An Intelligent System for Risk Classification of Stock Investment Projects

Antoaneta Serguieva\textsuperscript{a}, Tatiana Kalganova\textsuperscript{b}, Tariq Khan\textsuperscript{a}
\textsuperscript{a}Department of Business and Management
\textsuperscript{b}Department of Electronic and Computer Engineering
Brunel University, Uxbridge UB8 3PH, UK
Email: Antoaneta.Serguieva@brunel.ac.uk

The proposed paper demonstrates that a hybrid fuzzy neural network can serve as a risk classifier of stock investment projects. The training algorithm for the regular part of the network is based on bidirectional incremental evolution proving more efficient than direct evolution. The approach is compared with other crisp and soft investment appraisal and trading techniques, while building a multimodel domain representation for an intelligent decision support system. Thus the advantages of each model are utilised while looking at the investment problem from different perspectives. The empirical results are based on UK companies traded on the London Stock Exchange.

\textbf{Keywords:} finance, bidirectional incremental evolution, multimodel knowledge representation.

1 Introduction

Standard investment appraisal techniques have been continuously revised. The criteria have been reoptimised due to investment irreversibility [14] and the impact of a project on the investor’s total risk [45]. It has been realised that the removal of any of the perfect market assumptions destroys the foundation and reduces the effectiveness of the methods. Alternatively, a fuzzy criterion does not attempt to cope with a specific drawback of standard techniques but permits into the calculations as much uncertainty as the market could possibly suffer. The outcome is an effective method under restricted information, uncertain data and market imperfections. A fuzzy criterion and investment rating technique are first introduced in [5], then considered in a broader framework of accumulation and discount models in [10], and recently modified with an alternative fuzzification of the project duration in [30]. While those studies are theoretical in nature, the empirical results are a major emphasis in [23,40,42,43], where stock projects are evaluated and UK companies traded on the London Stock Exchange are considered. Simultaneously, the analysis of the empirical solutions to the fuzzy criterion facilitates there the induction of three general conclusions. An investment risk measure, an estimate of the project robustness towards market uncertainty modelled with the fuzzified data, and an alternative ranking technique based on the two measures.

When compared with previous studies, the proposed paper demonstrates the following advantages. First, nine representative projects are chosen from the database employed in [40], thus consistently emphasising the empirical results. Second, the projects are rated according to a modification of the risk measure suggested there, extending further the developed method. Third, a fuzzy-valued criterion is formulated and a regular fuzzy neural network, trained with a genetic algorithm, approximates its solution. Hence, the benefits of various soft techniques are blended to achieve a synergy in handling the investment appraisal problem. In comparison, [5,6,10,23,30] only study fuzzy criteria. Forth, the network is hybridised to discriminate between low-risk and high-risk projects. The threshold is agent-dependent, communicating the acceptable levels of risk. The variety of market agents work within diverse risk ranges. In the extreme, the behaviour of an investment fund differs from the behaviour of an individual investor. Consequently, an agent-dependent threshold will benefit the decision-maker. Fifth, an efficient training algorithm is suggested for the regular part of the fuzzy
network. Genetic algorithms are a promising tool in training fuzzy networks and recent studies successfully apply direct evolution to optimise the fuzzy weights in small regular networks resembling particular types of univariable fuzzy functions [8]. In comparison, here the network approximates a multivariable criterion and the number of nodes depends on the investment horizon. The complexity of the problem requires a corresponding evolutionary strategy and the implementation of bidirectional incremental evolution is suggested. Incremental strategy has been already applied to evolve neural networks controlling a robot’s motion [18]. In addition to incremental evolution, the bidirectional strategy further incorporates divide-and-conquer evolution. Thus the complex task is first gradually divided into simpler subtasks, then each subtask is evolved separately, and lastly the evolved subsolutions are merged incrementally to optimise the overall solution. The technique allows overcoming the stalling effect in direct evolution and has already proved more efficient in evolving logic functions [24].

Finally, the developed classifier is compared with other classical and soft investment appraisal and trading methods, and the advantages of each approach are utilised while building a multimodel knowledge domain for an intelligent decision support system. This will allow looking at the investment problem from various conceptual or purpose angels and choosing an adequate technique for each situation. A beneficial feature of such knowledge organisation, in comparison with single-model representations, is the flexibility it offers in overcoming specific model drawbacks. Navigating within a space of multiple models has been already applied to technical and medical domains [4,26,27,29]. There, the increased capability of the related intelligent systems in supporting users through various tasks is examined, and particularly in overcoming limitations inherent in mono-model approaches. In the investment domain, such system will enhance the ability of the decision-maker to analyse projects from multiple perspectives, and will support him in solving problems with varying objectives and complexity.

The proposed approach is developed for the following reasons. First, there have been suggested fuzzy techniques for investment appraisal and based on them ranking procedures [5,40], but no investment classifier is yet considered. Therefore, a decision would only be taken after applying the fuzzy criterion to all available projects, then rating them accordingly, and finally choosing the acceptable opportunities. The introduction of a classifying system will significantly simplify the process, especially for a large number of continuously updated projects and regularly taken investment decisions. Once trained, the network will be an effortless instrument in the hands of the decision-maker, whenever the information available is subject to change. Second, standard neural networks have been successfully applied to classify takeover targets [17]. Third, there have been already developed soft intelligent trading systems and artificial stock markets [12,32,46,47,48] that will further benefit from the introduction of the multiple modelling-perspective approach. It will power an investment agent to perform preliminary project evaluation and rating, risk analysis and classification of admissible stocks, and trading simulation.

Figure 1 describes the interrelations between financial, soft computing and artificial intelligence techniques, as well as between theoretical and empirical studies. All these approaches are involved into the process of formulating the investment classifier and building the multimodel intelligent system. The organisation of the paper follows a step by step approach, outlining the soft classifier in Section 2, building the multimodel knowledge representation in Section 3, and presenting the empirical results in Section 4. The classifier is introduced by first formulating in subsection 2.1 a fuzzy-valued criteria that is better capable of investment evaluation under uncertain market data then a crisp criteria. Next, in subsection 2.2, the structure of a hybrid fuzzy neural network is identified that is effective in approximating the above criteria. Finally, in subsection 2.3, a genetic algorithm is suggested and a bidirectional incremental evolutionary strategy is developed that is efficient in training the fuzzy network. The second part of the paper ends by stating the advantages of the soft investment classifier. In the third part, a multimodel investment domain is built up including the developed classifier. First, in subsection 3.1, a number of relevant models are referred to and their characteristics are analysed. The models are chosen with the purpose to give the
intelligent system the ability to perform various tasks. From preliminary crisp evaluation of a project, through more informative fuzzy investment appraisal and ranking, going on with project risk and robustness estimation, an agent-dependent risk classification and recommendation of attractive projects, and finally suggesting stock-trading strategies. Next, in subsection 3.2, the position of the models is considered according to several conceptual perspectives as rigidity, resolution, precision, and a multiperspective domain space is constructed. The paper ends with presenting empirical results based on UK companies traded on the London Stock Exchange, and discussing the advantages of the multimodel intelligent system.

**Figure 1:** Technique interrelations in formulating the investment classifier and building the multiple-model intelligent system

Boxes in bold follow the development of the proposed approach.
2 Soft Classifier

2.1 Fuzzy valued criterion

The need for classical investment model revision and the grounds for new types of models are well recognised. Various modifications [14,45] eliminate only some of the problems associated with the standard criteria and it has been realised that the removal of any of the perfect market assumptions typically destroys the foundations of generally accepted investment-selection techniques. In [45], it is emphasised that having rules which lead to correct decisions in the presence of capital market imperfections would lead to a situation where modern finance theory can be applied consistently. Furthermore, the mathematics underlying the standard financial techniques neglects extreme situations and regards large market shifts as too unlikely to matter. Such techniques may account for what occurs most of the time in the market, but the picture they present does not reflect the reality, as major events happen in the residual time and investors are ‘surprised’ by ‘unexpected’ market movements. It is reasoned in [2] that the perception of concepts inherent or surrounding the investment process, whose character is not principally measurable, is best handled by the soft mathematics emerging from the theory of fuzzy sets. Thus, a fuzzy approach allows for market fluctuations well beyond the probability type of uncertainty permitted by the standard financial methods. It does not impose predefined data or market behaviour, there is only an attempt to model as much uncertainty as the environment can possibly embody, producing better estimates of the investment risk.

Considering the above arguments, we formulate a fuzzy-valued criterion at the first stage of the proposed investment appraisal method. The initial point is the price-dividend relationship, where we consider time-varying stock return. Prices are too volatile to be rational forecasts of future dividends discounted at a constant rate and empirical tests have convinced many financial economists that stock returns are time-varying rather than constant [11]. Allowing time-varying returns transforms the price-dividend relation into nonlinear. If its loglinear approximation is considered, and the resulting equation is solved forward for the log stock price, a price estimation \( \hat{p}_t \) is produced in period \( t = 0 \), (see Appendix A1 for details). Let \( T \) be the investment horizon, then a project is profitable when the estimated share price exceeds the market share price \( \hat{p}_0 > p_0 \) at \( t = 0 \).

