A retirement decision-making model

Econophysics

Pietro Folco

Fisica dei Sistemi Complessi
Università degli Studi di Torino

A retirement decision-making model

Econophysics

Pietro Folco

Abstract

The aim of this work is to analyse the retirement decision-making of Italian people under the 2008 reform by considering several parameters that could influence their decision. To achieve this goal, I developed an agent-based simulation model on NetLogo trying to take in account the major peculiarities of a person: gender, age, work conditions, health, family condition, income and social relations. From the results of this model, I have tried to find a relationship between the decision-making process and each of these peculiarities.

Contents

Chapter 1

Introduction

1.1 The agents and their features

This model would simulate the trend of the retirement decision-making of Italian people under the 2008 reform trying to analyse particularly this trend in function of several parameters. In this model there are three types of agent:

- **Hasty Agents:** they retire at the earliest possible age allowed by the reform.
- **Rational Agents:** when they reach the retirement eligibility age, their retirement decision is given by the combination of several parameters.
- **Random Agents:** when they reach the retirement eligibility age, every year they could retire with probability p. We do not care about the output of this type of agent, but we are interested in the effect of the number variation of these agents on the rational agents retirement decision-making.

Each agent will be characterized by several parameters:

- **Gender.**
- **Age.**
- **Type of worker:** each agent could be "employee", "self-employed", "unemployed" or "retired".
- **Work:** the agents could be blue collar or white collar. A blue-collar worker does a weary work and his income could be low (40 %) or medium (60 %); a white-collar worker does a non-weary work and his income could be low (20 %), medium (50 %) or high (30 %). Agents cannot change from blue-collar to white-collar during the simulation.
- **Years of seniority.**
- **Health:** "good" or "bad" health.
- **Family:** each agent could have "small" or "big" family.

- **Income:** it could be "low", "medium" or "high". We suppose that a "low" income corresponds to a low retirement pension, a "medium" income corresponds to a medium retirement pension and a "high" income corresponds to a "high" retirement pension. Unemployed agents have as income parameter "none".

1.2 A description of Rational Agents features

When they reach the retirement eligibility age, their retirement decision is given by the combination of several parameters: from the retirement eligibility age, every year until the agent retires, each parameter contributes to the retirement decision with a score. The agent retires when scores 130. The parameters that contribute to the retirement decision are:

1. **Weary or non-weary work:** a *weary work* scores 50 the first year of eligibility and 25 the following years. A *non-weary work* scores 20 the first year and 10 the followings.
2. **Social network effect:** when, in the social network of agent i, the fraction f of retired people with respect to the eligible agents is greater than a fixed threshold t [1], agent i scores 40 points.
3. **Health:** bad health let score 60 points.
4. **Age:** agent i scores $\frac{age}{5}$ points every year.
5. **Income:** this feature let score points only in the first year of eligibility. Unemployed agents (income = "none") score 0, "low" income let score 10, "medium" income let score 25 and "high" income let score 40.
6. **Family:** "small" family scores 15 points in the first year of eligibility, then, in the following years, it scores 10.
7. **Random:** this parameter could score from 0 to 20 points the first year and from 0 to 10 the following years. This feature highlights the uniqueness of the work condition of each agent: retirement decision could be also affected by emotional and irrational components that we try to model with a random component.

1.3 What does happen at each time step while the model is running?

- A time step corresponds to one year: each agent's age counter increases by 1.
- Employed agents years of seniority counter increases by 1.
- Each agent could change her/his health from "good" to "bad" with a probability p increasing in age, moreover the agent will die when is age $>$ 99.

- When an agent dies a new agent is created, at the creation she/he is between 16 and 24 years old. When a new agent is created, she/he is "unemployed" and she/he could find a job from the next time step. An ≤ 18 years old agent created will be a blue-collar, an > 18 years old agent created will be a blue-collar with a probability of 30 %.
- If an agent has "small" family could turn in "big" family with a probability of 10 %.
- "Unemployed" agents find a job with a probability depending on age and by following the occupational rate from 2017 ISTAT data [2].
- Each agent evaluates if satisfied the requirement in age and seniority to retire. If she/he does, the model applied the procedures of the specific agent (hasty, random or rational).
- At time step 25, the simulation stops and the data are printed in a .txt file, clicking on the *go* button the simulation goes on until the time step 50 where the simulation stops again, data are printed in a .txt file and the simulation ends. We can disable this feature by turning off the switch "document".

Chapter 2

The Model

2.1 An interface overview

When we open "a retirement decision making model" in NetLogo, we should find a screenshot equivalent to Figure 2.1.

We must set all the parameters of the sliders and switches before initializing the simulation by clicking the setup button. Here a description of each parameter:

- **number_of_agents:** this slider has a minimum of 50 and a maximum of 500. It sets the total number of agents in the simulation. This number is constant, it must be set before the initialization and will not change during the simulation: when an agent dies, she/he is replaced by a new agent in the same time step.
- **randomAgentsPercent:** set the fraction of random agents in the simulation. It varies from 0 (0 % of random agents in the simulation) and 0.45 (45 % of random agents in the simulation). This percent is kept constant during the simulation. It is important to highlight that by setting this slider on the percent of random agents we are also setting the percent of hasty agents: 50 % of the agents are rational, so the remaining 50 % is divided in hasty and random with a percent set by this slider.
- **link-threshold:** this slider is related to the rational agents. It sets the threshold t of the social network effect: when the fraction f of retired people with respect to the eligible agents is greater than the threshold t, the agent i scores 40 points. *link-threshold* could vary in the range [0,1].
- **random_retirement_decision_threshold:** set the probability of retirement of a random agent that has reached the eligibility. It varies in the range [0,1].
- **document:** when it is turned off no .txt files are printed.
- **Switches (work?, social_network?, health?, income?, family?, random?):** these switches are related to the rational agents. If a switch is turned on, it means that the corresponding parameter of the agent will influence the retirement decision as described in section 1.2.

