A Framework for Designing of Optimization Software Tools by Commercial API Implementation

Daniela Borissova[1,2], Ivan Mustakerov[1]

[1]Department of Information Processes and Decision Support Systems, Institute of Information and Communication Technologies – Bulgarian Academy of Sciences, Sofia, Bulgaria
[2]University of Library Studies and Information Technologies, Sofia, Bulgaria

***Abstract* — *The customized software development aims to release a software product tailored to the specific user needs. The goal of such software is to best fit the business requirements in an efficient system. This is especially true when business should perform optimization of the used resources. The existence of application programming interfaces in many commercial mathematical programming systems makes it possible to develop custom tailored tools using proven over the years effective methods and algorithms for optimization. In the current paper, a framework for designing of customized software based on the implementation of LINDO API is described. A developed prototype is used for numerical testing of real life optimization problem. The testing results show the applicability of the proposed framework for designing of effective customized optimization tools.*

Keywords* — *Customized software, optimization problems, LINDO API.

I. INTRODUCTION

Many of business problems are associated with proper using of limited resources. To ensure the successful operation of companies and be able to make informed decisions they need to rely on properly developed information processing systems [1]. In some cases this requires using of optimization models and methods to get the right solution. Nowadays, there are many commercial solver systems such as MOSEK, CPLEX, SAS/OR, XPRESS, LINDO, MATLAB, GUROBI, Excel Solver, etc. They are based on well studied and widely used mathematical methods for practical optimization problems. Using of commercial optimization software is not always the best choice for some specific applications. The goal of development of commercial optimization software is to be universal and to do all things that many different users require. The commercial solvers must solve as much as possible types of real problems and to supply analysis options needed for operation research specialists. Along with this as most of the users of the software in industry have little optimization methods backgrounds, the codes must be robust. This means detection of inconsistent input data, automatic choice of proper optimization method, automatically adjusting of optimization parameters and last but not least providing of intuitive and easy to use graphical user interface (GUI) [2]. Not to forget also, that the price for industrial use of commercial optimization software is not to be overlooked. In many cases of commercial software application the customer is paying for options that are not needed and will never be used in his practice. For such cases the development of software tools tailored to specific business needs to solve clearly identified specific problems can be an effective alternative to the use of commercial software.

During the design process of custom software is necessary to determine such structured solution that meets all of the technical and operational requirements of the specific problem. This is accompanied by series of decisions based on wide range of factors, where each of decisions influences on the quality, performance, maintainability, and overall success of the software application. The common approach of designing of software systems should comply with the user requirements, the existing infrastructure, and the business goals. In this respect, the application architecture should provide a bridge between business and technical requirements by understanding use cases, and then to find ways to implement those use cases in the software [3]. The software architecture has to identify the basic requirements influencing on the structure of the application.

The problem-oriented software applications are developed to solve a certain type of problems. That is why the applications have to provide an appropriate custom user interface that is to be able to process the input data to obtain the necessary output information [4-6]. The development of optimization software application requires to take into account several main aspects: what optimization problems will be solved and what optimization modeling will be applied; how optimization problems will be presented; what kind of methods will be used for problems solution; how results should be visualized and analyzed; what help information should be provided to diagnose the possible faults and what help will be provided for model management. The main disadvantage of problem-oriented software development is that a limited set of methods and algorithms are used and in many cases they are not thoroughly tested in contrast to commercial software development. The better approach would be use of commercial optimization methods and algorithms in custom oriented software with appropriate designed user interface. This approach can be realized by using of the so called *Application Programming Interface* (API) that is available for many commercial optimization systems. The existence of APIs that provide access to libraries of methods and algorithms proved over years of usage improves the productivity of software development and is one of the important factors contributing to the success of custom tailored software development [7, 8].

The current work describes a framework that assisting the process of development of customized software tool for optimization. This framework is based on LINDO API because of the ability of LINDO API to solve wide range of optimization problems, including linear programs, mixed integer programs, quadratic programs, general nonlinear non-convex programs and so on [9]. A real life nonlinear mixed integer programming problem described in [10] is used to illustrate the practical implementation of the proposed framework for development of custom oriented optimization software.

