Dual Labor Market: an Agent-Based Model Representation

Alessandro Taberna
Università degli studi di Torino

February 16, 2019

Abstract

This model presents an agent-based model approach for a dual labor market populated by households and firms, which interact following a basic economic cycle. This work focuses on the observation of some macroeconomic variables such as wages and unemployment rate, price and GDP over a fixed period of time. In particular, the simulation experiments carry out the importance of skill matching between demand and supply in the labour market and how some categories are likely to gain and some to loose by eventual differences. Simulations over a wide set of parameters shows price stability and a plausible employment rate oscillation.

1 Introduction

Agent-Based Model (ABM) is a formal tool that through computational modelling is able to observe interactions among entities over time, namely agents. As computational model, we mean a model that takes certain inputs and generates outputs through algorithmic way. The agent is the individual element of our computations and it has own properties, states and actions.

ABMs are then powerful because they allow more interaction between user and its representation since they are easier to modify, dynamic and executable. Moreover, they are easier to understand even for people without a strong computational background and this happens because they use simple rules of individual behaviour closer to natural language and our natural thinking.

We can use ABMs for any natural phenomenon, but they are particularly useful for problems with heterogeneous agents, especially when this heterogeneity affects overall result of the model. For example, in our model, each household has different preferences about its minimum wage and each firm can decide its own salary that is different from the others. Moreover, we have not to keep track of each single agent characteristics once defined its properties. On the contrary, models with a large number of homogeneous agents usually do not need to bear the full power of ABMs, but detailed equations can better describe the interactions among individuals.

Furthermore, ABMs are more useful in the presence of adaptive agents since they allow individuals to keep a history of interactions and thus change their behaviour based on that. For example, we define in our model that households will decrease their minimum wage if they are not able to find a job for some periods, and at the same time firms change their offered wage based on the labour market conditions. Complex interactions are thus easier to model through and ABM.

The rich conception of time is another advantage of ABMs that allows a more detailed collection of information than equation-based models. This happens because in ABMs the interactions among agents follow a precise order helping the user to go beyond a static snapshot of the system and toward a dynamic evolution of the system's behaviour. In another words, we can get rich and detailed information of the process step by step and not just the final state. For instance, we can see how many households change their working status each period and not just the final rate of unemployment.

Additionally, there are models with equations very difficult if not impossible to solve. In this case, ABMs are a key tool that allows to see results not reachable otherwise. Even build a ABM can be computationally intensive, for example simulate a system with thousands or millions of individuals can require great computing power. There is then a trade-off depending on the precise model we want to implement that affects our choice regarding which one is better to use.

One of the most powerful tools of ABMs is their ability to determine causality. Since the degree of freedom of ABMs is just limited by the available knowledge of the interest phenomenon and not by the formalism they adopt, it is possible to model a more granular representation of reality. The ability to model realistically a phenomenon plays a crucial role

in understanding what are the main mechanisms behind the phenomenon itself, and in which manners they interact during the whole process.

ABMs result particularly useful in economic analysis, where there is still a strong mismatch between theoretical and empirical literature. The problem is that the main theoretical contributions follow the EBMs approach but, due to the complexity of the topics underlined, they often require a strong formalism and they are thus less realistic. It goes the other way with the economical empirical analysis that since usually is not matching with the traditional theory, it is considered inadequate and thus not deployed. The data difference between expected theoretical results and the empirical ones is due to fact that mainstream economic research is still based on the "as-if" principle rendering EBMs incapable to capture the algorithmic nature of behavioural data, giving up a certain degree of realism.

Instead, ABMs, can overcome this trade-off in theoretical and empirical literature by allowing the use of empirical knowledge in theoretical analysis and by helping researches addressing the main questions asked by the contemporary world. Another reason of ABMs power in economic research is their ability to address causality, for example about the developing of policy analyse, it is possible to investigate polices weak spots, their causes and not only its final result. In other words, it is possible give support and suggestion in terms of intervention to decision-makers. Furthermore, the use of heterogeneity can be very helpful in economic analysis, for instance, the traditional economic theory makes a broad use of the representative agents, attributing different behaviour to firms and consumer but they do not allow heterogeneity among agents. In other words, a representative consumer will interact with a representative firm but in which reality all consumers or firms are identical?

In conclusion ABMs have all the characteristics to improve economic both theoretical and empirical analysis by allowing a more realistic representation of phenomena.

2 The Model

2.1 Overview

In this model, there are two different type of agents:

- firms;
- households.

Furthermore, households are divided in high-skill and low-skill and they interact with firms following basic economic rules.

Households, when employed, receive a wage from the firms and consume a fraction of it; otherwise, they get an unemployment benefit entirely spent over each period. The total consumption of employed and unemployed households defines the aggregate demand for goods. This, divided by the total production of firms determines the market price.

Firms produce only one good and their amount depends on the number of workers and their skills. Indeed, high-skill workers have a higher productivity as well as higher wages than low-skill ones. Firms, for each period, looking at their previous performance contract or expand their workforce hiring new workers or firing existing ones. Nevertheless, if profits overcome a certain threshold a new firm is created, while, on the other hand, firms go bankrupt due to heavy losses. Moreover, for each period, firms adjust their wages based on the availability of unemployed households. At the same time, households' minimum wage decrease if they are not able to find a job for some periods.

2.2 Initial conditions

At first, the model works by creating a defined number of firms and households and a certain number of them is immediately connected. Each firm can recognize its workers and each worker knows which firms is working for. In this section, we examine the initial agents' characteristics and first interactions.

2.2.1 Households

Households' main attributes initialization are the following:

```
  to setup-households                                    ;
create-households nHouseholds
[
  set myFirm nobody
  ifelse random-float 100 > highSkilledHouseholdOnPopulation
  [ set householdType 1
    set color green
    set minWage random-normal avMinWageLowSkill stDevMinWageLowSkill ]
  [ set householdType 0
    set color red
```

```
      set minWage random-normal avMinWageHighSkill stDevMinWageHighSkill ]

  ]
end
```

We can set the number of households living in our world through `nHouseholds`, once settled all households start as unemployed. The most important attribute of this agent class is its skill level that can be high or low. The average percentage of high skilled households among the total population depends on the value of `highSkilledHouseholdOnPopulation` set before running the model. Once households are assigned to their skill level, they set their minimum wage. The minimum wage is different for each agent; it belongs to a normal distribution with mean `avMinWageLowSkill` and standard deviation `stDevMinWageLowSkill` for low-skill, and mean `avMinWageHighSkill` and standard deviation `stDevMinWageHighSkill` for high-skill, all these value can be set in a different way to examine the model's evolution.

