Revisiting Nelson-Winter models in Lsd

Version: 23 March 2004

Esben Sloth Andersen, esa@business.auc.dk
DRUID/IKE, Aalborg University, Denmark
http://www.business.auc.dk/evolution/esa/

This note suggests revisiting Nelson-Winter models by means of Lsd simulations using the nw2004 simulation program. The note is mainly designed for students who have already an elementary knowledge about evolutionary dynamics as well as about the Laboratory for Simulation Development (Lsd). While the former topic is now covered by a many papers and a few books (including Andersen, 1996), the topic of concrete exploration of existing simulation models and development of new models with the help of Lsd can only be covered by a ìhands-on approachî as described by Valente and Andersen (2002), but Lsd has changed since the paper was written..

1. Windows installation of Lsd and the nw2004 model

Download the newest version of Lsd from http://www.business.auc.dk/lsd/. Decompress it to the C drive so that it is placed as C:\Lsd. The Lsd directory may also be placed elsewhere, provided that you do **not** place Lsd in a directory whose name includes spaces (e.g. ìProgram Filesî). In the Lsd\Manual directory there is a 70 page Lsd manual called Manual.pdf. Print this file.

Then download the nw2004 program from

http://www.business.auc.dk/evolution/schump/jcourse/nw2004.zip

and unzip it into the Work directory as C:\Lsd\Work\nw2004.

Finally, localise the Lsd directory and doubleclick the file run.bat. In the dialogue window click OK, click Work in progress, and click nw2004. Now you are in the Lsd Model Manager and nearly ready to work with the nw2004 program. However, you might have to change the information that Lsd uses to localise itself on Windows machines. Select Model/System compilation options. If you have installed Lsd on the C drive, the third line should be LSDROOT=C:/Lsd. If not, please change the text exactly as given here (e.g. with an ordinary slash and not a backslash) and click OK.

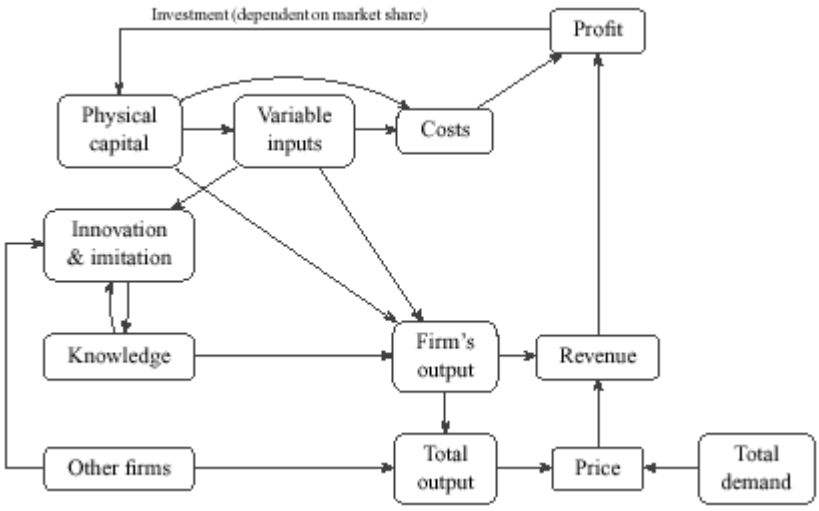
To work with the NW model presupposes some basic knowledge about the Lsd system. However, you may begin an initial exploration by following the present note. Later you will have to study the Lsd manual (accessible from the Lsd windows).

But we should not start exploring the NW model before knowing something about it. Some of the relevant information is given in sections 2 and 3. The hands-on information will continue in section 4.

2. Some background information on NW models

For simplicity, it is often convenient to refer to the Nelson-Winter models in general and to talk about the NW model or simply NW. The basic structure of this model is depicted in figure 1.

Figure 1. The basic Nelson-Winter industry model



The quickest way to describe the NW model of figure 1 is to consider how its dynamics is determined by two basic processes:

1. The selection process emerges from the fact that in the NW model all firms face the same price although they may have different unit costs. These differences are based on different capital productivities that imply that a given stock of capital (and thus a given cost level) can produce different quantities in different firms. Thus the differences lead to different profit rates. We may also say that firms are being selected in the sense of being given different profit rates; supernormal profits can be considered as rewards for high fitness, while subnormal profits are punishments for low fitness. If the bottom-line profits are directly transformed into expansion or contraction and if the firms have different productivities, we have a case of pure replicator dynamics. Here the best performing firm in a deterministic way will ultimately become the sole supplier of output.

2. The innovation-imitation process both produces and controls the variance of the firms' productivities. Since fitness variance is of central importance, the complex NW models show great ingenuity in the specification of the innovation-imitation process. One of them considers this search process as an ad hoc activity that emerges when firms experience unsatisfactory results, but the rest of the NW models have R&D as a permanent activity. These models may, in this respect, be considered to represent the standard NW model. This standard model demonstrates that the introduction of R&D reinforces the long-term tendency toward monopoly. The reason is that although R&D activities show constant returns with respect to change of productivities, there are increasing returns to the application of their results. This means that, compared to a small firm, a large firm will not only have more researchers but also larger returns from each R&D result. Therefore, if firms show no restraint on their expansion, there is a probability of one that one of them will ultimately become the sole supplier.

The process of capacity accumulation is specified as a countervailing force to the tendency to monopoly. In the standard NW model the solution is to introduce a monopolistic restraint on capacity accumulation. In the tradition of Cournot oligopoly analysis the restraint is modelled as the firm's increasing awareness of the overall demand curve as its market share increases. In the NW context a large firm, of course, also have to take into account the potential productivity change of its competitors, but nevertheless the propensity to accumulate will decrease as the market share increases; there will be zero accumulation before monopoly is reached. In other words, the process of concentration stops before monopoly is reached. An alternative strategy to control the tendency toward monopoly is to introduce new firms into the industry. These new firms enter from an exogenous pool of firms.

