Mathematical analysis of the War of Attrition through calculus methods and computer simulation in R

Michael Li
Andy Zhang
Nathan Zhang

Advisor: Sebastian J. Schreiber, Ph.D, Mathematics Professor at UC Davis

Mathematics Department, University of California at Davis, Davis, California

Published August 2nd, 2014

Abstract. This paper presents mathematical and computational examination of the war of attrition game and its Nash equilibria. In our work, we utilized differential and integral calculus as well as R simulation software. Through code in the R language, we were able to show that after 150 timesteps, the simulated population density curve did indeed match the Nash equilibrium curve for the particular scenario. Since the simulation approached the theoretical curve, we concluded that the Nash equilibrium for the war of attrition was stable and given by $p(x) = \frac{1}{V} e^{x/V}$, in which V is the value of the prize at stake, x is the cost associated with a particular strategy and $p(x)$ is the probability of continuing to play the game at a cost x.

i. Introduction

The War of Attrition is a game at the intersection of applied mathematics and economics that has existed for centuries. Perhaps the most familiar example of it is the auction. Multiple contestants enter into an auction for one item that each contestant desires. The cost of that item keeps on increasing as the contestants bid higher and higher for the item. Eventually, some contestants will drop out of the game because they feel that the cost is too high for them to afford. The purpose of this paper is to analyze the Nash equilibrium for this game and verify its stability through both mathematical and computational methods in the R language. The work presented has applications in corporate finance, e.g. when two corporations are vying for ownership of a smaller company. It can also be applied to animals competing for territory in the wild. Finding and utilizing a Nash equilibrium strategy can lead to valuable gains, whether financial, territorial or of another nature.

In our analysis process we assumed a War of Attrition game of a general nature, that is, competitors are simply defined as competitors and the prize value is unitless. Each competitor was assigned a pure strategy to play. After 150 simulations were run, the final results were graphed and compared to the theoretically predicted Nash equilibrium graph for the predetermined conditions.

ii. Theory

The Nash equilibrium of this game cannot be a pure strategy. In other words, it is not the best strategy to always forfeit the game or always compete indefinitely. Therefore, the Nash equilibrium must be a mixed strategy, with p.d.f.[1] $p(x)$ of continuing to play the game. The probability of continuing is dependent on the value V at stake, because to each opponent, the value may be different and thus the point in persisting may be different. In other words, $p(x)$ is the probability of accepting a cost x, and $p(x) * \partial x$ is the probability of accepting a cost between x and $x + \partial x$.

We will define Player 1's strategy as m, which is the cost of playing the War up until the point at which Player 1 withdraws.

Logically, the expected value of one's winnings in layman's terms would be

$$E[m, p(x)] \; = (Probability\ of\ losing)(Cost) + \; (Probability\ of\ winning)(Profit\ - Cost).$$

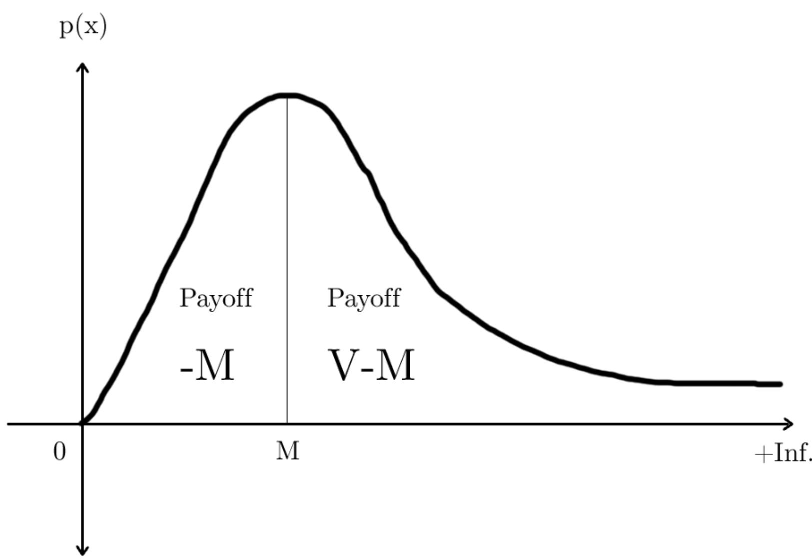

Figure 1.1 | P.d.f. graph

Using our knowledge of integral calculus, we can see that the integral of the p.d.f. on $[0, M]$ gives the probability of losing M. The integral of the p.d.f. on $[M, \infty]$ gives the probability of profiting $(V \, - \, M)$.

