Assignment 3
MACS 30000, Dr. Evans
Haowen Shang

Question 1. Simulation in Sociology, Moretti (2002)

In the field of scientific research, computer simulation has had an important role in the development of theories of nonlinear and dynamic systems, and the use of computer simulation has created a new interdisciplinary scientific area of study of complex systems. In the field of sociology, Forrester defined system dynamics, widely used to define computer simulations in the social sciences, as a model to analyze the variables of a global system. Recently the technologies derived from artificial intelligence and the theories of self-organizing adaptive systems have furnished new types of models such as cellular automata, multiagent systems, and genetic algorithm.

However, a weakness of computer simulation remains: its connection with the empirical word. For a scientific simulation, validity and realism are essential. Simulation model's main function is to determine the consequences regarding the changes of some variables or assumptions, but it cannot value if the model represents reality. The model does not tell us anything about the connection between theory and empirical data.

With regard to multiagent systems, the potential weaknesses of validity are shown as follows. Firstly, multiagent systems use of theories and models of rationality that are not so realistic, understandable, and can not be applied in the case of limited knowledge. In particular, theories of rationality need to be extended to learning and adaptation. Secondly, multiagent systems do not take all the aspects of psychological theories (emotions, motivations, desire, intent, consciousness) into consideration in formalization. Thirdly, one of the principal challenges in the development of multiagent systems is formalization of knowledge. The question whether it is indeed possible to formalize all types of knowledge—for example, common sense knowledge—and, in this case, what would be the best formalization remains to be solved.

With respect to cellular automata, the potential weaknesses of validity are shown as follows. Firstly, a limitation of cellular automata is the use of synchronous updating of states; we assume that there is a global clock according to which all cells are updated simultaneously. This assumption may not be found in real social processes, because individuals modify their attitudes and opinions at different moments. Secondly, an important limitation regards the restrictions imposed by spatial structures, establishing that each individual interacts only with a subset of the whole population. This type of restriction may be acceptable, because it is impossible for a person to interact with all the individuals in a population. However, it is very difficult to define the neighborhood of a unit. In the real world, interactions can also take place among individuals who are not "physically" close to one another and the neighborhood can change over time. When we model cellular automata, we must consider these aspects if we want the model to be plausible.

The system dynamics of Forrester clearly derive from some concepts expressed by cybernetics, such as the concept of feedback. Dynamic feedback is a key characteristic that computer simulation is good at modeling. Dynamic feedback is where some initial stimulus changes behavior, and then that change in behavior creates new stimuli which in turn cause further behavioral change.

For example, Forrester analyzed the relationship between decisions and policies. Policies are the rules that determine the making of decisions. If one knows the policy governing a point in a system, one then knows what decision will result from any combination of information inputs. Depending on the decisions made by people and their following behaviors, policy makers might search for a better set of policies that yield improved results. One builds a simulation model from policies that in turn make decisions. The model generates streams of decisions controlled by policies built into the model. System dynamics is most useful for understanding how policies affect behavior. Emphasis should be on designing policies that will yield systems with more favorable behavior. The policies make all the decisions step-by-step in time as the simulation unfolds. Then, if the resulting behavior is undesirable, one searches for a better set of policies that yield improved results. (Forrester, 1998)

For dynamic feedback in political science, a potential research question is “What is the effect of international collaboration on the importing tax rate?” We know that a country decided the tax rate when import commodities from other countries. When two counties collaborate in some fields and decide to decrease the tax rate on some special commodities, which in turn improve the collaboration between two countries. For example, China and Russia collaborate on agriculture and decrease the tax of agricultural commodities importing from each country, which in turn makes two countries create more job opportunities and production on agriculture and thus they collaborate more on agriculture.

