University of Leicester

Research Memorandum
ACE Project

No 99/1

Stochastic Optimization in Econometric Models – A Comparison of GA, SA and RSG

By

Adriana Agapie

March 1999

MEET IV

Adriana Agapie

Stochastic Optimization in Econometric Models – A Comparison of GA, SA and RSG

Institute for Economic Forecasting, Romanian Academy
Academy of Economic Studies, Bucharest, Romania
E-mail: agapie@clicknet.ro

September 1998

Adriana Agapie

Stochastic Optimization in Econometric Models – A Comparison of GA, SA and RSG

Abstract This paper shows that, in case of an econometric model with a high sensitivity to data, using stochastic optimization algorithms is better than using classical gradient techniques.

In addition, we showed that the Repetitive Stochastic Guesstimation (RSG) algorithm –invented by Charemza-is closer to Simulated Annealing (SA) than to Genetic Algorithms (GAs), so we produced hybrids between RSG and SA to study their joint behavior.

The evaluation of all algorithms involved was performed on a short form of the Romanian macro model, derived from Dobrescu (1996). The subject of optimization was the model's solution, as function of the initial values (in the first stage) and of the objective functions (in the second stage). We proved that *a priori* information help "elitist " algorithms (like RSG and SA) to obtain best results; on the other hand, when one has equal believe concerning the choice among different objective functions, GA gives a straight answer.

Analyzing the average related bias of the model's solution proved the efficiency of the stochastic optimization methods presented.

Adriana Agapie[1]

Stochastic Optimization in Econometric Models – A Comparison of GA, SA and RSG

Introduction

This paper is a study on the initial values' influence on finding the optimal solution for the Romanian macromodel. Empirical trials showed the high sensitivity of the model's output to small variations of the input values required to run the model. A second goal of our approach is rising some questions on the adequacy of different objective functions for the same economic model.

We will present a comparison between gradient techniques and stochastic search techniques, namely: Genetic Algorithms (GA), Simulated Annealing (SA) and Repetitive Stochastic Guesstimation (RSG). The analysis of the suitable optimisation techniques is depicted in *Section 1*. Besides the gradient-based methods (with their well-known advantage of good local search precision), we also gave credit to modern, probabilistic optimization techniques. The evolutionary algorithms are devoted to multi-modale optimization (GA – population based and SA – individual based) showed good performance on global search and search space exploration). RSG – individual based, was also devoted to multi-modale problems, yet paying more attention to the expert choice of the initial values.

The numerical results of applying the above techniques for finding the global extremum of the Romanian macromodel, including a comparative discussion on the several objective functions that one can associate to the macromodel, are presented in *Section 2*. A criticism of the whole analysis completes the paper. The short form of the Romanian macromodel, together with the mathematical considerations involved is presented in the *Appendix*.

[1] The research was undertaken with support from the European Union's Phare ACE Programme 1996. A special debt is owed to Woijciech Charemza for providing a consistent stream of encouragemnets and suggestions.

1 Stochastic Techniques for Function Optimisation

This section introduces the optimisation techniques used in the Romanian macromodel analysis, while the experimental results of applying these tools are presented in *Section 2*.

When initiating a numerical analysis of some economic model, one has to face a difficult problem: What optimisation algorithm should he/she use? Let us take a brief moment to explain how this question could be answered.

Usually, this problem is solved by a reduction mechanism: classical algorithms from Operation Research (SIMPLEX, e.g.) are excluded, as they are limited to convex regular functions. Gradient techniques (Gauss-Seidel, e.g.) are the first candidates, as their speed and precision on problems with a single local optima is highly appreciated. Probabilistic search algorithms (like Genetic Algorithms or Simulated Annealing) come next: they are easy to implement and they do not require supplementary information about the objective function; besides, they showed good performance on multi-modale problems, where gradient techniques usually fail. Another potential candidate should be the recent Repetitive Stochastic Guesstimation, which mimics the way a real economist solves a parameter approximation problem.

How should an econometrician choose among these different methods (and we have mentioned just a few of them)? Are there some mathematical guidelines supporting this decision? Generally speaking, the answer to this question is negative. As Wolpert and Macready have recently proved, there is *no free lunch theorem for optimization* Wolpert *et al*. (1997). That is, for each optimisation algorithm, the number of function classes on which it performs well is equal to the number of function classes on which it performs badly. In light of these results, the mathematician's advice would be the following: '*Concern about the particular function you want to optimise'*. Summing up, one has to take into account the existing experimental evidence (or convergence results, when available) for the particular class of problems he/she tackles. Any algorithm is welcomed, and even the hybrid techniques deserve careful attention.

This is the way we approached the problem of macromodel extremum analysis. *Section 1.1* concerns Genetic Algorithms, *Section 1.2* introduces the Simulated Annealing method,

while *Section 1.3* refers to Repetitive Stochastic Guesstimation (including its hybridisation with Simulated Annealing).

1.1. Genetic Algorithms

Genetic Algorithms (GA) – as presented by Goldberg (1989) or Michalewicz (1994) - are probabilistic search algorithms, which start with an initial population of likely problem solutions and then evolve towards better solutions. They are based on the mechanics of natural genetics and natural selection.

GA combines survival of the fittest among string structures (in most cases) with a structured yet randomised information exchanges to form a robust (meaning efficient and efficacious) search algorithm.

A simple GA requires the definition of five components: a genetic representation of potential problem solutions, a method for creating an initial population of solutions, a function verifying the fitness of the solution (called "objective function", or "fitness function"), genetic operators and some constant values for parameters used by the algorithm (such as population size, probability of applying an operator etc.).

The natural parameters of the optimisation problem, which represent a potential solution, have to be coded as a finite-length string over some finite alphabet. This string is called chromosome and its components are called genes. A population consists in a set of chromosomes.

The mechanics of a GA are surprisingly simple, involving only copying strings and swapping partial strings. Simplicity of operation and power of effect are two of the main attractions of the GA approach. A simple GA that yields well results in many practical problems is composed of three operators: reproduction (selection), crossover and mutation.

Reproduction is a process in which individual strings are copied according to their objective function values. This fact means that strings with a higher value have a higher probability of contributing one or more offspring in the next generation. This operator is an artificial version of natural selection.

The simple crossover is a two-step operator. First, members of the newly reproduced strings are mated at random and second, each pair of strings undergoes crossing over by

swapping some segments of genes with same size and position.

The last operator, mutation, is performed on a bit by bit basis, by complementing (in the binary logic sense) a single bit position of a string

Some of the parameters of the GA are the population size (mostly fixed from a generation to another), the probabilities of applying each operator which can vary during the algorithm, the STOP conditions which can be determine by some time requirements or, when time is not critical, by precision and efficiency needs.