II. MATHEMATICAL MODELING

No business can be considered successful if it does not perform optimization of the used resources. Mathematically reasoned optimization requires proper mathematical modeling of the problems. The quality of the resulting solution depends on the completeness of the model in representing the real business problem [11]. Despite of availability of various modeling approaches for business problems, many of mathematical models aimed to optimize a specific objective function subject to set of constraints – single objective optimization (SOO). On the other hand, the real life problems actually require simultaneous optimization of multiple objectives and lead to multi-objective optimization (MOO).

Nevertheless the presence of different methods for MOO all of them rely on some transformation to a single objective optimization [12]. Development of the most suitable optimization model could be an iterative process that requires making of modifications of initial optimization model. The steps of a generalized modeling process can be described as:

1) developing of an initial SOO or MOO model;
2) determining of optimal SOO solution or Pareto-optimal solution for MOO;
3) identification, whether as a result of the decision of the optimization model expected performance indicators are satisfied;
4) if this is not true – modification of the initial model (objective/s, variables, constraints) in accordance to the nature of the chosen mathematical modeling approach;
5) repeating of the steps until a satisfying of required performance indicators solution is reached.

This iterative process of modeling is visualized in Fig. 1.

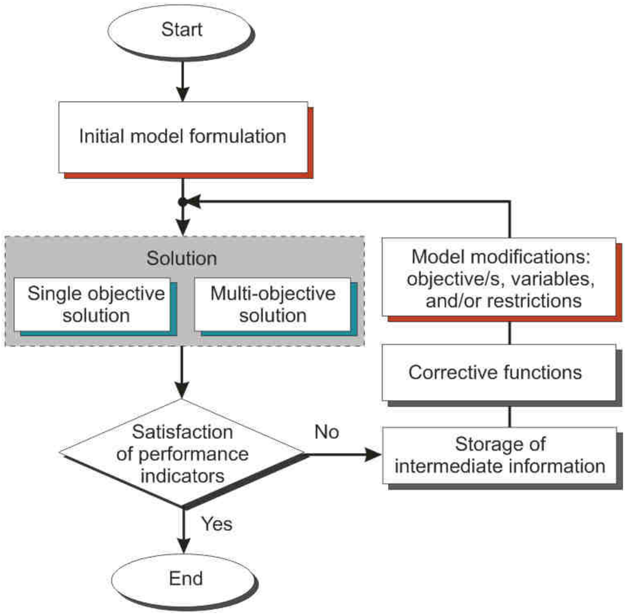

Fig. 1: Iterative modeling process

During the search for a better solution, the intermediate information is stored and used to improve the initial model. Some corrective functions can be involved to refine the model. In many cases, the corrective functions contribute not only for approximation to the optimal solution but also reduce computational effort to solve complex real life optimization problems.

III. FRAMEWORK FOR DEVELOPMENT OF CUSTOM OPTIMIZATION SOFTWARE WITH LINDO API IMPLEMENTATION

Application programming interfaces available for many commercial optimization systems provide set of functions and subroutines definitions, and input/output protocols needed to use them. APIs are valuable tools for building custom software oriented to specific applications. They improve code reuse, help to reduce development costs and promote programmers' productivity. In general, APIs make software developing easier by providing ready to use and tested building blocks, which can be used by the programmer. APIs help using of advanced technologies in applied software and developers can concentrate its efforts primarily on the development of an appropriate graphical user interface, which is an essential feature of custom tailored applications. This is especially important when designing software tools for optimization in industry. It is unlikely to expect users of these tools to be highly trained specialists in operations research and optimization methods. It would be best if the optimization algorithms work is hidden behind the corresponding GUI. It is also important to provide an intuitive entering of the input data and understandable decision data and analysis for managers with little experience in mathematical programming.

Following these principles, in the current paper a framework for development of optimization software tools using commercial API implementation is described. It is based on LINDO API which is available for high level programming languages as C/C++, C#, Delphi, Fortran 90, Visual Basic, Java/J+ under Windows, Linux and Solaris operating systems. It also provides interfaces to other systems such as MATLAB, Ox, .NET. The LINDO solvers for mathematical programming models are proven over more than 30 years usage as effective and easy to use tools for large scale real life problems. LINDO API provides a programming environment, where models can be entered in two ways: 1) by data structures in array (vector) representations and by 2) data text files in MPS, LINDO or MPI formats [9].

