University of Ostrava

Institute for Research and Applications of Fuzzy Modeling

Comparison of Data-Driven (Fuzzy) Modelling Methods tested on NOx Data

Martin Štěpnička, Edwin Lughofer and Viktor Pavliska

Research report No. 92

2006

Submitted/to appear:

FLLL/SCCH Master and PhD seminar '06

Supported by:

Project Aktion 41p19; Project DAR No. 1M0572 of the MŠMT ČR

University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22, 701 03 Ostrava 1, Czech Republic

tel.: +420-59-6160234 fax: +420-59-6120 478
e-mail: martin.stepnicka@osu.cz ; edwin.lughofer@jku.at ; viktor.pavliska@osu.cz

ABSTRACT:
This report is an application-experiment paper based on experimenting with real (**NOx**) data provided by Fuzzy Logic Laboratorium Linz-Hagenberg (*FLLL*) - Johannes Kepler University in Linz. The NOx data which are described below have been studied and functional dependencies between them successfully modelled by several fuzzy models identified by data-driven methods by the *FLLL* institute; some particular result can be found e.g. in [2]. Bilateral project Aktion 41p19 between *IRAFM-OU* and *FLLL-JKU* made possible to realize a deeper cooperation between both institutes. One of key issues of the proposed cooperation was (for *IRAFM*) to benefit from experiences based on many applications and industrial projects solved by *FLLL*. Second part of the issue was (for *FLLL*) to benefit from theoretical research and techniques developed in *IRAFM* and implemented in the software package LFLC2000. Based on the cited project, it was possible to obtain a real data and to make lots of experiments which enriched *IRAFM* by experiences and prompted several improvements in the techniques developed in the institute as well as changes and implementations and the software package. *FLLL* institute which cooperated on the project will be provided with all methods used in the experiments and experiment results.

1 Introduction - Data Description

Let us shortly describe a problem yielding the NOx data with a brief description of an on-line approach used in a solution realized by *FLLL*.

At an engine test bench the task was to identify a k step ahead prediction model for the emission channel NOx directly from online measurement data. The task emerged from two purposes: first to be able to have an early detection of accidents and faults in the whole emission cycle, e.g. a broken pipe which can get even dangerous for the test bench operators, and second to save expenses on a measurement sensor for NOx later on. The later aspect is due to the fact that the obtained prediction model can be used for calculating the value of NOx out of some other measured channels. Compared to a fault detection based on static models, with a k step ahead prediction models upcoming events can be earlier recognized and hence faults or even accidents prevented.

Originally, the input data matrix consisting of 6700 samples which were recorded with a certain frequency. This frequency was too high in order to obtain feasible time delays of the original channels, as shifts up to 100 steps had to be carried out producing 1600 additional channels out of the 16 original channels (for each channel 100 different shifts: $k-1, k-2, ..., k-100$) in order to get a good approximation for NOx. Thus, it turned out that a simple down-sampling by taking just each 10th point and throwing away all the others yielded a sufficient resolution. Hence, the input matrix was reduced to 670 sample, where then a time shift up to 10 was sufficient causing a manageable amount of 160 channels and finally 660 samples (due to this shift the first ten samples needed to be cut out). It is obvious that a delay in the new data matrix of $k-l$ belongs to a delay of $k-10l$ in the original one and vice versa. With the knowledge about the chosen frequency for sampling, namely 10 Hz per second, we can conclude to the real absolute delay for the impact of the input channels on NOx: 4 to 6 seconds. After applying variable selection it turned out, that at least four inputs (some original channels and their time delays) were needed in order to obtain an approximation quality higher than 0.9. The input channels for approximating NOx at time instant k consisted of the following list of channels (in the order they were selected): N = Engine Rotation Speed, P2offset = Pressure in Cylinder number 2, Te = Engine Output Torque, Nd = Speed of the Dynanometer together with their appropriate delays yielding a dynamic model in form of a four-step-ahead prediction of NOx

$$NOx(k) = f(\mathrm{N}(k-4), \mathrm{P2offset}(k-5), \mathrm{Te}(k-5), \tag{1}$$
$$\mathrm{Nd}(k-6), \mathrm{N}(k-6)) \tag{2}$$

where one step back denotes exactly one second. In this sense Te$(k-5)$ denotes linguistically 'the engine output torque five seconds ago', P2offset$(k-5)$ denotes 'Pressure in Cylinder number 2 five seconds ago', N$(k-4)$ denotes 'Engine speed five seconds ago' and so on. In this sense a four seconds ahead prediction model is yielded.

