Treinamento LSD

Aulas 2-10 – 2ª Parte

Baseado nos slides de Marco Valente

Agenda

- Aula 2: Linear model and LSD basics
- Aula 3: Random walk model and LSD model structure
- Aula 4: Logistic model and chaotic behavior
- Aula 5: Replicator dynamics model
- Aula 6: Extended replicator dynamics model
- Aula 7: Network externalities model – Part 1
- Aula 8: Network externalities model – Part 2
- Aula 9: Consumers' model – Part 1
- Aula 10: Consumers' model – Part 2

Analysing and extending a model

This lesson answers some of the questions posed by the model developed in the previous lesson. Moreover, shows how a model can easily be extended. The parameter **Quality** is replaced by a variable, represented by a random walk.

The new model becomes a stochastic model, and its results can be tested running several simulations runs and comparing their results.

Excercises

Questions:

1. Why is the result like that? What is the relation between **Quality** and **Sales**?
2. Why $Sales_9$, that is, the variable **Sales** in the 9^{th} firm first increases and then decreases?
3. What is the effect of a different value for **a**?
4. Are there limits to the values of **a** to obtain sensible results?
5. What do you think happens to the sum of all **Sales** in the model?

Replicator Dynamics Model

Question:

1. Why is the result like that? What is the relation between **Quality** and **Sales**?

Answer:

Sales increases with the *percentage* of difference between **Quality** and **AvQuality**