The GA can be simply represented in the following form:

Simple Genetic Algorithm

1. Set the iteration index to zero: j=0

2. Generate the initial population P(0)

3. Evaluate the chromosomes from P(0)

4. Sort P(0) by the objective function values

5. j=j+1

6. Select the chromosomes for the next iteration

7. Apply genetic operators: crossover, mutation

8. Evaluate the offspring

9. Generate a new population P(j)

10. Repeat 5-9, until some STOP conditions

Let us make a brief discussion on the theoretical analysis of GA: extrapolating the simulated annealing theory onto GAs, Davis provided a complete formalisation of the genetic operators, including both the homogeneous and inhomogeneous case – Davis (1991). The results made use of the Perron-Frobenius and ergodicity theorems associated to non-negative matrices and finite Markov chains, but they could not avoid the primitive form for the transition matrix, yielding non-convergence results only. Fogel independently drew this conclusion by proving the absorption of the canonical algorithm into the set of uniform populations – Fogel (1995). However, the matrix analysis of GAs came to a head when

Rudolph pointed out clearly that a canonical GA does not converge, but its *elitist*[*] variant does -Rudolph (1994).

Despite their indubitable correctness, we can not omit two major weaknesses of the convergence results proved up to this moment. First, they are too general: practically, these theorems make no difference between *elitist GA* and *elitist random walk*, for example. Both are convergent under the circumstances; but this does not correspond to the real case, where GA performs better (at least on some problem classes, in the light of the recent *no free lunch theorems for optimization* – Wolpert *et al*. (1997)). Second, the Markovian models designed up to this moment were not handling the *premature convergence*[**] of the algorithm, which still remains unsolved for common GA applications. Actually, this is the main problem in practice: at some moment in the GA evolution (depending on specific factors as problem difficulty, population size, mutation probability etc.) the current population is filled with instances of the same chromosome. Only accidentally improvements should be expected from that point further.

Finally, we must mention that GAs are extensively used in all areas of applications, including the economic analysis and forecasting – see e.g. Goonatilake *et al.* (1995), Przechlewski *et al.* (1996) or Agapie *et. al*. (1997), to mention only a few papers.

1.2. Simulated Annealing

Simulated Annealing (SA) is a stochastic relaxation technique introduced in 1983 by Kirkpatrick *et al.* (1983) and Laarhoven *et al.* (1987) for solving nonconvex optimisation problems. Its development was encouraged by good results on a large application area and also by a complete convergence theory (which is not the case in Genetic Algorithms, e.g.. We will briefly present in the following the SA algorithm, together with its theoretical properties.

SA is so named by analogy to the annealing of solids, in which a crystalline solid is heated to its melting point and then allowed to cool gradually until it is again in the solid phase at some nominal temperature. At the absolute zero final temperature, the resulting

[*] The algorithm maintaining the best solution from a generation to another.
[**] The GA stagnation in local optima, with all the chromosomes of the current population instances of a single individual.

solid achieves its most regular crystal configuration - corresponding to a (global) minimal value of the system's energy. Based on this natural methapor, SA provides a stochastic search algorithm, by identifying:

- the *energy function* to the *objective function*;
- the *temperature* to a non-negative *control parameter*, tending to zero as the iteration number increases;
- the *minimisation of the energy (objective) function* to the *optimisation task*.

Following Davis (1991), we introduce the SA algorithm in the following manner: Let a random variable be the system's (thermal) energy; at thermal equilibrium the probability distribution is completely determined by the system temperature. This distribution is called the Boltzman distribution (or the Gibbs distribution), and is given by:

$$Pr\{E = E(i)\} = \frac{exp\left\{-\frac{E(i)}{kT}\right\}}{Z(T)} \qquad (1)$$

where: **E** = the system energy (a random variable);

E(i) = the energy corresponding to state i (for readability, we consider only a finite state space - denoted S);

k = Boltzman's constant;

T = the system temperature;

Z(T) = the partition function, that is a normalisation function of the form:

$$Z(T) = \sum_h exp\left\{-\frac{E(h)}{kT}\right\}$$

The main idea of the SA usage relies on the interpretation of formula (1), that is: "*At elevated temperatures, the system occupies all states in its state space with nearly uniform probability, while at low temperatures, states having low energy are favoured. When the temperature approaches absolute zero, only states corresponding to the minimum value of energy have non-zero probability. Thus, the system's energy function can be effectively searched for its minimum value by starting the system at an elevated temperature and allowing it to cool gradually to absolute zero, at which point one of its minimum energy states is occupied with probability one*" – Davis (1991). We must notice that the success of

the SA method is constrained by the requirement that the system achieves equilibrium at each temperature; just in this case the system energy distribution follows the Boltzman form, (1). This is the fundamental limitation of the method: if this requirement is not satisfied, one can notice the algorithm's stagnation in local optimal points, instead of converging to the global optima.

Two more components must be defined for a complete picture of SA, namely:

- the (stochastic) next state generation (i.e., the transition rule from a solution to another - the analogous of the "guess" from *Repetitive Stochastic Guestimation (RSG)*, or of the "crossover + mutation" operation in *Genetic Algorithms(GA)*);
- the (stochastic) acceptance mechanism (deciding whether a new candidate solution is maintained, or not - this decision is deterministic in RSG, by the requirement of improvement in both objective functions, while in GA it corresponds to the "selection operator", which may be elitist, roulette, disruptive etc.).

The *acceptance* and *generation* operators are defined by matrices $A=(A_{ih})_{ih}$, res. $B=(B_{ih})_{ih}$,, usually given by the following transition probabilities from state i to state h:

$$A_{ih}(T) = min\left\{1, exp\left[\frac{E(i)-E(h)}{T}\right]\right\} \qquad - \quad \textit{the Metropolis criterion}$$

$$G_{ih}(T) = G_{ih} = G_{hi} = \begin{cases} 1/\ N_i & h \in S_i \\ 0 & otherwise \end{cases} \qquad (2)$$

where $S_i \subset S$ is the set of states accessible from state i in one transition, and N_i = card (S_i). Note that matrix G defined above is symmetric and independent of T.

Summing up, the SA algorithm can be described by the following schema:

Procedure Simulated Annealing

1. *Choose some initial solution*
2. *Choose some initial value for the control parameter $T=T_0$ (corresponding to n=0)*
    3. *j = 0*
    4. *Generate (at random) a new candidate solution*
    5. *Examine the candidate solution and decide: accept or reject*

6. *If accepted, it becomes the current solution; otherwise, keep the old one; j = j+1*
7. *IF an approximate equilibrium is achieved, THEN put* $T=T_{n+1}$ *and go to 3*

8. *Until some stopping criterion applies.*

Note 1. $\{T_0, T_1, T_2, ...\}$ must be some pre-defined, decreasing string of positive real numbers.
Note 2. The stopping criterion is actually two-folded, including a maximal number of iterations, j, and also a maximal number of control parameters T_n.
Note 3. The SA procedure resembles with the RSG procedure (see *Section 1.3*), by identifying the control parameter T with the "learning rate λ", the generation operator with a (uniform) random draw in some neighbourhood of the current solution, and the acceptance policy with an improvement achievement in the objective function(s). The goal is minimisation of some objective function (energy/error) in both algorithms. The only step from RSG apparently missing from the SA procedure is the modification of the searching intervals while passing from iteration to another (res., from a control parameter to another). Nevertheless, some kind of searching area restriction from iteration to another could be assumed in SA as well, by decreasing the number of accessible states, involved in (2).

