Contents


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 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 (σ_ε)
% 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 - 80Split.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
    for 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 - 80Split.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)
    ];

    % Normalize M values by dividing by 2 and clamping to [0.01, 1]
    %M = M / 2;
    %M = max(0.01, min(1, M));

    % --- Global Max Normalization ---
    global_max = max(robotChoice_Data{:, {'c11','c12','c13','c14','c21','c22','c23','c24','c31','c32','c33','c34'}}, [], 'all', 'omitnan');
    if ~isfinite(global_max) || global_max <= 0
        global_max = 1; % fallback in case of zero or NaN
    end

    M = M / global_max;             % Normalize by global max
    M = max(0.01, min(1, M));       % Clamp to [0.01, 1]

    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 × 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('✗ 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: 175044
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              0.53131                        0.27442                         0.31001                         0.053131
    Robot2              0.56469                        0.34829                         0.27287                         0.056469
    Robot3              0.53834                        0.28146                         0.31072                         0.053834

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: 175044
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	0.4991	0.26554	0.28347	0.04991
Robot2	0.49456	0.2957	0.24832	0.049456
Robot3	0.48951	0.26006	0.2784	0.048951

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 3
Participant: 141831
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	0.64723	0.39436	0.31759	0.064723
Robot2	0.55374	0.32678	0.28233	0.055374
Robot3	0.6102	0.38052	0.29071	0.06102

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 3

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 4
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	0.57548	0.32031	0.31272	0.057548
Robot2	0.57667	0.32959	0.30474	0.057667
Robot3	0.59855	0.33528	0.32312	0.059855

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 3

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 5
Participant: 181700
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	1	0.57644	0.52356	0.1
Robot2	0.94014	0.55706	0.47709	0.094014
Robot3	0.8242	0.4816	0.42501	0.08242

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 3

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 6
Participant: 123310
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	0.41557	0.25914	0.19799	0.041557
Robot2	0.39007	0.23465	0.19442	0.039007
Robot3	0.48055	0.31041	0.21819	0.048055

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 7
Participant: 214504
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	0.48079	0.30381	0.22505	0.048079
Robot2	0.47977	0.2866	0.24114	0.047977
Robot3	0.050229	0.03518	0.020072	0.01

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: 8
Participant: 123310
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	0.457	0.29531	0.20739	0.0457
Robot2	0.44162	0.3	0.18579	0.044162
Robot3	0.42026	0.25048	0.21181	0.042026

DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 3

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 9
Participant: 124737
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	0.40595	0.22564	0.2209	0.040595
Robot2	0.38863	0.19213	0.23536	0.038863
Robot3	0.44683	0.22874	0.26277	0.044683

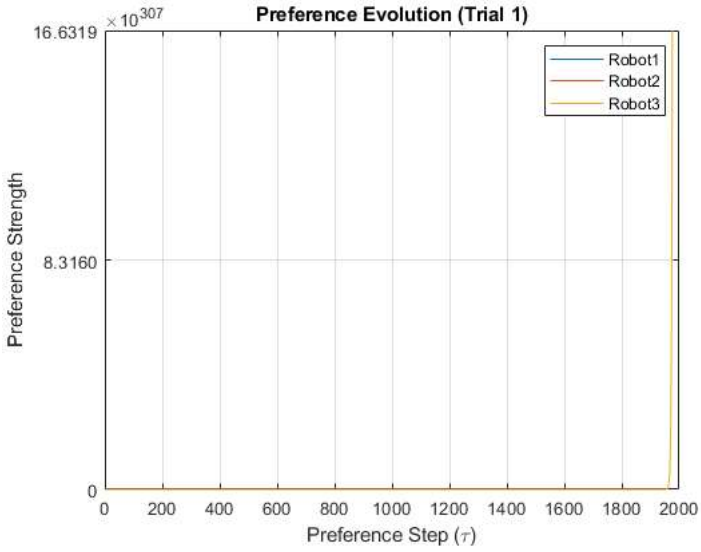
DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 1