Method	**APE norm 5 features / Av. No. Fuzzy Sets / No. of Rules**
ANFIS	5.25% / 2 / 32
genfis2	4.83% / 5 / 5
genfis2 loc.	4.86% / 5 / 5
genfis2 ext. (VQ-INC)	4.92% / 4 / 4
genfis2 ext2 (VQ-INC-MOD)	4.90% / 4 / 4
FLEXFIS-MOD sample (incremental)	4.98% / 5 / 5

Table 1: Comparison of data-driven modelling methods for fuzzy systems based on NOx data

2 Original Results

Let us briefly recall some original results reached by *FLLL* which should serve us as an exemplar we would like to approach. Obviously, one could hardly expect that by general methods we can reach results of the same quality and therefore we say, that the original results serve us as an exemplar. However, from further sections it will be obvious that we do not obtain results of the same quality in all aspects but we obtain results of the same quality in some aspects, for instance we get the same accuracy of the model while we use more fuzzy rules or viceversa.

In Table 1 the model qualities as well as the model complexities are demonstrated when taking the five input variables stated above. The qualities were measured by a normalized average percent error:

$$APE\_norm = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{\max y - \min y} \tag{3}$$

From this table it is obvious, that all methods performed similar, except *ANFIS*, which produced a quite high overfitting by generating all fuzzy set combination into rules (32 in this example). It has to be noticed that FLEXFIS-MOD is an incremental variant of fuzzy system modelling and hence applicable for online processes (e.g. online fault detection), where the models should be kept up-to-date as fast as possible (it processes point per point through its algorithm). When reducing the input dimensionality to 4 respectively 3, a drop of the normalized *APE* from 4.90% to 5.04% respectively 5.64% could be observed when applying *genfis2 ext2 (VQ-INC-MOD)* and the same number of rules.

Concluding, based on these fuzzy models (with an expected deviation smaller than 5% on fresh data), it is possible to detect faults with an intensity of approximately 2 times 5% = 10% or more in newly recorded measurements.

3 Newly Used Data-Driven Methods

This section is devoted to a short description of newly used methods implemented in the software package *LFLC2000*, see [1].

First of all, we succinctly introduce inference methods and learning algorithm implemented in the program and used for modelling the NOx prediction since not all the techniques in the package are appropriate for this purpose.

3.1 Inference Techniques and defuzzifications

Fuzzy approximation with conjunctions (FAC) is in fact the well known Mamdani-Assilian [3] approach since the mathematical interpretation of the fuzzy rule base of n rules is given by the following fuzzy relation R

$$R(x, y) = \bigvee_{i=1}^{n}(A_i(x) \wedge B_i(y)), \quad x \in X \subset \mathbb{R}, y \in Y \subset \mathbb{R} \tag{4}$$

where A_i and B_i is the i-th antecedent and consequent fuzzy set, respectively.

Fuzzy approximation with implications (FAI) is also well known but for ceratin reasons much less used in applications. It is based on the mathematical interpretation of the fuzzy rule base of n rules is given by the following fuzzy relation R

$$R(x, y) = \bigwedge_{i=1}^{n} (A_i(x) \rightarrow B_i(y)), \quad x \in X \subset \mathbb{R}, y \in Y \subset \mathbb{R} \tag{5}$$

where A_i and B_i is the i-th antecedent and consequent fuzzy set, respectively. In the *LFLC2000*, the *Lukasiewicz* residuation operation is used for an interpretation of the implication between antecedents and consequents represented by $\rightarrow$ in (5).

Concerning the inference technique, it is realized by the well known computational rule of inference (CRI) proposed by Lotfi A. Zadeh, see [11]. It is obvious that for a crisp input x' and a singleton fuzzifier the CRI gives an equivalent result to the simple evaluation of the fuzzy rule base interpretation at the node x' i.e. the inference is equal to $R(x', y) \underset{\sim}{\subset} Y$ where R is given either by (4) or by (5).

