Genetic Algorithm and Decision Tree based Oscillatory Stability Assessment

S. P. Teeuwsen, *Student Member, IEEE*, I. Erlich, *Member, IEEE,*
M. A. El-Sharkawi, *Fellow, IEEE* and U. Bachmann

***Abstract* – This paper deals with a new method for eigenvalue region prediction of critical stability modes of power systems based on decision trees. The critical stability modes result from inter-area oscillations in large-scale interconnected power systems. The existing methods for eigenvalue computation are time-consuming and require the entire system model that includes an extensive number of states. However, using decision trees, the oscillatory stability can be predicted based on a few selected inputs. Decision trees are fast, easy to grow and provide high accuracy for eigenvalue region prediction. Special emphasis is hereby focused on the selection process for the decision tree inputs. In this work, a genetic algorithm is implemented to search for the best set of inputs providing the highest performance in stability assessment.**

***Index Terms*– Feature Selection, Genetic Algorithm, Decision Tree, Oscillatory Stability Assessment, Large Power Systems**

I. Introduction

Inter-area oscillations in large-scale power systems are becoming more common especially for the European interconnected power system UCTE/CENTREL. The system experiences inter-area oscillations associated with the swinging of many machines in one part of the system against machines in other parts. Moreover, for certain load flow conditions, the system damping changes widely [1], [2]. The reasons for inter-area oscillations are different. On one side, the system grew in the last decade extremely fast. On the other side, the deregulation of electricity markets in Europe aggravated the problem because it led to an increased number of long distance power transmissions. In Germany, the network is becoming more stressed also by the transmission of wind power from the Northern coastal region to the Mid- and Southern consumption areas. As a matter of fact, the European power system was originally designed rather as a backup system to maintain power supply in case of power plant outages. The system is operated by several independent transmission utilities, interconnected by a large meshed high voltage grid. In the current competitive environment operators need more computational real-time tools to manage the system safety. Specially, there is need for fast power system stability assessment. The use of on-line tools is even more complicated since Transmission System Operators (TSO) exchange only restricted information. Each TSO controls a particular part of the power system, but the exchange of data between different parts is limited to a small number because of the high competition between the utilities. Of main interest in the European power system is the Oscillatory Stability Assessment (OSA). It can be assessed by the classical small-signal stability computation, which requires the entire system model. But it is time consuming for large power systems [3] and therefore, not applicable for online use. This paper proposes the use of Computational Intelligence (CI) for a fast on-line OSA based on Decision Trees (DT). The method needs only a small set of inputs for predicting eigenvalues with sufficient accuracy. Different CI methods are applicable for OSA [4] – [6]. However, DTs are particularly suitable due to their suitability for engineering decision making. Besides, it is computationally fast and provides always definite results.

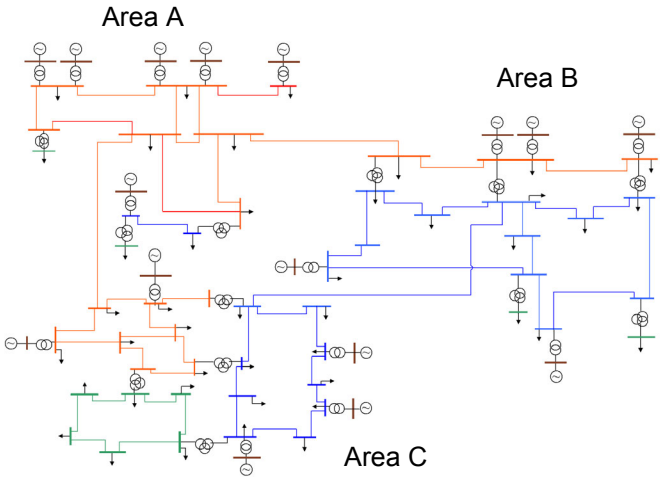


Fig. 1 One-Line Diagram of the PST 16-Machine Dynamic Test System

A 400/220 kV 16-machine dynamic test system shown in Figure 1 is used for demonstration in this study. The network consists of 3 strongly meshed areas, which are connected by long distance transmission lines. Therefore, the system experiences inter-area oscillations. It has been developed based on characteristic parameters of the European power system [7]. Since the operating point in real power systems changes continuously, 5 different operating conditions are considered. These operating conditions include high and low load situations like in winter and in summer, and changes of the network topology when transmission lines are switched off.