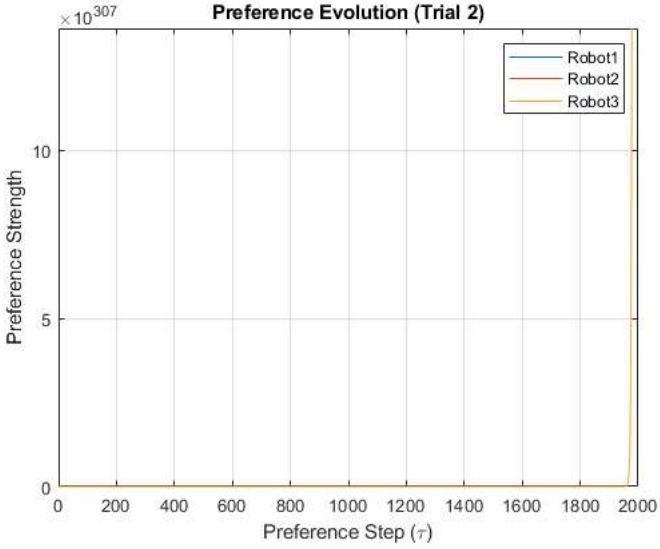
✓ Prediction matches actual choice
=== Trial Analysis ===
Trial: 10
Participant: 214504
Actual Choice: Robot 2
M matrix (alternatives × attributes):

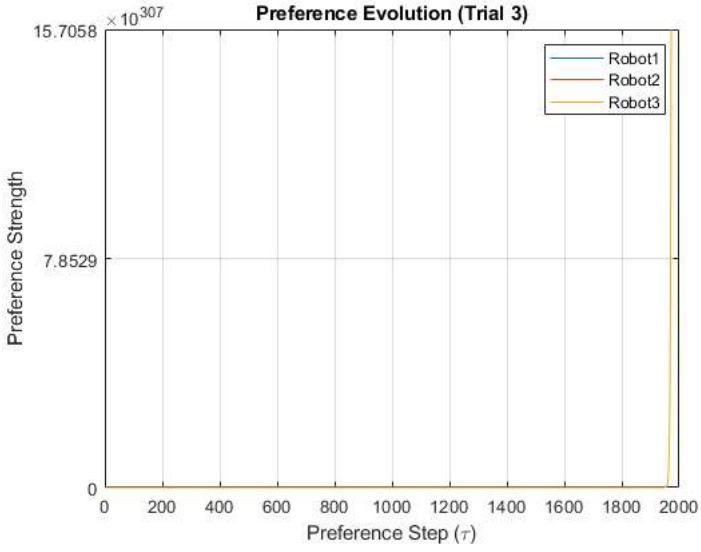
	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.26733	0.16662	0.12744	0.026733
Robot2	0.10843	0.064393	0.054875	0.010843
Robot3	0.10486	0.068926	0.046422	0.010486

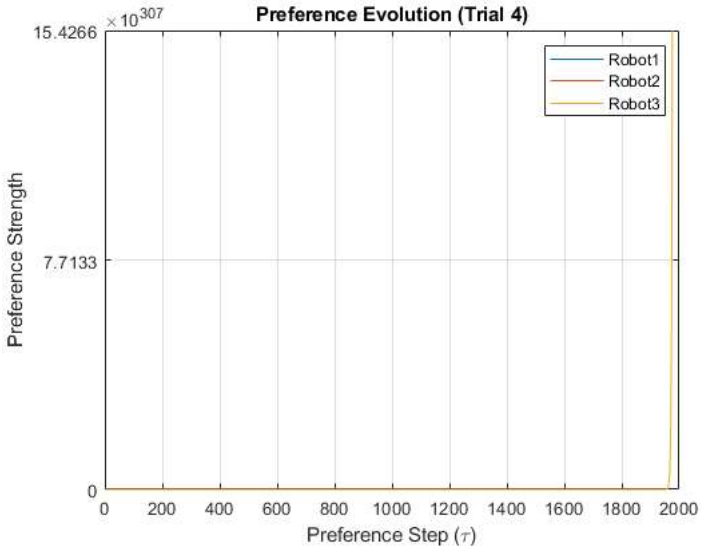
DFT Results:
E_P: NaN NaN NaN
Choice probabilities: NaN NaN NaN
Predicted choice: Robot 1
Actual choice: Robot 2

