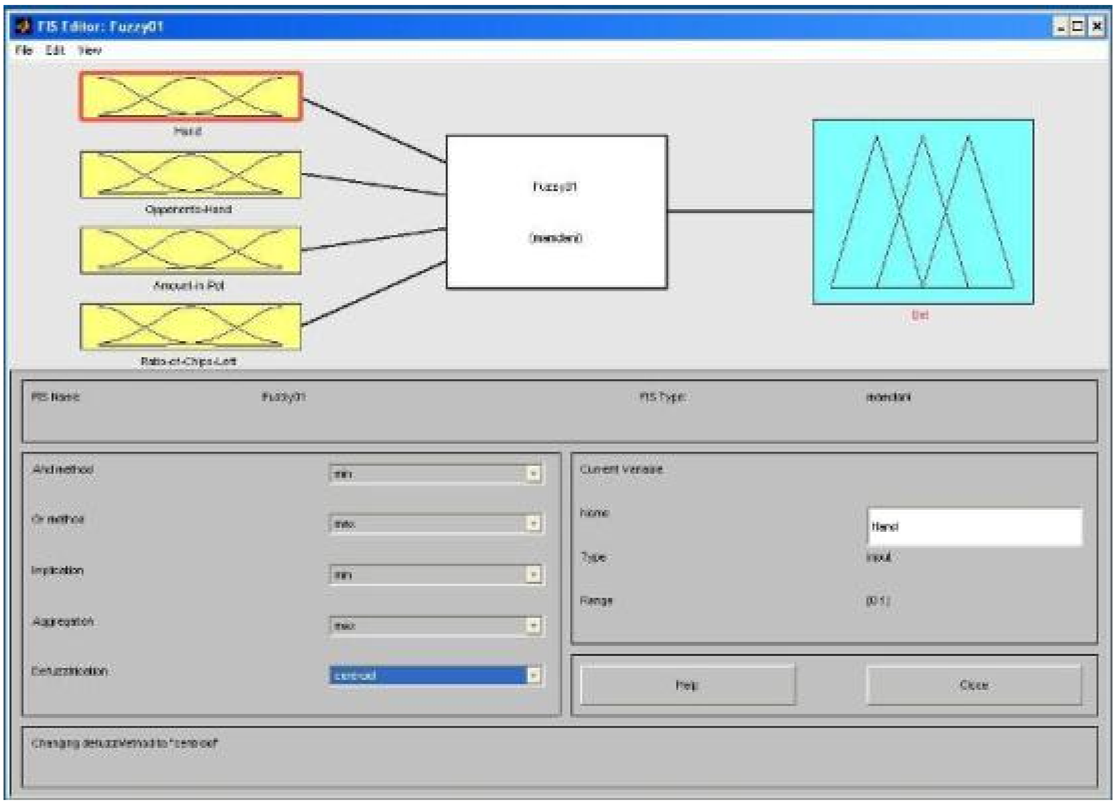
Fuzzy Logic Project Report

Report: Cathal Ormond (07617520)
Group: Mark Christiansen, Amy Davidson and Cathal Ormond

February 1, 2011

Introduction: Overview

The project we decided to do was to create a fuzzy logic system which would model a poker bet. The system takes in four inputs and gives an output which determines the player's move. Using this repeatedly, our system was designed to "play" a game of poker. Initially, it was convenient to assume that there was only one opponent, although it is not too hard to generalise this to multiple opponents. We chose this because there are several fuzzy input which one must consider when considering how to play a hand: what your hand is like, what your opponent bet last hand, whether you believe your opponent is bluffing or not.



FIS Name: Fuzzy01
FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: Hand
Type: input
Range: [0 1]

Changing defuzzMethod to "centroid"

