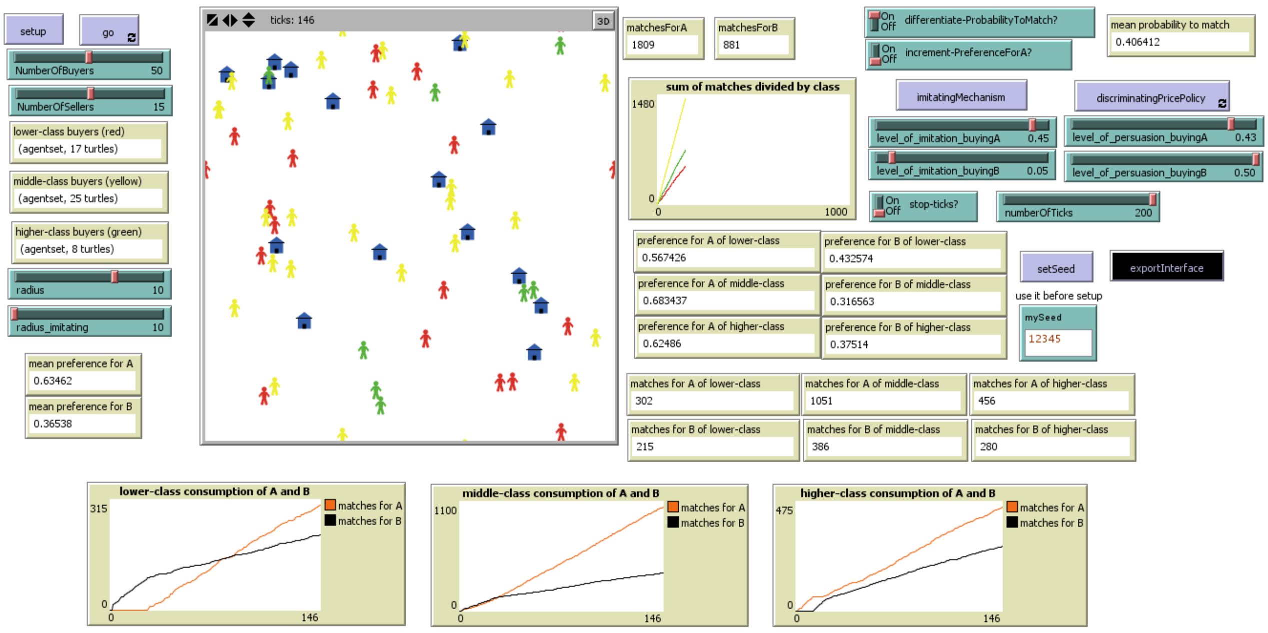
SIMULATION MODELS FOR ECONOMICS
A.Y. 2014/2015

Consumers' behavior


setup
go
ticks: 146
3D
imitatingMechanism
discriminatingPricePolicy
setSeed
exportInterface
use it before setup
sum of matches divided by class
1480
0
0
1000
lower-class consumption of A and B
315
146
middle-class consumption of A and B
1100
higher-class consumption of A and B
475
matches for A
matches for B

NumberOfBuyers 50
NumberOfSellers 15
lower-class buyers (red) (agentset, 17 turtles)
middle-class buyers (yellow) (agentset, 25 turtles)
higher-class buyers (green) (agentset, 8 turtles)
radius 10
radius_imitating 10
mean preference for A 0.63462
mean preference for B 0.36538
matchesForA 1809
matchesForB 881
On Off differentiate-ProbabilityToMatch?
On Off increment-PreferenceForA?
mean probability to match 0.406412
level_of_imitation_buyingA 0.45
level_of_imitation_buyingB 0.05
level_of_persuasion_buyingA 0.43
level_of_persuasion_buyingB 0.50
On Off stop-ticks?
numberOfTicks 200
preference for A of lower-class 0.567426
preference for B of lower-class 0.432574
preference for A of middle-class 0.683437
preference for B of middle-class 0.316563
preference for A of higher-class 0.62486
preference for B of higher-class 0.37514
mySeed 12345
matches for A of lower-class 302
matches for A of middle-class 1051
matches for A of higher-class 456
matches for B of lower-class 215
matches for B of middle-class 386
matches for B of higher-class 280

