G. Dobrowolski
T. Ryś
M. Żebrowski

Academy of Mining and Metallurgy
Institute for Control and System Engineering
System Research Group

ANALYSIS OF SD MODELS BASED ON EXPERIMENTS WITH FAMILIES OF TRAJECTORIES

Abstract for
The 1981 System Dynamics Research Conference

There is anquestionable need for sound and disciplined methodology of experimenting with SD models. Number of valueable papers shows various ways utilising sensivity analysis, programming of experiments and other approaches.

But should not we attack the problem more fundamentally before getting into more specific and costly analysis?

We would like propose a different kind of approach, it is analysis of SD models based on experiments with families of trajectories.

Using SD we aim at getting answers to the problems which can be generalised as following: is the specific goal attainable within acceptable initial conditions? can system unwind its trajectories within given limits /which cannot be explicite formulated in the model/ ? Even problems solved in the most classical industrial dynamics models, it is: how to explain in terms of internal structure and external shocks actual behaviour of the company? can be formulated in the above form. These general questions refer in fact not to the single but to the whole family of trajectories. Each particular question is equivalent to looking for family of model trajectories having the specific atribute. Therefore such a question may be transformed to the definition of the corresponding family. It is obvious that such families of trajectories are infinite. In approach which may be called traditional is able to perform only limited number of experiments and from its output, using his best knowledge has to draw

conclussions about system behaviour /sometimes of course utilising specific techniques for analysis/. Those experiments give in fact samples of infinite number of possible.

In this paper we propose a way of defining families of trajectories and method for finding them with finite number of simulations. The theory of method based on pattern recognition approach will be presented.

The examples of questions and corresponding definitions of families will be discussed in some detail.

The practical way of utilizing existing DYSMAP compiler in a specialised package will also be presented, including flowchart ilustrating the principle of this operation.

ANALYSIS OF SD MODELS BASED ON EXPERIMENTS WITH FAMILIES OF TRAJECTORIES

Abstract for 1981 System Dynamic Research Conference

G. Dobrowolski
T. Ryś
M. Żebrowski
System Research Group /1/
Institute for Control and System Engineering in the Academy of Mining and Metallurgy, Cracow, Poland

There is an unquestionable need for a sound and discipli-ned methodology of experimenting with SD models. A number of va-luable papers shows various ways of utilizing sensitivity analy-sis, programming of experiments and other approaches.

But should not we attack the problem more fundamentally before getting into more specific and costly analysis ?

We would like to propose a different kind of approach, it is the analysis of SD models based on experiments with families of trajectories.

Using SD we aim at getting answers to the problems which can be generalized as following: is the specific goal attainable within acceptable initial conditions? Can the system unwind its trajectories within the given limits /which cannot be formulated explicite in the model/ ? Even the problems solved in the most classical industrial dynamics models, i.e.:how to explain the ac-tual behaviour of the company in terms of the internal structure and external shocks ?, can be formulated in the above form. The-se general questions refer, in fact, not to the single but to the whole family of trajectories. Each particular question is equi-valent to looking for the family of model trajectories having the specific atribute. Therefore, such a question may be transformed

to the definition of the corresponding family. It is obvious that such families of trajectories are infinite. In the ap-proach, which may be called traditional, one is able to per-form only the limited number of experiments and from its out-put, using his best knowledge, has to draw conclusions about the system behaviour /sometimes, of course, utilizing speci-fic techniques for analysis/. These experiments give, in fact, just samples out of the infinite number of the possible ones.

In this paper we propose a formal way of defining families of trajectories and a method for finding them with the finite number of simulations. The theory of the method based on pattern recognition approach will be presented.

The examples of questions and corresponding definitions of families will be discussed in some detail.

The practical way of utilizing the existing DYSMAP compi-ler in a specialized package will also be presented, illustra-ting the principle of this operation.

Notes

/1/ System Research Group is organized and supported also by the Institute of Chemical Industry in Warsaw, Poland.