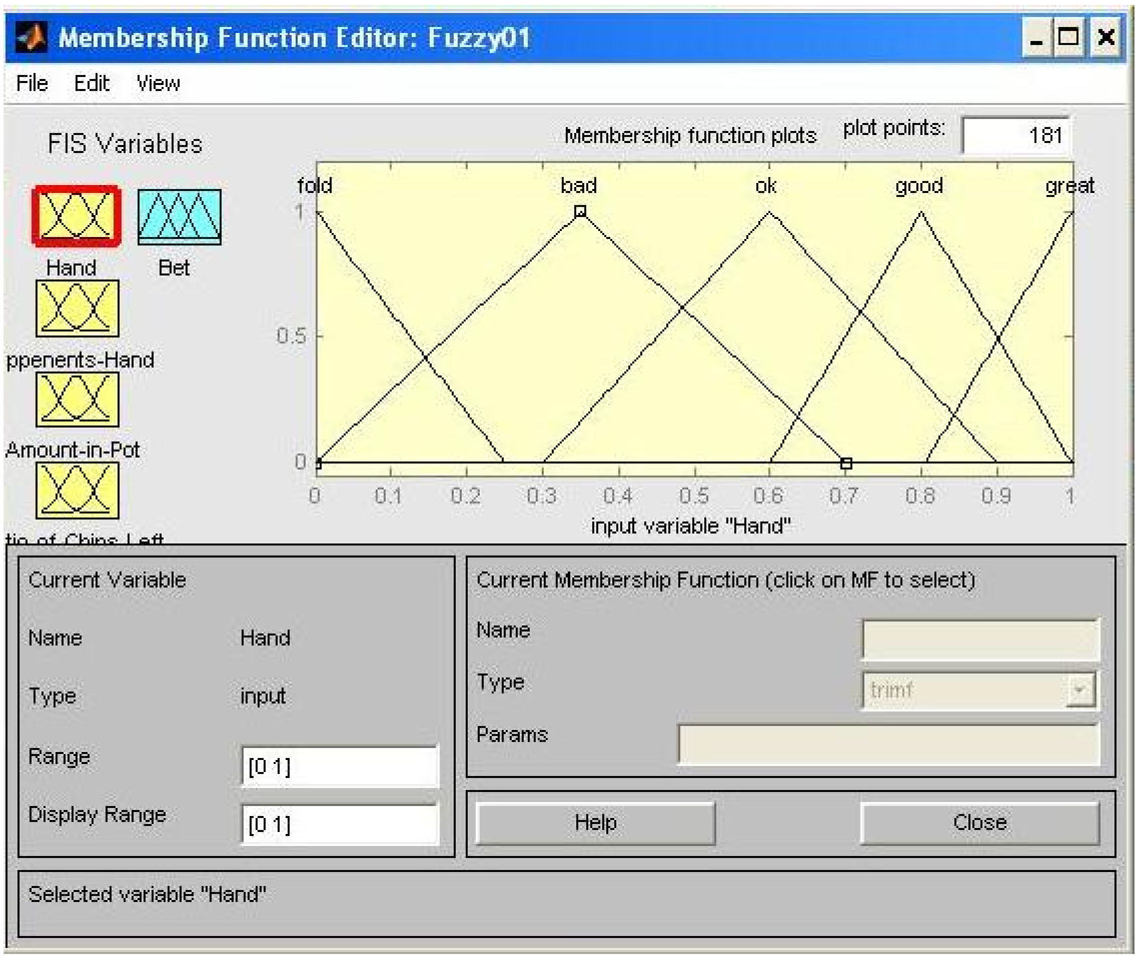
Table 1: Our Fuzzy System

Inputs

We decided on four inputs (linguistic variables) for our fuzzy system: Our Hand, Opponents Hand, The Amount in the Pot and The Ratio of Chips Left.

Our Hand

How good your hand is is given by a value in $[0, 1]$. In practise, this could not be strictly defined, as every poker player has their own idea of how good a particular hand is. When simulating, Mark (being the best poker player) determined how good the hands were. Our terms for this variable were: Fold, Bad, OK, Good, Great. As can been seen from the diagram, our definition of a good hand is quite conservative.



plot points: 181

input variable "Hand"

Current Variable

Name: Hand

Type: input

Range: [0 1]

Display Range: [0 1]

Current Membership Function (click on MF to select)

Name:

Type: trimf

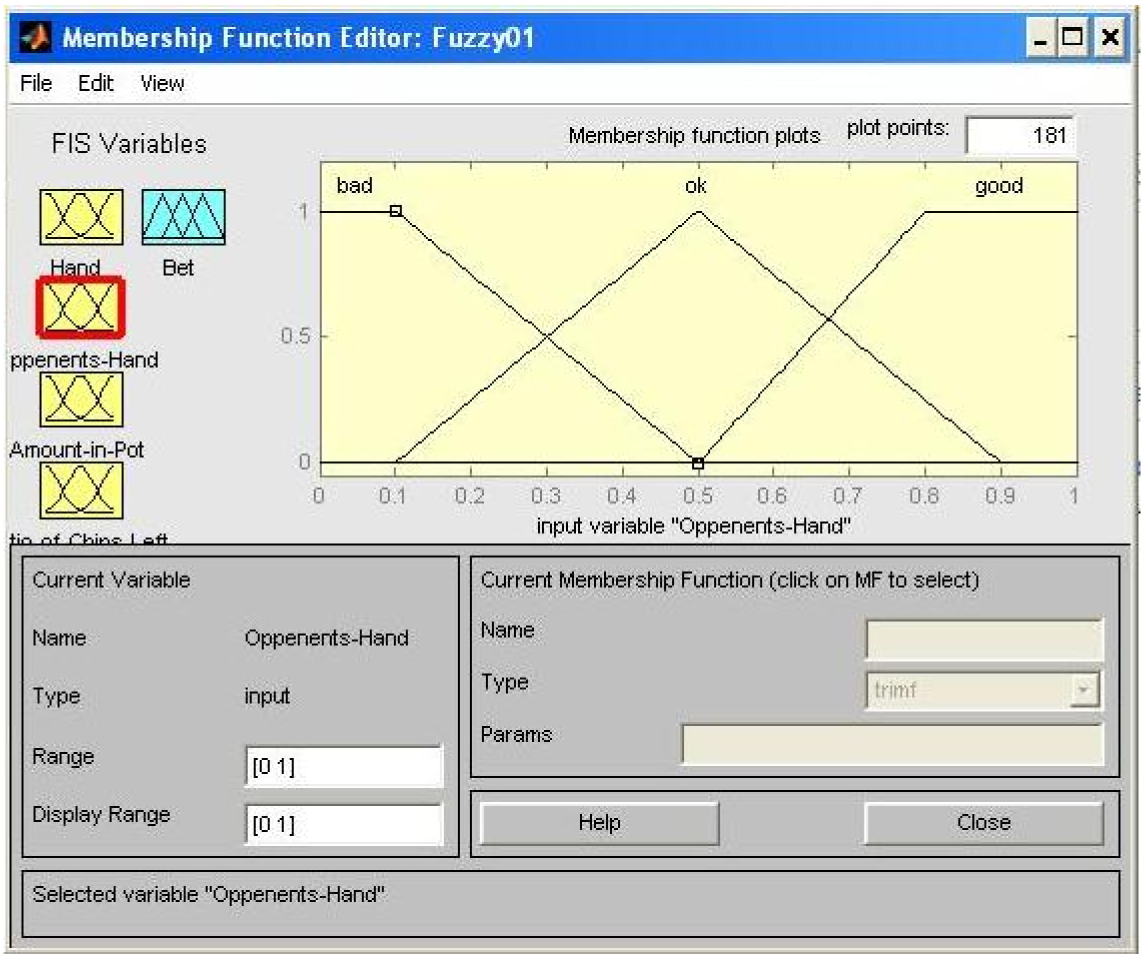
Params:

Selected variable "Hand"

Table 2: Our Hand Membership Function

Opponents Hand

Similar to the above, our opponents hand was also determined by a value in $[0, 1]$. The problem is that we do not know all the cards our opponent has, only between 3 and 5 of the cards that are out on the table. Hence, our estimate of how good our opponents hand is is slightly more vague than our estimate of our own hand. One can also take into account what the player has bet previously in determining how good our opponents hand is. Our terms for this variable were: Bad, OK, Good.



Current Variable

Name: Oppenents-Hand

Type: input

Range: [0 1]

Display Range: [0 1]

Current Membership Function (click on MF to select)

Name:

Type: trimf

Params:

Selected variable "Oppenents-Hand"