For every single one approach a different defuzzification method has to be used. For the first approach based on conjunctive interpretation of rules, the *center of gravity* (COG) is used and for the second one based on implicative interpretation rules *mean of maxima* (MOM) is used.

3.2 Learning

For this paper, we consider the word *learning* in a wider sense i.e. every single method leading to an automatic generation of a fuzzy rule base will be considered to be a learning so the word is not understood necessarily on a neural point of view.

Expertly based linguistic approaches like *Perception based logical deduction* [4] have the advantage they the fuzzy rule base can be built expertly. Since the problem of prediction NOx values is nothing else but an approximation of the data and there is no expert knowledge of the system some learning method has to be used.

LFLC2000 has been equipped with the so called *linguistic learning* [1] for an automatic generation of a linguistic fuzzy rule base appropriate for the perception based logical deduction. The learning can be described as follows: If new measured input/output pair (x_j, y_j) comes find the most appropriate linguistic fuzzy sets (A_j, B_j) where $A_j \underset{\sim}{\subset} X$ and $B_j \underset{\sim}{\subset} Y$ and create a rule

$$\text{IF } x \text{ is } \mathcal{A}_j \text{ THEN } y \text{ is } \mathcal{B}_j \tag{6}$$

where $\mathcal{A}_j$ and $\mathcal{B}_j$ are just evaluating linguistic expressions represented by fuzzy sets A_j and B_j, respectively. In this way, huge rule base is created and then duplicate rules erased as well as complexity, inconsistency (conflicts in rules) and redundancy by sophisticated algorithms solved.

For the both fuzzy approximation methods, no learning has been implemented yet and the real NOx project demanded its development and implementation as a part of the Aktion project.

For this stage of investigation, a learning based on the same principle as the linguistic one was implemented. Compared to the linguistic one, it does not need a consistency and redundancy analysis there are no fully overlapping fuzzy sets in the fuzzy rule base (usually uniform triangles). On the other, as well as other usual axis based approaches it suffers from the curse of dimensionality.

3.3 Fuzzy Transform

Another fuzzy modelling method which was employed in this experiment is the fuzzy transform (F-transform) [6]. It is a fuzzy approximation method (approximating a functional dependency i.e. a continuous function $f : X \rightarrow Y$) based on two transforms - a direct one and an inverse one. It deals with a fuzzy partition of the domain X given by fuzzy sets called *basis functions* $A_i \underset{\sim}{\subset} X$ $i = 1, \ldots, n$ fulfilling several conditions including the Ruspini condition [7]

$$\sum_{i=1}^{n} A_i(x) = 1 \quad \forall x \in X. \tag{7}$$

Usually, the technique deals with triangular shaped fuzzy sets or sinusoidal shaped fuzzy sets. For details see [5] or [6].

The *direct F-transform* is a discrete simplified representation of the function f given by a real vector $[F_1, \ldots, F_n]$ where

$$F_i = \frac{\int_X f(x)A_i(x)dx}{\int_X A_i(x)dx} \tag{8}$$

and if the function is given only at (measured) samples $(x_j, f(x_j))$ $j = 1, \ldots, m$ where $m >> n$, in principle, then

$$F_i = \frac{\sum_{j=1}^{m} f(x_j)A_i(x_j)}{\sum_{j=1}^{m} A_i(x_j)dx}. \tag{9}$$

The *inverse F-transform* is again a continuous function on X and it is given by a linear combination of the basis functions and the components F_i of the direct F-transform i.e.

$$f_n^F(x) = \sum_{i=1}^{n} F_i A_i(x). \tag{10}$$

In the terminology used in the previous subsection, we can say that the inverse F-transform is an inference method (belonging to singleton models, close to T-S rules of the 0th order) while the direct F-transform is its learning algorithm.

The recalled approximation method can be easily generalized for functions with more variables, see [9, 10]. In the *LFLC2000* there is such a method implemented for an arbitrary number of variables, in principle.

4 FAI and FAC Results

Based on the NOx data consisting of a file of training samples and a file of $N = 159$ testing samples provided by the *FLLL* we have tested the *LFLC2000* techniques described above. To have a relevant comparison we have measured the accuracy by the correlation coefficient and by the normalized average percent error (3) as well as in the original case. Moreover, since we model a process of prediction NOx during an engine activity a speed of a chosen inference method including its defuzzification method is of a high interest.