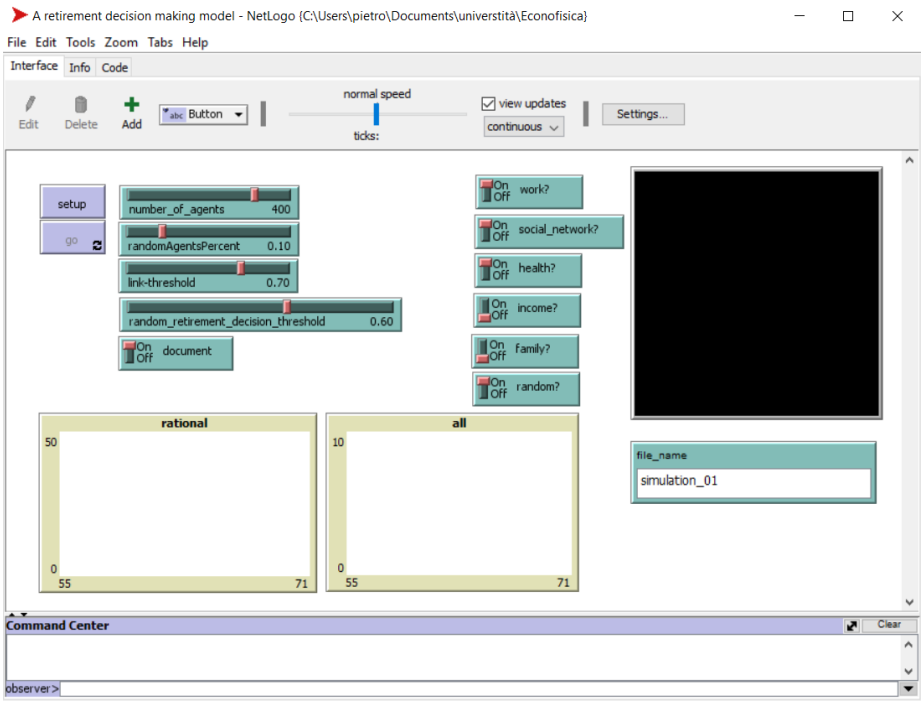

Figure 2.1: **A retirement decision making model: the NetLogo interface.**

  - work? is related to *weary or non-weary work*
  - social_network? is related to *social network effect*
  - health? is related to *health*
  - income? is related to *income*
  - family? is related to *family*
  - random? is related to *random*

  If the switch is turned off, the corresponding parameter will not influence the retirement decision. The score that the agent must reach to retire decreases by 20 points in order to cancel the effect of the turned off parameter.

- **file_name input box:** it names the file .txt that the program will generate at time step 25 and 50 (if document = True). All the generated files will have this name:

  - Time step 25: *a_retirement_decision_making_model_25TICKS_"file_name".txt*
  - Time step 50: *a_retirement_decision_making_model_50TICKS_"file_name".txt*

  We could write the name in the input box before or after the initialization of the model, but we must write before clicking on the *go* button.

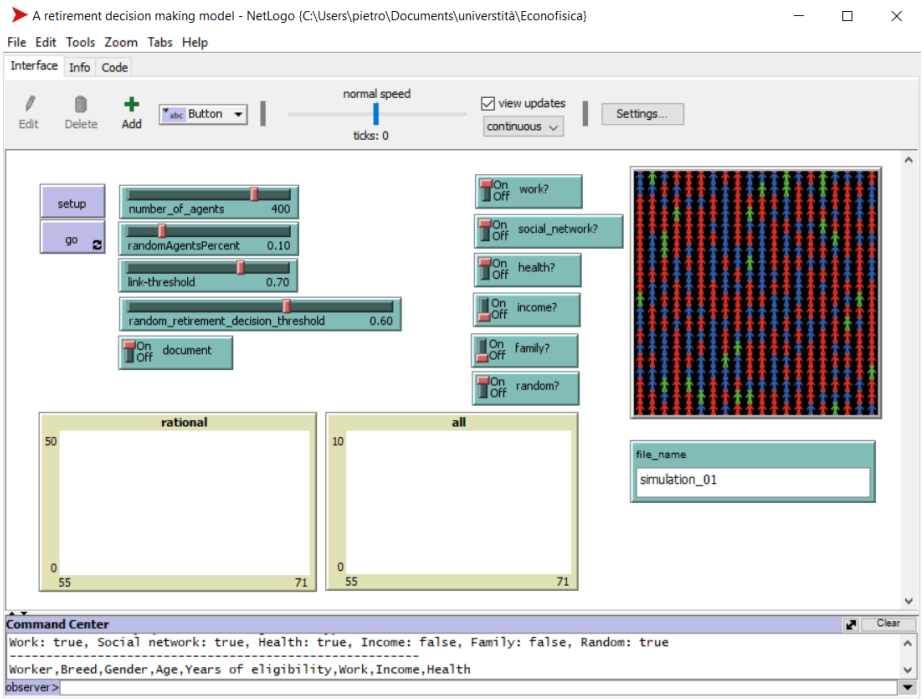

Figure 2.2: **A retirement decision making model: the NetLogo initialized interface.**

2.2 The initialization

First of all we set all the parameters, then we could click on setup to initialize the model (Figure 2.2).

The *go* button is now enabled: before initializing the model it was disabled, so if we accidentally click on it when the model is not already initialized nothing happens, otherwise NetLogo would be gone in error. We could see in the upper right corner of the screen the NetLogo world, now it is populated by 400 agents by a function called "populate_world" [3]. There are three colours for the agents. They represent the three different type of agents:

- **rational agents** are *red*
- **hasty agents** are *blue*
- **random agents** are *green*

The NetLogo world in this simulation is not important for the results of the model: I chose to represent it in this way only for aesthetic reasons. We could also observe that, in the command center, the state of all the parameters set before the initialization has been printed by a function called "info". This will be the heading of the .txt file printed at time step 25 and time step 50.