Table 3: Opponent's Hand Membership Function

Amount in Pot

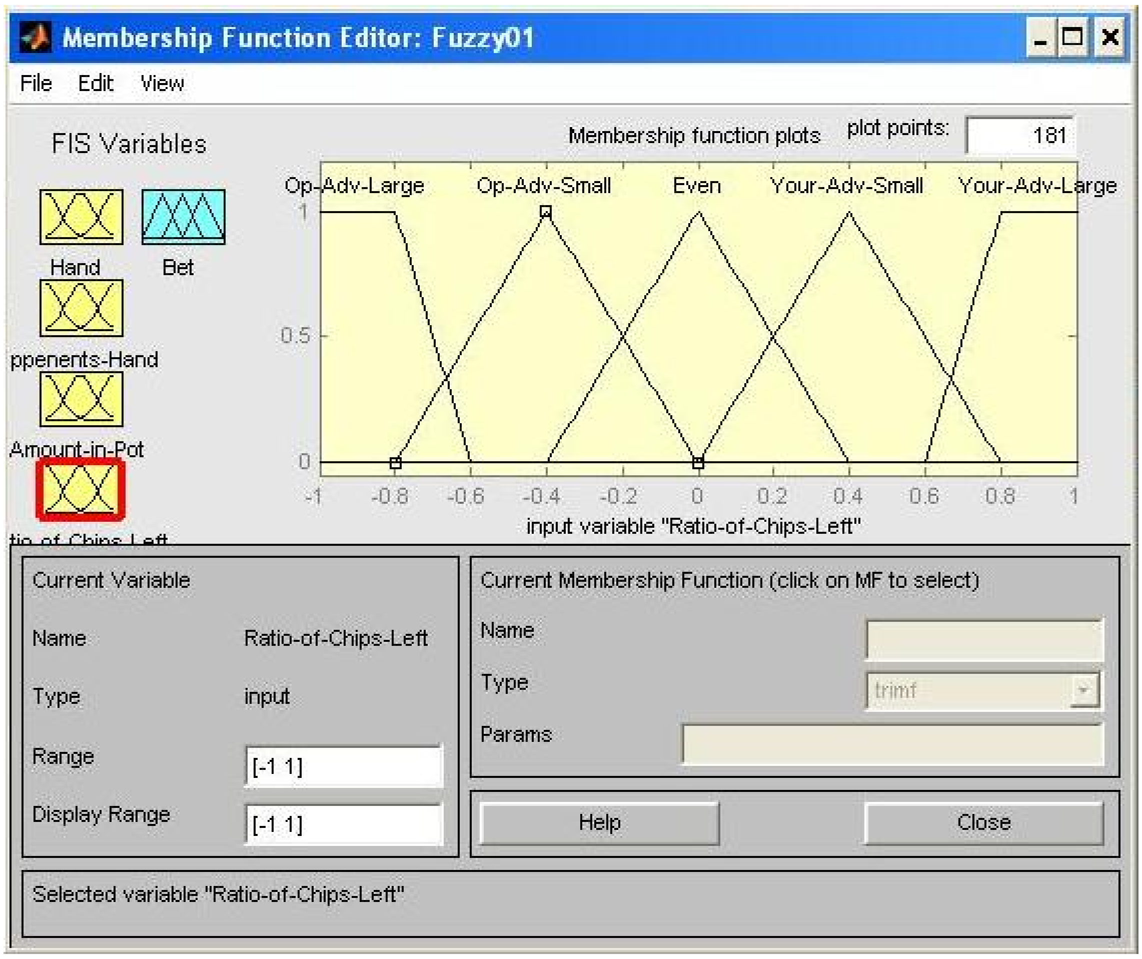
We initially assumed that there would be no more than 200 chips in the pot, and that each player would have 100 chips each. This can be modified easily enough to include more opponents, or a higher number of chips each. This value is not fuzzy, and lies in $[0, 200]$.

Chip Ratio

When placing a bet, it is often useful to determine if the chips remaining are to our advantage or to our opponents. If x is the number of chips we have left, and y is the number of chips our opponent has left, then a simple ration is given by

$$\text{Chip Ratio } = \frac{x-y}{x+y}$$

This value lies in $[-1, 1]$. Thus, if we have no chips left, i.e. $x = 0$, then the ratio is -1. If we have the same number of chips as our opponent, i.e. $x = y$, then the ratio is 0. Finally, if our opponent has no chips left, i.e. $y = 0$, then the ratio is 1. Our terms for this variable were: Opponent Advantage Large, Opponent Advantage Small, Even, Our Advantage Small, Our Advantage Large.



plot points: 181

Current Variable

Name	Ratio-of-Chips-Left
Type	input
Range	[-1 1]
Display Range	[-1 1]

Current Membership Function (click on MF to select)