Assignment 3 _Haowen Shang

October 22, 2018

1 Assigment 3

1.0.1 MACS 30000, Dr. Evans

1.0.2 Haowen Shang

Due Wednesday, Oct. 24 at 11:30 AM

1. Simulation in Sociology, Moretti (2002) Please see PDF version.

2. Simulating the income

```
In [1]: # Import initial packages
        import numpy as np
        import matplotlib.pyplot as plt
        from matplotlib.ticker import MultipleLocator
```

(a) Simulate the lifetime income. Plot one of the lifetime income paths.

```
In [2]: def income_sim(p):
            '''
            Requires a simulation profile, structured as a dictionary

            p = {
                'inc0': 80000,              #average starting income
                'P': 0.4,                       #positive dependence of today's income on
                                                    last period's income
                'g': 0.025,                     #long-run annual growth rate of income
                'st_year': int(2020),         #start year
                'work_years': 40,           #years to work
                'a': 0,                             #mean of the log of the error term
                'Sigma' :0.13,                #standard deviation of the log of the error term
                'num_draws': 10000       #simulations
                }
            '''

            #set random seed
            np.random.seed(524)
```

```
        log_errors = np.random.normal(p['a'], p['Sigma'], (p['work_years'],
                                                       p['num_draws']))

        #create a matrix of dimension (work_years, num_draws)
        ln_inc_mat = np.zeros((p['work_years'], p['num_draws']))

        #fill the matrix
        ln_inc_mat[0, :] = np.log(p['inc0']) + log_errors[0, :]

        #loop and apply model
        for yr in range(1, p['work_years']):
            ln_inc_mat[yr, :] = ln_inc_mat[yr, :] = ((1 - p['P']) * (np.log(p['inc0']) +
                                    p['g'] * (yr)) + p['P'] * ln_inc_mat[yr - 1, :] +
                                    log_errors[yr, :])

        #translate the log value back into income
        inc_mat = np.exp(ln_inc_mat)
        return inc_mat
```

