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 - 20Split.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: 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              0.70849                        0.36799                         0.41135                         0.070849
    Robot2              0.65283                        0.37784                         0.34028                         0.065283
    Robot3              0.60188                        0.34197                          0.3201                         0.060188

DFT Results:
E_P: -101.70     6.15    95.73
Choice probabilities: 0.000  0.000  1.000
Predicted choice: Robot 3
Actual choice: Robot 1

✗ Prediction differs from actual choice
=== Trial Analysis ===
```

Trial: 2
Participant: 125802
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	0.60907	0.35986	0.31012	0.060907
Robot2	0.67856	0.41958	0.32683	0.067856
Robot3	0.46398	0.17569	0.33468	0.046398

DFT Results:
E_P: -39.51 -159.62   199.32
Choice probabilities: 0.000 0.000 1.000
Predicted choice: Robot 3
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 3
Participant: 125802
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	0.70849	0.36799	0.41135	0.070849
Robot2	0.65283	0.37784	0.34028	0.065283
Robot3	0.60188	0.34197	0.3201	0.060188

DFT Results:
E_P: -101.70   6.15   95.73
Choice probabilities: 0.000 0.000 1.000
Predicted choice: Robot 3
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 4
Participant: 125802
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	0.89893	0.52464	0.46418	0.089893
Robot2	0.9514	0.5364	0.51013	0.09514
Robot3	1	0.63114	0.46886	0.1

DFT Results:
E_P: 89.49   -7.90 -81.40
Choice probabilities: 1.000 0.000 0.000
Predicted choice: Robot 1
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 5
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	0.75559	0.47366	0.35749	0.075559
Robot2	0.78853	0.51203	0.35536	0.078853
Robot3	0.72462	0.43603	0.36106	0.072462

DFT Results:
E_P: 1.28 -53.62   52.53
Choice probabilities: 0.000 0.000 1.000
Predicted choice: Robot 3
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 6
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	0.74509	0.41424	0.40536	0.074509
Robot2	0.7417	0.41084	0.40502	0.07417
Robot3	0.71779	0.41796	0.37162	0.071779

DFT Results:
E_P: -19.40 -13.59   33.18
Choice probabilities: 0.000 0.000 1.000
Predicted choice: Robot 3
Actual choice: Robot 3

✓ Prediction matches actual choice
=== Trial Analysis ===
Trial: 7
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	0.86123	0.53554	0.41181	0.086123
Robot2	0.79342	0.47712	0.39564	0.079342
Robot3	0.79187	0.48496	0.38609	0.079187

DFT Results:
E_P: -79.57   37.67   42.09