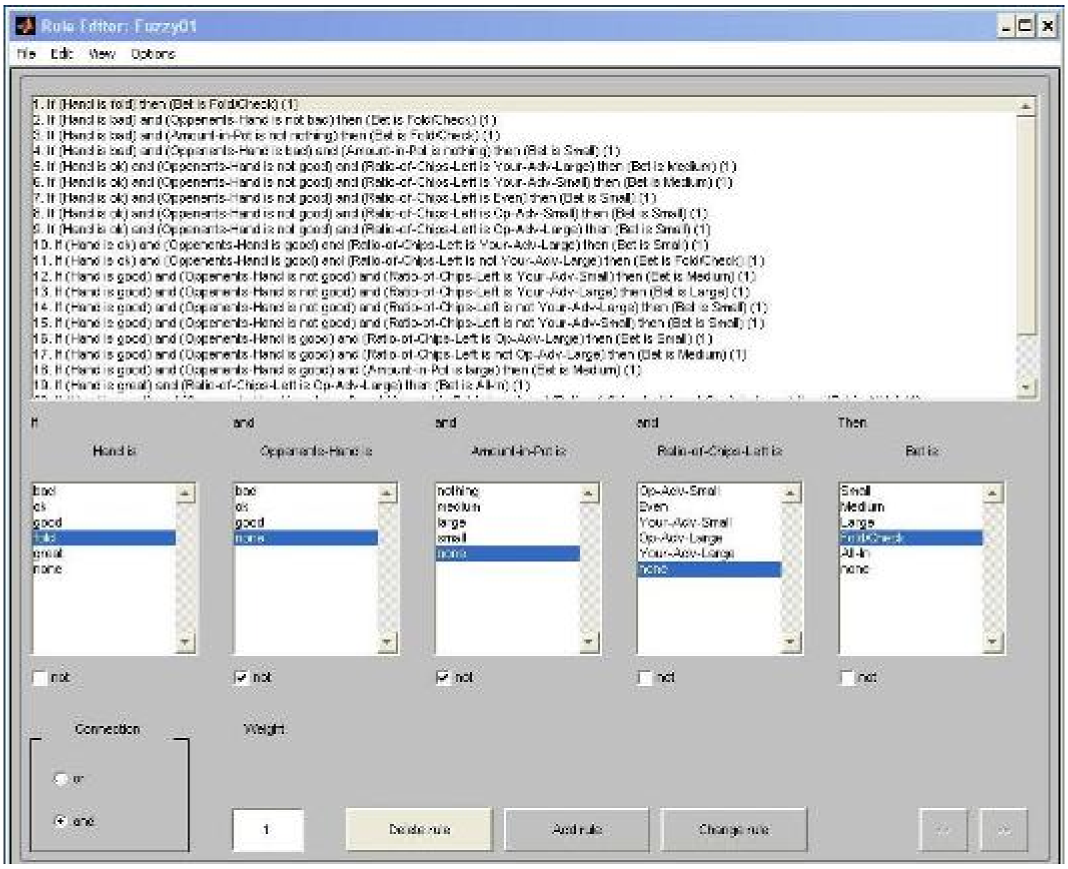
Name	
Type	trimf
Params	

Selected variable "Ratio-of-Chips-Left"