It is worth noting that a mathematically complete convergence theory (i.e., sufficient conditions for asymptotic convergence to global optima) exists for SA - see e.g., Aarts *et al.* (1989). The evolution of the search sequence of a SA algorithm as outlined, in which each succeeding solution in the sequence is stochastically determined from the current solution, suggests the Markov chain modelling of the SA. One may distinguish two different approaches: the first one involving only homogeneous Markov chains, while the second one involves an inhomogeneous chain. In the first modelling, our SA appears as a sequence of distinct Markov chains, where each chain corresponds to a fixed control parameter (and hence is homogeneous) and each succeeding Markov chain corresponds to a lower parameter value. Note that each Markov chain in the sequence must achieve its stationary distribution, in order to achieve the global convergence. In the second case, the SA appears as a sequence of solutions evolving as a single inhomogeneous Markov chain (as the transition matrix is dependent of the control parameter, T).

We summarise the existing SA convergence results by pointing out:

- the existence of a unique stationary distribution for the Markov chain corresponding to each strictly positive value of the control parameter;
- the existence of a stationary distribution limit as the control parameter approaches zero;
- the desired behaviour of the stationary distribution limit (i.e., optimal solution with probability one);
- sufficient conditions on the algorithm control parameter (i.e., a control parameter sequence bound) to ensure that the nonstationary algorithm achieves the limiting distribution.

1.3. Repetitive Stochastic Guesstimation

A stochastic sampling method somehow similar to Simulated Annealing has been independently proposed by Charemza (1996), inspired from (and originally devoted to) the usual *guessing* of the parameters involved in a complex, generally large, empirically oriented macroeconomic model. The method was called Repetitive Stochastic Guesstimation (RSG) and it showed good results on non-linear optimisation tasks, even competing with 'older' stochastic relatives, like Genetic Algorithms (GA) - Przechlewski *et al.* (1996).

Actually, there are three points where RSG takes advantage on other stochastic algorithms:

1. At the initial stage, by making use of the prior beliefs concerning the parameters to be guessed - according to the economist's expertise and intuition.
2. By successively restricting the search space from iteration to another, providing an asymptotic convergence of the algorithm in some extreme point.
3. By using two objective functions, instead of one.

Let us briefly discuss these main features of RSG. The first seems to be restrictive for general optimisation purposes, by limiting the application area of the method to a relatively well-known problem (that is, assuming that the user - guesstimator - makes mistakes, but not exceeding 40% of the true values of the parameters, e.g. Przechlewski *et al.* (1996), p.23). This assumption makes the comparison against evolutionary algorithms (like GA, Evolution Strategies or SA) somehow improper, as the last ones are commonly used in the

so-called Black Box Optimisation problems - where no information on the objective function is supporting the search. However, a basis for the comparison exists: many authors in the evolutionary computation field recommend the insertion of additional information into the initial population (of a GA, e.g.), whenever this information is available. As for the individual-based algorithms (SA, e.g.) the choice of initial values according to the user's expertise or intuition is welcomed and easy to implement.

The second feature makes the connection to the heuristics of "learning algorithms", namely by retrieving the common sense expectation of '*increasing the guesstimator's confidence by narrowing the interval from which the parameters are to be guessed, as time goes on'*, Charemza (1996). This is definitely an important difference against GA, with its immutable searching space all over the algorithm. Theoretically speaking, the possibility of limiting the searching area from an iteration to another is specific to individual-based algorithms only (notice that RSG enters this category), thus not to population-based methods (like GA).

Regarding the two criterions, namely the *unweighted* and *weighted* objective functions, the (penalty) weights in the last one are normally distributed according to the difference between the currently guessed and the previous best guess, Przechlewski *et al.* (1996). This makes RSG a dynamical optimisation method, by making the objective function time dependent.

In short, the RSG can be presented as follows:

Procedure Repetitive Stochastic Guesstimation

1. *Set the iteration index to zero: j=0*
2. *Choose some initial values and intervals for the parameters to be optimised*
3. *Choose/compute the initial value for the learning rate* λ_0
4. *Randomly generate (guess) a new candidate solution, inside the current intervals*
5. *Compare the candidate solution vs. the current one - w.r.t. both criterions - and decide: accept or reject*
6. *If accepted, it becomes the current solution; otherwise, keep the old one*
7. *Repeat 4-6, several times, until a better solution is obtained*

8. *j = j+1, decrease the learning rate, decrease the intervals' lengths and go to 4*

9. *Repeat 8, until some STOP conditions.*

One must notice some common points between the procedure above and the *Procedure_Simulated_Annealing*, presented in the previous section. Namely:

- RSG, as well as SA, makes random extractions (by uniform probability distribution, commonly) in order to achieve new candidate solutions.
- they both depend on a strictly decreasing parameter (called temperature - in SA, and learning rate - in RSG).

This resemblance inspired us in defining a hybrid algorithm, using in combination the *interval decreasing technique* (borrowed from RSG) and the *Metropolis acceptance criterion*, from SA - see *Section 2.1* for details. The new algorithm (referred in the remainder of the paper as RSG-SA) proved encouraging results on our macroeconomic modelling problem, outrunning both his parents - as one will see in *Section 2*.

It is also worth noting the resemblance between RSG and some specific types of GA. A confirmation of the existing relationship between RSG and GA came from Przechlewski *et al.* (1996), by introducing a hybrid algorithm, called RSG-GA. They concluded the experimental comparison by stating: *"There is no big difference in the efficiency of both the compared methods (i.e. RSG and RSG-GA). The RSG-GA is better with respect to improvement over initial guesses… but is inferior to RSG when looking at the parameter accuracy measured by RSME."*

From a theoretical point of view, RSG can be compared to the *Contractive Mapping GA*, as introduced in Michalewicz (1994). Both algorithms share the same 'elitist' acceptance criterion (different from the acceptance criterion presented for SA, e.g.); namely, the *Contractive Mapping GA* passes from a iteration to another *only if* an improvement in the objective function was achieved (see Step 7 from the RSG procedure above, for comparison). But this improvement may not occur for a long, long time. What should the practitioner do in that case? According to Charemza, *"if there is no improvement in both of the objective functions it is assumed that some kind of optimum is reached. Sometimes, if we have doubts whether this is a global or local optimum we repeat the process with different*

starting values and see, whether we have a similar solution." This certainly works for most problems, but still remains a theoretical bottleneck in the analysis of GA (and in RSG as well). Some steps in surpassing this lack have been already made in literature, by improving the exploration capabilities of the algorithm using additional statistical information.