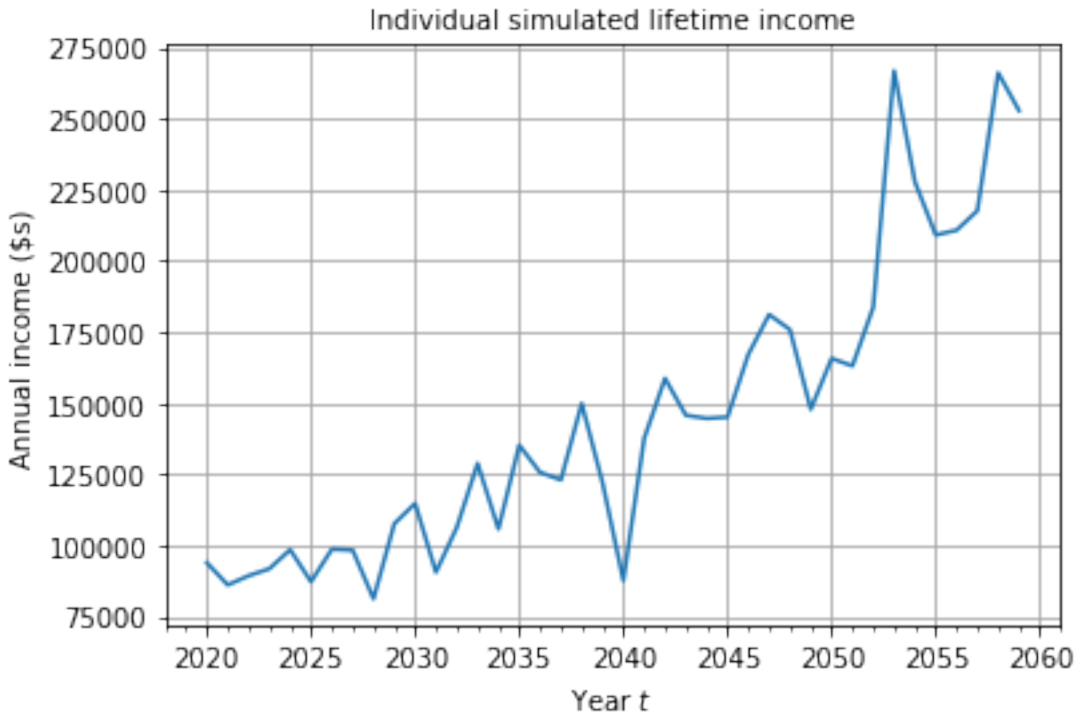
```
In [3]: simulation_profile = {
                'inc0': 80000,              #average starting income
                'P': 0.4,      #positive dependence of today's income on last period's income
                'g': 0.025,                 #long-run annual growth rate of income
                'st_year': int(2020),       #start year
                'work_years': 40,          #years to work
                'a': 0,                          #mean of the log of the error term
                'Sigma' :0.13,              #standard deviation of the log of the error term
                'num_draws': 10000      #simulations
                }

        inc_mat = income_sim(simulation_profile)
        print(inc_mat)
```

```
[[ 66409.15585396  98274.13534194 101939.81109509 ...  98720.39690442
   72404.51636886  68710.32820307]
 [ 80020.53020329  67383.19350738  84557.85626308 ...  68247.7770509
   74518.33613244  80555.96068584]
 [ 75805.26636606  66134.42494243  91458.20304692 ...  67268.53350159
   90012.42673528  80645.62355527]
 ...
 [272690.56519108 217821.73027242 184724.24512469 ... 159922.45424852
  253961.68337673 209741.55004062]
 [231539.17420799 202509.15149494 197955.96626493 ... 199502.43481758
  210951.71828579 205420.27946389]
 [197895.95201384 165115.10025278 172644.86927513 ... 248654.44847819
  234237.14656466 221566.29879732]]
```

```
In [4]: #plot the histogram
        %matplotlib inline
        p = simulation_profile
        year_vec = np.arange(p['st_year'], p['st_year'] + p['work_years'])
        individual = 500
        fig, ax = plt.subplots()
        plt.plot(year_vec, inc_mat[:, individual])
        minorLocator = MultipleLocator(1)
        ax.xaxis.set_minor_locator(minorLocator)
        plt.grid(b=True, which='major', color='0.65', linestyle='-')
        plt.title('Individual simulated lifetime income', fontsize=10)
        plt.xlabel(r'Year $t$')
        plt.ylabel(r'Annual income (\$s)')
```

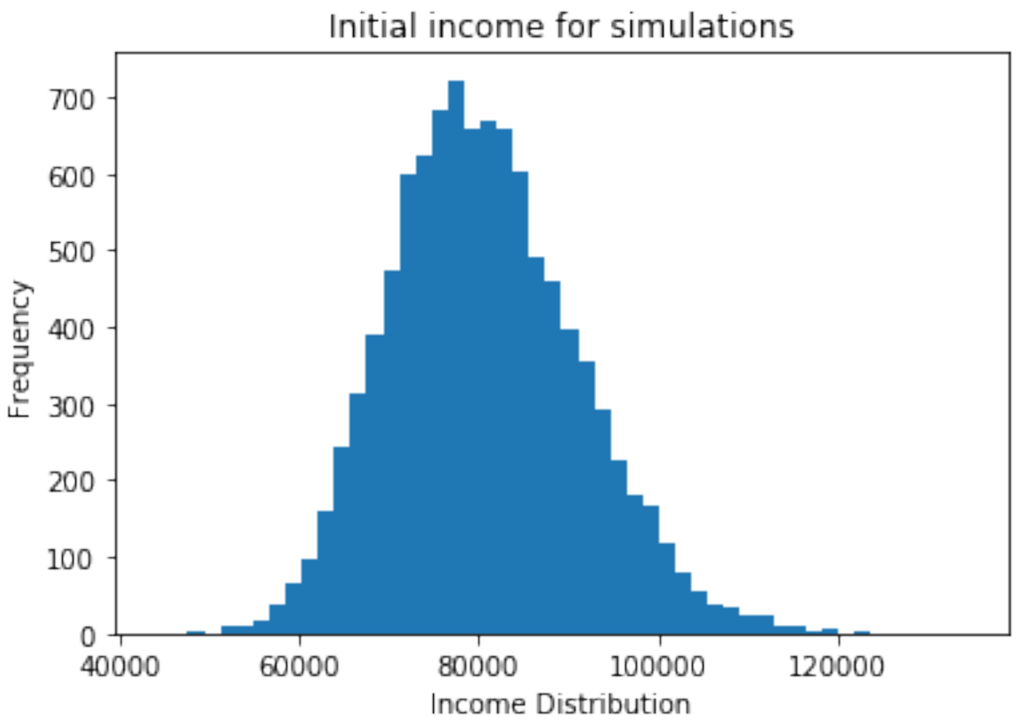
```
Out[4]: Text(0,0.5,'Annual income (\\$s)')
```