2.2.2 Firms

The next step is to create firms, through the following procedure:

```
    to setup-firms
  create-firms nFirms
  [
    set highWageAmount random-normal avHighWageAmount stDevHighWageAmount
    set lowWageAmount  random-normal avLowWageAmount stDevLowWageAmount
    set hireTot round random-normal averageWorkers stDevAverageWorkers
    set hireHigh  round ( hireTot * highSkilledWorkersRatio / 100 )
    set hireLow hireTot - hireHigh
  ]
end
```

As for households, we can decide the number of firms initially operating in this economy through `nFirms`. The main features for firms' initialization are the wage setting, how many workers they want to hire and how they split them between high and low skill workers. More in details:

- `highWageAmount` and `lowWageAmount` are the wages offered by the firm respectively to its high-skill and low-skill workers as well as to the unemployed households available in the labor market and belonging to the same skill level. Each firm offers different wages with respect to each other, the values value are set following a normal distribution with mean `avHighWageAmount` and standard deviation `stDevHighWageAmount` for high skilled workers, the same happens for low skilled ones;
- the number of workers that each firm wants to hire during the current period is `hireTot`. Initially, the amount of workers is set as previous variables following a normal distribution,

so that each firm needs a different number of workers. Thereafter, `hireTot` will be set based on the firm's profits with respect to its production;

- each round of assumption firms need to decide how divide the whole number of workers they want between high and low skill. In order to do that the variable `highSkilledWorkersRatio` sets the percentage of high-skill workers out of the total, the residual amount will be the number of low-skill workers. The value of these variables is flexible, we can then change it before or during each run and it might significantly change model's results.

2.2.3 Hiring

Once that we have created firms and households we need to connect the firms to their workers. This happens through the procedure `hiring`, we now analyse how it works for high skill household but it is specular for low skill ones:

```
if profits > 0 [
  let potentialHighWorkers count households with [
   myFirm = nobody and
   minWage <= [ highWageAmount ] of myself
   and householdType = 0
   ]
  let HH min list hireHigh potentialHighWorkers
  ifelse HH = hireHigh
  [ set gotAllHighWorkers gotAllHighworkers + 1 ]
  [ set gotAllHighWorkers 0 ]
  ifelse HH < hireHigh
  [ set notEnoughHighWorkers NotEnoughHighWorkers + 1 ]
  [ set NotEnoughHighWorkers 0 ]
  if profits > 0 and potentialHighWorkers > 0 [
    ask n-of HH households with [
     minWage <= [ highWageAmount ] of myself and
     householdType = 0 and
     myFirm = nobody]
    [ set myfirm  myself
      set myWage [ highWageAmount ] of myself
      set unemployedTime 0
    ]
   ]
  ]
```

This procedure works whenever a firm had positive profits in the previous period and wants to hire new workers in the current one. Since the model has not started yet we assume that firms start with a small positive profits when created, such as small initial investment, in order

to join the cycle. Furthermore, the variable potentialHighWorkers identifies all the firm's potential high skill workers through the fulfillment of three requirements:

- they must be unemployed;
- they have to belong to the skill level required;
- they must have a minimum wage less than the one offered by the firm.

Now the firm knows how many workers would accept its offer, then it chooses the minimum number between the number of workers it wants to hire, (previously defined hireHigh), and the ones available in the market. If the workers available to accept firm's offer in the labor market are less than the number wanted by the firm, it will hire as much as it can. The variable notEnoughHighWorkers tracks that for the current period the firm is not able to find entirely all workers it needs. On the contrary, if the workers available in the market are more or equal the number searched by the firm, it can hire the entire amount hireHigh. In this case, the variable gotAllWorkers records that for this period the firm found all the workers needed. The variables notEnoughHighWorkers and gotAllHighWorkers record wheter the firm is able or not to find all the workers it wants during the evolution of the model. This tracking allows each firm to adapt itself to the market conditions increasing or decreasing the wage offered to its workers. The last step is to link firms and households, each firm hires the final number of unemployed households, in this case LL, by setting their variables myFirm equal to its number and myWage equal to its wage. The workers are then able to recognize the firm they are working for. Firms follow exactly the same procedure for low-skill workers (see appendix A).

2.3 First cycle

Once that households and firms are set and linked to each other, the first cycle of the model can begin.

2.3.1 Agents' adaptation

Firms and households are adaptive agents, this means that they change their behaviour based on the evolution of the model. For example, as we showed before in the procedure, firms use the variable gotAllLowWorkers and notEnoughLowWorkers to record whether they hire all the low-skill workers they need or not, in the event they are not enough in the labor market, the same happens for high-skill workers. Furthermore, firms look at the value of these variables and modify their wages. For instance:

```
if gotAllHighWorkers = 4 [ set highWageAmount highWageAmount * 0.9 ]
```

Means that a firm will offer only the 0.9 of the previous wage to its high-skill employees and potential workers, if for three periods in a row it is able to hire all the high-skill workers it needs from the market. On the contrary, a firm will increase its wages if it is not able to find the whole amounts of workers it wants for some consecutive periods:

```
if notEnoughLowWorkers = 3 [ set lowWageAmount lowWageAmount * 1.1 ]
```

Moreover, the wage adaptation to market conditions can follow a different or even inverse path for high and low-skill workers. Indeed, it can happen that just one skill type is scarce in the market while the other is abundant, bringing to a wage increase for one skill and a decrease for the other one. The same do households, they change their minimum wage if they are not able to find a job for some periods using `changeExpectations` :

```
    to changeExpectations

  ask households  [
    if myFirm = nobody [ set unemployedTime unemployedTime + 1 ]
    if minWage > subsidyAmount [
     if unemployedTime = 3 [ set minWage 0.9 * minWage ]
     if unemployedTime = 4 [ set minWage 0.9 * minWage ]
     if unemployedTime >= 5 [ set minWage 0.8 * minWage ]
     ]
    ]
  end
```

When a household is unemployed it increases by one its variable `unemployedTime`. For instance, when `unemployedTime` is equal to 3, it means the household was unemployed for three periods in a row and its minimum wage starts to decrease.

