# Documentation

*Aaron Palumbo*

*Monday, December 08, 2014*

```
library(ggplot2)
library(reshape2)
library(plyr)
```

Objective: Generate figures to help explain game theory analysis

## Assumptions and Terms

We want to take a look at a tennis match from the perspective of game theory. In order to do this we will make some simplifying assumptions.

1. The main factor in determining the winner of a point is where the serve is placed
2. There are two distinct scenarios that present two distinct games:

- Deuce-court
  - Server has choice of serving left or right
  - Receiver has choice of guessing left or right

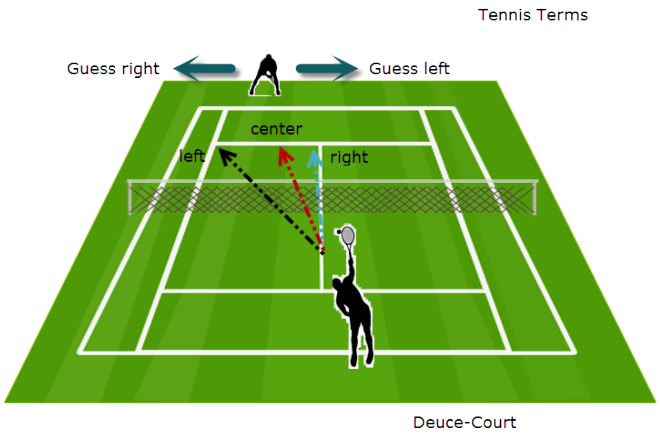



Figure 1

- Ad-court
  - Server has choice of serving left or right
  - Receiver has choice of guessing left or right

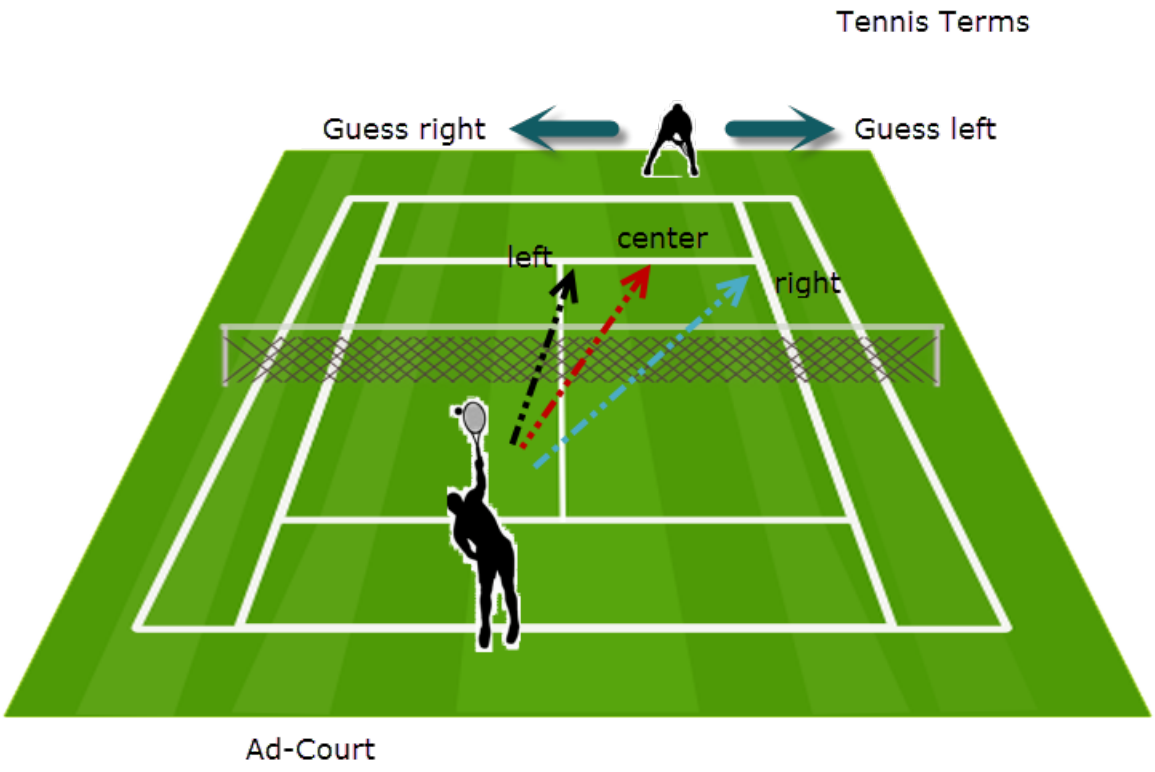



Figure 2

The option that is ignored is the center serve (from either court), since that rarely occurs.

