Bridge

Adam Sperber

11/9/2021

Introduction:

I learned to play contract bridge in the winter of 2000, enthralled with a card game which had a phase outside of simply placing cards on the table strategically. The idea of communicating with your partner across the table about what cards you hold, what your preferred suit is, and how many tricks (collections of four cards) you can take through a system comprised entirely of numbers and suits (or lack thereof) astounded me. And after playing it for a while, I joined the ranks of card players throughout history who asked the question I've been asking for decades:

Why are my hands always awful?

So, I decided to take a stab at reducing the absurdly complex bidding system into code form (while allowing for human nature!) in hopes of finding answers to questions which I'm sure have already been proven. To summarize the game quickly, four players partition a standard deck of 52 cards equally and initially determine their strengths by the length and cardinality of each of their four suits. Cardinality is determined by a system known as high card points, where each ace is worth 4, king 3, queen 2, and jack 1. Players work to determine the amount of tricks taken out of 13 with either a defined trump suit or a lack thereof, known as "no trump."

The system of bidding is essentially a more complex version of the old game where players keep putting their hands up a baseball bat, each grab eliminating all the space underneath it. Bids span seven levels of contracts including all four suits and no trump, starting in the order of (1 club, 1 diamond, 1 heart, 1 spade, 1 no trump...) until it reaches the seventh level of those suits–the agreed upon level implies the partnership must take that level plus a "book" (collection of six additional tricks).

Game level is a term used to denote a specific minimum level of contract which help end a match: 3 no trump, 4 hearts, 4 spades, 5 clubs, and 5 diamonds. A subset of these contracts and above are known as slam (all 6- and 7-level contracts), which entail taking 12 or 13 tricks after bidding to said levels. The catch is while bids may also be used to describe your hand, they are all in fact specific contracts themselves, so once a bid is made, it and all bids below it may no longer be called. There are additional bids such as passing and doubling, but passing will only be used here in terms of having a very weak opening hand, and doubling does not factor in as it depends on opponents' bids.

The two questions I attempted to answer is how often does a hand have the strength to open a round of bidding, and how often does a partnership combine to reach game or slam level. Now, there are many intricacies which would arguably take months to perfect over code, so I decided to focus mostly on the textbook requirements to bid these contracts.

Methods

Just as a note, this report is meant to be read along with the code and hashed comments. That may be unconventional but it's the best way, I feel, to understand what's going on.

Given the perennial factoid that the odds of obtaining a universally unique order of 52 playing cards trends to 1 after several shuffles, the two major methods I needed to use were generating hands of cards through (presumably) the Mersenne Twister algorithm and simulating their power through the Monte Carlo method.

We need to create a deck of 52 cards, each with its own cardinality and suit. For simplicity, jacks are 11, queens are 12, kings are 13, and aces are 14. As stated, each face card/ace has a corresponding high card point value (1, 2, 3, and 4, respectively) which goes towards calculating a hand's point value.

Since the two questions I'm answering are restricted to one hand and its partnership, we will only be concerned with North and South as opposed to East and West. I used a simple ordered triple to generate hands: suits of four, cardinality of thirteen, and one last dimension for specific high card points.