[1] Probability density function

We write the formal expected value equation as follows:

$$E[m, p(x)] = \int_0^m 1 \ (V - x)\, p(x)\, dx - \int_m^\infty 1 \ m\, p(x)\, dx$$

Now, we solve for the value of $p(x)$ that will produce a Nash equilibrium in $E[m, p(x)]$. If $p(x)$ is a Nash equilibrium, you can do no better than your opponent. Therefore changes in m will have no effect on the expected value of your winnings.
We take the derivative $\partial E / \partial m$ to find the point at which changes in m produce no changes in E. When $p(x)$ is a Nash equilibrium $\frac{\partial E}{\partial m} = 0$.

$$\frac{\partial E}{\partial m} = p(m)\,(V - m) - \left(\int_m^\infty 1 \ p(x)\, dx - \frac{\partial}{\partial m} \lim_{b \to \infty} \int_b^m 1 \ p(x)\, dx\right)$$

$$V \frac{dp}{dm} = p(m)$$

After inspection, we decide to have $p(m)$ represented by some function e^g.
The equation then can be converted into:

$$\frac{dg}{dm} e^g = \frac{e^g}{V}$$

It is immediately obvious that $g = m/V$.
Therefore $p(m)$ can be expressed as $\frac{1}{V} e^{m/V}$, which can be generalized to all x as $p(x) = \frac{1}{V} e^{x/V}$.

iii. Stability of Equilibria

To prove the stability of the Nash equilibrium, we must show that:

$$E(p(x), p(x)) > E(m, p(x)).$$

Or if $E(p(x), p(x)) = E(m, p(x))$, then $E(p(x), m) > E(m, m)$.

Considering the second condition, we must show that

$$E(p(x),m) - E(m,m) = 2Ve^{-m/V} - 1.5V - m > 0$$

In other words, we must show that the function is always positive. By differentiating with respect to m and finding the roots of the resulting function, we obtain that $m = 0.693V$. Substituting m into the original function gives us $0.193V$ as the result, which is a value that is always positive. Thus, we have demonstrated the validity of $E(p(x),m) > E(m,m)$ -- the Nash equilibrium is stable.

iv. Computer Simulation

The purpose of our computer simulation is to confirm that our calculated expression is a stable Nash Equilibrium. A Nash Equilibrium has the property that if a population of pure strategies were allowed to compete with themselves, the population density curve would look similar to the Nash Equilibrium. This property makes logical sense since each individual is playing against the population and since the optimal strategy for the opponent is the Nash Equilibrium, the whole population will take the form of a Nash Equilibrium.

We created a simulation of a population using R to see if the population density would eventually reach the Nash Equilibrium. The simulation used the following conditions. Each member would play a pure strategy, meaning they would keep playing the war of attrition until they reached their predetermined maximum cost and would quit. For simplicity's sake, we assume that the population stays constant, and every generation, the members that die are replaced by offspring. The ratios of the offspring that enter the population are based on the reproductive success of that strategy, which is calculated by averaging the payoff of that strategy against the rest of the population. Therefore, the frequency of the population for strategy m at the next timestep $(t+1)$ can be evaluated with the following equations:

$$f_{m,t+1} = (1-d)f_{m,t} + d\left(\frac{p_m(f_{m,t}) * f_{m,t}}{\sum_{i=1}^{\infty} p_i(f_{i,t}) * f_{i,t}}\right)$$

and

$$p_m(f_{m,t}) = \sum_{i=1}^{m-1} f_{i,t} * (V-i) + f_{m,t} * (V/2 - m) + \sum_{i=m+1}^{\infty} f_{i,t} * (-m)$$

with d representing the death rate, and V representing the prize value.