For both households and firms, we set some thresholds to their minimum wage and wage offered. For instance, households never decrease their minimum wage below or equal to the subsidy amount and firms will never offer less. This in order to stay as closer as possible to reality, indeed a person will never accept a job if it can take the same of money without working. We set even an upper threshold to firms' wages because having fixed productivity for each skill type a wage amount too high is not consistent.

By this representation agents are able to adapt their behaviour to the market conditions and model evolution, getting a better representation of how they behave in the real world.

2.3.2 Wages

The Firms identify their workers by counting the number of households for each skill level which have `myFirm` equal to their identification number `myself`:

```
    set nHighEmployee count households with
    [ myself = myFirm and  householdType = 0 ]
    set nLowEmployee  count households with
    [ myself = myFirm and  householdType = 1 ]
```

Thereafter each firm sets the total amount of wages to be paid, `myWages`, equal to the total amount of its employees multiplied by its wages:

```
set myWages ( lowWageAmount * nLowEmployee ) +
            ( highWageAmount * nHighEmployee )
```

Moreover, we set several colours depending on the firm's size, its total number of employees, in order to get a better representation in the interface and help the user.

2.3.3 Subsidies

In the next step we set the quantity of subsidies to be paid given the present market conditions, in this world firms bear the cost of subsidies for unemployed households by paying a fixed amount for each worker they have. To do that we use the following methods:

- by counting all unemployed households, so ones with `myFirm = nobody`, then multiplying it by the `subsidyAmount` set at the beginning;
- `single subsidy` simply divides `generalSubsidy` by the total number of workers in order to get the subsidy per worker;
- now that the subsidy per worker has been established, each firm sets its own total amount as:
  ```
  set mySubsidies singleSubsidy *
  ( nHighEmployee + nLowEmployee )
  ```

2.3.4 Production

After firms paying their subsidies and wages, they can set their production. The production of each firm depends on the number of employees and their productivity, which depends on skill level and can be set at the beginning or changed during the run, it is then equal to:

```
set production   ( nHighEmployee * productivityHigh +
                   nLowEmployee * productivityLow )
```

2.3.5 Households

The previous part of the model was more about firms functioning; now we are focusing a bit on the households' side to create the demand for goods and close the cycle.

First of all, households need to recognize if they are unemployed and this is done with the following command:

```
ifelse myFirm = nobody  [ set mySubsidy subsidyAmount ]
                        [ set mySubsidy 0 ]
```

In this model workers consume a fraction of their wage while unemployed households consume the subsidy entirely. The value of consume is a parameter set in the interface and can be modified during the model's evolution. We mainly assumed that employed households

consume 90% of their wage. The `householdsDemand` is then built by summing all the wages of households multiplied by their consume plus the sum of all the subsidies of each perdiod t:

$$D_t = (W_{ht} * c) + (W_{lt} * c) + S_t \tag{1}$$

Where:

- S is total subsidies amount
- W_h is the wage aggregation of high skill workers
- W_l is the wage aggregation of low- skill workers
- c is consumption parameter

2.3.6 Price

The economy, at this point of t, has a supply which is the firms production, and a demand which is the total consume of households, for goods.

We can match them and get the price that is a clearing price since no warehouses and stocks are assumed.

$$P_t = \frac{D_t}{Q_t} \tag{2}$$

Where Q is the sum of all firms production.

2.3.7 Profits

We evaluate firms' performance in order to see if they can close the current period with profits or losses. To have heterogeneity among firms' prices we add an exogenous shock to each firms price in the following way:

```
ask firms  [set myPrice price + random-normal 0 0.1 ]
```

So that each firm adds to its price a value belonging to a normal distribution with mean 0 and variance 0.1. Firms use the variable `myPrice` to calculate their revenues multiplying it with production:

```
set revenues myPrice * production
```

To get profits the firms need to calculate even their costs as the sum of wages and subsidies and subtract them to revenues:

```
set costs myWages + mySubsidies
set profits revenues - costs
```

2.4 Normal cycle

During the normal cycle, `ticks > 0`, firms and households change their behaviour and take decision based on what happened in the previous periods. In particular, they decide how many workers they want to hire or fire, if close the firm or open a new one.

2.4.1 Workers turnover

In this model, a firm hires new workers if in the previous period it had positive profits. The procedure `wantedWorkers` sets how many workers it wants to hire and in which ratio between high and low skill. Moreover, we decided to set the number of workers on the ratio between profits and production, for instance:

```
if profits / production > 0.2 [ set hireTot  3 ]
if profits / production > 0.3 [ set hireTot  4 ]
```

Means that if firm's profits are more than 20% of the production it will try to hire 3 additional workers during this period while if profits are more than 30% of its production, the firm will seek for 4 workers.

As we showed at the beginning `hireTot` represents the total amount of new workers a firm wants to hire, and it is split in high and low skill through `highSkilledWorkersRatio`.

On the contrary, if a firm recorded losses in the previous period, it needs to fire some workers. The procedure `cuttingWorkers` serves this purpose and it works in a specular way with respect to `wantedWorkers`, for example:

```
if profits / production < -0.1 [ set fireTot  2 ]
```

Means that if losses are at least 10% of firm's production, the firms will fire a total amount of 2 workers. Furthermore, again `highSkilledWorkersRatio` divide the number of fired workers, `fireTot`, between high and low skilled.

2.4.2 Opening and closing

If firm's profits of the previous period overcome a certain threshold it will open a new firm. Moreover, we set this threshold to half of the production, changing this parameter we can get different results from the model.