Introduction

There is an unquestionable need for a sound and disciplined methodology of experimenting with SD models. This need extends even further. We are facing the lack of the "appropriate technology" which would cover all stages of the SD approach, i.e. from the problem formulation through the model building simulation or experimenting with models and the analysis of the results. /We assume that these stages include the appropriate input data : verification, model validation and the like procedures/.

The SD models grow in size and complexity missing thus the goal of any modelling : to get the understanding of the heart of the matter which is pictured in the model through the simplification of a real world. The term "simplicity" is, of course, relative. The model may become more and more complex and may contain a great number of variables and parameters but still it can remain simple and understandable, provided that we have tools enabling us to manipulate and handle them easily through all stages of the problem solving. We would not state here such obvious rules for practical problem solving, if such tools forming discipline and methodology would exist and be sufficient. On the contrary, the demand for it exists and is growing through all twenty five years old history of SD. The lack of the ful fillment of this demand is one of the reasons standing behind the disappointment of SD, its practical applications and impact. The lack of the problems which have been solved with the help of SD combined with other tools, technics or methods can be considered an offshoot of the described situation in the field of methodology. And very often the real life problems cannot

be solved with one particular method ; this view being supported by many practitioners.

In this paper we would like to concentrate on this stage of methodology of SD application which deals with experimenting with the models. In the next section we will present the roots of our approach and idea of the method of the SD models analysis based on experiments with, so called, families of trajectories or MEFT /Method of Experiments with Family of Trajectories/. This will be followed by the section dealing with the theory which was developed and utilized in order to convert our idea and approach into the practical tool. It is very important to underline that the theoretical solution has led to the utilization of a very general theory of dynamic models in conjunction to the method based on the theory of pattern recognition. All this seems to be rather remote from the traditional SD approach. We mention this here since one of the dangers affecting also SD is something which we call the "tooler's effect". This effect takes place when people are oriented towards the application of the particular tool /favour it/ rather than towards taking the problem solving orientation.

Theoretical section is followed by the description of the implementation of MEFT in the DYSMAP compiler. The problem is formulated in terms of MEFT and then the assumptions taken for the implementation are described. The way of the implementation within the compiler is also given.

The Idea of MEFT

We have already mentioned that there is an unquestionable need for a sound and disciplined methodology of experimenting with the System Dynamics models. A number of valuable papers shows various ways of utilizing sensitivity analysis, programming of experiments and other approaches.

before getting into more specific and costly analysis ?

We would like to propose a different kind of approach, it is the analysis of SD models based on experiments with families of trajectories.

Using SD we aim at getting answers to the problems which can be generalized as following: is the specific goal attainable within acceptable initial conditions? Can the system unwind its trajectories within the given limits /which cannot be formulated explicite in the model/ ? Even the problems solved in the most classical industrial dynamics models, i.e. : how to explain the actual behaviour of the company in terms of the internal structure and external shocks ?, can be formulated in the above form. These general questions refer, in fact, not to the single but to the whole family of trajectories. Each particular question is equivalent to looking for the family of trajectories having the specific atribute. Therefore, such a question may be transformed to the definition of the corresponding family. It is obvious that such families of trajectories are infinite. In the approach, which may be called traditional, one is able to perform only the limited number of experiments and fromits output, using his best knowledge, has to draw conclusions about the system behaviour /sometimes, of course, utiliz ing specific techniques for analysis/. These experiments give, in fact, just samples out of the infinite number of the possible ones.

In this paper we propose a formal way of defining families of trajectories and a method for finding them with the finite number of simulations.

The negative side effect of traditional experimenting with SD models can be called the output information flood. From every experiment we obtain a substantial amount of information. This growing

amount tends to be more and more difficult for comparison between results, processing of data and drawing the final conclusions. With this respect MEFT may be also helpful in dealing with the "information flood". It responds to the need for simplification and suppresion of the information volume which, as we have said in the previous section, is one of the important goals of modelling.

Theory

For the simplicity of the description concerning theoretical principles of the presented method and also the conditions for its application, we start from mathematical properties of SD models.