During preliminary programming, we established a test model that would utilize the following parameters. The prize value was set at 20. There would be 1000 different strategies ranging from quitting at a cost of 0.1 to quitting at a cost of 100. Half of the population would die each generation, and the simulation would proceed for 150 generations.

The results of our simulations are displayed below. The black line represents the population density curve at a certain time step while the red line represents the Nash Equilibrium. The first graph displays the original population in which the frequency of all the strategies are the same, hence the flat black line.

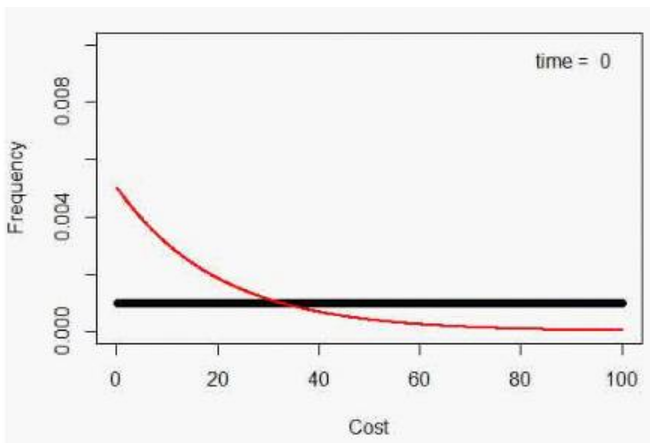

Figure 1.2 | Simulation 1 Results

The next graph displays the frequencies after 18 timesteps. The population is clearly beginning to approach the Nash Equilibrium.

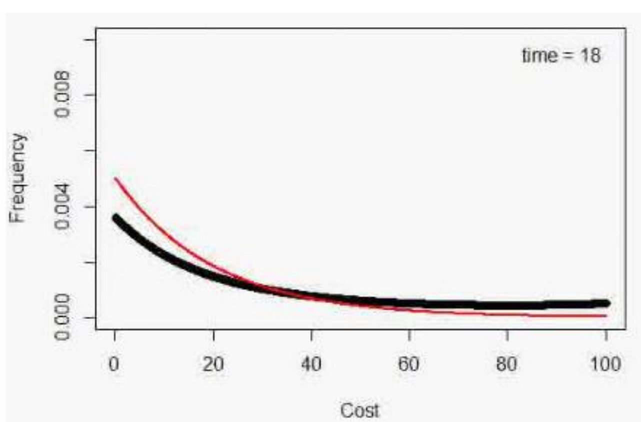

Figure 1.3 | Simulation 2 Results

Our simulation was programmed to terminate after 150 generations, and the population density curve almost exactly fit the Nash Equilibrium line, with very slight deviations at a few strategy values.

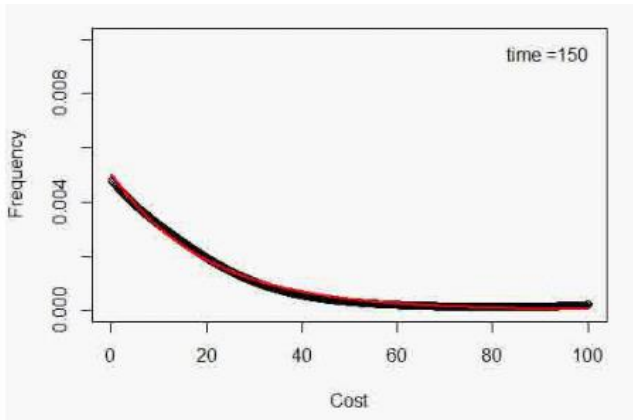

Figure 1.4 | Simulation 3 Results

The extreme unity of the Nash Equilibrium and the population density curve after such a long simulation lends strong support to the accuracy and stability of our calculated Nash Equilibrium.