A work of
Marcello Matranga and Eleonora Priori

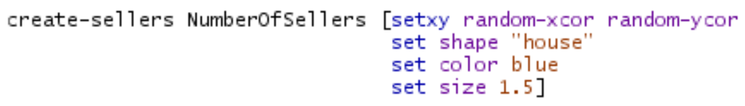
Index

Introduction

We have simulated a simplified market where only two kinds of the same variety of product exist: product A represents the high-quality and expensive typology and product B represents the low-quality and cheap typology.
Buyers are divided in classes on the basis of their wealth, although wealth actually never appears in the model since it is only represented by a color: in fact low-class buyers are signed in red, middle-class buyers are signed in yellow and high-class buyers are signed in green. Each buyer owns some features, which basically are the probability to match, the preference for product A and the preference for product B (which are complements).
The model is based on the assumption that prices, which actually never appear, are latent variables, which partly determine the level of preference for the two products.
Imitating mechanisms among buyers are allowed in the model and therefore the choices of each buyer affect the decisions of other agents. Representing this process with a correct specification of the parameters help us simulate a model as realistic as possible.
The aim of our work is analyzing consumers' behavior when a discriminating price policy is implemented. This mechanism allows a paradox, that is: buyers can pay different prices to get the same product since they are "discriminated" on the basis of their usual preferences through the data which the fidelity card registers at each match.
In fact, in the model sellers assign buyers a fidelity card after their third match: the fidelity card registers which kind of product (A or B) buyers choose at each match and proposes buyers
some discounts (which appear as changes in the level of preferences) on the basis of their previous choices. The particular feature is that the fidelity card proposes discounts which do not refer to the product that buyers usually buy but reduces the price of the other one (i.e. it enlarges the level of preference displayed for the product that a given agent under "normal" conditions would not buy). This allows us to check whether and in which measure the prices level modifies the choices of buyers, by evaluating the sensibility of buyers' preferences with respect to the price.
This model aims to underline how buyers perceive the gap between the prices of the two typologies of products and the related gap between the quality levels of the two products sold in the market. Rather unsurprisingly, we will discover that each class of buyers is sensitive to changes in prices and therefore we will focus our analysis in their consumption decisions under given conditions.
In further works it would be interesting introducing in the simulation prices as a variable, in order to observe whether such a price policy, which is currently being implemented by some companies (mainly supermarkets) in USA and Germany, is really effective in producing a real reduction of the general price level.
Our work is only focused on explaining consumers' behaviors in order to check how they make their choices if some features of the economic system which they act into change.

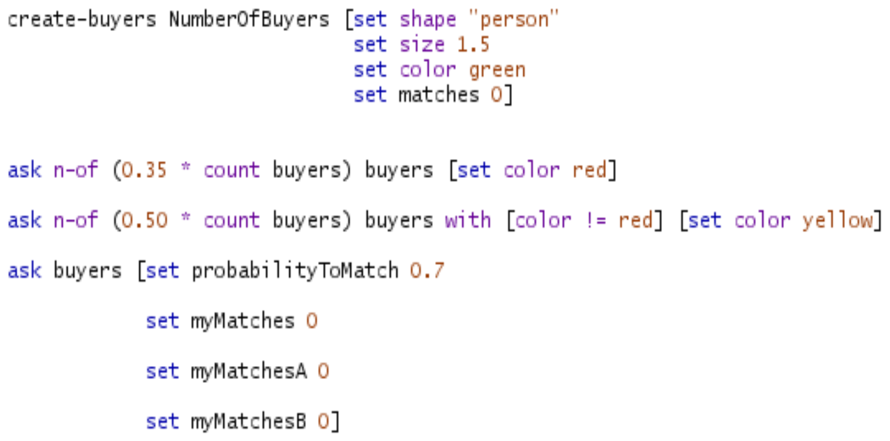
1. The code

By using NetLogo we have built a model which displays some consumers' behaviors in a simplified market where only two typologies of the same variety of product exist.
We have created two breeds: sellers and buyers. By pressing the "setup" button, we can observe sellers and buyers and their main features.
Sellers sell two different kinds of the same product. Product A represents the high-quality (i.e. the more expensive) typology, whereas product B represents the low-quality (i.e. the cheaper) typology of the - unique- product sold in the market. Actually prices never appear in the model as features of the products because we have focused our work on preferences and on their changes assuming that preferences are partly determined by prices, which therefore are latent variables. Sellers are asked to spread randomly in the display in the interface.



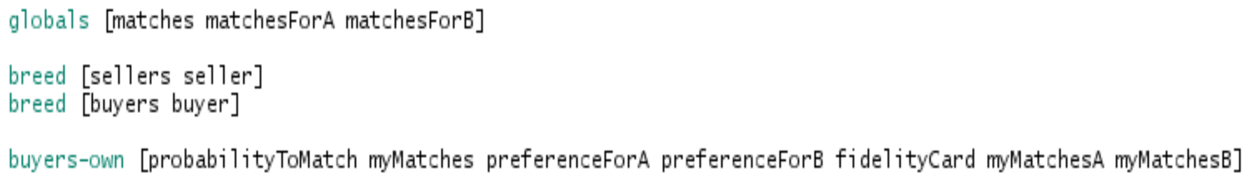
```
create-sellers NumberOfSellers [setxy random-xcor random-ycor
                                set shape "house"
                                set color blue
                                set size 1.5]
```

Buyers are divided in three classes on the basis of their wealth: a lower class (which is signed in red and represents the 35% of the entire buyers' population), a middle-class (which is signed in yellow and represents the 50% of the population) and a higher class (which is signed in green and represents the left 15%). Each class of buyers is endowed with some features which depend on the wealth (i.e. on the color of the class). These features basically are the probability to match and the preference for product A or for product B.