For each game we would expect there to be an equilibrium point. For example, the deuce-court serve might look like this:

```
## pdf
##   2
```

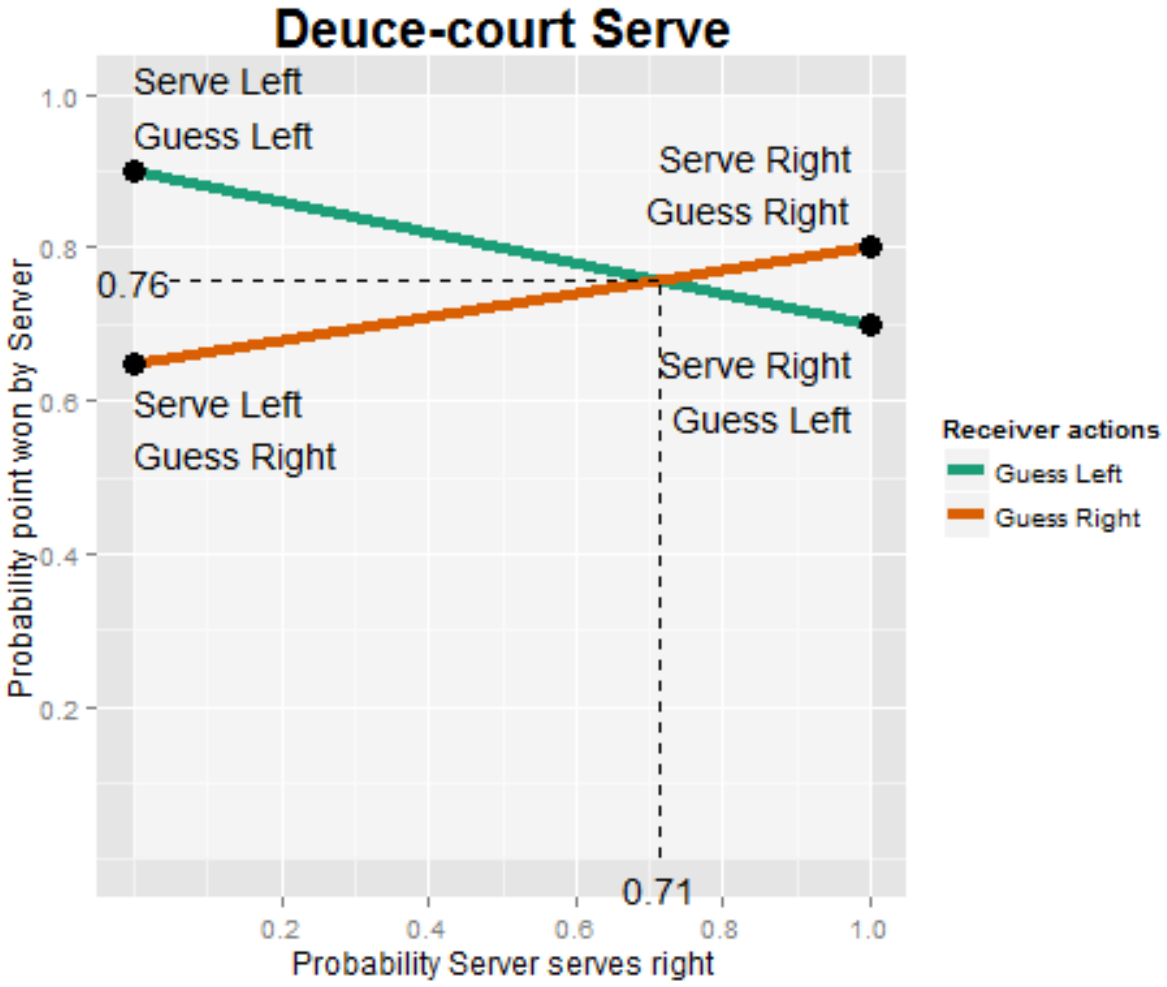



Figure 3. Shows assumed values for this game.

## Simulation

We will use simulation to explore the game space in an effort to determine how to evaluate whether or not a real tennis match is being played in equilibrium.

### Simulation function

Create a funtion to simulate results of a specified number of games.