After generating the deck, we split the cards into groups of thirteen, transmute into lists similarly to the Monopoly problem, and generate the hands to pass to bidding functions.

```
deal <- function(){
  # to deal four hands of 13 cards

  vals <- c(rep((2:14), 4))
  # creates a numeric equivalent from two to ace, low to high

  suit <- c(rep((1:4), each = 13))
  # creates the four suits, clubs being the lowest, then diamonds, hearts, and spades

  hcp <- rep(c(rep(0,9), 1:4), 4)
  # high card point count, jacks 1 queens 2 kings 3 aces 4

  deck <- data.frame(cbind(vals, suit, hcp))
  # merges all cards into a deck, each card having a value for cardinality, suit, and high card point c

  shuffle <- sample(1:52, 52)
  deck <- deck[shuffle,]
  # randomizes (shuffles) the deck

  south <- deck[1:13,]
  # west <- deck[14:26,]
  north <- deck[27:39,]
  # east <- deck[40:52,]
  # partitions the deck into four hands
  # only needed two

  south <- south[order(-south$suit, -south$vals),]
  # west <- west[order(-west$suit, -west$vals),]
  north <-north[order(-north$suit, -north$vals),]
  # east <- east[order(-east$suit, -east$vals),]
  # sorts the hands by suit in descending order

  SOUTH <- list()
  SOUTH$spades <- south$vals[south$suit == 4]
  SOUTH$hearts <- south$vals[south$suit == 3]
  SOUTH$diamonds <- south$vals[south$suit == 2]
  SOUTH$clubs <- south$vals[south$suit == 1]
  SOUTH$hcp <- sum(south$hcp)

  # I needed to make them lists for upcoming bidding functions

  NORTH <- list()
  NORTH$spades <- north$vals[north$suit == 4]
  NORTH$hearts <- north$vals[north$suit == 3]
```

```
  NORTH$diamonds <- north$vals[north$suit == 2]
  NORTH$clubs <- north$vals[north$suit == 1]
  NORTH$hcp <- sum(north$hcp)

  return(list(SOUTH = SOUTH, NORTH = NORTH))
  # generates hands
}
```

At this point, hands have been generated in the forms of transmuted ordered triples with an additional counter for high card points. Now we need to take those hands and determine the bids of which they're capable. That entails determining distributional points as well as high card points.

"Natural" opening bids are primarily at the 1-level, indicating points and either a suit of preference or a strong, balanced hand for no trump. Preemptive bids imply a weaker hand in terms of high card points but a very long and possibly strong single suit; such an occurrence means there's a higher chance that your opponents have stronger hands (more points to be evenly divided between the three remaining players) and you would want to take up their bidding space which could have been used to communicate.

Most of what follows is a massive if(){} jungle, but I created constants for each suit length and a vector combining them to represent suit distribution, a key component of bid requirements. In order to throw in a little chaos, I added a random feature generator which accounts for some players to inflate their hands; one out of 200 would increase their hand strength by 2, with 1 out of 20 increasing it by 1.

```
open <- function(hand = SOUTH) {
  S <- length(hand$spades)
  H <- length(hand$hearts)
  D <- length(hand$diamonds)
  C <- length(hand$club)
  # simplifies suit length

  distro <- c(S, H, D, C)
  # creates hand distribution vector

  points <- sum(hand$hcp)
  b <- sample(1:200, 1)
  if (b == 200) {
    points <- points + 2
  } else if (b > 189) {
    points <- points + 1
  }
  # this accounts for the tendency for players to bid somewhat aggressively

  if (min(distro) == 3) {
    points <- points - 1
  }
  # an unspoken rule is to devalue your hand by a point for 4-3-3-3 distribution due to lack of suit st

  for (i in 1:4) {
    if (distro[i] == 0) {
      points <- points + 4
    } else if (distro[i] == 1) {
      points <- points + 1
    }
  }
```

```
# a voided suit in a trump contract is very powerful, as it almost guarantees smaller trump cards to

if (points >= 22) {
  return("2C")
}
# an opening bid of two clubs signals your hand is widely regarded as the strongest opening bid as op

if (points >= 20 &
    min(distro) == 2 & max(distro) == 4) {
  return("2NT")
}
if (points > 14 &
    points < 18  & min(distro) == 2 & max(distro) == 4) {
  return("1NT")
}
if (points >= 12) {
  if (S >= 5 & max(distro) == S) {
    return("1S")
  }
  else if (H >= 5 & max(distro) == H) {
    return("1H")
  }
  else if (D >= C) {
    return("1D")
  }
  else {
    return("1C")
  }
}

# these are the "natural" opening bids which accurately describe your strength based on high card poi
# the following are preemptive bids, or inflated bids from low point-count hands based on suit length

if (points > 5) {
  if (length(hand$spades) >= 8 & hand$spades[1] >= 12) {
    return("4S")
  }
  if (length(hand$hearts) >= 8 &
      hand$hearts[1] >= 12) {
    return("4H")
  }
  if (length(hand$diamonds) >= 8 &
      hand$diamonds[1] >= 12) {
    return("4D")
  }
  if (length(hand$clubs) >= 8 &
      hand$clubs[1] >= 12) {
    return("4C")
  }

  if (length(hand$spades) == 7 &
      hand$spades[1] >= 12) {
    return("3S")
```

```
  }
  if (length(hand$spades) == 8 &
      hand$spades[1] >= 11) {
    return("3S")
  }
  if (length(hand$hearts) == 7 &
      hand$hearts[1] >= 12) {
    return("3H")
  }
  if (length(hand$hearts) == 8 &
      hand$hearts[1] >= 11) {
    return("3H")
  }
  if (length(hand$diamonds) == 7 &
      hand$diamonds[1] >= 12) {
    return("3D")
  }
  if (length(hand$diamonds) == 8 &
      hand$diamonds[1] >= 11) {
    return("3D")
  }
  if (length(hand$clubs) == 7 &
      hand$clubs[1] >= 12) {
    return("3C")
  }
  if (length(hand$clubs) == 8 &
      hand$clubs[1] >= 11) {
    return("3C")
  }

  if (length(hand$spades) == 6 &
      hand$spades[1] >= 12) {
    return("2S")
  }
  if (length(hand$spades) == 7 &
      hand$spades[1] >= 11) {
    return("2S")
  }
  if (length(hand$hearts) == 6 &
      hand$hearts[1] >= 12) {
    return("2H")
  }
  if (length(hand$hearts) == 7 &
      hand$hearts[1] >= 11) {
    return("2H")
  }
  if (length(hand$diamonds) == 6 &
      hand$diamonds[1] >= 12) {
    return("2D")
  }
  if (length(hand$diamonds) == 7 &
      hand$diamonds[1] >= 11) {
    return("2D")
```

```
    }
    # 2 clubs is reserved for artificial strength

  } else {
    return("PASS")
  }}
```

