

Contents

- [Step 1: Import CSV Data](#)
- [Step 2: R Bridge Implementation](#)
- [Step 3: MDTF Formulation to Calculate Preference Dynamics](#)
- [Helper Functions](#)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Main Simulation Script for BoundingOverwatch Project with R Integration
% Randomly 10 trials is used to validate predictions
% DFT Parameters:
% phi1 - sensitivity to attribute differences (typically 0.5-2)
% phi2 - memory/feedback strength (0-1)
% tau - decision time steps (integer > 0)
% error_sd - noise standard deviation (sigma_epsilon)
% beta - attribute weights from R estimation
% w - attention weights (default [0.5;0.5])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc;
clear all;
```

Step 1: Import CSV Data

(reference apolloMain_5 amd apolloMain_6 as example for data manipulation) biasData = readtable('user_choices.csv'); % Replace with the path to your data file disp('User bias data imported successfully.');

taskChoice_Data = readtable('user_choices.csv'); % Replace with the path to your data file disp('User task choice data imported successfully.');

```
robotChoice_Data = readtable('G:\My Drive\myResearch\Research Experimentation\Apollo\apollo\data\Bounding_Overwatch_Data\HumanData_Bounding_Overwatch_NORMALIZED.csv');
% Convert all column headers to lowercase
robotChoice_Data.Properties.VariableNames = lower(robotChoice_Data.Properties.VariableNames);
disp('User robot choice data imported successfully.');
```

% Randomly select 10 rows (or all rows if fewer than 10)

```
numRows = height(robotChoice_Data);
randomIndices = randperm(numRows, min(10, numRows));
robotChoice_Data = robotChoice_Data(randomIndices, :);
```

% Extract robot state attributes dynamically

```
robot_states = struct();
attributeSuffixes = {'traversability', 'visibility'}; % No leading underscores
for i = 1:3
    attr = attributeSuffixes
    csvColName = sprintf('robot%d_%s', i, attr{1}); % Matches CSV column names
    structFieldName = attr{1}; % Valid field name
    if ismember(csvColName, robotChoice_Data.Properties.VariableNames)
        robot_states.([ 'robot' num2str(i) ]). (structFieldName) = robotChoice_Data.(csvColName);
    else
        warning('Missing attribute column: %s', csvColName);
        robot_states.([ 'robot' num2str(i) ]). (structFieldName) = NaN(height(robotChoice_Data), 1);
    end
end
end
```

% Extract choice data and other metadata

```
choices = robotChoice_Data.choice;
participant_ids = robotChoice_Data.id;
stake_types = robotChoice_Data.stakes;
time_spent = robotChoice_Data.timeelapsed;
```

User robot choice data imported successfully.

Step 2: R Bridge Implementation

```
disp('Initializing R bridge...');
```

% Configure paths

```
rscript_path = 'C:\Program Files\R\R-4.4.2\bin\x64\Rscript.exe';
r_script = 'G:\My Drive\myResearch\Research Experimentation\Apollo\apollo\example\DFT_Bounding_Overwatch.R';
csvFile = 'G:\My Drive\myResearch\Research Experimentation\Apollo\apollo\data\Bounding_Overwatch_Data\HumanData_Bounding_Overwatch_NORMALIZED.csv';
outputDir = 'G:\My Drive\myResearch\Research Experimentation\Apollo\apollo\Output_BoundingOverwatch';
```

% Verify installations

```
if ~isfile(rscript_path)
    error('Rscript.exe not found at: %s', rscript_path);
elseif ~isfile(r_script)
    error('R script not found at: %s', r_script);
elseif ~isfile(csvFile)
    error('Input CSV not found at: %s', csvFile);
elseif ~isfolder(outputDir)
    warning('Output folder does not exist, creating: %s', outputDir);
    mkdir(outputDir);
```

```

end

% Execute R with JSON output
try
    % Use proper argument formatting
    cmd = sprintf(['"%s" "%s" ', ...
        '-i "%s" -o "%s"', ...
        rscript_path, r_script, csvFile, outputDir]);