We will first create a vector, serve, of serves modeled with a binomial distribution with parameters n and p where n = 1 (right=1, left=0) and p = {probability server serves right}, input at runtime.

We will also create a vector, receive, of receiver guesses modeled with a binomial distribution with parameters n and p where n = 1 (right=1, left=0) and p = {probability receiver guesses right}, also input at runtime. NOTE: right refers to a direction, not a correctness. See Figure 1.

We will assume the values given in Figure 3.

| Serve Direction | Receiver Guesses | Probability Server Wins Point |
|---|---|---|
| Left | Left | 0.9 |
| Left | Right | 0.65 |
| Rigth | Left | 0.7 |
| Right | Right | 0.8 |

Table 1

We can now break out the 4 combinations of the serve and receive vectors and generate 4 different vectors modeled as binomial sequences with parameters n and p where n = 1 (win=1, lose=0) and p is given in Table 1.

Putting everything back together we now have our simulation that consists of three vectors:

- serve (1=right, 0=left)
- receive (1=guess right, 0=guess left)
- outcome (1=server wins, 0=server loses)

```
simulate.points <- function(probs, sv_rt.act, rc_rt.act, n.pts){
  # ============================================================
  # Unroll probabilities    # Probability server wins when:
                            # |--------------|--------------------|
                            # | Server       | Receiver           |
                            # |--------------|--------------------|
  sv_lf.rc_lf <- probs[1]   # | Serves left  | Guesses left       |
  sv_lf.rc_rt <- probs[2]   # | Serves left  | Guesses right      |
  sv_rt.rc_lf <- probs[3]   # | Serves right | Guesses left       |
  sv_rt.rc_rt <- probs[4]   # | Serves right | Guesses right      |

  # sv_rt.act = Probability server serves right
  # rc_rt.act = Probability receiver guesses right
  # ** these do not have to equal equilibrium values

  # n.pts = number of points to simulate
  # ============================================================

  # Simulate serves and receive-guesses
  # Serve left = 0, Serve right = 1
  # Guess left = 0, Guess right = 1
  sv <- rbinom(n.pts, 1, sv_rt.act)
  rc <- rbinom(n.pts, 1, rc_rt.act)

  # Simulate winner of point based on given probabilities
  # sv.wins = 1, server wins
  # sv.wins = 0, server loses
  point.results <- data.frame(sv=sv, rc=rc, sv.wins=NA)

  # Create filters for the four combinations:
  f.sv_lf.rc_lf <- sv == 0 & rc == 0     # Serve left  - Guess left
  f.sv_lf.rc_rt <- sv == 0 & rc == 1     # Serve left  - Guess right
  f.sv_rt.rc_lf <- sv == 1 & rc == 0     # Serve right - Guess left
```

```
  f.sv_rt.rc_rt <- sv == 1 & rc == 1     # Serve right - Guess right

  # Simulate results:                              Number of points       Prob of winning
  point.results$sv.wins[f.sv_lf.rc_lf] <- rbinom(sum(f.sv_lf.rc_lf), 1, sv_lf.rc_lf)
  point.results$sv.wins[f.sv_lf.rc_rt] <- rbinom(sum(f.sv_lf.rc_rt), 1, sv_lf.rc_rt)
  point.results$sv.wins[f.sv_rt.rc_lf] <- rbinom(sum(f.sv_rt.rc_lf), 1, sv_rt.rc_lf)
  point.results$sv.wins[f.sv_rt.rc_rt] <- rbinom(sum(f.sv_rt.rc_rt), 1, sv_rt.rc_rt)

  return(point.results)
}
```

## Explore simulation results

### Server at Equilibrium, Receiver at Equilibrium, 300 points

Simulation Parameters:

Assumed values:

| Serve Direction | Receiver Guesses | Probability Server Wins Point |
|---|---|---|
| Left | Left | 0.9 |
| Left | Right | 0.65 |
| Rigth | Left | 0.7 |
| Right | Right | 0.8 |

Calculated from given values:

| Variable | Value at Equilibrium |
|---|---|
| Probability Server Wins | 0.76 |
| Probability Server serves right | 0.71 |
| Probability Receiver guesses right | 0.57 |

---

Simulation Inputs:

| Variable | Input Value |
|---|---|
| Probability Server serves right | 0.71 |
| Probability Receiver guesses right | 0.57 |
| Number of games to simulate | 300 |