Based on this, the fuzzy-valued criterion is formulated following a procedure in four steps. First, for each project, the parameters of linearisation \( \delta \) and \( \lambda \) are obtained and considered crisp. Second, market uncertainty is introduced to the data covering share prices, dividend yields and returns, applying a specific calibration technique to produce positive nonlinear fuzzy coefficients \( \tilde{A}, \tilde{B}, \tilde{C} \), \( 1 \leq t \leq T \). Third, the fuzzy-valued log share price \( \tilde{P} \) at \( t = 0 \) is presented as

\[
\tilde{p}_0 = \sum_{i} \lambda^{-i} \left[ (1 - \lambda)^i d_j + \tilde{A}_p \right] + \delta - \tilde{B}_r + \tilde{C}_r + \tilde{g}_0 =
\]

\[
= \tilde{A}_g + \ldots + \tilde{A}_T \tilde{g}_A - \tilde{B}_T \tilde{g}_B - \ldots - \tilde{B}_1 \tilde{g}_B + \ldots + \tilde{C}_T \tilde{g}_C + g, \quad 0 < \lambda \delta < 1.
\]

(1)

where all \( g \) functions are continuous and defined on the crisp market data employed to evaluate a project. Thus, the fuzzy log share price from [40] is transformed into a continuous multivariable fuzzy-valued function. The modification provides that the fuzzy neural network introduced in the next subsection is capable of approximating \( \tilde{P}_0 \) to an arbitrary degree of accuracy. Forth, applying the extension principle, the nonlinear membership function of the solution is obtained. For the specific formulation of \( \tilde{P}_0 \), the \( \alpha \)-cut \( \Omega_\alpha (\alpha) \) is equivalent to the interval arithmetic solution. The details of the four-step procedure are described in Appendix A1.
Solving equation (1) identifies the set of estimated log share-price values corresponding to all future log share prices, dividend yields and returns possible at some level of uncertainty, \( u \). This set is situated at the same level \( u \). Therefore, there is a critical level of uncertainty, \( u_{\text{project}} \), embodied into the market data we use to evaluate a project, and this level delimits the project’s investment risk. The risk measure \( \alpha_{\text{project}} \) below is derived from the membership level \( \mu(p_0 | \tilde{P}_0) \) of the market price \( p_0 \) to the evaluated fuzzy share price \( \tilde{P}_0 \).

\[
\alpha_{\text{project}} = \mu(p_0 | \tilde{P}_0) = \sup \{ \alpha | y_p = p_0 \}, l - u_{\text{project}} = \alpha_{\text{project}} \in [0,l].
\]

The following reasons advocate such risk definition. The lower the critical level of uncertainty at which there is a chance for the project being unprofitable, the higher the investment risk. Furthermore, \( \alpha_{\text{project}} \) is the membership level of the fuzzy log share price, below and at which the solution includes values smaller or equal to the initial log market price, and above which the project is profitable.

Finally, as the variety of market agents work within diverse risk ranges, the same project will be acceptable for some of them and too risky for others. Thus, an agent-dependent threshold will benefit the decision-maker. In conclusion, the investment criterion is described with

\[
\alpha_{\text{project}} \leq \alpha_{\text{agent}},
\]

where \( \alpha_{\text{agent}} \) communicates the admissible risk value. Consequently, for a particular agent, a project is worth investing in if the associated risk does not exceed an acceptable level.

### 2.2 Fuzzy neural network

The second stage of the method consists of building a regular fuzzy neural network, to approximate the continuous multivariable fuzzy-valued function \( \tilde{F}_0 \), and subsequently including two more layers to discriminate between risky projects. The approximating capabilities of regular fuzzy networks have been intensively studied in the last few years. It is demonstrated in [8,9] that such neural nets are not able to represent continuous fuzzy functions to an arbitrary degree of accuracy. On the other hand, [33] proves that they are universal approximator for continuous fuzzy-valued functions.

Let \( \mathbb{R} \) is the real number set, and \( F_0(\mathbb{R}) \) is the set of all fuzzy numbers on \( \mathbb{R} \). Then a continuous fuzzy function \( F_{\text{fuzzy}} \) is a mapping from and to the fuzzy number set \( F_{\text{fuzzy}} : F_0(\mathbb{R}) \rightarrow F_0(\mathbb{R}) \). A continuous fuzzy-valued function \( F_{\text{fuzzy-valued}} \), on the other hand, is a projection from the real number set to the set of fuzzy numbers, \( F_{\text{fuzzy-valued}} : \mathbb{R} \rightarrow F_0(\mathbb{R}) \). We have constructed the share price as a function of the second type, as described in the previous subsection.

Based on the theorems proved in [33] about single-variable functions, we formulate four Remarks concerning the multivariable case. Detailed mathematical explanation is given in Appendix A2. Thus a particular class of regular fuzzy networks is identified that are capable of representing multivariable continuous fuzzy-valued functions. The neural net structure includes four layers, with sigmoid transfer functions and shift terms in the first hidden layer, and identity transfer functions with no shift terms in the input, the second hidden and the output layer. In order to reduce the complexity of the network training task without weakening the approximating capabilities of the neural net, the weights and the shift terms are restricted to be real numbers with the exception of the weights to the output layer that are triangular fuzzy numbers.
Figure 2 introduces the network structure classifying investment projects according to the agent-dependent risk criterion (3). The configuration consists of a regular fuzzy neural network part and a hybrid segment. The dashed box outlines the regular module. Its input layer is fed with crisp data - log stock prices $p_i$, log returns $r_i$ and log dividend yields $dy_i$ - while the output layer produces the nonlinear fuzzy number $\tilde{P}_{RFNN}$.

$$
\tilde{P}_{RFNN}(p_1, ..., p_T, r_1, ..., r_T, dy_1, ..., dy_T) = \sum_{i=1}^{q} \sum_{j=1}^{m} V_j \left( \sum_{j=1}^{q} e_j \left( \sum_{i=1}^{T} (w_{ij}p_i + u_{ij}r_i + z_{ij}dy_i) + \theta_j \right) \right),
$$

where $w_{ij}$, $u_{ij}$, $z_{ij}$ and $e_j$ are real weights, $\theta_j$ are real shift terms, $V_j$ are triangular fuzzy weights, $\sigma$ is the sigmoid transfer function, $T$ is the investment horizon, $m$ and $q$ are the

![Figure 2: Neural network structure classifying projects with acceptable risk levels](image)

**Regular Fuzzy Neural Network is outlined in the dashed box**

**network size:** $(3 \times T) \times m \times q \times 1$

$T$ is the investment horizon

$m$ and $q$ are the number of nodes in the first and second hidden layer, respectively

crisp inputs: log stock prices $p_i$, log returns $r_i$ and log dividend yields $dy_i$

weights: real numbers $e_j, w_{ij}, u_{ij}, z_{ij}$, and triangular fuzzy numbers $V_j$

first hidden layer: sigmoid transfer functions $\sigma$ with real shifts $\theta_j$

second hidden layer: identity transfer functions $i$ with no shift terms

nonlinear fuzzy output $\tilde{P}_{RFNN}$ approximating the fuzzy log share price $\tilde{P}_o$ at $t = 0$

**Attached Hybrid Part**

$\varphi$ is a transfer function producing an approximation $\alpha_{\text{project}}^{\text{FNN}}$ of the project’s risk $\alpha_{\text{project}}$

based on the regular part output $\tilde{P}_{RFNN}$ and the market price $p_0$ at $t = 0$.

$\psi$ is a hard limit transfer function with an agent-dependent threshold $\alpha_{\text{agent}}$

output = 1 → identified acceptable-risk investments (recommendation buy)

output = 0 → rejected projects as too risky for the concerned market agent (sell)
number of nodes in the first and second hidden layer respectively. \( \tilde{P}_{\text{REFNN}} \) is an approximation of the estimated fuzzy log price \( \tilde{P}_y \), and is illustrated with its membership function in the network diagram below. The extension principle and the interval arithmetic evaluation of such fuzzy networks are equivalent and the \( \alpha \)-cuts of the output \( \tilde{P}_{\text{REFNN}} \) are computed as in Appendix A2. The hybrid part includes two nodes with transfer functions \( \varphi \) and \( \psi \), correspondingly. The first node produces the investment risk \( \alpha_{\text{project}} \) relevant to the evaluated share price \( \tilde{P}_{\text{REFNN}} \) and the market price \( p_0 \) at \( t = 0 \). Thus \( \varphi \) is described with

\[
\alpha_{\text{project}} = \varphi(\tilde{P}_{\text{REFNN}}, p_0) = \mu(p_0 | \tilde{P}_{\text{REFNN}}) = \sup(x | y_{p_0} = p_0),
\]

where \( \mu \) is the nonlinear membership function of \( \tilde{P}_{\text{REFNN}} \), \( \mu(p_0 | \tilde{P}_{\text{REFNN}}) \) is the level of membership of \( p_0 \) to \( \tilde{P}_{\text{REFNN}} \), and \( \alpha_{\text{project}} \) can take any value in the closed interval \([0,1]\).