The standard implementation of NW in Lsd is quite simple. We shall, however, mainly apply a somewhat modified version called nw2004. It differs from the standard by:

- allowing switching off monopolistic restraint of investment,
- including fission of large firms,
- emphasising the statistics for decomposing evolutionary change,

- demonstrating how significant experiments can be designed.

3. Parameters and initial values for nw2004 simulations

To perform a NW simulation, the parameters and state variables have to be given values and special information for the simulation has to be provided. Let us start with the parameters, where I have emphasised the ones that we shall especially consider with boldface. The list is:

```
Statistics-level parameters
   Param:   Stats    - Switches population statistics on/off

Industry-level parameters
   Bank       - External financing rate
   Dem_Coeff  - Coefficient of the demand function
   Dem_elast  - Elasticity of Demand
   Dep_rate   - Depreciation rate of physical capital
   Inn_stddev - Standard deviation of innovative draws
   Regime_imi - Imitative regime of the industry
              0: no imitation
              1: imitation is based on the industry's best productivity
              2: imitation is based on the industry's mean productivity
   Regime_inno - Innovative regime of the industry
              0: no innovation
              1: innovation is based on the industry's mean productivity
              2: innovation is based on the firm's present productivity
   Regime_restraint - Monopolistic behaviour in the industry
              O: no monopolistic behaviour
              1: monopolistic restraint due to mark-up pricing
   Regime_fission - Splitting of large firms
              0: no fission of large firms
              1: fission of large firms
   Fiss       - Determines the probability of fission, since
                fiss * ms is lambda in a Poisson distribution
                that determines whether fission takes place
   Stats      - Parameter determining whether advanced
                population is calculated. If 1, the statistics
                is calculated

Firm-level parameters
   RIM - Invetment rate on imitation
   RIN - Investment rate on innovation
   Inn - Boolean (0 or 1) parameter; 1 if both innovator and
                imitator, 0 if only imitator
   C   - Production cost by unit of capital
   AN  - Productivity of Innovative R&D Investment
   AM  - Productivity of Imitative R&D Investment

Lsd system level: Set Simulation Settings (choose Run Menu/Sim. Settings)
   Number of Simulations = 1
   Initial Seed = 1
   Simulation Steps = 2000
   Insert Debugger at = 0

Lsd system level: Variable Settings (double-click each variable name)
   Debug (unchecked)
   To save (checked)
   Run Time Plot (checked for plot, otherwise unchecked)
   The number of each object type, e.g. 4 or 16 firms.
```

The next thing we have to consider is the initial values of the lagged variables (the state variables). For convenience, I list all the variables and emphasise the variables that need initialisation with boldface.

```
Root-level variables
   T

Statistics-level variables
   InvHerf        The inverse Herfindahl concentration index
   HP_index       The Hymer-Pashigian instability index for market shares
   A_Mean         Mean productivity weighted by capital shares (s)
   Plus a series of variables relation to George Price's equation, see below

Industry-level variables
   Price
   A_average      Average productivity (unweighted)
   A_mean         Mean productivity weighted by market shares (ms)
   Supply
   A_Max          Best-practice technology in the industry
   K_total        The sum of all the firms' capital

Firm-level variables
   A       Productivity
   K       Physical capital
   PROF    Profits per unit of capital
   A_IN    Innovative result
   A_IM    Imitative result
   Inn_mean The mean of the normal distribution that determines the innovative
           result for firms
   Q       Output (lagged for calculating statistics)
   s       Capital share (lagged for calculating statistics)
   ch_ms   Change in market share (for statistics)
   ms      Market share (lagged for calculating statistics)
   I_rate     Investment rate
   MU_target  Target markup, influenced by share if on restraint
   MU_actual  Actual markup
   Krepro     Reproduction coefficient
   Fission    Determines whether a split-up of the firm takes place
```