---

Output Check (These should match the input variables):

| Variable | Output Value |
|---|---|
| Probability Server serves right | 0.73 |
| Probability Receiver guesses right | 0.56 |

| Variable | Output Value |
|---|---|
| Probability Server wins serving right | 0.79 |
| Probability Server wins serving left | 0.71 |
| **Overall probability Server wins** | **0.77** |

We can use the chi squared test to as a measure of how independent serve direction is from whether the point is won or lost. We set up the contingency table as follows:

| | Point Lost | Point Won |
|---|---|---|
| Serve Left | count(lose\|left) = 23 | count(win\|left) = 57 |
| Serve Right | count(lose\|right) = 46 | count(win\|right) = 174 |

If the serve direction is independent of the point outcome, we expect to se a low chi squared statistic and a high p-value:

| Pearson's Chi Squared Test | |
|---|---|
| – Chi Squared | 2.04 |
| – P-value | 0.15 |

**Server at Equilibrium, Receiver at 0.1 Right, 10000 points**

Simulation Parameters:

Assumed values:

| Serve Direction | Receiver Guesses | Probability Server Wins Point |
|---|---|---|
| Left | Left | 0.9 |
| Left | Right | 0.65 |
| Rigth | Left | 0.7 |
| Right | Right | 0.8 |

Calculated from given values:

| Variable | Value at Equilibrium |
|---|---|
| Probability Server Wins | 0.76 |
| Probability Server serves right | 0.71 |
| Probability Receiver guesses right | 0.57 |

Simulation Inputs:

| Variable | Input Value |
|---|---|
| Probability Server serves right | 0.71 |
| Probability Receiver guesses right | 0.1 |

| Variable | Input Value |
|---|---|
| Number of games to simulate | 10000 |

---

Output Check (These should match the input variables):

| Variable | Output Value |
|---|---|
| Probability Server serves right | 0.72 |
| Probability Receiver guesses right | 0.1 |

---

| Variable | Output Value |
|---|---|
| Probability Server wins serving right | 0.7 |
| Probability Server wins serving left | 0.88 |
| **Overall probability Server wins** | **0.75** |

We can use the chi squared test to as a measure of how independent serve direction is from whether the point is won or lost. We set up the contingency table as follows:

| | Point Lost | Point Won |
|---|---|---|
| Serve Left | count(lose\|left) = 326 | count(win\|left) = 2467 |
| Serve Right | count(lose\|right) = 2132 | count(win\|right) = 5075 |

If the serve direction is independent of the point outcome, we expect to se a low chi squared statistic and a high p-value:

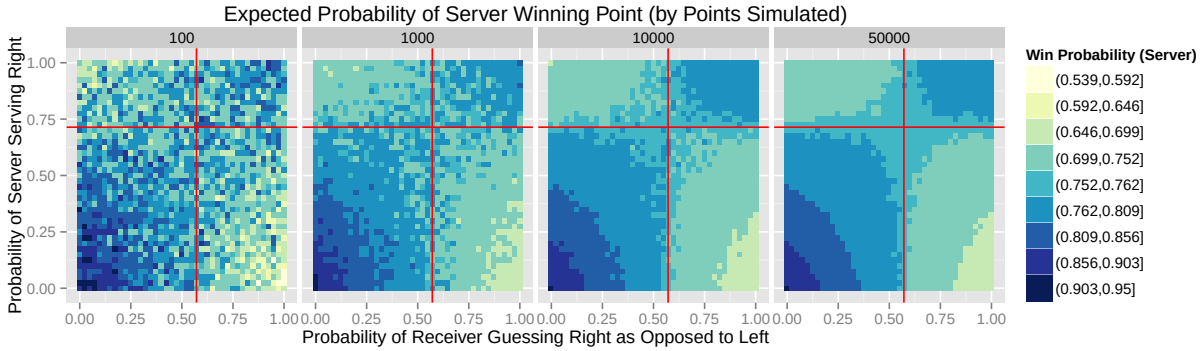
| Pearson's Chi Squared Test | |
|---|---|
| – Chi Squared | 348.31 |
| – P-value | 0 |

# Use simulation to look at game space

Now let's expand our view to all combinations of Server/Receiver play. To do this we will use a heat map to display the measure of interest on a grid where the x-axis represents the probability the Receiver guesses right and the y-axis represents the probability the Server serves right.