Finally, \( \psi(\alpha_{\text{project}}, \alpha_{\text{agent}}) \) is a hard limit transfer function comparing the evaluated risk level of the investment project under consideration and the acceptable risk level \( \alpha_{\text{agent}} \) of the concerned market agent. The tolerable risk \( \alpha_{\text{agent}} \) can take any value in the open interval \((0,1)\) and act as a threshold in \( \psi \). As a result, only projects that are admissible for the particular investment agent will be recommended with a network outcome of \( 1 \), and the rest of the projects will be rejected producing \( 0 \) that indicates they are quite risky to be considered as investment opportunities. Each agent can also revise the threshold to suit his current position. Thus, it is recognised that the same project will have different bearings on the risk position of various market players and their particular circumstances at that time. Such differentiation suggests that if we consider the outcomes of \( 1 \) and \( 0 \) as recommendations to buy and sell, then the advice is also agent-dependent. This is a distinctive feature of the method in comparison with the general practice of project-related only recommendation.

### 2.3 Bidirectional incremental evolution

At the next stage of the developed technique, the regular part of the fuzzy network in Figure 2 is trained to approximate the fuzzy log share price \( \tilde{P}_y \), using a genetic algorithm and following a bidirectional incremental evolutionary strategy. The genetic algorithm is specified with its initialisation, selection and recombination operators. The initialisation step includes chromosome encoding and generating the first population. Let a triangular fuzzy weight \( \tilde{V}_i \) be presented with three real numbers \( (v_i^a, v_i^b, v_i^c) \) corresponding to its support and vertex. Then the relations \( v_i^a < v_i^b < v_i^c \), \( 1 \leq i \leq q \), are valid. Thus the neural net can be coded into a chromosome \( \chi \) of size \( M = 3mT + m + qm + 3q \), where \( M \) is equal to the number of real weights and shift terms plus three times the number of fuzzy weights in the network. The initial population \( X \) of \( s \) individuals \( \chi \) is generated simultaneously as a matrix of size \( M \times s \). The elements of \( X \) are realisations of a random variable with standard normal distribution. A block representation of \( X \) helps to concurrently sort its elements according to the inequality restrictions above. Next, a breeding subpopulation \( X_{\text{SUB}} \) is selected, which consists of the \( s_i \) best-fitted chromosomes, where \( s_i < s \). The selection is based on the fitness function \( f_\xi \).
\[ f_{\xi} = \begin{cases} 0, & \xi > \xi_A \\ (\xi - \xi_A) \xi_B, & \xi \leq \xi_A \end{cases} \]

(6)

where \( \xi_A \) and \( \xi_B \) are parameters related to the approximating precision of the coded network, and \( \xi \) represents the error of the fuzzy network for the corresponding chromosome. Consequently,

\[
\xi = \max_{a \in \{0,1,\ldots,t\}} \left( \max \left[ P_{\alpha} \left( P_{RFNN \left( \alpha \right)} \left( P_{\alpha} - P_{RFNN \left( \alpha \right)} \right) \right), P_{\alpha} \left( P_{RFNN \left( \alpha \right)} \left( P_{\alpha} - P_{RFNN \left( \alpha \right)} \right) \right) \right] \right),
\]

(7)

where \( P_{RFNN \left( \alpha \right)} \left( P_{\alpha} \right) \) and \( P_{\alpha} \left( P_{RFNN \left( \alpha \right)} \right) \) are the \( \alpha \)-cuts of the output \( P_{RFNN} \) and the share price \( P_0 \), correspondingly. Thus the best subpopulation \( X_{SUB} \) is identified, and on its basis the recombination process finally builds the new generation \( X_{NEW} \). A multipoint crossover operator is applied on \( X_{SUB} \) to produce a temporary full-size population \( X_{TEMP} \).

The number and position of crossover points is randomly chosen every generation. A real-number network parameter is recognised as a gene within the chromosome, and a triplet representing a fuzzy-number parameter is also considered as a single gene. Then crossover points are only set between genes. Randomly chosen chromosomes from \( X_{SUB} \) are combined as in Figure 3 to obtain two offspring, only one of which is included in \( X_{TEMP} \).

![Figure 3: Multipoint crossover operator](image)

---

The mutation operator concludes the recombination step, transforming the temporary population into the new generation of fuzzy-network representations. This step involves a constant mutation rate \( \tau \), where \( 0 < \tau < 0.2 \). Thus the number of mutated genes is constant \( \tau \cdot (3mT + m + qm + q) \), but their place in the chromosome is randomly chosen every generation. Modified elements are generated as realisations of a random variable with standard normal distribution. All the mutated triplet genes are concurrently sorted according to the inequality restrictions above.

This genetic algorithm is applied within a bidirectional incremental evolutionary strategy, suggested on the following grounds. A complex task is difficult to evolve, as it is not possible to directly discover a general solution. The stalling effect in direct evolution may be defeated by implementing incremental strategies, where neural networks learn complex behaviour while starting with simple functioning and gradually increasing the complexity of the task [18,21]. The problem is that the relevant subset of simple tasks as well as their sequence is not uniformly defined, and so is an incremental strategy. Consequently, it is appropriate to
identify the efficient subset of tasks and their efficient sequence \[24\]. Bidirectional incremental evolution applied here deals with the question of efficiency by first identifying the subtasks and their sequence, then evolving them separately, and finally merging the tasks gradually while following the efficient sequence. When the investment paradigm is considered, the domain of subtasks is described as follows. If \( T \) is the investment horizon, \( N_t \) is the number of periods in which projects are available, \( \Delta t_1 = [t_{01}, t_{01} + T], \ldots, \Delta t_{N_t} = [t_{0N_t}, t_{0N_t} + T] \), and \( N_2 \) is the number of companies, then there are \( N = N_1 \times N_2 \) single-company single-period projects constituting the first level of subtasks with lowest complexity. The next levels involve increasing number of single projects. They may concern the same company over several periods or the same type of companies over one or a number of periods, or may include different types of companies. According to its investment risk, there exist three modes of a single project. It can be profitable and not risky when \( \alpha_{\text{project}} = 0 \), risky for \( 0 < \alpha_{\text{project}} < 1 \), or too risky and unprofitable when \( \alpha_{\text{project}} = 1 \). There are also five types of companies. Those with investment risk 0 over all the periods are quite rare, having in mind that the market is modelled as highly uncertain. The other types include companies with continuously improving levels of risk, and companies with unstable risk levels \( 0 \leq \alpha_{\text{project}} \leq 1 \) without a particular direction. Finally, those with constantly worsening investment risk, and the ones with \( \alpha_{\text{project}} = 1 \) over all periods. Providing the training set includes companies of all types and single projects covering the three important modes of \( \alpha_{\text{project}} \), then the evolved regular fuzzy neural network will sufficiently predict the fuzzy log share price \( \tilde{P}_0 \) of new investment opportunities.

The overall idea of the evolutionary strategy is presented below, while detailed description is provided in Appendix A3. The fitness function applied throughout the evolution is \( f_\xi \) from definition (6), while the objective function is dynamic and is updated at each step of the strategy. First a random initial population of size \( s \) is generated and a fuzzy network is evolved over the full set of single projects for a probing number of generations \( N_{\text{gen}} \) using \( f_\xi \). Then, if the objective, based on the average value of the error \( \xi \) over the resulting subpopulation of \( s \) best-fitted chromosomes, is above a test limit \( \frac{1}{s} \sum_{i=1}^{s} \max (\xi (\chi_i)) > \xi_{\text{DEC1}} \), the evolution starts again with a new random initial population. Otherwise the single projects are probed with an updated objective function over the best subpopulation. If there does not exist a single project with an objective value below an updated test limit, then a new full-size population is generated by recombination of the best subpopulation and the evolution is continued for another \( N_{\text{gen}} \) iterations. Otherwise, the projects satisfying the condition are probed again opposite the second limit to identify project sets of maximum size rather then single projects with a satisfying combined objective value. Then the full training set \( n \) is partitioned into subsets \( n = \{n_{11}, \ldots, n_{1j}, \ldots, n_{1J}, n_2\} \), where \( J \) is the number of subsets of projects satisfying the second limit, and \( n_2 \) is the subset of single projects that do not satisfy the condition. This is the first decomposition level of the training set. Now \( J + 1 \) different full-size populations are generated by recombination of the same breeding subpopulation and a separate fuzzy network is evolved in \( N_{\text{gen}} \) generations for each training subset of \( n \). Then the steps up to here, with revised objective functions, are repeated for each training subset. Thus several levels of decomposition of the training set are identified, and each level is characterised with a unique partitioning into subsets and a specific number \( J \). The decomposition stage of the evolutionary strategy completes when the neural networks evolved over each training subset reaches an average value of the error over the first half of the best subpopulation below a test limit \( \xi_{\text{DECEND}} \). Then, the incremental part of the evolution starts
Figure 4: Bidirectional incremental evolution
from the highest decomposition level. Each of the training subsets at this level has an evolved best subpopulation associated with it. A full-size population is generated by recombination based on all subpopulations. The evolution continues over the whole training set existing at the highest decomposition step until a revised objective function gets below $\xi_{\text{INC}}$. This is the first incremental step. Then the second highest decomposition level is considered. The training set existing at this level includes by definition the whole training set from the highest decomposition step and some further subsets of projects. Again one full-size population is generated by recombination of the breeding subpopulations associated with each training subset here. The subset equivalent to the highest decomposition set of projects is presented with the best subpopulation evolved at the first incremental level. The objective limit is again $\xi_{\text{INC}}$ but the objective function includes more projects. This is the second incremental level. The procedure continues by analogy until the first decomposition step is reached, where the partitioning was applied over the initial full training set. It will be the final incremental level. A neural network is evolved until the error of one chromosome only, rather than over a subpopulation of chromosomes, but including all the projects, falls below $\xi_{\text{INCEND}}$ where $\xi_{\text{INCEND}} < \xi_{\text{INC}}$. This best chromosome $\lambda_{\text{INCEND}}^{\text{best}}$ represents the completely evolved regular fuzzy neural network.