Table 4: Ratio of Chips Membership Function

Rules

For our section on rules, we used both common sense and strategy to determine what one should do with the situation presented. For example, if our hand is Bad, and our opponents hand is anything but Bad, then there is no point playing as there is little chance of winning. Thus, our output should be to fold. Similarly, say if our hand is Good, and our opponents hand is not Good, then we should indeed bet, but how much is determined by the chip ratio. If the chip ratio to your advantage and small, make a medium bet. If it is large to your advantage, bet large. Otherwise, bet small.



```
1. If (Hand is fold) then (Bet is Fold/Check) (1)
2. If (Hand is bad) and (Oppenents-Hand is not bad) then (Bet is Fold/Check) (1)
3. If (Hand is bad) and (Amount-in-Pot is not nothing) then (Bet is Fold/Check) (1)
4. If (Hand is bad) and (Oppenents-Hand is bad) and (Amount-in-Pot is nothing) then (Bet is Small) (1)
5. If (Hand is ok) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Your-Adv-Large) then (Bet is Medium) (1)
6. If (Hand is ok) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Your-Adv-Small) then (Bet is Medium) (1)
7. If (Hand is ok) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Even) then (Bet is Small) (1)
8. If (Hand is ok) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Op-Adv-Small) then (Bet is Small) (1)
9. If (Hand is ok) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Op-Adv-Large) then (Bet is Small) (1)
10. If (Hand is ok) and (Oppenents-Hand is good) and (Ratio-of-Chips-Left is Your-Adv-Large) then (Bet is Small) (1)
11. If (Hand is ok) and (Oppenents-Hand is good) and (Ratio-of-Chips-Left is not Your-Adv-Large) then (Bet is Fold/Check) (1)
12. If (Hand is good) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Your-Adv-Small) then (Bet is Medium) (1)
13. If (Hand is good) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is Your-Adv-Large) then (Bet is Large) (1)
14. If (Hand is good) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is not Your-Adv-Large) then (Bet is Small) (1)
15. If (Hand is good) and (Oppenents-Hand is not good) and (Ratio-of-Chips-Left is not Your-Adv-Small) then (Bet is Small) (1)
16. If (Hand is good) and (Oppenents-Hand is good) and (Ratio-of-Chips-Left is Op-Adv-Large) then (Bet is Small) (1)
17. If (Hand is good) and (Oppenents-Hand is good) and (Ratio-of-Chips-Left is not Op-Adv-Large) then (Bet is Medium) (1)
18. If (Hand is good) and (Oppenents-Hand is good) and (Amount-in-Pot is large) then (Bet is Medium) (1)
19. If (Hand is great) and (Ratio-of-Chips-Left is Op-Adv-Large) then (Bet is All-In) (1)
```

If	and	and	and	Then
Hand is	Oppenents-Hand is	Amount-in-Pot is	Ratio-of-Chips-Left is	Bet is
bad	bad	nothing	Op-Adv-Small	Small
ok	ok	medium	Even	Medium
good	good	large	Your-Adv-Small	Large
[✓ fold]	[✓ none]	small	Op-Adv-Large	[✓ Fold/Check]
great		[✓ none]	Your-Adv-Large	All-In
none			[✓ none]	none
☐ not	☑ not	☑ not	☐ not	☐ not

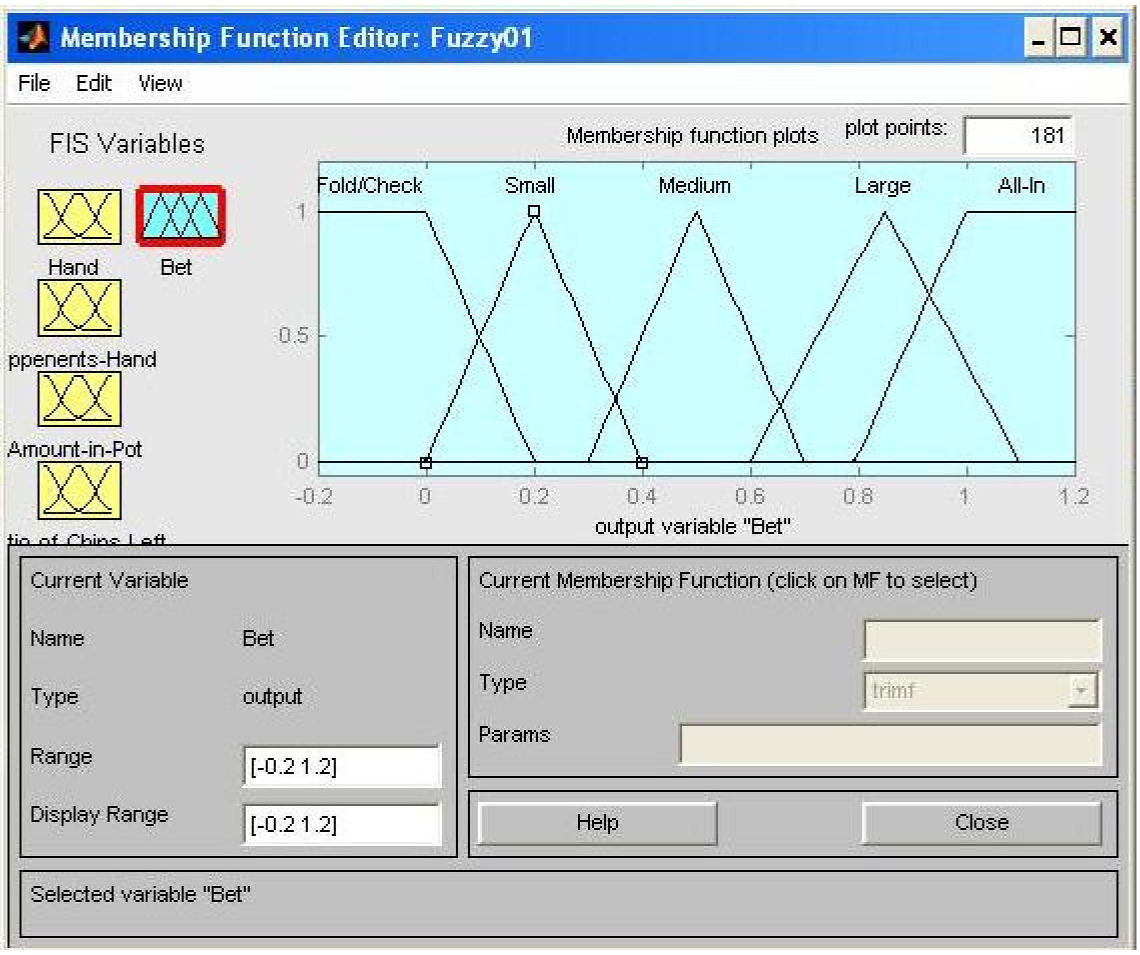
Connection: ○ or [✓ ● and]