4. Quick firm-oriented experiments with nw2004

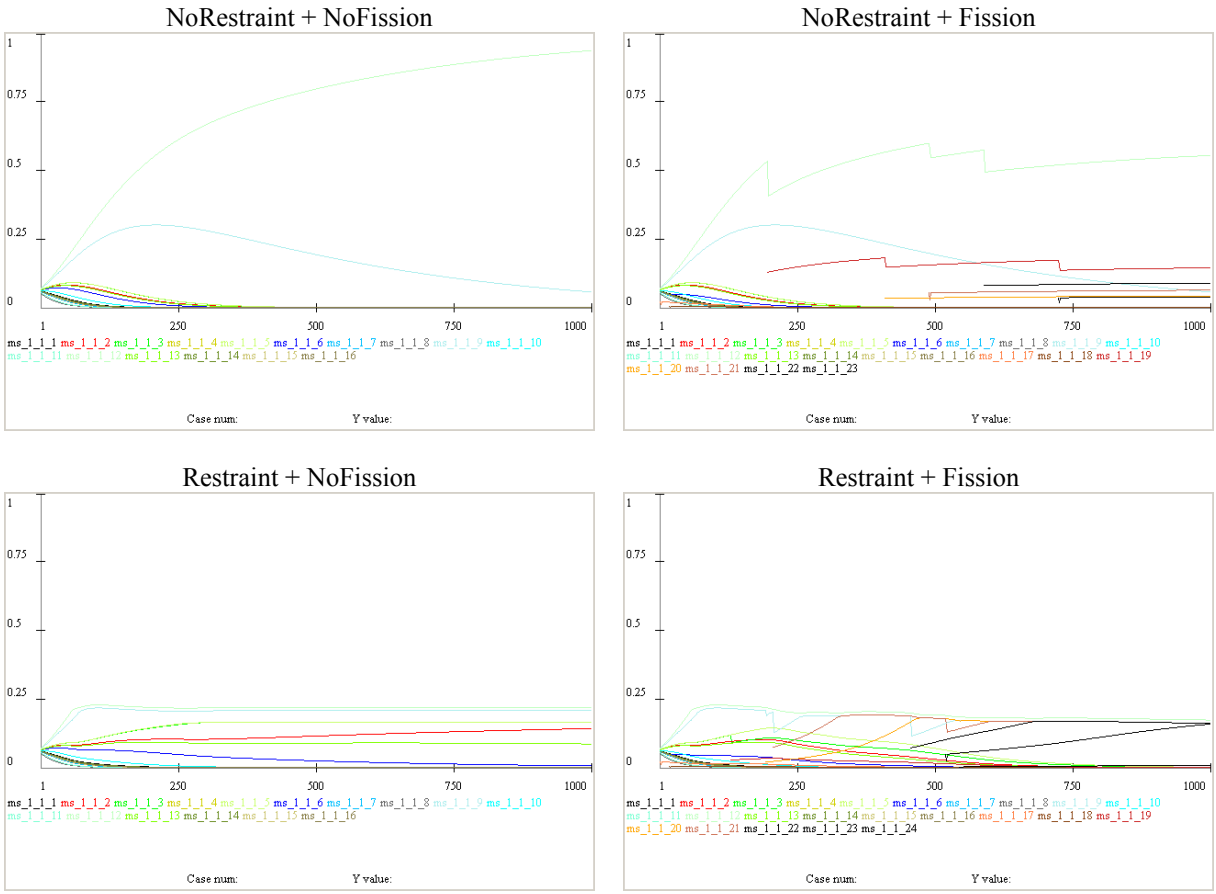
In the nw2004 model directory there is a series of Lsd configuration files that allows quick experiments with the modelís facilities. For some of the configurations, you should go to Data/Analysis Result to get an impression of what is going on. But it is also important that you compare across the different configurations. Here the Run Time Plots with time series for the firmsí capital levels might give impressions of what is going on. But it should be emphasised that it is *only* first impressions. The analysis of simulation results requires serious statistical work at the population level. Let us, however, enjoy the first impressionsÖ

4.1. From pure replicator dynamics to restraint and fission

We start with `nw04_Imi0_Inno0_Res0_Fis0.lsd` that has already set Regime_Inno = 0, Regime_imi = 0, Regime_restraint = 0, Regime_fission = 0. Furthermore, the parameters RIN, RIM and Inn are set to zero for all firms (if not, the system does this automatically). This means that there is no innovation or imitation, no monopolistic behaviour and no splitting of firms. This brings us as close as possible to pure replicator dynamics. The only problem is that firms only shrink due to a 0.03 physical depreciation of capital in each period.

To prepare the configuration files, it is necessary to add variance of the firmís initial productivities. This could be done (but do not do it now!) by going to the firm object, choosing Data/Initial Values, clicking Set All for the productivities (A), writing 0.16 and 0.03 in the first two fields, and then selecting normal distribution. This has already been done in the predefined configuration files. So you just have to run the configuration and think about the results.

Figure 2. Replicator dynamics and variants (Regime_inno = 0; Regime_imi=0)



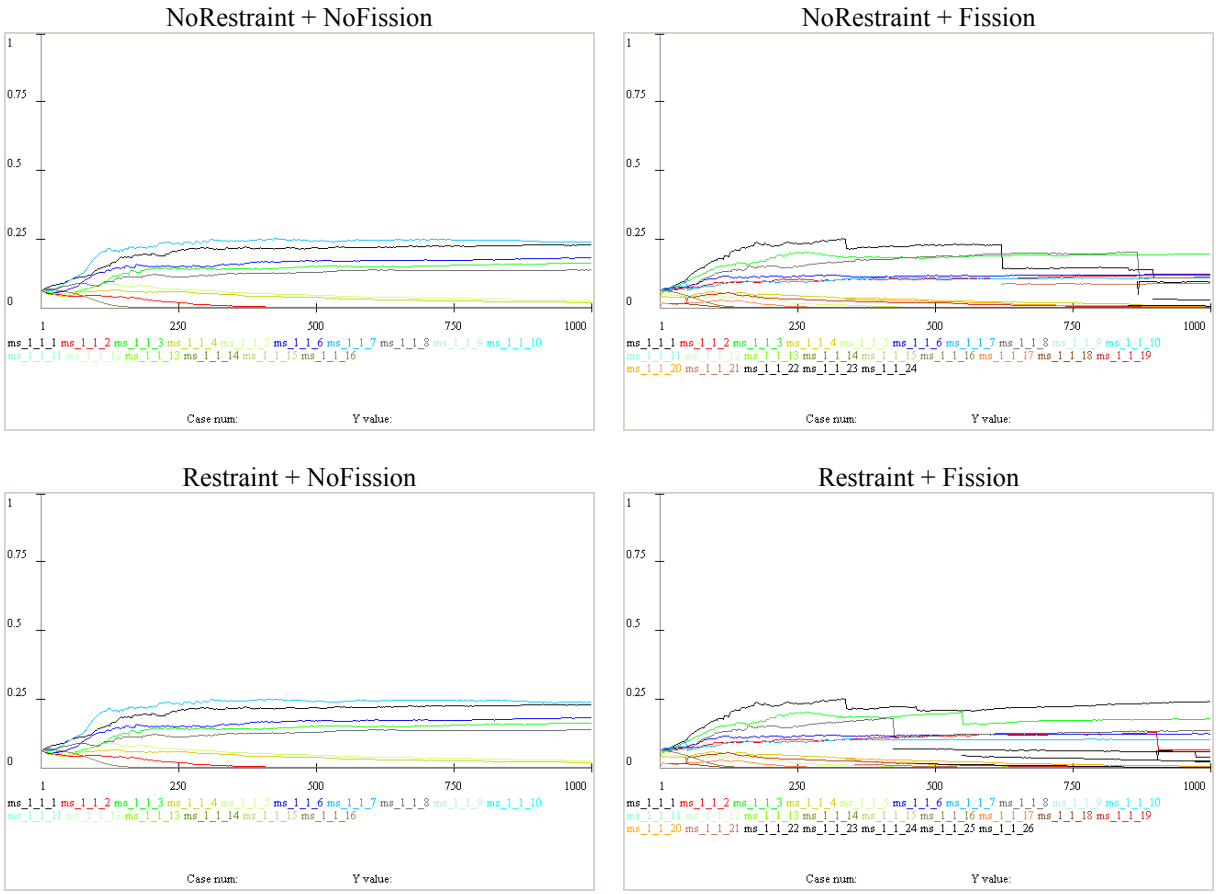
Then try out switching on restraint and/or fission. You can either do it yourself or choose the predefined configuration files `nw04_Imi0_Inno0_Res1_Fis0` and `nw04_Imi0_Inno0_Res0_Fis1` and `nw04_Imi0_Inno0_Res1_Fis1`. You get an impression of what is going on from figure 2 (but your results may not be indentical to those in the figure!).