Figure 4 above represents the evolutionary strategy. For simplicity, the decomposition part is slightly generalised in the diagram omitting the test limits $\xi_{\text{DEC}}$, (see Appendix A3). The bidirectional incremental algorithm implements a dynamic objective function. Its fitness function $f_\xi$, on the other hand, will be the same over all steps, and equal to the one used in direct evolution, for an easy comparison of the results.

2.4 Conclusions

In conclusion, the advantages of the developed method cover the following aspects. The investment classifier is based on a fuzzy-valued criterion that better handles uncertain market data than crisp criteria [11]. Whatever reason one has for modifying the classic result [14,45], the allowances provided by the fuzzy-valued criterion will cover these specific circumstances and will include the modified value, as well as other possible values, increasing the flexibility of the involved calculations. Moreover, a measure for the risk associated with each project is suggested, thus providing grounds for an alternative investment ranking. This result emphasises the method as more informative to the decision-maker, in comparison with other fuzzy approaches [5,6,10,30]. Next, a fuzzy neural network structure is identified capable of approximating the multivariable criteria. The configuration is derived by developing further the result for single-variable fuzzy-valued functions from [33]. Then, a genetic algorithm is suggested for optimising the parameters of the network. Starting with a procedure from [8] for a simple network configuration, and recursively testing it on and modifying it for the more complex structure from Figure 2, the algorithm is developed to demonstrate increased speed and efficiency. Finally, a bidirectional incremental evolutionary strategy is outlined that is particularly effective in training fuzzy networks with a larger number of nodes and layers. Direct evolution fails to solve the task. We can compare our empirical results (see subsection 4.1) with other applications of genetic algorithms to optimising fuzzy network parameters. In [8] quite a simpler structure is trained, with direct evolution only, to approximate an elementary single-variable fuzzy function.

Optimising the classifier is time-consuming and the complexity of the task increases with expanding the investment horizon. Still, it is not a disadvantage of the method, as the model is only incorporated into the intelligent system after training, and then instantly provides support to the decision-maker. Moreover, several soft classifiers can be included into the multimodel based system (see Section 3), each trained to give recommendation over different horizon-length. Another beneficial feature is the agent-dependent recommendation. Rather than providing an abstract advice, the model relates every investment project to the possible circumstances or preferences of each major type of market agents.
3 Multimodel knowledge representation

3.1 Model selection

The financial techniques to be incorporated into the multimodel domain are chosen with the purpose to give the intelligent system the ability to support a decision-maker through various investment inquiries. Starting with a share-price estimation based on the log-linearisation of the price-dividend relation which is solved forward over the project length [11]. This will provide a preliminary project evaluation, handling time-varying returns. If the estimated figure encouragingly diverges positively from the market price, further investigation can be undertaken. A slightly more flexible criterion will be employed to tackle the timing of the initial outlay. It is shown in [14] that if the value of the project at the time it is evaluated is stochastically fluctuating and depending on a random start time, then there exists a trade-off between a larger versus a later net benefit. Thus the optimal timing of buying the shares will be identified, taking into account investment irreversibility. The best start period equals the time at which the benefit reaches a threshold. In comparison, if the standard criterion were used, then the agent would invest sooner but rely on larger rates to get the same profit.

Up until now the decision-maker has an idea which projects could be beneficial and is intent to get more reliable evaluation. Although with increasing flexibility in comparison with standard investment criteria, all models above are crisp and therefore working under relatively narrow presumptions about the underlying distributions and market behaviour. Next some fuzzy criteria will be introduced into the multimodel domain that are effective under a high level and diverse forms of uncertainty, i.e. market imperfections, absence of complete or precise data, and presence of human involvement in price formation. In [5] an evaluation technique is suggested that can be applied here in the following aspect. Using the standard arithmetic of fuzzy numbers, and modelling uncertain amounts in the project cashflow as well as the estimated rate with nonlinear trapezium membership functions, two different formulae are applied when an amount is positive or negative. For example, if the project involves buying shares in the beginning and selling them in the end, the first formula will be only relevant. If the project prescribes various optimal points in time for buying and selling different shares, then the two formulae will be involved. Other fuzzy models extend even further the flexibility of the evaluation technique. In [10], the option of associating a different rate to each amount is offered. The main contribution in [30] consists in fuzzifying the project duration over the relevant time periods. Usually fuzzified duration is presented as a discrete fuzzy set with a membership function defined by a collection of positive integers each corresponding to the end of a time period according to the accepted time division, e.g. months, weeks, days. Alternatively in [30], the duration is a real fuzzy number allowing for the project to finish at any moment, not only at the end of a time period, i.e. in the middle of the month or throughout the week. Finally, among this group of fuzzy criteria mainly based on the stock-price evaluation, the technique from [36] finds its place. There, a hybrid network is developed and a fuzzy membership array is embedded into a neural network. Although the approach is applied to predicting a stock index, it effectively can be used in single-asset or portfolio price evaluation. The out-of-sample forecasts of the hybrid net up to 24 months ahead without updating the model are compared with these of several crisp regressions and the neuro-fuzzy model is found to perform substantially better.

The next group of models will present the decision-maker with still reliable but further progressively informative solutions, comparing with the above techniques. Questions considered include fuzzy ranking, investment risk evaluation, measuring project robustness towards market uncertainty, and agent-dependent classification of attractive projects. Using a procedure suggested in [5], the projects can be ranked according to the evaluated membership functions for the fuzzy criterion. Further in [23,43], a risk measure is proposed and an alternative risk-based ranking technique is described. The solution procedure there involves a multiple interval analysis, which enables investors to consider a project and take decisions based on varying levels of uncertainty. Increasing the range of uncertainty modelled into the fuzzy data, one can determine the robustness of the investment risk associated with each
project, as in [40,42]. Thus the ranking technique is refined and based on both the projects’ risk and robustness evaluation. Analogous risk measure is used in [41] and in Section 2 here, when building the soft classifier. Several such fuzzy networks can be included into the multimodel intelligent system, each network trained over projects with different duration. All the classifiers will be trained prior to entering the system, and ready to provide agent-dependent recommendation. The classifying model can go one step further by incorporating into the neural network structure the robustness measure as well. Thus there will be agent-dependent thresholds for both investment risk and project robustness, following the intuition that projects with a small and a highly robust investment risk are preferable. Consequently, the fuzzy criterion evolves into a considerably informative method.

### Table 1: Selected models for the multiperspective intelligent system

<table>
<thead>
<tr>
<th>models</th>
<th>characteristics</th>
<th>major</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimal rules for irreversible investment [14]</td>
<td>Helps identify the optimal time of buying the assets, considering investment irreversibility and a benefit threshold. Increased flexibility in handling market data. A crisp criterion.</td>
<td>reliable evaluation, fuzzy or soft techniques</td>
</tr>
<tr>
<td>fuzzy present value [5]</td>
<td>Increased reliability in market evaluation. A fuzzy criterion.</td>
<td>informative ranking and classification, fuzzy and soft techniques</td>
</tr>
<tr>
<td>generalised fuzzy criterion [10]</td>
<td>Allows a different rate to be associated with each amount in the project cashflow.</td>
<td></td>
</tr>
<tr>
<td>generalised fuzzy criterion [30]</td>
<td>Provides alternative fuzzification of the project duration.</td>
<td></td>
</tr>
<tr>
<td>neuro-fuzzy price evaluation [36]</td>
<td>Exploits financial market inefficiencies and extracts nonlinear patterns that even sophisticated econometric models can not cope with. Combines fuzzy logic with a neural network.</td>
<td></td>
</tr>
<tr>
<td>fuzzy ranking [5]</td>
<td>Suggests a technique for ranking fuzzy numbers and considers the membership functions of projects’ fuzzy present value.</td>
<td></td>
</tr>
<tr>
<td>investment risk evaluation [23,43]</td>
<td>Applies $\alpha$-cut arithmetic in fuzzy price evaluation, instead of standard fuzzy arithmetic. Derives an investment risk measure and ranks the projects accordingly.</td>
<td></td>
</tr>
<tr>
<td>project robustness evaluation [40,42]</td>
<td>Uses two calibration procedures when fuzzifying the market data, thus modelling an increased range of uncertainty. Identifies which projects are more robust based on the change in the investment risk measure under varying data calibration. Suggests an alternative ranking using both the risk and robustness measures.</td>
<td></td>
</tr>
<tr>
<td>classifying low-risk projects [41, manuscript]</td>
<td>Develops a genetic algorithm and a bidirectional evolutionary strategy to train a fuzzy neural network to classify stock projects according to the investment risk measure and an agent-dependent threshold.</td>
<td></td>
</tr>
<tr>
<td>classifying highly-robust projects [future research]</td>
<td>Will incorporate the measure of the project robustness into the soft-classifying model, following the heuristic that low-risk and highly-robust projects are preferable.</td>
<td>portfolio selection and stock trading, soft or hybrid techniques</td>
</tr>
<tr>
<td>portfolio selection [46]</td>
<td>Investment expert knowledge is reflected into the stock return possibility or fuzzy probability distributions. Selection is based on minimising corresponding parameters in the portfolio distribution.</td>
<td></td>
</tr>
<tr>
<td>artificial stock market [47]</td>
<td>Suggests that the agents have the ability to compress numerous crisp trading rules and conditions into a few fuzzy notions and analyse them using fuzzy logic.</td>
<td></td>
</tr>
</tbody>
</table>
The final set of models suggests portfolio-selection and stock-trading strategies. In [46], two portfolio selection models are proposed where investment expert knowledge is reflected into the fuzzy probability or possibility distributions of stock returns. Distributions are obtained depending on stock possibility grades offered by experts. The aim is to minimise the variance of the fuzzy probability distribution or to minimise the spread of the possibility distribution of the return on the portfolio. Further in [35], a dynamic portfolio selection is outlined. A fuzzy control model is used to manage a portfolio in a fuzzy dynamic environment in the presence of financial constraints. One riskless and n risky assets are employed, modelling the factor imprecision with fuzzy sets. A fuzzy linear tracking problem is solved using the Kalman filter. Next, the effort in [32] is to develop a stock-trading system, integrating neural nets and fuzzy Delphi methods. The result shows that without and under transaction costs the integrated model outperforms the single neural network. Although the genetic-fuzzy model in [22] is applied to currency trading, no further adjustment is necessary to use it in stock trading, as the techniques has all the features to cope with the problem efficiently. The approach provides an automated method for discovering trading knowledge. Finally, an architecture for an artificial stock market is suggested in [12] that is agent-based and includes evolving successful traders. In [47] another artificial stock market model is introduced. It is proved there that similar results can be obtained, either considering traders capable of handling a large number of rules with numerous conditions, or allowing the reasoning process to be severely limited. In the latter case the results are even improved. This case is also closer to reality, and agents have the ability to compress information into a few fuzzy notions that they can process and analyse with fuzzy logic. All selected models are included in Table 1. The system is still open for more models, but each choice should be justified.