    [status,result] = system(cmd);

    if status == 0
        % Handle output path (whether directory or file)
        if isfolder(outputDir)
            jsonFile = fullfile(outputDir, 'DFT_output.json');
        else
            jsonFile = outputDir;
        end

        % Parse JSON output
        if exist(jsonFile, 'file')
            jsonText = fileread(jsonFile);
            params = jsondecode(jsonText);

            % Extract parameters with validation
            %Boundedphi1, phi2 parameters
            phi1 = max(0, validateParam(params, 'phi1', 0.5)); % Ensure non-negative
            phi2 = min(max(0, validateParam(params, 'phi2', 0.8)), 1); % Constrain 0-1

            %Raw phi1, phi2 parameters
            %phi1 = validateParam(params, 'phi1', 0.5);
            %phi2 = validateParam(params, 'phi2', 0.8);
            tau = 1 + exp(validateParam(params, 'timesteps', 0.5));
            error_sd = min(max(0.1, validateParam(params, 'error_sd', 0.1)), 1); % still clip here

            % Extract attribute weights
            beta_weights = [
                params.b_attr1;
                params.b_attr2;
                params.b_attr3;
                params.b_attr4
            ];

            % Get initial preferences from ASCs
            initial_P = [
                validateParam(params, 'asc_1', 0);
                validateParam(params, 'asc_2', 0);
                validateParam(params, 'asc_3', 0);
            ];

            disp('Estimated Parameters:');
            disp(['phi1: ', num2str(phi1)]);
            disp(['phi2: ', num2str(phi2)]);
            disp(['tau: ', num2str(tau)]);
            disp(['error_sd: ', num2str(error_sd)]);
            disp('Initial Preferences (from ASCs):');
            disp(initial_P);
        else
            error('R output file not found');
        end
    else
        error('R execution failed: %s', result);
    end
catch ME
    disp('Error during R execution:');
    disp(getReport(ME, 'extended'));
    [phi1, phi2, tau, error_sd] = getFallbackParams();
    beta_weights = [0.3; 0.2; 0.4; 0.5]; % Default weights
    initial_P = zeros(3,1); % Neutral initial preferences
end

```

Initializing R bridge...

Step 3: MDFT Formulation to Calculate Preference Dynamics

(MDFT calculations based on estimated parameters) Create M matrix from current trial's attributes C11-C14 are consequence attributes for Robot 1 C21-C24 are consequence attributes for Robot 2 C31-C34 are consequence attributes for Robot 3

```

for current_trial = 1:height(robotChoice_Data)
    num_attributes = 4;

    M = [
        robotChoice_Data.c11(current_trial), robotChoice_Data.c12(current_trial), robotChoice_Data.c13(current_trial), robotChoice_Data.c14(current_trial);
        robotChoice_Data.c21(current_trial), robotChoice_Data.c22(current_trial), robotChoice_Data.c23(current_trial), robotChoice_Data.c24(current_trial);

```

```

robotChoice_Data.c31(current_trial), robotChoice_Data.c32(current_trial), robotChoice_Data.c33(current_trial), robotChoice_Data.c34(current_trial)
];

%M = M / 100;           % Convert all values from 0-100 to 0.00-1.00 scale

attributes = {'C1 - Easy Nav, Low Exposure', 'C2 - Hard Nav, Low Exposure', 'C3 - Easy Nav, High Exposure', 'C4 - Hard Nav, High Exposure'};
beta = beta_weights ./ sum(abs(beta_weights));
beta = beta';

[E_P, V_P, choice_probs, P_tau] = calculateDFTdynamics(...
    phi1, phi2, tau, error_sd, beta, M, initial_P);