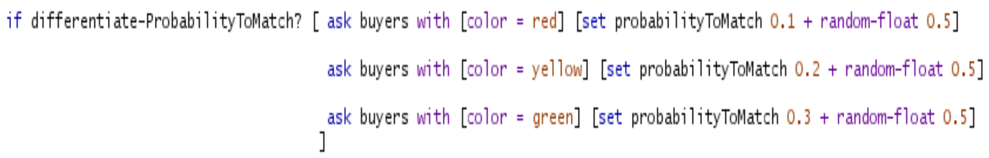
```
create-buyers NumberOfBuyers [set shape "person"
                              set size 1.5
                              set color green
                              set matches 0]

ask n-of (0.35 * count buyers) buyers [set color red]

ask n-of (0.50 * count buyers) buyers with [color != red] [set color yellow]

ask buyers [set probabilityToMatch 0.7

            set myMatches 0

            set myMatchesA 0

            set myMatchesB 0]
```

In order to make the model a little more manageable, we have set the number of matches (both as a total value and as a value divided on the basis of the product chosen by buyers) both as global variables and as local variables which belong only to buyers.



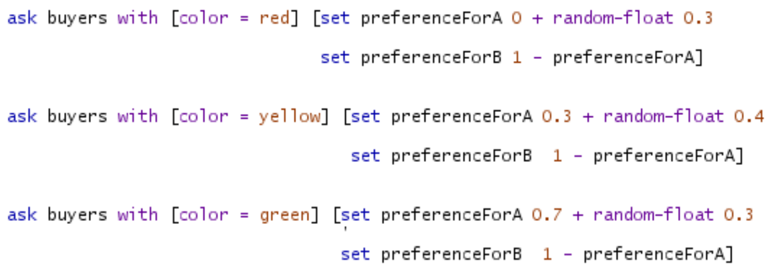
```
globals [matches matchesForA matchesForB]

breed [sellers seller]
breed [buyers buyer]

buyers-own [probabilityToMatch myMatches preferenceForA preferenceForB fidelityCard myMatchesA myMatchesB]
```

In the basic version of the code all buyers own the same probability to match (which is arbitrarily set at a 0.7 level). In this way buyers match at each tick with their counterparts since if the probability to match is greater than 0.5 buyers conclude positively their match with sellers, whereas if it is smaller than 0.5 there is no match.
By turning on the switch related to the differentiation of the probability to match, each buyer owns a different (and personal) probability to match. The probability to match is still a value bounded between zero and this value is partly given by the class which the buyers belong to and partly randomly determined. Each class of buyers assigns buyers a fixed starting value (which is 0.1 for the lower class, 0.2 for the middle class and 0.3 for the higher class), but the left part of the probability to match is determined randomly with values bounded between zero and 0.5 (with no differences among the three classes). This mechanism allows each buyer to own a different probability to match.
This procedure is called by the "go" button in order to allow the observers to change the features of the model while running it and not only in the "setup" phase.



```
if differentiate-ProbabilityToMatch? [ ask buyers with [color = red] [set probabilityToMatch 0.1 + random-float 0.5]

                                        ask buyers with [color = yellow] [set probabilityToMatch 0.2 + random-float 0.5]

                                        ask buyers with [color = green] [set probabilityToMatch 0.3 + random-float 0.5]
                                       ]
```

As regard as preferences we have set the model in this way: preference for A is a value bounded between zero and one and preference for B is its complement. Each buyer decides whether buying product A or product B on the basis of the higher preference he displays, conditional to the fact that his probability to match has to be greater than 0.5 in order to conclude positively the exchange.
Since product A is assumed to be the expensive and high-quality version of the product, whereas product B is assumed to be the cheap and low-quality typology of product, low-class buyers are expected to display lower preferences for A and higher preferences for B, high-class buyers are expected to display higher preferences for A and low preferences for product B and the middle class is expected to split its preferences between product A and product B. Therefore we have set the values of the preferences as follows.



```
ask buyers with [color = red] [set preferenceForA 0 + random-float 0.3
                                set preferenceForB 1 - preferenceForA]

ask buyers with [color = yellow] [set preferenceForA 0.3 + random-float 0.4
                                   set preferenceForB  1 - preferenceForA]

ask buyers with [color = green] [set preferenceForA 0.7 + random-float 0.3
                                  set preferenceForB  1 - preferenceForA]
```

In order to display how the consumers' choices change when preferences change, we have created a switch which allows to increment the preferences for A. This feature is very interesting in order to observe combined effects of changes in variables. This procedure is called by the "go" button for the same reasons of the previous case (probability to match).

```
if increment-PreferenceForA? [ask buyers with [color = red] [set preferenceForA 0.2 + random-float 0.4
                                                              set preferenceForB 1 - preferenceForA]
                              ask buyers with [color = yellow] [set preferenceForA 0.4 + random-float 0.3
                                                                 set preferenceForB 1 - preferenceForA]
                              ask buyers with [color = green] [set preferenceForA 0.9 + random-float 0.1
                                                               set preferenceForB 1 - preferenceForA]
                              ]
```

The “go” button is a continuous procedure and it calls two procedures, which are: “move and match” and “get a fidelity card”. We have created a switch which allows stopping the running of the model at a given number of ticks in order to observe experiments in a consistent way.

```
to go

  tick
  moveAndMatch
  getFidelityCard

  if stop-ticks? [if ticks >= 200 [stop]]

end
```

The “move and match” procedure is crucial in order to observe how matches occur. Buyers exchange with sellers when they are in a given radius of distance, conditional to the fact that their probability to match has to be higher than 0.5. When these conditions are satisfied the match occurs and therefore we can upgrade the values of the matches (both as global variables and as local variables belonging to buyers).