The new firms follow the same procedure as the firms did at the beginning when they were established. Thus, the firm sets its wage amount for high skill workers and for low skill ones as well as the number of workers it wants to hire and how split them according to their skill levels. Furthermore, the firm will check if in the market there are enough workers available that satisfy its conditions, if so the firm hires them and enter in the world otherwise can not open and immediately `die` (see Appendix A).

At the same time, if a firm has unaffordable losses it has to fire all its workers and close, we set that this happens is a firm has more losses than its production:

```
if  ( profits / production ) < -1 [
 ask households with [myFirm = myself] [set myFirm nobody]
 die
]
```

2.4.3 Hiring and firing

The most important procedure of this model is how the firms hire and fire their workers. We already showed under the initial conditions how firms hire their worker, during the cycle they follow the same procedure `hiring` but using `hireHigh` and `hireLow` provided by `workersAmount` instead of `averageWorkers`.

On the other hand, the procedure `firing` allows firms to fire some of their workers in case of losses. `firing` works like `hiring` and it does the same process twice, one for high skill workers and one for low skill ones. The procedure works through the following steps:

- if a firm has losses, < 0, it counts the number of workers for skill level it currently has;

```
let myLowWorkers count households with
 [ myFirm = myself and  householdType = 1 ]
```

- afterwards, the firm takes the minimum number between `fireLow`, recorded during the procedure `cuttingWorkers`, and `myLowWorkers`, got in in previous row;

  ```
  let FL min list fireLow myLowWorkers
  ```

- to conclude, the firm fires the number of workers it would like if they are more than its actual employees, otherwise it fires all its workers remaining with only one. We set this threshold to avoiding firms remaining without workers, we decided that a firm closes only if its losses go over a certain value with respect to its production.

```
ifelse  FL = myLowWorkers and FL > 0
[  ask n-of (FL - 1) households with
 [myFirm =  myself and householdType = 1 ]
 [ set myFirm nobody ]
]
[ ask n-of FL households with
 [myFirm =  myself and householdType = 1 ]
 [ set myFirm nobody ]
```

After this part the cycle continue as a normal first cycle, starting from the agent's adaptation to the market conditions.

3 Results

In this section we evaluate the effectiveness of the model and present some results. The section is divided in two main parts: the first one regards the outcomes of the simulations through a well dened set of parameters, which can be found in table (1); the second one presents the sensitivity analysis of the model to the above-mentioned parameters.

nHouseHlds	1000
nFirms	100
consumption	0.9
highSkilledHouseholdOnPopulation	50%
highSkilledWorkersRatio	55%
fireHighWorkersRatio	50%
averageWorkers	10 ± 2
productivityHigh	1.1
productivityLow	0.8
wageAmountLowSkill	0.8 ±0. 1
wageAmoutHighSkill	1.1 ± 0.1
minWageLowSkill	0.8 ± 0.1
minWageHighSkill	0.8 ± 0.1
Number of cycle	200

Figure 1: *Baseline parameters*

3.1 Baseline model

The following results arise from simulations with `nHouseholds`= 1000, `Firms`= 100 and over a fixed number of 200 cycles. The latter choice deals with the interpretation of a time step: given that firms' behaviour involves hiring, firing and production plans, it seems valid to assume a cycle to be equivalent to month. This interpretation does not actually affect the households, which in the model perceive a certain wage and consume a fraction of it: as a matter of fact, one could take this behaviour as an intensive property of a household.

We compose our population with 40% of high skill workers, OECD average of graduate total population is 37%. Moreover, firms hire 55% high skill workers and 45% low skill ones, this is because nowadays the labour demand for high skill workers is higher. On the contrary the ratio of fired worker is split equally between high and low skill. At the beginning, in order to keep the model balanced we set parameters in a equilibrate way (See figure 1). Indeed, the minimum wages of households are on average 1.1 and 0.8 for high and low skill respectively, both with a standard deviation of 0.1. At the same time, firms offer on average a wage of 0.8 and 1.1, even in this case with a standard deviation of 0.1. Moreover, to keep the price close to 1 we set a productivity for high skill workers of 1.1 while it is 0.8 for low skill ones. When employed households produce, on average, as much as they get and consume 90% of their wage, otherwise they consume 100% and the clearing price starts really close to one. Only the shock on the single firm's price can affect it at the beginning while hereafter even the wage adaptation of firms to the market conditions.

We now summarize the main results of the model on macroeconomics variables:

- *unemployment rate*:

the average unemployment rate is about 5%, but it decreases over time, its average is 6% in the first half ($t < 100$) of the run while it is 4% in the second half; this decrease is due to the adaptation of firms and households to the labour market conditions . Moreover, if we split the unemployment rate by skill type we see that high skill workers average unemployment rate (red line figure 2) is 4% while for low skill households it is 6%, anyway they are both decreasing over time. This is consistent with the parameters of the model; indeed, firms hire more high skill households (55% vs. 45%) but fire equally (50%) and they are even a minority in the population (40%);

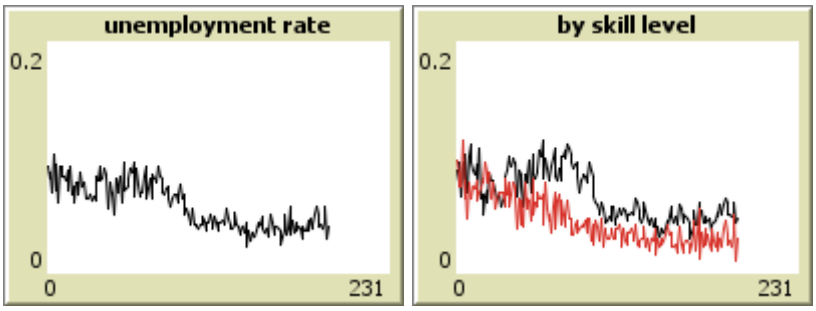

Figure 2: *Unemployment rate*

- *wages*:
  as we specified before, firms must adapt their wages to the market conditions. This adaptation is consistent with the unemployment rate evolution. Indeed, since there is a shortage of high skill workers, firms have to increase their wages to hire them (from an average of 1.1 to an average of 1.3). On the contrary, low skill workers must decrease slightly (from an average of 0.8 to 0.7) their minimum wage since they face more competition to get a job;