3.2 Building the multiperspect iverse model space

A multimodel investment domain will help the decision-maker with the process of multiperspective analysis and multilevel reasoning. Being exposed to a variety of possible interpretations, will effortlessly enriches his vocabulary of relationships and enlarges his search space for hypothesis construction. He will be able to look at the investment problem from various angles thus improving his ability to understand and solve specific queries.

Building the multimodel space starts with choosing the modelling dimensions [26,27,28,29]. This is the set of properties that represents fundamental characteristics associated with the context or framing of the investment problem. Each dimension - i.e. scope, resolution, rigidity - denotes a different context, which directs the multiperspect iverse reasoning. For example, scope is concerned with the extent of the problem being modelled. We may simply want to evaluate a project. Broadening the scope, an optimal portfolio can be identified. Finally, the functioning of an artificial stock market may be under question. Further, resolution dictates how much detail is included in a model. As an illustration, the factor analysis when managing a dynamic portfolio may be based on a few major determinants only. Increasing the resolution, some minor factors will be included, or the major determinants can be decomposed into several smaller ones and their effect studied separately. When rigidity is considered, we focus on the ability of the model to handle uncertain data and real-market situations. In this sense, the crisp models are most rigid, and progressing along the fuzzy and soft techniques, the hybrid approaches are most flexible.

The multimodel space can be visualised as a geometrical shape with several axes or dimensions. It is easier to think of a cubical form and Figure 5 presents the three dimensions described above: scope, resolution and rigidity. In practice, there are more perspectives and each component cube explodes into another cubical structure. This time axes are different, e.g. generality, precision and accuracy, and provide support for considering additional aspects of context. By choosing a value for every dimension, one can navigate to a specific model in the multiple structure. Neighbouring models differ only slightly, i.e. two soft classifiers recommending projects over different investment horizon. Distantly related techniques, on the other hand, are quite unlike.
4 Empirical results and conclusions

4.1 Empirical results

The empirical data involve three UK companies - Goodwin, Dixons Group and Marks & Spencer - over the period from June 1998 to December 1999. They are chosen from a database of 35 firms to represent three major types of company behaviour. In Figure 6, the fuzzy log-dividend time-trajectory related to each firm illustrates modelled market uncertainty. Furthermore, a six-month investment horizon is selected, producing three projects per company and nine in total. This set of projects is divided into three subsets, used correspondingly for training, testing and predicting with the fuzzy neural network. Table 2 introduces the division of the data set and the accepted project notation. Table 3 presents the risk level for each project evaluated with the fuzzy criteria.
In the second column of Table 3, Goodwin demonstrates a continuously improving risk measure, reaching $\alpha_{\text{project}} = 0$ for project 7. In the third column, Dixons Group indicates oscillating levels $0 \leq \alpha_{\text{project}} \leq 1$, without a particular direction over the related projects. Finally, Marks&Spencer exhibits the highest risk level $\alpha_{\text{project}} = 1$ all over. Thus, it is provided that three major types of companies are represented. It is also guaranteed that the training set includes single projects covering the three important risk modes: $\alpha_{\text{project}} = 0$ (project 4), the open interval $0 < \alpha_{\text{project}} < 1$ (project 2), and $\alpha_{\text{project}} = 1$ (projects 1, 3, 5, and 6).
Figure 7 illustrates how the risk measure for a project is derived based on the estimated fuzzy-valued log share price \( \tilde{P}_0 \) and the market price \( p_0 \).

Next, the genetic algorithm and the bidirectional incremental evolutionary strategy from subsection 2.3 are applied to train the regular fuzzy network module to approximate the fuzzy-valued log share price over the training set, project 1 to project 6. Each project comprises six-month data on log share prices, returns and dividend yields. Consequently, the number of nodes in the input layer of the network is \( 3T = 3 \times 6 = 18 \). Then experimenting with several network structures and considering the trade-off between a simpler configuration and evolution convergence, the values \( m = 5 \) and \( q = 3 \) are selected, for the number of nodes in the first and second hidden layer correspondingly. Detailed results for each step of the evolutionary strategy from Appendix A3 are provided in Appendix A4. Here, only the major conclusions are discussed. The strategy involves a decomposition and an incremental part. During the decomposition part, the problem is accordingly divided into subtasks of decreasing complexity by partitioning the training set of projects at several levels. Figure 8 presents the identified partitioning. Then the subtasks are merged incrementally in reverse direction.

The number of levels and the partitioning at each level is unique to every simulation of the bidirectional strategy. Consequently, it is not possible to average the performance of the strategy over several simulations, as the fitness function trajectory over the generations will go through different training subsets addressed at different times. On the other hand, direct evolution is easily averaged. For more representative comparison, Figure 9 presents a simulation of bidirectional incremental evolution against the averaged result from five direct-evolution simulations. Maximum fitness per generation is presented for each strategy. The bidirectional strategy evolves a fully functional fuzzy network in 148,243 generations. Direct evolution reaches only 46.33% maximum fitness in 500,000 generations. Thus, the empirical results prove decisively the efficiency of the developed evolutionary strategy.
Once the regular fuzzy network module is optimised over the training set, it is applied to the test set, project 7 and project 8 (see Table 2). Thus, the out-of-sample performance of the module is examined in approximating the fuzzy-valued log share price. The error $\xi$ from definition (7) is applied to measure the divergence of $\hat{P}_{\text{RFNN}}$ from $\hat{P}_0$, and consequently the performance of the module. The test results, provided in Table 4, correspond to quite satisfactory approximating capabilities. Consequently, we can use now the hybrid network over the predicting subset, project 9, to estimate the investment risk. In Table 5, the predicted risk value approaches the true value from Figure 7.

**Figure 5: Performance of bidirectional incremental evolution and direct evolution in maximum fitness per generation**

*Black line:* bidirectional incremental evolution advances through several decomposition and incremental tasks and solves the general problem in 148,243 generations.

*Lighter line:* direct evolution makes some initial progress and then stalls.

Once the regular fuzzy network module is optimised over the training set, it is applied to the test set, project 7 and project 8 (see Table 2). Thus, the out-of-sample performance of the module is examined in approximating the fuzzy-valued log share price. The error $\xi$ from definition (7) is applied to measure the divergence of $\hat{P}_{\text{RFNN}}$ from $\hat{P}_0$, and consequently the performance of the module. The test results, provided in Table 4, correspond to quite satisfactory approximating capabilities. Consequently, we can use now the hybrid network over the predicting subset, project 9, to estimate the investment risk. In Table 5, the predicted risk value approaches the true value from Figure 7.