S. P. Teeuwsen, University of Duisburg-Essen, Germany, (teeuwsen@uni-duisburg.de)
I. Erlich, University of Duisburg-Essen, Germany, (erlich@uni-duisburg.de)
M. A. El-Sharkawi, University of Washington, Seattle, USA, (elsharkawi@ee.washington.edu)
U. Bachmann, Vattenfall Europe Transmission GmbH, Berlin, Germany, (udo.bachmann@vattenfall.de)

To generate learning samples for DT construction, various load flow scenarios under the 5 operating conditions are considered. These scenarios are generated by real power exchange between 2 areas. The different load flow scenarios result in 5,360 patterns for learning of the DT.

This paper is organized as follows: Section II introduces DTs and its implementation. Section III proposes a new DT based method for OSA. The selection of inputs for OSA is described in Section IV. The selection method requires understanding the OSA process and is explained at the end of this paper. The selection of inputs is performed off-line only once in the beginning. The selected inputs are then the same for each OSA run.

II. Decision Trees

DT techniques belong to CI methods and became highly popular in the age of modern computers. They are based on a sequence of questions that can be answered by either yes or no. Each question queries whether a predictor satisfies a given condition, whereby the condition can be both continuous and discrete. Depending on the answer to each question, one can either proceed to another question or arrive at a response value. DTs can be used for non-linear regression (Regression Tree) when using continuous variables or they can be used for classification (Classification Tree) when using discrete classes. When used for feature selection, the DT is grown as a regression tree [8].

Without prior knowledge of the nonlinearity, the regression tree is capable of approximating any non-linear relationship using a set of linear models. Although regression trees are interpretable representations of a non-linear input-output relationship, the discontinuity at the decision boundaries is unnatural and brings undesired effects to the overall regression and generalization of the problem [9].

A. Tree Growing

To construct a tree, the data is divided into two sets. One set is used to learn the tree and the other set is used to test it afterwards. For tree growing, there are different algorithms available depending on the kind of tree desired. Regardless of the algorithm, the first task is to find the root node for the tree. The root node is the first node splitting the entire data set into two parts. Therefore, the root must do the best job in separating the data. The initial split at the root creates two new nodes, called branch nodes. The algorithm searches at both branch nodes again for the best split to separate the sub sets, and following this recursive procedure, the algorithm continues to split all branch nodes by exhaustive search until either a branch node contains only patterns of one kind, or the diversity cannot be increased by splitting the node. The nodes, where the tree is not further split, are labeled as leaf nodes. When the entire tree is split until only leaf nodes remain, the final tree is obtained [8] – [10].

1) Growing a Classification Tree

In classification, the algorithm splits the patterns in such a way, that each branch node *t* performs best in splitting the data into separate classes. An error measure $E(t)$ is defined to quantify the performance at node *t*. It is also referred to as impurity function ϕ describing the impurity of the data or classes under a given node. If all patterns belong to the same class, the impurity attains a minimum value at zero. If the patterns are equally distributed over all possible classes, the impurity function will reach a maximum value.

When p_j is the percentage or probability of cases in node *t* that belong to class *j* in the *J*-class problem, the error or impurity measure $E(t)$ of node *t* is computed by the impurity function ϕ in node *t* by

$$E(t) = \phi(p_1, p_2, \ldots, p_J) \tag{1}$$

with

$$\sum_{j=1}^{J} p_j = 1 \text{ and } 0 \le p_j \le 1 \tag{2}$$

and the impurity measure for the complete tree is computed by the summation of all branch node impurities. However, the impurity function ϕ can be computed in different ways, the best known functions for classification are the *Entropy Function* Equation (3) and the *Gini Diversity Index* Equation (4). The *Entropy Function* is also known as *Maximum Deviance Reduction*. Both functions are always positive unless all patterns belong to the same class. In this case, the functions obtain zero impurity. They obtain their maximum value when the patterns are equally distributed over all possible classes.

$$\phi = -\sum_{j=1}^{J} p_j \cdot \ln(p_j) \tag{3}$$

$$\phi = 1 - \sum_{j=1}^{J} p_j^2 \tag{4}$$

2) Growing a Regression Tree

When growing a regression tree, a local model is employed to fit the tree at node *t* to the data set at node *t*. In analogy to most regression methods, the error measure is implemented by the least square approach. The mean squared error or residual at node *t* is computed by

$$E(t) = \frac{1}{N_t} \cdot \min_{\boldsymbol{\theta}} \sum_{i=1}^{N_t} \left(y_i - d_t(\mathbf{x}_i, \boldsymbol{\theta})\right)^2 \tag{5}$$