And that's it! Now we just run the hands a few thousand times and see what pops up.

```
counter <- list()
pointscounter <- c()
for (i in 1:100000) {
  hands <- deal()
  hands
  SOUTH <- hands$SOUTH
  counter[i] <- open(SOUTH)
}
```

For teams, it's nearly impossible to accurately render a method for reaching specific contracts since play depends heavily on random breaks in the opponents as well as their own playing nuances, so I focused primarily on when your partnership gets dealt strength. Here we introduce a concept of a golden fit, which is when a partnership controls 8/13 or more of a certain suit, often deemed necessary for game-level contracts and absolutely essential for suited slam contracts.

I used some of the code from my open() function and applied it to both teams in the partnership but had all hand distributional aspects contribute towards the team point total. To allow for suited contracts with the possibility of no trump, I set a preference for the length of the partnership's best suit to positively influence their point count if it existed, but not detract if the partnership had no such good fits. I didn't resort to specific game- and slam-level contracts when it came to outputting results, rather just sheer hand strength.

```
goodFit <- function(hand1 = SOUTH, hand2 = NORTH) {
  points <- sum(hand1$hcp, hand2$hcp)
  b <- sample(1:200, 1)
  if (b == 200) {
    points <- points + 2
  } else if (b > 189) {
    points <- points + 1
  }

  S1 <- length(hand1$spades)
  H1 <- length(hand1$hearts)
  D1 <- length(hand1$diamonds)
  C1 <- length(hand1$club)

  S2 <- length(hand2$spades)
  H2 <- length(hand2$hearts)
  D2 <- length(hand2$diamonds)
  C2 <- length(hand2$club)

  distro1 <- c(S1, H1, D1, C1)
  distro2 <- c(S2, H2, D2, C2)

  if (min(distro1) == 3) {
```

```
    points <- points - 1
  }

  if (min(distro2) == 3) {
    points <- points - 1
  }


  for (i in 1:4) {
    if (distro1[i] == 0) {
      points <- points + 4
    } else if (distro1[i] == 1) {
      points <- points + 1
    }
  }

  for (i in 1:4) {
    if (distro2[i] == 0) {
      points <- points + 4
    } else if (distro2[i] == 1) {
      points <- points + 1
    }
  }

  s <- length(hand1$spades) + length(hand2$spades)
  h <- length(hand1$hearts) + length(hand2$hearts)
  d <- length(hand1$diamonds) + length(hand2$diamonds)
  c <- length(hand1$clubs) + length(hand2$clubs)

  bestSuit <- max(c(s, h, d, c))

  # partnership suit totals
  # expected value is 6.5 between the two, but a game level contract requires 8+

  teamPoints <- points + max(c(bestSuit - 8), 0)

  # a golden fit implies a partnership has 8 of a single suit, enough for a game-level contract in it

  if (teamPoints < 26) {
    return("1. No game")
  } else if (teamPoints < 33) {
    return("2. Game")
  } else {
    return("3. Slam")
  }
}
```

