## preface

*foreword*

In one and a half year – starting from August 2005 – the idea of creating a structural design optimisation tool grew to the realisation of a script, written in VBA and using AutoCAD 2007 as a visualisation tool, which has the ability to optimise the allocation of a variable number of columns in a structural transfer zone. An extended preliminary study is executed in (computational) structural optimisation – book two –, and the aspects and characteristics of transfer zones, modular dimensions, and structural grid layouts are discussed – book one. Based on the knowledge of optimisation methods gained from the preliminary study phase and the research stage, a design tool, intended to be used in the conceptual and preliminary design stage, is written, resulting in a VBA-script [for information on VBA, see Appendix A] of over 4000 lines. A test structure is presented in the addendum, including the determination of the optimal GA operator parameters.

This addendum is part of an integral report containing all findings of the Master's thesis on structural optimisation for the Delft University of Technology, Faculty of Civil Engineering and Geosciences, department of Building Engineering, and the Structural Design Lab (SDL). Two books including this addendum form the complete and final report of this Master's thesis.

**book one:** **transfer zones in multi-use buildings and the development and analysis of the optimisation tool**
**book two:** **background studies on structural optimisation**
**addendum:** **determination of the optimal GA operator parameters and the presentation of test results based on example structures**

The reader of the complete report will find that the emphasis is mainly on the study into genetic algorithms and the onset of the design optimisation tool.

This addendum presents the various test results, mainly dealing with the genetic algorithm operator parameters. Furthermore, two example structures are presented in order to demonstrate the capabilities of the VBA script.

Delft, January 2007

R.J. (Roel) van de Straat

*graduation committee*


| | |
|---|---|
| name: | **prof. dipl.-ing. J.N.J.A. Vamberský** – chairman of the committee |
| university: | Delft University of Technology |
| | Faculty of Civil Engineering and Geosciences |
| address: | Stevinweg 1 |
| | 2628 CN Delft, room 1.35 St-II |
| | The Netherlands |
| telephone: | +31152783174 (secretariat) |
| e-mail: | j.n.j.a.vambersky@citg.tudelft.nl |

| | |
|---|---|
| name: | **ir. J.L. Coenders** – daily supervisor |
| university: | Delft University of Technology |
| | Faculty of Civil Engineering and Geosciences |
| address: | Stevinweg 1 |
| | 2628 CN Delft, room 1.58 St-II |
| | The Netherlands |
| telephone: | +31152785711 |
| e-mail: | j.l.coenders@citg.tudelft.nl |

| | |
|---|---|
| name: | **ir. J.W. Welleman** |
| university: | Delft University of Technology |
| | Faculty of Civil Engineering and Geosciences |
| address: | Stevinweg 1 |
| | 2628 CN Delft, room 6.15 |
| | The Netherlands |
| telephone: | +31152784856 |
| e-mail: | j.w.welleman@citg.tudelft.nl |

| | |
|---|---|
| name: | **ir. S. Boer** |
| organisation: | ONL [Oosterhuis_Lénárd] (till March 2006) and Mecanoo architecten |
| home address: | Goudsesingel 464 |
| | 3011 KP Rotterdam |
| telephone: | +31648082965 |
| e-mail: | sanderboer@myrealbox.com |


*consultant*


| | |
|---|---|
| name: | **dipl. inform. F. Scheurer** |
| university: | Eidgenössische Technische Hochschule Zürich |
| | Faculty of Architecture, Chair of CAAD |
| address: | Hoenggerberg HIL E15.1 |
| | CH-8093 Zürich |
| telephone: | +41446334025 |
| e-mail: | scheurer@arch.ethz.ch |

*acknowledgements*

Many people helped in the realisation of this Master's thesis, and it is therefore underlined that, without their input and effort, the challenge of writing this report would be far greater. That is the reason why here those people are thanked wholeheartedly.

**The graduation committee**
in general is thanked for the positive input, the advice, and the criticism given as a group. Unanimity, consistency, and perseverance are characteristics the committee cannot be denied.
In particular, **prof. dipl.-ing. J.N.J.A. Vamberský**, for his ability to give an answer to a question just by listening, his great passion for the building engineering practise (and the way this is transferred to his students), and the eagerness to comprehend new techniques and methods shown by his students. **Ir. J.W. Welleman**, for his enormous ardency and enthusiasm, especially after having spent a considerable amount of time at home due to illness. Extensive collaboration on the Discrete Element Method led to the implementation of this method in this Master's thesis. **Ir. J.L. Coenders**, for the introduction to and shared knowledge of computational design and structural optimisation. Due to his work for the Structural Design Lab the addition of the name of this group on the front page of this Master's thesis, really means something in the computational and building engineering practise. **Ir. S. Boer**, as the only committee member not associated with the Delft University of Technology, for dealing with the extra time he put into this project, and for his insights in the architectural practise. Something not only used for this Master's thesis, but which will also be used for the upcoming Master's study Architecture.

**Dipl. inform. F. Scheurer**
is willingly thanked for being the non-official member of the graduation committee and for sharing his knowledge on computational design and especially on genetic algorithms. Based on his papers, and visits to Delft, better insights were received into this complex matter.

**Ir. A. Habraken**
of Arup Amsterdam who was open for an invitation from him to the Arup office in Amsterdam and could answer various structural related questions on the Groningen Twister project, and thereby supplied better understanding in the design of column supported slabs.

**ir. S. de Groot and ir. E. Quispel**
for their preliminary insights on the subject of stacking different functional layouts within a multi-use building and their willingness to share their work.

**H. Ohmori, prof. dr. J. Strasky, and ir. R. Uytenhaak**
who supplied images and text files after requests via e-mail.

**ir. R. Pijpers**
who, with enormous patience, was able to install AutoCAD 2007 on the desktop at the workspace, which was of essential importance for finishing this Master's thesis.

**the graduation students of room 0.72**
for their tips and tricks on how to graduate, the laughs that were shared during the complete Master's thesis period, and for the many final presentations and accompanying drinks from those who already graduated.

**the fellow board members and friends of the Stichting Dispuut Utiliteitsbouw**
who made the finishing of the 'U-profiel' of May 2006 a lot more bearable, who opened up the continent of Asia for an unforgettable study tour to Dubai and the east of China at end of the winter of 2006, where, visiting Shanghai, the research stage of this Master's thesis already could be presented to Chinese fellow students. And of course for their friendship, during the whole combined period of working on the Master's thesis and the board membership.

**my parents, my brother, and his girlfriend**
for being the biggest support, a graduate student could wish for. Enough said.

And of course everyone else, who, although not contributing directly to the content of this Master's thesis, showed that there is more to life than columns, structural grids, and genetic algorithms, but who contributed to the social and emotional evolution of the author.

## summary

*general introduction*

As more functional requirements are implemented in one building, more pressure on the structural designers, both architects and building engineers, will be transferred. Especially in dense urban areas, multi-use buildings usually have several different layouts, for instance for an underground parking, a ground floor, office floors, and mechanical floors. One of the main problems here is the transfer of loads between the different structural grid systems of the different functions, and as a result, numerous structures are built and will be built with inefficient transfer zones considering the volume of material needed for a floor or the amount of reinforcement or prestressing needed. In lieu of making compromises in the design of structural grid systems per function in order to vertically align the grid lines or grid bands, a structural transfer zone in an intervening floor can be designed. By designing an optimal allocation of vertical oriented structural elements, loads can be transferred between the different structural grid systems, without adversely affecting the functionality and usability of the intervening floor.

*structural optimisation design tool*

The Master's thesis 'Optimisation of structural transfer zones in multi-use buildings' deals with the development of a computational structural design tool based on an artificial intelligence method, that can determine the optimal solution for the allocation of columns between two structural grid systems. Based on genetic algorithms as an optimisation technique and by using basic and simple rules, the design tool can be used for the determination of size, angle, and placement of load bearing columns of the intervening floor. This tool, implemented in VBA and using AutoCAD as a visualisation tool, allows the user to generate and optimise the configuration of the load bearing elements for an arbitrary design, with the rules following from the demands of several aspects of a structural and functional design. With the addition of non-structural criteria (such as costs, aesthetics and construction) the engineer can complete the design, and create an optimised design that is based on the integration of the load bearing elements into a functionally efficient building floor, rather than being only based on the stress-weight optimisation of the individual components.