4.2. Industrial dynamics with a conservative learning from the industryís average

We now turn innovation and imitation on, as it has already been done in the config files. This means that RIN and RIM has small positive values and that Inn is 1 for at least some of the firms. In this section and the next, halt of the firms have both innovation and imitation, while half has only imitation. This is not according to the KISS principle (Keep It Simple, Stupid!), but we shall suggest how to plan a better sequence of experiments later.

In this section, technical change takes place within the condition Regime_Inno = 1, i.e. firms obtain innovative results from a normal distribution whose mean is the weighted mean of the productivities of the industry (see figure 3).

Figure 3. Dynamics with unfavourable innovation (Regime_inno = 1; Regime_imi=0)



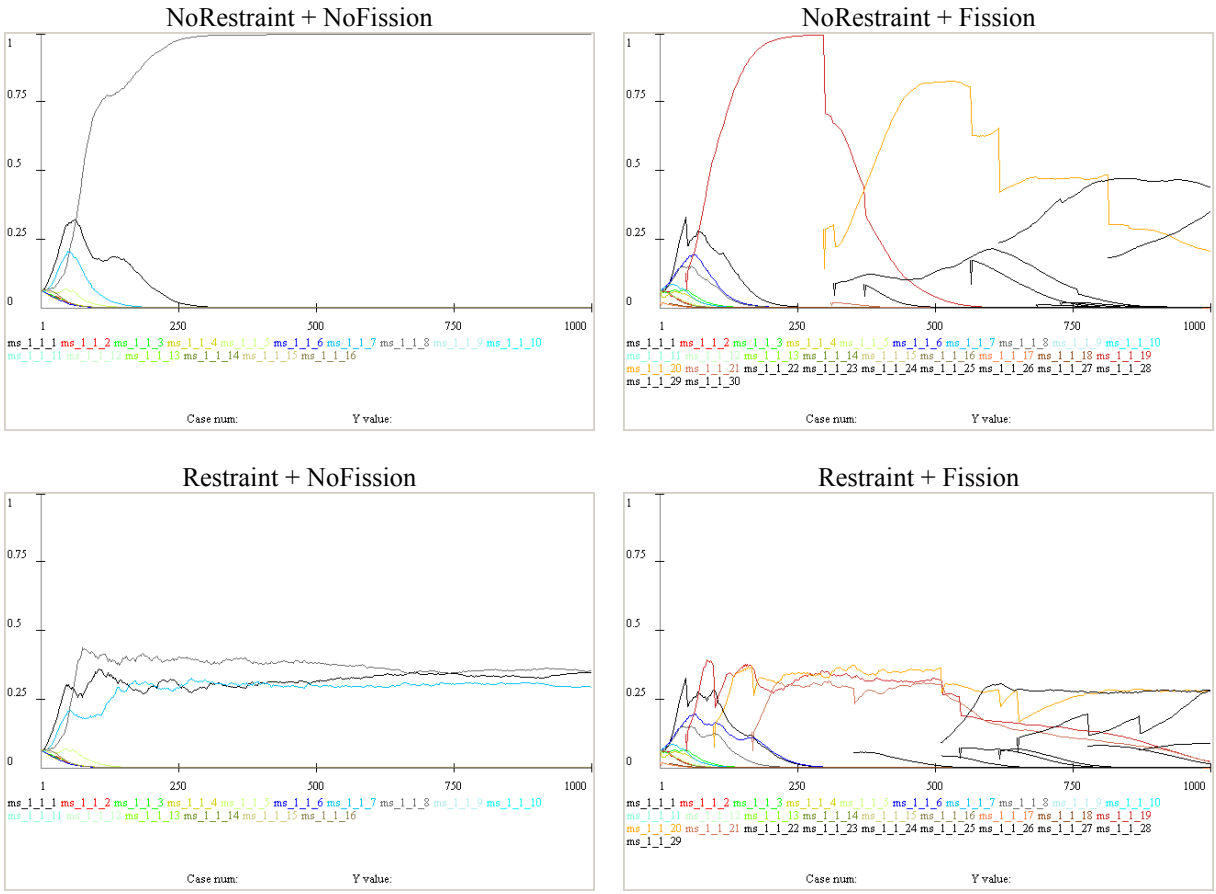
`nw04_Imi0_Inno1_Res0_Fis0` includes this cumulative regime of technical change in which the mean of the normally distributed innovations of the firms is the industry average productivity. This implies that strong firms are seriously hampered in their expansion. To see the pure effect of this we start in this config file with no investment restraint and no fissions. The results demonstrate that monopoly can be avoided. Try out `nw04_Imi0_Inno1_Res1_Fis0` and `nw04_Imi0_Inno1_Res0_Fis1` and `nw04_Imi0_Inno1_Res1_Fis1`, and compare with figure 3.