2.3 Agents

Each agent is characterized by a breed and by a set of variables.

The breed denotes the agent type: rational agents ($breed = rationals$), hasty agents ($breed = hasties$), random agents ($breed = randomAgents$).

```
to assignBreed
  ask turtles
  [
  let r random-float 1
  ifelse r < randomAgentsPercent
      [set breed randomAgents]
  [ifelse r > randomAgentsPercent and r < 0.5
      [set breed hasties]
      [set breed rationals]]
  ]
end
```

- $randomAgentsPercent$ of probability to set breed $randomAgents$;
- $0.5 - randomAgentsPercent$ of probability to set breed $hasties$;
- 0.5 of probability to set breed $rationals$.

Each agent has a set of variables assigned at the initialization of the model:

- **gender:** "male" or "female" [4];
- **age:** the age assignment follows this distribution [4]:
  - $16 \leq age \leq 24$ $probability$: 11.19%
  - $25 \leq age \leq 54$ $probability$: 48.82%
  - $55 \leq age \leq 64$ $probability$: 16.04%
  - $age \geq 65$ $probability$: 23.95%
- **health:** I set the probability to be in good health by following this criterion:
  - $16 \leq age \leq 24$ $probability$: 99.9%
  - $25 \leq age \leq 54$ $probability$: 99.5%
  - $55 \leq age \leq 64$ $probability$: 99.0%
  - $age \geq 65$ $probability$: 85.0%
- **seniorityYears:** this parameter set the years of seniority of the agent. I assumed that, with a probability of 67 %, the agent starts to work when she/he was between 16 and 24 years old and never stopped (maximum value for seniorityYears is 40). There is instead a probability of 33 % that the agent would have been unemployed during her/his life and so she/he would have a lower seniorityYears value;

- **workerType:** could be unemployed, self-employed, employee or retired. Agents would be "retired" if their age is greater that 70 or if they satisfy the eligibility retirement criteria based on age and seniorityYears. Otherwise, the workerType feature is set "unemployed" with a probability given by the 2017 ISTAT data [2] (10.3 % for male and 12.4 % for female), "self-employed" or "employee" with the same probability;
- **eligible:** this variable is 0 for all the agents that does not satisfy the eligibility retirement criteria of the Italian 2008 reform. When the requirements are satisfied, eligibility increases by 1. Then, if the agent decides to retire, workerType is changed on "retired", otherwise eligibility increases by 1 at each time step until the agent decides to retire;
- **income:** it is "none" for unemployed, could be "low" or "medium" for blue-collar and could be "low", "medium" or "high" for white-collar;
- **score (rational agents only):** it is initially set on 0, when the rational agent become eligible, it could increase by following the criteria explained in section 1.2.

These features are initially set by a procedure called "assignFeatures":

```
to assignFeatures
  ask turtles
  [
    ;;;;;;;;;;;;;;;;;;;;;;; assing gender ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

    ifelse random-float 1 < 0.5181
          [set gender "female"]
          [set gender "male"]

    ;;;;;;;;;;;;;;;;;;;;;;; assign age ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

    ifelse random-float 1 < 0.1119
    [set age 16 + random 8] ; 16<age<24 percentage: 0.1119
    [ifelse random-float 1 < 0.5497
      [set age 25 + random 29] ; 25<age<54 percentage: 0.4882
       [ifelse random-float 1 < 0.4011
        [set age 55 + random 9] ; 55<age<64 percentage: 0.1604
        [set age 65 + random 25]]]; 65<age<90 percentage: 0.2395

    ;;;;;;;;;;;;;;;;;;;; assign family ;;;;;;;;;;;;;;;

    ifelse random-float 1 < 0.7
    [set family "small"]
    [set family "big"]
```

```
;;;;;;;;;;;;;;;;;; assing health ;;;;;;;;;;;;;;;

if age < 25
[ifelse random-float 1 < 0.999
  [set health True]
  [set health False]]
if age > 24 and age < 54
[ifelse random-float 1 < 0.995
  [set health True]
  [set health False]]
ifelse age > 53 and age < 64
[ifelse random-float 1 < 0.99
  [set health True]
  [set health False]]
[ifelse random-float 1 < 0.85
  [set health True]
  [set health False]]

;;;;;;;;;;;;;;;;;;;; assign work ;;;;;;;;;;;;;;;;;;;;;;;;;

ifelse age <= 18
[set work "blue-collar"]
[ifelse random-float 1 < 0.6
  [set work "blue-collar"]
  [set work "white-collar"]]

;;;;;;;;;;;;;;;;;; assign seniorityYears ;;;;;;;;;;;;;;;;;;

ifelse random-float 1 < 0.67                ;agents never been unemployed
[
   let blue-seniority age - 16 - random 8
   if blue-seniority > 40
     [set blue-seniority 40]
   if blue-seniority < 0
     [set blue-seniority 0]
   let white-seniority age - 18 - random 6
   if white-seniority > 40
     [set white-seniority 40]
    if white-seniority < 0
     [set white-seniority 0]

   ifelse work = "blue-collar"
   [set seniorityYears blue-seniority]
```

```
    [set seniorityYears white-seniority]]
[                                                ;these agents could have been unemployed
    let blue-seniority random age  - 16 - random 8
    if blue-seniority > 40
      [set blue-seniority 40]
    if blue-seniority < 0
      [set blue-seniority 0]
    let white-seniority random age - 18 - random 6
    if white-seniority > 40
      [set white-seniority 40]
    if white-seniority < 0
      [set white-seniority 0]

    ifelse work = "blue-collar"
    [set seniorityYears blue-seniority]
    [set seniorityYears white-seniority]]

 ;;;;;;;;;;;;;;;;;; assign type of worker ;;;;;;;;;;;;;;;;;;;;;;

ifelse age > 70
[set workerType "retired"]
[ifelse age >= 61 and seniorityYears >= 35 and age + seniorityYears >= 97
[ifelse random-float 1 < 0.7
   [set workerType "retired"]
   [ifelse gender = "male"
     [ifelse random-float 1 < 0.103
       [set workerType "unemployed"]
       [ifelse random-float 1 < 0.5
         [set workerType "self-employed"]
         [set workerType "employee"]]]
     [ifelse random-float 1 < 0.124
       [set workerType "unemployed"]
       [ifelse random-float 1 < 0.5
         [set workerType "self-employed"]
         [set workerType "employee"]]]]]
 [ifelse gender = "male"
     [ifelse random-float 1 < 0.103
       [set workerType "unemployed"]
       [ifelse random-float 1 < 0.5
         [set workerType "self-employed"]
         [set workerType "employee"]]]
     [ifelse random-float 1 < 0.124
       [set workerType "unemployed"]
```

```
        [ifelse random-float 1 < 0.5
          [set workerType "self-employed"]
          [set workerType "employee"]]]]

    ;;;;;;;;;;;;; assign eligible ;;;;;;;;;;;;;;

    ifelse workerType = "retired"
      [set eligible 1]
      [set eligible 0]

    ;;;;;;;;;;; assign income ;;;;;;;;;;;;;;;;;;;;;;;;;;

    ifelse workerType = "unemployed"
    [set income "none"]
    [
    if work = "blue-collar"
    [ifelse random-float 1 < 0.4
      [set income "low"]
      [set income "medium"]]

    if work = "white-collar"
    [ifelse random-float 1 < 0.5
      [set income "medium"]
      [ifelse random-float 1 < 0.4
          [set income "low"]
          [set income "high"]]]]

  ]

  ;;;;;;;;;;;;;;;;;;;;;;;; assign score to rationals ;;;;;;;;;;;;;;;;;;;;;;;;;;

  ask rationals [set score 0]

end
```

