V_{\text{centre}}(s) + c_{\text{black}} \sum_{i \in F} w_i f_i(s, \text{black}) - c_{\text{white}} \sum_{i \in F} w_i f_i(s, \text{white}) + \mathcal{N}(0,1) \quad (3)$$

where $c_{\text{black}} = C$ and $c_{\text{white}} = 1$ whenever black is to move in state s, and $c_{\text{black}} = 1$ and $c_{\text{white}} = C$ when it is white's move. The final term $\mathcal{N}(0,1)$ represents additive Gaussian noise with mean zero and unit variance.

In conclusion, we have 10 parameters to estimate:

1. 5 feature weights
2. active-passive scaling constant C
3. pruning threshold θ
4. stopping probability γ
5. feature drop rate δ
6. the lapse rate γ

```
function theta = pad_input(theta)
%PAD_INPUT Add other fixed parameters for four-in-a-row model.

g=sprintf('%f ', theta);
fprintf('Theta = %s\n', g)

thresh = theta(1);
delta = theta(3);
w_center = theta(6);
w = [theta(7); theta(8); theta(9); theta(10)];
lambda = theta(4);
c_act = theta(5);
gamma = theta(2);
theta=[10000;  thresh; gamma; lambda; 1; 1; w_center; repmat(w,4,1); 0; c_act*repmat(w,4

end
```

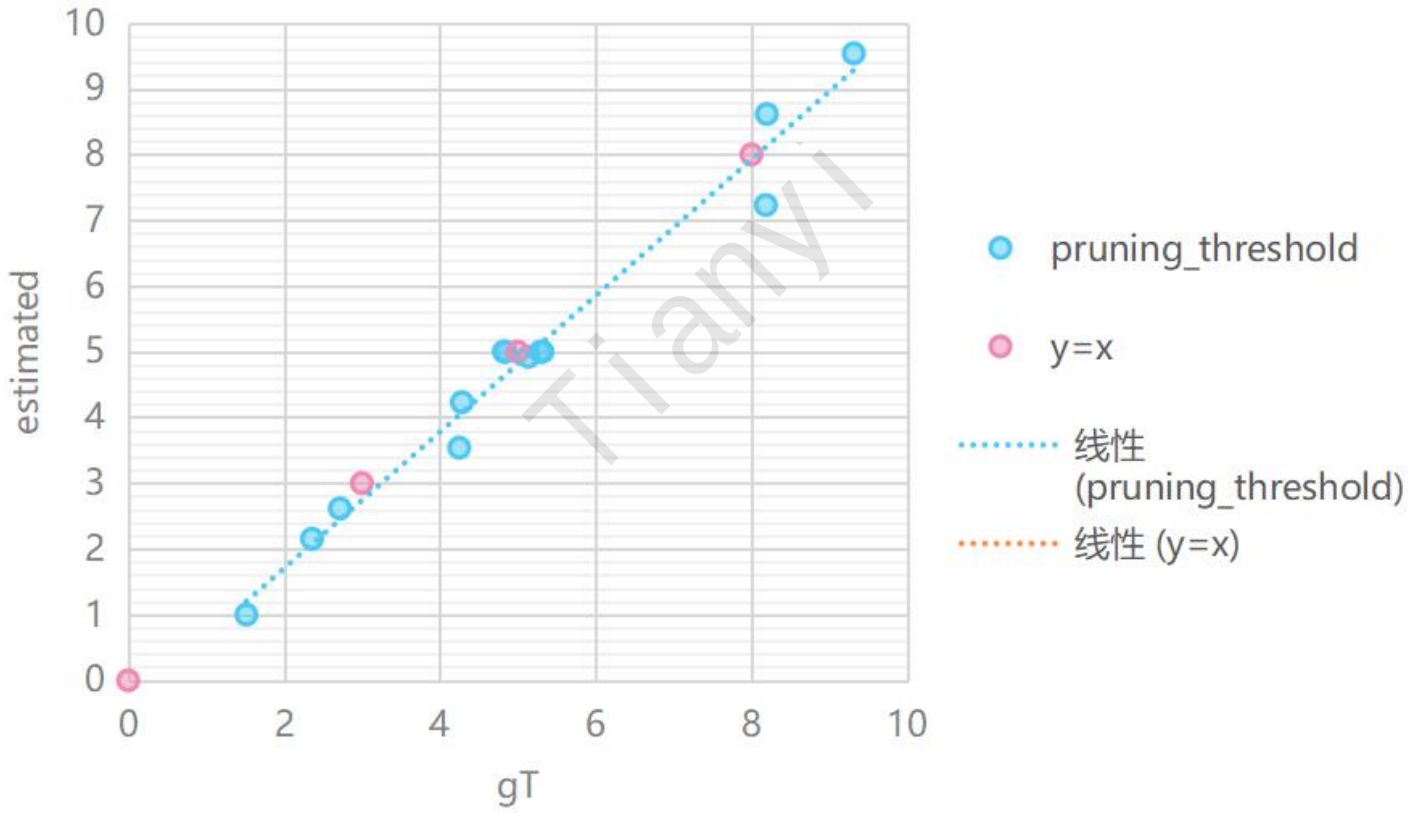
Looking deeper into the parameters

In conclusion, we have 10 parameters to estimate:

1. 5 feature weights
2. active-passive scaling constant C
3. pruning threshold θ
4. stopping probability γ
5. feature drop rate δ
6. the lapse rate γ

- **planning depth:** the length of the principal variation in the model's decision tree (averaged across simulations of the model with a given a parameter vector in a fixed set of probe positions), which primarily depends on *pruning threshold θ* and *stopping probability γ* (could be calculated)
- **feature drop rate:** parameter δ
- **pruning threshold:** parameter θ

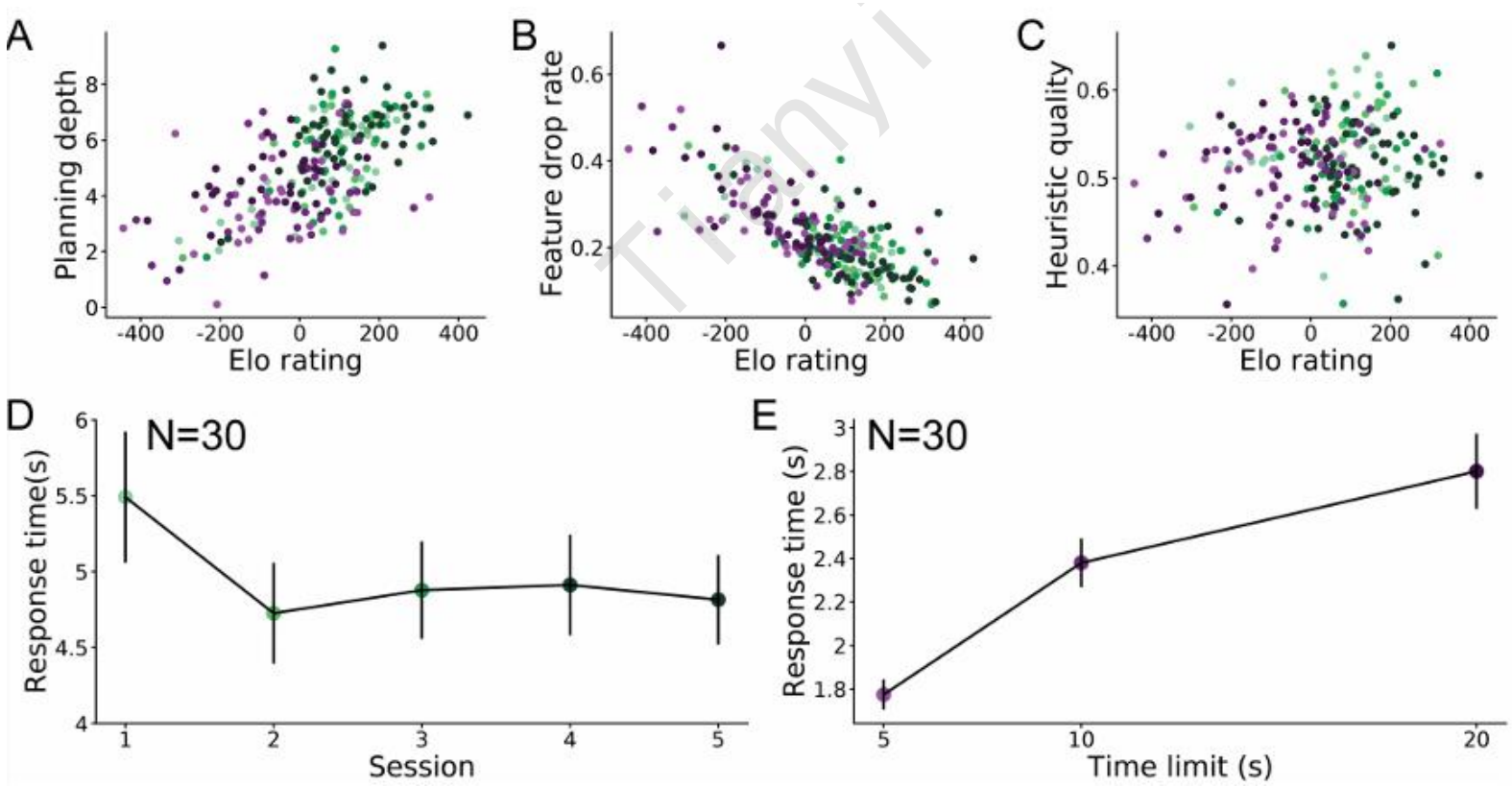
Parameter recovery



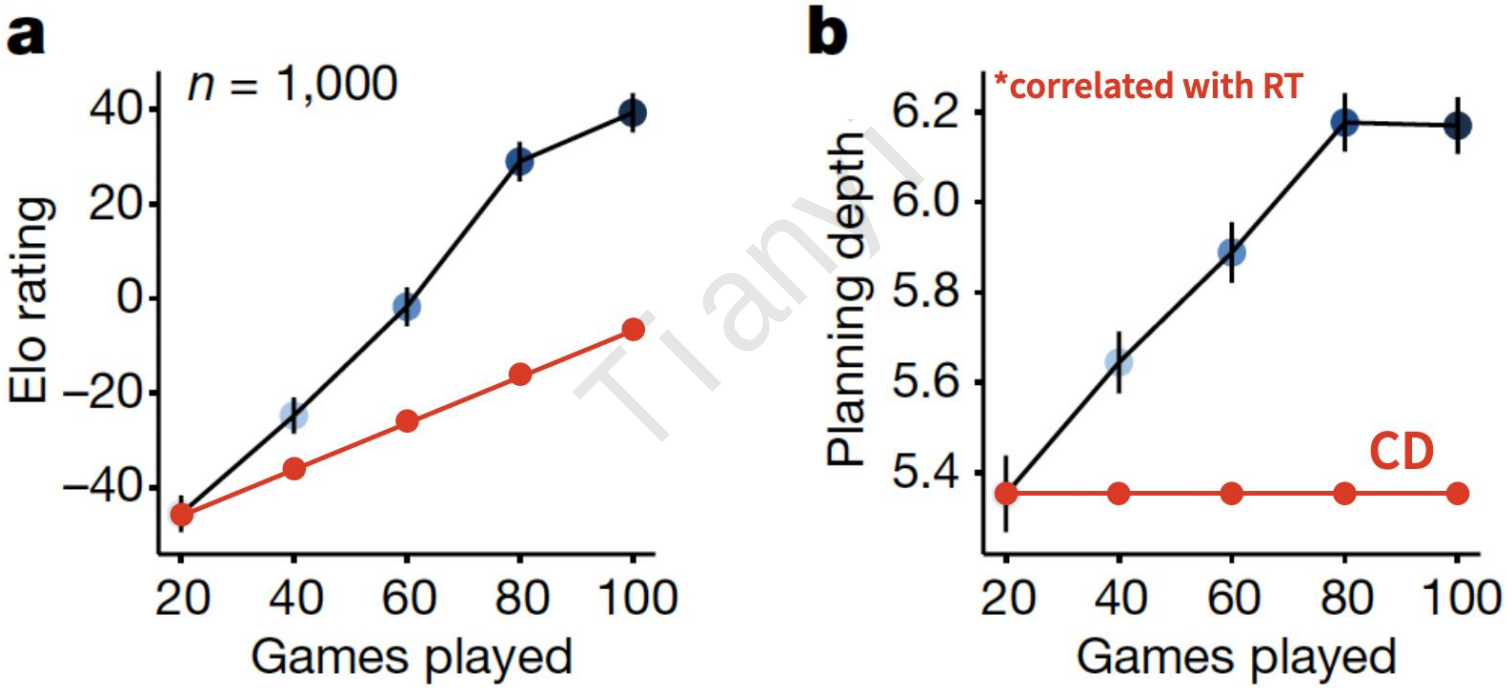
Using the dataset of 500 moves with different real pruning thresholds.

Parameter estimation

Elo rating: estimating a player's playing strength from games against computer opponents. participants' Elo rating correlates strongly with **planning depth** and **feature drop rate.**