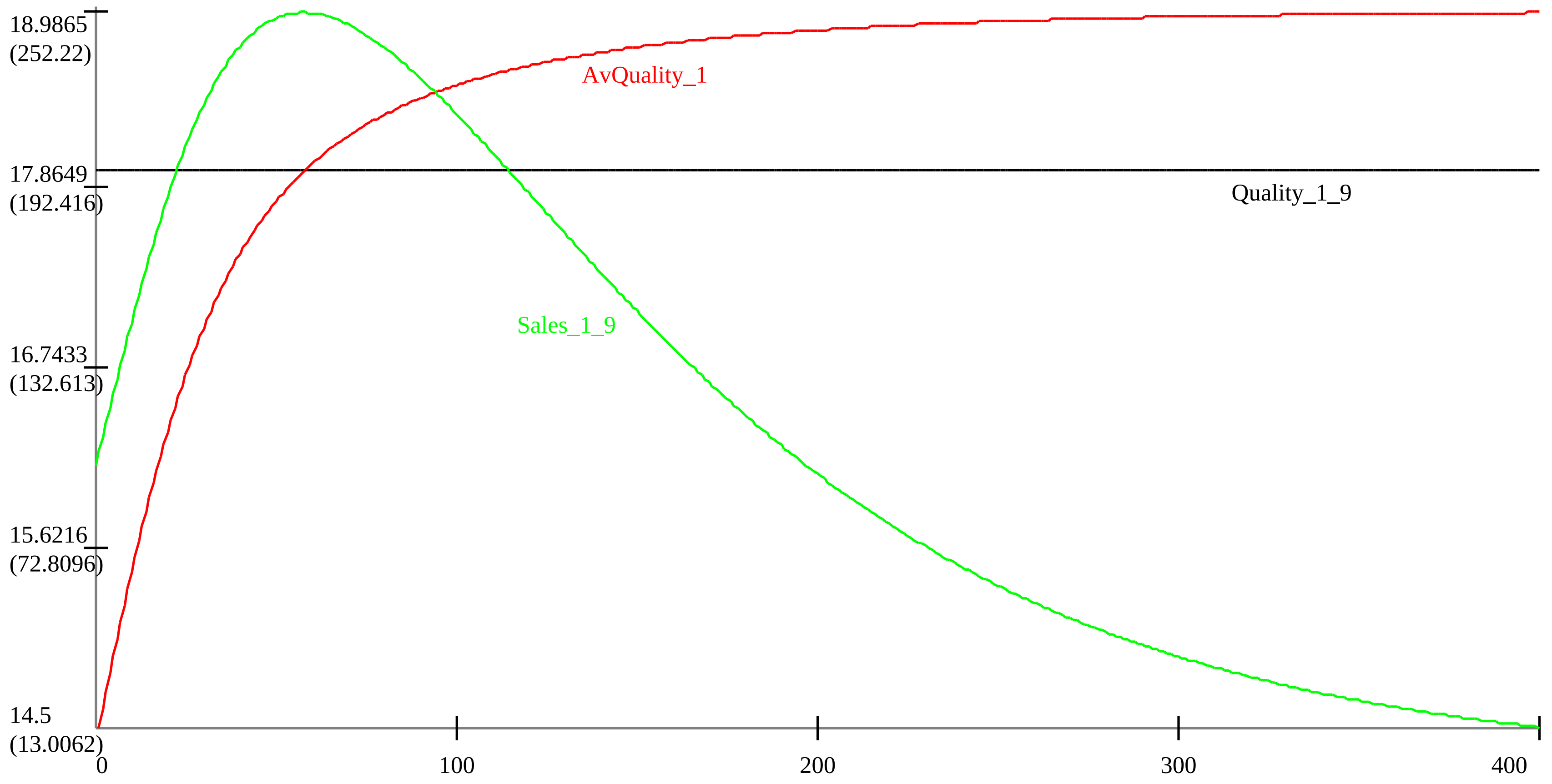
Replicator Dynamics Model

Question:

2. Why $Sales_9$, that is, the variable **Sales** in the 9th firm first increases and then decreases?

Answer:

Quality's are constant but **Sales** are not. Therefore, $AvQuality_t = \frac{\sum_{i=1}^{N} Sales_{t-1}^{i} \times Quality^{i}}{\sum_{i=1}^{N} Sales_{t-1}^{i}}$ changes, increasing since the **Sales** of the higher quality firms increase. Therefore, the $Quality_9$ is before below the average, and after above average.



18.9865
(252.22)
17.8649
(192.416)
16.7433
(132.613)
15.6216
(72.8096)
14.5
(13.0062)
0
100
200
300
400
AvQuality_1
Quality_1_9
Sales_1_9

Replicator Dynamics Model

Question:

5. What do you think happens to the sum of all **Sales** in the model?

Answer:

Remain constant. The increment of one variable is identical to the decrement of the others. Try to introduce a variable summing up all **Sales** to prove it.

Extending the R.Dyn. Model

Let's extend the model. Let's assume that **Quality** is no longer a constant parameter, but a variable.

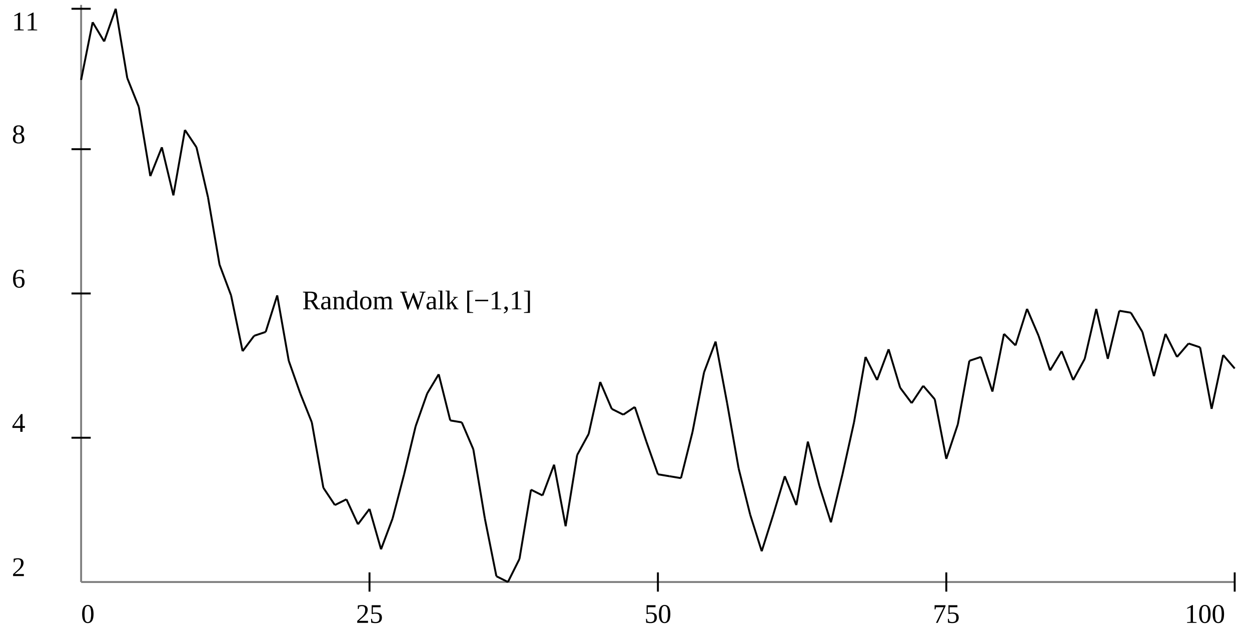
Let's assume that **Quality** changes as a *Random Walk*. A random walk is a variable that changes according to the following function:

$$RW_t = RW_{t-1} + U(-k, +k)$$

where $U(-k, +k)$ is a random number chosen between $-k$ and $+k$.