When describing optimization model the following information should be provided to the solver system: 1) maximizing or minimizing optimization direction; 2) coefficients of objective function; 3) coefficients of constraints matrix; 4) constraint senses; 5) right-hand side values of constraints and 6) upper and lower bounds of variables. Many real problems formulated as optimization tasks are characterized by a large number of constraints and variables, but with small number of non-zero coefficients of the restrictions matrix (sparse constraints matrix). In such cases it is possible to represent all of the constraints matrix data by single vector. Taking in to account the large number of zero coefficients this will require considerable amount of storage. LINDO API provides an efficient way to reduce storage requirements for sparse coefficient matrix of the model by three vectors used to store information only about non-zero elements of coefficients matrix: 1) *value vector* containing all non-zero coefficients of restriction matrix ordered by columns; 2) *column-start vector* recording which points in the *value vector* represent start of columns; 3) *row-index vector* storing information about the row entries of the *value vector* points. Using of these vectors allows uniquely recovering of the original matrix of constraints' coefficients.

Another possibility is using of optional fourth vector for flexible description of model when additional constraints (rows in matrix) are expected to be added in the future. This is *column-length vector* describing number of non-zeroes in each column. The *value vector* and *row-index vector* for this case are modified to contain symbols *X* to reserve space for future extensions. These symbols are ignored when solver routines process model description. This option is very useful because it allows not developing new model and software when business, which he describes, expands with new requirements or constraints.

The second approach of LINDO API for input the optimization model is by using data text files in MPS, LINDO or MPI formats [9]. MPS file format is a format developed by IBM. It is accepted as industry standard for passing linear and quadratic models from one optimization platform to another. Unlike MPS format which is column-oriented format, LINDO format is row-oriented. It is very simple and easy to use format for describing of variety of optimization problems. The MPI is another format developed by LINDO to support portability of different mathematical programming models. When input model data are entered (by array data structures or by data files) they are passed to API to create a model ready to be processed by solver system.

Considering the LINDO API specifics, the proposed generalized framework for development of custom software tools for optimization is shown in Fig. 2.

From the user point of view, the GUI is the main part of software system that aims to enhance the efficiency and ease of use based on logical design. The most common combination of elements in GUI is made of windows, icons, menus, and pointer. The provided way for interaction should be as simple and efficient as possible.

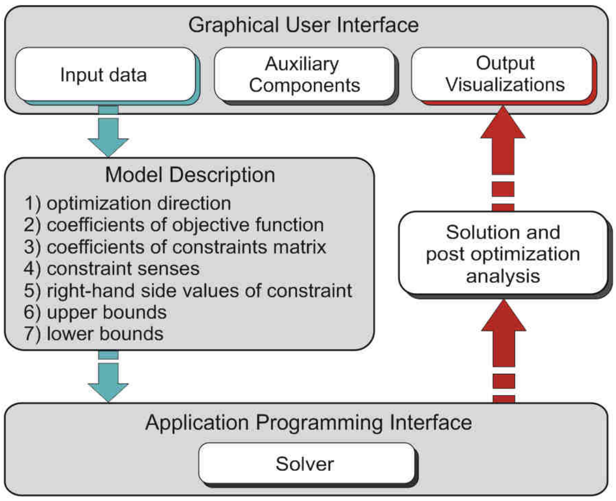

Fig. 2: A framework for development of optimization software tools using LINDO API

The GUI can be defined as fundamental platform for human-computer interaction including input and editing modules, output and visualization modules, and other auxiliary components (data base links, graphical representations, help, etc.). The main steps to be performed to use the LINDO API are shown in Fig. 3.

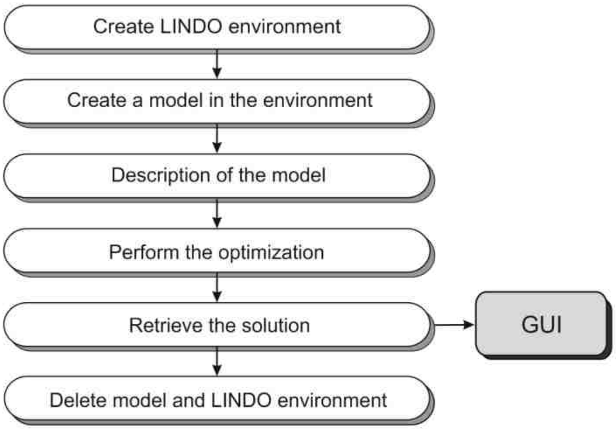

Fig.3: Main steps of LINDO API implementation

Once the model is described and passed as API object, it is solved and solution information is returned to GUI level where it is presented in format corresponding to specific user needs.