Let us assume that each SD model can be presented as Cauche's problem in the following way:

$$\begin{cases} \dot{x} = f(x,t) \\ x(0) = x_0 \end{cases} \qquad /1/$$

provided that SD models which are written in problem oriented programming languages are not, in fact, continuous and the differential operator is realized by the numerical algorithm. Properties of the SD model can be identified by carring out the simulation of its behaviour, i.e. through obtaining the numerical solutions of the problem /1/. Simultaneously, the restriction of the model is assumed, i.e. $t \in [0,T]$; $x \in A$; $x_0 \in S$; and $S \subset A \subset R^n$; u is a number related to the number of the model levels.

Assumption 1.

For each $x_0 \in S$ there is one and only one solution of the problem /1/ /in a numerical sense/.

Let us denote this unique solution as $\varphi(t,x_0)$

The following set we will further on call the trajectory of the

SD model from the point x_o:

$$\overline{\Pi}(x_o) = \{x \in A : \varphi(t, x_o),\ t \in [0, T]\} \qquad /2/$$

The set of all model trajectories denotes then the set of starting points.

$$\overline{\Pi}(S) = \{\overline{\Pi}(x_o) : x_o \in S\} \qquad /3/$$

The described method MEFT assumes that the relation $Q \in \overline{\Pi}(S) \times \overline{\Pi}(S)$ generates the partition of the set $\overline{\Pi}(S)$ into the families of trajectories according to Q :

$$\overline{\Pi}(S)/Q = \{\overline{\Pi}_i(S_i)\}_{i \in I} \qquad /4/$$

where I is the set of families names and

$$\bigcup_{i \in I} \overline{\Pi}_i(S_i) = \overline{\Pi}(S) \qquad /5/$$

$$\overline{\Pi}_i(S_i) \cap \overline{\Pi}_j(S_j) = \emptyset \quad ;\ i, j \in I\ ;\ \emptyset \text{ - an empty set}$$

As the result of the assumption 1, the sets S_i, $i \in I$ constitute also the partition of the set S with properties /5/ ; i.e. that the families of trajectories can be represented by the respective sets of starting points.

Assumption 2.

The problem /1/ representing the SD model is continuous regarding the initial conditions with respect to the relation Q. It is the restriction of the classical continuity condition of the differential equation solution with regard to the initial conditions only to the interior of the sets S_i , $i \in I$, and it is allowed that the sets S_i consist of the finite number of disjoint fragments.

The method MEFT assumes that the global studies of SD models

properties can be reduced to the search for the families of trajectories of the model with regard to the previously defined relation Q. The relations of this very kind reflect the questions which an experimentator can ask as far as the dynamics of the model is concerned.

In order to find the families of trajectories /of the sets S_i, $i \in I$ / it will be suggested further on to utilize the method of the pattern recognition treated as an extrapolation-interpolation algorithm.

We shall give here the general definition of the pattern recognition system /1/ availing ourselves of the already used symbols so as to present the way of application of this system for the method MEFT. By the pattern recognition system we mean the aggregate:

/ S, I, R, u/

where:

S - the space of the recognition objects ; here, the set of starting points of the SD model

I - the set of the pattern names ; here, the set of names of the model trajectory families /and also of the sets S_i/

R - the family of the recognition functions :

$R \ni r : S \to I$

u - the learning function

$u : S \times I \supset U \to r \in R$

adding that the set U termed as the learning set consists of the pairs /x,i/ - the object / the starting point from the set S representing the trajectory/ and the name of the trajectories family to which the trajectory , derived from this point, belongs.

The system defined in this way will be searching for the families of trajectories in two steps. First, the learning set will be generated basing on a small number of simulation runs. It will be accomplished through the derivation of trajectories for the starting points from the random generator and determination of the names for them with reference to the relation Q. Next the families of trajectories will be pointed out through the recognition of the chosen starting points.

In the definition of the learning system symbols R,u represent the structure of this system. A considerable number of various pattern recognition systems is known and described in the literarure at present, i.e. the systems corresponding to the above given definition and at the same time suitable for application in the method MEFT.

Assumption3.
The form of the function R,u /structure of the pattern recognition system/ is such that enables, in a geometrical sense, the approximation of the sets S_i, $i \in I$ with the aid of this system.