where $\mathbf{x}_i$ is the input vector. The corresponding output (target) is y_i and N_t is the number of patterns at node *t*. The local model at node *t* is given by $d_t(\mathbf{x}, \boldsymbol{\theta})$, whereby $\boldsymbol{\theta}$ is a vector with modifiable regression parameters. If $d(\mathbf{x}, \boldsymbol{\theta}) = \theta$ is a constant function independent of the input vector $\mathbf{x}_i$, the local model $d_t(\mathbf{x}, \boldsymbol{\theta})$ at node *t* is computed by the mean value of the target y_i, which is

$$\overline{y_t} = \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} y_i \tag{6}$$

since any subset of y will minimize Equation (5) for the mean of y as local model, and thus

$$E(t) = \frac{1}{N_t} \cdot \sum_{i=1}^{N_t} \left(y_i - \overline{y_t}\right)^2 \quad (7)$$

If $d(\mathbf{x}, \boldsymbol{\theta})$ is a linear model with linear parameters $\boldsymbol{\theta}$, a least squares method is applied to identify the minimizing parameters $\overline{\mathbf{y}}$ [8] and [9]. According to the chosen model, the tree is split in order to maximize the error decrease for the complete tree. The process of learning is fast and applied once at the beginning. When the tree is grown, it can be used to predict an output depending on the presented inputs. It has been observed, that large decision trees usually do not retain their accuracy over the whole space of instances and therefore tree pruning is highly recommended [11].

B. Tree Pruning

Pruning is the process of reducing a tree by turning some branch nodes into leaf nodes, and removing the leaf nodes under the original branch. Since less reliable branches are removed, the pruned DT often gives better results over the whole instance space even though it will have a higher error over the training set. To prune a tree, the training set can be split into the growing set (for learning the tree) and the pruning set (for tree pruning). Different pruning approaches use the testing data for pruning. However, pruning is necessary to improve the tree capability and reduce the error cost. Moreover, large trees become specific on the used growing data and some lower branches might be affected by outliers.

Pruning is basically an estimation problem. The best tree size is estimated based on the error cost. When only training data are used to prune a tree, the estimation is called internal estimation. Accuracy is computed by counting the misclassifications at all tree nodes. Then, the tree is pruned by computing the estimates following the bottom-up approach (post-pruning). The resubstitution estimate of the error variance for this tree and a sequence of simpler trees are computed. Because this estimation probably under-estimates the true error variance, the cross-validation estimation is computed next. The cross-validation estimate provides an estimate of the pruning level needed to achieve the best tree size. Finally, the best tree is the one that has a residual variance that is no more than one standard error above the minimum values along the cross-validation line [8].

III. Oscillatory Stability Assessment

The DT based OSA performs eigenvalue region prediction as an assessment method, which is described in detail in [5] and [6]. The prediction method requires that the observation area in the complex eigenvalue space is defined first. The observation area is located in the region of insufficient damping in the range between 4% and –2%. Then, this area is sampled along the real axis between 4% and –2%. This is done for different frequencies. The sampling results in a set of 47 sampling points, whereby the number of sampling points depends on the sampling step widths along the real and imaginary axis. Small step widths result in a high number of sampling points to process, but increase the assessment accuracy. The distances between sampling points and dominant eigenvalues are computed and the sampling points are activated depending on these distances. The closer an eigenvalue to a sampling point, the higher the corresponding activation. Once this is computed for the entire set of patterns, the DT is grown using these activations. The DT can be used in real-time to compute the sampling point activations. Then, the activations are transformed into predicted regions where the eigenvalues are obviously located. These predicted regions are characterized by activations higher than a given limit.

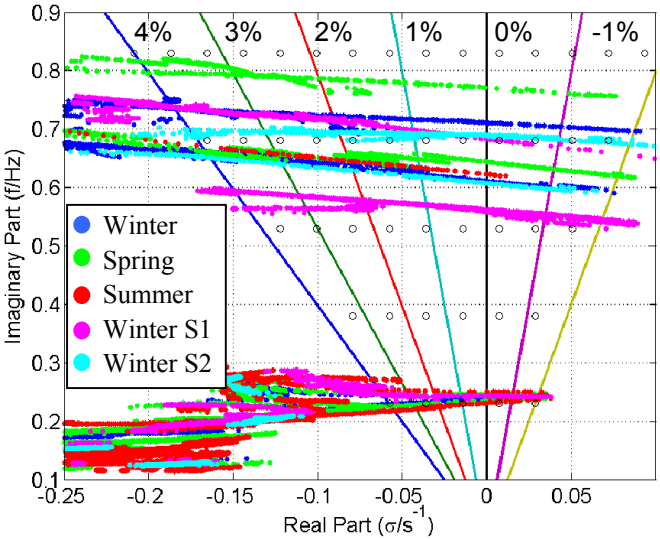


Fig. 2 Eigenvalues for different Operating Conditions and Sampling Point Locations in the Observation Area

The dominant eigenvalues from the PST 16-Machine Test System for 5 operating conditions and the sampled observation area is shown in Figure 2. The sampling points are marked by circles.