*features of the tool*

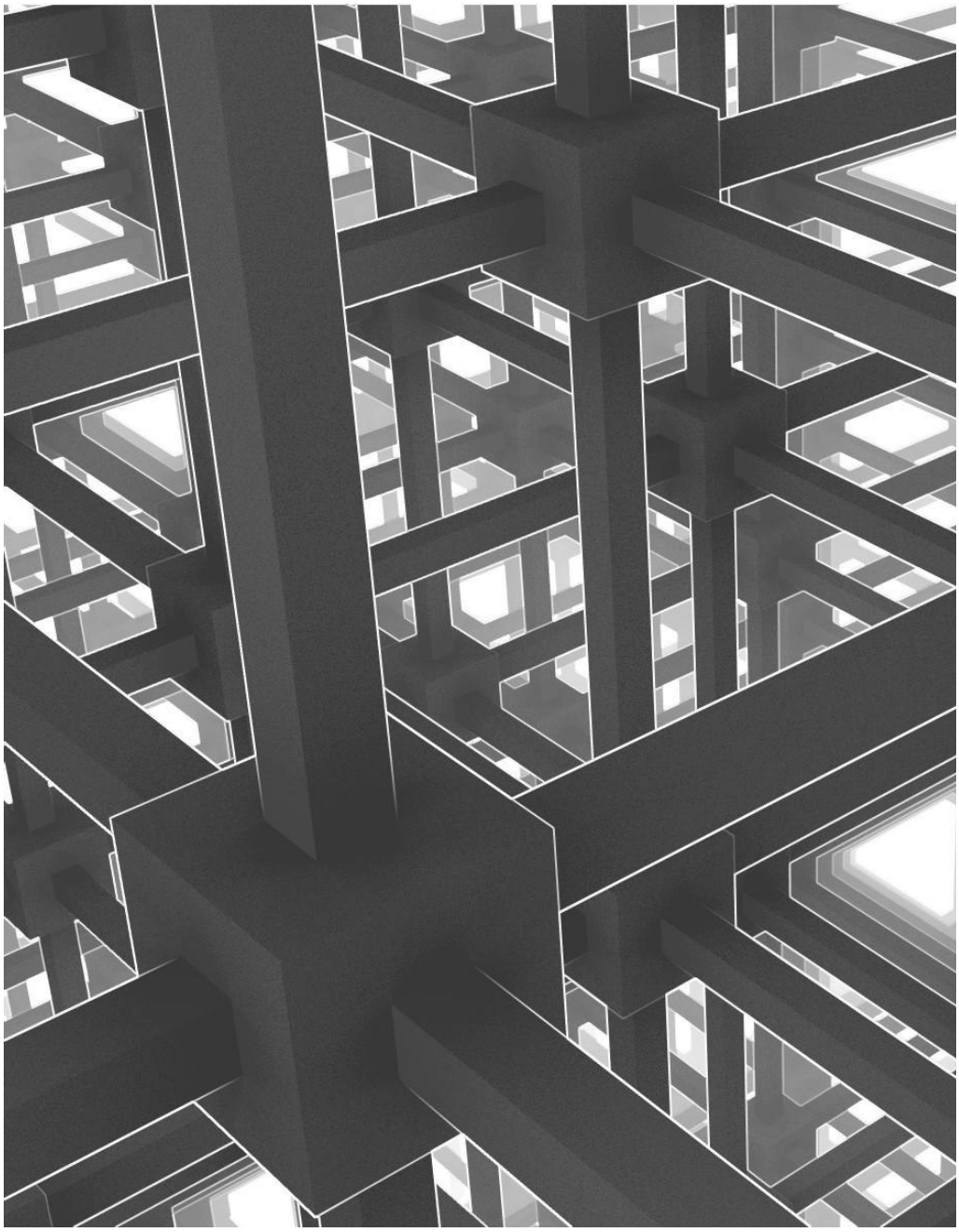
In the AutoCAD environment, the user needs to give the starting and end point of the structural grid line graphically in both the bottom and top layer of the intervening floor to determine the possible location of the columns. Subsequently, the vertical point load, the horizontal point loads, and the moments in the centre of gravity acting on the load transferring structure need to be given numerically. Based on the user input, the tool then randomly generates several solutions in what is called the first generation. By determining the fitness, or in other words the overall compliance with the prescribed desired conditions, the genetic algorithm will use the best solutions to create a new population. This process, including several other genetic algorithm operators, will be repeated until the fittest or best solution per population remains unaltered during a number of generations.

*concluding remarks*

This Master's thesis shows the capability of an artificial intelligence based design optimisation tool in a predefined setting of the allocation problem of columns in a structural transfer zone. At the same moment, it is made clear that progress can be made for the presented design tool and in scripting design tools for the building practise in general. This also means that it can be expected that tools similar to the tool presented in the Master's thesis will be used more often in the near future. This, however, does not mean that the structural engineer will lose his or her position, as hand calculations and logical interpretations of the result of the design tools will always have to be made.

# VII summary

# table of contents



**Cubic Space Division, M.C. Escher, litho, 1925, 27 x 26,5 cm**[1]

*One another intersecting rectangular beams divide each other in pieces of equal length, which each are the edge of a cube. So, the space is filled to infinity with cubes of the same volume*

---

[1] Escher, M.C., *M.C. Escher – Grafiek en tekeningen*. Köln: Taschen GmbH, 2001

# 1 introduction

*general introduction to the addendum*

As stated in the preface, in book one and book two of the Master's thesis *'optimisation of structural transfer zones in multi-use buildings'*, an extended preliminary study is executed in (computational) structural optimisation – book two –, and the aspects and characteristics of transfer zones, modular dimensions, and structural grid layouts are discussed – book one. The final two chapters of book one deal with the implementation of various parametrical aspects of the design tool and the validation of the VBA-script. This addendum presents the results of numerous test runs, based on fixed and/or variable GA operator parameters, and demonstrates the capabilities of the VBA-script coupled to two example structures.

*objectives of the tests*

The main goal of the tests is the determination of the optimal values of the population size, the crossover rate, and the starting mutation rate including the halving time. In general, these tests are executed with all fixed parameter values except the one under consideration. Most test runs are performed and optimised for the stability and volume fitness parameters, as this combination result in an easy-to-check evolutionary process, and will be executed for 200 generations. Both parameters are weighed with a factor '1', thus the optimal value is '2'.

In deciding on the overall success of the tests, three aspects are of main importance:

1. the maximum fitness after 200 generations
2. the speed of convergence
3. the diversity of the population

Based on these aspects, the test runs and the accompanying parameter values are evaluated.

*composition of the addendum*

In the second chapter, the three GA operators – the population size, the crossover rate, and the starting mutation rate including the halving time – are subjected to various test runs, in order to determine the optimal values It is important to state that the determination of these values are under the restriction of the problem to be solved. This means that the optimal parameter values for the operators may differ when, for example, increasing of decreasing the possible number of columns.
Based on the optimal values presented in Chapter 2, for two example structures – a test box and the residential building La Fenêtre [see also Sections 3.1 and 4.1 of book one – the optimal column allocation is determined based on the VBA-script in the subsequent chapters.

# 2 optimisation of the GA operator parameters

In this chapter, three genetic algorithm operator parameters are tested for their optimal value, considering their contribution to optimal end results, convergence, and diversity. In Section 2.1 the population size values are examined. Section 2.2 deals with the optimal value for the crossover rate. The starting mutation rate value is tested in Section 2.3, including the halving time parameter, needed for the exponential decrease of the mutation probability.

## 2.1 optimising the population size value

*introduction*

The number of individuals per generation, or the population size, is an important parameter in the evolutionary process. Although there are various techniques for managing the population size, the examples in this Master's thesis deal with a fixed number of individuals per generation.

For a large value of the population size, it holds true that, since many samples from the search space are used, the probability of convergence to a global optimal solution is higher than when using a small population size. However, a large population size means the simultaneous handling of many solutions and increases the computation time per iteration. Orin other words, running a simulation with large models can be expensive in terms of time, so a smaller population may be desirable. Another potential problem is that the search may be flooded with an abundance of genetic diversity, resulting in generations with all identical individuals, and an average fitness value equal to the maximum fitness value. On the other hand, if the population is too small then the loss of genetic diversity may compromise the search.

In the 'charts, and graphical and numerical output' chapter at the end of this addendum, 24 test results are presented including various values of the population size. The crossover rate, starting mutation rate, halving time, and the number of columns are fixed [values are given in the subsection *population size tests*]. As the genetic evolutionary process starts with a random initial population, the eight different values – 3, 9, 27, 45, 69, 99, 135, and 199 individuals per generation – are tested three times.

*results*

The graphs in the Subsection *population size tests* of the final chapter, show that for a population size equal to or smaller than 27, the maximum value barely exceeds the 1,50. The difference between the maximum fitness value (black line) and the average fitness value (magenta line) converges to zero, and for the population size of 3 and 9, the evolutionary process already convergence to the optimum value before reaching the number of 200 generations (which can be a good thing, but not in this case).

For the number of individuals per generation from approximately 50 and up, the average maximum value for the volume parameter (blue line in the average values chart) fluctuates between 1,55 and 1,60. In addition, the difference between the maximum and the average fitness values for these numbers of population size, shows that every generation contains a certain diversity in the populations.

Purely based on the average maximum value for the volume parameter, the optimal number of individuals per generation lies between sixty and eighty.

## 2.2 optimising the crossover rate value

*introduction*

Crossover allows selected members of the population to exchange characteristics of the design among themselves, and roughly mimics biological recombination between two single-chromosome organisms. In the case of crossover, 'superior' individuals should have more opportunities to reproduce. In doing so, the offspring contains combinations of the genetic material of the best individuals. The next generation is therefore strongly influenced by the genes of the fitter individuals. However, it is not necessary that every couple of parents produces offspring. In fact, crossover takes place at a certain probability, referred to as the *crossover rate* $p_c \in [0,1]$.

The choice of the crossover rate as a control parameter can be a complex nonlinear optimisation problem to solve itself. The increase of crossover probability would cause the recombination of building blocks to rise, and at the same time, it increases the disruption of good chromosomes. The settings are critically dependent upon the nature of the objective function. As stated in book one, the selection of the crossover rate issue still remains open to suggestion although some guidelines have been introduced. For large populations ($N_p \geq 60$), the crossover rate can be set at 0,6. For small populations ($N_p < 60$) the crossover rate can be set at 0,9.

