Labor Market Matching Affected by Unemployment Insurance Benefit and Duration

By Rachael DeVeau

May 2020

With special thanks to Professor Patrick Walsh and Zachary Copland

Table of Contents

Abstract

This agent-based simulation model was created to find the optimum level of unemployment insurance and its duration by utilizing search theory to ensure a more efficient society with a lower unemployment rate. The model consists of applicants (the unemployed) and employers meeting randomly each time period. Optimum matches occur with the correct skill level and when no superior match can be found in the next time period. The productivity level is measured for each match and the closer a match is to the optimum, the more productive the match. Unemployment insurance/benefit is utilized to enable the highest level of societal productivity possible to ensure a thriving economy. Overall, this study found increased benefit level, decreased productivity and a decreased success rate of matching. Benefit level had no effect on unemployment duration and an increase in the unemployment population positively correlates with increased unemployment levels.

I. Introduction

Unemployment rate is an indicator of labor market performance, the higher it is the worse an economy is preforming and the lower it is the better an economy is performing. Keeping the unemployment rate low helps individual families stay fed and happy, in addition to help keeping the country's economy growing. When people become unemployed, they lose their purchasing power, families lose wages, the economy loses productivity, and those who are working need to put in more time and effort without additional compensation due to the economy taking a downturn. This causes even more people to become unemployed and the situation negatively escalates.

In the United States, among other counties, people who are unemployed receive unemployment insurance benefits for some amount of time. It is designed to provide temporary

relief to those between jobs. Unemployment insurance is utilized to produce a more productive society by allowing the unemployed (applicants) to be more selective before accepting a job (Diamond, 1981). This enables them to focus on finding a job that provides them with the most utility and the job they can be the most productive at. This is preferable to applicants accepting the first job they find for it pays the bills while having a low productivity level. In a recession or down turned economy when many people are looking for a job, if most applicants accept the first job they can find, this low productivity compounds and negatively affects the economy. While people now have jobs, they are working jobs they dislike and are not productive at (Mortensen, 1977). This effect leads to a less productive society, which then causes the economy to take even longer to thrive again. That is, to effectively improve a down turning economy, having high economic productivity is crucial.

The most common duration of unemployment benefits is 26 weeks or less; however, for the Great Recession it was extended up to 99 weeks for many states (Chikhale, 2017). Despite this, many people were still unemployed after 99 weeks. Was this due to the change in duration of unemployment benefit? Or was this due to the level of generosity of the benefit? Finding the answers to these questions will help a recovering economy return to high productivity and growth. It is hypothesized that a median level of benefit will enable productivity to increase over time. The research question of this paper is: how does the change in duration and level of generosity for umployment insurance affect economy labor productivity and market matching success rates?

To answer this question, a model was built using a software called PyCharm to run Python 3.7 (Appendix). All of the data used in this study was created by the model, which utilized search theory. The model assigned random skill levels (1-100) to each of the five skills

(A-E) for each applicant and assigned a random desired skill to each employer. Then, the applicants and employers are randomly paired during each time period. If the applicants and employers pass all the constraints required for them to match, a match is made. This means the applicant and employer who matched are pulled out of the pool of people waiting to make a match. The simulation is over when all possible matches between applicants and employers have been made or all time periods have been run, whichever comes first. For this model, four different scenarios were run to look at the effect of four different benefit levels: no benefit, 25, 50, 75, and 100. Benefit indicates the monetary amount of unemployment insurance allotted to those who are unemployed. The duration an applicant would have this unemployment insurance benefit was changed while other variables were kept constant to analyze its effect on economy productivity as well as market matching success rates.

After performing upwards of 30,000 simulations, it was found that with each benefit level increment, the lower the productivity of society and the lower the success rates of matches became. It was also found that the higher the population (applicants and employees) in the seeking pool, the more the duration of unemployment was affected. Benefit level was found not to have an effect on unemployment duration.

II. Literature Review

One of the most utilized solutions during a recession is unemployment insurance, thus, it has been greatly studied to ensure it is a viable solution. One important finding is that job destruction rates increase more rapidly during a recession while those rates fall during an economic boom. Pissarides (2013) found the higher the number of unemployed, the longer the mean duration of unemployment becomes. This is caused by a number of factors but is largely due to limited job creation. The more individuals laid off, the less people consume, and the more

people save, which leads to businesses making less profit (Pissarides, 2013). Businesses that make less profit cannot afford many employees and consequently are forced to fire them. Unemployment insurance allows consumers to keep doing just that, consume. This prevents further economic downturn. Most of the research completed on the topic have been mathematical models, regressions, and most recent models have included search theory. Decisions that affect the rate one is employed by has been the focus of earlier analysis.

The incorporation of search theory became very valuable to economic research regarding unemployment benefit since its beginnings. Search theory is an individual's optimum strategy considering all their opportunities with the assumption that waiting to choose will come at a cost. Holmlund (1998) utilized search theory in conjunction with a union bargaining mathematic model to demonstrate the assumption that decisions about the acceptance/rejection of job offers are made by individuals acting alone. This resulted in finding that a higher replacement ratio in the long run has a high impact on aggregate unemployment (Holmlund, 1998). However, some of these assumptions by Holmlund are quite restrictive. His results concluded higher generosity of unemployment insurance leads to longer unemployment as well as higher overall unemployment levels.

Filits took a different approach in his paper *The Effect of Unemployment Insurance Generosity on Unemployment Duration and Labor Market Transitions*. He used data from 2002 to 2012 about unemployment generosity in Turkey and its effect on the duration people are unemployed for. Filits ran a regression using the Turkish Employment Agency (ISKUR) data. After the analysis, he came to the conclusion that the duration of time people are unemployed for increases as unemployment benefits increase (Filis, 2017). Another similar study was in 2000 by Card and Levine. They found a positive correlation between unemployment benefit and

unemployment duration (Titio, 2010). Meaning they found the higher the level of unemployment insurance, the longer the unemployment duration is. Even though Holmlund's (1998) approach was different from Filits (2017) and Card and Levine (2000). All three came to the same conclusion: high unemployment benefit leads to high unemployment duration.

Mortensen (1977) performed a study with two groups, one being those who did qualify for unemployment insurance and the other being those who did not qualify for unemployment insurance. For those who qualified, his results differed from those of Holmlund and Filits. Mortensen's conclusion was that he could not conclusively find that an increase in unemployment insurance had an effect on unemployment duration for those who qualified for the unemployment insurance. But he did conclusively find that those who did not qualify for unemployment insurance found employment more quickly (Mortensen, 1977). His analysis was that those who found jobs quickly were those who had lowered their expectations and searched more intensely.