% Display results for the trial
disp('=== Trial Analysis ===');
disp(['Trial: ', num2str(current_trial)]);
disp(['Participant: ', num2str(participant_ids(current_trial))]);
disp(['Actual Choice: Robot ', num2str(choices(current_trial))]);

disp('M matrix (alternatives x attributes):');
disp(array2table(M, ...
    'RowNames', {'Robot1','Robot2','Robot3'}, ...
    'VariableNames', attributes));

disp('DFT Results:');
disp(['E_P: ', num2str(E_P), '%.2f ']);
disp(['Choice probabilities: ', num2str(choice_probs), '%.3f ']);
[~, predicted_choice] = max(choice_probs);
disp(['Predicted choice: Robot ', num2str(predicted_choice)]);
disp(['Actual choice: Robot ', num2str(choices(current_trial))]);
disp(' ');

if predicted_choice == choices(current_trial)
    disp('✓ Prediction matches actual choice');
else
    disp('X Prediction differs from actual choice');
end

% Plot evolution
figure;
plot(0:tau, P_tau);
xlabel('Preference Step (\tau)');
ylabel('Preference Strength');
legend({'Robot1','Robot2','Robot3'});
title(sprintf('Preference Evolution (Trial %d)', current_trial));
grid on;
end

%{
%% Step 4: Output Results
disp('Saving results to CSV...');
output_table = table(E_P, V_P, P_tau(end,:), ...
    'VariableNames', {'ExpectedPreference', 'VariancePreference', 'FinalPreferences'});
writetable(output_table, 'results.csv');
disp('Results saved successfully!');
%}

```

```

=== Trial Analysis ===
Trial: 1
Participant: 124737
Actual Choice: Robot 1
M matrix (alternatives x attributes):

```

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.2459	1	0.37705	0
Robot2	0.2623	0.39344	0.22951	0.7541
Robot3	0.32787	0.65574	0.21311	0.45902