The crossover rate is tested for the values of 0,0; 0,1; 0,3; 0,5; 0,7; 0,9 and 1,0.

*results*

The test runs for the determination of the optimal crossover rate value where executed with a population size of 69, a starting mutation rate value of '0,1', a halving time of 30 generations, and a number of columns of 28.

With these values, the overall performance of the test runs where relatively successful. For all tested crossover rates, the speed of convergence (although rather slow), the diversity of the solutions (the difference between the average fitness value and the maximum fitness value), and the maximum value at $t = 200$ the results are seemingly the same. When the averages of three test runs for the various crossover rates are compared [see the chart in the final chapter, Subsection *crossover rate tests*] it can be concluded that the small peak at a crossover rate of approximately 0,5 indicates the optimal value.

Probably, the main reason for the small contribution of the crossover for this problem, and thus the relative inconsequentiality of the optimal crossover rate, results from the chosen method of one-point crossover instead of multi-point crossover. This latter method is not evaluated in this Master's thesis.

## 2.3 optimising the starting mutation rate and halving time values

*introduction*

The aim of mutation is to introduce new genetic material into an existing individual, that is, to add diversity to the genetic characteristics of the population. Mutation is used in support of crossover to make sure that the full range of allele values is accessible in the search. Mutation also occurs at a certain probability $p_m$, referred to as the *mutation rate*. Usually, a small value for $p_m \in [0,1]$ is used (so close to 0) to ensure that good solutions are not distorted too much. Should the mutation probability increase, this would transform the genetic search into a random search, but would help to reintroduce the lost genetic material. It is proposed that for large populations ($N_p \geq 60$), the constant mutation rate can be set at 0,001. For small populations ($N_p < 60$) the constant mutation rate can be set at 0,01. However, it is stated again that these values are fully dependant on the problem under consideration. However, research has shown that an initial large mutation rate that decreases exponentially as a function of the number of generations improves convergence speed and accuracy. The initial large $p_m$ ensures that a large search space is covered, while the $p_m$ becomes rapidly smaller when individuals start to converge to the optimum.

*results*

Whereas the influence of the crossover rate is relatively small for this problem, the influence of the starting mutation rate and the accompanying halving time are extremely important.

Thirty-one tests runs initially have been executed to test the influence of the following starting mutation rate and halving time.

| | |
|---|---|
| starting mutation rate: | 0,001; 0,005; 0,01; 0,1 and 0,5 |
| halving time: | 1, 10, 50, 100, 500, 5000 generations |

These results are shown in Chapter 4, Subsection *starting mutation rate and halving time tests*. The results clearly show that for small halving times (≤ 10 generations), or in other words a rapidly decreasing mutation rate, and especially for small starting mutation rates, the diversity of the generation converges to zero. The reason for this, is the fact that with these parameters only crossover takes part in the evolutionary process, and the solutions converge to optima based solely on this GA operator. For large starting mutation rate values (≥ 0,5), the distortion of the generation prevents the convergence to good solutions, as can be seen in the horizontal lines, indicating the average fitness of the fitness values of a generation.

The results are summarised in two 2D-charts and one 3D-chart. From these charts the optimal values for the starting mutation rate and the halving time can be distilled. The first aspect of the GA operator, the starting mutation rate, is at an optimal value around 0,01. The halving time parameter value is preferably set to 50 for the problem considered.
However, as the steps around the optimal starting mutation rate parameter is relatively large, this parameter is tested again, this time for the fixed halving time of 50. When comparing the test result of an average of two tests for four different values between 0,005 and 0,1 – 0,0075; 0,010; 0,025 and 0,050 – still the value of 0,010 is indicated as the optimal value.

## 2.4 concluding remarks

In the three sections of this chapter, three genetic algorithm parameters are tested in order to determine their optimal value. The tests show that the influence of the population size and the starting mutation rate, including the halving time is considerably larger than the influence of the crossover rate. As stated in Section 2.2, this is due to the inferior method of one-point crossover for the large chromosomes in this problem compared to the method of multi-point crossover.

The optimal values for the various GA parameters are

| | |
|---|---|
| population size: | 69 |
| crossover rate: | 0,5 |
| starting mutation rate: | 0,01 |
| halving time: | 50 |

However, it is again stated that these values are completely dependant on the problem under consideration. And although the given optimal parameter values are probably applicable for multiple problems concerning the allocation of columns with the VBA-script, slight changes might improve their performance considerably.

# 3 example structure: the test box

In this chapter, a test box, used to check various aspects of the VBA-script during the design and development stage, is subjected to a load. Subsequently, an optimised structure, driven by the VBA-script is generated. This structure is presented in Section 3.2.

## 3.1 basic aspects of the text box

The test box has the dimensions as given in Figure 3.1.

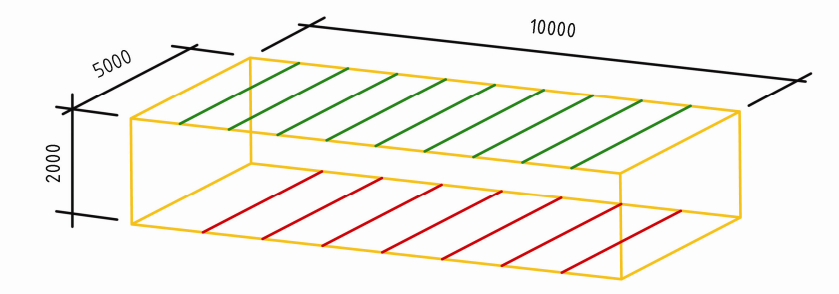



Fig. 3.1. Dimensions of the test box

The test box is loaded with only a vertical load of -10.000 kN. Section properties of the columns consist of circular hollow steel (S235) sections, with a wall thickness of 12,5 mm and a varying diameter of 120, 150, and 180 mm. The number of columns (awake and asleep) is 20, with a maximum length of 2400 mm. The fitness parameters are valued as follows

1 – stability
3 – bearing capacity
4 – volume
1 – slanting
0 – intersection
1 – variation

And finally the number of possible grid positions of the 7 grid lines at the bottom is 8, and the number of possible grid positions of the 9 grid lines at the top is 9.

## 3.2 the optimised structure for the test box

Based on the input of Section 3.2 the GA operator parameters are also valued.

| | |
|---|---|
| population size: | 69 |
| crossover rate: | 0,5 |
| starting mutation rate: | 0,1 |
| halving time: | 50 |

With 1500 generations, and a run time of approximately 45 minutes, the following results were obtained, Figure 3.2.

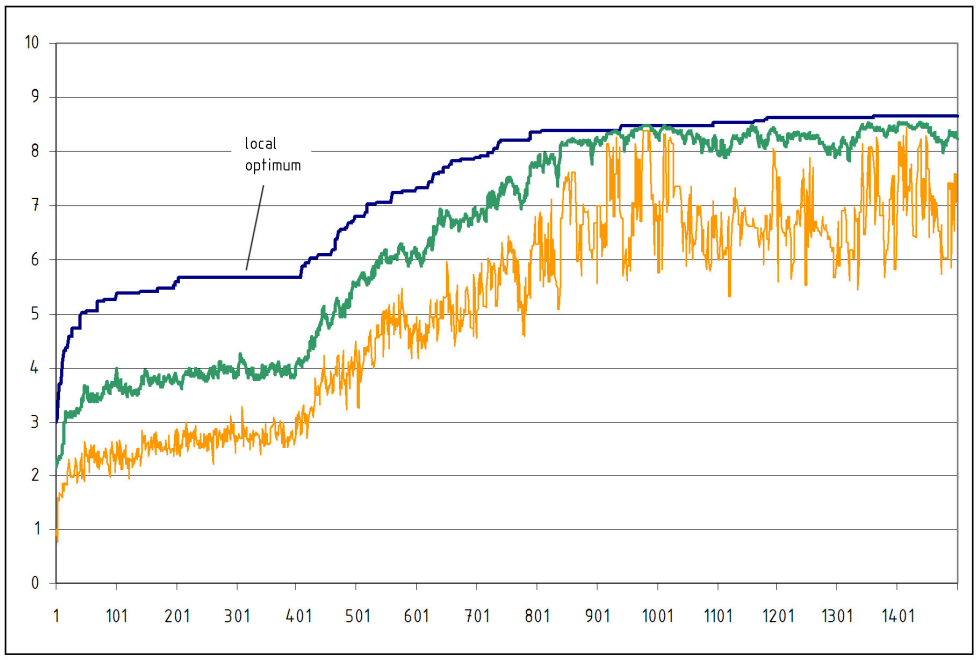



Fig. 3.2. The maximum (blue), the average (green, and minimum (orange) fitness values of every generation

The optimal summed fitness value is '10'. The VBA-script drove the evolutionary process to a value of over 8,65. An excellent result, if it is kept in mind that the bearing capacity and volume fitness parameters can never be fully complied. For the fist fitness parameter is holds true that due to discrete values in location and column thicknesses, it can never be valued '1'. The volume parameter cannot be driven to '1', as all column are then completely vertical, resulting in a structural mechanism.