Rational agents have also another variable called *my-friends-want-me-retired*. It is a boolean variable that contributes to the score by the social network effect. It is initially set on "False", then, when in the social network of the agent, the fraction f of retired people with respect to the eligible agents is greater than a fixed threshold t (that we called *link-threshold*), the variable is set on "True". The creation of the social network of the agent and the evaluation of f with respect to the *link-threshold* is provided by the function "linked-agents".

```
to linked-agents
  ask rationals [
    let eligibility 0
    let retired 0
    let counter 0

    while [counter < 5 + random 20]
    [
    let myFriend one-of other turtles

    if myFriend != nobody [

    while [abs (age - [age] of myFriend) > 5]
        [set myFriend one-of other turtles]

    ask myFriend [
      if age + seniorityYears > 97 and age > 61
        [set eligibility eligibility + 1
         if workerType = "retired"
              [set retired retired + 1]]]
      set counter counter + 1 ]]

    ifelse eligibility = 0
    [set my-friends-want-me-retired False]
    [ifelse retired / eligibility > link-threshold
    [set my-friends-want-me-retired True]
      [set my-friends-want-me-retired False]]  ]

end
```

2.3.1 Procedure "go"

Now, with the initialized model, we can click on "go" to start the simulation (Figure 2.3). While the simulation is running, the "go" button is black coloured and "ticks" (time steps) increases. Two histograms that show the frequency of retirement for each age, they change while the simulation is running. The histogram named "rational" shows the frequency for rational agents while the histogram named "all" shows the frequency of all the agents. Moreover, for any agent who retires, her/his workType, breed, gender, age, eligibility (it counts how many years the agent waited before retiring from when she/he became eligible), work, income and health are printed in the command center.

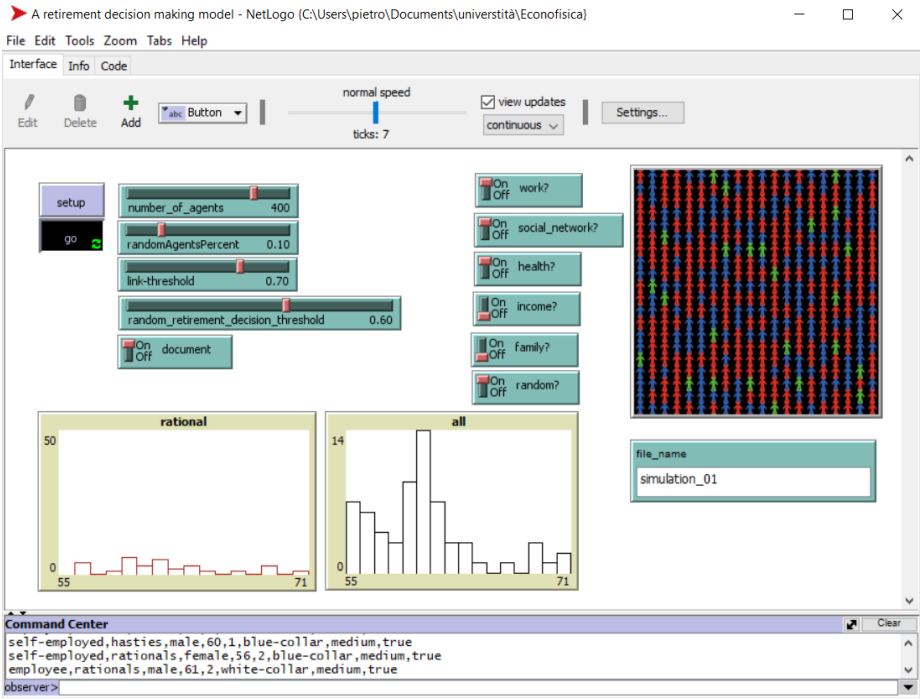

Figure 2.3: **A retirement decision making model: while the simulation is running.**