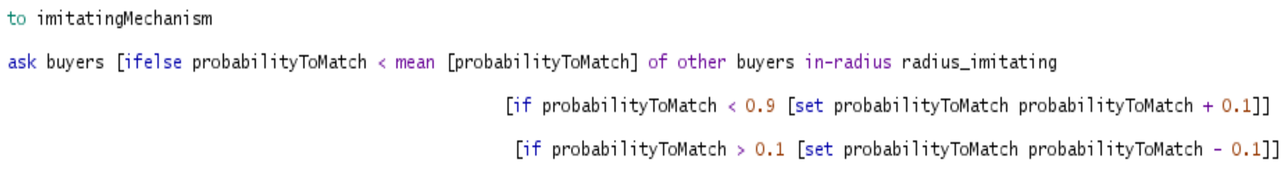
```
to moveAndMatch

  ask buyers [setxy random-xcor random-ycor]


  ask buyers [let mySeller sellers in-radius radius

             if mySeller != nobody and probabilityToMatch > 0.5 [set matches matches + 1

                                                                 set myMatches myMatches + 1

                                                                 ifelse preferenceForA > preferenceForB [set matchesForA matchesForA + 1

                                                                                                         set myMatchesA myMatchesA + 1]

                                                                                                        [set matchesForB matchesForB + 1

                                                                                                         set myMatchesB myMatchesB + 1]
                                                                ]
            ]
```

As we said before, the procedures for “differentiate probability to match” and for “increment preference for A” are called by the “go” button and, more precisely, the lines of code related to these switches are contained in the “move and match” procedure.
The “get fidelity card” procedure is very simple and it allows us to experience the discriminating price policy only on buyers who have yet matched with sellers at least three times.

```
to getFidelityCard

  ask buyers with [myMatches >= 3] [set fidelityCard 1]

end
```

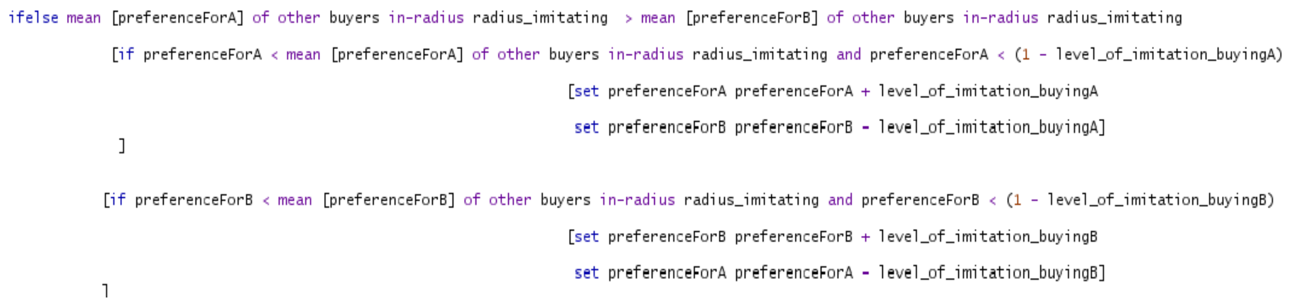
As we can see by observing the interface, other two buttons which trigger interesting procedures exist.

The first one is the “imitating mechanism” button. This button allows buyers in the model to imitate the behavior of other buyers in a given radius of distance. The imitating mechanism affects both the probability to match and the level of preferences.
The imitating mechanism about the probability to match allows each buyer to compare its probability to match with the mean of that one of all the other buyers. If the probability to match of this agent is lower than the mean, he is asked to adjust its probability to match enlarging it of 0.1. If otherwise this agent displays a probability to match greater than the mean, he is asked to adjust it reducing it of 0.1. We set this feature in order to keep the value bounded between zero and one.



```
to imitatingMechanism

ask buyers [ifelse probabilityToMatch < mean [probabilityToMatch] of other buyers in-radius radius_imitating

                                                 [if probabilityToMatch < 0.9 [set probabilityToMatch probabilityToMatch + 0.1]]

                                                  [if probabilityToMatch > 0.1 [set probabilityToMatch probabilityToMatch - 0.1]]
```