The power of genetic algorithms to overcome local minima is also demonstrated in this graph. Between generation 203 and 407, the GA encounters a local optimum. But as the diversity in the population of these generations is adequately large, the GA is able to follow its path in finding the global optimal solution.

Below the values for the fitness parameters and the maximum, minimum, and average fitness are given for generation 1499, Figure 3.3.

| generation number | | stability parameter | bearing cap. parameter | volume parameter | slanting parameter | intersection parameter | variation parameter | | maximum fitness | | minimum fitness | average fitness |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| generation 1499 | | 1 | 2,465035452 | 3,297952219 | 0,990142419 | 0 | 0,9 | | 8,653130091 | 0 | 7,066466354 | 8,237999273 |

Fig. 3.3. Fitness values for generation 1499

Finally, the images of the optimised structure and the evolutionary process of it are given in Figure 3.4 and 3.5.

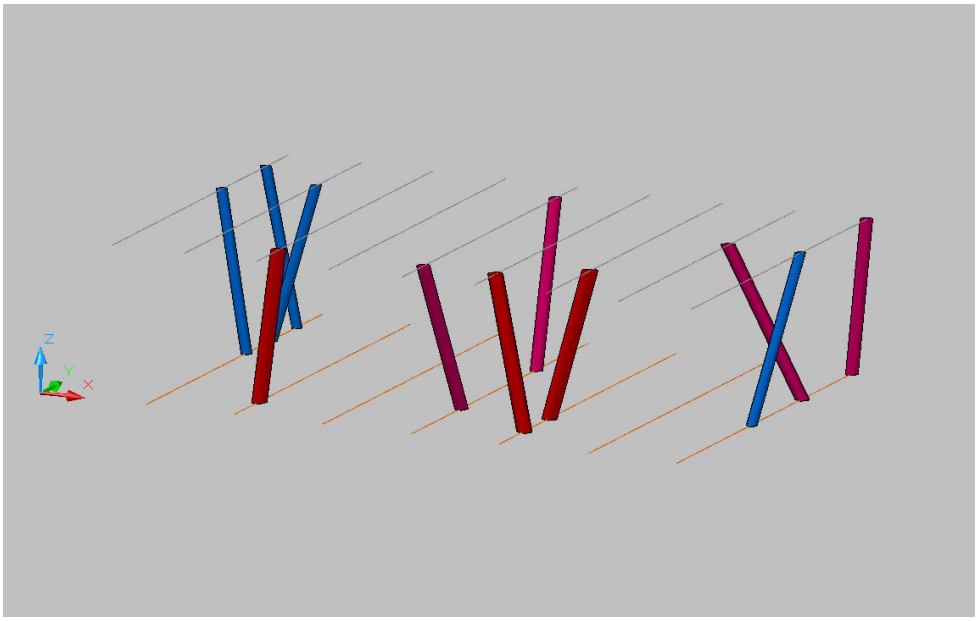



Fig. 3.4. The optimised structure for the test box for a vertical load of -10.000 kN

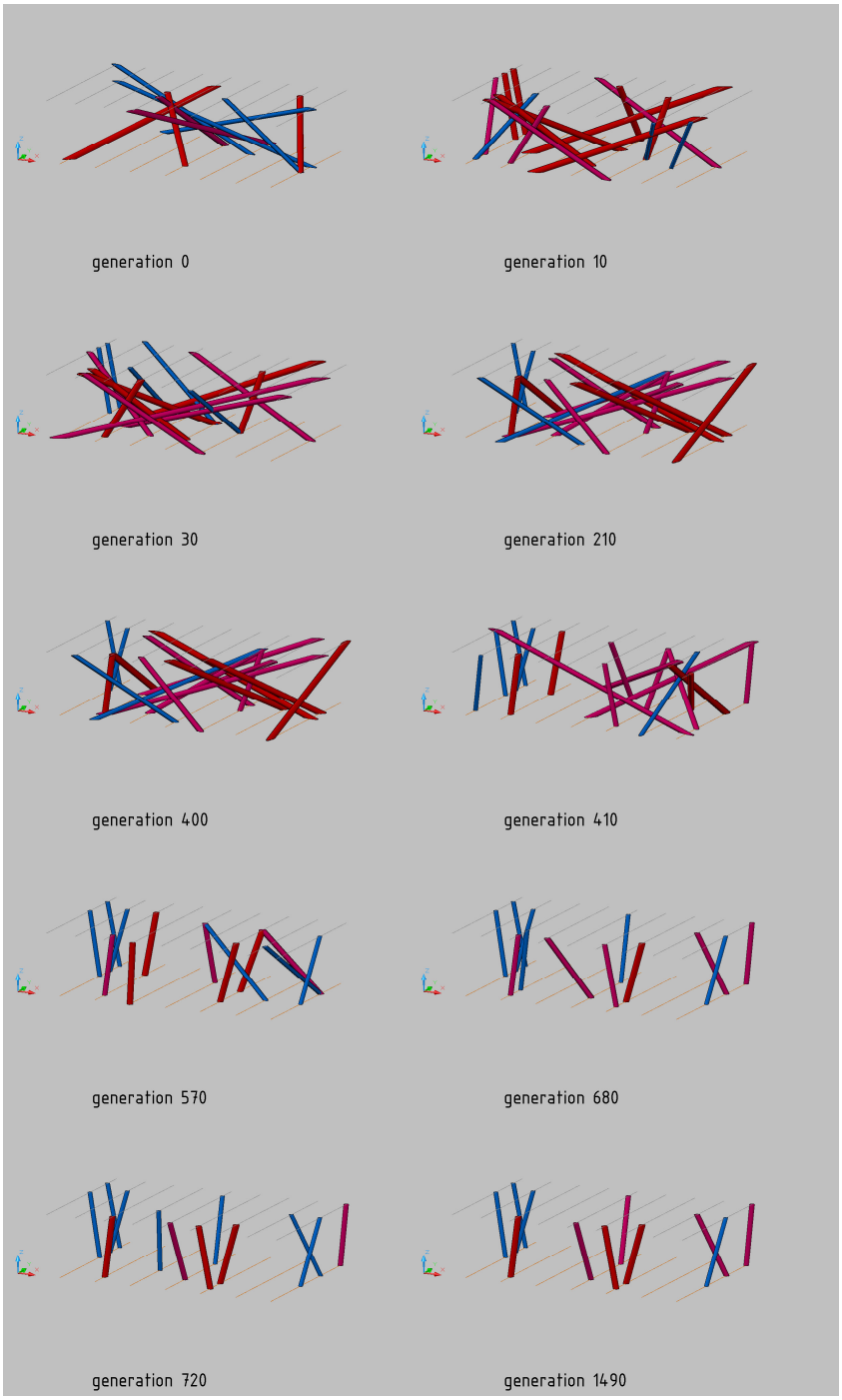



Fig. 3.5. The evolutionary process of the optimised structure

# 4 concluding remarks

Although the maximum number of generations for the evolutionary process of finding the optimal solution for the test box perhaps is not quite yet large enough (there was still a small improvement at *t = 1360*), it is shown in this addendum that the VBA-script is capable of generating a load bearing structure of columns, and optimising it according several fitness parameters.

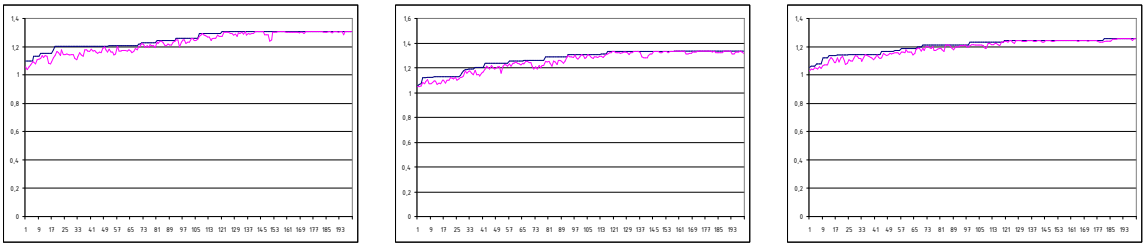
And however, due to the considerable run time increase of implementing the intersection fitness parameter, the parameter is omitted, the other parameters automatically drive the solution towards a structure without intersecting columns.

In conclusion, it can be stated that the VBA-script lives up to the expectations of finding a structure capable of transferring loads between two different grid systems and simultaneously complying with other denoted fitness parameters, such as the variation is column aspects.