Second, one must notice about the RSG paradigm that the expert's "guess" is applied (using the prior economic information) at the beginning of the algorithm only: by the initial setting of the parameters' intervals (mean + length) and by defining the objective and learning functions. Afterwards, the "guess" is actually a random (uniform) draw inside the current interval. In this light, the RSG would resemble to an adaptive GA (that is, a GA with directed mutation) which, in addition, incorporates some information about the parameters in his initial stage.

Concluding, the comparison among the algorithms above may be summed up in the following table:

Table 1. Comparison of stochastic optimization methods

	Type of algorithm	**Initial point(s)**	**Operators**	**Acceptance of a new solution**	**Avoid stagnation**
GA	population -based	random	crossover, mutation	roulette/elitist/ disruptiv	yes/ no
SA	individual -based	random/ fixed	mutation	Metropolis criterion	no
RSG	individual -based	random/ fixed	mutation, interval decreasing	elitist	no
RSG-SA	individual -based	random/ fixed	mutation, interval decreasing	Metropolis criterion	no

2. Numerical Results

This chapter is divided into two sections. The first one focus on finding the global optimum of the Romanian macromodel using four different optimization techniques, namely:

- Gradient technique for non-linear systems (as included in EXCEL 7.0 - Solver, referred as EXCEL);
- Genetic Algorithm with real-valued chromosomes (GA, implemented in Visual Basic);
- Simulated Annealing (SA, two different programms, implemented in C++ and T Pascal);
- Repetitive Stochastic Guesstimation - Simulated Annealing (RSG-SA, a hybrid technique, implemented in C++ and T Pascal).

Section 2.2 presents an analysis of the adequacy of the three different objective functions to the considered system. One will see how the second objective function is preferred, both by theoretical and empirical reasons.

2.1. Finding the Global Optimum of the Romanian Macromodel

We start this section mentioning that all the calculus below is restricted to the 1995-1996 anual data of the economic indicators, as they appear in Dobrescu (1996).

We considered some fixed intervals for each initial value of the endogeneus variable (required at start, for each solving procedure). These intervals are large enough for enssuring a good exploration of potential solutions, yet appropriate to the specific economic data in study. As mentioned in *Appendix*, we used 30 endogeneus variables, but for readability reasons we selected the following, most representative nine variables.

In Figures 1, 2 and 3, we present comparisons on the three objective functions between the four algorithms, namely:

- RSG-SA with randomly starting points drawn from the fixed intervals considered above;
- SA with randomly starting points;
- GA with randomly starting points;
- EXCEL with fixed starting points.

The subject for comparison is, as in Przechlewski *et al.* (1996), the *average bias*, computed

for each endogeneus variable, as follows:

$$BIAS_i = \left| (1/m) * \Sigma_{k=1,m}(b_i^{(k)} - B_i) \right| = \mathbf{b_i} - B_i,$$

where B_i denotes the recorded statistical value for endogeneus variable i, $b_i^{(k)}$ stands for the endogeneus variable value i obtained in the k-th run, m is the number of runs (m=10) and $\mathbf{b_i}$ is the mean value of the endogeneus variable i across the m runs. Let us take a brief moment to explain the experimental assumptions on each of the algorithms involved.

First, the RSG-SA: despite the theoretical considerations on RSG (see *Section 1*), we made an exception - for the sake of a fair comparison - by giving randomly starting points to this algorithm. Besides the technical details for SA presented in *Section 1.3*, we make the connection to RSG by defining the following objects:

- the objective function is the sum of square errors between estimated and recorded statistical values; we didn't use a second objective function (as introduced in Charemza's paper on RSG) due to the particular requirements of the problem we tackled: we are not looking for some coefficients involved in a forecasting equation (as in Charemza (1996) or in Przechlewski *et al.* (1996)), but for the initial values corresponding to solving the model for a single year;
- the learning function - λ - is assimilated to the "*temperature*" from SA, and computed by:

  $$\lambda(t) = T(t) = T_{max}/(1+\ln(t))$$

  where T_{max} is a constant value set up at 10,000, and t is the iteration index.
- the set of admissible parameter intervals is modified by the application of the learning function λ, as follows:

  $$\Theta(t) = \theta(t-1) \mp \ell(\Theta(t-1))/((\ln((10-\lambda(t)/1000))+1))$$

  where $\Theta(t)$ is the interval at iteration t, $\theta(t-1)$ is the mean value of that parameter (i.e., center of the interval) at iteration t-1, and $\ell(\Theta(t-1))$ stands for the interval's length.

On ten different runs (corresponding to ten uniform random draws for the initial values of the endogeneus variables) the number of RSG iterations was between 5 and 50, with 50 replications per iteration. This corresponded to a final value of T between 2,100 and 4,200 (note that we used an error-threshold termination criterion for the algorithm, not a maximal

number of iterations).

Second, the SA: the maximal number of cycles was 350, with 50 uniform random draws on each cycle. The temperature (arameter T, from *Section1.2*) decreased from 10,000 down to 1390 (on the longest run). As in the previous experiment, the results are averaged on 10 different runs. One must notice that the values obtained by SA are closer to the reference values (stat_val) than those obtained by GA - see below. Referring, e.g., to the domestic aggregate demand (dad) and gross domestic product (gdp), the differences for the second objective function between GA and SA are obviously favourable to SA.

Next, we present the results obtained by applying a GA with real-valued chromosomes, uniform mutation - mutation rate: 0.03 - shuffle (i.e., multi-point) crossover - crossover probability: 0.8 - disruptive selection (i.e., the "worst" chromosome is maintained from a generatio to another, in order to improve the information exchange in case of GA stagnation with the current population filled up with samples of the same individual), up to 100 chromosomes in a generation, and up to 1000 generations in one run. The initial population was randomly generated (within the fixed mentioned intervals), and the results have been averaged on 10 different runs.

As for the results obtained by EXCEL, the starting points were the statistical values corresponding to year 1996. In case of the first objective function (see *Appendix*), as the model is proved to have a unique solution, the results can be obtained with different starting points, located "close" to the initial (statistical) values. It is worth noting that for extremal starting points, the Gradient Technique needs two succesive runs (vs. one run, in case of "close" points) for reaching the desired solution. Going deeper in the Gradient algorithm procedure, one can explain this two-step behaviour by succesively missing and finding the correct search direction. Concerning the second objective function, the system solution is strongly depending on the initial values (due to the existence of multiple solutions, in this case). We stress that the choice of recorded statistical values as starting points is justified by our intention to give Gradient technique the best possible start. The third objective function was introduced for comparison purposes, since it appeared in the first version of the model (see *Appendix*).