## Expected Value

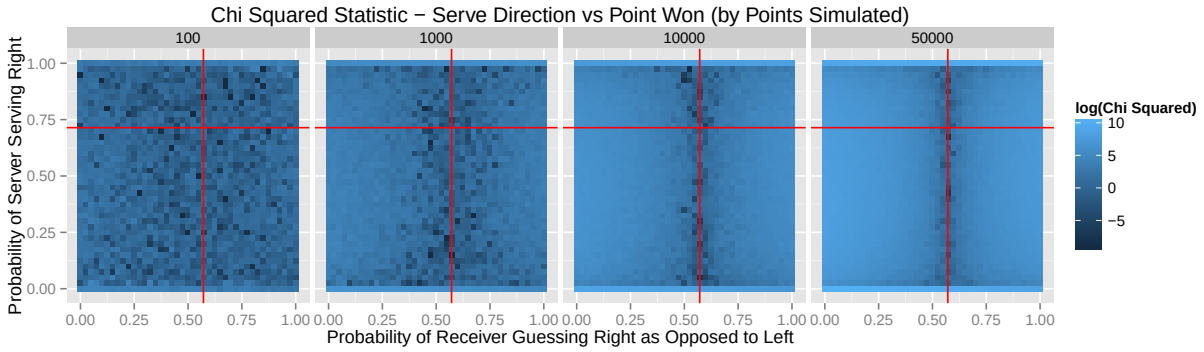
```
## Joining by:
## Joining by: sv.act, rc.act, ev, chi.sv, pv.sv, chi.rc, points, pv.rc
## Joining by: sv.act, rc.act, ev, chi.sv, pv.sv, chi.rc, points, pv.rc
## Joining by: sv.act, rc.act, ev, chi.sv, pv.sv, chi.rc, points, pv.rc
```




We can see that equilibrium play is achieved when the server serves 71% right and the receiver guesses 57% right. We can also see that we need to look at a lot of games to reduce noise.

Since we can't observe which way the receiver is guessing we need another measure that we *can* calculate, that will show us when the play is in equilibrium.

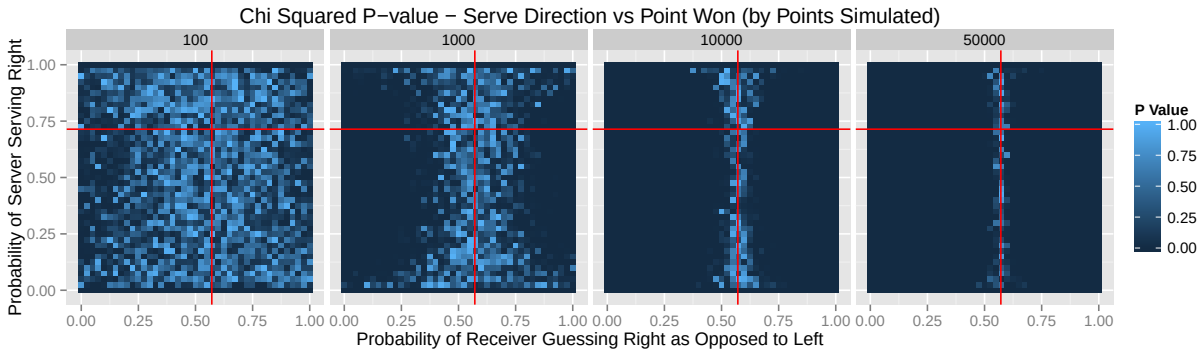
## Pearson Chi Squared Test, Chi Squared - Serve Direction vs Point Outcome