Extending the R.Dyn. Model

Random walks are frequently used random functions because they change slowly, at most k, but are impossible to predict where they end up (tech. they have infinite variance random variables). They look like many real economic series.



Random events

Randomness cannot be generated by computers. To obviate to this problem there are programs that generate **pseudo** random numbers. That is, series of numbers that appear to come from a random event, though they are generated with rather sophisticated, but deterministic, processes.

The best way to understand what pseudo random numbers are think of sequences of truely random numbers, for example obtained by counting the number of heads tossing a coin 100 times. Different sequences will have different values, but all of them will have common general properties: values between 0 and 100; mean about 50, etc.

The values are naturally random, but they can be repeated exactly. Just take again the same sequence and you will obtain the same values.

Extending the R.Dyn. model

The new equation for **Quality** is:

```
EQUATION("Quality")
/*
Quality level, implemented as a Random Walk
*/
v[0]=VL("Quality",1);
v[1]=V("Max");
v[2]=V("Min");
v[3]=v[0]+UNIFORM(v[1],v[2]);
RESULT(v[3] )
```

Extending the R.Dyn. model

Compile the model and make the following changes:

- Double click on the label for **Quality** and then again on its label, until it allows to be transformed in a variable with 1 lag.
- Add in **Market** the parameters **Min** and **Max**.
- Initialize **Min**=-1 and **Max**=1.
- Initialize $Quality_0 = 10$ in all **Firm**'s.

The model behaves "randomly". But repeating a simulation, the model replicates exactly the same results. Not that random, after all.

Seed for the random generator

C++ offers several sequences of random numbers. Users can decide which sequence to use and obtain exactly the same results. This permits the replication of results, crucial for any scientific analysis. I can send to a colleague a model and a random sequence and he will observe the same results.

In Lsd users decide which sequence to use in menu **Model/Sim.Setting/Init. Seed**. The name is due to the actual system used to generate random numbers. They are the results of a complicated (but deterministic) mathematical function generating the sequence of number. Given a seed, this function "grows" a sequence appearing as a random.

Try using the same initialization of the model but different seeds. The results will differ.

Testing against randomness

When using random models we have some parts of the model that depends from the model structure, and another part that depends on the randomness.

We may ask if the results we obtain depend on the random part or depend on the structure of the model. For this we need to repeat the same simulation many times and seeing the frequency of a given results.

Lsd offers this possibility by repeating many times a simulation run with identical initialization but different seeds, that is, different random sequences.

Testing against randomness

Let's set our model to have all firms identical but one with a small advantage. We ask whether the advantage is enough to make this firm win more frequently.

- **a**=0.2, **Min**=-0.05 and **Max**=0.05.
- All $\mathbf{Sales}_0$=100.
- All $\mathbf{Quality}_0^i$=10 but the first $\mathbf{Quality}_0^1$=10.5
- Set in menu **Run/Sim.Setting** the values **Num. of Simulations**=100 and **Num. Steps**=500.
- Use menu **Run/Remove Plot Flags** to avoid having 100 Run Time Plot windows.

Running multiple simulations

Run the simulation. Now the system executes automatically the following steps:

1. Set the seed to the **Init. Seed**.
2. Runs the 500 steps of a simulation run with the current seed.
3. Saves the result in a file with extension **res** and the seed value in the name.
4. Reload the configuration.
5. Changes the seed increasing the current seed of 1.
6. Repeat from 2.

Click in the **Log** window on the button **Fast**.

Running multiple simulations

At the end, we have 100 files containing each the history of the simulation with the seed indicated in the name. Moreover, we have a file, extension **tot**, containing the last value of each saved variable from each simulation. The **tot** file is not the history of a simulation, but permits to compare the final results from each simulation.

Select all the **Sales** variables and click on **Statistics**. Nothing happens, but observe the **Log** window. We have the some descriptive statistics computed over the 100 values for each variable at the end of their simulation runs.

The columns `Min` and `Max` show the minimum and maximum value for each variable. Clearly, every variable happened to win some runs. But the first one won more frequently, since its **Average** is much higher.