In Table 2 there is an overview of some chosen results reached by the methods implemented in the *LFLC2000* software. By methods FAI and FAC we do not mean only fuzzy approximation with implications and conjunctions, respectively as inference methods but the whole data-driven method in the software i.e. fuzzy approximation learning, deleting duplicate rules, respective defuzzifiaction method (COG for FAC and MOM for FAI). Number of fuzzy sets expresses how many uniform triangular fuzzy sets on each axis have been used for generating the rule base. The speed of inference expresses time needed for getting all output values for 159 inputs from testing samples. This information is very rough and imprecise since always depends on used hardware, installed operational system and running software (in our case Intel® Pentium® IV 1.7GHz, 512MB SDRAM, Win XP Professional).

Obviously, it can be stated that the result do not reach such accurate values as in the original case but tend to them. Unfortunately, only if the number of rules increases rapidly.

5 Improvements

Clearly, the biggest problem is hidden in the curse of dimensionality since no fuzzy cluster analysis is used. To decrease the number of rules while keeping the given methods since their interpretability is very high, if we understand it as a possibility to interpret, say linguistically, the model or its unique rules, a new algorithm has to be implemented. Of course, in case of a high number of rules, a possibility of understanding them decreases and moreover, speed of an inference method including its respective defuzzification method decreases as well. The following algorithm has been implemented.

Var.	Method	No. of F. Sets	No. of Rules	APE norm	Correlation	Speed
3	*FAI*	5	83	9.25%	0.595	5.56 s
3	*FAC*	5	83	9.33%	0.734	2.88 s
3	*FAI*	7	182	8.59%	0.666	10.60 s
3	*FAC*	7	182	6.57%	0.849	4.16 s
3	*FAI*	10	362	7.49%	0.751	18.50 s
3	*FAC*	10	362	5.96%	0.859	5.61 s
4	*FAI*	7	323	6.01%	0.862	20.00 s
4	*FAC*	7	323	5.72%	0.875	7.77 s
5	*FAI*	5	181	8.18%	0.731	14.60 s
5	*FAC*	5	181	7.74%	0.842	7.56 s
5	*FAI*	7	350	6.10%	0.863	22.50 s
5	*FAC*	7	350	5.81%	0.879	8.66 s

Table 2: Some chosen results by *LFLC2000*.

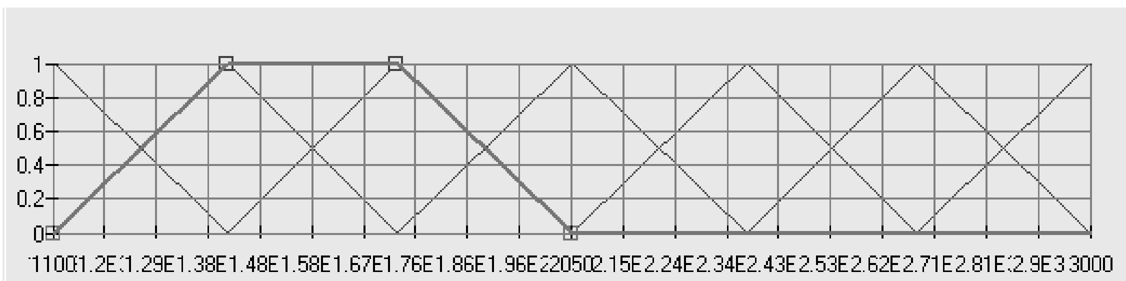

Figure 1: An example of two uniform triangular shaped fuzzy sets merged into a trapezoidal one.

Let on ith axis K triangular fuzzy sets A^i_k $k = 1, \dots K$ are created for $i = 1, \dots, I$. Let a fuzzy approximation rule base is generated by the introduced algorithm. If there are two rules of the form

$$\text{IF } x_1 \text{ is } \mathcal{A}^1_{k_1} \text{ and ... and } x_i \text{ is } \mathcal{A}^i_{k_i} \text{ and ... and } x_I \text{ is } \mathcal{A}^I_{k_I} \text{ THEN } y \text{ is } \mathcal{B}, \tag{11}$$