Fig. 1. Average BIAS - Objective Function 1

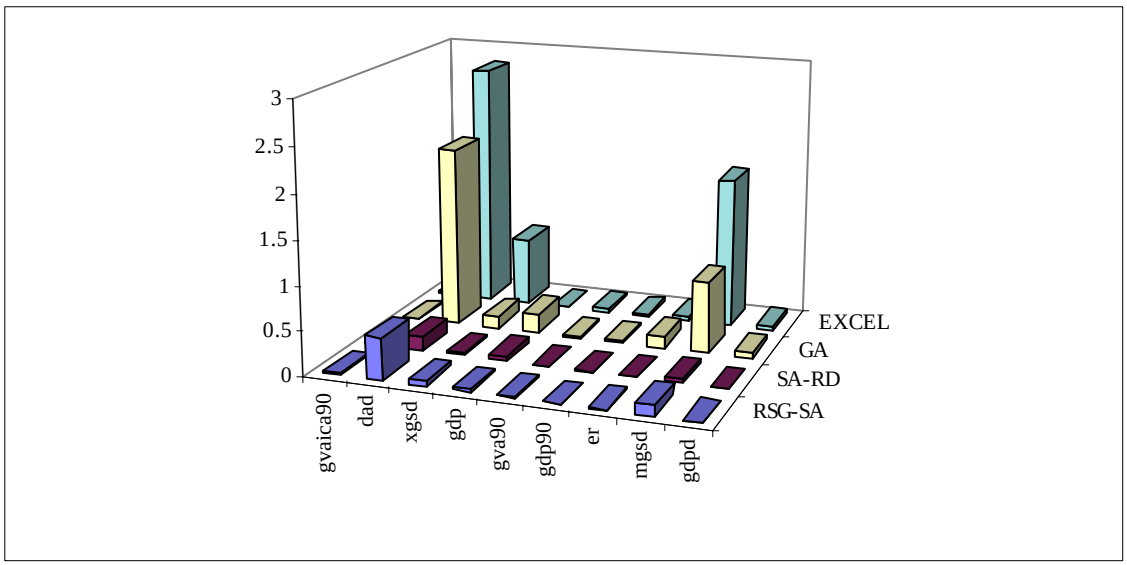


Fig. 2. Average BIAS - Objective Function 2

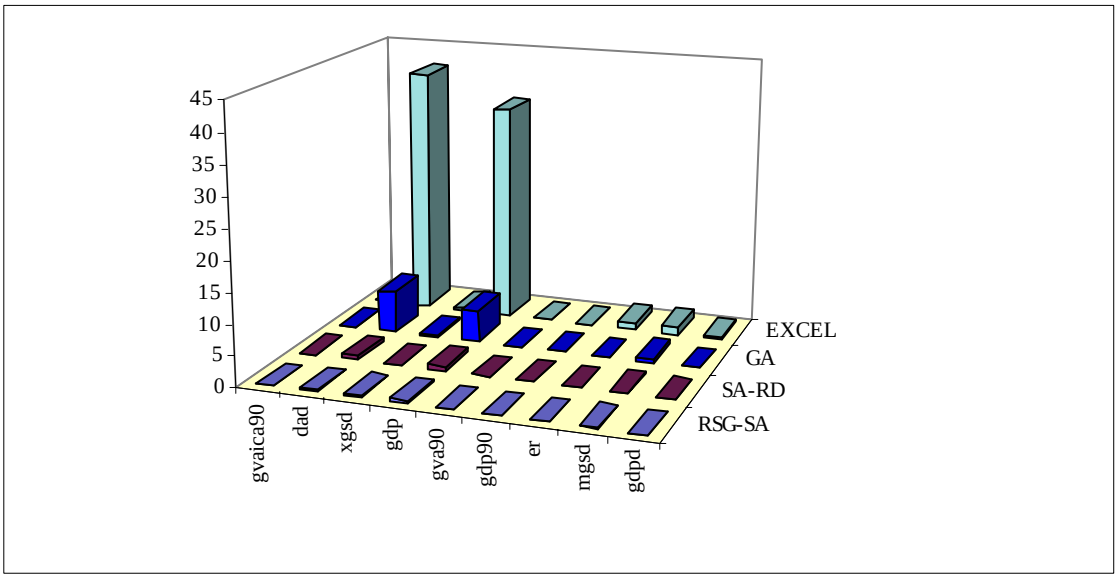
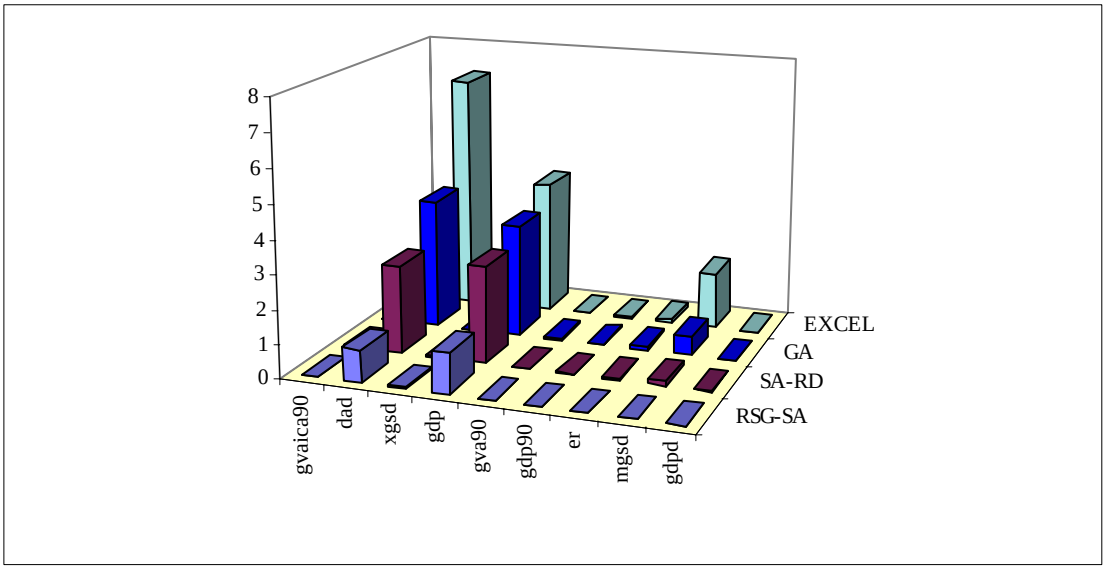


Fig. 3. Average BIAS - Objective Function 3



2.2. Finding the Best Objective Function for the Romanian Macromodel

In this section, as in the previous one, all the calculus is restricted to the 1995-1996 anual data of the economic indicators, as they appear in Dobrescu (1996).