It is very important in the method MEFT to choose the pattern recognition system adequate to the geometry of the sets S_i, i.e. to the problem of their cohesion, the type of a boundary curve, etc.

The assumptions mentioned above determine in a general way the conditions concerning the application of the method MEFT. These assumptions usually are fulfilled but there is always the possibility of checking them up at each stage of its application.

<u>DYSMAP Implementation</u>

An Illustrative Case

The implementation/2/ assumes the possibility of SD models examination through searching for the answer to the following problem. There is given the SD model and hypothetical or desirable range of levels values in which the system should contain itself in a given time. It is necessary to determine the subset of starting points from which the trajectories find their way to the target determined in the above presented way.

It i s the problem of finding two families of trajectories /two subsets of starting points related to them/ which in terms of MEFT is as follows :

There is given the target set $B \subset A$ and the time of obtaining of the target $t_B \in [0,T]$. The relation Q in this case is following :

$$\left(\Pi(x_o), \Pi(x_o)\right) \in Q \Leftrightarrow \varphi(x_o, t_B) \in B \qquad /6/$$

The set of the names I contains only two names : i_1= "in the target", i_2= "outside of the target".

In the described implementation one of the historically earliest pattern recognition system was used, namely so called NEAREST NEIGHBOUR. This system avails itself of the existence of the metrics in the space of objects / Euclidean metrics has been accepted/. The form of the recognition function is following :

$$r(x) = i \Leftrightarrow f_{\hat{\xi}}(x) = \min_{\xi} f_{\xi}(x) , \qquad /7/$$
$$f_{\xi}(x) = \sqrt{\sum_{j \in I}^{n} (x_j - \xi_j)^2} \; ; \; (\xi, i) \in U$$

Symbols used in the formula /7/ are analogical to those accepted in the preceding chapter.

On the basis of the learning set the system determines the attachment of the trajectory to the family taking into consideration only its starting point /without the run of the model/. In case of the applied pattern recognition system it will be the fam ily to which the nearest /in terms of Euklidean metrics/ starting point of the learning set belongs.

DYSMAP Simulation Language

DYSMAP /Dynamics System Modelling and Analysis Package/ is a Pre-Compiler specially designed for computer simulation of dynamic systems. Its character set and syntax are based on the DYNAMO /5/. The package is written in FORTRAN and compiles models to FORTRAN, all the routines of which may be used.

Therefore, DYSMAP belongs to the family of computer languages like, for instance, DYNAMO. The full information about DYSMAP is given in /4/.

Assumptions for Implementation

In the implementation of the method MEFT the following assumptions have been accepted which enable, with relative ease, for the implementation of the method :

1/ There are two phases of the method. The first phase closely connected with DYSMAP /simulation/ produces the results suitable for the second phase which, in turn, gives answers to the user's questions.

2/ Retaining of the compiler modular structure. The utilization of the method MEFT during the compilation of the model is realized through the MEFT routines without essential alterations in the existing routines of the compiler.

3/ Parametric control of the compiler run. Since for the purposes of the method it is necessary to have relative data concerning SD model and also a specified way of storing the simulation results, a parameter of a compiler control card was introduced, which brings about the corresponding mode of its work. Such an approach enables for the utilization of the DYSMAP package not only for the purposes of the typical simulation without any loss of the compiler time, central or outside memory but also for the purposes of MEFT. The above mentioned remarks are related to the fact that the transfer of data between the first and second phase /point1/ takes place through the external disc file.

4/ Retaining of syntax of the DYSMAP language.

For the purposes of the method MEFT it was necessary to introduce the additional NBT instruction defining the interval of the initial conditions for the model levels and also the desirable interval of the final values /target/.