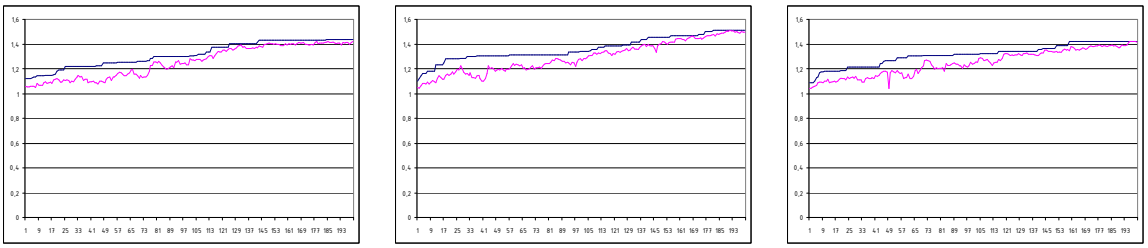
## charts, and graphical and numerical output

*population size tests*

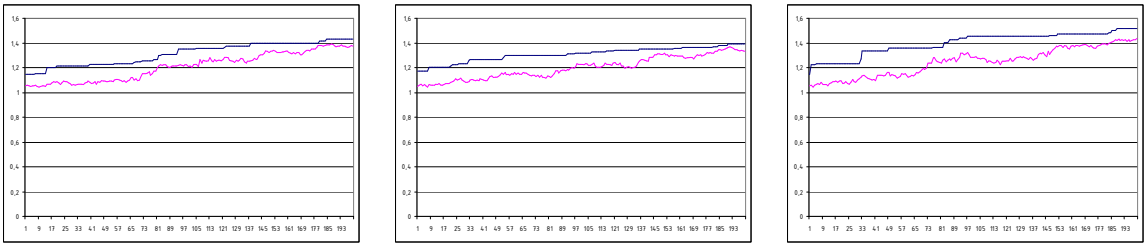
| | |
|---|---|
| number of generations: | 200 |
| crossover rate: | 0,50 |
| starting mutation rate: | 0,10 |
| halving time: | 30 generations |
| number of columns: | 28 |
| possible grid line positions bottom: | 11 |
| possible grid line positions top: | 12 |



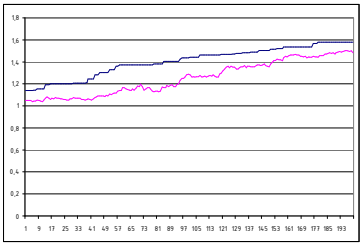
population size = 3

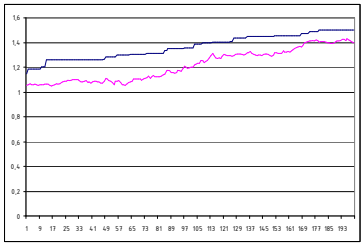


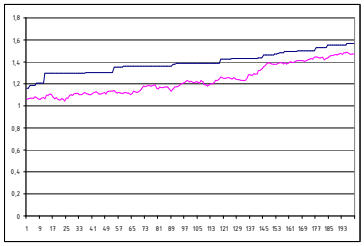
population size = 9



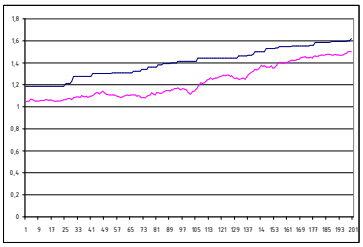
population size = 27

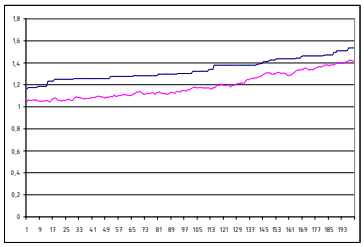


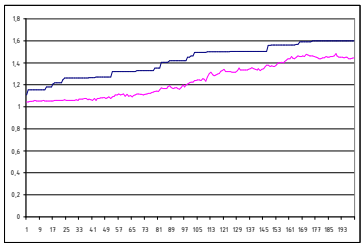




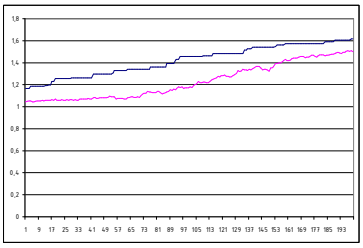
population size = 45

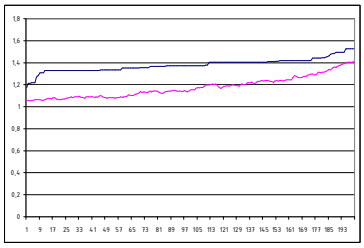


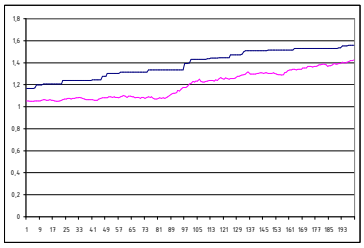




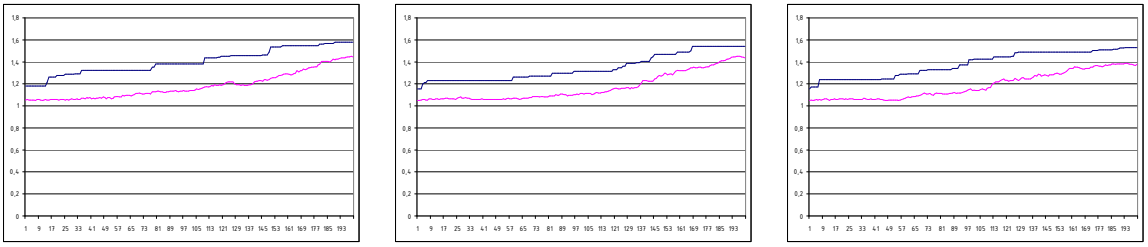
population size = 69



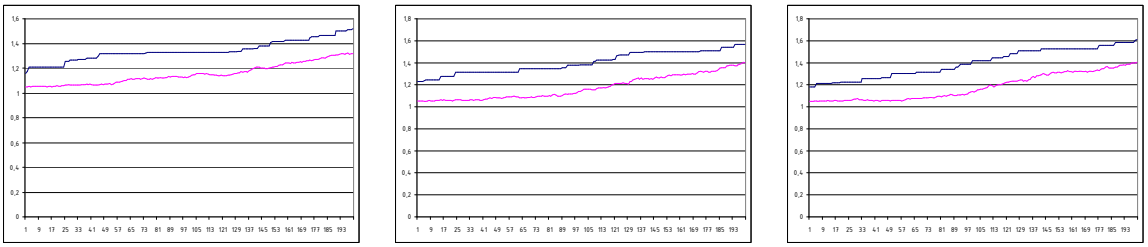




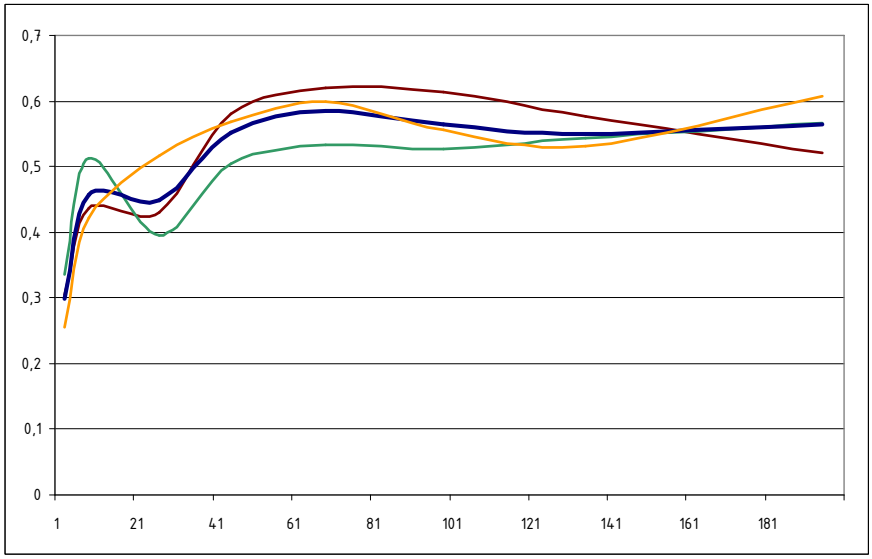
population size = 99



population size = 135



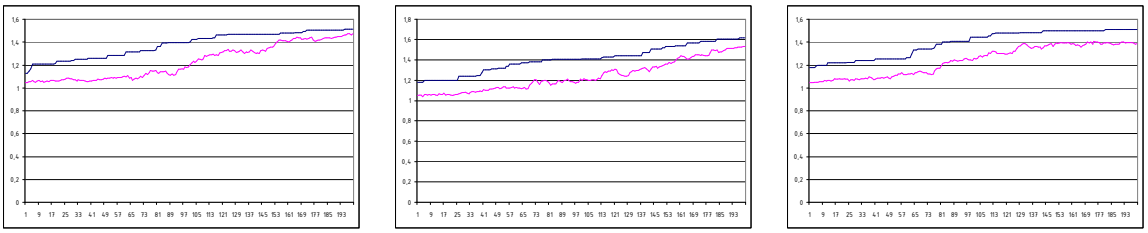
population size = 195



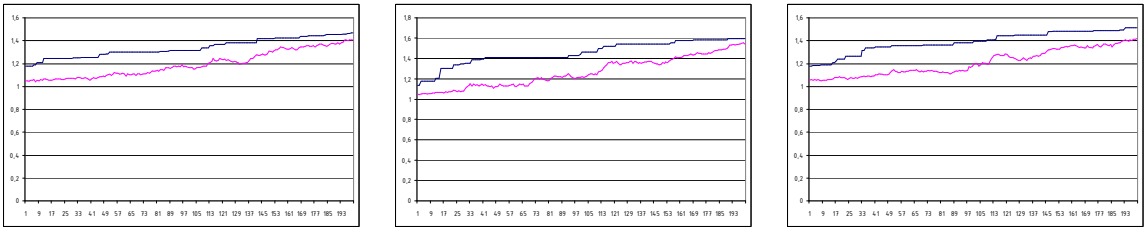

average values: test 1 – brown line, test 2 – green line, test 3 – orange line, average – blue line