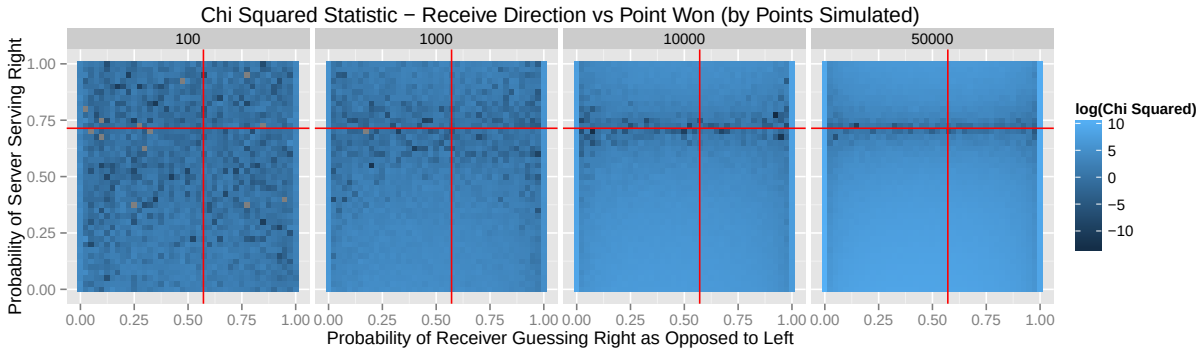

The Pearson Chi Squared Test is a measure of independence. We note that if the players are playing at equilibrium, then the proportion of time the server serves left and right should be independent of the number of points won to the left and to the right. This is convenient, since we can measure both quantities. Unfortunately, this is only true when the receiver is playing optimally, as shown in the above figure. So we can use this method to determine the optimal play of the receiver, but not the server. The other problem with this method is that it is subject to a high number of false positives when the number of points is low. Since a typical match is less than 300 points, it is rare to have enough data to make a good estimate.

## Pearson Chi Squared Test, P-value - Serve Direction vs Point Outcome

We can also look at the p-value of the chi squared test, which should correlate:

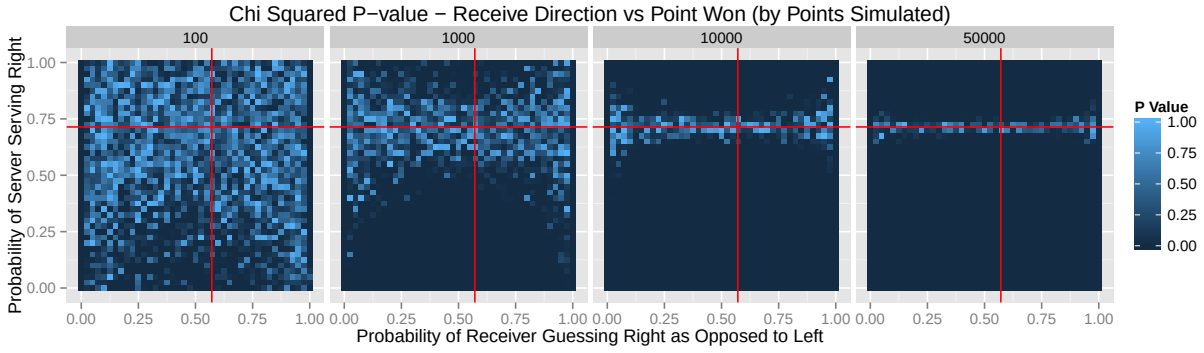



## Pearson Chi Squared Test, Chi Squared - Receive Direction vs Point Outcome




This figure shows a very similar test. In this case, however, we are measuring the independence between the receivers choice of left vs right and the outcome of the point. Unfortunately, we can not use this for real world data since there is no way to know the receivers choice. It is also subject to the same issues with noise.

## Pearson Chi Squared Test, P-value - Receive Direction vs Point Outcome




In this case, we are using the p-value from the test instead of the chi squared statistic, but it gives the same information.

# Conclusions

It's difficult to find good statistics on average points per match. From lots of searching we might guess an average of 60 points per set and a maximum of 5 sets which would give 300 - 400 points per match. Split that among the four game scenarios and we're around 100 points per scenario. If I'm counting correctly, the game we used has a total of 227 points. This is a long game so it seems reasonable than 300 is probably high. Then those 300-400 points get split among the four scenarios identified at the ouset. Looking at the noise in the chi squared statistic (many false positives) this does not seem to be an effective method of determining if equilibrium play was achieved.

Things for further exploration:

- Next steps that would be interesting to consider would be running a Bayesian analysis on the data to see what kind of error our measures have
- It might be possible to determine equilibrium play in two steps from observable data. One, find the receivers equilibrium value. Two, use that value to calculate the expected value. Finally, use the expected value to determine the server's equilibrium value.

# Appendix

## Validate Chi Squared Calculation

From paper: Minimax Play at Wimbledon By Mark Walker and John Wooders

Match 82: Wimbldn, Conners, McEnroe

Conners, Ad:
Serves: L - 32, R - 46, Total - 48
Points: L - 16, R - 32

Pearson statisitc = 3.052
p-value = 0.081

Let's make sure I can match this calculation:

| | Win | Lose | |
|---|---|---|---|
| Serve L | **16** | 16 | **32** |
| Serve R | 32 | 14 | **46** |
| | 48 | 30 | **78** |

```
values <- c(16, 16, 32, 14)
m <- matrix(values, nrow=2, byrow=TRUE)
chisq.test(m, correct=FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  m
## X-squared = 3.0522, df = 1, p-value = 0.08063
```

These values match the paper