An interesting analysis was by Lalive and Zweimuller who came to the conclusion that by not correcting for policy endogeneity, the potential to significantly overestimate the negative effects of unemployment insurance on unemployment duration is highly probable (Titio, 2010). Diamond (1981) found that with unemployment insurance, the unemployed are able to be more selective in their job choosing decisions, which leads to a more productive society. The unemployed are willing to wait for more attractive offers if given the chance, unemployment insurance provides that chance (Diamond, 1981). Diamond's findings in 1981 provide a strong case for the utilization of unemployment insurance. In general, most studies have found there are more positive effects of unemployment insurance on the duration of unemployment when unemployment insurance is measured in terms of the duration of benefits (Titio, 2010).

While there are studies analyzing unemployment benefit across the duration of unemployment, there are few that do this using an agent-based simulation. A simulation allows for better control over the study to prohibit uncontrollable variables from affecting the results. Of all past papers there was a lack of applied application in a closed environment for this study to analyze the direct results. An agent-based simulation is able to provide valuable insight to the study of unemployment insurance by directly finding the effects of all inputs.

III. Methodology

Search theory was utilized in this agent-based simulation to test economic theory about labor market matching. The model was chosen to be a simulation because of its vast possibilities, data which can be easily collected, and variables that can be easily manipulated/controlled. All data collected for this study came from the simulations this model produced using Python 3.

The agents are the unemployed (applicants) and employers. The duration (time period) and unemployment insurance (benefit) levels are the influencer variables. The matches are the action variables. Each applicant has five skills that are all random skill levels (1 through 100) and the employers are each given a random skill out of the five that they are looking for. These are the fixed variables in the model.

The constraints of the model are what create a match between an employer and an applicant and what prevent a less optimum match. Applicants and employers only match if all of the following constraints are true:

1. One of the top three skills an applicant has is the skill an employer is looking for.
2. For the skill an employer is wants, the skill level an applicant has is higher than or equal to the threshold (minimum skill level) the employer expects.

3. The benefit an employer would receive by hiring an applicant in this time period needs to be higher than the possible benefit the employer would receive by hiring an applicant in the next time period.
4. The value for an applicant to search for another job in the next time period needs to be lower than if that applicant were to accept a job in this time period.

Each individual applicant and employer have the goal of finding a match. The better the match, the higher the payoff an employer and the higher the offer for an applicant. The payoff an employer receives once matched and the offer an applicant receives once matched are both recorded as well as what time period the two matched in.

i. The Logic of Matching

The duration of unemployment insurance and the level of benefit each applicant receives effects the match success rate and productivity. These are focal points of the study. All applicants will have A,B,C,D,E skills and will have a random skill level assigned to each in the form of 1-100. Applicants are looking to accept a job in one their top three skills that gives them the highest offer. This offer needs to be more than what the applicant could get if they search again. The employers are looking to maximize the payoff they receive by hiring the most qualified (highest skill level) applicant within the shortest amount of time. An employer only offers an applicant a job if the skill the employer is looking for is in the applicant's top three skills and if that skill is above the employer's threshold. There are a fixed number of time periods to prevent the simulation from running continuously.

The job search happens in time periods, with employers and unemployed encountering each other randomly and being able to identify the match quality. If the match quality is what both are looking for then the applicant and employer are able to make an immediate match, but if

it is not, then both move on to the next time period. This is where the level of generosity and duration of unemployment benefits comes into play. If there is high unemployment insurance benefit for a long duration and the offer the employer gives is lower than the unemployment insurance benefit, then it is predicted people will stay unemployed. However, if there is low benefit for a short duration, then it is predicted the applicant will go with the first possible employer despite a high skill level in another skill. This is unideal, for even though the job they chose is paying the bills, they are being less productive at this job, than if they were working at a job with a higher match quality. When this is amplified by all unemployed people, the economy is less productive and stagnate.

A small-scale example with 5 time periods of the simulation is as follows:

Applicant 1 has skills: 55 A, 10 B, 52 C, 03 D, and 42 E

Applicant 2 has skills: 23 A, 45 B, 0 C, 60 D, and 37 E

Applicant 3 has skills: 66 A, 75 B, 90 C, 25 D, and 10 E

Employer 1 is looking for skill D

Employer 2 is looking for skill C

Employer 3 is looking for skill A

These 6 people encounter each other randomly in a double-blind pairing and when they meet. The employer desires to maximize the level of skill they are looking for to ensure the highest payoff possible. The applicant desires the highest offer.

For all employers, their threshold, value to search again, and payoff is the same for each time period but none of those are the same in each time period. For each applicant, their value to search again as well as their offers are contingent upon their skill set.

Threshold (the minimum skill level an applicant needs for that time period) is a skill level of about 48, the payoff is about 370, and the value to search again is about 236 for all employers in time period one (calculated on page 12). From knowing this, the employers want to find an employee in the first time period for the payoff in time period 1 is 370, which is higher than what employers would expect it to be in time period 2, which is 236.

If employer 2 meets applicant 3:

Applicant 3 has top skills in C, B, and A and employer 3 is looking for skill C. Employer 2 will offer applicant 3 a job because as was stated earlier the employer's payoff for this time period is larger than what it would be in the next and applicant 3 has skill level C in their top three skill set. Applicant 3's highest skill level is in skill C because of this they are wanting to match in skill C more than the other two options for this is the highest offer they can receive. Only after all this, does applicant 3 accept and the two are pulled out of the pool of unmatched.

If employer 2 was looking for skill A, which is in applicant 3's top skills, because this is the first time period, applicant 3 would not accept. The odds of finding another employer who is looking to for an applicant with skill C is 1 in 4. In time period 1, the applicant's value to search is higher than that of the employer's offer for skill A. However, if it were time period 4 or 5 and applicant 3 had not yet matched, they would be more likely to settle for a lower offer. This is because after all time periods are over, none of the applicants or employers receive any payoff if they did not match. This incentivizes employers and applicants to match sooner rather than later for there is a cost to delaying their decision. This is search theory applied.

IV. Model

i. The Employer

Once a random employer has met with a random applicant, the employer checks if the skill they are looking for is in the applicant's top three skills and if it is, then the employer checks if the skill is above the expected threshold for that time period. If an applicant passes both those check points, they are offered a job and an offer. If the applicant does not pass one of those check points than the employer does not make them an offer of employment. It is at this point that the code is written to take the unmatched applicant and employer out of the pool so that they do not meet with anyone else for that time period.