A. Distance Computation

The width of one σ sampling step is called Δσ, the width of one f sampling step is called Δf. After the observation area is sampled, the sampling points are activated according to the positions of the eigenvalues. Thus, the distance between the eigenvalues and the sampling points is used to compute the activation for the sampling points. The eigenvalues are defined by their real part σ_{ev} and their frequency f_{ev}. The sampling points are defined by their location (σ_s, f_s). Then, the distance d between a given eigenvalue and a given sampling point is computed as follows:

$$d = \sqrt{\left(\frac{\sigma_s - \sigma_{ev}}{k_\sigma}\right)^2 + \left(\frac{f_s - f_{ev}}{k_f}\right)^2} \quad (8)$$

Because σ and f use different units and cannot be compared directly, they are scaled. Hence, σ and f are divided by the

constants k_σ for the real part and k_f for the frequency, respectively. The maximum possible distance between an eigenvalue and the closest sampling point occurs when the eigenvalue is located exactly in the geometrical center of 4 neighboring sampling points. This is shown in Figure 3.

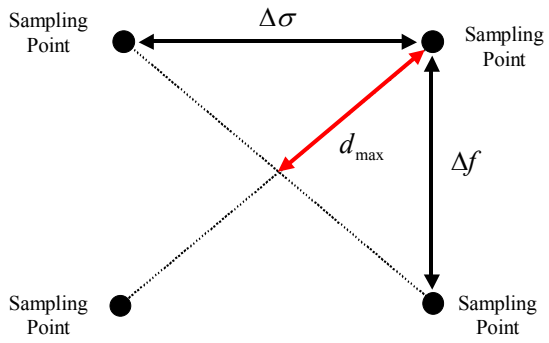


Fig. 3 Definition of the Maximum Distance $d_{\max}$

According to Figure 3 and Equation (8), the maximum distance can be computed as

$$d_{\max} = \sqrt{\left(\frac{\frac{\Delta\sigma}{2}}{k_\sigma}\right)^2 + \left(\frac{\frac{\Delta f}{2}}{k_f}\right)^2} \tag{9}$$

B. Sampling Point Activation

Based on this maximum distance, the activation value *a* for a sampling point is defined as a linear function depending on the distance *d* between a sampling point and an eigenvalue:

$$a = \begin{cases} 1 - 0.5 \cdot \dfrac{d}{d_{\max}} & 0 \le d \le 2d_{\max} \\ 0 & d > 2d_{\max} \end{cases} \tag{10}$$

The activation *a* is computed for one given sampling point and all eigenvalues resulting from one pattern. The final activation value *act* for the given sampling point is the summation of all activations *a*

$$act = \sum_{i=1}^{n} a \tag{11}$$

where *n* is the number of considered eigenvalues. The maximum distance, Equation (9), and the activation function, Equation (10), lead to the minimum activation for a sampling point when at least one eigenvalue is nearby, which means closer than $d_{\max}$:

$$act \ge 0.5 \quad \forall \quad d \le d_{\max} \tag{12}$$

C. Scaling

The success of the prediction depends strongly on the choice of the scaling parameters. These parameters impact both the training process of the CI method and the accuracy of the predicted region. However, there are different possible approaches for scaling. From Equation (8), the distances between neighboring sampling points in σ and *f* direction are

$$d(\Delta\sigma, 0) = \frac{\Delta\sigma}{k_\sigma} \tag{13}$$

$$d(0, \Delta f) = \frac{\Delta f}{k_f} \tag{14}$$

The sampling step widths Δσ and Δ*f* are constant. Assume $d_{\max}$ is constant for comparison purposes. Hence, there are 3 different approaches for choosing k_σ and k_f. Since the eigenvalues move mostly parallel to the real axis, the distance $d(\Delta\sigma, 0)$ between two neighboring sampling points along the real axis can be equal, smaller, or greater than the maximum distance $d_{\max}$. Table I shows examples for the 3 different scaling approaches described above.