We have worked in a similar way in order to set the imitating mechanisms about preferences. In fact each buyer observes whether other buyers prefer product A or product B by comparing the mean of the preferences of all the other buyers in the model. If the mean of preferences for A is greater than the mean of preferences for B and if the agent displays a level of preferences for A lower than the mean, the agent is asked to enlarge its level of preference for A (and therefore to reduce its level of preference for B) of a given value which can be set via a slider in the interface (this value is bounded between zero and 0.5 in order to make things manageable).
Vice versa, if the mean of the preferences for A is smaller than the mean of the preferences for B and the level of preference for B of the buyer is lower than the mean, the agent adjusts its preferences enlarging preference for B and reducing preference for A. Again, we set this feature in order to keep the two level of preferences bounded between zero and one.



```
ifelse mean [preferenceForA] of other buyers in-radius radius_imitating  > mean [preferenceForB] of other buyers in-radius radius_imitating

         [if preferenceForA < mean [preferenceForA] of other buyers in-radius radius_imitating and preferenceForA < (1 - level_of_imitation_buyingA)

                                                 [set preferenceForA preferenceForA + level_of_imitation_buyingA

                                                  set preferenceForB preferenceForB - level_of_imitation_buyingA]

          ]

        [if preferenceForB < mean [preferenceForB] of other buyers in-radius radius_imitating and preferenceForB < (1 - level_of_imitation_buyingB)

                                                 [set preferenceForB preferenceForB + level_of_imitation_buyingB

                                                  set preferenceForA preferenceForA - level_of_imitation_buyingB]

        ]
```

The “imitating mechanism” button is not continuous but affects the values of the model only once. If it were continuous, it would make the model to collapse forcing buyers to choose only one of the two products by triggering a self-sustaining mechanism which enlarges the preference for only one of the two products and reduces the other one at each tick. In order to observe how the consumers’ behaviors change after running the imitating mechanism we have to keep the “go” button on.

The last button triggers a “discriminating price policy”. Such a policy is implemented via the fidelity card (which is assigned to buyers after their third shop and registers which of the two products is preferred by each buyer). The particular feature of this discrimination price policy is that the fidelity card allows buyers to access some discounts on the product that they usually would not buy without the discount: i.e. if a buyer, for instance, usually prefers (and buys) product B, the fidelity card registers this fact and proposes him a discount on product A (and vice versa). Since in our model prices do not exist, the discount is shown as a change in the preferences of the buyer, which are incremented by a given level (which can be chosen via a slider in the interface).

```
to discriminatingPricePolicy

   tick
   ask buyers with [fidelityCard = 1]
                              [ifelse myMatchesA > myMatchesB

                                         [if preferenceForB < (1 - level_of_persuasion_buyingB) [set preferenceForB preferenceForB + level_of_persuasion_buyingB
                                                                                                 set preferenceForA preferenceForA - level_of_persuasion_buyingB]
                                         ]
                                         [if preferenceForA < (1 - level_of_persuasion_buyingA) [set preferenceForA preferenceForA + level_of_persuasion_buyingA
                                                                                                 set preferenceForB preferenceForB - level_of_persuasion_buyingA]
                                         ]
                               ]
```

Since the “discriminating price policy” button is continuous, we have to press it after switching the “go” button off and, for the same reason, the lines of the code related to this procedure contain the same instructions that we used in the “move and match” procedure in order to ask buyers to match with sellers.

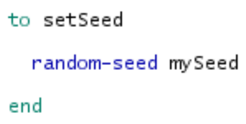
```
ask buyers [setxy random-xcor random-ycor
            let mySeller sellers in-radius radius
             if mySeller != nobody and probabilityToMatch > 0.5 [set matches matches + 1
                                                                  set myMatches myMatches + 1
                                                                  ifelse preferenceForA > preferenceForB [set matchesForA matchesForA + 1
                                                                                                          set myMatchesA myMatchesA + 1]
                                                                                                         [set matchesForB matchesForB + 1
                                                                                                          set myMatchesB myMatchesB + 1]
                                                                 ]
```

In order to sketch an exhaustive picture of the code, we have to explain the features of other two buttons which are not relevant from an economic point of view, but seem to be very useful.

The first one is the “export interface” button: by clicking it we can get a screenshot of the interface which will be automatically saved on the desktop of the user. Since the interface contains many monitors and graphs explaining the trend of the model, taking a photo of these outputs in different moments help us compare the results of the experiments that we can run.

```
to exportInterface
  export-interface " CounsumersBehavior.png"
end
```

The last lines of the code are devoted to the “set seed” button. This button allows the user to replicate an experiment choosing the same random values, and therefore under the same conditions, by choosing the seed via an output in the interface. Users have to press this button before setting up the model.



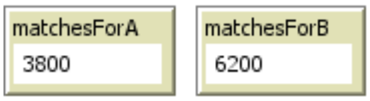
```
to setSeed
  random-seed mySeed
end
```

2. The experiments

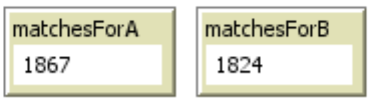
The aim of our work is to show how the buyers' behavior changes as some features of the model change. In order to do so, we have chosen to keep the same seed (which is arbitrarily set at 12345) so that we can compare in a consistent way the results of the tests we have experienced.

2.1. Probability to match

The first experiment is about the probability to match of the buyers.
By keeping fixed the seed and repeating the same experiment with the probabilities to match switched off and on, we can observe how the number of matches is drastically reduced in the second case.