The employer's threshold is the minimum skill level an employer would expect an applicant to have for a particular time period. As each time period unfolds and as the employer continues to search, the expected threshold for a skill is lowered for it is in the employer's best interest to hire a qualified person as soon as possible. The equation for threshold is the value for the employer to search in the next time period divided by the remaining time periods. The variable "n" is the total number of time periods and the variable "time" is the current time period.

$$Threshold = \left(\frac{\mathrm{VSearchE}}{\mathrm{n-time}}\right)$$

Payoff is what an employer will receive if they accept an applicant in that time period and all following. This is also dynamic and changes every time period. It is what an employer would expect the next skill level to be ((100 + threshold) / 2) multiplied by the remaining time periods (n – time) plus the current time period (+ 1). In the last time period, the average level of skill an

applicant would have for any particular skill would be 50, for each skill has a uniform distribution along the interval [0,100].

$$Payoff = \left(\frac{100 + \text{threshold}}{2}\right) * (\text{n} - \text{time} + 1)$$

For employers to ensure they maximize their highest possible payoff, the employers compare the payoff they would receive after hiring someone with what they could get if they searched again. In addition, because there is a uniform distribution of skill level along the interval [0,100], this indicates that in the second to last time period (n – 1) the employer would expect the value of their next search to be 50. In the last time period, the value to search in the next time period is zero for there are no more time periods.

VSearchE =

In the last time period: 0

In the second to last time period:

$$\left(\frac{100 - \text{threshold}(\text{time} + 1)}{100}\right) * \left(\frac{100 + \text{threshold}(\text{time} + 1)}{\text{n} - \text{time} + 1}\right) = 50$$

For all other time periods:

$$\big(\text{ProbS}(\text{time} + 1)\big) * \big(\text{Payoff}(\text{time} + 1)\big) + \big(\text{ProbA}(\text{time} + 1)\big) * \big(\text{Payoff}(\text{time} + 1)\big)$$

The equation VSearchE in the majority of the time periods uses the probability an employer will search again for another applicant (ProbS). This is the current employer threshold

divided by 100. VSearchE also uses the probability that an employer will accept an applicant (ProbA). This is one minus ProbS.

$$ProbS = \frac{\text{threshold}}{100}$$

$$ProbA = 1 - \text{ProbS}$$

The following is an example table how the equations interact for one employer.

Time Period	Threshold	ProbS	VSearchE	ProbA	Payoff
1	48.12	0.48	235.60	0.52	370.30
	$\left(\frac{235.60}{5}\right)+1$	$\frac{48.12}{100}$	$(0.43*168.80)+(0.57*286.40)$	$1-0.48$	$\left(\frac{100+48.12}{2}\right)*5$
2	43.20	0.43	168.80	0.57	286.40
	$\left(\frac{168.80}{4}\right)+1$	$\frac{43.20}{100}$	$(0.36*106.24)+(0.64*204.62)$	$1-0.43$	$\left(\frac{100+43.20}{2}\right)*4$
3	36.41	0.36	106.24	0.64	204.62
	$\left(\frac{106.24}{3}\right)+1$	$\frac{36.41}{100}$	$(0.26*50)+(0.74*126)$	$1-0.36$	$\left(\frac{100+36.41}{2}\right)*3$
4	26.00	0.26	50.00	0.74	126.00
	$\left(\frac{50}{2}\right)+1$	$\frac{26}{100}$	$\left(\frac{100-0}{100}\right)*\left(\frac{100+0}{2}\right)*(1)$	$1-0.26$	$\left(\frac{100+26}{2}\right)*2$
5	0.00	0.00	0.00	1.00	50.00

As is shown in the table above, the equations work backwards through the time periods. All the equations are dependent on the equations for VSearchE and Threshold. The dashed orange lines follow the path VSearchE takes to create Threshold all within the same time period.

The solid blue lines connect the equations that are all reliant on the equation for Threshold. The solid orange line is showing all the equations are inputs for VSearchE in the next time period.

This small example contains only five time periods. The simulation was created to eliminate these calculations by hand to compute them on a large scale.

ii. The Applicant

Upon meeting with a random employer who is looking for a random skill (A-E), if the employer makes that random applicant an offer the applicant has two options. One is to accept the job, which an applicant will only do if the employer's offer is greater than the applicant's probability of finding a better offer in the next round. The employer's offer depends on the applicant's skill level. Therefore, the offer is always going to be the applicant's skill level for the skill the employer is looking for multiped by the number of remaining time periods in addition to the time period they matched in. The variable "n" is the total number of time periods and the variable "time" is the current time period.

$$\text{valOfHighest} = \text{highestSkill} * (\text{n} - \text{time} + 1)$$

$$\text{valOf2Highest} = \text{secondHighest} * (\text{n} - \text{time} + 1)$$

$$\text{valOf3highest} = \text{thirdHighest} * (\text{n} - \text{time} + 1)$$

The other option an applicant has is to not accept the employer's offer. An applicant would do this if the value to search for another offer is higher than the offer provided by the current employer the applicant is meeting with. The value for an applicant to search again is calculated by multiplying the probability of an employer offering a job in the applicant's highest skill with the offer they would get for their highest skill. Added to the probability of an

employer offering a job in the applicant's second highest skill multiplied with the offer they would get for their second highest skill. Plus, the probability of an employer offering a job in the applicant's third highest skill multiplied with the offer they would get for their third highest skill. Added to one minus the probability of an employer offering a job in the applicant's highest skill minus the probability of an employer offering a job in the applicant's second highest minus probability of an employer offering a job in the applicant's third highest skill. Added to the applicant's value to search in the next time period after that plus unemployment insurance. All of this is calculated in for the next time period.

$$\begin{aligned} \text{VSearchA} = {} & (\text{probHiOff} * \text{valOfHighest}) + (\text{prob2offer} * \text{valOf2Highest}) \\ & + (\text{prob3offer} * \text{valOf3highest}) \\ & + (1 - \text{probHiOff} - \text{prob2offer} - \text{prob3offer}) \\ & + (\text{VSearchA}(\text{time} + 1) + \text{unemployment insurance}) \end{aligned}$$

The variables "probHiOff", "prob2offer", and "prob3offer" used in VSearchA are the probabilities of being offered a job in the applicant's highest, second highest, and third highest skill respectively. The probability is 20% if the skill is greater than or equal to the employer's threshold of the current time period and is zero if the skill is less than the employer's threshold of the current time period.