TABLE I
EXAMPLES FOR SCALING APPROACHES

#	k_σ	k_f	$d_{\max}$	$d(\Delta\sigma,0)$	$d(0,\Delta f)$
S1	$\frac{\Delta\sigma}{1.2}$	$\frac{\Delta f}{1.6}$	1.00	1.200	1.600
S2	$\Delta\sigma$	$\frac{\Delta f}{\sqrt{3}}$	1.00	1.000	1.732
S3	$\frac{\Delta\sigma}{0.624}$	$\frac{\Delta f}{1.9}$	1.00	0.624	1.900

Note that the choice of scaling approach S2 leads to a distance of 1 between neighboring sampling points in s direction. If an eigenvalue is located exactly on the position of a sampling point, its neighboring sampling points are still activated according to Equation (10) and Equation (12) because the distance of 1 leads to an activation of 0.5. Thus, at least 2 sampling points are activated. For scaling approach S1, the 2 sampling points are only activated once, when the eigenvalue is in the middle of them and both are affected. Otherwise, only 1 sampling point is activated. Finally, using scaling approach S3, at least 3 sampling points are activated because of the short distance between the sampling points. However, approach S2 is chosen for the following computations.

D. Eigenvalue Mapping

Eigenvalue mapping is the procedure of transforming sampling point activations back to eigenvalue positions. The activation values given from the sampling points are used to setup an activation surface, which is used to construct a region in which the eigenvalue is located. The activation surface is shown in Figure 4. It is constructed by linear interpolation between all rows and columns of sampling points.

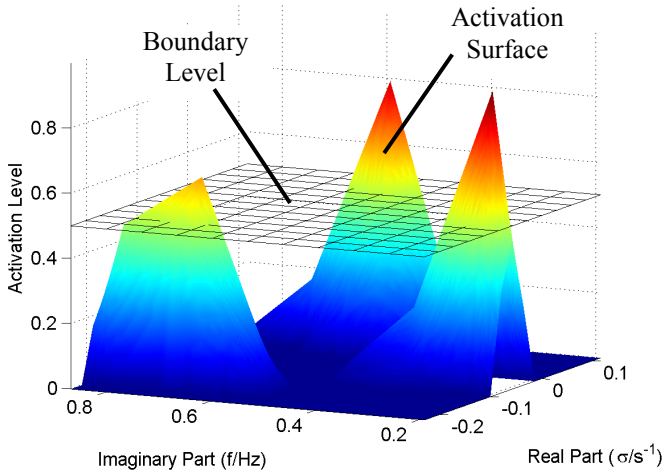


Fig. 4 Activation Surface constructed by Sampling Point Interpolation and Boundary Level

From Equation (12), the minimum activation value is 0.5 when at least one eigenvalue is nearby. Therefore, the surface at the constant level of 0.5 is called boundary level. However, the intersection of the activation surface and the boundary level leads to a region, which is called predicted eigenvalue region. To illustrate the eigenvalue mapping, an example is given in Figure 4. The figure shows the interpolated sampling points for one certain load flow scenario. When the activation surface is set up, the predicted eigenvalue regions can be constructed easily by the intersection of the activation surface and the boundary level. Figure 5 shows the view from above onto the boundary level. Thus, the predicted eigenvalue region in the complex eigenvalue plain, plotted in red color, can be determined. In this case, the real eigenvalue locations, marked with black dots, are inside the predicted eigenvalue regions.

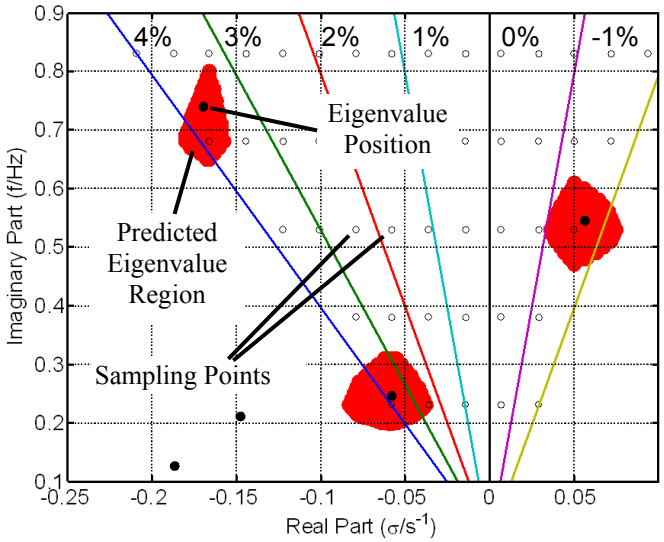


Fig. 5 Complex Eigenvalue Plain with Predicted Eigenvalue Regions and Eigenvalue Locations