Syntax of the NBT instruction in Bachus-Naur notation is following :

$\langle$instruction NBT$\rangle$ :: = NBT $\langle$the name of the level$\rangle$ = $(\langle u_1\rangle, \langle u_2\rangle, \langle u_3\rangle, \langle u_4\rangle)$

where

$\langle$the name of the level$\rangle$ :: = $\langle$the name defined by the instruction L without the time subscript /4/$\rangle$

$\langle u_i,\ i = 1,4\rangle$ :: = $\langle$number$\rangle$ | $\langle$constant$\rangle$ |*

$\langle$number$\rangle$ and $\langle$constant$\rangle$ defined as in /4/

Semantics of the instruction NBT

The instruction NBT sets up two intervals for a variable $\langle$the name of the level$\rangle$:

$[u_1, u_2,]$ - the admissible interval of initial values for a given level

$[u_3, u_4]$ - the admissible interval of final values for the given level

The following relations are required to take place between parameters u_i of the instruction :

$$u_1 \leqslant u_2\ , \qquad u_3 \leqslant u_4$$

The meaning of the parameter u_1 in case of the constant or number is obvious. The case of "*" can be presented as follows:

- if $\left\{ \begin{matrix} u_1 \\ u_3 \end{matrix} \right\}$ = "*" then $\left\{ \begin{matrix} u_1 \\ u_3 \end{matrix} \right\}$ assumes the value $-\infty$

  which corresponds with the lower bound of the variable REAL of a given computer

- if $\left\{ \begin{matrix} u_2 \\ u_4 \end{matrix} \right\}$ = "*" then $\left\{ \begin{matrix} u_2 \\ u_4 \end{matrix} \right\}$ assumes the value $+\infty$

  which corresponds with the upper bound of the variable REAL

Examples :

NBT ▿ LEVEL = (-1 Ø., Ø.3E+Ø2, -2.1E+Ø1, 3ØØØ.1)

in this case parameters u_i assume the following values:

u_1 = -1Ø.

u_2 = 3Ø.

u_3 = 21.

u_4 = 3ØØØ.1

The cas e in which parameters u_1, u_3, u_4 areconstant, defined in the instruction C, is presented below.

NBT ▿ LEV = (C1, 1Ø.1, C3, C4)

C ▿ C1 = Ø.5, C3 = 2Ø.1, C4 = 3ØØØ

NBT ▿ LE = (* , 1ØØ.1, - 3ØØØ., *)

The above case defines the following intervals of admissible initial and final values of the level LE :

$(-\infty$, 1ØØ.1) - initial values

(-3ØØØ., $+\infty$) - final values

NBT ALA = (1Ø, 1Ø, C1, C3)

In the last case the level ALA has the initial condition identi-

cal with the one in the instruction : N ALA = 1Ø. The instruction NBT format is analogical to the remaining instructions format which are described in /4/.

5/ In case of the method MEFT it is required to apply the instruction NBT for each level and not the instruction N.

6/ In case of the method MEFT there is no posibility of carrying out the typical simulation / from the viewpoint of the package/, and what follows, no possibility of carrying out re-RUNs of the model.

7/ In case of the method MEFT the alteration of the resulting code, which is generated by the compiler, brings about a number of procedures which examine conditions and transfer the data to the phase II.

8/ For the purposes of the method implementation the additional diagnostics generated by the compiler was introduced /fatal errors, warnings, information about simulation/.

The above assumptions enabled for the implementation of the method MEFT in a version described. From the user's point of view, carrying out the simulation by means of the method MEFT is divided into two phases /as it was already mentioned/ ; see Fig.4.1.

In the first phase the user should exchange all cards N for cards NBT in a source model, point out to the mode of the compiler work through the control card parameter and accomplish the single simulation /from his point of view/. In fact, the repeated simulation is accomplished in order to complete the relative set of data for the phase II.

In the second phase the user chooses the type of learning procedure availing himself of the program LEARN and also gives the answer to the question stated in the section : An Illustrative Case. It is worth while mentioning that this answer is obtained without the