Admittedly, the model does its best to simulate a real population but contains have imperfections. A true population of these strategies allows for any differential value dm, while our model only allowed increments of 0.1. This could account for slight differences all across the graph. Additionally, our model set a maximum value of 100, a clearly ineffective strategy for a prize value of 20. However, since there were no strategies that played a strategy higher than 100, playing a strategy of 100 guaranteed the prize value of 20 and thus was more effective than it should have been. For a finite maximum value, this slight overrepresentation of frequency is inevitable and accounts for the slight increase in the maximum cost strategies compared to the calculated Nash Equilibrium. Lastly, the Nash Equilibrium is not a population density function so the total area did not add up to 1. For our purposes, we scaled it down to act like a population density function by dividing each value by the sum of the area under it, making the total value 1.

Our simulated population revealed that our Nash Equilibrium is indeed stable since the population of pure strategies eventually had densities similar to the Nash Equilibrium.

v. Simulation Code in the R language

```r
kmax=1000                    #when *dx will get largest length
dx=0.1                       #amount of each step
A=matrix(0,kmax,kmax)
V=20                         #amount you get from winning
numI = 150                   #number of iterations
d=.5                         #amount of population that dies each round
tplot=2

xAxis=numeric(kmax)

for(i in 1:kmax)
{
  xAxis[i]=i/10
}

pauseit <- function(x) { p1 <- proc.time()
                         Sys.sleep(x)
                         proc.time() - p1 # The cpu usage should be negligible
}


for(x in 1:kmax)
{
  #my strategy is x; my opponents is y
  for(y in 1:x)
  {
    if(y==x){}
    else
    {
      A[x,y]=V-y*dx+kmax*dx
    }

  }
  for(y in x:kmax)
  {
    if(y==x)
    {
      A[x,y]=V/2-x*dx+kmax*dx
    }
    else
    {
      A[x,y]=-x*dx+kmax*dx
    }
  }
}
nash=numeric(kmax)
for(i in 1:kmax)
{
  nash[i]=1/V*exp(-i*dx/V)
}

sNash = sum(nash)
for(i in 1:kmax)
{
  nash[i]=nash[i]/sNash
}
#for(i in 1:kmax*dx)
#{
#  nash[i]=1/V*exp(-i/V)
#}
p=matrix(0,kmax,numI)
p[,1]=1/kmax

#initial plot
plot(xAxis,p[,1],ylim=c(0,.01),type="b",xlab="Cost",ylab="Frequency")
lines(xAxis,nash,col="red",lwd=2.5)
text(89,.0095,"time = ")
text(97,.0095,"0")
pauseit(1.0)

for(i in 2:numI)
{
  p[,i]=(1-d)*p[,i-1]+d*p[,i-1]*(A%*%p[,i-1])/sum(p[,i-1]*A%*%p[,i-1])

  #p[1,i]=(1-d)*p[1,i-1]+d*p[1,i-1]*(A%*%p[,i-1])[1]/sum(A%*%p[,i-1])
  if(i%%tplot==0)
  {
    plot(xAxis,p[,i],ylim=c(0,.01),type="b",xlab="Cost",ylab="Frequency")
    lines(xAxis,nash,col="red",lwd=2.5)
    text(89,.0095,"time = ")
    text(97,.0095,i)
    pauseit(0.05)
  }
}




#pGraph = numeric(kmax*dx)
#nGraph = numeric(kmax*dx)
#
#for(i in 1:kmax*dx)
#{
#  pGraph[i]=0
#  nGraph[i]=0
#  for(j in 1:10)
#  {
#     pGraph[i]=pGraph[i]+p[i*10-j+1,numI]
#     nGraph[i]=nGraph[i]+nash[i*10-j+1]
#  }
#}

```

vi. References

Smith, John Maynard. Evolution and the Theory of Games. Cambridge: Cambridge UP, 1982. Print.