Probability of offing a job for the applicant's highest(probHiOff), 2nd highest(prob2offer), & 3rd highest(prob3offer) skill:

Is 0.2 or 20% if skill >= threshold

Is 0 if skill < threshold

The following is an example table without unemployment insurance for one applicant.

Skill	Skill Level
Highest	55
2nd Highest	52
3rd Highest	42

Time Period	Value to Search Again	Probability of an offer for highest skill	Value of Highest Offer	Probability of an offer for 2nd highest skill	Value of 2nd Highest Offer	Probability of an offer for 3rd highest skill	Value of 3rd Highest Offer
1	156.40	0.20	275	0.20	260	0.00	210
	$(0.2 * 220) + (0.2 * 208) + (0.2 * 168) + (1 - 0.2 - 0.2 - 0) + (118.01 + 0)$	$55 > 48.12 \therefore 0.20$	$(55 * 5)$	$52 > 48.12 \therefore 0.20$	$(52 * 5)$	$42 < 48.12 \therefore 0$	$(42 * 5)$
2	118.01	0.20	220	0.20	208	0.00	168
	$(0.2 * 165) + (0.2 * 156) + (0.2 * 126) + (1 - 0.2 - 0.2 - 0.2) + (71.52 + 0)$	$55 > 43.20 \therefore 0.20$	$(55 * 4)$	$52 > 43.20 \therefore 0.20$	$(52 * 4)$	$42 < 43.20 \therefore 0$	$(42 * 4)$
3	71.52	0.20	165	0.20	156	0.20	126
	$(0.2 * 110) + (0.2 * 104) + (0.2 * 84) + (1 - 0.2 - 0.2 - 0.2) + (29.80 + 0)$	$55 > 36.41 \therefore 0.20$	$(55 * 3)$	$52 > 36.41 \therefore 0.20$	$(52 * 3)$	$42 > 36.41 \therefore 0.20$	$(42 * 3)$
4	29.80	0.20	110	0.20	104	0.20	84
	$(0.2 * 55) + (0.2 * 52) + (0.2 * 42) + (1 - 0.2 - 0.2 - 0.2) + (0 + 0)$	$55 > 26.00 \therefore 0.20$	$(55 * 2)$	$52 > 26.00 \therefore 0.20$	$(52 * 2)$	$42 > 26.00 \therefore 0.20$	$(42 * 2)$
5	0.00	0.20	55	0.20	52	0.20	42

The black lines represent equations that do not rely on other equations, but only the level of skill and applicant has. The orange lines represent the all the inputs for value to search again for the applicant (VSearchA). The table shows all the equations are needed to find VSearchA.

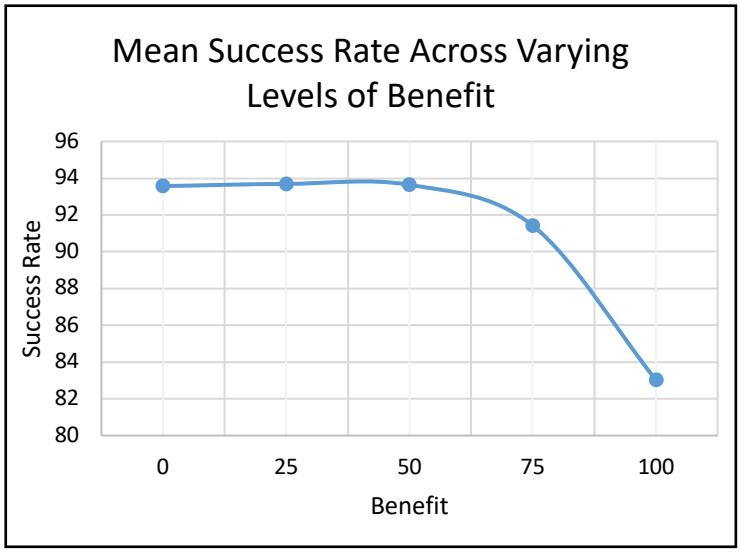
In the fifth time period, the value for the highest offer, second highest, and third highest offer are the highest skill, second highest skill, and third highest skill respectively because of the way the equation was designed. It has the applicant's skill level being multiplied by how many time periods are left plus the current time period. This was done to easily assign an offer which can acclimate to different skill levels and give a higher payoff to higher skill levels for that is what is desired. Unemployment insurance is the constant 0 in the value to search again column in this table.

If unemployment insurance was calculated into the table above it would allow for the applicants to wait for better offers instead of accepting the first offer possible. For instance, this table demonstrates that if this applicant were to meet an employer in the first time period, of which the value to search again is greatest, the applicant would accept any offer in their top three skills. This is indicated by the value to search again being at only 156, which is less than any offer this applicant would reactive for their top three skills. When unemployment is added into this mixture. It increases the value to search again, which allows the applicant to wait for an offer in their top skill level, one that applicant can be most productive at.

V. Results

This paper seeks to find the optimum level of benefit and the optimum duration of benefit. The following graphs have been devised from data created and collected from the simulations run with the code listed in the appendix.

Figure 1: Success Rate by Benefit Levels



A population (applicants and employers 2,000 each) of 4,000 over a duration of 50 time periods was run for 100 simulations for each benefit level. In total, there were 500 simulations run. For each of those simulations, the success rate of matches was recorded, and the mean was taken for each benefit level.

Benefit levels 0 through 50 had almost no change in success rate, staying at almost 94%. This means that at the end of all 100 simulations for a benefit level of 0, 25, and 50, about 94% of applicants and employers were matched. The success rate dropped to about 92% when the benefit level reached 75. The greatest drop in success rate with a 9% difference was from a benefit level of 75 at about a 92% success rate, to about an 83% success rate at a benefit level of 100. Figure 1 demonstrates that the higher the benefit level, the lower the success rate is for matching an employer with an applicant.