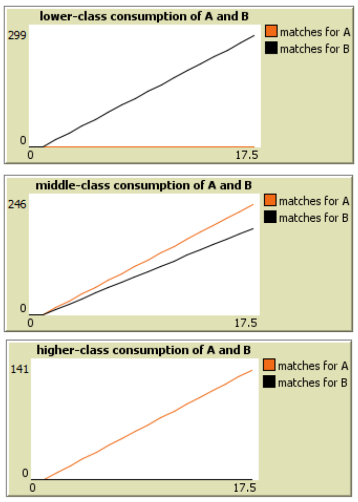


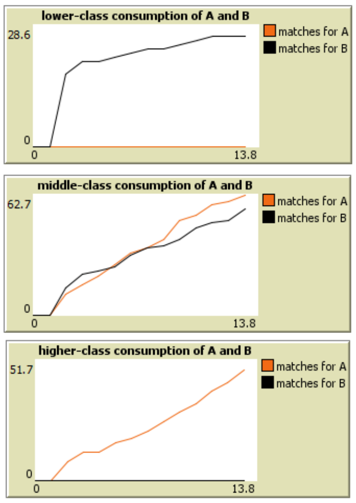
matchesForA	matchesForB
3800	6200



matchesForA	matchesForB
1867	1824

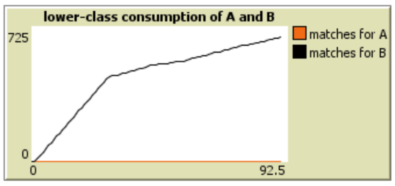
There are also significant changes in the composition of the consumption of the whole population. In fact the number of matches for product A is higher than the number of matches for product B since high-class buyers, who prefer product A, match with sellers more often than low-class buyers, who generally prefer product B.
We can observe that, running the model keeping the differentiation of the probability to match switched off, there is no track of noise. Instead, if we turn the switch linked to the probability to match on, we can observe that the graph describing the consumption of product A and product B divided by class displays a huge element of disturbance: the trend is no more represented by a straight line but it consists in an indented line.

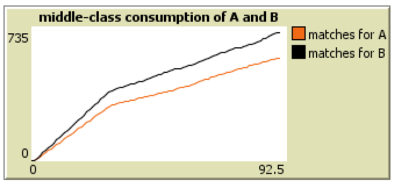


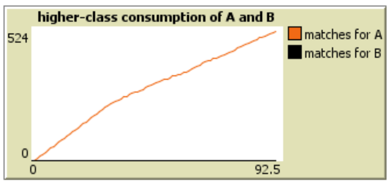


.

As we can observe in the following pictures that we are attaching, the slope of the trend describing the consumption is dramatically reduced as we turn the switch of the probability to match off because buyers do not match at each tick as they did when the switch of probability to match was turned off.

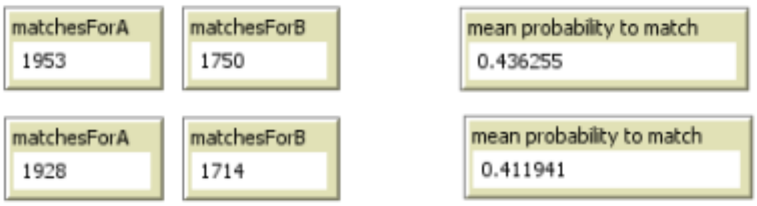






2.2. Imitating mechanisms

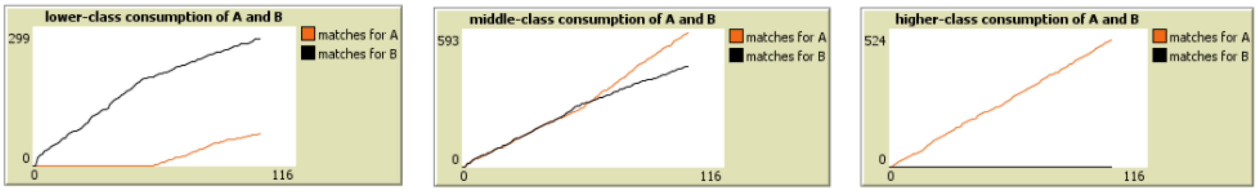
The second experiment is about imitating mechanisms among buyers. We have set the imitating mechanisms so that they affect both the probability to match and the preferences and we have created a button which activates the experiment. The button is not continuous since if it were continuous, it would make the model to collapse forcing buyers to choose only one of the two products by triggering a self-sustaining mechanism which enlarges the preference for only one of the two products and reduces the other one at each tick. Keeping the probability to match switched off and the two sliders which determine the level of imitation about preferences set at a zero level, imitating mechanisms do not affect the model.
In order to observe the effects of the imitating mechanism affecting only the probability to match, we have to switch the differentiation of the probability to match on and to keep at zero the two sliders related to the level of preferences for A or for B.
In order to compare the results, we run the model twice keeping the same seed: the first time we run the model with no imitating mechanisms and the second one we activate this procedure before running the model instead.
The results are straightforward: we can observe how the number of matches for A and for B (and therefore the total sum of the matches) is lower in the second case and we can observe that this fact happens because in the second case the mean of the probability to match displays lower values than the values displayed by the "control group".