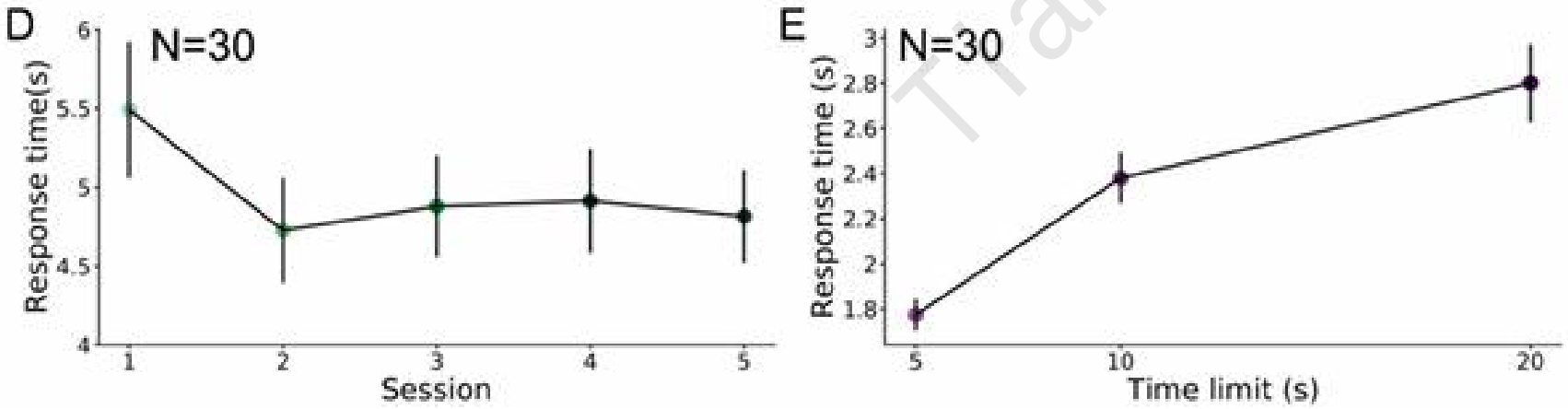


Learning impairment in cerebellar patients?



Prospect—response time

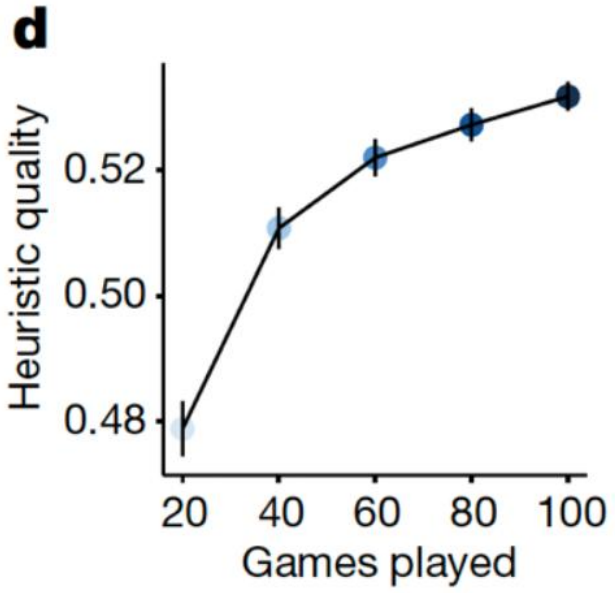
Participants play slightly faster in later sessions. Therefore, our finding of increased planning in later sessions is not confounded by an increase in thinking time as people appear to plan more while using less time.

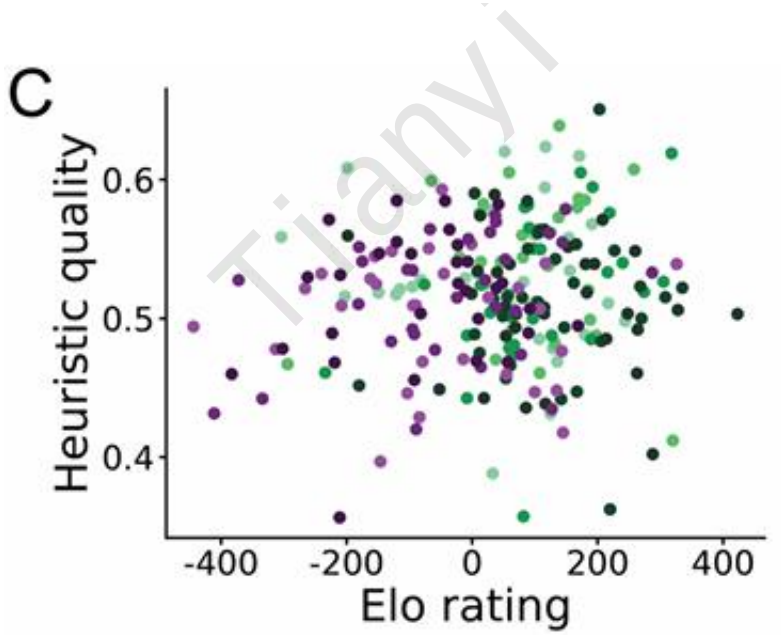


The time limit manipulation is effective at increasing participants' response times, even though they use only a fraction of the available time on average.

Prospect—social interactions

The current features only consider horizontal, vertical or diagonal patterns. But what if we include patterns that can be considered offensive or defensive? These are strategies that involve inferring the ”personality” of the opponent.





The exact effect of heuristic quality remains unknown.

Conclusion

Our experiment is based on the four-in-a-row game with search tree model as the computational cognitive model.

We plan to use this method to explore performance differences between cerebellar patients and healthy individuals in gameplay.

However, the experimental design is ongoing and in need of more theoretical motivation.

Thank you Sabrina for all your supports and inspirations!
I’ve had an extremely wonderful research experience in CognAc Lab!
And thank you Prof. Jonathan Tsay for your feedback and patience!