Figure 3: *Wages evolution of low (black) and high (red) skill*

- *firms*:
  the turnover of firms is slightly negative, they decrease of 3%. Firms are 97 at the end of the run, because employed households do not consume entirely their wage but only 90% of it. For the same reason profits have bigger negative peaks but, on average, they are almost null since the shock that affects firm's price has mean 0 and standard deviation

0.1. As a matter of fact, firms decrease on number, but they increase their size. At the beginning each firm had an average of 10 workers while at the end of the run there are 5 firms with more than 50 employees, 8 with more than 25, 63 between 25 and 10 while only 21 firms have 10 employees or less;

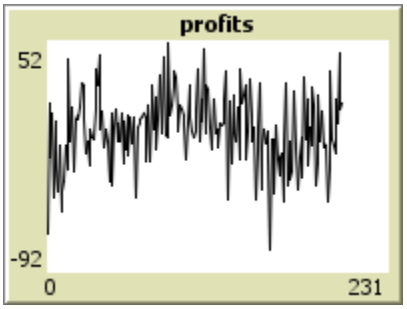

Figure 4: *profits of firms*

- *global price and GDP*:
  despite the small decrease of firms, the GDP, that we defined as the total production of firms, increase of 7% during the entire run. This is correlated with the decrease of unemployment rate, since firms have more workers and produce more. The real GDP (GDP / price) increases even more (+10%) since the market price decreases about 15%. Indeed, it starts from an average of 1.01 at $t = 1$ and it slightly but constantly decreases up the final average of 0.85. We have a decrease of price because of the employment rate increase. Indeed, there are more households which does not consume entirely their wage (while, when unemployed, they consume entirely the subsidy) so the aggregate demand increases less proportionately than the quantity, bringing the price to a final lower level;

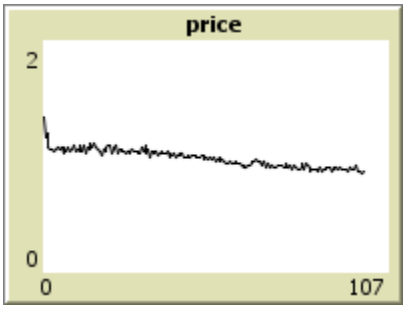

Figure 5: *price*

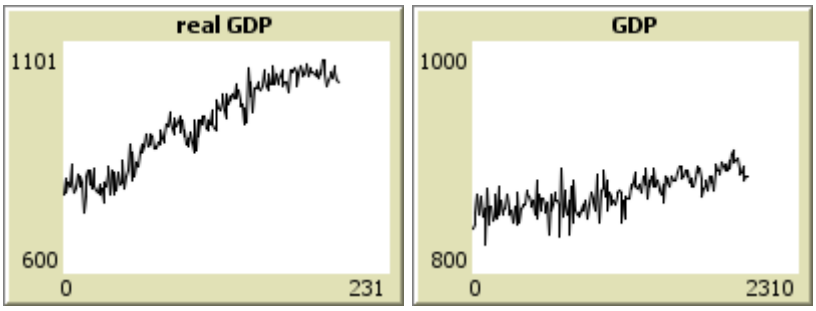

Figure 6: *GDP*

3.2 Sensitive analysis

The model turns out to be robust to the variation of the initial parameters in figure 1 giving reliable expected outcomes.

3.2.1 Job creation and population

For instance, increasing ratio of high skilled workers hired by firms to 75% of the total (`highSkilledWorkersRatio`) we get catastrophic results in our economy. Indeed, there is a constant increase of unemployment rate over time (average 30%); when we split it by skill type we see that almost 50% of low skill households are unemployed while only 1% of high skill are.

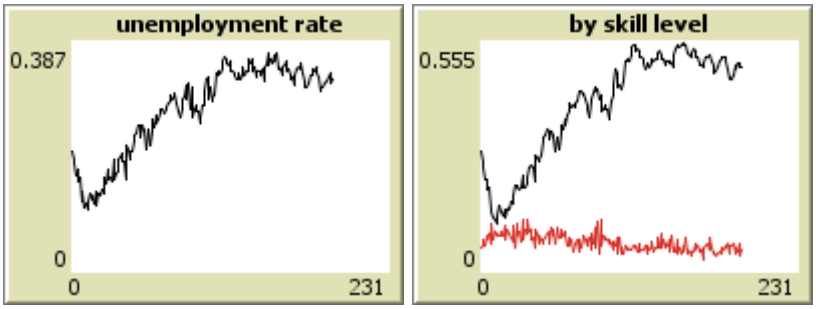

Figure 7: *Unemployment rate*

The increase of unemployment rate is related to the decrease of new job opportunities for low skill unemployed households (only 25% of new job creation) keeping constant the other variables such as their share out of the total population and initial minimum wages. Firms are then able to decrease their wage for low skill households, while they have to strongly increase the ones for high skill that now have more job opportunities.

Figure 8: *Wages of low (black) and high (red) skill*

At the same time, the increase of unemployment rate brings GDP to tumble over time, indeed firms production is proportional to their workers, that are constantly decreasing. On the other hand, the price increases proportionally to the decrease of production.

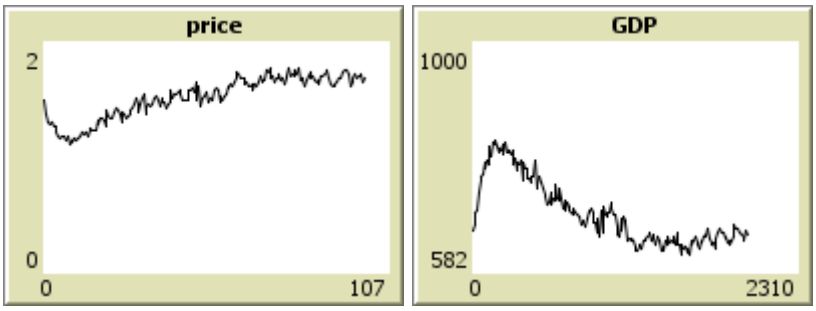

Figure 9: *Price and GDP*

The decrease of job opportunities for low-skill workers keep other variables constant and it mirrors nowadays scenario. Indeed, the raise of artificial intelligence and robotic is bringing the same results in many countries that are now facing high rates of unemployment among their low qualified population. On the contrary, if we increase even the ratio of high skill workers on total population (`highSkilledHouseholdOnPopulation`) to 75% we partially offset the negative results previously obtained. Indeed, we have an employment rate of 10% and a positive economic growth. The latter results show how much is important the match between population's skill and the ones required by the job market.