matchesForA 1953 | matchesForB 1750 | mean probability to match 0.436255

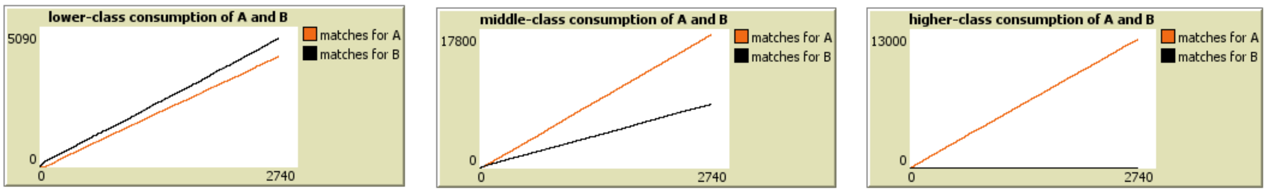
matchesForA 1928 | matchesForB 1714 | mean probability to match 0.411941

The second part of the experiment related to the imitating mechanisms displays how buyers' decisions change if we allow imitating mechanisms about preferences.

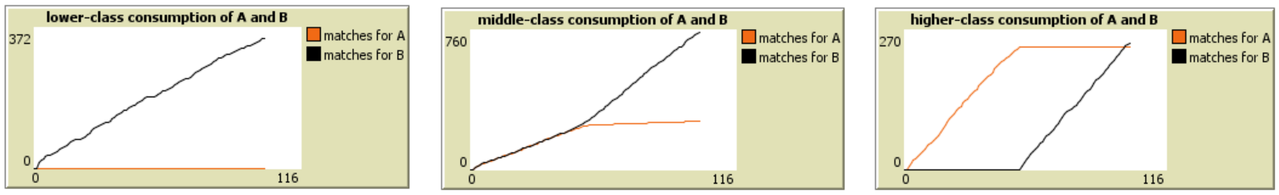
In order to observe the results of this experiment, we run the model for the first 50 ticks by keeping only the probability to match switched on. After the first 50 ticks, we press the button which activates the imitating mechanism about the preferences by choosing a level of imitation for product A equal to 0.50 (which is the maximum level we set) and a level of imitation for B equal to zero. Rather unsurprisingly, by keeping the level of imitation of one of the two products set at zero and enlarging the other one, buyers' consumption composition drastically change. We can observe that the level of consumption for A of the lower class starts increasing, whereas for the middle-class we can observe that the matches for A overcome the matches for B. As regard the higher class, we have registered no significant changes because the matches for B were at a zero level yet.



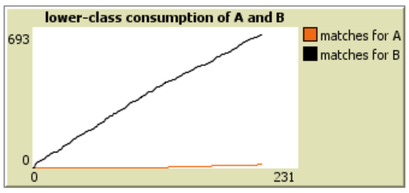
As we can see by incrementing the number of the ticks (we have run the model until 2540 ticks), in the long run the trend which describes the consumption of A and B for the lower class keeps a constant pattern and the consumption of A is not expected to overcome the consumption of B. As regard the high-class consumption, again we do not register significant changes. Instead we can observe how the two patterns of the middle-class consumption gradually diverge. This experiment highlights how we should focus our analysis on the behaviors of the middle-class consumption since it displays the most meaningful results.

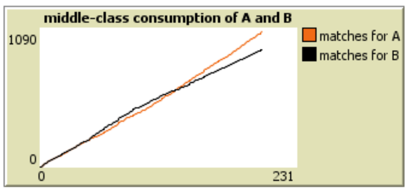


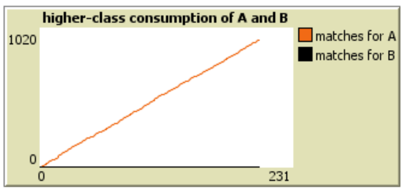
By running the model in the opposite situation (i.e. setting the level of imitation for A equal to zero and the level of imitation for B at 0.50), we don't obtain specular results since the low-class members are numerically more and therefore the weight of the influence of the low class is higher. This causes that the matches for A of the middle class slows down faster and the same happens for the high class, whereas the low class displays the same behavior we observed before for the high class since low-class buyers would not buy product A anyway.



Trivially, setting the same positive level in the two sliders, the number of matches for B is much higher than the number of matches for A since the low class is numerically larger.
As we said before, we have found out that the most meaningful results are displayed by the middle-class buyers' behaviors and therefore we pick another experiment: by assuming that the larger part of the consumers imitate the behaviors of the higher classes, but few consumers could imitate lower classes' consumption behaviors, we have set the level of imitation for A at 0.25 and the level of imitation for B at 0.05 in order to replicate a situation as realistic as possible.
In this case the results display that, introducing imitating mechanisms after 100 ticks, the patterns of the middle-class consumption result to be reversed since after activating the procedure, we can observe an increment in consumption for product A, whereas before middle-class buyers used to prefer product B.