IV. Feature Selection by Genetic Algorithm

Stability assessment methods require input information to make reliable predictions. These inputs should characterize the system sufficiently. Besides, the variables must be measured and made available by the utilities. Large power systems include much state information such as transmission line flows, generated powers and demands, voltages, and voltage angles. However, any effective CI method like DT requires a small number of inputs, which must be selected first from the full set of variables. When used with too many inputs, CI methods lead to long processing times and cannot be managed well [4]. In general, the selected input features must represent the entire system, since a loss of information in the reduced set results in loss of both performance and accuracy in the CI methods. In large interconnected power systems, it is difficult to develop exact relationships between features and the targeted oscillatory stability. This is because the system is highly nonlinear and complex. For this reason, feature selection cannot be performed by engineering judgment or physical knowledge only, but it must be implemented according to a mathematical procedure or algorithm. In the CI science, different methods are developed for this purpose. The selection technique applied in this study is based on physical pre-selection followed by a GA based on DT. Before the GA is applied, the feature set is pre-selected by engineering judgment, and the data are normalized. The complete feature selection process is shown in Figure 6.

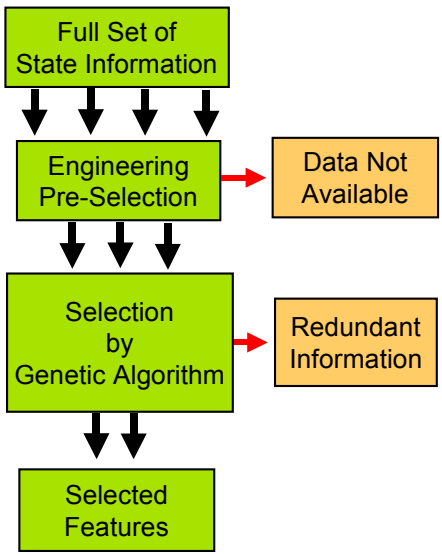


Fig. 6 Basic Concept of the Feature Selection Procedure

A. Feature Pre-Selection

The key idea of the pre-selection is collecting as many data from the power system as possible, which are assumed to be of physical interest for OSA. The focus is hereby on those features, which are both measurable in the real power system and available from the power utilities. Altogether there are 252 pre-selected features from the PST 16-Machine Test System. Note that the pre-selected features contain also voltage angles. Voltage angles are not measured in power systems according to the current practice, but it should be

emphasized that voltage angles provide important information about the system stability. Moreover, they can be measured if necessary since the necessary equipment already exists. However, the used features in this study are generator related features, which are the generated real and reactive power of individual machines and their summation per area. The rotating generator energy is defined as the installed MVA of running blocks multiplied by the inertia constant. Since the number of running blocks is adjusted during the computation depending on the generated power, this feature provides information about the rotating mass in the system. The rotating generator energy is computed for both individual machines and all machines in the same area. Moreover, the real and reactive power on all transmission lines in the system and the voltages as well as the voltage angles on all bus nodes are used as features because they contain important information about the load flow in the system.

B. Genetic Algorithm

The GA is a global search technique based on the evolutionary process in nature. It is highly applicable in case of discrete selection or mixed integer programming. The GA is an optimization algorithm to search an input space while minimizing an objective function under given constraints [12].

The GA involves a population of individuals, which include coded information. Individuals performing poorly on the given problem are discarded while more successful ones produce variants of themselves. Based on the Darwinian analogy of "survival of the fittest", the population will improve over time and obtain one optimal solution of the problem. However, the solution found may be a local optimum and is not necessarily a global optimum. In this study, the individuals are of binary type containing *zeros* and *ones*. Each individual has one bit for each feature, which can be selected from the total feature set. This is a size of 252 bits. Any bit with a value of *one* means that the corresponding feature is selected. The entire population consists of many individuals starting at different initial selection conditions. The fitness for those individuals is computed by an objective function based on DTs. In the process, the objective function grows one DT for each of the 47 sampling points introduced in section III. The DTs are grown with 4,824 learning data patterns and the error is evaluated with testing data only. When z is the DT target and y the DT output, the Sum Squared Error (SSE) is computed over all 536 testing patterns and accumulated over all 47 DTs. Thus, the error is computed according to

$$Error = \sum_{j=1}^{47}\sum_{i=1}^{536}(z_{ij} - y_{ij})^2 \qquad (15)$$

When an input selection shows a minimal error, the DTs perform excellent in computing the activations of all sampling points in the observation area, and thus the eigenvalue region prediction leads to highly accurate results. The block diagram of the GA using DT as objective function is shown in Figure 7.