If document = "True", all these data will be printed in the .txt file produced at tick = 25 and at tick = 50.
The "go" procedure contains a list of other procedures :

```
to go

  ask turtles[set age age + 1]
  increasing-seniority
  getting-ill
  to-die
  create-new-turtles
  family-growing
  find-a-job
  linked-agents
  retire
  graph

  if ticks = 25
  [export-output (word "C:/Users/pietro/Documents/universtità/Econofisica/a_retirement_decision_making_model_25TICKS_" file_name ".txt")
    tick
    stop
  ]
  if ticks = 50
  [export-output (word "C:/Users/pietro/Documents/universtità/Econofisica/a_retirement_decision_making_model_50TICKS_" file_name ".txt")
    stop
  ]

  tick

end
```

The first thing that the "go" procedure does is to increase the age and, through "increasing-seniority", the seniorityYears of all the agents.
The procedure "getting-ill" changes the health state from "True" (good health) to "False" (bad health) with a probability depending on age.

The procedure "to-die" lets die all the agents that in the current time step are > 99 years old. Thanks to the procedure "create-new-turtles", each dead agent is immediately replaced by a new agent with an age between 16 and 24; in this procedure are also assigned all the features to the new agent.
The procedure "family-growing" changes the family parameter from "small" to "big" with a 10 % of probability.
Thanks to the procedure "find-a-job" the unemployed agents could find a job (and so change her/his workerType from "unemployed" to "self-employed" or "employee") with a probability that changes in age [2].
Then, "linked-agents" and "retire" procedures are called. "Retire" is the most important part of the model so we will discuss it separately.
The procedure "graph" defines the histograms that we can see in the interface.
The "if" conditions export a .txt file with the data produced by the simulation respectively until the 25^{th} tick and the 50^{th} tick, then they stop the simulation.
When the "if" conditions are not called (and so the model keeps running) "tick" let increase ticks by 1.

2.3.2 Procedure "retire"

The first thing that this procedure does is to verify for any agent with $eligibility = 0$ (that means that the agents did not satisfy the eligibility retirement criteria in the last time step) if, in this step, satisfy the eligibility retirement criteria [5]. If she/he does eligibility is set equal to 1.

2008 reform eligibility retirement criteria			
Employee		**Self-employed**	
Seniority pension	**Old-age pension**	**Seniority pension**	**Old-age pension**
Men and women: min age 61 +, min seniority 35 years, + sum (age; seniority) $\geq$ 97	*Men*: age 60 +, seniority 15 years *Women*: age 55 +, seniority 15 years	*Men and Women*: min age 62 +, min seniority 35 years, + sum (age; seniority) $\geq$ 98	*Men*: age 60 +, seniority 15 years *Women*: age 55 +, seniority 15 years

When the agent becomes eligible, the procedure distinguishes three cases (1 for each breed):

- **hasty agents:** workerType changes in "retired";
- **random agents:** with a probability set by *random_retirement_decision_threshold* the agent's workerType changes in "retired". Otherwise eligibility increases by 1;
- **rational agents:** the procedure evaluates any active parameter (weary or non-weary work, social network effect, health, age, family, income, random) with the criteria explained in section 1.2. If the score of the agent is greater or equal to a fixed threshold the agent's workerType changes in "retired", otherwise eligibility increases by 1.

When, in the procedure, the workerType changes in "retired", in the command center is printed workType, breed, gender, age, eligibility, work, income and health of the agent.

```
to retire

  ask turtles [

    if eligible = 0
    [if workerType = "employee" [
      ifelse age >= 61 and seniorityYears >= 35 and age + seniorityYears >= 97
        [set eligible 1]
        [ifelse gender = "female"
        [if age >= 55 and seniorityYears >= 15 [
          set eligible 1]]
        [if age >= 60 and seniorityYears >= 15 [
          set eligible 1]]]]

    if workerType = "self-employed" [
      ifelse age >= 62 and seniorityYears >= 35 and age + seniorityYears >= 98
        [set eligible 1 ]
        [ifelse gender = "female"
        [if age >= 55 and seniorityYears >= 15 [
          set eligible 1]]
        [if age >= 60 and seniorityYears >= 15 [
          set eligible 1 ]]]]

    if workerType = "unemployed" [ ;only old-age pension for unemployed
      ifelse gender = "female"
        [if age >= 55 and seniorityYears >= 15 [
          set eligible 1]]
        [if age >= 60 and seniorityYears >= 15 [
          set eligible 1]]]

      if age > 70 [set eligible 1]
    ]
```

```
  if eligible >= 1 and workerType != "retired"

  [if age > 70 [type workerType
    set workerType "retired" set retired_at age
    type "," type breed type "," type gender type "," type age type ","
    type eligible type "," type work type "," type income type "," print health
    ]
  if breed = randomAgents
    [ifelse random-float 1 < random_retirement_decision_threshold [
      type workerType set workerType "retired" set retired_at age
    type "," type breed type "," type gender type "," type age type ","
    type eligible type "," type work type "," type income type "," print health
    ]
      [set eligible eligible + 1]]
   if breed = hasties
    [type workerType
     set workerType "retired" set retired_at age
     type "," type breed type "," type gender type "," type age type ","
     type eligible type "," type work type "," type income type "," print health
]]]
   ;;;;;;;;;;; rational agents ;;;;;;;;;;;;;
  ask rationals [
  let work-penalty 0
  let social_network-penalty 0
  let health-penalty 0
  let income-penalty 0
  let family-penalty 0
  let random-penalty 0

  if work? = False
  [set work-penalty 20]
  if social_network? = False
  [set social_network-penalty 20]
  if health? = False
  [set health-penalty 20]
  if income? = False
  [set income-penalty 20]
  if family? = False
  [set family-penalty 20]
  if random? = False
  [set random-penalty 20]
```

```
    if eligible >= 1 and workerType != "retired" [
      set score score + age / 5
     ;;;;; first year ;;;;;;;;;;
      ifelse eligible = 1 [
        ifelse work = "blue-collar" and work? = True
         [set score score + 50]
         [set score score + 20]
        if my-friends-want-me-retired = True and social_network? = True
         [set score score + 40]
        if health = False and health? = True
         [set score score + 60]
        if income = "none" and income? = True
        [set score score + 0]
        if income = "low" and income? = True
        [set score score + 10]
        if income = "medium" and income? = True
        [set score score + 25]
        if income = "high" and income? = True
        [set score score + 40]
        if family = "small" and family? = True [set score score + 15]

        if random? = True
        [set score score + random 20]]
      ;;;;;;;;;;;; following years ;;;;;;;;;;;;
      [if work? = True [ifelse work = "blue-collar"
        [set score score + 25]
        [set score score + 10]]
        ;family and following years
        if family = "small" and family? = True [set score score + 10]
        if random? = True [set score score + random 10]]

      if breed = rationals [ifelse score >= 130
        - work-penalty - social_network-penalty
        - health-penalty - income-penalty - family-penalty - random-penalty
       [type workerType
        set workerType "retired" set retired_at age
        type "," type breed type "," type gender type ","
        type age type "," type eligible
        type "," type work type "," type income type "," print health
       ]
        [set eligible eligible + 1]  ]]]
end
```