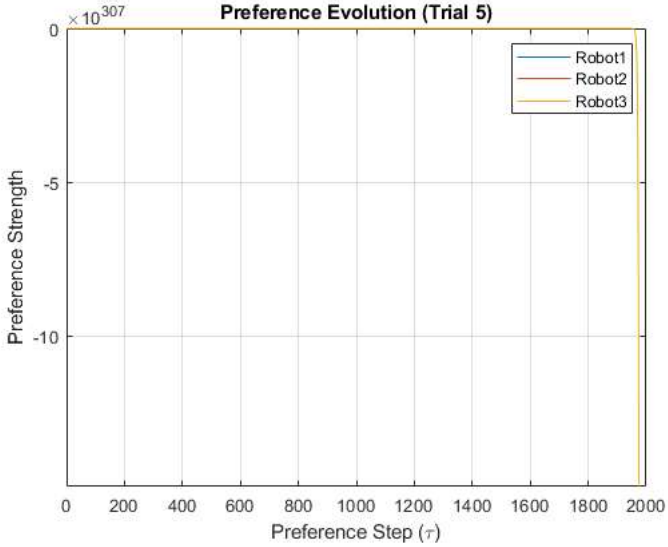
✗ Prediction differs from actual choice

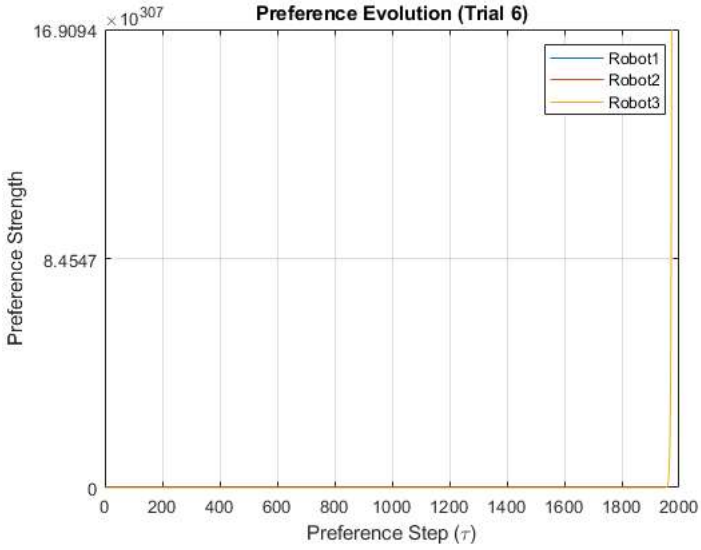

Preference Evolution (Trial 1)
×10^307
16.6319
8.3160
0
Preference Strength
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Preference Step (τ)
Robot1
Robot2
Robot3

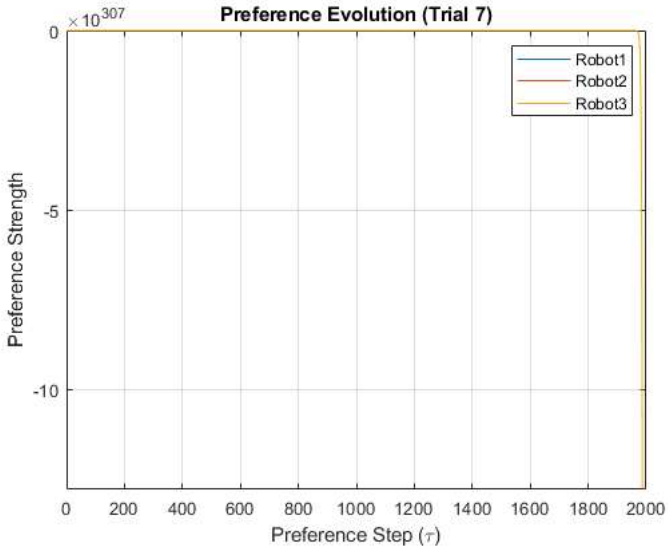

Preference Evolution (Trial 2)
×10^307
10
5
0
Preference Strength
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Preference Step (τ)
Robot1
Robot2
Robot3

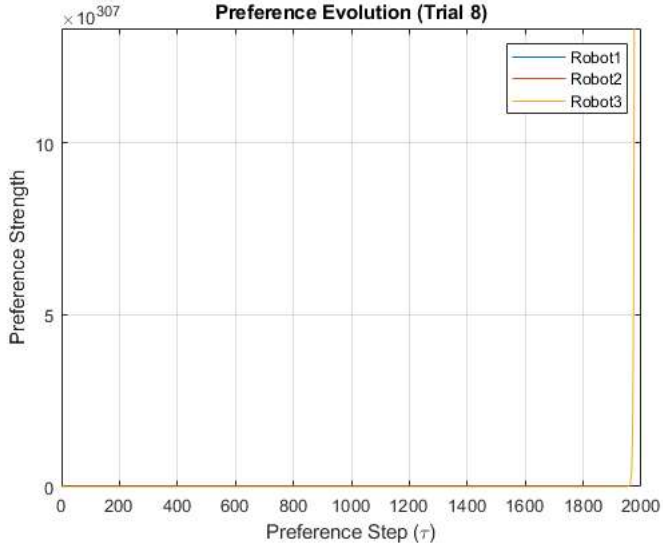

Preference Evolution (Trial 3)
×10^307
15.7058
7.8529
0
Preference Strength
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Preference Step (τ)
Robot1
Robot2
Robot3