The following is information about Figure 2:

Productivity was analyzed by being split up into two groups: reality and ideal. Productivity of reality was calculated by multiplying the skill an applicant was matched with by the remaining time periods plus one to include the time period they were matched in. The ideal situation for all applicants would be to match with an employer in the first time period. By multiplying the highest skill level for each applicant by the number of total time periods is how

the ideal productivity if found. After dividing the productivity of reality with the ideal productivity, then subtracting that number by 100 was how the difference was found. To get the percent difference, that number was multiplied by 100. The variable "n" is the total number of time periods and the variable "time" is the current time period

$$Reality\ Productivity = (\text{skill of applicant}) * (\text{n} - \text{time} + 1)$$

$$Ideal\ Producivity\ = (\text{skill of applicant}) * (\text{n})$$

$$Percent\ Difference\ = 100 * \left(\left(\frac{\text{Reality}}{\text{Ideal}}\right) - 100\right)$$

Figure 2: Productivity by Benefit Levels

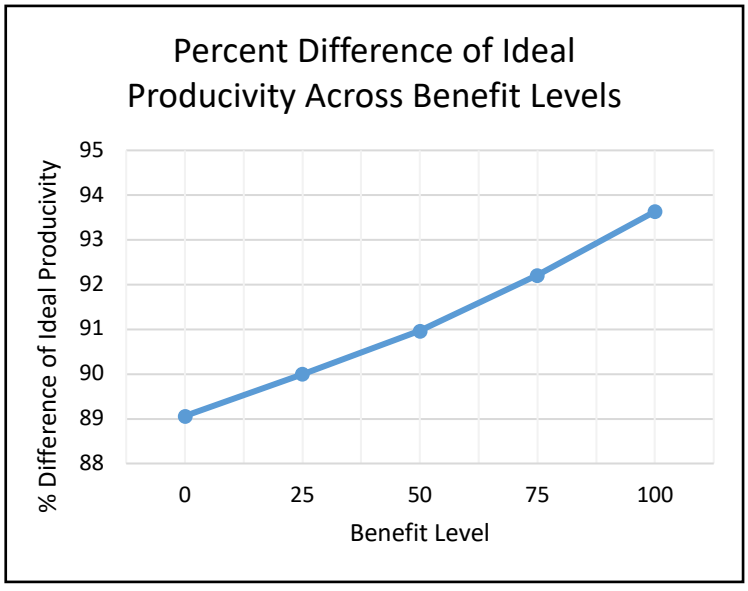


Figure 2 displays a population of 4,000 (2,000 applicants and 2,000 employers) for 50 time periods run for 100 simulations each benefit level. At the end of each 100 simulations for each benefit level the mean for both the reality productivity and the ideal productivity was taken. Then the percent difference was found from these two means for each benefit level.

At a benefit level of zero is when reality of productivity is as close to ideal productivity as it ever will be throughout all 500 simulations. An 89% difference is as close to the ideal situation as reality will ever get, given current assumptions. The percent difference

between the productivity of reality and the ideal productivity only increases with each increased increment of benefit. At a benefit level of 100, there is about a 94% difference between the productivity levels. This is an increase in the difference between the productivity of reality and the ideal productivity by about 5%. While this is not an exponentially large increase in difference, it indicates that benefit level has a negative effect on productivity.

There is a large percent difference between the reality of productivity and the level of ideal productivity for it is highly improbable for an applicant to match on the first round. Therefore, when an applicant does match on their second highest skill on the eighth time period out of 50, there is no question the ideal productivity will be a much larger number. While it is impossible to ever reach ideal productivity, the point remains, productivity does decrease when the benefit level increase.

Figure 3: Matches per Time Period with a Benefit Level of 0

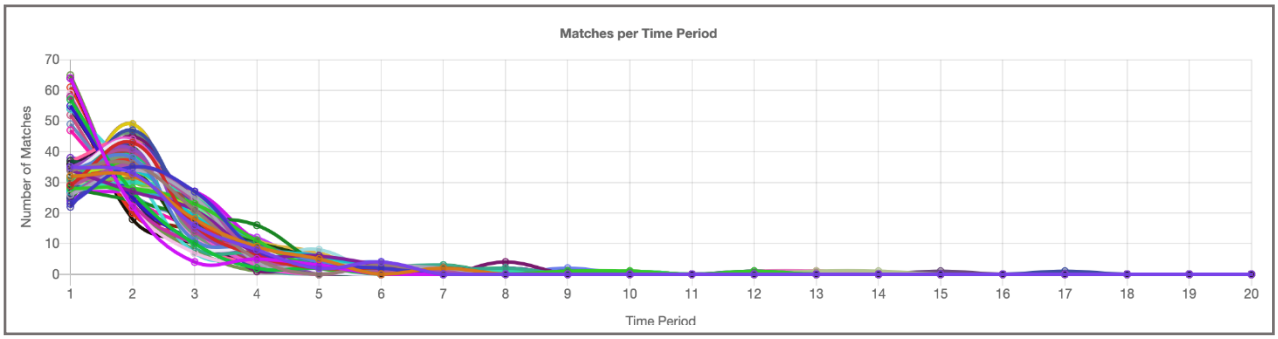


Figure 4: Matches per Time Period with a Benefit Level of 100

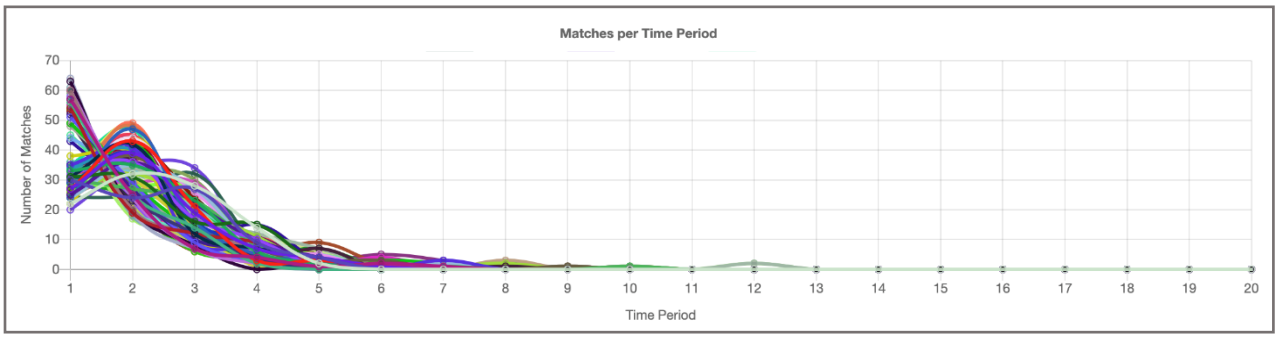
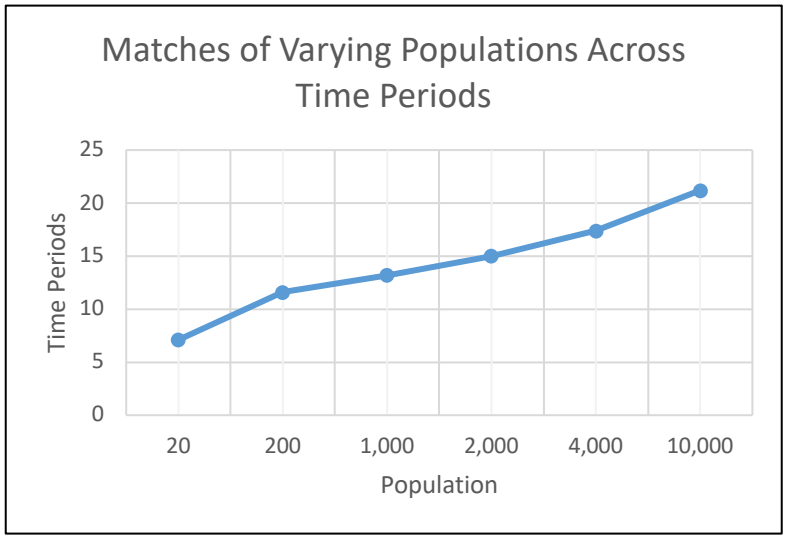