Chapter 3

Data analysis

In order to analyse the trend of the retirement decision-making, I consider two parameters of output: age (at the time step of the agent retirement) and eligibility, I am interested in these outputs of the rational agents. I am also interested in the age at retirement of hasty agents to highlight the average age of first eligibility. First, I ran the model with all the parameters of rational agents turned on, *number_of_agents = 380, randomAgentsPercent = 0.10, link-threshold = 0.50, random_retirement_decision_threshold = 0.50* (Figure 3.1 and Figure 3.2).

From the histograms of Figure 3.1 and Figure 3.2 we could see two distribution: we could expect it because the eligibility retirement requirements are different for male and female. For this reason, I separated the data by gender for the retirement age analysis (Figure 3.3 and Figure 3.4).

It is interesting to compute the average age of retirement and the eligibility counter separately for female and male, I also computed the total average of the eligibility counter (Table 1).

work? *True;* **social_network?** *True;* **health?** *True;* **income?** *True;* **family?** *True;* **random?** *True*								
Table 1					**Hasty agents**			
	Age of retirement [years]	**σ_{Age} [years]**	**Eligibility [years]**	**σ_{El} [years]**	**Age of retirement [years]**	**σ_{Age} [years]**	**Eligibility [years]**	**σ_{El} [years]**
25 ticks (Female)	59.7	0.6	2.0	0.1	58.1	0.7	-	-
50 ticks (Female)	58.7	0.5	2.0	0.1	57.3	0.5	-	-
25 ticks (Male)	61.7	0.4	1.7	0.1	61.8	0.6	-	-
50 ticks (Male)	61.3	0.3	1.7	0.1	61.1	0.4	-	-
25 ticks (All agents)	-	-	1.9	0.1	-	-	-	-
50 ticks (All agents)	-	-	1.9	0.1	-	-	-	-

From these data we could assert that:

- the average age of retirement decreases from the 25 time step to the 50 time step while the eligibility counter remains the same. This is probably caused by the initialization of the model: when we set it up the model