<table>
<thead>
<tr>
<th>period / company</th>
<th>Goodwin project: $\text{RFNN error } \xi$</th>
<th>Marks&amp;Spencer project: $\text{RFNN error } \xi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>July-December, 1999</td>
<td>Project 7: 0.0091&lt;0.01</td>
<td>Project 8: 0.0024&lt;0.01</td>
</tr>
</tbody>
</table>

**Table 5: Predicting investment risk with the hybrid fuzzy network**

<table>
<thead>
<tr>
<th>period / company</th>
<th>Dixon Group project: $\alpha_{\text{project}}$</th>
<th>Dixon Group project: $\text{true risk } \alpha_{\text{project}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>July-December, 1999</td>
<td>Project 9: 0.88</td>
<td>Project 9: 0.858</td>
</tr>
</tbody>
</table>

Finally, if the agent-dependent threshold is at 0.9 risk level, than the project will be accepted. On the other hand, if the limit is at 0.8, then this asset will be rejected over the related investment horizon.
4.2 Conclusions

A soft classifier has been constructed that recommends projects according to their evaluated risk and an agent-dependent threshold for the acceptable risk levels. The model is developed in three stages. First, an alternative investment criterion is formulated, including a fuzzy-valued share price estimate. The approach is more reliable than analogous crisp techniques, as it is derived allowing market fluctuations well beyond the probability type of uncertainty permitted by standard financial methods. The criterion is also more informative in comparison with other fuzzy approaches, as it further suggests an investment risk measure. The second stage in the soft model development involves identifying a fuzzy network structure capable of approximating the criterion formulated at stage one. Building on previous studies of fuzzy network approximating qualities concerning simple fuzzy functions or single-variable fuzzy-valued functions, we deduce a structure adequate for the multivariable criterion. The final stage focuses on training the network. A genetic training algorithm is developed that demonstrates high speed and efficiency. Then, an effective bidirectional evolutionary strategy is elaborated, as direct evolution fails to rich a solution to the complex problem of optimising the weights and shift terms in the fuzzy network over a set of investment projects. The strategy involves a decomposition and an incremental part. The integral problem is first divided into subtasks of decreasing complexity by partitioning accordingly the training set of projects. Then the subtasks are merged incrementally to optimise the integral solution.

We next explain how the classifier, once developed and trained, can be used in problem solving. It will be incorporated into a multimodel domain knowledge structure that is to be the expert module in an intelligent system. Building the system includes three stages. First, relevant models are selected to give the environment the ability to generate answers to a variety of queries with increasing complexity. From single-project evaluation, through ranking and agent-dependent risk-classification of available stocks, up to trading recommendation and market simulation. In addition to addressing various though related investment questions, the models have different mathematical grounds. From crisp, through fuzzy and soft computing techniques, up to artificial intelligence and hybrid methods. Thus the same problem can be solved using more or less rigid mathematical approaches. Furthermore, in each model the number of parameters may be reduced or increased according to the nature of the inquiry. The second stage in building the multiple expert structure of the system concerns the identification of major contextual perspectives. We have recognised the dimensions of scope, resolution, rigidity, generality, precision, and accuracy. All models are arranged along these axes, creating a cubical structure where each three-dimensional component explodes into a secondary cubical structure, thus accommodating all six dimensions. So constructed, the multimodel domain facilitates a multiperspective analysis of a single problem. It further provides a powerful environment, in comparison with mono-model based intelligent systems, for solving a variety of problems rather than one particular query. Finally, investment strategies can be tested that involve an ordered set of queries and require a decision at each step. Navigating through the model space, the optimal solution trajectory will be drawn. The third stage in developing the environment is the software design of an intelligent-agent architecture for the system, where the multimodel expert module is incorporated. This step is not addressed here, but the work on it is in progress. In [25], ideas on object-oriented patterns for model-based reasoning are presented that are particularly suitable for designing the multimodel domain, and in [4] an intelligent-agent architecture is described for a system based on such expert module. One further quality of the multimodel based system is the flexibility it can provide in diagnosing user behaviour in order to provide a better decision support, and this is a topic for future research.

Acknowledgements

This work was partly supported by the Nuffield Foundation and by the Engineering and Physical Sciences Research Council Grant GR/R51346/01.
References

Appendix

A1. Formulating the fuzzy valued criterion

**Preliminary:** Let us consider the loglinear approximation of the price-dividend relation under time-varying returns,

\[ r_{it} = \delta + p_{it} + (1-\lambda)dy_{it} - p_t, \]

where \( p_{it}, r_{it}, \) and \( dy_{it} \) stand for the log share price, log return and log dividend yield, correspondingly. If the equation is solved forward for the log stock price, the following estimation is produced in period \( t=0 \),

\[ \hat{p}_0 = \frac{1}{T} \sum_{t=1}^{T} \left( (1-\lambda)(dy_t + p_t) + \delta - r_t \right) + (1-e^{\delta - \lambda}) p_t, \]

\[ 1 < e^{\delta - \lambda} < 2, 0 < e^{\delta - \lambda} < 1, p_t > 0, 0 < r_t < \ln(2). dy_t < 0, \]

where the parameters of linearisation, \( \delta \) and \( \lambda \), are evaluated with continuous functions of \( \{dy_t,...,dy_T\} \). If the investment horizon is \( T \), then a project is profitable at \( t=0 \) when the estimated share price exceeds the market share price, \( \hat{p}_0 > p_0 \). Based on this, the fuzzy valued criterion is formulated following a procedure in four steps.

**Step 1:** For each project, parameters \( \delta \) and \( \lambda \) are considered crisp and obtained from

\[ \lambda = \left[ 1 + e^{\frac{\delta - \lambda}{\epsilon}} \right]^{-1} = f_s(dy_1,...,dy_T), 0 < \lambda < 1, \]

\[ \delta = (1-\lambda)ln(1-\lambda) - \lambda ln(\lambda) = f_s(dy_1,...,dy_T), 0 < \delta < 1. \]

**Step 2:** Market uncertainty is introduced applying initially a calibration technique based on the 95% confidence interval of Student’s distribution \( t \) for the market level data. Thus, triangular fuzzy numbers are produced corresponding to fatter-tail possibility distributions. Then, the level-log data transformation causes nonlinear rather than triangular membership functions for the fuzzy log share prices \( \tilde{P}_t \), rates of return \( \tilde{R}_t \) and dividend yields \( \tilde{D}_t \), \( 1 \leq t \leq T, 1 \leq i \leq N \). Here \( T \) is the investment horizon and \( N \) is the number of projects. Finally, positive nonlinear fuzzy coefficients \( \tilde{A}_i, \tilde{B}_i, \tilde{C}_i \) are obtained from

\[ \tilde{A}_i = \frac{1}{N} \sum_{t=1}^{T} \tilde{P}_i \tilde{R}_i = \frac{1}{N} \sum_{t=1}^{T} \tilde{P}_i \tilde{C}_i = \frac{1}{N} \sum_{t=1}^{T} \tilde{D}_i, 1 \leq t \leq T, 1 \leq i \leq N. \]

The initial calibration is slightly modified, assuming

\[ \tilde{P}_0 = \tilde{A}_0 \tilde{p}_0, \tilde{R}_i = \tilde{B}_i r_i, \tilde{D}_i = \tilde{C}_i d_i, 1 \leq t \leq T, 1 \leq i \leq N. \]

**Step 3:** The fuzzy log share price \( \tilde{P} \) at \( t_0 = 0 \) is presented as

\[ \tilde{P}_0 = \sum_{t=1}^{T} \left[ (1-\lambda)(\tilde{C}_t d_t + \tilde{A}_t p_t) + \delta - \tilde{B}_t r_t + \sum_{t=1}^{T} (1-\lambda) A_{t,s,ui} + \cdots - \tilde{B}_{t,s,ui} + \cdots - \tilde{B}_{t,s,ui} + \cdots - \tilde{C}_{t,s,ui} + g. \right. \]

where \( g_{s,ui} = g_{s,ui}(x), g_{s,ui} = g_{s,ui}(x), g = g(x), x = [p_1,...,p_{t-1},r_1,...,r_{t-1},d_1,...,d_{t-1}], 1 \leq t \leq T, p_t > 0, 0 < r_t < \ln(2), d_t < 0, \]

are continuous functions defined on the market data employed to evaluate a project. Thus, the fuzzy log share is described as a continuous multivariable fuzzy-valued function.

**Step 4:** Applying the extension principle, the nonlinear membership function of the solution is defined by

\[ \mu(x, \tilde{P}_t) = \sup \{ \alpha | P_t \in \Omega_t(\alpha) \} \]

where

\[ \Omega_t(\alpha) = \{ x_1(g_1(x)) + \cdots + c_t g_t(x), x = \{ p_1,...,p_t,r_1,...,r_t,d_1,...,d_T \}, 1 \leq t \leq T, p_t > 0, 0 < r_t < \ln(2), d_t < 0, \]

and for the specific formulation of \( \tilde{P}_t, \) the \( \alpha \)-cut \( \Omega_t(\alpha) \) is equivalent to the interval arithmetic solution

\[ \Omega_t(\alpha) = \left[ P_t^{\alpha} \left| P_t^{\alpha} \right| \right]. \]
from equation (1), and apply fitness function \( f_0 \) from definition (6).

V: Keep the result of the evolution - the breeding subpopulation \( X^{(b)} \) - where \( s_b \) is the number of breeding chromosomes and \( s_b < s \).

VI: Apply the objective function \( \frac{1}{s_b} \sum_{i=1}^{s_b} \max_{s_i} \left[ \xi(x) - \xi^{(c)} \right] < 0 \). If the average error over the breeding subpopulation \( X^{(b)} \) in the complete training set is above the limit \( \xi^{(c)} \), then go to step III.

VII: Apply the objective function \( \frac{1}{s_b} \sum_{i=1}^{s_b} \xi(x) - \xi^{(c)} < 0 \). If there do not exist single projects satisfying the condition, then generate a full-size population \( H^{(b)}(X^{(b)}) \) by recombination of \( X^{(b)} \). Go to step IV.
VIII: Group the single projects satisfying the condition in step VII into $n^{(k)}_{i}$ subsets of projects, $n^{(k)}_{i} = \{n^{(k)}_{i,1},...,n^{(k)}_{i,s}\}$, where each $n^{(k)}_{i,s}$ is of maximum size, subject to the condition $\frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} < \xi_{\text{DECEND}}^{(k)}$.