Figure 3 and Figure 4 display matches per time period for 100 simulations over 20 time periods for a population of 4,000 (2000 applicants and 2,000 employers). The difference lies in the benefit level. Figure 3 has a benefit level of zero and Figure 4 has a benefit level of 100. The different colored lines represent each of the 100 simulations. A benefit level of zero has been shown in Figure 1 and 2 to have the highest success rate and productivity. While a benefit level of 100 has shown to have the lowest success rate and productivity as shown in Figures 1 and 2. Therefore, to test if benefit level has an effect on how quickly the population matches with each other, benefit of levels zero and 100 were used.

However, Figures 3 and 4 look almost exactly the same. Nearly all of the population had found a match by time period 11. In both figures, the majority of people matched in time periods 1 through 4. Additionally, at time periods 8 and 12 there were small spikes of people matching in both Figures 3 and 4. This demonstrates that benefit level does not affect the duration of unemployment. If it did there would be a difference in the Figures. This would be the case for Figure 3 has a benefit level of zero, which was shown (in figures 1 and 2) to have the highest level of productivity and success rate and Figure 4 had a benefit level of 100, which was the benefit level with the lowest productivity and success rate. But as it is not, this indicates there is no correlation between benefit level and unemployment duration under the assumptions made in this study.

Figure 5: Last Time Period the Population Matched In



For each simulation run the time period was 25 with zero benefit. 100 simulations were run for each variation of applicants and employers. The mean of the longest time period it took for all possible matches to be made is the number graphed for each population increment.

Zero benefit was chosen for it had the highest productivity and success rates. Indicating more people match with zero benefit. In order to find what the last time period people matched in, it is crucial to choose the benefit level that has the highest success rate of matches to gauge how long the population is waiting before matching.

Figure 5 shows that with a population of 20 (10 applicants and 10 employers), all possible matches happen by about time period 7. When there is a population of 10,000 (5,000 applicants and 5,000 employers) the time period everyone is matched in is 21. That is 3 times the duration it takes the population to match when there is a population of 20. This indicates the higher the population looking for employment and seeking employees leads to longer durations of unemployment.

VI. Conclusion

Unemployment insurance benefit and duration was gauged in different simulations through use of this agent-based simulation model. Overall, this study found higher the benefit level, the lower productivity and the success rate. Benefit level does not affect unemployment

duration, but population does. This model, which also encompassed search theory was utilized to create data on a large scale that would provide valuable insight to the study of unemployment insurance. Simulations like this one are able to help the government find the optimum benefit level and duration for the unemployed. This would be an astronomical help to the economy if it were to fall into a recession or depression. Because knowing the correct amount of unemployment insurance and its duration would enable applicants to find jobs they are most productive at. This in turn would help the economy pick back up more quickly than before.

This study has found that success rates of matching an applicant and employer has been affected by benefit levels as shown in Figure 1: Success Rate by Benefit Levels. At zero benefit the success rate of matching was at just under 94%, then at a benefit level of 50 the success rate reached 94%; however, it exponentially declined from there. This indicates that not only does benefit level have an insignificant affect when benefit is at level 50 or less, but also from benefit level 75 and up, there is a negative correlation on the success rate of matching caused by the benefit level. In Figure 2: Productivity by Benefit Levels the percent difference in productivity has a positive correlation with benefit level. Indicating the percent difference between the reality of productivity and the ideal productivity is getting larger as benefit level increases. When benefit level is zero, there is a percent difference of about 89%, which is extremely large. This means that the ideal productivity is unrealistic and out of reach with current methods, including unemployment insurance. However, there is only a 5% increase in the percent difference across all 500 simulations. This finding is similar to Mortensen and Pissarides (1994) findings. They conclude that productivity is not a primary influencer of unemployment (Mortensen and Pissarides, 1994).

Figure 3: Matches per Time Period with a Benefit Level of Zero and Figure 4: Matches per Time Period with a Benefit Level of One Hundred indicate no affect of benefit level on unemployment duration. Zero benefit level has had the best results throughout this entire study and a benefit level of 100 has had the worst. This suggests if there was an affect by benefit level on the duration of unemployment it would be shown in Figures 3 and 4. But there is no difference between the durations. For Figure 5: Last Time Period the Population Matched In it was found that the lower the number of applicants, the lower the average number of time periods it took to make all possible matches. In other words, the higher the population (of applicants and employers seeking) who is in this pool, the longer it takes for applicants to get jobs and for employers to find the best employees for their business. This is strikingly similar to what Pissarides found in his 2013 paper *Unemployment in the Great Recession*. His reasoning for this is due to limited job creation and "the increase in unemployment is explained by higher durations" of unemployment (Pissarides, 2013).

During a recession lots of people are unemployed and they stay unemployed for longer than they would if the economy was not in a slump. This is because the more people who are unemployed, the longer the population stays unemployed. This also provides insight as to the reasoning the government had by extending the unemployment insurance during the recession by 73 weeks longer than average (Chikhale, 2017). Even though this study has found that unemployment insurance lowers the productivity of society and the success rate of matches, having lower unemployment insurance for a longer amount of time would not affect these two. The results indicated unemployment insurance at and below level 50 did not significantly affect the economy negatively. For the benefit level did not start to negatively affect the success rate till the benefit level was at 75 and the productivity level was only affected marginally by a

benefit level of 75. This indicates unemployment insurance is still able to be of help to the economy especially during recessions and depressions as long as the benefit level is at or below 50% of the maximum amount.