*crossover rate tests*

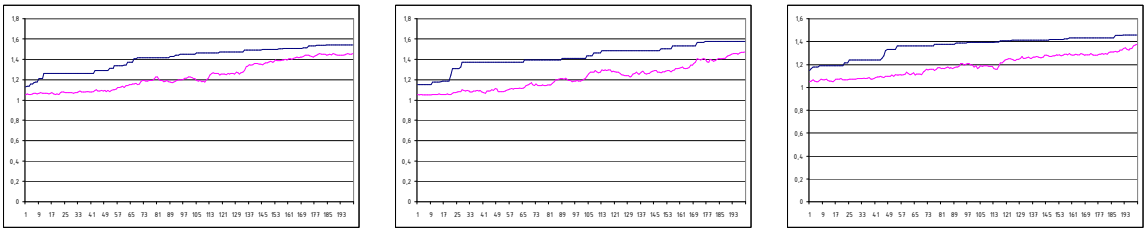
| | |
|---|---|
| number of generations: | 200 |
| population size: | 69 |
| starting mutation rate: | 0,10 |
| halving time: | 30 generations |
| number of columns: | 28 |
| possible grid line positions bottom: | 11 |
| possible grid line positions top: | 12 |



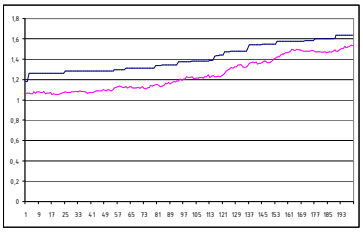
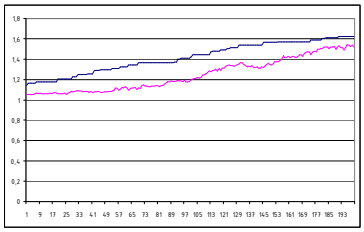
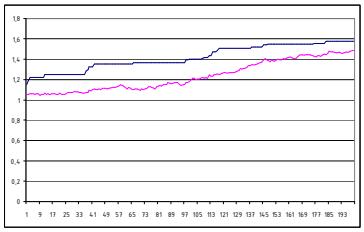
crossover rate = 0,0



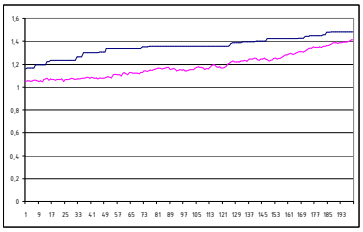
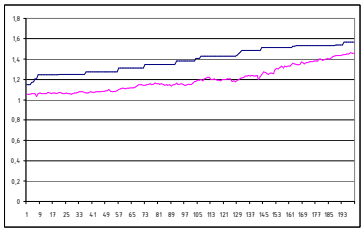
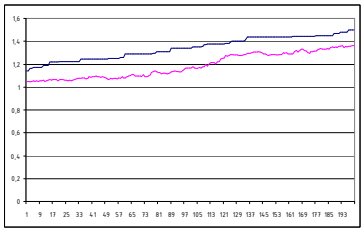
crossover rate = 0,1



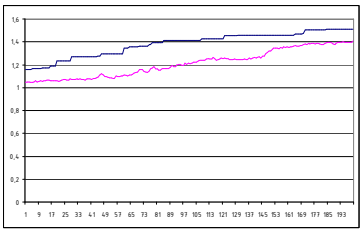
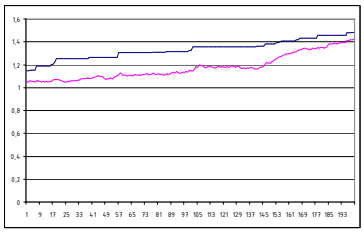
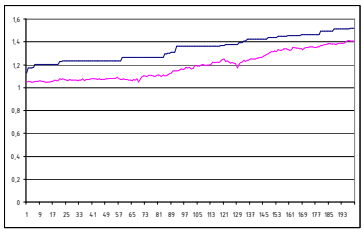
crossover rate = 0,3

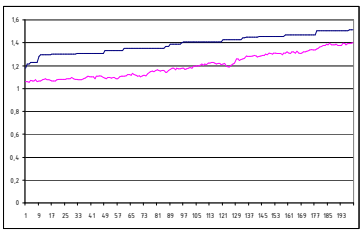
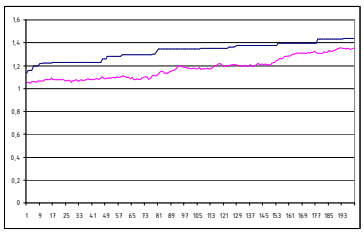
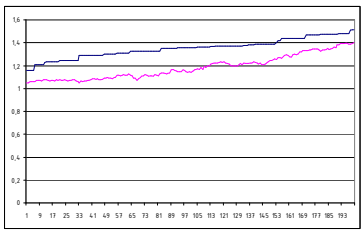
crossover rate = 0,5

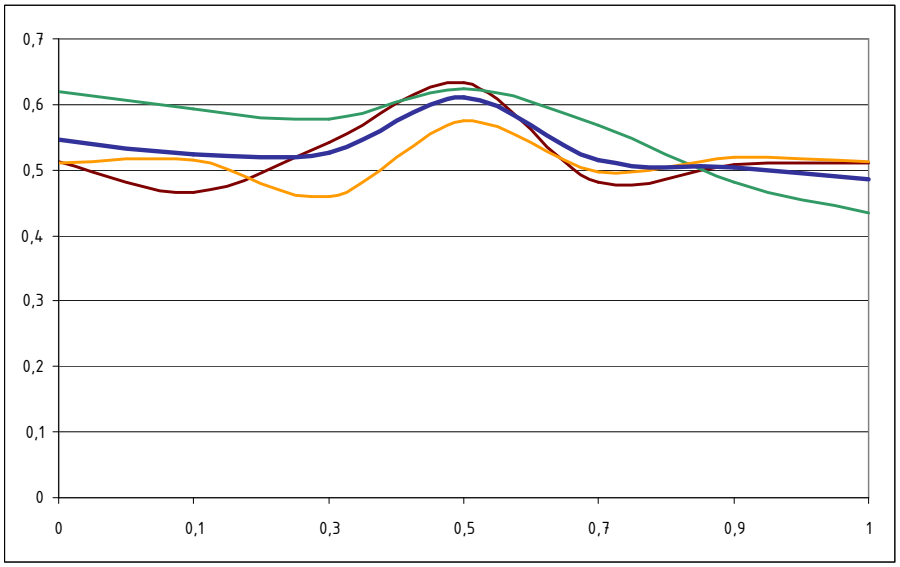
crossover rate = 0,7

crossover rate = 0,9

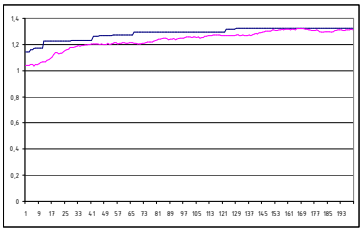
crossover rate = 1,0



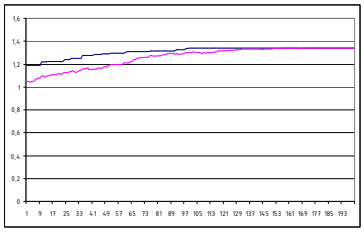

average values: test 1 – brown line, test 2 – green line, test 3 – orange line, average – blue line

*starting mutation rate and halving time tests*

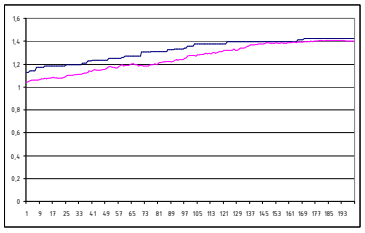
| | |
|---|---|
| number of generations: | 200 |
| population size: | 69 |
| crossover rate: | 0,50 |
| number of columns: | 28 |
| possible grid line positions bottom: | 11 |
| possible grid line positions top: | 12 |



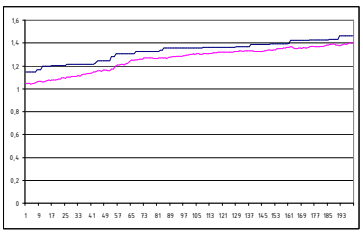
starting mutation rate = 0,001
halving time = 1



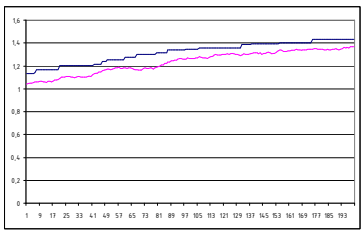
starting mutation rate = 0,001
halving time = 10



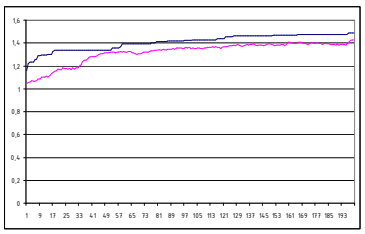
starting mutation rate = 0,001
halving time = 50