Let's run this mother.

```
counter3 <- list()
for (i in 1:100000) {
  hands <- deal()
  hands
```

```
  SOUTH <- hands$SOUTH
  NORTH <- hands$NORTH
  counter3[i] <- goodFit(SOUTH, NORTH)
}
```

Results

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

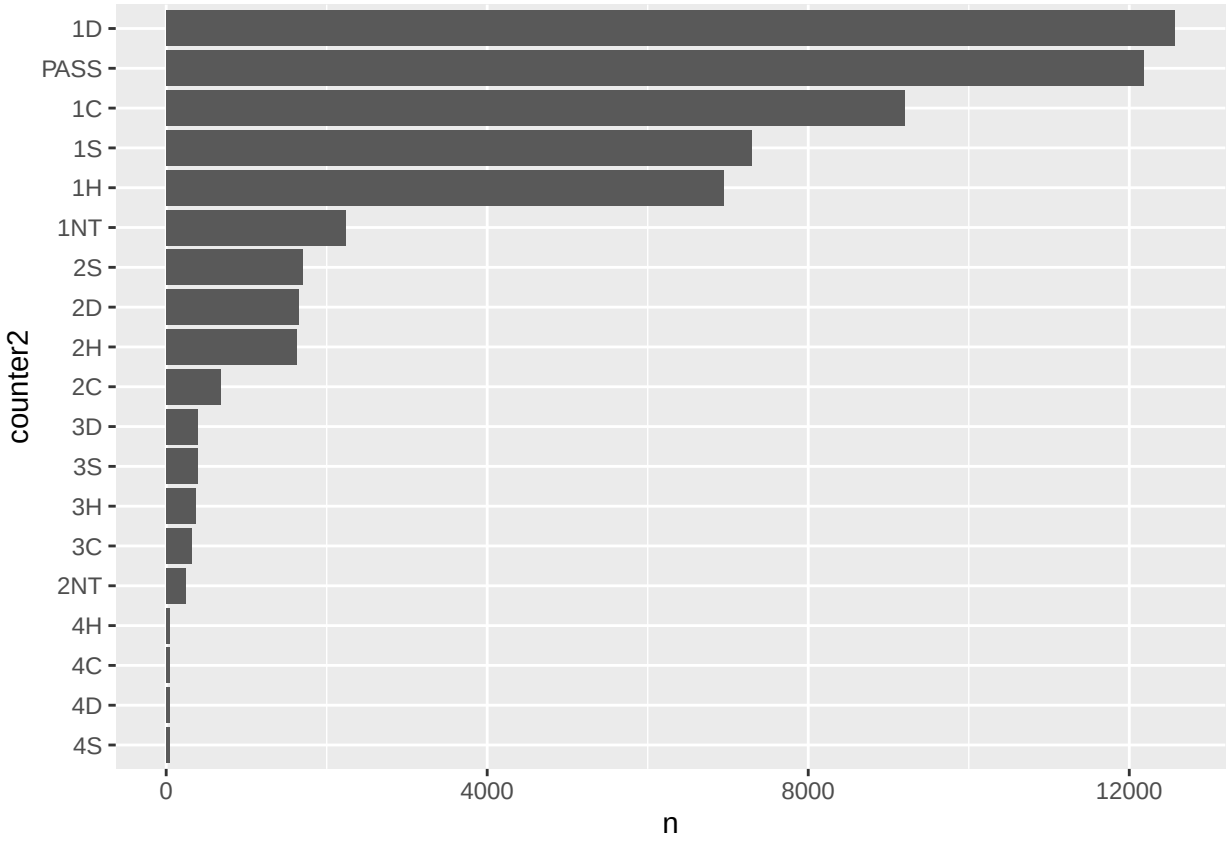
```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'stringr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
counter2 <- unlist(counter, use.names = FALSE)
counter2 <- data.frame(counter2)
counter2 %>%
  count(counter2) %>%
  mutate(counter2 = fct_reorder(counter2, n)) %>%
  ggplot(aes(y = counter2, x = n)) + geom_col()
```




```
table(counter2)
```

```
## counter2
##    1C    1D    1H   1NT    1S    2C    2D    2H   2NT    2S    3C    3D    3H
##  9206 12569  6944  2242  7292   681  1647  1629   248  1705   319   395   366
##    3S    4C    4D    4H    4S  PASS
##   387    45    44    50    40 12180
```