IV. DESCRIPTION OF DEVELOPED RESEARCH PROTOTYPE OF OPTIMIZATION SOFTWARE APLICATION WITH LINDO API

To illustrate the advantages of custom optimization tools based on implementation of commercial optimization system API a research prototype is developed. Proven over the years LINDO API is implemented using Java as application tool for wind farm optimization. Renewable energy sources, including wind farms consistently increased their economic attractiveness. Using them as a part of hybrid system for generating energy requires optimal use of each resource [13]. The developed research prototype is aimed to support optimal design of wind farm layout. The full description of the optimization problem can be found in [10]. Shortly, the wind farm layout is determined by choice of wind turbines number and type for known wind farm site dimensions and predominant wind direction. The wind turbines rotor diameter is crucial for determination of separation distances between turbines which in turn determine the number of turbines for given wind farm site dimensions. The practice shows that there exists the so called “wake effect” that influences on the wind farm energy production. It is a result of wind speed and turbulence changes behind turbines that causes impact of energy production of neighboring turbines. The turbine's rotor diameter is decisive for the air disturbance behind turbine. To neutralize the wake effect a proper separation distances between turbines should be determined. The goal of wind farm layout optimization is to minimize costs/energy ratio, where costs are function of the turbines number and energy is function of the chosen turbine rated power, overall number of turbines and utilization coefficient. The formulation of optimization model for this problem is:

$$\min\left(N\left(\frac{2}{3}+\frac{1}{3}e^{-0.00174\,N^2}\right)\right)\frac{1}{h_y \eta N P_{wt}} \tag{1}$$

subject to

$$N = N_{row} N_{col} \tag{2}$$

$$N_{row} = 4000 / (k_{row}D) + 1 \tag{3}$$

$$N_{col} = 1000 / (k_{col}D) + 1 \tag{4}$$

$$k_{row} \geq 1.5 \tag{5}$$

$$k_{row} \leq 3.0 \tag{6}$$

$$k_{col} \geq 9.0 \tag{7}$$

$$k_{col} \leq 11 \tag{8}$$

$$\sum_{i=1}^{10} x_i = 1, \quad x_i \in \{0,1\} \tag{9}$$

$$P_{wt} = 0.33x_1 + 0.8x_2 + 0.8x_3 + 0.85x_4 + 0.9x_5 + + 1.65x_6 + 2x_7 + 2x_8 + 2.3x_9 + 3x_{10}; \tag{10}$$

$$D = 33.4x_1 + 48x_2 + 52.9x_3 + 52x_4 + 44x_5 + + 82x_6 + 80x_7 + 82x_8 + 71x_9 + 90x_{10}; \tag{11}$$

$$SD_x = k_{col}D \tag{13}$$

$$SD_y = k_{row}D \tag{14}$$

The dimensionless costs per year are calculated as $Costs = N\left(\frac{2}{3}+\frac{1}{3}e^{-0.00174N^2}\right)$ [14, 15] and expected wind energy output per year is denoted by $Energy = h_y$□NP_{wt} [10], where h_y = 8760 hours over year, nominal utilization coefficient is taken as□□□= 0.3, N is number of installed turbines, P_{wt} represent the turbines rated power of selected type. Wind farm site is of rectangular shape with dimensions 4 km x 1 km. Coefficients k_{col} and k_{row} are used for determination of separation distances between turbines (SD_x and SD_y) to overcome the influence of wake effect. The binary integer variables x_i are decision variables used to select the best turbine type. The diameter of the selected turbine is denoted by D. The data for rated power and rotor diameter for set of 10 wind turbines types are used to design optimal wind farm layout. The main window of the developed research prototype is shown in Fig. 4.