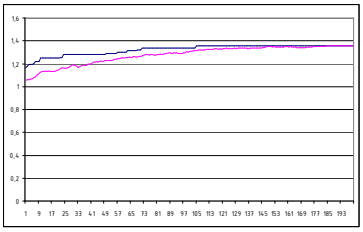
starting mutation rate = 0,001
halving time = 100



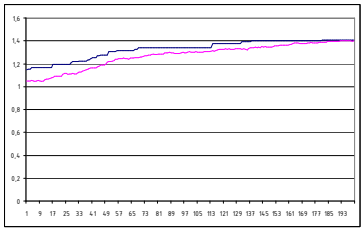
starting mutation rate = 0,001
halving time = 500



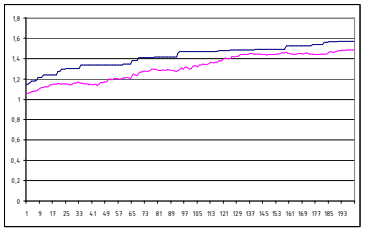
starting mutation rate = 0,001
halving time = 5000



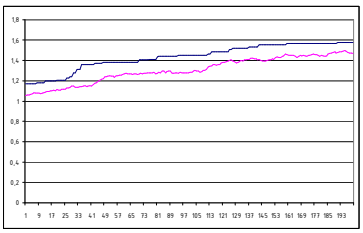
starting mutation rate = 0,005
halving time = 1



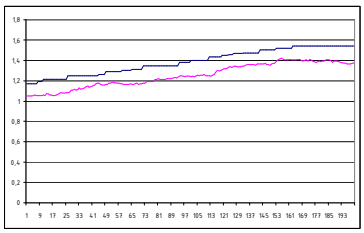
starting mutation rate = 0,005
halving time = 10



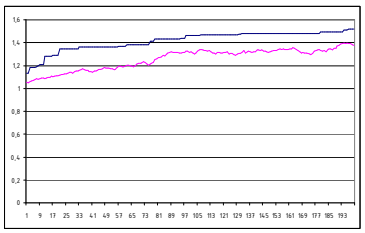
starting mutation rate = 0,005
halving time = 50



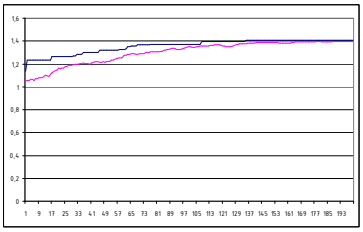
starting mutation rate = 0,005
halving time = 100



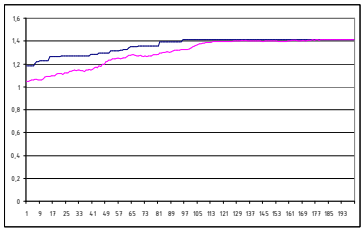
starting mutation rate = 0,005
halving time = 500



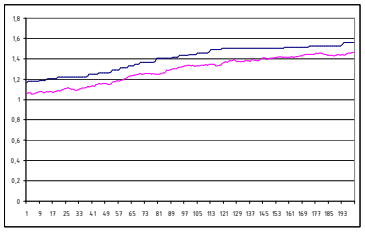
starting mutation rate = 0,005
halving time = 5000



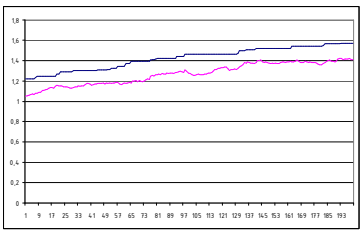
starting mutation rate = 0,01
halving time = 1



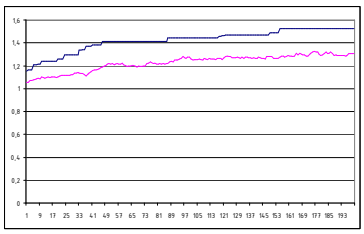
starting mutation rate = 0,01
halving time = 10



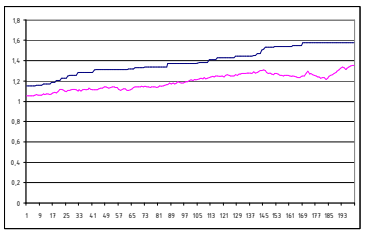
starting mutation rate = 0,01
halving time = 50



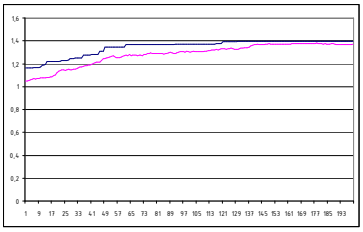
starting mutation rate = 0,01
halving time = 100



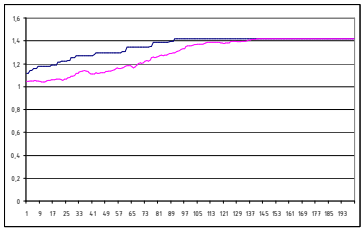
starting mutation rate = 0,01
halving time = 500



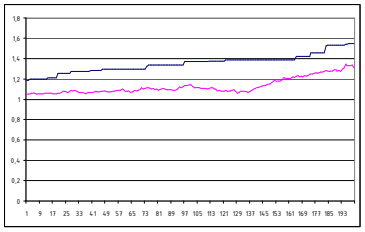
starting mutation rate = 0,01
halving time = 5000



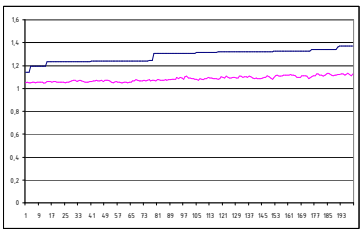
starting mutation rate = 0,1
halving time = 1



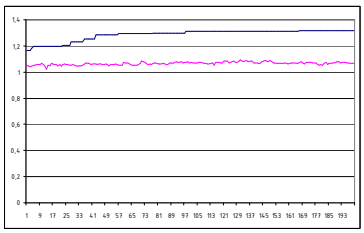
starting mutation rate = 0,1
halving time = 10



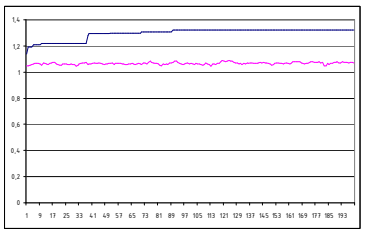
starting mutation rate = 0,1
halving time = 50



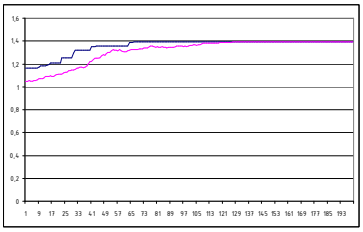
starting mutation rate = 0,1
halving time = 100



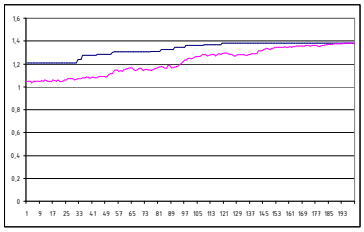
starting mutation rate = 0,1
halving time = 500



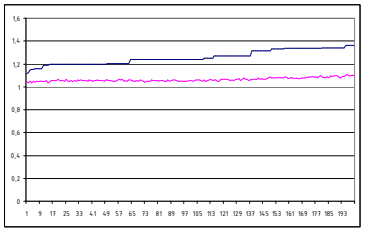
starting mutation rate = 0,1
halving time = 5000



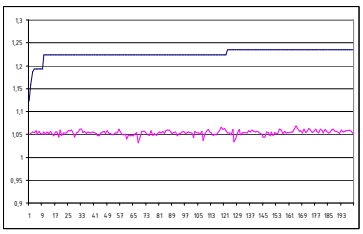
starting mutation rate = 0,5
halving time = 1



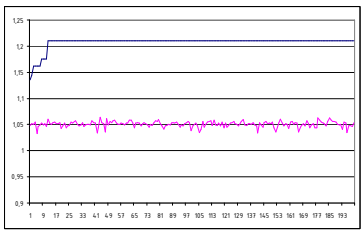
starting mutation rate = 0,5
halving time = 10



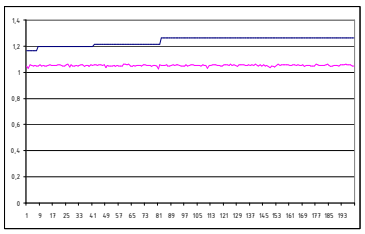
starting mutation rate = 0,5
halving time = 50



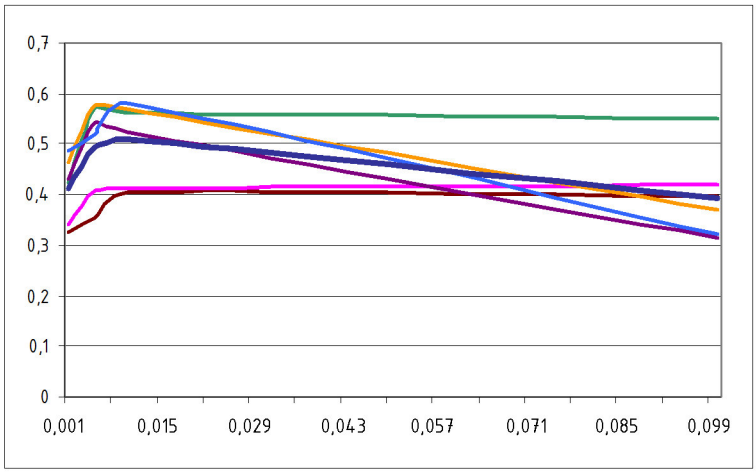
starting mutation rate = 0,5
halving time = 100