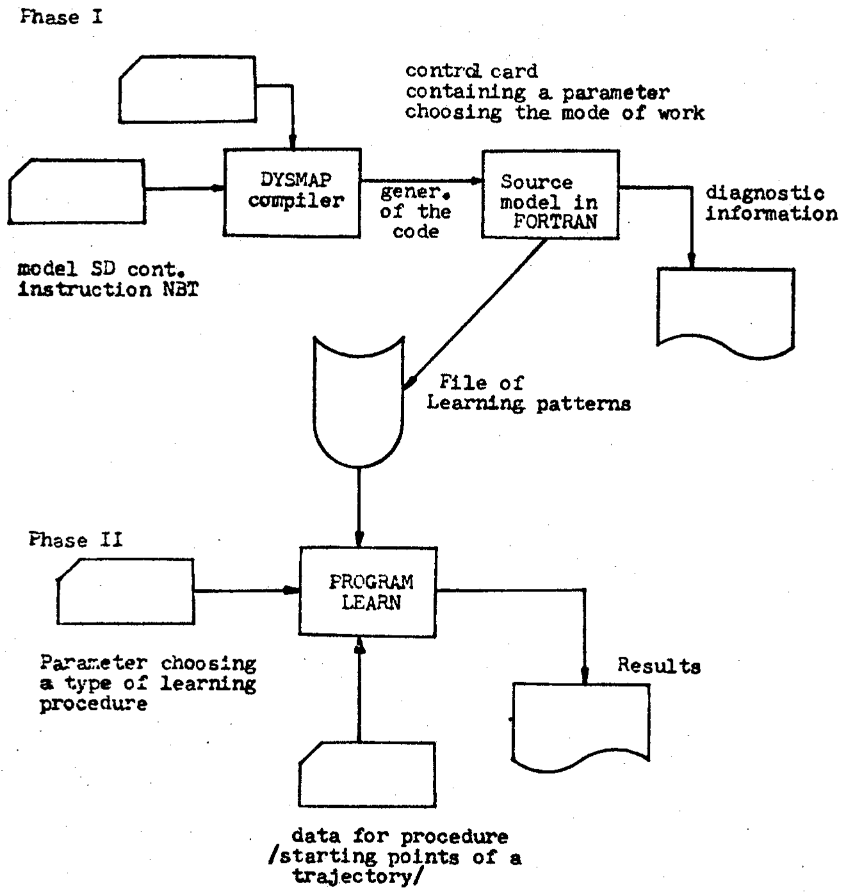
Phase I
control card containing a parameter choosing the mode of work
DYSMAP compiler
gener. of the code
Source model in FORTRAN
diagnostic information
model SD cont. instruction NBT
File of Learning patterns
Phase II
PROGRAM LEARN
Parameter choosing a type of learning procedure
Results
data for procedure /starting points of a trajectory/

Fig. 4.1.The computer implementation of the method MEFT

necessity of carrying out simulation experiments. It seems resonable to accomplish the phase II in the conversational mode.

The method MEFT has been wholly implemented in the language FORTRAN on mc. CDC CYBER 72 under O.S. SCOPE 3.4.

Summary

In this paper we have shown the method called MEFT for experiments with SD models based on the concept of the family of trajectories. The origin of the idea was displayed as the one stemming from the true demand of SD modelling. The theoretical background was given to show the mathematical foundation of the method. However, not all of the possible "MEFT problems" were displayed. For the illustration the attention was concentrated on the specific case of the MEFT application. The number of other cases is also possible which would lead, in practical terms, to the implementation of the new DYSMAP instruction or modification of parameters of those already developed. Of course, the implementation of the method is not at all limited to the DYSMAP language. It can be implemented in any other language provided that tha same formal conditions of a general not limiting type are met.

References

/1/ Grzegorz Dobrowolski- Some Aspects of Pattern Recognition Theory and its Application, The Institute for Control and System Engineering in the Academy of Mining and Metallurgy, Cracow, Poland 1980, phd /in Polish/

/2/ K.Przepłata, J.Surowiecka- Implementation of MEFT-Method for Experiments with Families of Trajectories for SD Models. MS Thesis. The Institute for Control and System Engineering in the Academy of Mining and Metallurgy, Cracow, Poland 1981.

/3/ Alexander L.Pugh III- DYNAMO User's Manual, The MIT Press Cambridge, Massachusetts, and London, England ,1967

/4/ Ajita Ratnatunga /ed. by C.J.Stewart/ - DYSMAP User Manual, System Dynamics Research Group, University of Bradford. 1979