Weight: 1

Table 5: Our rules

Output

In this project, we chose a Mamdani system. The output was designed not only to tell us what to do but, if betting, how much we should bet. Therefore, we had the following terms: Fold/Check, Small, Medium, Large, All in. Our output was originally designed to be in the range $[0, 1]$, but we allowed some small values outside these bounds, so in practise, the range was $[-0.2, 1.2]$. Anything below 0 means that we should not bet anything, so fold or check if possible. Anything above 1 means that we should go all in. Thus, the values between 0 and 1 represent - roughly - the ratio of chips that we have left that we should bet.



Current Variable

Name Bet

Type output

Range [-0.2 1.2]

Display Range [-0.2 1.2]

Current Membership Function (click on MF to select)

Name

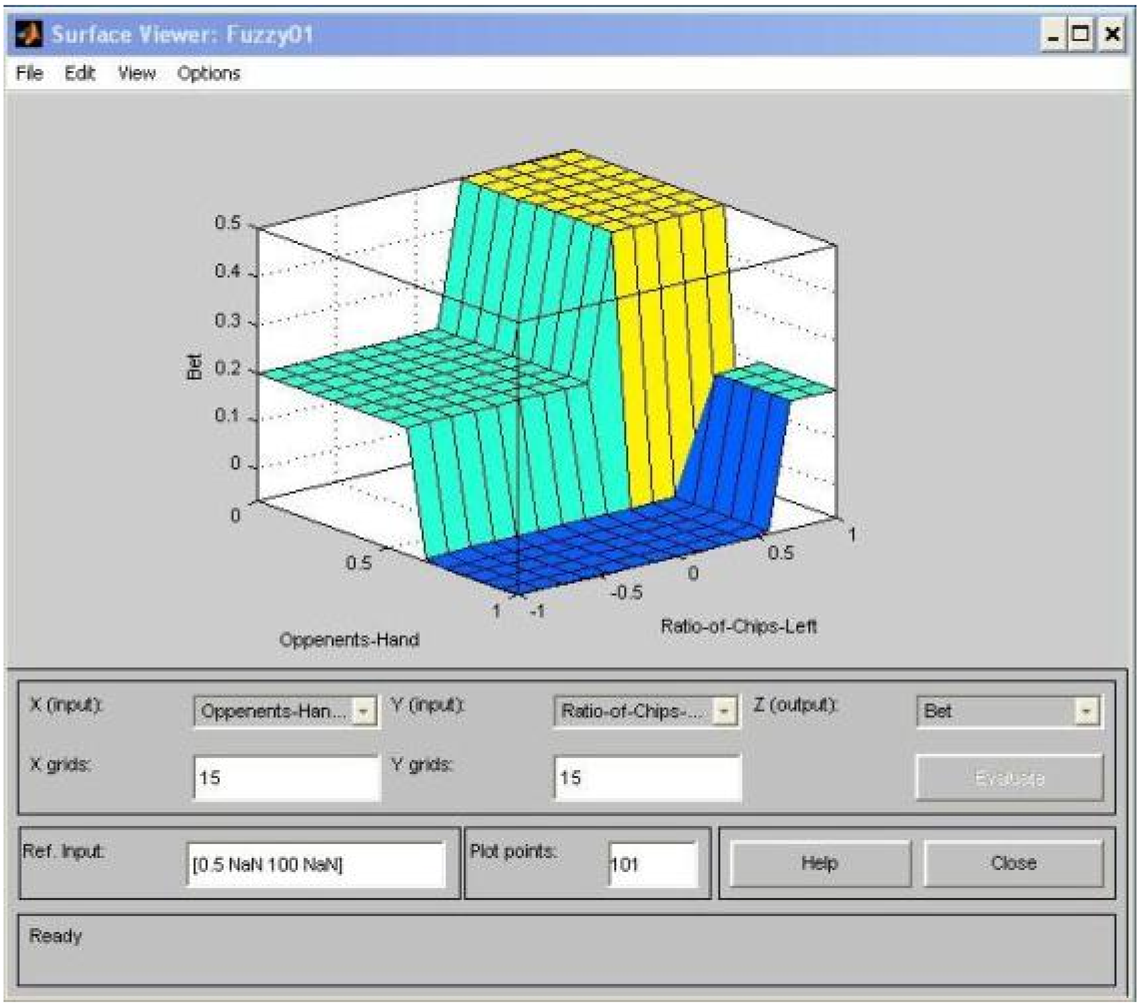
Type trimf

Params

Selected variable "Bet"

Table 6: Our Output

We considered the output based on our opponents hand versus the chip ratio. As you can see in the picture below, the worse your opponents hand is, the higher the bet. Also, once the chip ratio is even, and you opponents hand is anything but great, the bet increases.



X (input): Oppenents-Han... Y (input): Ratio-of-Chips-... Z (output): Bet
X grids: 15 Y grids: 15
Ref. Input: [0.5 NaN 100 NaN] Plot points: 101
Ready

Table 7: Input based on Opponents Hand and Chip Ratio

Stages

There were four stages in our Mamdani system: fuzzification, inference, composition and defuzzification.

Fuzzification

We took in crisp values for our input and determined by the membership functions which terms they applied to. For example, the following data:

$$\text{Our Hand } = 0.55, \text{ Opponents Hand } = 0.4, \text{ Amount in Pot } = 20, \text{ Chip Ratio } = 0.3$$

would mean that our hand is Good, our opponent's hand is OK, there are 20 chips in the pot, and we have a small chip advantage.

Inference

When our rules fire, we use inference. In the above example, we have a rule stating that when our hand is good, and our opponents hand is not good (i.e. ok) with the ratio of chips is small to our advantage that we should bet medium. In this case, the amount in the pot doesn't actually affect the output.

Composition