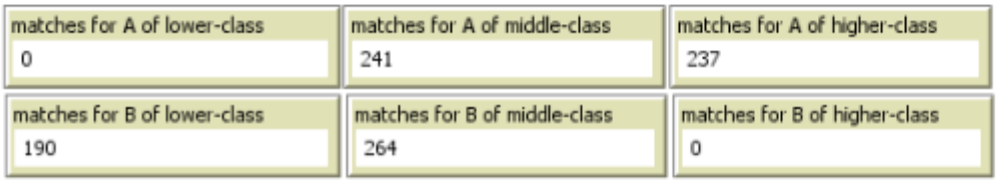




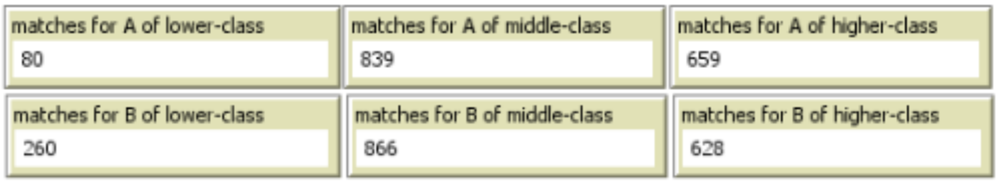


2.3. Discriminating price policy

The third experiment is about discriminating price policy.
In order to run the model in a correct way, we have to run the first ticks by pressing the "go" button: after the first 50 ticks we switch the "go" button off and we press the "discriminating price policy" button. Since this button is continuous it also contains the instructions contained in the "move and match" procedure.
By pressing the "discriminating price policy" button and by setting the wished level of persuasion for product A and/or for product B, we can observe how buyers modify their level of preferences and start buying the product that they did not buy before. In this case we are keeping the maximum level of persuasion both for product A and product B (i.e. 0.5) and the results are straightforward: low-class buyers would not buy product A and high-class buyers would not buy product B under "normal" conditions, but they start buying the other product since we have implemented the discriminating price policy.

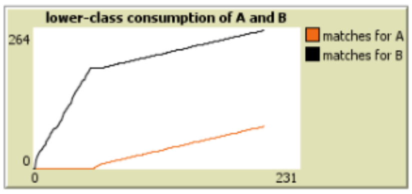


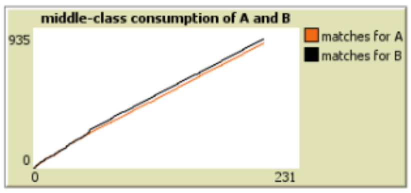
matches for A of lower-class	matches for A of middle-class	matches for A of higher-class
0	241	237
matches for B of lower-class	**matches for B of middle-class**	**matches for B of higher-class**
190	264	0

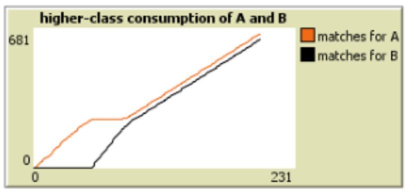


matches for A of lower-class	matches for A of middle-class	matches for A of higher-class
80	839	659
matches for B of lower-class	**matches for B of middle-class**	**matches for B of higher-class**
260	866	628

We can analyze this output also by observing the graphs of the consumption divided by class, which highlights an interesting result: here we should not focus our attention on the middle class because they are not affected by this policy since their consumption was yet split between product A and product B, whereas it is much more interesting analyzing the behaviors of the low class and the high class.

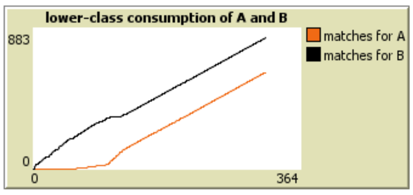


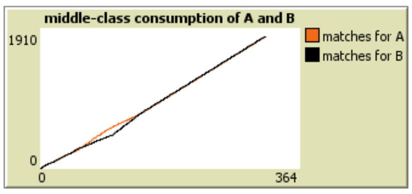


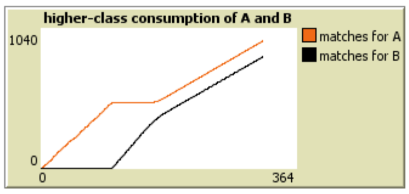


2.4. The combined effect of imitating mechanisms and discriminating price policy

In order to provide a general framework we have decided to implement a double experiment which displays the effects of two joint phenomena, i.e. the imitating mechanisms and the discriminating price policy.
As we said before, for the imitating mechanisms we have chosen the value of 0.25 as regard the level of imitation of product A and a 0.05 level for the imitation in buying B because these values reflect what could be a realistic situation.
As regard the discriminating price policy, we have set the level of persuasion of buying A at the maximum level (i.e. 0.5) and the level of persuasion of buying B at 0.25, since we believe that sellers should offer a high incentive to low-class buyers in order to convince them switching their preferences, whereas high-class buyers require only a slight stimulus in order to buy some pieces of product B.
In order to do so, we have run the first 50 ticks under "normal" conditions and the following 50 ticks applying the imitating mechanisms. After that we have activated the "discriminating price policy" procedure and we have observed the following long-run results.







As the graphs show, in the long run each class splits its consumption choices between the two products in a consistent way: the low class displays a slight majority of matches for B but keep on buying some portion of A, the middle class splits its choices between A and B with an equal proportion and the high class displays a slight majority of matches for A but keep on buying B.