Choosing a certain objective function is more often a subjective matter. In our case, choosing the first objective function (i.e., min $[GDP-EXTDR]^2$) could be explained by the intention to "hang" the system to the nominal side due to the presence of high inflationary expectations. This choice is supported, easily than others, because potential users of that model can "feel" easier what are supposed to be for the next period the total disposable revenues, rather than the ratio between GDP and its gross domestic deflator (GDPD). On the other hand, the model is supposed to have a unique solution level for the deflator, when the GDP equals the EXTDR, and this means determined value for all the endogeneus variables involved. The disadvantadge would be that endogeneus variables expressed in constant prices are not so well "catched", because of the inertial behaviour which makes these variables to have the solution level closer to the previous than to the current year. Another lack is the solution level for the GDPD differing from the recorded statistical one with +0.06, which makes some rates (including this one) to take negative values, instead of

positive ones - similar to the recorded statistical values.

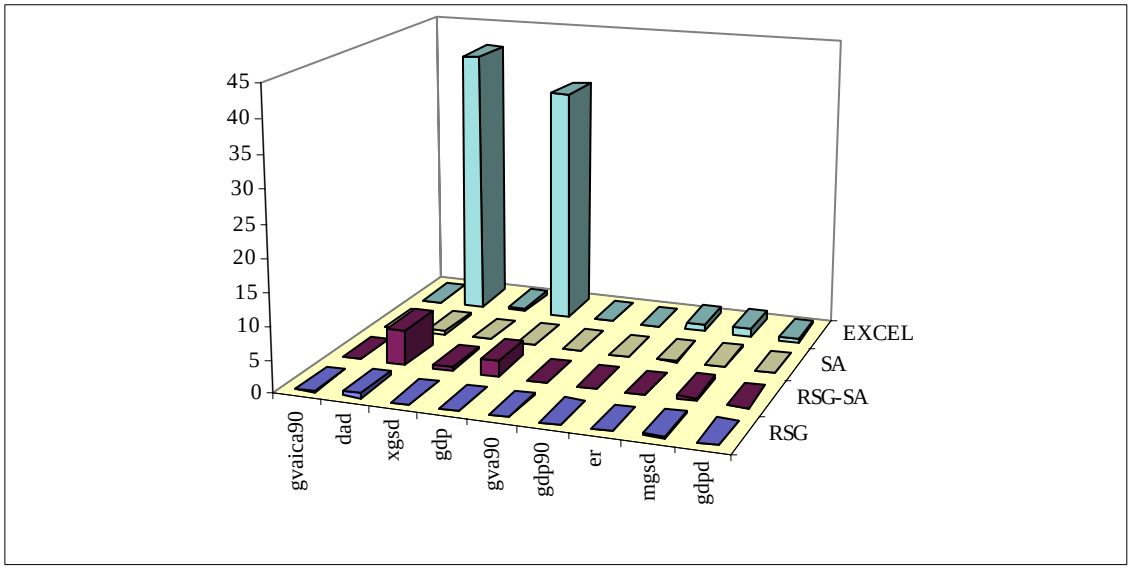
The second objective function (i.e., min $[GDP*EXGDPD/(GDPD*EXTDR)-1]^2$) expresses the model connection to the real terms endogeneus variables. In our case, the number of variables expressed in constant prices is higher than the number of variables expressed in curent prices. Also, the first ones can be put in forms which underline the ratio GDP/GDPD. In this case, two major disadvantadges occur:

- oposite to the previous situation, the nominal variables are less "catched", so one may assume that, choosing the second objective function, the user is concerned in real terms variables;
- the model has not unique solution, which makes it very sensitive to the initial start of the endogeneus variables. In fact, this makes the second objective function to be out of use if the model is solved by EXCEL only, as in this case - even starting with "perfect" values for the endogeneus variables - the Gradient technique trappes the systems in a solution point with inacceptable GDPD, GDP, DAD, ER.

If, in case of first objectiv function, the usage of optimization alternative techniques was not so obvious, now the need of several altenatives is quite ncessary. The key of the problem would be the correct chose of admissible intervals for each endogeneus variables involved. It is worth noting that, when applying stochastic search algorithms the fixing of initial intervals (i.e., defining the searching space) is not such a crucial matter, as these types of algorithms (GA, SA, RSG) are not so easily fooled up by local optima.

In Figure 4, the compared (absolute value of) average bias for the four algorithms (RSG, RSG-SA, SA, EXCEL) is depicted, w.r.t. the second objective function. The starting points were fixed at the recorded statistical values corresponding to the endogeneus variables of 1996. As one can easily notice through this comparison, the best performs RSG, followed by SA and RSG-SA. As expected, all the three stochastic techniques are outperforming EXCEL.

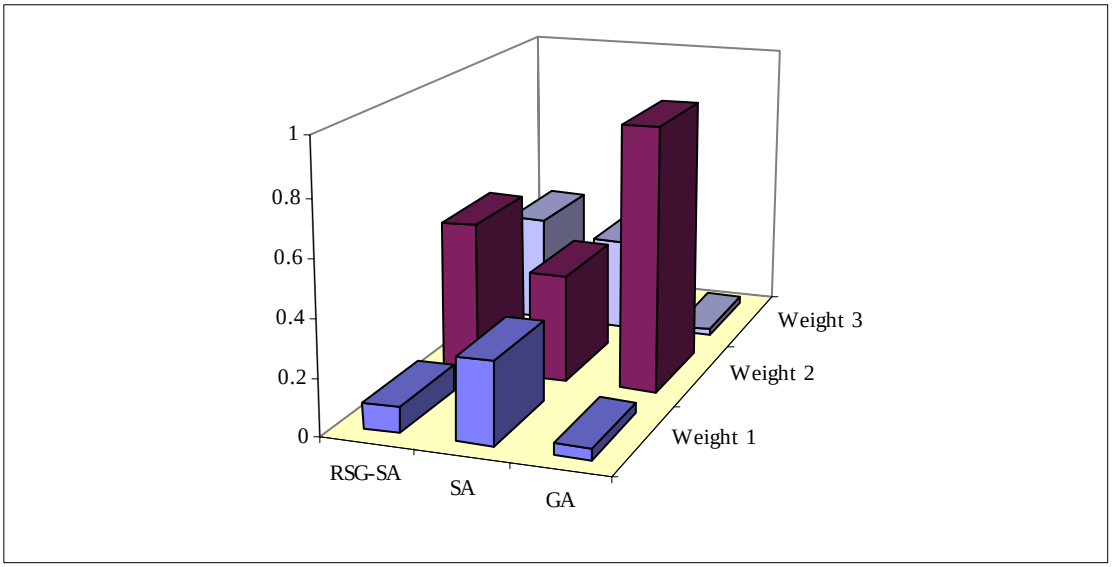
Fig. 4. Average BIAS with statistical starting points - Objective Function 2



The third objective function (i.e., min $\{[GDP*EXGDPD/(GDPD*EXTDR)-1]^2+[GDP/EXTDR -1]^2\}$)is supposed to be a connection between the model and his both nominal and real term sides. This was considered for comparison reasons, as it occured in the first version of the Romanian macromodel (see Dobrescu (1996)). The results obtained in this regard are weaker than the previous ones, perhaps due to the fact that the minimul value on this function is not achieved when (1) GDP/GDPD=EXTDR/EXGDPD and (2) GDPD=EXGDPD, but somewhere in a middle point, placed in the convex set generated by the extremal points, resulting in achieveing (1) or (2).