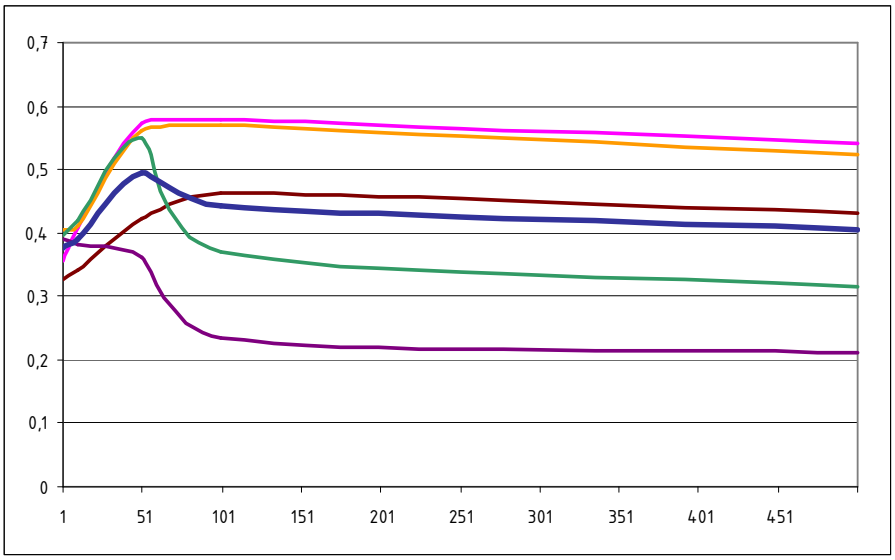
starting mutation rate = 0,5
halving time = 500



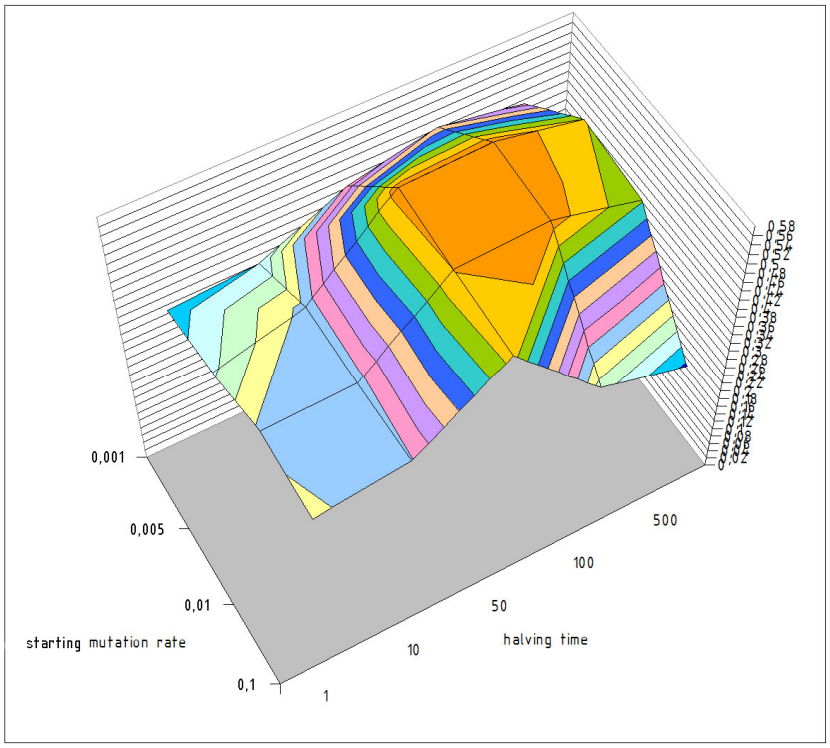
starting mutation rate = 0,5
halving time = 5000



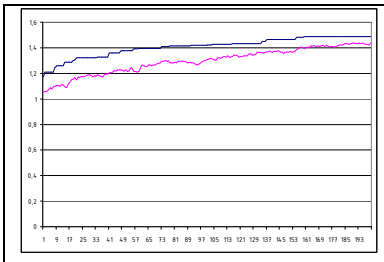

average value starting mutation rate tests: average – blue line



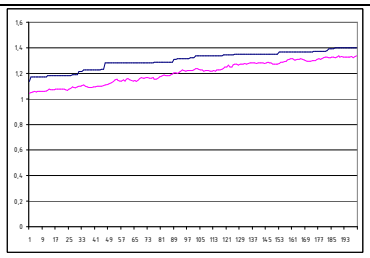

average value halving time tests: average – blue line



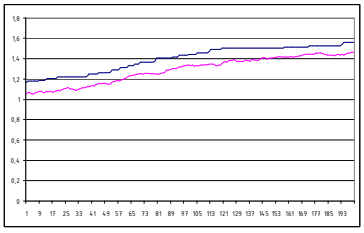

3D results combining the starting mutation rate and the halving time parameter values (not scaled)



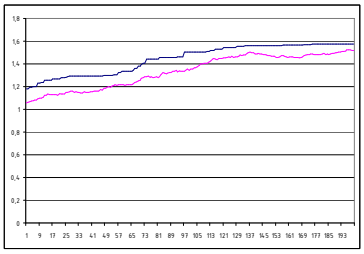
starting mutation rate = 0,0075
halving time = 50



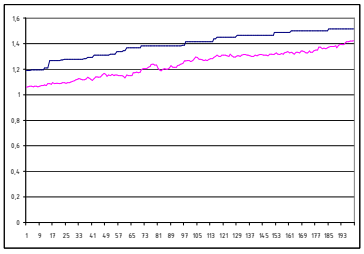
starting mutation rate = 0,0075
halving time = 50



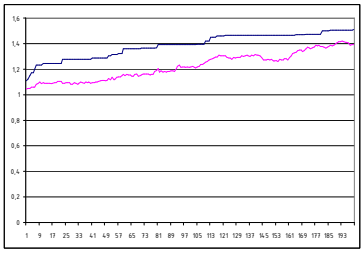
starting mutation rate = 0,010
halving time = 50



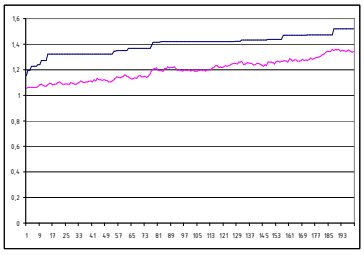
starting mutation rate = 0,010
halving time = 50



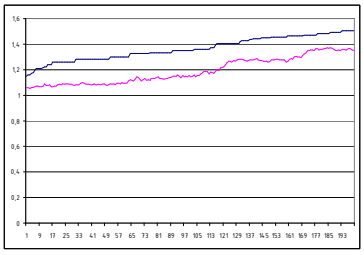
starting mutation rate = 0,025
halving time = 50



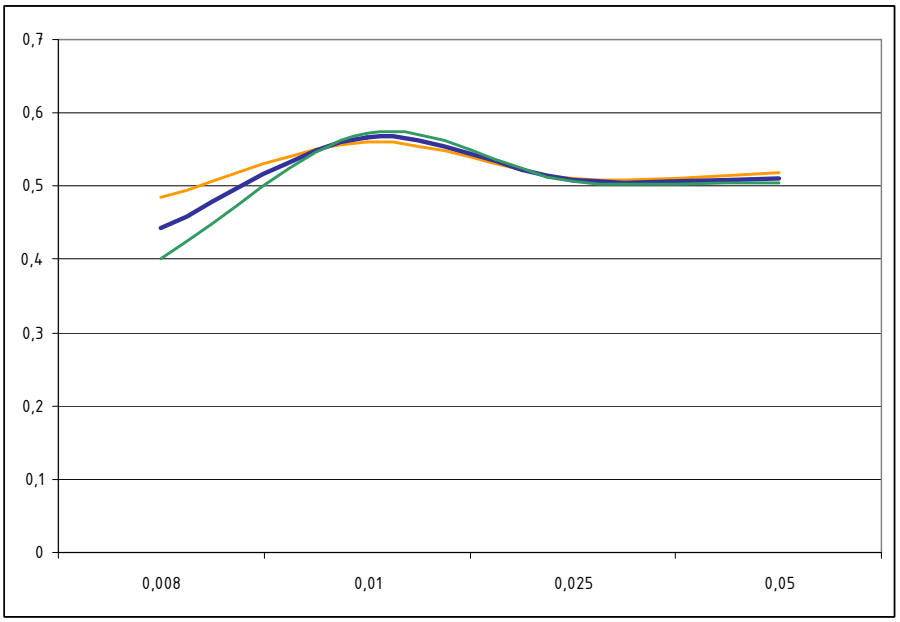
starting mutation rate = 0,025
halving time = 50



starting mutation rate = 0,050
halving time = 50



starting mutation rate = 0,050
halving time = 50




average value starting mutation rate tests: average – blue line (not scaled)