(b) Plot a histogram and analyse

```
In [5]: plt.hist(inc_mat[0,:], bins=50)
        plt.xlabel("Income Distribution")
        plt.ylabel("Frequency")
        plt.title("Initial income for simulations")
```

```
Out[5]: Text(0.5,1,'Initial income for simulations')
```



```
In [6]: # calculate the percent of the class will earn more than $100,000
        #in the first year out of the program
        inc_more = (inc_mat[0, :] > 100000).sum() / 10000
        print("The percent of the class will earn more than $100,000 in the first year \
                 out of the program is {:.2f}%".format(inc_more*100))
```

```
The percent of the class will earn more than $100,000 in the first year out of the program is 
```

The percent of the class will earn more than $100,000 in the first year out of the program is 4.17%.

```
In [7]: # calculate the percent of the class will earn less than $70,000
        #in the first year out of the program
        inc_less = (inc_mat[0, :] < 70000).sum() / 10000
        print("The percent of the class will earn less than $70,000 in the first year \
                   out of the program is {:.2f}%".format(inc_less*100))
```

```
The percent of the class will earn less than $70,000 in the first year out of the program is 1
```

The percent of the class will earn less than $70,000 in the first year out of the program is 15.12%.
The distrubution is slightly right-skewed, but overall the distrubution is approximately normal.