Hence, a new objective function (Ob.F. 1+2+3) was introduced, as a linear convex combination of the three previous ones, with three weights which are subject to optimization. Figure 5 depicts the weights (averaged on ten different runs), on the aggregate objective function, respectively by RSG-SA, SA and GA, with randomly starting points from the fixed intervals.

Fig. 5. The weights for the aggregate objective function



As one can observe in the histogram, Objective Function 2 (corresponding to Weight 2) proved to be preferred by each optimization algorithm. Yet, the GA made the difference between objective functions really striking, followed by RSG-SA and SA. For this reason, we find of interest to present the evolution of error versus objective functions' weights, over a single run, 2,000 iteration GA run - see Figure 6. One will see the progressive disjunction between the three objective functions as the error goes to zero.

Fig. 6. Error vs. objective functions' weights

3 Concluding Remarks

When one deals with a multi-modale optimization problem the gradient techniques are inefficient, so stochastic optimization techniques should be used.

Choosing the variation boundaries for each endogeneus variables can be done in a permissible manner than when applying fixed starting points algorithms. In this case, the advantage of RSG-SA over SA and GA can be explained by the mergere between the ideas of decreasing intervals (borowed from RSG) and the Metropolis acceptance crterion (specific to SA), which made the hybrid algorithm not "elitist", thus assuring a necessary diversity evolution.

When starting with fixed points, in the case of a multi-modale problem, comparing over the best starting points, RSG is proved to be the best, closely followed by SA. Both these two algorithms are proved to be more precise than RSG-SA, all of them being far superior to EXCEL Gradient method.

In Przechlewski *et al.* (1996), an interesting problem was raised: *"What is more important from the point of view of the prospective forecast accuracy - it is obtaining better parameter values, or, maybe, not so good parameter values but smaller forecast errors?"* In our paper, the second objective function (the one measuring the forecasting errors - specific to RSG) was not considered, due to the particular requirements of the problem we tackled: we are not looking for some coefficients involved in a forecasting equation (as in Charemza (1996) or Przechlewski *et al.* (1996)), but for the initial values corresponding to solving the model for a single year. Even so, an equivalent form of the above question may be addressed in our case, also. This is: "*When dealing with small-size economic model, are we more interested in obtaining solutions close to the statistical records (expecting a similar behaviour on forecasting tasks), or to find a small error model solution but accepting systematic deviations of the endogeneus variable as being inherent?"*

This may be an open question, generating several subjective opinions - this is the reason for performing a large empirical evidence, comparing to each other different optimization methods, as well as different objective functions.

The main goal of our approach consists in raising questions on the utility of different objective functions for a macromodel - a problem usually disconsidered by the a priori (and sometimes missleading) assumption that a certain objective function is proper to several years.

References

Aarts, E.H.L, Eiben A.E. and van Hee. K.M. (1989), 'A General Theory of Genetic Algorithms', Computing Science Notes, Eindhoven University of Technology.

Agapie, Ad., Agapie, Al. (1997), 'Forecasting the Economic Cycles Based on an Extension of the Holt-Winters Model. A Genetic Algorithms Approach, in *Proc. of IEEE/IAFE Conference on Computational Intelligence for Financial Engineering (CIFEr), New York, 1997*, pp. 96-100.

Charemza, W.W. (1996), 'An introduction to guesstimation: back to basics or a mockery?', in *2nd Int. Conf. on Computing in Economics and Finance, Geneva, 1996*.

Davis, T.A. (1991), 'Toward an Extrapolation of the Simulated Annealing Convergence Theory onto the Simple Genetic Algorithm', Doctoral Diss., University of Florida.

Dobrescu, E. (1996), *Macromodels of the Romanian Transition Economy*, Expert Publ. House, Bucharest.

Fogel, D.B. (1995), *Evolutionary Computation – Toward a New Philosophy of Machine Intelligence*, IEEE Press, pp. 121-143.

Goldberg, D.E. (1989), *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Publishing Company.

Goonatilake, S., Treleaven, P. (1995), *Intelligent Systems for Finance and Business*, Wiley.

Gujarati, D.N. (1997), *Basic Econometrics*, 2nd ed., McGraw Hill.

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P. (1983), 'Optimization by Simulated Annealing', *Science*, vol.220, **4598**, pp. 671-680.

Laarhoven, P.J.M, Aarts, E.H.L. (1987), *Simulated Annealing*, D. Reidel Publ. Comp., Dordrecht, Holland.

Michalewicz, Z. (1994), *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd ed., Springer-Verlag, pp. 64-69.

Rudolph, G. (1994), 'Convergence Analysis of Canonical Genetic Algorithms', *IEEE Trans. on Neural Networks* vol.5, **1**, pp. 98-101.

Przechlewski, T., Strzala, K. (1996), 'Evaluation and Comparison of the RSG and RSG-GA', *ACE Project Research Memoranda*, **17**.

Wolpert, D., Macready, W. (1997), 'No Free Lunch Theorems for Optimization', *IEEE Trans. on Evolutionary Computation* vol. 1, **1**, pp. 67-82.

APPENDIX - The Short Form of the Romanian Macromodel

The equations are:

GVAICA90=GVAICA90(-1)*(1+RICA90)

RICA90=C(1)*RID90+C(20*RIX+C(3)*RIM+C(4)*DRGCBE

RID90=DAD/(GDPD90*DAD90(-1))-1

GDPD90=GDPD*GDPD90(-1)

RIX=XGSD/XGSD(-1)-1

RIM=M2(-1)*GDPD/M2-1

DRGCBE=gcbe-gcbe(-1)

GVATO90=GVATO90(-1)*(1+RITO90)

RITO90=C(7)*RID90+C(8)*RIX

GVAPS=GVAPS/GDPD90

GVAPS=RPSBE*GCBE

GCBE=gcbe*gdp

RPSBE=RPSBE(-10+DRPSBE

DRPSBE=C(12)*RIBE90

RIBE90=GCBE/(GCBE(-1)*GDPD)-1

GVA90=GVAICA90+GVATO90+GVAPS90

DGDP90=C(15)*(GVA90-GVA90(-1))

GDP90=GDP90(-1)+DGDP90

GDP=GDP90*GDPD90

DAD=GDP-ER*(XGSD-MGSD)