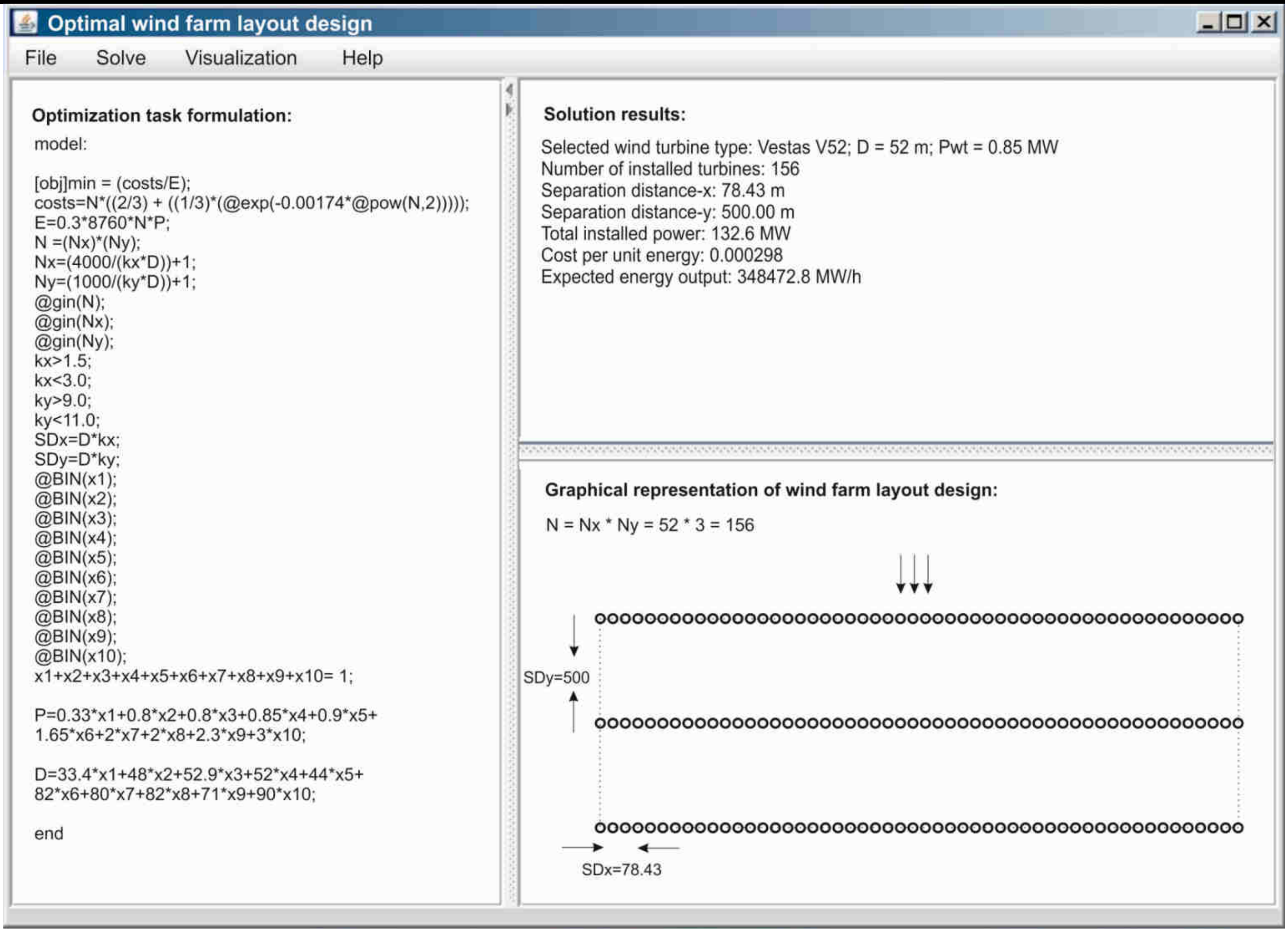

Fig. 4. Main screen of developed prototype

Left pane contains LINDO model which can be loaded from text file or directly entered in LINDO format. This pane is editable and corrections and modifications of the model are possible. The right pane is divided on two sections showing numerical results of solution in above section and graphical representation of optimal wind farm layout in lower section. Both of numerical solution results and image layout can be saved for later analysis.

V. RESULTS AND ANALYSIS

The results for the described nonlinear mixed integer optimization model (1) – (14) are shown in Fig. 4. The solution of the described optimization model via developed software prototype coincides with the solution obtained by commercial software system LINGO v. 11 [10, 16]. In both cases, the values of the objective function and variables and solution times are identical. This proves the correct implementation of LINDO API for developing of problem oriented software tool for wind farm layout optimal design.