$$\text{IF } x_1 \text{ is } \mathcal{A}^1_{k_1} \text{ and ... and } x_i \text{ is } \mathcal{A}^i_{k_i+1} \text{ and ... and } x_I \text{ is } \mathcal{A}^I_{k_I} \text{ THEN } y \text{ is } \mathcal{B} \tag{12}$$

i.e. consequent fuzzy sets and antecedent fuzzy sets excepting one ith antecedent fuzzy set are equal and the $i$the antecedent fuzzy sets are neighboring half-overlapping fuzzy sets then the rules are merged to one

$$\text{IF } x_1 \text{ is } \mathcal{A}^1_{k_1} \text{ and ... and } x_i \text{ is } \mathcal{T}^i_{k_i\_k_i+1} \text{ and ... and } x_I \text{ is } \mathcal{A}^I_{k_I} \text{ THEN } y \text{ is } \mathcal{B} \tag{13}$$

where the linguistic expression $\mathcal{T}^i_{k_i\_k_i+1}$ is represented by a trapezoidal fuzzy set

$$T^i_{k_i\_k_i+1}(x) = A^i_{k_i+1}(x) \oplus A^i_{k_i+1}(x) \tag{14}$$

and where $\oplus$ is the Łukasiewicz t-conorm.

Similarly, if there is (after several merging) a rule, which on some antecedent axis is totally overlapped by some trapezoidal fuzzy set from another rule and fuzzy sets on the other axes are equal in both rules, then this rule is erased. In fact, instead of clustering data before an automatic generation of rules, this algorithm clusters already generated rules.

In Table 3, there is a short overview of some chosen results

Var.	Method	No. of Fuzzy Sets	No. of Rules	APE norm	Correlation	Speed
3	*FAI*	5	6	10.85%	0.585	0.33 s
3	*FAC*	5	6	10.85%	0.745	0.30 s
3	*FAI*	7	18	11.27%	0.503	0.41 s
3	*FAC*	7	18	7.87%	0.830	0.48 s
5	*FAI*	5	18	9.76%	0.631	0.97 s
5	*FAC*	5	18	7.47%	0.842	0.89 s
5	*FAI*	7	64	8.17%	0.719	2.34 s
5	*FAC*	7	64	7.55%	0.802	1.75 s

Table 3: Some chosen results by *LFLC2000* after merging rules.

6 F-transform Results

Finally, the F-transform method was tested on NOx data as well. This method suffers from the curse of dimensionality a bit less then the previous fuzzy approximation techniques although it simply creates all possible combination of fuzzy subdomains and searches for their discrete representatives. The advantage is that the inverse F-transform is computationally extremely simple and fast. It is just a linear combination of fuzzy sets and there are no time requirements for a defuzzification. The difference can be seen in the speed columns of Tables 2, 3 and 4.

So, the curse of dimensionality does not cause a computational complexity growth and inference speed problems. Interpretability is very natural since the F-transform method in fact claims that

$$\text{IF } x_1 \text{ is } \mathcal{A}^1_{k_1} \text{ and ... and } x_i \text{ is } \mathcal{A}^i_{k_i} \text{ and ... and } x_I \text{ is } \mathcal{A}^I_{k_I} \text{ THEN } y \text{ is } F_i. \tag{15}$$

Although one can hardly expect that from a huge number of, say "rules", a human can understand relationships between variables, the F-transform with its interpretability can easily provide a user with a local type of information i.e. what happens on some chosen subdomains.

7 Comparison, Conclusions and Further Work

All in all, it can be stated that the *LFLC2000* methods did not bring as good results as the original approach [2] in all monitored features (No. of rules, accuracy, speed etc.) at ones. On the other hand, in some concrete features the results can be at least as good if we allow e.g. higher number of rules. Especially the F-transform method can be significantly useful in getting high accuracy and speed properties.

The fuzzy approximation methods *FAI* and *FAC* has been demonstrated to be useful universal methods which are especially effective in lower number of input variables. Otherwise, they suffer from curse of dimensionality some cluster analysis has to be used to improve their results. Within the Aktion project, some promising improvements based on the rule merging algorithm have been reached.

The F-transform algorithm featured by extremely high inference speed and accuracy below 5% available. Its high number of fuzzy set combinations (curse of dimensionality) did not influence the speed. The interpretability is natural as well but the transparency of such model composed of hundreds or even more combinations is questionable.