(c) years to pay off 95,000 of zero-interest debt with initial income = 80000

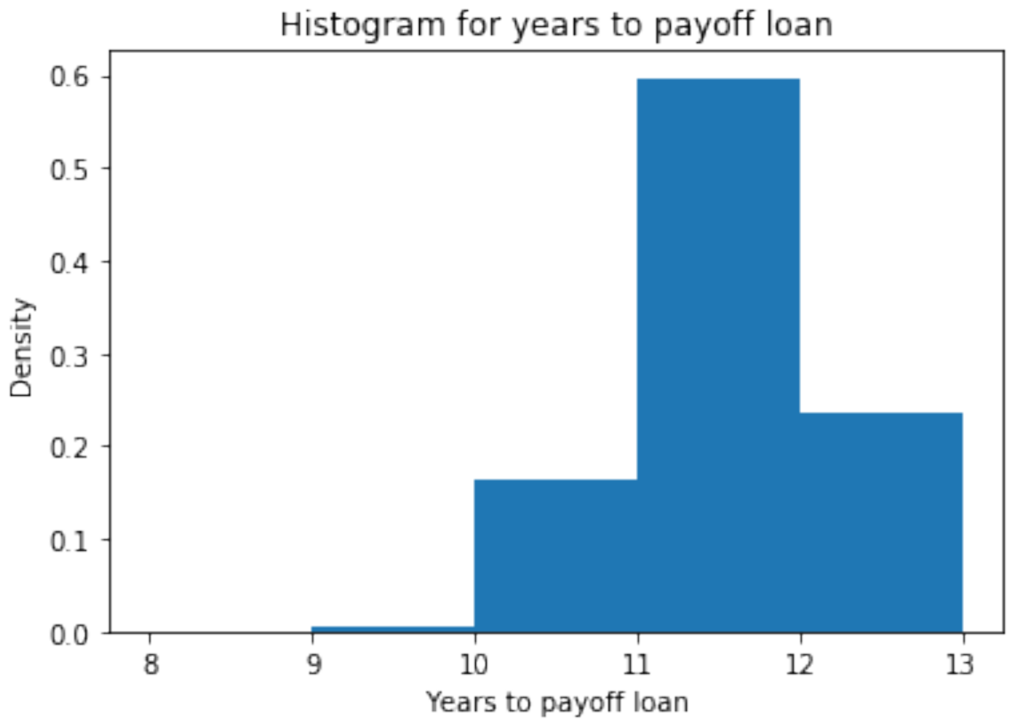
```
In [8]: #Creating a matrix create a matrix of payoff (dimension (num_draws, work_years))
        payoff_mat = np.zeros((p["num_draws"], p["work_years"]))

        #loop and fill the matrix
        for yr in range(0, p["work_years"]):
            payoff_mat[: , yr] = inc_mat[yr, :] * 0.1

        #create a list of how many years it takes to pay off the loan in  10,000 simulations
        payoff_yr = []
        for row in payoff_mat:
            yr = 0
            total = 0
            for i in row:
                yr = yr +1
                total = total + i
                if total >= 95000:
                    payoff_yr.append(yr)
                    break

        #plot the histogram
        plt.hist(payoff_yr, density = True,
                  bins = np.arange(min(payoff_yr) - 1, max(payoff_yr) + 1))
        plt.xlabel(r'Years to payoff loan')
        plt.ylabel(r'Density')
        plt.title(r'Histogram for years to payoff loan')

Out[8]: Text(0.5,1,'Histogram for years to payoff loan')
```



```
In [9]: #calculate the percent of the simulations who is able to pay off the loan in 10 years
        count = 0
        for yr in payoff_yr:
            if yr <= 10:
                count = count + 1
        percentage = count/10000
        print("percent of the simulations of people who is able to pay off the loan \
                  in 10 years is {:.2f}%".format(percentage*100))
```

```
percent of the simulations of people who is able to pay off the loan in 10 years is 16.78%
```

The percent of the simulations of people who is able to pay off the loan in 10 years is 16.78%.

(d) years to pay off 95,000 of zero-interest debt with initial income = 90000

```
In [10]: #modify the income matrix
         new_simulation_profile = {
                   'inc0': 90000,             #average starting income
                   'P': 0.4,       #positive dependence of today's income on last period's incom
                   'g': 0.025,                    #long-run annual growth rate of income
                   'st_year': int(2020),       #start year
                   'work_years': 40,          #years to work
                   'a': 0,                          #mean of the log of the error term
```

```
                 'Sigma' :0.13,           #standard deviation of the log of the error term
                 'num_draws': 10000      #simulations
                 }

        inc_mat_modified = income_sim(new_simulation_profile)
        print(inc_mat_modified)

[[ 74710.30033571 110558.40225969 114682.28748197 ... 111060.44651748
   81455.08091496  77299.11922846]
 [ 90023.0964787   75806.0926958   95127.58829597 ...  76778.74918227
   83833.128149    90625.45577157]
 [ 85280.92466182  74401.22806023 102890.47842778 ...  75677.10018929
  101263.9800772   90726.32649968]
 ...
 [306776.88583997 245049.44655647 207814.77576527 ... 179912.76102958
  285706.89379882 235959.2437957 ]
 [260481.57098399 227822.7954318  222700.46204805 ... 224440.23916978
  237320.68307151 231097.81439688]
 [222632.94601557 185754.48778437 194225.47793452 ... 279736.25453796
  263516.78988524 249262.08614698]]
```