4.3. Industrial dynamics with firm-based knowledge cumulation

It is now time to turn to a more realistic version of technical change, where the mean of the firmsí innovations are their present productivities. This setting, of course, strengthens the position of dominant firms. We start as usual with a simplified version of the model set-up, `nw04_Imi0_Inno2_Res0_Fis0`, where Regime_Inno = 2. But now we see a dynamics between two firms that are both innovators and imitators, and are able to interact in a long-term coexistence (see figure 4, panel 1).

The coexistence of market leaders becomes broader in the following configurations. In `nw04_Imi0_Inno2_Res1_Fis0` the monopolistic restraint implies that there is room for four more or less coexisting firms. In `nw04_Imi0_Inno2_Res0_Fis1` the fission of large firms means that we see a complicated pattern. Since there is no restraint, each productivity leader quickly expands. But as time goes by, all surviving firms are clones or dominant firm, so the industry tends to have a lower and lower concentration. In `nw04_Imi0_Inno2_Res1_Fis1` we mix restraint and fission. Thereby new firms can enter the team of leading firms.

Figure 4. Dynamics with favourable innovation (Regime_inno = 2; Regime_imi=0)



4.4. Analysing experiments

As soon as you learn to to set the parameters yourself, you can perform a large number of experiments like the ones suggested in sections 4.1-4.3. However, at some point of time you need to move from the study of individual simulations to a characterisation of the general behaviour of the model for particular parameter settings.

The problem is that each simulation may give a different result. To see this, load the configuration `nw04_Imi0_Inno1_Res0_Fis0` into the Lsd Browser, open Run/Sim Settings, and run. Study the Run Time Plot (e.g. the colour of the winning firm. Then choose File/Re-Load, change the Initial Seed to another integer number, and run again. Now the winner is probably another firm. If not, try again with another seed.

What you have just seen is the effect of the Random Number Generator that governs the innovative results. In computer simulation, random numbers are actually pseudo-random numbers. You get a series of numbers that is identical to real random numbers. But they are actually deterministic in the sense that if you give an initial seed to the Random Number Generator, then you get a well-defined sequence. So to get another result, you need to give another seed. This is done automatically if you make the run for several industries. When this is done, you can do a statistical analysis to find the typical behaviour of the model for given parameters. You may, e.g., check whether the model converges to monopoly after a given number of periods (e.g. 5000).

Another way of thinking about the dynamic stability of the long-term outcome is to study the sum of the sum of the absolute values of the changes in market shares (or capital shares). This is called the instability index or the Hymer-Pashigian index. You may perform a statistical analysis of this index for a number of runs.

5. The population-level approach to the dynamics of NW

Evolutionary analysis is primarily population-level analysis, so we need lots of statistics to understand and present our evolutionary results. For nw2004 I have already sketched out some relevant statistics that primarily deal with the issue of decomposing evolutionary change (see papers at Andersenís website). Here I apply Priceís equation to partition total evolutionary change into three different effects:

$$\text{Total evolution} = \begin{matrix}\text{Interfirm} \\ \text{selection effect}\end{matrix} + \begin{matrix}\text{Intrafirm} \\ \text{innovation effect}\end{matrix}$$

On the left-hand side of this equation we have total evolutionary change that is at present simply the change of the mean level of productivity in the industry, where productivity is weighted by the market shares (or the capital shares) of the firms. The total change is partitioned into two effects. Interfirm selection is the covariance between their fitnesses and their productivity levels. Intrafirm innovation is the mean change of trust productivity due to transformation of individual firm behaviour. Thus ëinnovationí is used as a local concept (including innovation in the narrow sense, imitation, and more). When you study it, the partitioning may seem obvious, but it is analytically very helpful. Furthermore, it the simulation model has different levels (e.g. the overall industry, local districts and firms), then the decomposition may be extended:

$$\text{Total evolution} = \begin{matrix}\text{Interregional} \\ \text{selection effect}\end{matrix} + \begin{matrix}\text{Intraregional} \\ \text{selection effect}\end{matrix} + \begin{matrix}\text{Intrafirm} \\ \text{innovation effect}\end{matrix}$$

Since nw2004 is still very simple, we shall stick to the decomposition into two effects. This is really an identity first presented by George Price in the 1970s. Let us present it in a NW-like form:

$$\Delta A\_Mean = \frac{\mathrm{Cov}(Krepro, A\_Mean)}{Krepro\_Mean} + \frac{\mathrm{E}(Krepro \Delta A\_Mean)}{Krepro\_mean}$$

Priceís equation can be used for any model and any real-life data on evolution since it emphasises that the two components into which mean productivity change can be rewritten are produced in a tautological manner. Let us shortly consider the different elements of the equation. There are, however, many fine details that cannot be presented, and even the statistics part of nw2004 is produced in a few hours so that it is not fully correct. (The major problem is that I as a quick hack use market shares instead of capital shares; this will soon be corrected at my website.)

It should be emphasised that the characteristics of population members (like productivities) are defined as weighted means (e.g. $A\_Mean = \sum s_i A_i$). Second, the reproduction coefficient $Krepro_i = K_i' / K_i$ is a complex measure that should be handled with care. Third, the change in the characteristic ΔA_Mean may include complex processes that may be due to both between-firm selection and within-firm selection (e.g. between plants).