There are likely many flaws with this agent-based simulation model due to its assumptions and constraints. Ways to improve this study would be to find the optimum amount of benefit and its duration for each income level. Another way would be to find the exact amount of unemployment insurance benefit that would allow for high success rates while still having a productivity level significantly close to what the ideal productivity level is.

VII. Works Cited

Chikhale, Nisha. "The Importance of Unemployment Benefits for Protecting against Income Drops." *Equitable Growth*, 23 Apr. 2017, equitablegrowth.org/the-importance-of-unemployment-benefits-for-protecting-against-income-drops/.

Diamond, Peter A. "Mobility Costs, Frictional Unemployment, and Efficiency." *Journal of Political Economy*, vol. 89, no. 4, 1981, pp. 798–812. *JSTOR*, www.jstor.org/stable/1833035. Accessed 16 Apr. 2020.

Filiz, Elif S. "The Effect of Unemployment Insurance Generosity on Unemployment Duration and Labor Market Transitions." *Labour*, vol. 31, no. 4, Dec. 2017, pp. 369–393. *EBSCOhost*, search.ebscohost.com/login.aspx?direct=true&db=eoh&AN=1690667&site=eds-live&scope=site.

Holmlund, Bertil. "Unemployment Insurance in Theory and Practice." *The Scandinavian Journal of Economics*, vol. 100, no. 1, 1998, p. 113. *EBSCOhost*, search.ebscohost.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.3440764&site=eds-live&scope=site.

Mortensen, Dale T., and Christopher A. Pissarides. "Job Creation and Job Destruction in the Theory of Unemployment." *The Review of Economic Studies*, vol. 61, no. 3, 1994, pp. 397–415. *JSTOR*, www.jstor.org/stable/2297896. Accessed 16 Apr. 2020.

Mortensen, Dale T. "Unemployment Insurance and Job Search Decisions." *Industrial and Labor Relations Review*, vol. 30, no. 4, 1977, pp. 505–517. *JSTOR*, www.jstor.org/stable/2523111. Accessed 16 Apr. 2020.

Pissarides, Christopher A. “Unemployment in the Great Recession.” *Economica*, vol. 80, no. 319, 2013, pp. 385–403., www.jstor.org/stable/24029603. Accessed 16 Apr. 2020.

Titio, Boeri. “Chapter 13: Institutional Reforms and Dualism in European Labor Markets.” *Handbook of Labor Economics*, by Orley Ashenfelter and David Card, vol. 4B, North Holland, 2010, pp. 773–1823.

VIII. Appendix

My associate Zachary Copland has used the Python 3.7 code to create a sister system in JavaScript and HTML. It is a website that utilizes this code and turns it into an interactive site. For further information visit: http://zachcopland.com/labor-market-matching