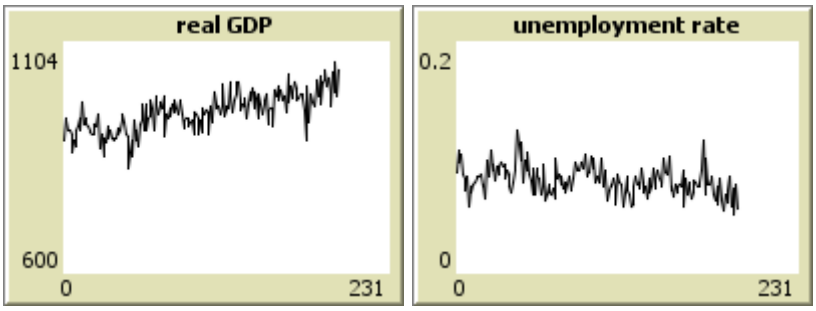

Figure 10: *Price and GDP increasing high skill households*

3.2.2 Wages

Furthermore, after restoring initial settings, we can analyse model's sensitivity with respect to wages. First, we increase the average minimum wage of both high and low skill households up to 1.5 and 1.2. The results are consistent with our expectations, indeed there is a high employment rate during the first part (70% of unemployed households at $t = 10$) of the run since households have minimum wages too high compared to the ones offered by firms. Anyway, during the run, households that do not find a job decrease their minimum wage and this brings to market to its normal conditions in the long run.

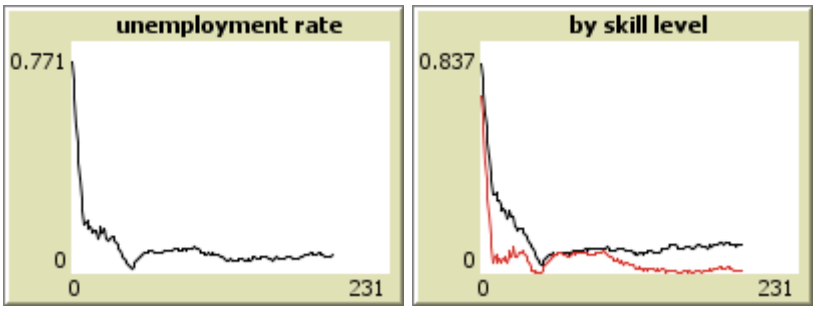

Figure 11: *Unemployment rate with the variation of wages*

As before, the initial high unemployment rate brings to a low production of firms and perhaps a low GDP level as well as an higher price since the demand, driven by the subsidies of unemployed households, is a way higher that production. Anyway, all these variables come back to their normal level once households have adjusted their wage to market conditions.

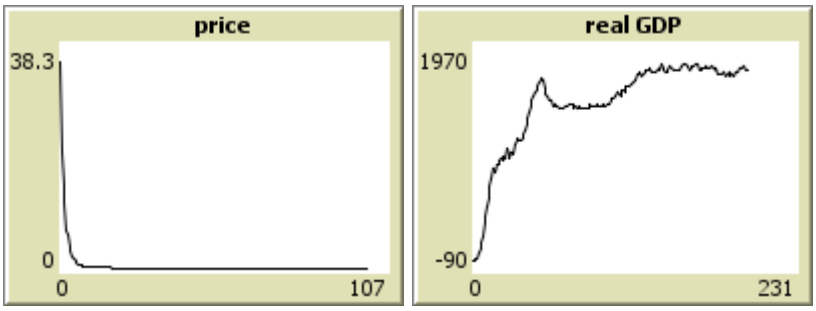

Figure 12: *Price and GDP with the variation of wages*

3.2.3 Productivity

Moreover, if we increase workers' productivity up to 1.5 and 1.1 for high and low skill workers respectively, keeping other settings constant we see the average production of firms increases (GDP). The boost of productivity has even a positive impact on profits which before were zero, on average, because workers productivity was set equal to their costs. In this model, workers productivity is no linked to their wages so that firms raise their production, keeping costs fixed with a positive effect on total GDP.

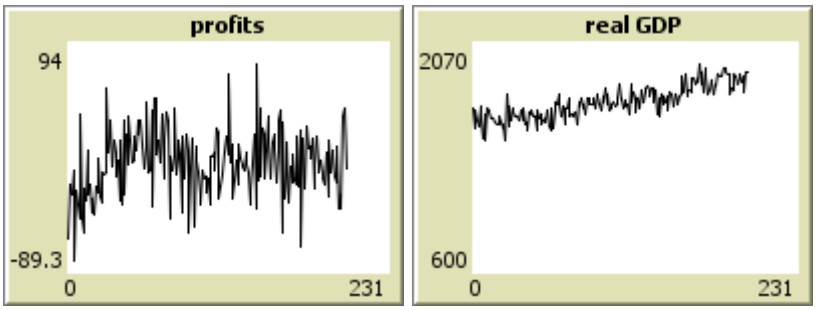

Figure 13: *Real GDP and profits increasing productivity*

3.3 Conclusions

The main aim of the work was to present an economic model able to reproduce a dual labour market of a country with reliable characteristics. This has been accomplished through an agent based approach, with the agents being households and rms. The main achievement of this model is that it represents a closed economic system which depends entirely on the (bounded) rationality of the agents. The simulation experiments carry out the importance of skill matching between demand and supply in the labour market and how some categories are likely to gain and some to loose by eventual differences. Moreover, it shows a stability in production and pricing . Though, the model is a baseline for further improvements: as a matter of fact, it could be expanded with the involvement of other types of agents such as banks and, at the same time, the public administration. In particular a mechanism able to capture the fraction of consume saved by households would be included.