When more than one rule fires, we may not have a readily available output for our system. We use composition to combine these rules into one fuzzy set.

Defuzzification

Finally, we can defuzzify our system through our inputs to come up with a single output for our system.

Simulation

We tested our system a couple of times: firstly to determine if we were on the right track with our membership functions and then to determine if our system worked. At first, we considered a professional game of poker online. We chose a particular player and before he made his bet, we estimated the values for our input and had our fuzzy system determine what the players bet should be. We found some flaws with our values and that helped us to re-evaluate our rules.

Once we thought our system was accurate enough, we all played a game of poker with Mark recording his hand, what he felt our hands were, and what the chip ratio was. We all started with 100 chips and took note of our bets. At the end, we inputted our data into our system to determine what it thought Mark should have done. In almost all cases, the betting made was consistent with our outputs.

Analysis of System

The main problem with our system was that we had no way of dealing with players bluffing, and the system does not have any method of recommending that we bluff. For example, in our first test, our player had a hand that we determined as fold-worthy and his opponent had a bad hand. Our system gave the output of folding, but our player ended up going all in and winning the hand.

Had we spent more time testing the data and adapting the rules, I feel our output would have been more accurate. Also, it would have been a good idea to try a game of poker where our bets were determined by our system's output, instead of checking afterwards if our output matched the bets made.

Conclusion

In the end, we were happy with the output of our system. Our inputs were both fuzzy and crisp, and we had both full and partial information. There were only a few small cases in which we bet too conservatively compared to the system, but to have changed these rules would have meant skewing the entire membership functions. The system was perhaps too conservative for professional poker players and often failed when bluffing was encountered, but for non-professional games of poker, the system worked wells.