IX: Partition the training set $n^{(h)}_{i}$ into $n^{(h)} = \{n^{(h)}_{i,1},...,n^{(h)}_{i,s},n^{(h)}_{i,s+1}\}$, where $J_{s}$ is the number of subsets satisfying the objective function from step VIII. $J_{s}$ is specific for the level of decomposition $k$. The subset of single projects not satisfying the condition in step VIII is $n^{(s)}_{i} = n^{(h)} / n^{(h)}$. In the extreme, it can be $n^{(s)}_{i} = \emptyset$ or $n^{(s)}_{i} = n^{(h)}$.

X: Generate different full-size populations $IP_{ij}\left(\chi^{(i)}_{1}\right)$, ..., $IP_{ij}\left(\chi^{(i)}_{s}\right)$ and $IP_{i}^{\text{gen}}\left(\chi^{(h)}_{i}\right)$ by recombination of the same breeding subpopulation $\chi^{(h)}_{i}$.

XI: Evolve a separate neural network in $N_{\text{gen}}$ generations for each training subset of $n^{(h)}_{i}$. Keep the evolution results – the breeding subpopulations $\chi^{(h)}_{i,1},...,\chi^{(h)}_{i,s},\chi^{(h)}_{i,s+1}$.

XII: If $\frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} < \xi_{\text{DECEND}}^{(k)}$, then generate a full-size population $IP_{ij}\left(\chi^{(i)}_{s}\right)$. Else, go to step XIII.

XIII: Evolve a fuzzy network, using the training set $n^{(h)}_{i}$, until the average network error over the first half of the breeding subpopulation $\chi^{(h)}_{i,1}$ is less than $\xi_{\text{DECEND}}^{(h)}$. This means that the objective function is $\frac{2}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} < \xi_{\text{DECEND}}^{(k)}$. Keep the result of the evolution – half of the breeding subpopulation $\chi^{(h)}_{i,1}$.

XIV: If $n^{(h)}_{i}$ consists of a single project and $\xi_{\text{DECEND}}^{(k)} < \frac{1}{s} \sum_{s} \xi_{\text{DECEND}}^{(h)}$, then generate a new full-size population $IP_{ij}\left(\chi^{(h)}_{i}\right)$. Else, go to step XV.

XV: Evolve a regular fuzzy neural network for $N_{\text{gen}}$ generations, using the training set $n^{(h)}_{i}$. Keep the result of the evolution–the breeding subpopulation $\chi^{(h)}_{i}$. Go to step XI.

XVI: If $\xi_{\text{DECEND}}^{(k)} < \frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\}$ and $n^{(h)}_{i}$ consists of a subset of projects, then generate a full-size population $IP_{ij}\left(\chi^{(i)}_{s}\right)$. Consider the subset $n^{(h)}_{i}$ as a complete training set $n^{(h+1)}_{i} = n^{(h)}_{i}$ and increase $k = k + 1$. Go to step IV.

XVII: If $\xi_{\text{DECEND}}^{(k)} < \frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\}$ and $n^{(h)}_{i}$ consists of a subset of projects, then generate a new full-size population $IP_{ij}\left(\chi^{(h)}_{i}\right)$. Consider the subset $n^{(h)}_{i}$ as a complete training set and increase decomposition level $k = k + 1$. Go to step IV.

XVIII: Apply objective function $\frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} - \xi_{\text{DECEND}}^{(k)} < 0$. If it is satisfied, then generate a full-size population $IP_{ij}\left(\chi^{(h)}_{i}\right)$, else generate a new initial population $IP_{ij}\left(\chi^{(h)}_{i}\right)$. Consider $n^{(h)}_{i}$ as a complete training set and increment $k = k + 1$. Go to step IV.

XIX: If $n^{(h)}_{i}$ consists of a single project, then evolve a fuzzy network until the objective $\frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} < \xi_{\text{DECEND}}^{(k)}$ is met and keep the result $\chi^{(h)}_{i,1}$ END. Else consider $n^{(h)}_{i}$ as a complete training set, increment $k = k + 1$ and go to step IV.

XX: Set $k$ at the highest level of partition $k = K = \max(k)$. Consider the training set $n^{(K)} = \{n^{(K)}_{i,1},...,n^{(K)}_{i,s}\}$ and generate a full-size population $IP_{ij}^{\text{gen}}\left(\chi^{(K)}_{i,1}\text{END} ..., \chi^{(K)}_{i,s}\text{END} \right)$.

XXI: Evolve a fuzzy network, using the training set $n^{(i)}_{i}$, until $\frac{1}{s} \sum_{s} \max \{\xi^{(k)}_{i,s}\} - \xi_{\text{DECEND}}^{(k)} < 0$. Keep the result $\chi^{(k)}_{i,1}$ END.

XXII: Decrease $k = k - 1$, which is equivalent to increasing the incremental level. Consider the training set $n^{(k)} = \{n^{(k)}_{i,1},...,n^{(k)}_{i,s},n^{(k+1)}_{i}\}$ and generate a full-size population $IP_{ij}^{\text{gen}}\left(\chi^{(K)}_{i,1}\text{END} ..., \chi^{(K)}_{i,K-1}\text{END} \right)$. If $k > 1$, go to step XXI.

XXIII: Evolve a regular fuzzy neural network, using the training set $n^{(i)}_{i}$, until the error of the best chromosome gets below $\xi_{\text{DECEND}}^{(i)}$, $\max \{\xi^{(i)}_{\text{DECEND}}\} - \xi_{\text{DECEND}}^{(i)} < 0$. Keep the best chromosome $\xi_{\text{DECEND}}^{(i)}$. It represents the completely evolved network.
A4: Bidirectional incremental evolution: empirical results

During evolution apply fitness function $f_i = \begin{cases} 1, & \xi > 0.1515 \\ 0.1515 - \frac{\xi}{\xi}, & 0 \leq \xi \leq 0.1515 \end{cases}$

Decomposition part

Start with the training set $n^{(i)} = \{\text{project1, project2, project3, project4, project5, project6}\}$. Generate a random initial population $IP$ of size $s = 100$, and evolve a fuzzy neural network for $N_{\text{pop}} = 10,000$ generations. The result for the first objective function is $\frac{1}{s_i} \sum_{i=1}^{s_i} \max_s(\xi_j(\chi_j)) = 0.1276 < \xi^{(i)}_{\text{dec1}} = 0.1525$. Consequently, keep the evolved breeding subpopulation $X^{(i)}_{s_i}$ of size $s_i = 30$.

Check which single projects satisfy the second objective. $\frac{1}{s_i} \sum_{i=1}^{s_i} \xi^{(i)}_{\text{dec2}} = 0.0525$.

```
objective / project | project 1 | project 2 | project 3 | project 4 | project 5 | project 6 | project 5 & 6
-------------------|---------|---------|---------|---------|---------|---------|----------
\frac{1}{s_i} \sum_{i=1}^{s_i} \xi_j(\chi_j) | 0.1255  | 0.1276  | 0.1114  | 0.1206  | 0.0972  | 0.0703  |
```

All six projects produce larger numbers then the parameter $\xi^{(i)}_{\text{dec2}} = 0.0525$. Consequently, no partition is possible yet.

Generate a full-size population $IP_{\text{full}}(X^{(i)}_{s_i})$ by recombination of best subpopulation $X^{(i)}_{s_i}$ and continue training with $n^{(i)}$ for another $N_{\text{pop}} = 10,000$ generations.

```
objective / project | project 1 | project 2 | project 3 | project 4 | project 5 | project 6 | projects 1 & 5
-------------------|---------|---------|---------|---------|---------|---------|----------
\frac{1}{s_i} \sum_{i=1}^{s_i} \xi_j(\chi_j) | 0.0448  | 0.0739  | 0.0691  | 0.0633  | 0.0472  | 0.0531  | 0.0557
```

Now the value for project 1 is less then the second parameter $\xi^{(i)}_{\text{dec2}} = 0.0525$ and so is the value for project 5, while both projects together produce a larger number then $0.0525$. Consequently, the following first-level decomposition is identified.

**Level 1:** $n^{(i)} = \{n^{(i)}_{\text{1}}, n^{(i)}_{\text{2}}, n^{(i)}_{\text{3}}\} = \{\text{project 1}\}, n^{(i)}_{\text{2}} = \{\text{project 5}\}$, $n^{(i)}_{\text{3}} = \{\text{project 2, project 3, project 4, project 6}\}$

Keep the resultant breeding subpopulation in $X^{(i)}_{s_i}$.