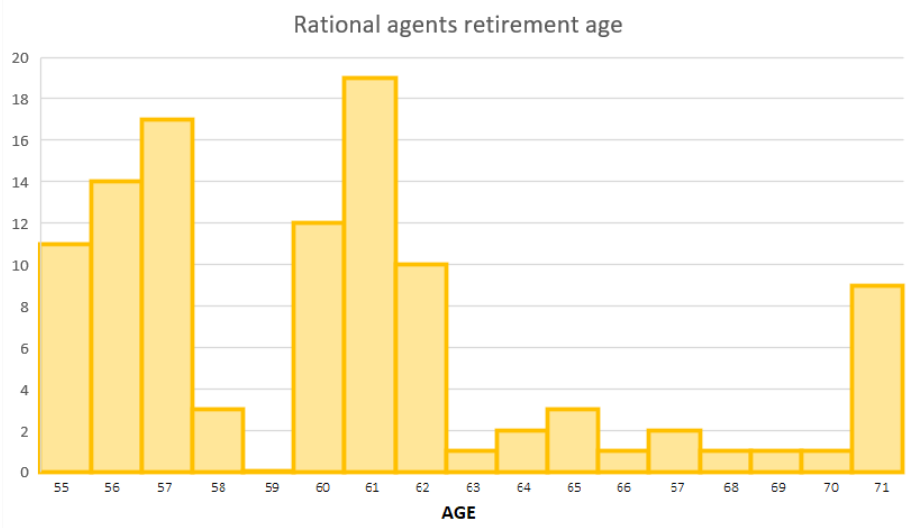

Figure 3.1: **Frequency of rational agents retirement age (25 ticks)**

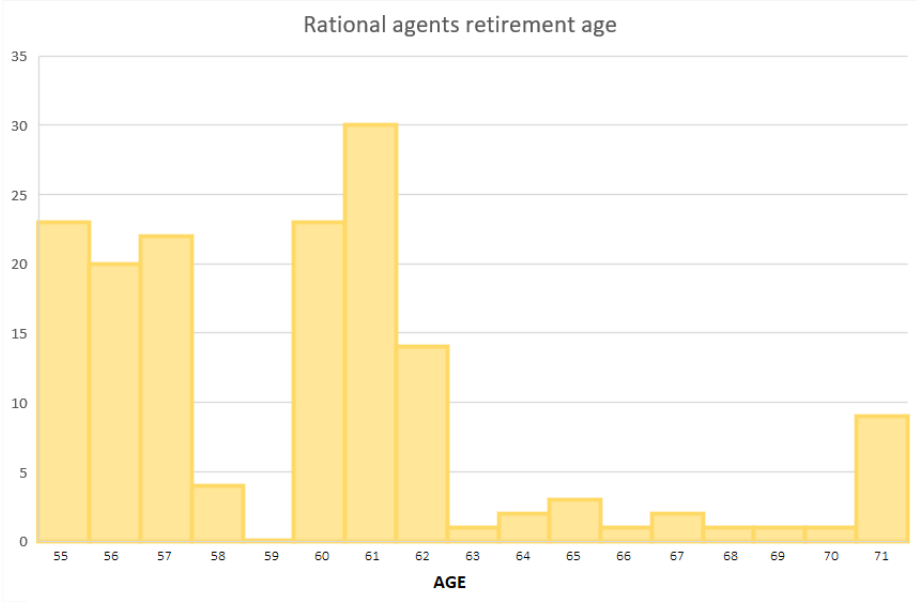

Figure 3.2: **Frequency of rational agents retirement age (50 ticks)**

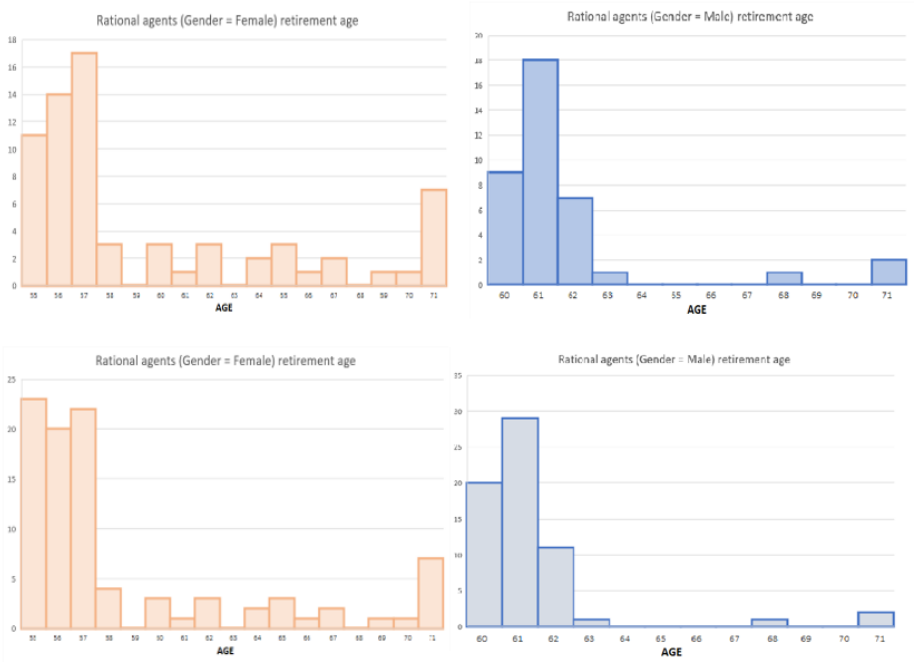

Figure 3.3: **Frequency of rational agents retirement age (25 ticks above and 50 ticks below)**

creates a certain number of agents with $55 \leq age \leq 70$ that could become eligible later then we expect from the eligibility criteria, but this happens just because these agents are created older than the necessary to become eligible. In fact, when they become eligible, follow the same criteria of other agents and they do not affect the average of the eligibility counter. This effect is muted in the 50 ticks average because of the big number of data, for this reason we see this decreasing trend;

- the average of the eligibility counter of female is strictly greater than the eligibility of male. A reason could be that the eligibility criteria allows the female agents to be eligible when they are younger with respect to the male agents. Since one of the eligibility criteria is age, if an agent becomes eligible younger, the age effect will be smaller and so the total score gained at each step will be lower;

- the average age of retirement of male rational agents is equivalent to the one of the hasty agents. This fact highlights a problem of this model: the criteria used to assign the scoring to each parameter in section 1.2 have a high degree of arbitrariness. To determine the value of these parameters in a quantitative way would be a very interesting work that unfortunately I did not have time to investigate.

We could see the responsiveness of the agents in income and weary or non-weary work by separating rational agents in sub-category by using as criteria income or work (I decided to show the results for 50 ticks only to avoid overloading this

section). The results are shown in Table 2 and Table 3.

Table 2	Rational agents				
	Age of retirement [years]	σ_{Age} [years]	Eligibility [years]	σ_{El} [years]	Work
50 ticks (Female)	57.7	0.5	2.6	0.1	White-collar
50 ticks (Male)	61.6	0.5	2.1	0.2	White-collar
50 ticks (Female)	59.2	0.8	1.6	0.1	Blue-collar
50 ticks (Male)	61.2	0.3	1.5	0.1	Blue-collar

As we expect, agents who do a non-weary work (white-collar) have an average eligibility higher than agents who do a weary work. What we do not expect is that retirement age of white-collar female agents is lower than retirement age of blue-collar female agents: this is a contradiction with respect to the eligibility results. To explain this fact, I analysed the rough data discovering that there is a not negligible number of blue-collar female that became eligible quite older, the inevitable effect of this fluctuation is to increase the average age of retirement.
Another thing to observe is that the eligibility of blue-collar is considerably lower than the total eligibility or the eligibility of white-collar even if blue-collar has an average income lower than the white-collar. Instead, we could see now that if we consider the eligibility of agents who have low income (blue-collar + white-collar), even if blue-collar agents are the majority, the eligibility is higher than the all agent average eligibility because in this case white-collar agents are doubly encouraged to keep working.