To understand the effects of the mechanism of selection we need a lot of statistics about the populationís central variables. Since selection operates on the variance of these characteristics values, we obviously need information on this variance. With respect to, for instance, the groups of the overall population we need to calculate $\mathrm{Var}(A) = \sum s_i (A_i - A\_Mean)^2$, where *z* is the weighted mean of the whole population. But this information is not sufficient to describe the selection mechanism. The main problem is that this mechanism may to different extents exploit the variance present in the population. So the real functioning of the selection mechanism is better expressed by the covariance of the expansion coefficients with the values of the characteristic:

$$\mathrm{Cov}(Kreprod, A) = \sum s_i (Kreprod_i - Kreprod\_Mean)(A_i - A\_Mean).$$

If this covariance is positive, then groups with above-average characteristic value may be said to have been selected. There are two conditions for such a selection. First, there must in the pre selection population exist variance between groups with respect to the value of the characteristic. Second, there must be a positive relation between performance with respect to the characteristic and performance with respect to expansion coefficients. These conditions become obvious by rewriting the formula for covariance. Let is first note that the simple regression coefficient of expansion on characteristic values is $\mathrm{Cov}(Krepro, A) = \beta(Krepro, A)\,\mathrm{Var}(A)$. This expression of the covariance allows us to consider selection as a product of two terms. Here we may interpret the regression coefficient as the intensity of selection and the variance as the material on which selection operates.

If we reduce variance in the model to zero, the covariance term is also zero and no selection effect can be present. Thus we recognise that the second component is a pure innovation effect. This effect it based on changes of the productivities of the individual population members (ΔA_i), but its size is determined by the way in which large productivity changes corresponds to the capacity shares in the final population ($\sum s_i (Krepro_i / Krepro\_Mean)\Delta A_i = \sum s_i' \Delta A_i$).

All these statistical variables are implemented and described in the equation file.

```
Statistics-level variables (for explanations, see the equation code)
   A_Mean
   A_Var
   Krepro_Mean
   Covar_Krepro_A
   Regres_Krepro_A
   A_Mean_Delta
   Innovation_effect
   Selection_effect_est
```

The results of using these statistics will not be included in this note, since the Statistics object of nw2004 is working correctly (see above). However, by plotting the data and drawing the regression line, we obtain an impression of how existing variance with respect to the characteristic is exploited by the selection process. This can be done in Lsdís Data Analysis window by putting the set of *A* variables into the Selected Series first and then the *Krepro* variables. Then check Points, XY-plot and Cross Section. Then clicking Plot and specifying the time step to do the plotting produce a Gnuplot diagram.

6. NW and Lsd/C++ programming

When an Lsd simulation is running, the system will for each time step try to compute the value of each variable in each object. If we have added a new variable in the configuration file (e.g. Bankrupt[i,t] for Firm; not implemented!) and we have not yet written the equation for calculating this variable, then the system stops the computation and gives us an error message. If we, however, have an equation that add a new object of type Firm (in the equation for Fission), the system have no problems. We just have a clone (with inherited productivity and a share of the employment of the mother firm), so the system is already provided with the necessary equations.

The importance of the model structure goes deeper. In old-fashioned programming firms were normally represented by arrays of variables and the programmer had to be careful about the indexes. Since Lsd is extensively using so-called pointers to objects, such considerations can largely be ignored by the Lsd model developer. If a firm uses a parameter like Regime_Inno to calculate whether a R&D result is obtained, the Lsd system first searches for this parameter in the firm itself and then it moves upward in the hierarchy. Since the parameter is not defined in the firm but in the industry, which contains it, Regime_Inno is found. Even if there are different Regime_Inno in different Industries, the Regime_Inno governing the firm is uniquely defined. In some cases this method of finding values is too constraining, so the Lsd system provides a number of alternative search methods. Therefore, it is important to develop the configuration file in parallel with the equation file.

Let us consider a simple NW example: the Lsd code for the calculation of the outcome of an innovation of a firm. This code is written in the Lsdís macro language. Let us consider the following "equation":

```
EQUATION("A_IN")
/*
…
   If the research succeeds, then the productivity assumes the a random
value draw from a normal function, around the average productivity of
the industry in the beginning of the period and with a standard
deviation given by Std_Prod. If the research does not succeed the
returned value is 0.
*/
if (V("Inn") == 0)
  v[10]=0;
if( V("Inn") == 1 && (v[0]=RND) < (v[1]=VL("K",1)*V("RIN")*V("AN")))
      {
      if (V("Regime_Inno") == 1)
        v[10]=norm(VL("A_average",1),V("Std_Prod"));//Industr cumul
   else
        v[10]=norm(VL("A",1),V("Std_Prod")); // Cumulative at firm level
   }
else
   v[10]=0; //no innovation
RESULT(v[10])
```

The code in capital letters (like EQUATION(ì xxxî)) is Lsd macro language, which automatically gets translated into C++ under compilation of the program. But we want to concentrate on how to improve the program. We have already seen that it is strange that industry-oriented knowledge in Regime_Inno=1 is influenced by small and nearly dead firms through a draw in NormalDistribution(A_average, Std_Prod). We want to insert A_mean instead (which is already calculated in the program). So we open the equation file in LMM, find the above code, **write A_mean instead of A_average** (with correct capitalisation and keeping the goose-marks). Then we save the file, call Model/Run Model, and then we obtain an Lsd model Browser with the corrected equation.

But stop a moment! We are now using A_Mean with a lag of one, VL(ì A_Meanî), so we have to check whether this value is defined in the configuration file! Luckily this lagged value is already used for calculating statistics (available in object Statistics), so there is no problem. If this was not the case, we would have to change A_Mean to a lagged variable by clicking on its name, clicking in the top bar of the box that emerges (an Lsd trick), and changing the lag from 0 to 1. Then we would have to call initial values and give it a value (in NW models often 0.16). But we do not have to worry at the moment. Just run the model and it will show substantially different behaviour than previously.