ER=ER (-1)*GDPD*(1+RIERG)

RIERG=C (16)*DIR

DIR=IR+1-GDPD

XGSD=xgdp90*GDP90/ER90

xgdp90=C(20)+C(21)*RIG90(-1)+C(12)*DMX(-1)

MGSD=XGSD-rnx*GDP/ER

rnx=gcbb+DRNXBB

DRNXBB=C(24)*DERG90+C(25)

DREG90=ER/(GDPD90*ER90)-1,

where all the symbols reffer to the annual data, as follows:

GDP	Gross domestic product, current prices, trillion ROL
GDPD	Current gross domestic product deflator
GDPD90	GDP price index, 1990 = 1
	GDP90=GDP/GDPD90

$DGDP90 = GDP90 - GDP90(-1)$

$RIG90 = IGDP90 - 1$

GVAICA	Gross value added in industry , construction and agriculture, current prices, trillion ROL
	GVAICA90=GVAICA/GDPD90
	RICA90=GVAICA90/GVAICA90-1
GVATO	Gross value added in transport, post and communications, trade, financial, banking and insurance activities, real estate and other services, current prices, trillion ROL

$$GVATO90 = \frac{GVATO}{GDPD90}$$

RITO90=GVATO90/GVATO90(-1)-1

GVAPS Gross value added in public services, current prices, trillion ROL

GVAPS90=GVAPS/GDPD90

GVA	Total gross value added, current prices, trillion ROL

GVA90=GVA/GDPD90

GCBE — Expenditures of the general consolidated budget (state budget, local budgets, social insurance budget and similar funds), trillion ROL

gcbe=GCBE/GDP

GCBE90=GCBE/GDPD90

DRGCBE=gcbe-gcbe(-1)

RPSBE=GVAPS/GCBE

DRPSBE=RPSBE-RPSBE(-1)

RIBE90=GCBE90/GCBE90(-1)-1

GCBB — Surplus (+) or deficit (-) of the general consolidated budget, trillion ROL

$$GCBB = GCBR - GCBE$$

gcbb=GCBB/GDP

DAD — Domestic aggregate demand (final consumption of households and private nonprofit institutions serving households, final consumption of general government, gross capital formation), current prices, trillion ROL

XGSD — Exports of goods and services, current prices, billion USD

xgdp90 — Real export to GDP ratio

MGSD — Imports of goods and services, current prices, billion USD

ER — Exchange rate, thousand ROL per USD

M2 — Broad money (currency outside banks, demand deposits of economic agents, household deposits, time and restricted deposits, forex deposits of residents), trillion ROL

IR — Interest rate

$$dir = IR + 1 - GDPD$$

The exogenous variables of the model are: EXTDR (total disposable revenues), M2, gcbe, gcbb, IR and the endogeneous variables corresponding to the previous year.

The stationarity of the series involved in the econometric equations was tested with

the Augmented Dickey Fuller test.

The econometric functions can be separated into two different groups: the real output of the economy (depending on the gross added value) and the gross domestic product's distribution (based on the domestic aggregate demand and exports).

Considering the 29 equations, with 30 endogenous variables, one will easily notice that the system has one degree of freedom, given by variations of the GDPD. This is the reason for adding an objective function. The 1996 version of the model [Do] made use of the following objective function:

$$\min \{ [(GDP/GDPD)*(EXGDPD/EXTDR)-1]^2 + [(GDPD/EXGDPD) -1]^2 \}$$

where: EXTDR is the expected total disposable revenues and EXGDPD is the expected gross domestic product deflator.

When testing the model on the statistical data, EXTDR and EXGDPD are replaced with their statistical values.

The 1997 version has a simplified form for the objective function, which is:

$$\min \{ GDP - EXTDR\}^2$$

For forecasting purposes, estimations for EXTDR are necessary. Due to the fact that Romanian economy is characterized by inflationary expectations, estimations of EXTDR imply sociological research.

As in the extended version, the model analyzed below seems to be subject of the following remark (see [Do]) : "by a very complicate social and political mechanism, sometimes transparent and often invisible, the economy gravitates around the variables EXTDR and IR. From these variables, EXGDPD follows by the relation:

$$EXGDPD=[EXTDR/GDP(-1)]^{c1}*(EXGDPD+dir)^{c2} \text{ "}$$

Analyzing the reduced form of the macromodel equations, one will see that the ratio between GDP and GDPD is crucial in determining the real production values.

The macromodel's users, as more explicit preferred the new expression of the objective function. Also, the nominal indicator EXTDR expresses the inflation expectation. One can easily gave a mathematically prove to the fact that the macromodel does *not* need a special econometric function for the GDPD, his level being deduced both on the system and the first objective function (min $\{GDP - EXTDR\}^2$). Usually, the so-called reduced-form

equation is one that expresses an endogenously variable solely in terms of pre-determined variables and the stochastic distubances. As in this case one has no econometric determination for GDPD, the reduced-form equation is an implicit function, depending on GDP and GDPD:

$$GDP=\tau(1)*GDPD+\tau(2)*GDPD^2+\tau(3)*GDP^2/GDPD+\tau(4)*GDP*GDPD$$

where $\tau(i)$, i=1,4 are constant coefficients, their values correspondind to the year 1996 beeing equal to: $\tau(1)$=91.05252/70.70547; $\tau(2)$=-6.8612/-4.63907;$\tau(3)$=-0.--167/-0.00315; $\tau(4)$=0.00435/0.00560

Based on elementary calculus, all the endogenous variables can be express as functions of GDP and GDPD.For example, the expressions of the few main economic indicators in terms of GDP and GDPD are the following:

$$GVAPS90=\alpha(12)*GDP/GDPD+\alpha(13)*(GDP/GDPD)^2$$

$$XGSD=\xi(1)+\xi(2)*GDP/GDPD+\xi(3)*(GDP/GDPD)^2+GDPD*(\xi(4)*GDP/GDPD+\xi(5))$$

$$DAD=\zeta(1)*GDP+\zeta(2)*GDP*GDPD$$

$$GDP=a_1+a_2*GDP/GDPD+a_3*(GDP/GDPD)^2+a_4*GDP+a_5*GDPD$$

$$ER=a_6*GDPD+a_7*GDPD^2$$

$$RNX=a_8-a_9*GDPD$$

Basically, economic variables expressed in real terms are dependent on ratio GDP/GDPD, while economic variables expressed in nominal terms appear as functions on ratio GDP/GDPD and the deflator GDPD.

Even if the system is proved to have a single solution, one can not be satisfied with the "inertial" values for the economic indicators in real terms. So comes the idea of replacing the former objective function with:

$$Min\ [(GDP/GDPD)*(EXGDPD/EXTDR)-1]^2$$

for better connections in real terms. But in this case, there are infinitely combinations of GDP and GDPD with constant ratio, so the system is strongly dependent on the starting values for the endogenously variables.