BRMaximin Documentation

1. Purpose
    a. Identify behaviorally robust solutions to matrix games with varying forms of uncertainty by leveraging the Cognitive Hierarchy model to describe the behavior of boundedly rational adversaries.
2. Contents
    a. The toolbox contains 8 functions and multiple testing codes.
        i. Functions
            1. CogHierSol.m
                a. Identifies the CH solution associated with a given game and the estimated τ value.
                b. Requires inputs `payoffarray, tau, max_k`
                c. Outputs CHsolution
            2. CogHierExpM.m
                a. Identifies the Expected value an M-step thinker assigns with playing each action over all τ values in discrete uncertainty set.
                b. Requires inputs `payoffarray, tau_LB, tau_UB, tau_inc, max_k`
                c. Outputs `value,tau_rng`
            3. BRmaximin_R1.m
                a. Finds a behaviorally robust solution to a matrix game utilizing a discrete uncertainty set for τ.
                b. Requires inputs `payoffarray, U_tau, max_k, agent`
                c. Outputs `x` (BR solution)
            4. BRmaximin_S1.m
                a. Finds a behaviorally robust solution to a matrix game utilizing a discrete probability distribution over τ.
                b. Requires inputs `payoffarray, U_tau, Dist, max_k, agent`
                c. Outputs `x` (BR solution)
            5. BRmaximin_DR1.m
                a. Finds a behaviorally robust solution to a matrix game utilizing an ambiguity set of discrete probability distributions over τ.
                b. Requires inputs `payoffarray, U_tau, c1,c2,c3,c4, max_k, agent`
                c. Outputs `x` (BR solution)
            6. BRmaximin_R2.m
                a. Identifies the Expected value an M-step thinker assigns with playing each action over all τ values in continuous uncertainty set. Discretizes the interval into mesh for numerical calculation.
                b. Requires inputs `payoffarray, tau_LB, tau_UB, tau_inc, max_k, agent`
                c. Outputs `x` (BR solution)

7. BRmaximin_S2.m
   a. Finds a behaviorally robust solution to a matrix game utilizing some beta probability distribution over τ. Discretizes the interval into mesh for numerical calculation.
   b. Requires inputs `payoffarray, tau_LB, tau_UB, tau_inc, beta_a, beta_b, max_k, agent`
   c. Outputs `x` (BR solution)
8. BRmaximin_DR2.m
   a. Finds a behaviorally robust solution to a matrix game utilizing an ambiguity set of continuous probability distributions over τ. Discretizes the interval into mesh for numerical calculation.
   b. Requires inputs `payoffarray, tau_LB, tau_UB, tau_inc, c1,c2,c3,c4, max_k, agent`
   c. Outputs `x` (BR solution)

ii. Testing Codes
1. StahlandWilsoCheckCH.m
   a. Code to check the CogHierSol code against Camerer (2004) and Stahl and Wilson results.
2. CheckCHSolCode.m
   a. Code to check CogHierSol with tables provided in Camerer (2004).
3. CheckCHExpM.m
   a. Code illustrating how to use this function and displaying behavior over games in Camerer (2004).

3. Input Variables
   a. Payoffarray
      i. This is the payoff matrix for the normal form game. It should be inputted as follows:

         payoffarray(# of player receiving payoff, player 1 action, …., play n action)

         For example, "Matching Pennies" is represented as

```
payoffarray(1,1,1)=1
payoffarray(1,1,2)=0
payoffarray(1,2,1)=0
payoffarray(1,2,2)=1
payoffarray(2,1,1)=0
payoffarray(2,1,2)=1
payoffarray(2,2,1)=1
payoffarray(2,2,2)=0
```

   b. tau_LB: Scalar value representing lower bound of uncertainty set. Often this is 0.
   c. tau_UB: Scalar value representing upper bound of uncertainty set.
   d. tau_inc: Scalar grid spacing in the approximation of the continuous interval.

   e. c1: Scalar, mean of distributions in ambiguity set (c1>=0)
   f. c2: Scalar b/w 0 & 1, Density of probability distributions in ambiguity set b/w [c3,c4]
   g. c3: Lower bound of density requirement (c3>=0)
   h. c4: Upper bound of density requirement, Scalar b/w 0 and 1 (c4>c3)
   i. max_k: Scalar representing the max k-level thinking used in the algorithm
      i. This is also utilized as the big M value
   j. agent: Scalar, number of the player for whom we are optimizing their choice
   k. U_tau: Vector containing all values of τ being considered
   l. Dist: Vector, discrete probability distribution over U_tau. Elements correspond to once another (e.g., p(U_tau(1)) = Dist(1)). Vector values must sum to 1.
4. Outputs
   a. Each output of x has a slightly different form depending on the underlying needs of the optimization formulation.
   b. BRmaximin_R1.m and BRmaximin_R2.m
      i. x = [prob action 1, …., probaction n, maximin value]
   c. BRmaximin_S1.m and BRmaximin_S2.m
      i. x = [prob action 1, …., probaction n]
   d. BRmaximin_DR1.m and BRmaximin_DR2.m
      i. x = [dual variable 1, …, dual variable 3, prob action 1, …., probaction n]