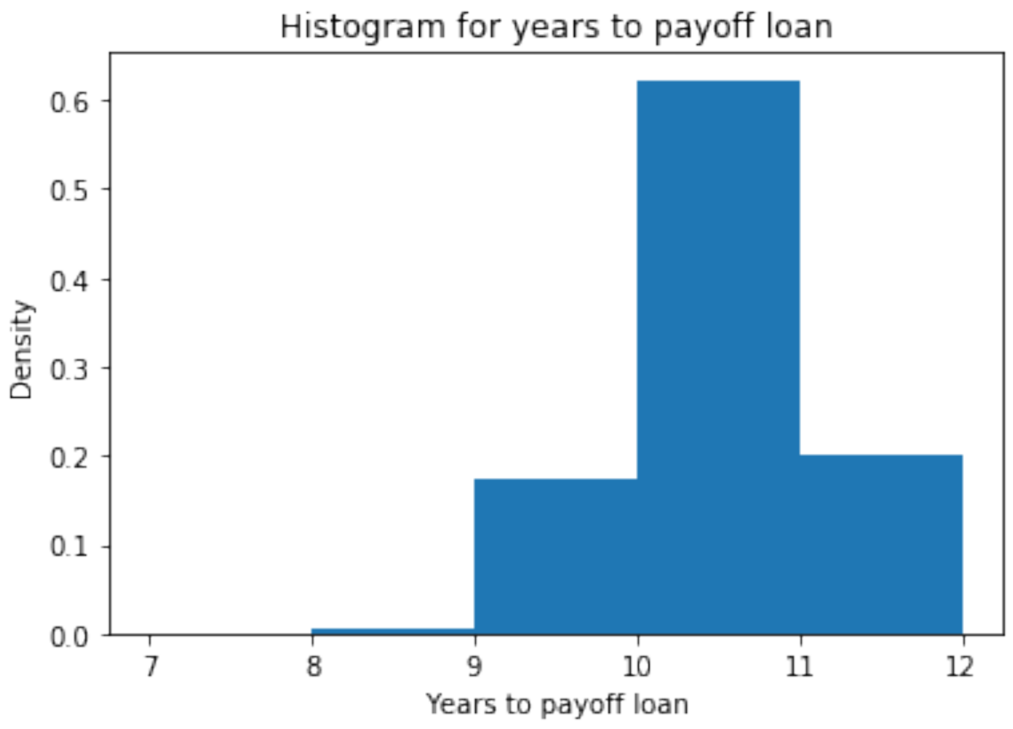
```
In [11]: #Creating a matrix create a matrix of payoff (dimension (num_draws, work_years))
         payoff_mat_modified = np.zeros((p["num_draws"], p["work_years"]))

         #loop and fill the matrix
         for yr in range(0, p["work_years"]):
             payoff_mat_modified[: , yr] = inc_mat_modified[yr, :] * 0.1

         #create a list of how many years it takes to pay off the loan in 10,000 simulations
         payoff_yr_modified = []
         for row in payoff_mat_modified:
             yr = 0
             total = 0
             for i in row:
                 yr = yr +1
                 total = total + i
                 if total >= 95000:
                     payoff_yr_modified.append(yr)
                     break

         #plot the histogram
         plt.hist(payoff_yr_modified, density = True,
                  bins = np.arange(min(payoff_yr_modified) - 1, max(payoff_yr_modified) + 1))
         plt.xlabel(r'Years to payoff loan')
         plt.ylabel(r'Density')
         plt.title(r'Histogram for years to payoff loan')

Out[11]: Text(0.5,1,'Histogram for years to payoff loan')
```



```
In [13]: #calculate the percent of the simulations who is able to pay off the loan in 10 years
         count = 0
         for yr in payoff_yr_modified:
             if yr <= 10:
                 count = count + 1
         percentage = count/10000
         print("percent of the simulations of people who is able to pay off the loan \
         in 10 years with initial salary of 90000 is {:.2f}%".format(percentage*100))
```

```
percent of the simulations of people who is able to pay off the loan in 10 years with initial 
```

The percent of the simulations of people who is able to pay off the loan in 10 years with initial salary of 90000 is 79.92%.