Table 3	Rational agents				
	Age of retirement [years]	σ_{Age} [years]	Eligibility [years]	σ_{El} [years]	Income
50 ticks (Female)	59	1	2.2	0.2	Low
50 ticks (Male)	61.8	0.5	2.0	0.1	Low
50 ticks (Female)	59.0	0.7	1.7	0.1	Medium
50 ticks (Male)	60.9	0.3	1.4	0.1	Medium
50 ticks (Female)	57.1	0.7	2.3	0.2	High
50 ticks (Male)	61.0	0.3	2.0	0.3	High

We could observe from this table that the agents with the lowest eligibility are the medium income agents. We expected it because medium income does not affect very much the retirement decision, moreover many of these agents are blue-collar so agents very motivated to retire. High income agents even if they are encouraged to retire by their high income, we have to remember that they are all white-collar agents. So the sum of these two features gives us an eligibility quite high to this sub-category.
Another way to investigate the effect of each parameter on the retirement decision of rational agents is to turn off one parameter to each simulation and analyse the changing with respect to the results in which all the parameters are turned on.

Table 4	Rational agents				Hasty agents				
	Age of retirement [years]	σ_{Age} [years]	Eligibility [years]	σ_{El} [years]	Age of retirement [years]	σ_{Age} [years]	Eligibility [years]	σ_{El} [years]	Parameter set on "False"
50 ticks (Female)	57.3	0.4	1.8	0.1	56.8	0.5	-	-	Work
50 ticks (Male)	61.5	0.3	1.8	0.1	61.3	0.4	-	-	Work
50 ticks (Female)	57.4	0.4	1.8	0.1	56.7	0.5	-	-	Social Network
50 ticks (Male)	61.6	0.3	1.8	0.1	61.1	0.4	-	-	Social Network
50 ticks (Female)	57.7	0.4	2.1	0.1	57	1	-	-	Health
50 ticks (Male)	61.6	0.4	1.4	0.1	60.8	0.3	-	-	Health
50 ticks (Female)	57.6	0.4	2.3	0.1	56.9	0.6	-	-	Family-Income
50 ticks (Male)	62.4	0.5	1.5	0.2	61.1	0.4	-	-	Family-Income
50 ticks (Female)	58.1	0.5	1.9	0.1	57.3	0.6	-	-	Random
50 ticks (Male)	61.0	0.3	1.4	0.1	61.1	0.4	-	-	Random

The most curious thing of this results is the inverse proportionality of eligibility between male and female with respect to the data of Table 1: when the eligibility of female agents in Table 4 decreases with respect to the eligibility of female agents in Table 1, the eligibility of male agents in Table 4 increases with respect to the eligibility of male agents in Table 1 and vice versa (the only exception is for "parameter set on False: Random", but just because of the random attitude of this case I assume it has no meaning). The explanation of this fact could be found in the role played by age in the scoring system that determine the retirement of rational agents, unfortunately I cannot prove it.

Chapter 4

Conclusions

This is a rough model that simulate the retirement decision making, that's why my results are so qualitative. My work is based on the complete and exhaustive work of M. Borella, F. Coda Moscarola [6]. To improve the results of my model and to make it more quantitative, we should determine in a quantitative and strict way the parameters of the scoring system. After this, we could improve the income system by substituting the values "low", "medium", "high" with the statistics of Italian salary and wealth and by implementing a 2008 reform compatible conversion system from salary to pension (as M. Borella and F. Coda Moscarola did in their paper). With these implementations it would be possible even investigate on the newer "Riforma Fornero" and "Quota 100" just by setting others eligibility criteria and another conversion system from salary to pension. On the other hand, this model could be a good start point to investigate deeply and more quantitatively the retirement decision behaviour of Italian workers.

Bibliography

[1] The social network architecture is inspired by Robert L. Axtell and Joshua M. Epstein, Coordination in Transient Social Networks: An Agent-Based Computational Model of the Timing of Retirement (May, 1999) Available at: https://www.semanticscholar.org/paper/Coordination-in-Transient-Social-Networks\%3A-An-Model-Axtell-Epstein/eb41a2bfa2ed5c97cd652a2168ad3672203694f0

[2] Tasso di occupazione (occupational rate). Available at: http://dati.istat.it/Index.aspx?DataSetCode=DCCV_TAXDISOCCU1

[3] "populate_world"'s code is taken from: Pietro Terna, CDA_trend_model. Available at: https://terna.to.it/econophysics19/NetLogo_examples/CDA_trend_model.nlogo

[4] Gender and age distribution in the model are assigned by following the 2017 ISTAT data. Available at: http://demo.istat.it/

[5] Margherita Borella, Flavia Coda Moscarola, Microsimulation of pension reforms: behavioural versus nonbehavioural approach (November 23, 2009); Table1. Main characteristics of the pension reforms in Italy (page 586). Available at: https://www.cambridge.org/core/journals/journal-of-pension-economics-and-finance/article/microsimulation-of-pension-reforms-behavioural-versus-nonbehavioural-approach/EC3C877EB14440BA01FCA6740C22FDFC

[6] Margherita Borella, Flavia Coda Moscarola, Microsimulation of pension reforms: behavioural versus nonbehavioural approach (November 23, 2009). Available at: https://www.cambridge.org/core/journals/journal-of-pension-economics-and-finance/article/microsimulation-of-pension-reforms-behavioural-versus-nonbehavioural-approach/EC3C877EB14440BA01FCA6740C22FDFC