Another key issue of the data-driven model is hidden in incremental vs. batch type of learning (also on-line vs. off-line). Every single method in the *LFLC2000* is implemented for an off-line learning applications and therefore all results in the tables in the previous sections are results of batch models. On the other hand, any of them can be in principle modified to an incremental one without a significant influence.

Fuzzy approximation models *FAI* and *FAC* do learn from individual incoming data and step by step generate individual rules. The suggested rule merging algorithm can be used in on-line merging as well without any change.

Variables	F. Set Shape	No. of Fuzzy Sets	APE norm	Correlation	Speed
3	*triang.*	2	10.59%	0.824	0.05 s
3	*sinus.*	2	9.65%	0.869	0.05 s
3	*triang.*	5	5.70%	0.891	0.06 s
3	*sinus.*	5	5.55%	0.879	0.03 s
4	*triang.*	3	7.23%	0.875	0.03 s
4	*sinus.*	3	6.88%	0.876	0.05 s
4	*triang.*	5	4.77%	0.925	0.05 s
4	*sinus.*	5	4.91%	0.904	0.05 s
5	*triang.*	2	9.59%	0.824	0.05 s
5	*sinus.*	2	8.45%	0.829	0.06 s
5	*triang.*	4	6.10%	0.863	0.06 s
5	*sinus.*	4	5.23%	0.897	0.06 s
5	*triang.*	6	4.69%	0.908	0.06 s
5	*sinus.*	6	4.62%	0.912	0.06 s

Table 4: Some chosen results by the F-transform method in *LFLC2000*.

In case of the F-transform, it is a bit more complicated since formula (9) is a typical batch formula. On the other hand, if the partial summations are kept in a memory of the learning machine, the coefficients F_i can be on-line modified. Moreover, an on-line neural approach to the F-transform based on the gradient descent method was already suggested, see [8]. This approach unfortunately has not been tested on NOx data since it is programmed only for single-input-single-output problems yet. On the other, we consider it to be a promising method which should be tested within the on-going work since it also allows us to adapt the *basis functions* which increases accuracy and decreases number of fuzzy sets.

Acknowledgement

This investigation has been supported by projects Aktion 41p19 and DAR 1M0572 of the MŠMT ČR.

References

[1] Dvořák A., Habiballa H., Novák V. and Pavliska V. (2003), The Software Package LFLF2000 - Its Specificity, Recent and Perspective Applications. *Computers in Industry*, 51, 2003, 269–280.

[2] Lughofer E. (2005), Data-Driven Incremental Learning of Takagi-Sugeno Fuzzy Models, PhD-Thesis, Department of Knowledge-Based Mathematical Systems, University Linz, Austria.

[3] Mamdani E. and Assilian S. (1975), An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. *International Journal of Man-Machine Studies*, 7, 1975, 1–13.

[4] Novák V. (2005), Perception-Based Logical Deduction as Alternative Approximate Reasoning Method. *Proc. of FUZZ-IEEE '05*, Reno, Nevada, USA, 1032–1037.

[5] Perfilieva I. (2003) Fuzzy Approach to Solution of Differential Equations with Imprecise Data: Application to Reef Growth Problem. In: *Fuzzy Logic in Geology* (R.V. Demicco and G.J. Klir, Eds.), Academic Press, Amsterdam, Chap. 9, 275–300.

[6] Perfilieva I. (2006), Fuzzy Transforms. *Fuzzy Sets and Systems*, in press.

[7] Ruspini E. H. (1969), A New Approach to Clustering. *Inform. and Control*, 15, 22–32.

[8] Štěpnička M. and Polakovič O. (2006), The Fuzzy Transform from the Neural Network Point of View. *Proc. IPMU 2006*, submitted.

[9] Štěpnička M. and Valášek R. (2004), Fuzzy Transforms and Their Applications to Heat Flow Equation. *Abstracts of the 2nd FLLL/SCCH Master and PhD Seminar 2004*, 51–56.

[10] Štěpnicka M. and Valášek R. (2005), Numerical Solution of Partial Differential Equations with Help of Fuzzy Transform. *Proc. of FUZZ-IEEE '05*, Reno, Nevada, USA, 1104–1109.

[11] Zadeh L. A. (1973), Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Trans. Systems, Man and Cybernet.* **SMC-3**, 28–44.