```

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 1

✓ Prediction matches actual choice
=== Trial Analysis ===
Trial: 2
Participant: 123310
Actual Choice: Robot 1
M matrix (alternatives x attributes):

```

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.42105	0.47368	0.33333	0.52632
Robot2	0.4386	0.22807	0.38596	0.7193
Robot3	0.2807	0.15789	0.31579	1

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 1

✓ Prediction matches actual choice

=== Trial Analysis ===

Trial: 3

Participant: 123310

Actual Choice: Robot 1

M matrix (alternatives × attributes):

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0	0.37705	0.4918	0.7541
Robot2	0.065574	0.11475	0.91803	0.52459
Robot3	0.32787	1	0.16393	0.14754

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 1

✓ Prediction matches actual choice

=== Trial Analysis ===

Trial: 4

Participant: 141831

Actual Choice: Robot 2

M matrix (alternatives × attributes):

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.086957	0.47826	0.11594	0.78261
Robot2	0.23188	0.2029	0.014493	1
Robot3	0.27536	0.7971	0.30435	0.072464

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 2

X Prediction differs from actual choice

=== Trial Analysis ===

Trial: 5

Participant: 125802

Actual Choice: Robot 1

M matrix (alternatives × attributes):

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.22785	0.025316	0.025316	1
Robot2	0.012658	0.35443	0.075949	0.82278
Robot3	0.48101	0.12658	0.48101	0.17722

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 1

✓ Prediction matches actual choice

=== Trial Analysis ===

Trial: 6

Participant: 125802

Actual Choice: Robot 3

M matrix (alternatives × attributes):

	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.40426	0.70213	0.10638	0.91489
Robot2	0.38298	0.7234	0.021277	1
Robot3	0.38298	0.7234	0.042553	0.97872

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 3

X Prediction differs from actual choice

=== Trial Analysis ===

Trial: 7

Participant: 124737

Actual Choice: Robot 3

M matrix (alternatives × attributes):

C1 - Easy Nav, Low Exposure

C2 - Hard Nav, Low Exposure

C3 - Easy Nav, High Exposure

C4 - Hard Nav, High Exposure

Robot1	0.85417	0.1875	1	0.041667
Robot2	0.27083	0.97917	0.375	0.45833
Robot3	0.083333	0.79167	0.58333	0.625

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 3

X Prediction differs from actual choice

=== Trial Analysis ===

Trial: 8

Participant: 124737