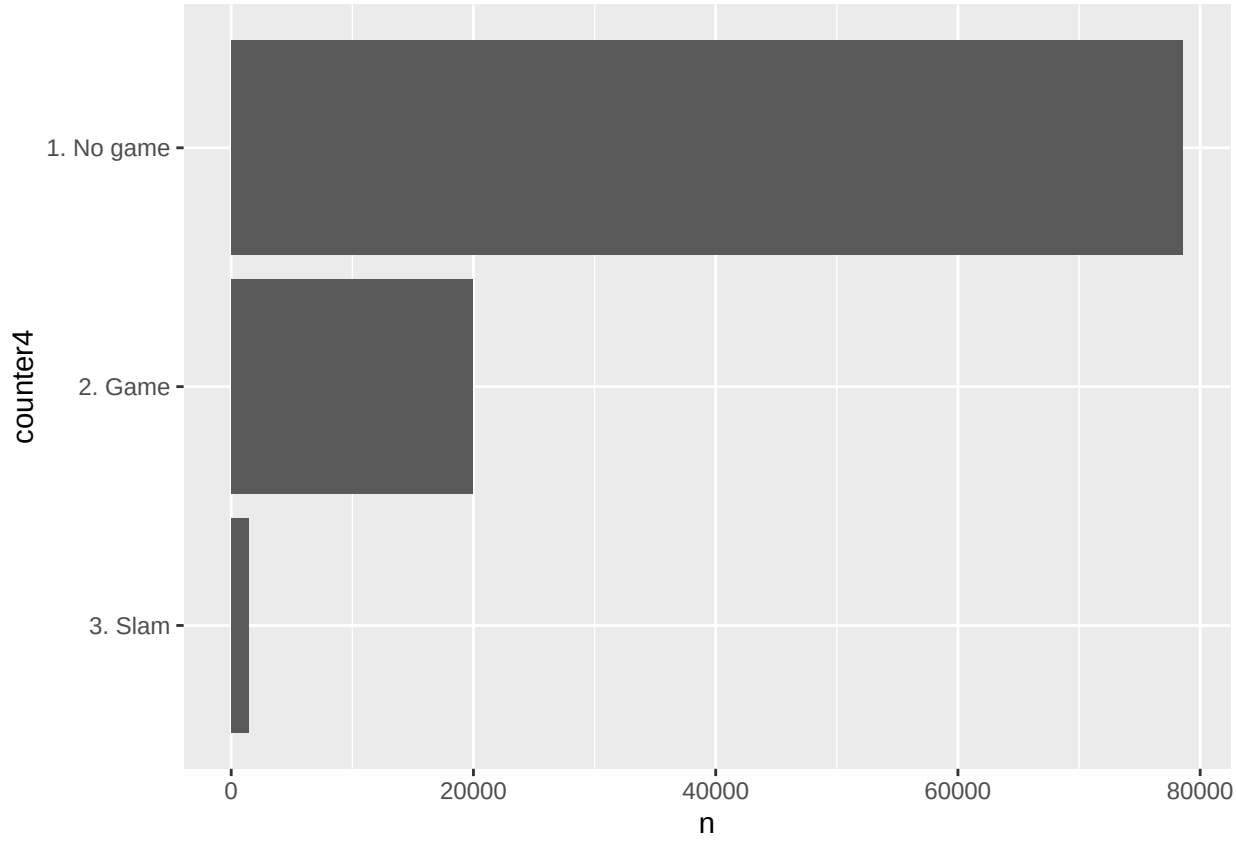
Now, the results will be slightly different when I knit this report versus what I'm seeing now, but you can simply divide all these results by 10^5 to get respective probabilities. The way things are coded, a purely passed opening hand occurs between 12 and 12.5% of the time, second only to an opening of 1 diamond, between 12.5 and 13% of dealt hands. Opening of one of a major suit (hearts or spades) each happens roughly 7 percent of the time. The preemptive bids (2 diamonds, hearts, spades, and all 3- and 4- level bids) happen about as frequently as opening one of a major, which from my experience seems about right!