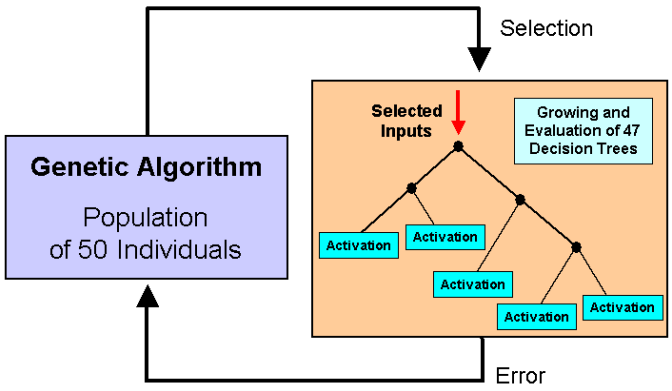


Fig. 7 Block Diagram of the Genetic Algorithm using an Objective Function based on 47 Decision Trees to compute Sampling Point Activations

The GA evaluates in each generation (iteration step) the fitness (error) for all individuals. The individuals showing the best selection and therefore the smallest error will be chosen as parents for the next generation. The next generation is created based on the fitness of the individuals and a reproduction and crossover process to find the best parents, which will generate the best offspring. The reproduction involves a random process with a particular probability for being chosen as parent. Crossover is performed on the result and mutation applied randomly to allow discontinuities in the search algorithm, which may lead to significant improvement. Individuals, which involve less than 5 or more than 40 features, will not be considered in the selection process. Any other individual may give a solution.

In the initial step, 50 binary individuals were created including the initial selections. Hereby is considered to start with individuals, which contain only few selected features in the range of 10-20. The GA was applied over 120 generations. The error defined in Equation (15) versus the generations is shown in Figure 8.

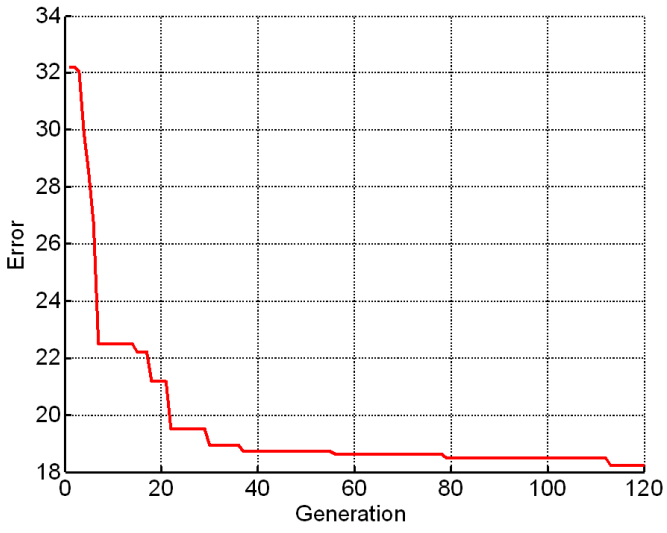


Fig. 8 GA Optimization Error versus Generations

Depending on the number of generations, there is a different solution found by the GA including a selection showing a high performance. Table II shows final results for selections at different generational steps within the GA when evaluated by DT and eigenvalue region prediction. Each solution is hereby the best of its generation.

TABLE II
FINAL GA RESULTS AFTER DIFFERENT GENERATIONS

Generation	Selected Inputs	Learning Error	Testing Error
15	12	0.89 %	2.96 %
30	13	0.96 %	2.66 %
45	16	0.99 %	2.37 %
60	18	0.99 %	2.66 %
75	19	0.99 %	2.66 %
90	19	0.99 %	2.66 %
105	20	0.99 %	2.66 %
120	20	0.93 %	2.66 %

V. Conclusion

This study proposes the implementation of DTs for on-line OSA. The DT method is based on a small number of inputs only, which is an important issue in a liberalized power market. When applied on-line, the DT is fast and provides accurate information for TSOs. The OSA method computes a predicted eigenvalue region, which provides the TSO with information about the position of dominant eigenvalues. The TSO gains knowledge about the distance to system instability and blackouts.

Since the DT method is both fast and not influenced by any random process, it can be implemented for a GA based feature selection. The GA will find the best set of DT input features based on the DT evaluation with testing data. The GA finds not only the best combination of features but also the best size for the feature set. The selection by GA is performed only once in the beginning. When the best input selection is determined, the selected input features will the same for each OSA run.