Actual Choice: Robot 2

M matrix (alternatives × attributes):

C1 - Easy Nav, Low Exposure

C2 - Hard Nav, Low Exposure

C3 - Easy Nav, High Exposure

C4 - Hard Nav, High Exposure

Robot1	0.18367	0.61224	0.30612	0.95918
Robot2	0.36735	1	0.40816	0.26531
Robot3	0.40816	0.81633	0.26531	0.57143

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 2

X Prediction differs from actual choice

=== Trial Analysis ===

Trial: 9

Participant: 141831

Actual Choice: Robot 2

M matrix (alternatives × attributes):

C1 - Easy Nav, Low Exposure

C2 - Hard Nav, Low Exposure

C3 - Easy Nav, High Exposure

C4 - Hard Nav, High Exposure

Robot1	0.15789	0.64912	0.29825	0.64912
Robot2	0.15789	1	0.12281	0.45614
Robot3	0.36842	0.98246	0.2807	0.12281

DFT Results:

E_P: NaN NaN NaN

Choice probabilities: NaN NaN NaN

Predicted choice: Robot 1

Actual choice: Robot 2

X Prediction differs from actual choice

=== Trial Analysis ===

Trial: 10

Participant: 142426

Actual Choice: Robot 2

M matrix (alternatives × attributes):

C1 - Easy Nav, Low Exposure

C2 - Hard Nav, Low Exposure

C3 - Easy Nav, High Exposure

C4 - Hard Nav, High Exposure

Robot1	0.2766	1	0.06383	0.78723
Robot2	0.21277	0.76596	0.2766	0.87234
Robot3	0.40426	0.55319	0.42553	0.74468