However, the whopper hands, 2 clubs and 2 no trump openers, happen about 1% of the time, which, again, feels right. Given that 2 no trump depends on even distribution, as described in the code, while 2 clubs, a stronger hand, can be aided by having smaller or even void-length suits so as to emphasize a strong suit, the more than 2:1 ratio between them seems appropriate.

But my original question mostly accounts for strong opening hands, so we're looking at the 1-level, 2 clubs, or 2 no trump. Give or take a percent, it comes out to 40% of the time. We'll get back to this later.

```
counter4 <- unlist(counter3, use.names = FALSE)
counter4 <- data.frame(counter4)
counter4 %>%
```

```
  count(counter4) %>%
  mutate(counter4 = fct_reorder(counter4, n)) %>%
  ggplot(aes(y = counter4, x = n)) + geom_col()
```




```
table(counter4)
```

```
## counter4
## 1. No game    2. Game    3. Slam
##      78567      19966       1467
```

This chart is much more easily interpreted: slam hands happen ~1.4 % of the time, game levels around ~20%, and a more even spread of points and cards around 78%. Given that these hands were determined mostly by point count, and given how high card points are static but distributional points are quite variable, these results also make sense based on personal experience, though they could be subject to confirmation bias. If I'm honest, the game levels seem a little low, but there's a reason for this:

Contracts are not guaranteed to be bid based on hand holdings alone. There would have to be a significant amount of machine learning involved to incorporate the chaos of human behavior outside of my added inflation factor. However, it seems like more than 1/5 hands reach the game level in real life, so this may be an overly conservative model. These partnerships suck more than in real life!

Conclusions/Future Work

For what it's worth, I don't really care to separate these two sections as they're closely connected.

I'll start by saying my open() results are fairly wrong, in that passing is not the most common bid. It (I think) shouldn't occur more than a diamond, though I believe I've pinpointed why: certain cards in

certain situations are being double-counted, which should form half the basis of future coding. Consider the following scenario: you have a hand with 13 high card points, but one suit consists solely of a king. Another suit consists solely of an ace. By this logic, those two cards sum to 9 points: 7 for the king and ace, and a point apiece for the nature of just having one card in a suit. The ace is almost certainly a guaranteed trick, worth its four points at face value, and then it creates a gap in your hand which you can use to trump an opponent's card. However, the king is worth less than its three-point face value simply because there's a 2/3 chance your opponents old the ace! If they lead a trick with it, you're forced to lay down your king and its power becomes meaningless. The same scenario occurs with suits such as {king, queen}, where the queen becomes less valuable for the same reason. Additionally, there are different rules based on which seat opens. For example, if South West and North all pass to start the round, East is able to open with a weaker hand than were they the initial dealer, and thus start the bidding.

With game level bidding, slam potential is certainly higher than what was given, though in many instances there aren't viable ways to get to the bid you want. To calibrate that function, it would involve advanced measures which, at the very least, would involve incorporate linked bidding–possibly starting with a vector of all 35 suited/no trump bids which would truncate itself after each bid, a passing function instead of merely an opening bid, and a way to incorporate the double bid, which isn't relevant to this project. Most partnerships have at least some form of a golden fit, it just depends on how distribution falls–that's the real defining factor of hand strength, not simply high card points.

I wouldn't necessarily call this simulation a failure, rather a good start, especially for folks who haven't played bridge before. I'm just glad I could take a lifelong passion and bring it into statistical relevance, even though the actual statistics were minimal. It's the code I always wanted to do.