Choice probabilities: 0.000 0.012 0.988
Predicted choice: Robot 3
Actual choice: Robot 3

✓ Prediction matches actual choice
=== Trial Analysis ===
Trial: 8
Participant: 125802
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	0.58062	0.37908	0.2596	0.058062
Robot2	0.6076	0.3545	0.31386	0.06076
Robot3	0.69833	0.41538	0.35278	0.069833

DFT Results:
E_P: 91.01   34.79 -125.61
Choice probabilities: 1.000 0.000 0.000
Predicted choice: Robot 1
Actual choice: Robot 2

✗ Prediction differs from actual choice
=== Trial Analysis ===
Trial: 9
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	0.70612	0.35131	0.42541	0.070612
Robot2	0.88877	0.52466	0.45298	0.088877
Robot3	0.87669	0.49747	0.46689	0.08767

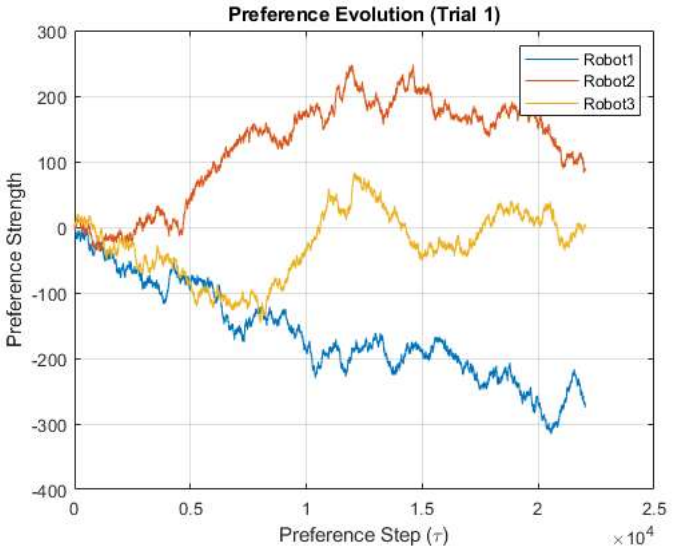
DFT Results:
E_P: 202.58 -109.89   -92.50
Choice probabilities: 1.000 0.000 0.000
Predicted choice: Robot 1
Actual choice: Robot 1

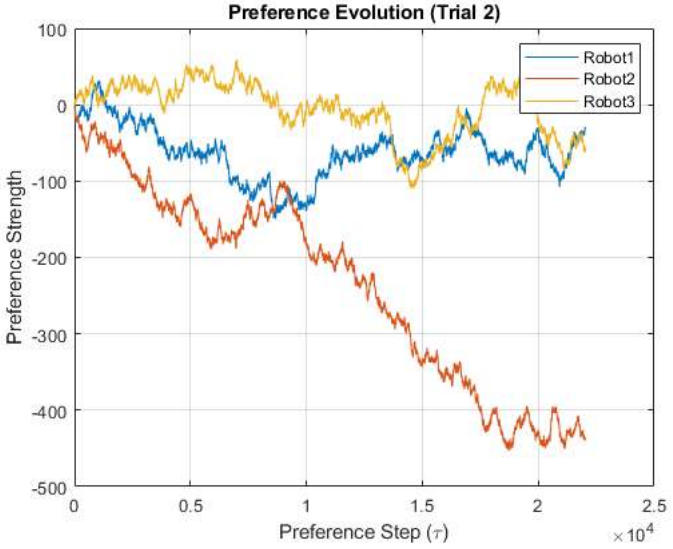
✓ Prediction matches actual choice
=== Trial Analysis ===
Trial: 10
Participant: 125802
Actual Choice: Robot 2
M matrix (alternatives × attributes):

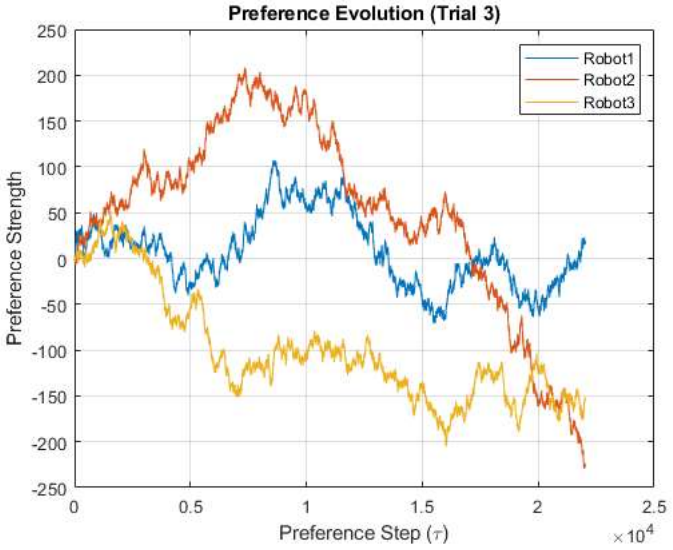
	C1 - Easy Nav, Low Exposure	C2 - Hard Nav, Low Exposure	C3 - Easy Nav, High Exposure	C4 - Hard Nav, High Exposure
Robot1	0.71446	0.42918	0.35673	0.071446
Robot2	0.56856	0.30467	0.32074	0.056856
Robot3	0.56828	0.27725	0.34785	0.056828