DFT Results:

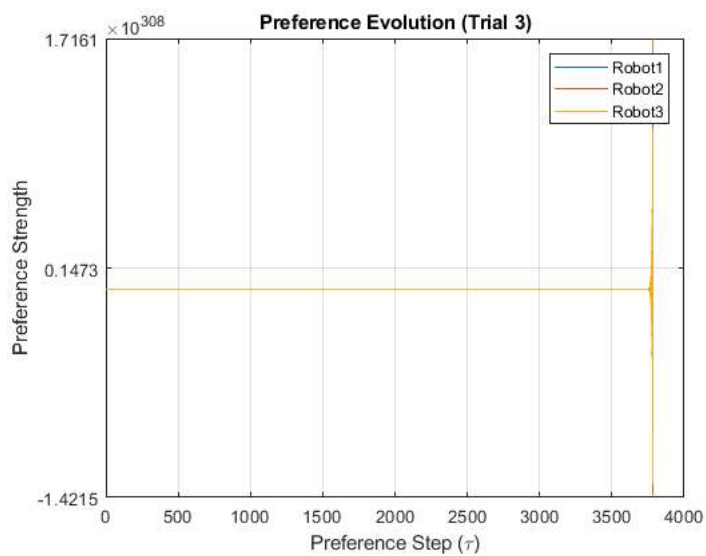
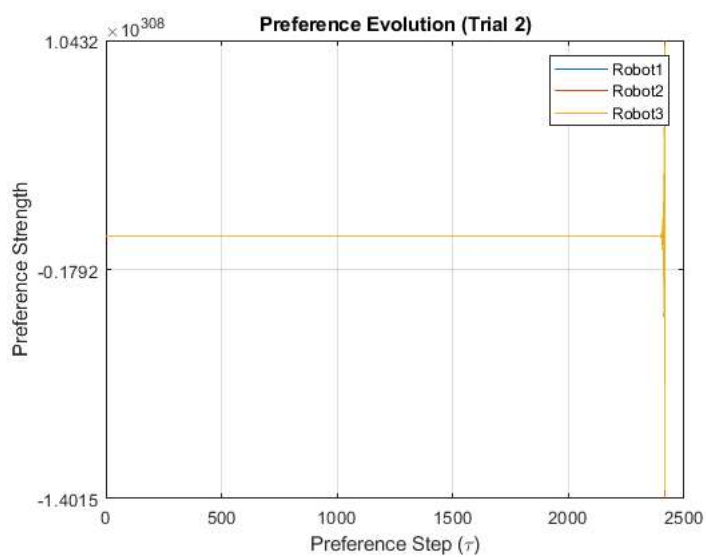
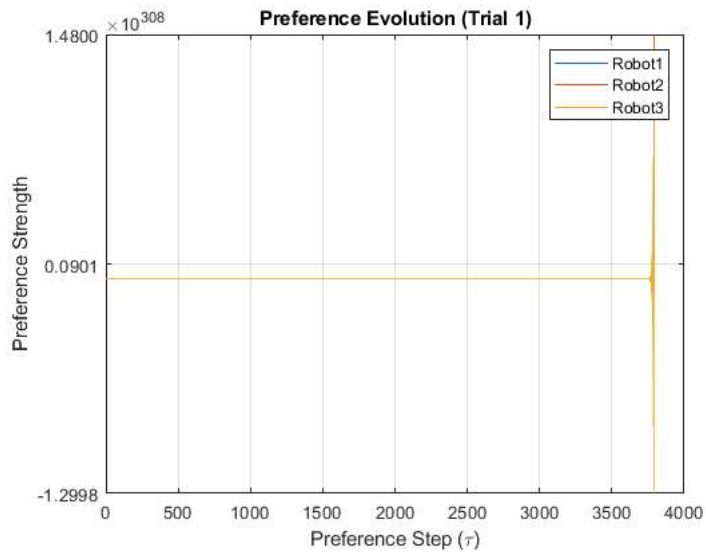
E_P: NaN NaN NaN

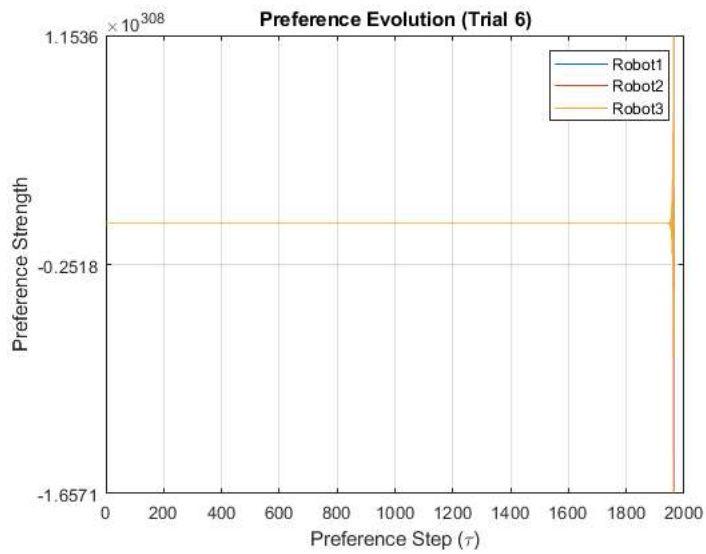
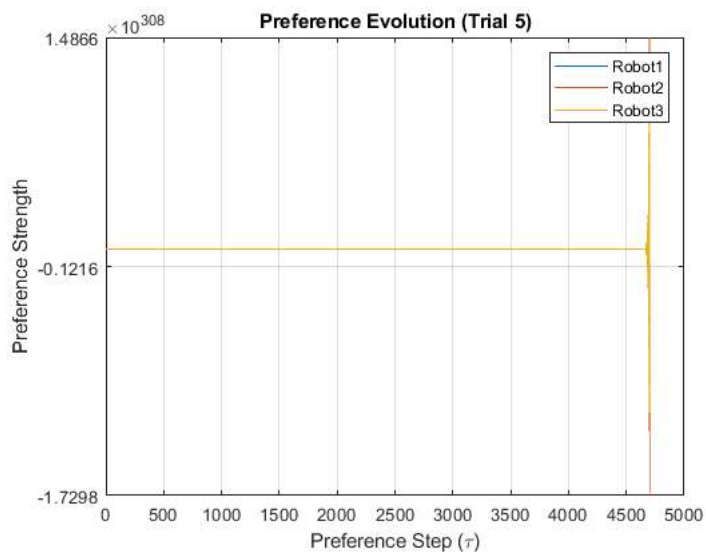
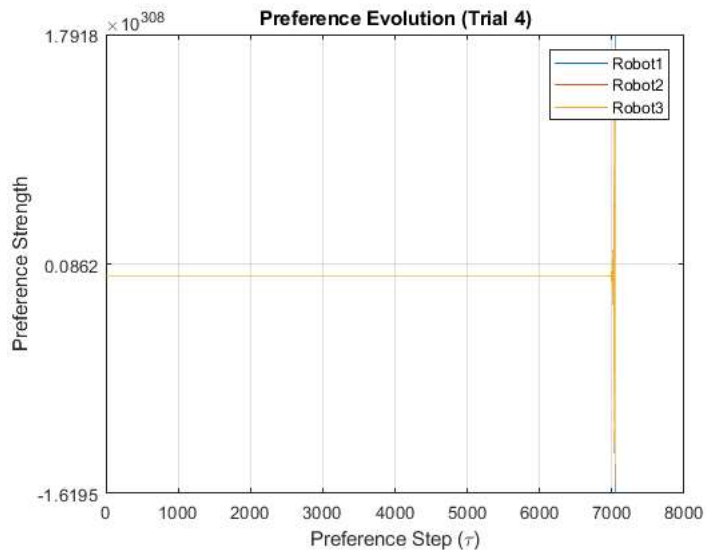
Choice probabilities: NaN NaN NaN

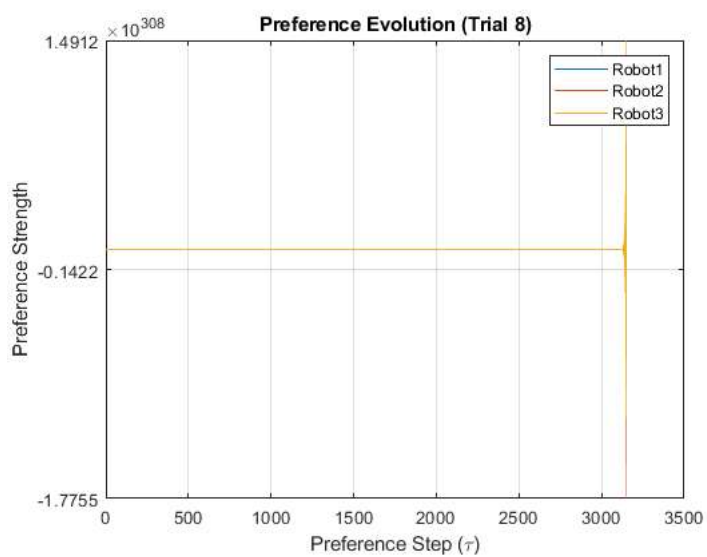
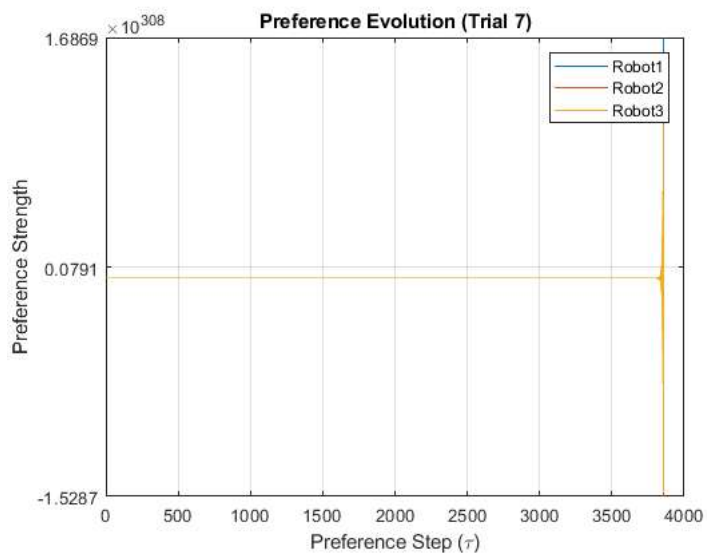
Predicted choice: Robot 1

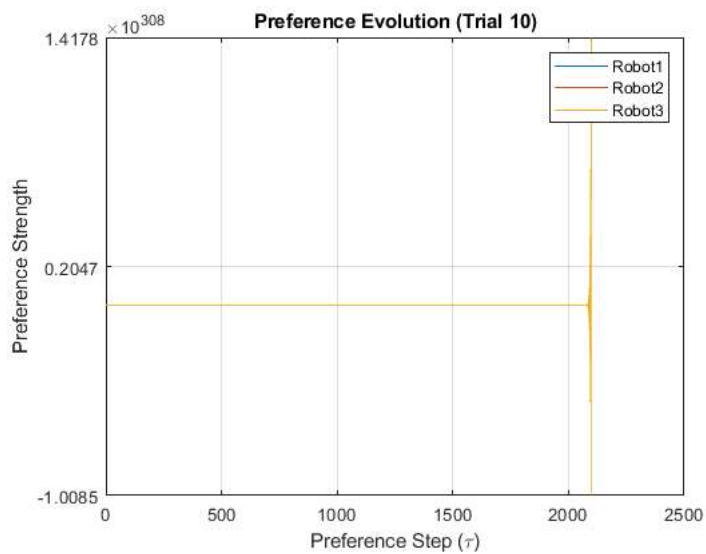
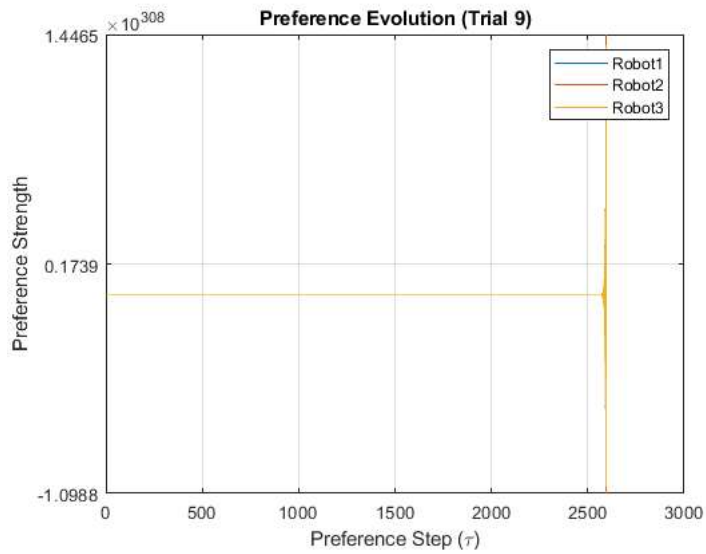
Actual choice: Robot 2

X Prediction differs from actual choice









Helper Functions

```
function param = validateParam(params, name, default)
    if isfield(params, name) && isnumeric(params.(name))
        param = params.(name);
    else
        warning('Using default for %s', name);
        param = default;
    end
end

function [phi1, phi2, tau, error_sd] = getFallbackParams()
    phi1 = 0.5 + 0.1*randn();
    phi2 = 0.8 + 0.1*randn();
    tau = 10 + randi(5);
    error_sd = 0.1 + 0.05*rand();
    warning('Using randomized default parameters');
end
```

Estimated Parameters:
 phi1: 2.277
 phi2: 0.8135
 tau: 22027.4658
 error_sd: 0.1301
 Initial Preferences (from ASCs):
 -0.1321 0.0606 0