This correction is simple, but not exactly satisfactory. Instead, we would like to keep the old Regime and add a new one. Thus we in all have three regimes, e.g. Regime_Inno = 1 means that A_Average is used, Regime_Inno = 3 means that A_Mean is used, and any other number means that A is used. The corrected code would be something like:

```
EQUATION("A_IN")
…s
if( V("Inn") == 1 && (v[0]=RND) < (v[1]=VL("K",1)*V("RIN")*V("AN")))
{
  if (V("Regime_Inno") == 1)
    v[10]=norm(VL("A_Average",1),V("Std_Prod")); //Primitive industr cumul
  else
    {if (V("Regime_Inno") == 3)
      v[10]=norm(VL("A_Mean",1),V("Std_Prod")); //Primitive industr cumul
     else
       v[10]=norm(VL("A",1),V("Std_Prod")); //Normal industr cumul
    }
}
…
RESULT(v[10])
```

If you make this correction, there is a good chance of making mistakes. In the end you will, however, succeed. But here is something more complicated to consider. The task of the equation for Fission is not only to return a value (true if the firm has undergone a fission in the present period, false otherwise). The equation also has the effect of creating a new firm with the same productivity as the mother firm and with a share of its capital. And most importantly, it creates a new object firm during the run, so that the number of firms will increase. I have to time to give many comments. But here is the code for inspiration:

```
EQUATION("Fission")
/*
Fissions of firms are determined after the new productivity is found. The
probability of a fission depends on the parameter psi times the market share of the
firm (as a Poisson process). If a fission takes place, the capital of the firm is
split according to a uniform distribution, and the smallest is set up as a new firm
with the same productivity as the mother firm.
*/
v[0] = V("K"); //ensures update
v[1] = V("A"); //ensures update
v[2] = V("Fiss");
v[3] = V("ms");
v[4]=RND;
v[5]=0.49*v[4]*v[0];
v[6]=(1 - 0.49*v[4])*v[0];
v[7] = V("Regime_fission");
if (poisson(v[2]*v[3])>0 && v[7]==1)
  {
  cur=p->up;
  cur=cur->add_an_object("Firm",p);
  cur->write("K",v[5],t);
  cur->write("ms",v[3]*0.49*v[4],t);
  p->write("K",v[6],t);
  p->write("ms",v[3]*(1 - 0.49*v[4]),t);
  }
RESULT(v[6])
```

Even the slightest inspection of the code demonstrates that the returned value is not the main effect of Fission. Therefore, mathematicians characterise it as a function with serious "side effects" that complicate the analysis of its effects. These side effects are, of course, the main purpose of the codeó to generate a new firm and partition the capital of the mother firm between itself and the new firm. So let us consider this part of the program:

The first thing to note is that it is a Poisson process that determines whether is fission takes place (like other Poisson processes determine R&D results), so we really have to come to grips with such processes that are well known in discrete event simulation of e.g. service processes where customers come irregularly to a server with a limited capacity. In such cases we often know the average number of arrivals per time unit (lambda) or the average time between arrivals (1/lambda). A Poisson distribution is characterised by such a single parameter. The Poisson function of C++ uses the random number generator to produce a stochastic series of arrivals. To handle the R&D process, we the Poisson process has been interpreted in the following simple way: We assume that for a given firm the average number of fissions per time period depends on the market share of the firm times an industry-wide parameter (lambda = Fiss*ms). There are, of course, many other possibilitiesó and really we should rather model *employment* than capital to give a realistic picture.

Given that fission is about to take place, the problem is how to implement the possible split-up of the firm. This, of course, depends on our theory of that matter, and this is largely related to future developments of NW. At present we just suppose that any new firm emerging from a split or spin-off of a mother firm will contain the smallest part of the firm. Therefore, we call an Lsd version of the standard random number generator. This RND function gives a random number that is uniformly distributed between 0 and 1.

The next step is to create a new firm within the economy that contains the mother firm. To perform this operation we need some knowledge about the NW model structure, the use of address variables (pointers) in C++, and the special facilities for address handling by the Lsd system. The ëparentí Object of a Variable is denoted by p->. To get the address of the Industry that contains this parent

object (a Firm), we use p->up. This address is stored as a temporary pointer variable, cur. In the next line the Lsd system is requested to create a new Firm based in the Industry pointed to by cur. The Lsd function add_an_object("Firm",p) has as its second parameter the parent (or mother) Firm defined by the p address. This means that the function add_an_object makes a clone of the mother Firm (including values of parameters and lagged variables). We reset the address contained in cur to the address of this new Firm.

There is, however, one task left: to reset the capital of the mother Firm and the new Firm. In both cases we ask the Lsd system to (over)write the values of capital at time t (a global variable that contains the period under computation). The K variable of the new Firm is addressed by cur-> while the mother/parent Firm is addressed by p->. After the completion of this update the returned value is set to true (1).

The "equation" for Fission is the most complicated parts of the Lsd implementation of the NW model. It demonstrates that is can be quite time consuming to study the details of a simulation program. Luckily, there are simpler ways of understanding the logic and behaviour of the NW model, so the discussion of Fission is largely for would-be Lsd programmers.

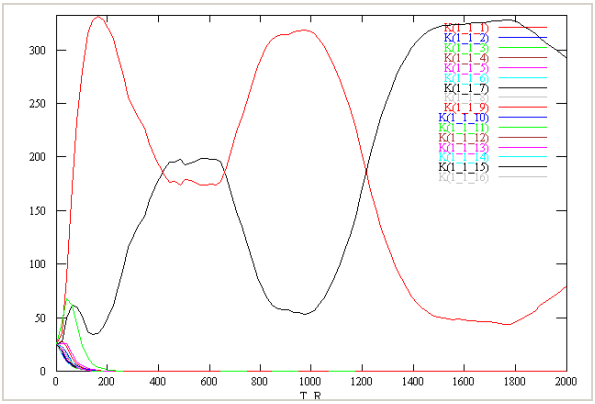
7. How to produce documents with Lsd results

If you revise or develop a model for a report, this model should be made available as an appendix. To make the documentation is easy. When you in the Lsd model Browser have loaded a configuration file, then do two things. First, chose Model/Generate Automatic Descriptions. Then information on parameters and variables are gathered (but be aware not to overwrite your own comments to parameters!). Then choose Model/Create Report. Then an extensive report is made in HTML format, ready for printing or web publishing. Please inspect such reports to understand their values.