4 Appendix A

```
Breed [ households household ]
Breed [ firms firm ]
Globals [ quantity price totalSubsidies singleSubsidy
          subsidyAmount demand ]

firms-own [ myWages mySubsidies nHighEmployee nLowEmployee
            production revenues costs profits myPrice
            lowWageAmount highWageAmount hireTot hireLow
            hireHigh notEnoughLowWorkers notEnoughHighWorkers
            fireLow fireHigh fireTot gotAllLowWorkers gotAllHighWorkers ]

Households-own [ worker myFirm myWage mySubsidy minWage
                 unemployedTime householdType ]

to setup
  clear-all
  reset-ticks
  setup-households
  setup-firms
  hiring
  set subsidyAmount  0.5
end

to setup-households                                  ;
  create-households nHouseholds
  [
    setxy random-xcor random-ycor
    set shape "person"
    set unemployedTime 0
    set myFirm nobody
    ifelse random-float 100 > highSkilledHouseholdOnPopulation
    [ set householdType 1
      set color green
      set minWage random-normal avMinWageLowSkill stDevMinWageLowSkill ]
    [ set householdType 0
      set color red
      set minWage random-normal avMinWageHighSkill stDevMinWageHighSkill ]
  ]
end
```

```
to setup-firms
  create-firms nFirms
  [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set NotEnoughLowWorkers 0
    set NotEnoughHighWorkers 0
    set gotAllLowWorkers 0
    set gotAllHighWorkers 0
    set profits 0.1
    set highWageAmount random-normal avHighWageAmount stDevHighWageAmount
    set lowWageAmount  random-normal avLowWageAmount stDevLowWageAmount
    set hireTot round random-normal averageWorkers stDevAverageWorkers
    set hireHigh  round ( hireTot * highSkilledWorkersRatio / 100 )
    set hireLow hireTot - hireHigh
  ]
end

to go

  if count households with [ myFirm = nobody ] = nHouseholds [ stop ]
  if ticks > 0 [ wantedWorkers ]
  if ticks > 0 [ opening ]
  if ticks > 0 [ cuttingWorkers ]
  if ticks > 0 [ closing ]
  if ticks > 0 [ hiring ]
  if ticks > 0 [ firing ]
  changeExpectations
  changeWages
  firmWages
  if not any? firms [ stop ]
  generalSubsidy
  firmSubsidies
  firmProduction
  HouseholdsIncome
  householdsDemand
  clearing
  firmsProfit
  if ticks > 250 [stop ]
tick
```

```
end

to wantedWorkers
    ask firms [
     if production > 0 [
      if profits / production > 0.1 [ set hireTot  2 ]
      if profits / production > 0.2 [ set hireTot  3 ]
      if profits / production > 0.3 [ set hireTot  4 ]
      if profits / production > 0.5 [ set hireTot  5 ]
      set hireHigh  round ( hireTot * highSkilledWorkersRatio / 100 )
      set hireLoW hireTot - hireHigh
   ]
  ]
end



to opening

  ask firms [
    if profits >  0.5 * production  [
      hatch 1
      [
       set shape "house"
       setxy random-xcor random-ycor
       set color blue
       set NotEnoughLowWorkers 0
       set NotEnoughHighWorkers 0
       set gotAllLowWorkers 0
       set gotAllHighWorkers 0
       set profits 0.1
       set highWageAmount random-normal avHighWageAmount stDevHighWageAmount
       set lowWageAmount  random-normal avLowWageAmount stDevLowWageAmount
       set hireTot round random-normal averageWorkers stDevAverageWorkers
       set hireHigh  round ( hireTot * highSkilledWorkersRatio / 100 )
       set hireLow hireTot - hireHigh
       let potentialLowWorkers count households with [
        myFirm = nobody and
        minWage <= [ lowWageAmount ] of myself and
        householdType = 1
        ]
```

```
       let potentialHighWorkers count households with [
        myFirm = nobody and
        minWage <= [ highWageAmount ] of myself and
        householdType = 0 ]
       ifelse potentialHighWorkers < hireHigh or
              potentialLowWorkers < hireLow
        [ die ]
        [ ask n-of hireLow households with [
         minWage <= [ lowWageAmount ] of myself and
         householdType = 1 and
         myFirm = nobody
        ]
          [ set myfirm  myself
            set myWage [ lowWageAmount ] of myself
            set unemployedTime 0
            ]
         ask n-of hireHigh households with [
          minWage <= [ highWageAmount ] of myself and
          householdType = 0 and
          myFirm = nobody
         ]
          [ set myfirm  myself
            set myWage [ highWageAmount ] of myself
            set unemployedTime 0
          ]
         ]
        ]
       ]
end



to cuttingWorkers
   ask firms [
    if production > 0 [
      if profits / production < -0.1 [ set fireTot  2 ]
      if profits / production < -0.2 [ set fireTot  3 ]
      if profits / production < -0.3 [ set fireTot  4 ]
      if profits / production < -0.5 [ set fireTot  5 ]
      set FireHigh  round ( fireTot * highSkilledWorkersRatio / 100 )
      set fireLoW fireTot - fireHigh
```

```
   ]
  ]

end




 to closing
  ask firms [
    if production > 0 [
    if  ( profits / production ) < -0.75 [
     ask households with [myFirm = myself] [set myFirm nobody]
     die
    ]
   ]
  ]

end




to hiring

  ask firms [
    if profits > 0 [
     let potentialLowWorkers count households with [
      myFirm = nobody and
      minWage <= [ lowWageAmount ] of myself and
      householdType = 1
     ]
     let LL min list hireLow potentialLowWorkers
     ifelse LL = hireLow
     [ set gotAllLowWorkers gotAllLowworkers + 1 ]
     [ set gotAllLowWorkers 0 ]
     ifelse LL < hireLow
     [ set notEnoughLowWorkers NotEnoughLowWorkers + 1 ]