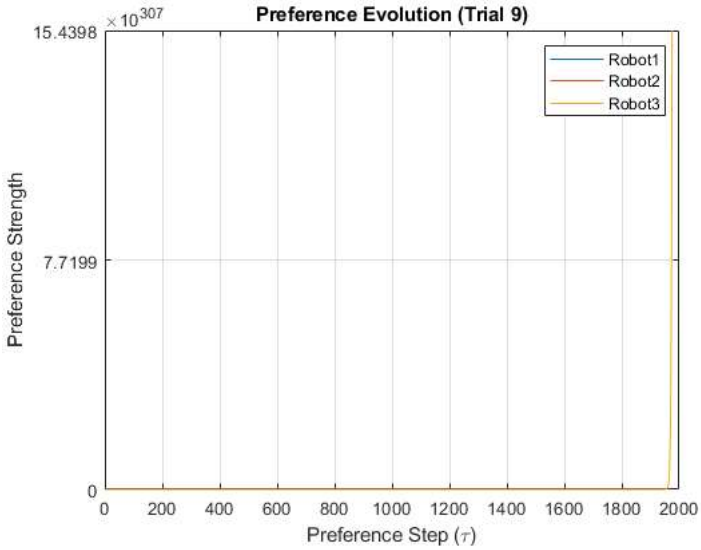

Preference Evolution (Trial 4)
×10^307
15.4266
7.7133
0
Preference Strength
0 200 400 600 800 1000 1200 1400 1600 1800 2000
Preference Step (τ)
Robot1
Robot2
Robot3

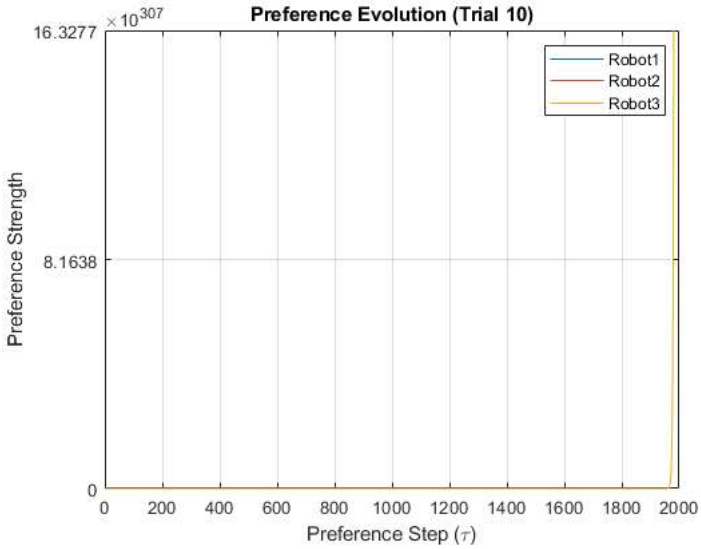

Preference Evolution (Trial 5)
×10^307
0
-5
-10
Preference Strength
0 200 400 600 800 1000 1200 1400 1600 1800 2000
Preference Step (τ)
Robot1
Robot2
Robot3


Preference Evolution (Trial 6)
×10^307
16.9094
8.4547
0
Preference Strength
0 200 400 600 800 1000 1200 1400 1600 1800 2000
Preference Step (τ)
Robot1
Robot2
Robot3


Preference Evolution (Trial 7)
×10^307
0
-5
-10
Preference Strength
Robot1
Robot2
Robot3
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Preference Step (τ)


Preference Evolution (Trial 8)
×10^307
10
5
0
Preference Strength
Robot1
Robot2
Robot3
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Preference Step (τ)





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: 1.3524
phi2: -0.1444
tau: 22027.4658
error_sd: 1
Initial Preferences (from ASCs):
    0.0804    0.1052         0
```

Published with MATLAB® R2022b