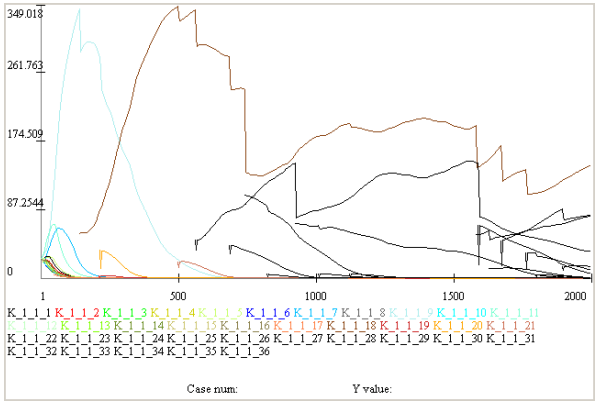
But you also need to analyse the outcome of your simulations. Here advanced statistical tools are necessary. But different plots are also useful. The present note has used plots to depict the outcome of simulation experiments. Each figure has used two types of plots. Screenshots of time series produced by Lsd itself (right panel of figure 7), and plots produced by the integrated and advanced programme Gnuplot (left panel.). By the way, there is a reason why a do not show any Gnuplot images of runs with fissions. Gnuplot has a problem with missing values, and I have had no time for correcting the underlying data files!

Figure 7. Different plotting methods for LNW data

Screenshot of a preliminary plot
created by Gnuplot in Lsdís data analysis window



Screenshot of an ordinary Lsd plot
created in Lsdís data analysis window



Simple Lsd plots are produced in the following way: After each simulation the variables are selected and plotted in the Data Analysis window. It might be relevant to specify the endpoints of the y-axis to obtain nicer values than through Lsd's auto-settings. Then the plot window should be collapsed (not closed!) and PostScript selected. After selecting the name of the plot and clicking OK, a plot in portrait form (with width 70 mm) is used for small (and nearly unreadable) output and then the plot is saved. After that the PostScript plot may be opened in GSview (freely available, e.g. from http://www.cs.wisc.edu/~ghost/)ó with option EpsClip. Finally the picture is copied and inserted in a

Word document through PasteSpecial (device independent, not float over text). The file uses colours and is better for on-screen viewing than for black printing.

Gnuplot output (nearly all the graphs) is better for journal-quality outputó but somewhat more difficult to produce (see the Lsd Manual under Manual/mdatares.html#Gnuplot and XY Plotting). To obtain these plots the following procedure may be used: In the Data Analysis window the first variable to be put in "Series Selected" is the help variable T (i.e. time; see in the equation file how it records the system time). Then comes the rest of the series, i.e. all the A-series of productivities. Then "XY-plot" is checked and "Plot" is chosen. This means that Lsd calls the program gnuplot to produce a (sometimes ugly) graph. If the graph is good enough and you are in a hurry, use a program to take a screenshot (like in this note).

If you want better results, you have to learn about Gnuplot (it is not easy!). Close the graph (but not Data Analysis!). Then locate within the model directory a new subdirectory called plotxy_1. In this directory there are two relevant files: data.pg and gnuplot.gp. Copy both to a new model subdirectory (e.g. called GnuPlots) and rename the files if you want to add more files later (e.g. to data1.gp and gnuplot1.gp). Now you can edit the file gnuplot1.gs (e.g. in the editor of LMM). The version used for the first LNW data set is:

```
set yrange [0.000000:1.000000]
set data style lines
set nokey
set nolabel
plot 'data1.gp' using 1:2 t "A(1_1_1)", 'data1.gp' using 1:3 t \
"A(1_1_2)", 'data1.gp' using 1:4 t "A(1_1_3)", 'data1.gp' using 1:5 t \
"A(1_1_4)", 'data1.gp' using 1:6 t "A(1_1_5)"
pause -1 "Click to close"
```

This text is produced by minor modifications of the file gnuplot1.gs ("\" indicates that a single line is used). To produce the actual output, you may start c:/lsd4.0/gnu/bin/wgnuplot.exe, change directory to the location of the file and write " load 'gnuplot1.gp' ". You see the plot in a separate window, and from there you can copy the contents and paste it into a word document. By pasting it into a cell of a word table you should be able to obtain an automatic rescaling of the plot (perhaps after some experimentation). It is also possible to save the plot as a PostScript file.

References

Andersen, E. S. (1996), *Evolutionary Economics: Post-Schumpeterian Contributions*, paperback edn, Pinter, London.

Andersen, E. S. (2003), Population thinking and evolutionary economic analysis, Postscript to E. S. Andersen, *Evolutionary Economics: Post-Schumpeterian Contributions* [in Japanese], Tokyo: Springer. Forthcoming as a paper in *Evolutionary and Institutional Economics Review*, **1** (2004).

Laboratory for Simulation Development (Lsd). Download from http://www.business.auc.dk/lsd/ and print the Lsd Manual from Lsd/Manual/Manual.pdf

Nelson, R. R. and Winter, S. G. (1982), *An Evolutionary Theory of Economic Change*, Cambridge, Mass. and London: Harvard University Press.

Valente, M. and Andersen, E. S. (2002), A hands-on approach to evolutionary simulation: Nelson and Winter models in the Laboratory for Simulation Development, *Electronic Journal of Evolutionary Modeling and Economic Dynamics* **1**. http://www.e-jemed.org/1003/

Winter, S. G. (1984), Schumpeterian competition in alternative technological regimes, *Journal of Economic Behavior and Organization* **5**, 287ñ320.