VI. CONCLUSION

The development of custom software aims to serve a specific user or group of users. In contrast to the widespread commercial software systems custom oriented software is designed to address the specific users' needs. The advantage of developing custom software is the fact that it provides just the features that are necessary for the specific user and is not bundled with options that he will probably never use (but will pay for). The risk associated with the development of custom oriented software is the dependency on the developed and used algorithms and the extent to which they have been tested in practice. Using of commercial APIs for developing custom software allows integrating of libraries of methods and algorithms that are proven over the years by many users. This is especially important when complex mathematical problems are to be solved. Errorless processing of the data is extremely important for development of optimization tools for practical applications. Using of API allows the developers to concentrate over design of the required input and output components in proper graphical user interface and contributes to success of custom tailored software tools.

The proposed framework for development of optimization software using LINDO API could be customized for other applications which need solution of optimization problems. This framework can be considered also as blueprint for individual development of different optimization applications.

REFERENCES

[1] M. Chergui, H. Nahla, A. Chakir, S. Elhassnaoui, Y. Sekhara and H. Medromi. "Empirical study: Moroccan information systems specificities for better IT governance". International Journal of Advanced Engineering, Management and Science, vol. 2(5), 2016, pp. 391-396.

[2] H. Thomas, M. Zhou and U. Schramm. "Issues of commercial optimization software development". Structural and Multidisciplinary Optimization, vol. 23(2), 2002, pp 97-110.

[3] Microsoft Patterns & Practices Team. Microsoft® Application Architecture Guide (Patterns & Practices), 2nd Edition, 2009, 560 pages.

[4] I. Xie and K. K. Matusiak. Discover digital libraries: Theory and practice, Chapter 7 – Interface design and evaluation, Elsevier Science Publishing Co Inc., 2016, pp. 205-230.

[5] D. Borissova and I. Mustakerov. "Web-based tool for preliminary assessment of wind power plant design". In Proc. of 4th Int. Conf. Information Systems and Technologies, March 22-24, 2014, Valencia, Spain, pp. 139-149.

[6] W. Nagel. Multiscreen UX Design: Developing for a Multitude of Devices, Chapter 7 – Content design and user interface architecture for multiscreen projects: methods and rules for the conception, design, and implementation of layout, contents, and workflows in the building block principle. Morgan Kaufmann, 2016, pp.247-268.

[7] H. Niu, I. Keivanloo and Y. Zou. "API usage pattern recommendation for software development". The Journal of Systems and Software, 2016, http://dx.doi.org/10.1016/j.jss.2016.07.026.

[8] K. Jezek, J. Dietrich and P. Brada. "How Java APIs break – An empirical study". Information and Software Technology, vol. 65, 2015, pp. 129-146.

[9] LINDO API 6.1, User's Manual, 2010, 647 pages.

[10] I. Mustakerov and D. Borissova. "Wind turbines type and number choice using combinatorial optimization". Renewable Energy, vol. 35(9), 2010, pp. 1887-1894.

[11] H. A. Taha. Operations Research: An Introduction, 8th Edition, 2007, 832 pages.

[12] R. T. Marler and J. S. Arora. "Survey of multi-objective optimization methods for engineering". Structural and Multidisciplinary Optimization, vol. 26, 2004, pp. 369-395.

[13] F. Z. Kadda, S. Zouggar and M. El Hafyani. "Optimization of the managed electrical energy within a hybrid renewable energy system". International Journal of Advanced Engineering, Management and Science, vol. 2(6), 2016, pp. 588-594.

[14] G. Marmidis, S. Lazarou and E. Pyrgioti. "Optimal placement of wind turbines in a wind park using Monte Carlo simulation". Renewable Energy, vol. 7, 2008, pp. 1455-1460.

[15] R. Shakoor, M. Y. Hassan, A. Raheem and Y.-K. Wu. "Wake effect modeling: A review of wind farm layout optimization using Jensen's model". Renewable and Sustainable Energy Reviews, vol. 58, 2016, pp. 1048-1059.

[16] LINGO v. 11, LINDO™ software products, http://www.lindo.com/