VI. References

[1] U. Bachmann, I. Erlich and E. Grebe, "Analysis of interarea oscillations in the European electric power system in synchronous parallel operation with the Central-European networks", IEEE PowerTech, Budapest 1999

[2] H. Breulmann, E. Grebe, et al., "Analysis and Damping of Inter-Area Oscillations in the UCTE/CENTREL Power System", CIGRE 38-113, Session 2000

[3] P. Kundur, Power System Stability and Control, McGraw-Hill, New York, 1994

[4] L.A. Wehenkel, Automatic Learning Techniques in Power Systems, Kluwer Academic Publishers, Boston, 1998

[5] S.P. Teeuwsen, I. Erlich, U. Bachmann, "Small-Signal Stability Assessment of the European Power System based on Advanced Neural Network Method", IFAC 2003, Seoul, Korea, September, 2003

[6] S.P. Teeuwsen, I. Erlich, M.A. El-Sharkawi, "Small-Signal Stability Assessment based on Advanced Neural Network Methods", IEEE PES General Meeting, Toronto, Canada, July, 2003

[7] S.P. Teeuwsen, I. Erlich, M.A. El-Sharkawi, "Neural Network based Classification Method for Small-Signal Stability Assessment", IEEE PowerTech, Bologna, Italy, June, 2003

[8] Breiman, et al., Classification and Regression Trees, Wadsworth Int., Calif., 1984

[9] R. Jang, Neuro-Fuzzy and Soft Computing, Prentice Hall, NJ, 1997

[10] J.R. Quinlan, C4.5 Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, Calif., 1993

[11] H. Kim, G.J. Koehler, "Theory and Practice of Decision Tree Induction", Omega, Vol. 23, No. 6, pp.637-652, 1995

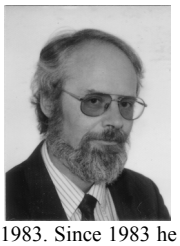
[12] K.F. Man, K.S. Tang, S. Kwong, Genetic Algorithms, Springer, London, 1999

VII. Biographies

Simon P. Teeuwsen (1976) is presently PhD student in the Department of Electrical Power Systems at the University of Duisburg-Essen/Germany. He started his studies at the University of Duisburg in 1995. In 2000, he went as exchange student to the Unversity of Washington, Seattle, where he performed his Diploma Thesis. After his return to Germany in 2001, he received his Dipl.-Ing. degree at the University of Duisburg. He is a member of VDE, VDI, and IEEE.

Istvan Erlich (1953) received his Dipl.-Ing. degree in electrical engineering from the University of Dresden/Germany in 1976. After his studies, he worked in Hungary in the field of electrical distribution networks. From 1979 to 1991, he joined the Department of Electrical Power Systems of the University of Dresden again, where he received his PhD degree in 1983. In the period of 1991 to 1998, he worked with the consulting company EAB in Berlin and the Fraunhofer Institute IITB Dresden respectively. During this time, he also had a teaching assignment at the University of Dresden. Since 1998, he is Professor and head of the Institute of Electrical Power Systems at the University of Duisburg-Essen/Germany. His major scientific interest is focused on power system stability and control, modelling and simulation of power system dynamics including intelligent system applications. He is a member of VDE and IEEE.

Mohamed A. El-Sharkawi received the B.Sc. degree in electrical engineering in 1971 from Cairo High Institute of Technology, Egypt, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of British Columbia, Vancouver, B.C., Canada, in 1977 and 1980, respectively. In 1980, he joined the University of Washington, Seattle, as a Faculty Member. He served as the Chairman of Graduate Studies and Research and is presently a Professor of Electrical Engineering. He is the Vice President for Technical Activities of the Neural Networks Society. He organized and taught several international tutorials on intelligent systems applications, power quality and power systems, and he organized and chaired numerous sessions in IEEE and other international conferences. He is a member of the editorial board and Associate Editor of several journals, including the IEEE TRANSACTIONS ON NEURAL NETWORKS.



Udo Bachmann (1952) received his grad. Engineer degree in electrical power grids and systems from the Leningrad Polytechnic Institute /Russia in 1977. After his studies, he worked in Berlin in the field of development and management by renewal and reconstruction of power grid protection. From 1980 to 1983, he joined the Department of Electrical Power Plant and Systems of the Leningrad Polytechnic Institute again, where he received his Ph.D. degree in 1983. Since 1983 he worked in the National Dispatch Center as Engineer and senior specialist in the field of management of grid protection from system view. During the last 15 years he is responsible both for steady state and dynamic stability computation and short circuit computation as well as network reactions in the Vattenfall Transmission Company (former VEAG Vereinigte Energiewerke AG).