The following is the Python 3.7 code used in for this model:

```
import random

total = 10
cols = 8
u = input("Enter monetary utility benefit. If not desired, type none.")
if u == "none":
    u = 0
else:
    u = int(u)
print()

# Format of Applicant 2d array
# [  A   B   C   D   E  Chosen Time period  Chosen Qual        (0 = false, 1 = true)
#   [00, 30, 80, 50, 55, 0, 0, -1],                    Applicant 1
#   [90, 60, 30, 30, 75, 0, 0, -1],                    Applicant 2
#   [55, 50, 75, 80, 85, 0, 0, -1],                    Applicant 3
# ]
# Format of S array
# [max quality for App1, max quality for App2, ..., max quality for App n]
# Format of Employer 2d array
# [Qual Attr wanted     Time period
#   [2(C),              0],                  Employer 1
#   [4(E),              0]                   Employer 2
# }

# Generate array of applicants
Skill = [[0 for x in range(cols)] for y in range(total)]
S = []


#Variables
n = int(input("Enter number of time periods"))
n+=1
```

```
timePeriodArray = [[0 for x in range(n)] for y in range(n)]
simulation = 100
Q = random.randint(10,90)
Q = Q/100
H = 1-Q
Max = 100
Min = 0
s1 = 0


def convertNumtoLet(num):
    if num == 0:
        return "A"
    elif num == 1:
        return "B"
    elif num == 2:
        return "C"
    elif num == 3:
        return "D"
    elif num == 4:
        return "E"
    else:
        return -1

def populateAppArray(newTotal):
    S.clear()
    global Skill
    Skill.clear()
    Skill = [[0 for x in range(cols)] for y in range(newTotal)]
    for i in range(newTotal):
        for j in range(cols):
            if j<cols-1:
                Skill[i][j] = random.randint(0,100)  #random qualities
            else:
                Skill[i][j-2] = 0  #none chosen at first
                Skill[i][j-1] = 0  #time period checker
                Skill[i][j] = -1  #Chosen quality
            j+=1
        S.append(max(Skill[i]))
    i+=1

Employers = [[0 for x in range(2)] for y in range(total)]

def populateEmpArray(newTotal):
    global Employers
    Employers.clear()
```

```
    Employers = [[0 for x in range(2)] for y in range(newTotal)]
    for i in range(newTotal):
        Employers[i][0] = random.randint(0,4)
        Employers[i][1] = 0

def getSecondHighestAttr(i):
    new_Skill = []
    for x in range(0,5):
        new_Skill.append(Skill[i][x])
    new_Skill.sort()
    second = new_Skill[-2]
    where = Skill[i].index(second)
    return where

def getThirdHighestAttr(i):
    new_Skill = []
    for x in range(0,5):
        new_Skill.append(Skill[i][x])
    new_Skill.sort()
    third = new_Skill[-3]
    where = Skill[i].index(third)
    return where

def ProbS(time):
    return threshold(time) / 100

def ProbA(time):
    return 1 - ProbS(time)

def AllPeriod(time):
    return ((100 + threshold(time)) / 2) * (n - time + 1)

def Benefit(time):
    if time == n:     # last time period
        return 0
    elif time + 1 == n:   # second to last time period
        return ((100 - threshold(time + 1)) / 100) * ((100 + threshold(time + 1)) / (n - time + 1))
    else:
        return ((ProbS(time+1) * Benefit(time+1)) + ProbA(time+1) * AllPeriod(time+1))

def threshold(time):
    if time in thresholdDict.keys():
        return thresholdDict[time]
    else:
        x = round(Benefit(time) / (n - time),2)
        thresholdDict[time] = x
```

```
        return x

def HighestOffer(time, skill):
    if skill >= threshold(time):
        return 0.2
    else:
        return 0

def VSearchA(time,ra):
    if time == n:
        return u
    else:
        highestSkill = max(Skill[ra])
        secondHighest = Skill[ra][getSecondHighestAttr(ra)]
        thirdHighest = Skill[ra][getThirdHighestAttr(ra)]

        valOfHighest = highestSkill * (n - time + 1)
        valOf2Highest = secondHighest * (n - time + 1)
        valOf3highest = thirdHighest * (n - time + 1)

        probHiOff = HighestOffer(time, highestSkill)
        prob2offer = HighestOffer(time, secondHighest)
        prob3offer = HighestOffer(time, thirdHighest)

        result = probHiOff * valOfHighest + prob2offer * valOf2Highest + prob3offer *
valOf3highest
        result += (1 - probHiOff - prob2offer - prob3offer) * (VSearchA(time + 1, ra) + u)
        result = round(result,2)
        return result


def outputAppArray():
    for i in range(total):
        i += 1

def outputVsearchA():
    for i in range(total):
        global s1
        s1 = S[i]          # finds max skill


def Match():
    global thresholdDict
    thresholdDict = {n: 0}
    global benefitDict
    benefitDict = {n: 0}
```

```
    suc = 0
    period = 1
    productivity = 0
    happiness = 0
    idealhappy = 0
    numOfMatches = 0
    periodReached = 0

    while period < n:
        ra = random.randint(0, total - 1)  # random applicant
        re = random.randint(0, total - 1)  # random employer
        end = 0
        matched = 0        # tells whether app & emp matched & for which attribute

        if time_checker(period) == 1 and period < n:
            timePeriodArray[simulation][period-1] = numOfMatches
            period += 1
            numOfMatches = 0

        while Employers[re][0] == -1 or Employers[re][1] >= period:  # Ensuring no replacement of
employers
            re = random.randint(0, total - 1)  # for new time period, new employer
            end += 1
            if end == total:
                break  # preventing infinite loop
        end = 0
        #print("2")
        while Skill[ra][5] == 1 or Skill[ra][6] >= period:  # Ensuring no replacement of Applicants
            ra = random.randint(0, total - 1)  # for new time period, new applicant
            end += 1
            if end == total:
                break  # preventing infinite loop
        #print("3")
        if Employers[re][1] == period or Skill[ra][6] == period:  # either Employer or Applicant (or
both) is/are on the current time period, skip
            continue
        elif Skill[ra][5] == 1 or Employers[re][0] == -1:
            continue  # prevents double matching

        # Set chosen Employer & Applicant to current time period
        Employers[re][1] = period
        Skill[ra][6] = period
        emp_threshold = threshold(period)

        if Employers[re][0] == Skill[ra].index(max(Skill[ra])):
```

```
            # employer random desired quality = max skill for random applicant
            if emp_threshold <= max(Skill[ra]) and VSearchA(period, ra) <= (max(Skill[ra]) * (n -
period + 1)):
                # employer threshold <= max skill of ra   and  vsearchA <= max skill * remaining time
periods
                matched = 1         # Matched on 1st highest
                numOfMatches += 1
        elif getSecondHighestAttr(ra) == Employers[re][0]:
            # employer random desired quality = second highest skill for random applicant
            if emp_threshold <= Skill[ra][getSecondHighestAttr(ra)] \
                    and VSearchA(period, ra) <= (Skill[ra][getSecondHighestAttr(ra)] * (n - period +
1)):
                # if employer threshold is less than or equal to second highest skill of random app and
if the VSearchA is less than or equal to the value offered for the 2nd highest
                matched = 2         # Matched on 2nd highest
                numOfMatches += 1
        elif getThirdHighestAttr(ra) == Employers[re][0]:
            # employer random desired quality = third highest skill for random applicant
            if emp_threshold <= Skill[ra][getThirdHighestAttr(ra)]\
                    and VSearchA(period, ra) <= (Skill[ra][getThirdHighestAttr(ra)] * (n - period + 1)):
                matched = 3         # Matched on 3rd highest
                numOfMatches += 1

        # Employer & Applicant matched on some attribute
        # (attribute indicated by numeric value in matched var)
        if matched != 0:
            Skill[ra][5] = 1  # Applicant matched
            if matched == 1:
                Skill[ra][7] = Skill[ra].index(max(Skill[ra]))
                happiness = Skill[ra].index(max(Skill[ra])) * (n -period + 1)
            elif matched == 2:
                Skill[ra][7] = getSecondHighestAttr(ra)
                happiness = Skill[ra][getSecondHighestAttr(ra)] * (n -period + 1)
            elif matched == 3:
                Skill[ra][7] = getThirdHighestAttr(ra)
                happiness = Skill[ra][getThirdHighestAttr(ra)] * (n -period + 1)
            Employers[re][0] = -1  # Remove Employer from pool
            suc += 1  # Increase successes
            productivity += happiness

    for j in range(1, period + 1):  # output success rate for each time period
        matches = 0  # initialize matches to 0 for each time period
        for k in range(total):  # find Applicants who matched during time period j
            if Skill[k][5] == 1 and Skill[k][6] == j:
                matches += 1
        srate = matches / total * 100  # Calculate success rate for time period j
```

```
        srate = round(srate, 2)

    for h in range(total):
        idealhappy += max(Skill[h]) * n    # (highest skill)(remaining periods)  compounded

    suc_rate = suc / total * 100
    suc_rate = round(suc_rate, 2)
    compare = (productivity / idealhappy) * 100
    compare = 100 - compare
    compare = round(compare, 2)
    return productivity, idealhappy, compare,suc_rate


def time_checker(period):          # to check if time period should be increased
    for i in range(total):
        if Skill[i][5] == 0:
            if Skill[i][6] != period:
                return 0
        if Employers[i][0] != -1:
            if Employers[i][1] != period:
                return 0
    return 1


compareVals = []
happyVals = []
prodVals = []
sucVals = []
for i in range(100):
    total = 10
    populateAppArray(2000)
    populateEmpArray(2000)
    productivity, idealhappy, compare, suc_rate = Match()
    compareVals.append(compare)
    happyVals.append(idealhappy)
    prodVals.append(productivity)
    sucVals.append(suc_rate)
    simulation += 1

print("values to plot")
print("Success rate:",sucVals)
print("Reality:",prodVals)
print("Ideal:",happyVals)
print("as close to ideal productivity %:",compareVals)
```