     [set NotEnoughLowWorkers 0 ]
     if profits > 0 and potentialLowWorkers > 0 [
      ask n-of LL households with [
```

```
       minWage <= [ lowWageAmount ] of myself and
       householdType = 1 and
       myFirm = nobody
      ]
      [ set myfirm  myself
        set myWage [ lowWageAmount ] of myself
        set unemployedTime 0
      ]
     ]
     let potentialHighWorkers count households with [
      myFirm = nobody and
      minWage <= [ highWageAmount ] of myself and
      householdType = 0
      ]
     let HH min list hireHigh potentialHighWorkers
     ifelse HH = hireHigh
     [ set gotAllHighWorkers gotAllHighworkers + 1 ]
     [ set gotAllHighWorkers 0 ]
     ifelse HH < hireHigh
     [ set notEnoughHighWorkers NotEnoughHighWorkers + 1 ]
     [set NotEnoughHighWorkers 0 ]
     if profits > 0 and potentialHighWorkers > 0 [
      ask n-of HH households with [
       minWage <= [ highWageAmount ] of myself and
       householdType = 0 and
       myFirm = nobody
       ]
     [ set myfirm  myself
       set myWage [ highWageAmount ] of myself
       set unemployedTime 0
     ]
    ]
   ]
  ]
end



to firing

  ask firms [
    if profits < 0 [
```

```
     set gotAllHighWorkers 0
     set NotEnoughHighWorkers 0
     set gotAllLowWorkers 0
     set NotEnoughLowWorkers 0
     let myHighWorkers count households with [
      myFirm = myself and
      householdType = 0
      ]
     let FH min list fireHigh myHighWorkers
     ifelse  FH = myHighWorkers and FH > 0
      [ ask n-of (FH - 1) households with [
       myFirm =  myself and
       householdType = 0
       ]
       [ set myFirm nobody ]
      ]
     [ ask n-of FH  households  with [
      myFirm =  myself and
      householdType = 0
      ]
      [ set myFirm nobody ]
     ]
    let myLowWorkers count households with [
     myFirm = myself and
     householdType = 1
     ]
    let FL min list fireLow myLowWorkers
    ifelse  FL = myLowWorkers and FL > 0
      [ ask n-of (FL - 1) households with
        [myFirm =  myself and householdType = 1 ]
        [ set myFirm nobody ]
      ]
      [ ask n-of FL households with
      [myFirm =  myself and householdType = 1 ]
      [ set myFirm nobody ]
     ]
    ]
   ]


end
```

```
to changeExpectations

  ask households  [
    if myFirm = nobody
     [ set unemployedTime unemployedTime + 1 ]
    if unemployedTime = 3 [ set minWage 0.9 * minWage ]
    if unemployedTime = 4 [ set minWage 0.9 * minWage ]
    if unemployedTime >= 5 and minWage > subsidyAmount
     [ set minWage 0.8 * minWage ]
  ]

end


to firmWages

  ask firms [
    set nHighEmployee count households with
    [ myself = myFirm and  householdType = 0 ]
    set nLowEmployee  count households with
    [ myself = myFirm and  householdType = 1 ]
    set myWages  ( ( lowWageAmount * nLowEmployee ) +
                   ( highWageAmount * nHighEmployee ) )
    if  nHighEmployee = 0 and  nLowEmployee = 0 [ die ]
    if  nHighEmployee + nLowEmployee > 10 and
        nHighEmployee + nLowEmployee <= 25
        [ set color grey ]
    if  nHighEmployee + nLowEmployee > 25 and
        nHighEmployee + nLowEmployee <= 50
        [ set color brown ]
```

```
    if  nHighEmployee + nLowEmployee > 50
        [ set color white ]

  ]

end

to changeWages

  ask firms [
    if gotAllLowWorkers = 3
    [ set LowWageAmount LoWWageAmount * 0.9 ]
    if gotAllHighWorkers = 3
    [ set highWageAmount highWageAmount * 0.9 ]
    if gotAllLowworkers = 4
    [ set lowWageAmount lowWageAmount * 0.9 ]
    if gotAllHighWorkers = 4
    [ set highWageAmount highWageAmount * 0.9 ]
    if gotAllLowWorkers >= 5 and lowWageAmount > subsidyAmount
    [ set lowWageAmount lowWageAmount * 0.8 ]
    if gotAllHighWorkers >= 5 and highWageAmount > subsidyAmount
    [ set highWageAmount highWageAmount * 0.8 ]
    if notEnoughLowWorkers = 2
    [ set lowWageAmount lowWageAmount * 1.1 ]
    if notEnoughHighWorkers = 2
    [ set highWageAmount highWageAmount * 1.1 ]
    if notEnoughLowWorkers = 3
    [ set lowWageAmount lowWageAmount * 1.1 ]
    if notEnoughHighWorkers = 3
    [ set highWageAmount highWageAmount * 1.1 ]
    if notEnoughLoWWorkers >= 4 and lowWageAmount <  4
    [ set lowWageAmount lowWageAmount * 1.2 ]
    if notEnoughHighWorkers >= 4 and highWageAmount < 5
    [ set highWageAmount highWageAmount * 1.2 ]
  ]

end



to generalSubsidy
```

```
  set totalSubsidies
  ( count households with [ myFirm = nobody ] * subsidyAmount )
  set singleSubsidy
  totalSubsidies / (sum [ nHighEmployee ] of firms +
                    sum [ nLowEmployee ] of firms )

end

to firmSubsidies

  ask firms [
    set mySubsidies singleSubsidy *
    ( nHighEmployee + nLowEmployee )

 end

to firmProduction

  ask firms [
    set production   ( nHighEmployee * productivityHigh +
                       nLowEmployee * productivityLow )
  ]

end

to householdsIncome

  ask households [
    ifelse myFirm = nobody
    [ set mySubsidy subsidyAmount ]
    [ set mySubsidy 0 ]
  ]

end

to householdsDemand

  set demand  ( ( sum [ myWage ] of households * consumption ) +
                sum [ mySubsidy ] of households )

end
```

```
to clearing

  set price demand  / sum [ production ] of firms

end


to firmsProfit
  ask firms [
    set myPrice price + random-normal 0 0.1
    set revenues myPrice * production
    set costs myWages + mySubsidies
    set profits revenues - costs
   ; show highWageAmount
  ]

end
```