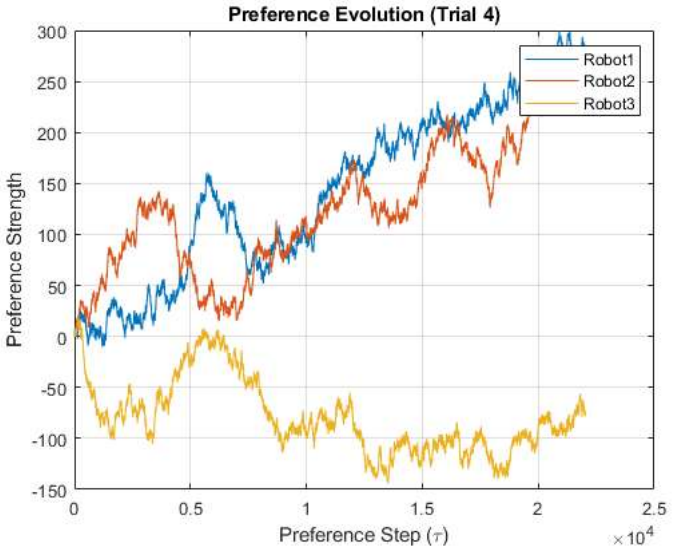
DFT Results:
E_P: -166.55   85.91   80.83
Choice probabilities: 0.000 0.994 0.006
Predicted choice: Robot 2
Actual choice: Robot 2

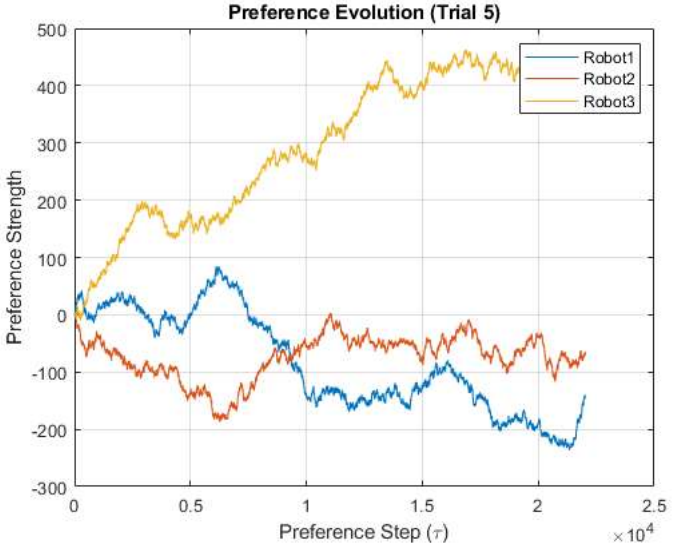
✓ Prediction matches actual choice

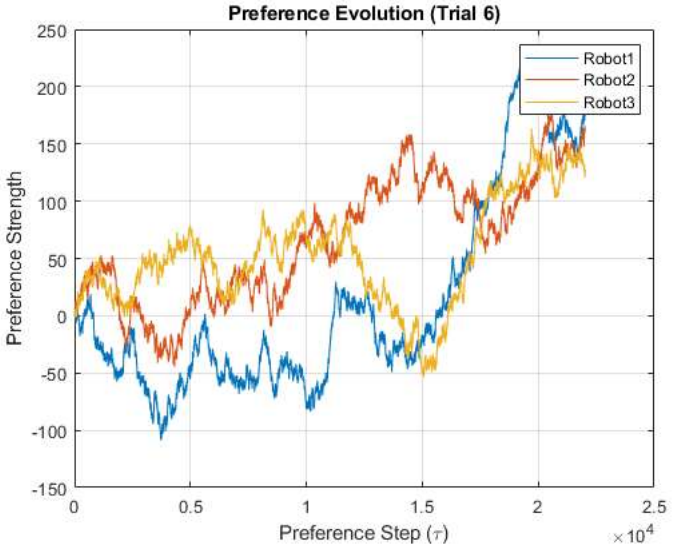

Preference Evolution (Trial 1)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$

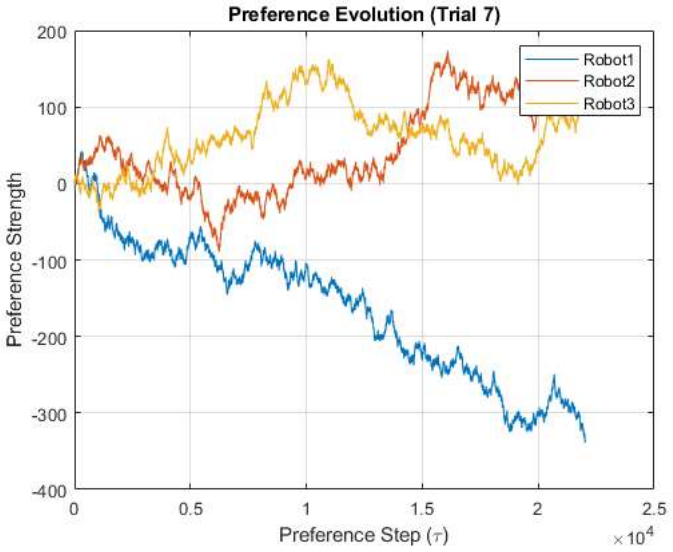

Preference Evolution (Trial 2)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$

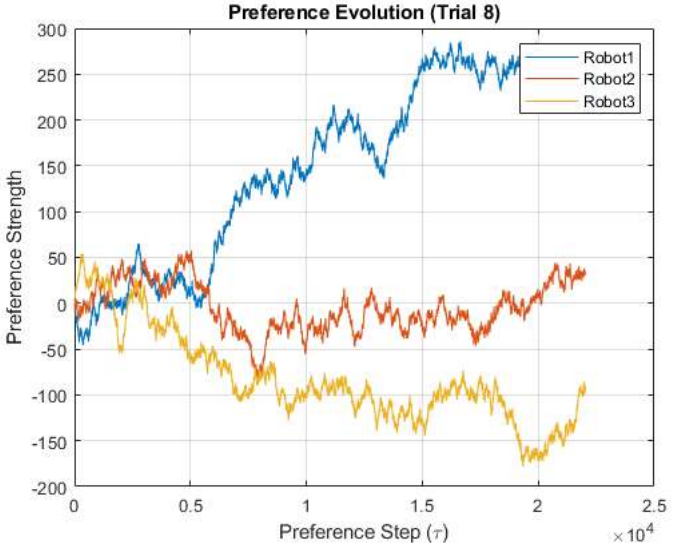

Preference Evolution (Trial 3)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$

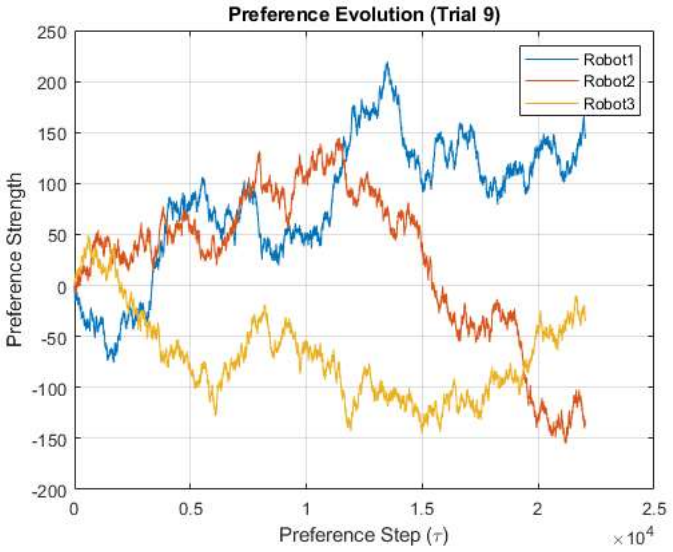

Preference Evolution (Trial 4)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$

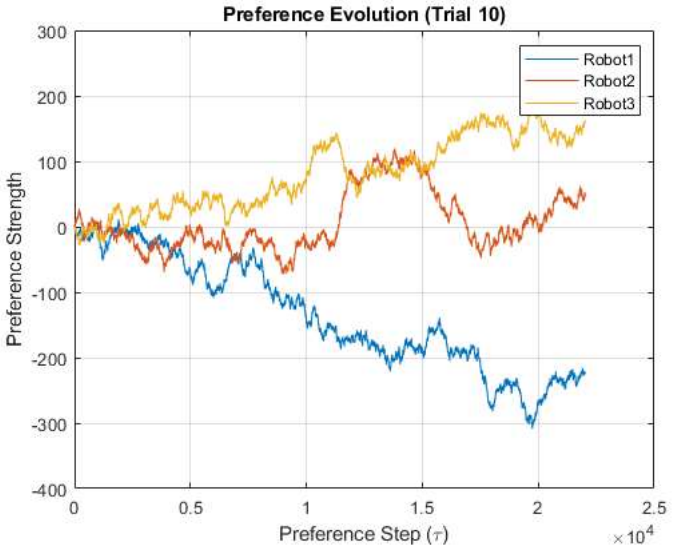

Preference Evolution (Trial 5)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$


Preference Evolution (Trial 6)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$


Preference Evolution (Trial 7)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$


Preference Evolution (Trial 8)
Robot1
Robot2
Robot3
Preference Strength
Preference Step (τ)
$\times 10^4$





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
tau: 22027.4658
error_sd: 1
Initial Preferences (from ASCs):
    0.0804    0.1052         0
```

Published with MATLAB® R2022b