Generate different full-size populations $IP_{\text{full}}(X^{(i)}_{s_i})$, $IP_{\text{full}}(X^{(i)}_{s_i})$, and $IP_{\text{full}}(X^{(i)}_{s_i})$ by recombination of the same breeding subpopulation $X^{(i)}_{s_i}$, and evolve a separate fuzzy network in $N_{\text{pop}} = 10,000$ generations for each training subset of $n^{(i)}$.

```
project 1 | project 5 | project 2,3,4,6
-------------------|---------|---------|-----------
\frac{1}{s_i} \sum_{i=1}^{s_i} \xi_j(\chi_j) | 0.0646  | 0.0125  | 0.0066  \
\frac{1}{s_i} \sum_{i=1}^{s_i} \xi_j(\chi_j) | 0.1256  | 0.0125  | 0.0066  \
\frac{1}{s_i} \sum_{i=1}^{s_i} \xi_j(\chi_j) | 0.1256  | 0.0125  | 0.0066  
```

Consequently, keep the evolved breeding subpopulations $X^{(i)}_{s_i}$, $X^{(i)}_{s_i}$ and $X^{(i)}_{s_i}$.

Generate full-size populations $IP_{\text{full}}(X^{(i)}_{s_i})$ and $IP_{\text{full}}(X^{(i)}_{s_i})$, and evolve a separate fuzzy neural net for each subset $n^{(i)}_{\text{1}}$ and $n^{(i)}_{\text{2}}$ until the average value of the network error $\xi$ over the first half of the breeding subpopulation $X^{(i)}_{s_i}$ is less than $\xi_{\text{decend}}$.

```
project | objective | number of generations
--------|-----------|----------------------
project 1 | $2 \sum_{j=1}^{s_i} \max_s(\xi_j(\chi_j)) = 0.002482 < \xi_{\text{decend}} = 0.0025$ | 9,014
project 5 | $2 \sum_{j=1}^{s_i} \max_s(\xi_j(\chi_j)) = 0.0024996 < \xi_{\text{decend}} = 0.0025$ | 13,465
```

Consequently, keep the first half of the breeding subpopulations $X^{(i)}_{s_i}$ and $X^{(i)}_{s_i}$. Their size is $s_i/2 = 15$.

Check which single projects, elements of $n^{(i)} = n^{(i)} = \{\text{project2, project3, project4, project6}\}$, satisfy the objective $\frac{1}{s^{(i)}_{\text{1}}} \sum_{j=1}^{s^{(i)}_{\text{1}}} \xi_j(\chi_j) < \xi^{(i)}_{\text{dec2}} = 0.0225$.

```
objective / project | project 2 | project 3 | project 4 | project 6
-------------------|---------|---------|---------|---------
\frac{1}{s^{(i)}_{\text{1}}} \sum_{j=1}^{s^{(i)}_{\text{1}}} \xi_j(\chi_j) | 0.0644  | 0.0631  | 0.0608  | 0.0181
```

The value for project 6 is less than 0.0225. Consequently, the following second-level decomposition is identified.

**Level 2:** $n^{(i)} = \{n^{(i)}_{\text{1}}, n^{(i)}_{\text{2}}\} = \{\text{project 6}\}$, $n^{(i)}_{\text{3}} = \{\text{project 2, project 3, project 4}\}$

Generate a full-size population $IP_{\text{full}}(X^{(i)}_{s_i})$ and evolve a separate network for $n^{(i)}_{\text{3}}$ until the average error $\xi$ over the first half
of the breeding subpopulation \( X_{\alpha_{\mu}^{(2)}}^{(1)} \) is less than \( \xi_{\text{DECEND}} \).

<table>
<thead>
<tr>
<th>project</th>
<th>objective</th>
<th>number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>project 6</td>
<td>( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.0002468 &lt; \xi_{\text{DECEND}} = 0.0025 )</td>
<td>5,147</td>
</tr>
</tbody>
</table>

Consequently, keep half of the breeding subpopulation \( X_{\alpha_{\mu}^{(1)}}^{(2)} \).

\( \Rightarrow \) Generate a full-size population \( IP_{\mu} \left( X_{\alpha_{\mu}^{(1)}}^{(2)} \right) \) and evolve for \( N_{\mu^2} = 10,000 \) generations a separate network with the training set \( n_{i_1}^{(1)} = n^{(1)} = \{ \text{project 2, project 3, project 4} \} \). The result for the next objective is \( \frac{1}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.0638 < \xi_{\text{DECEND}} = 0.064 \).

Consequently, keep the evolved breeding subpopulation \( X_{\alpha_{\mu}^{(1)}}^{(1)} \).

\( \Rightarrow \) Generate a full-size population \( IP_{\mu} \left( X_{\alpha_{\mu}^{(1)}}^{(1)} \right) \) and evolve further the network with \( n^{(1)} \), for a multiple of \( N_{\mu^2} \) generations, until some of the elements of \( n^{(1)} \) meets the objective \( \frac{1}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) < \xi_{\text{DECEND}} = 0.0425 \). Only after 40,000 generations we get the result.

<table>
<thead>
<tr>
<th>objective / project</th>
<th>project 2</th>
<th>project 3</th>
<th>project 4</th>
<th>projects 3&amp;4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{1}{s_1 \sum_{i=1}^{b_1}} \sum_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) )</td>
<td>0.0554</td>
<td>0.0380</td>
<td>0.0381</td>
<td>0.0421</td>
</tr>
</tbody>
</table>

Project 3 and project 4 meet the condition. Moreover, they meet the condition together, as well. Consequently, the following third-level decomposition is identified.

\[ \text{Level} \colon n^{(1)} = [n_{j_{1}}^{(1)}, n_{j_{2}}^{(1)}], n_{j_{1}}^{(1)} = \{ \text{project 3, project 4} \}, n_{j_{2}}^{(1)} = \{ \text{project 2} \} \]

\( \Rightarrow \) Evolve a separate fuzzy neural network for \( n_{j_{1}}^{(1)} \) and for \( n_{j_{2}}^{(1)} \), until the average network error \( \bar{\xi} \) over the first half of the breeding subpopulation \( X_{\alpha_{\mu}^{(1)}}^{(1)} \) is less than \( \xi_{\text{DECEND}} \).

<table>
<thead>
<tr>
<th>project</th>
<th>objective</th>
<th>number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>projects 3&amp;4</td>
<td>( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.002426 &lt; \xi_{\text{DECEND}} = 0.0025 )</td>
<td>1,403</td>
</tr>
<tr>
<td>project 2</td>
<td>( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.002476 &lt; \xi_{\text{DECEND}} = 0.0025 )</td>
<td>1,312</td>
</tr>
</tbody>
</table>

Keep half of the breeding subpopulations, \( X_{\alpha_{\mu}^{(2)}}^{(1)} \) and \( X_{\alpha_{\mu}^{(2)}}^{(1)} \). This concludes the decomposition part of the bidirectional evolutionary strategy, as \( n_{j_{1}}^{(1)} \) consists of a single project.

**Incremental part**

\( \Rightarrow \) Consider the training set \( n^{(1)} = [n_{j_{1}}^{(1)}, n_{j_{2}}^{(1)}] \) and generate a full-size population \( IP_{\mu} \left( X_{\alpha_{\mu}^{(2)}}^{(2)} \right) \). Evolve a network until the objective \( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) < \xi_{\text{INC}} = 0.002 \) is met. Keep half of the resulting breeding subpopulation \( X_{\alpha_{\mu}^{(2)}}^{(2)} \).

<table>
<thead>
<tr>
<th>projects</th>
<th>objective</th>
<th>number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2&amp;3&amp;4</td>
<td>( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.001962 &lt; \xi_{\text{INC}} = 0.002 )</td>
<td>1,918</td>
</tr>
</tbody>
</table>

\( \Rightarrow \) Consider the training set \( n^{(1)} = [n_{j_{1}}^{(1)}, n_{j_{2}}^{(1)}] \) and generate a full-size population \( IP_{\mu} \left( X_{\alpha_{\mu}^{(2)}}^{(2)} \right) \). Evolve a network until another objective \( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) < \xi_{\text{INC}} = 0.002 \) is satisfied. Keep the resulting half-subpopulation \( X_{\alpha_{\mu}^{(2)}}^{(2)} \).

<table>
<thead>
<tr>
<th>projects</th>
<th>objective</th>
<th>number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>2&amp;3&amp;4&amp;6</td>
<td>( \frac{2}{s_1 \sum_{i=1}^{b_1}} \max_{\alpha_{\mu}^{(2)}} \bar{\xi}(X_{\alpha_{\mu}^{(2)}}) = 0.001984 &lt; \xi_{\text{INC}} = 0.002 )</td>
<td>503</td>
</tr>
</tbody>
</table>

\( \Rightarrow \) Consider the training set \( n^{(1)} = [n_{j_{1}}^{(1)}, n_{j_{2}}^{(1)}, n_{j_{3}}^{(1)}] = [n_{j_{1}}^{(1)}, n_{j_{2}}^{(1)}, n_{j_{3}}^{(1)}] \) and generate a full-size population \( IP_{\mu} \left( X_{\alpha_{\mu}^{(2)}}^{(2)} \right) \). Evolve a network until the error of the best chromosome is \( \bar{\xi} < \xi_{\text{INC}} = 0.0015 \).

<table>
<thead>
<tr>
<th>projects</th>
<th>objective</th>
<th>number of generations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1&amp;2&amp;3&amp;4&amp;5&amp;6</td>
<td>( \bar{\xi} = 0.0014904 &lt; \xi_{\text{INC}} = 0.0015 )</td>
<td>15.481</td>
</tr>
</tbody>
</table>

Keep the best chromosome \( X_{\text{INC}} \). It represents the fully evolved fuzzy neural network.