Université Joseph Fourier GRENOBLE

Grenoble INP

Master in Systems, Control and Information Technologies (MiSCIT)

Specialty: Control and Systems Theory (CST)

A Derivative-free approach in Constrained Global Optimization of Dynamic Systems in Preliminary Design Phase using NOMAD

Kaveh BABANEZHAD

24 June 2013

Internship performed at G-SCOP

Laboratoire des Sciences pour la Conception, l'Optimisation et la Production de Grenoble

Under the supervision of:

Jean BIGEON, Directeur de recherche au CNRS

Khaled HADJ-HAMOU, Enseignant chercheur au Grenoble INP

[February/July] **2013**

This work was funded by the Labex **PERSYVAL-lab**[1] via the program called "Investissement d'Avenir[2]", which was launched by the French government and implemented by the ANR.

[1] http://persyval-lab.org/
[2] http://investissement-avenir.gouvernement.fr/

Acknowledgement

I would like to thank all of the people listed below for their helpful advice and continuous assistance throughout the conduct of this work:

Dr. Jean BIGEON

Dr. Khaled HADJ-HAMOU

Dr. Emmanuel WITRANT

Issam MAZHOUD

Jonathan CURRIE

I am also very grateful to PERSYAL-Lab and G-SCOP laboratory for providing me with the opportunity to perform this internship.

Abstract

Analytical models are widely used in engineering design and particularly during preliminary design phase of an engineering design process. Numerical optimization methods often come into play at this phase to improve these models under certain constraints. Amongst these methods derivative-free optimization (DFO) approaches are gaining popularity for solving optimization problems in various practical contexts especially when problems happen to be non-differentiable, non-linear and non-convex. During this work the basics of DFO have been explored and a particular software package called NOMAD has been studied both in theory and practice.

Keywords: DFO, NOMAD, GNLP, NLP, derivative-free, numerical optimization, global optimization

Contents

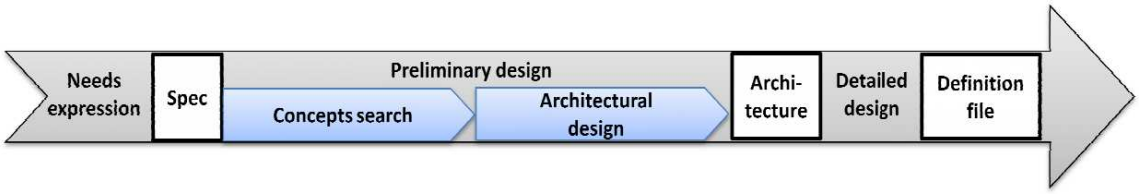
List of Figures

List of Tables

1 - Introduction and Outline of the Internship

A engineering design process is the conduct of a plan to help build a product with a specified performance goal within its associated standards. This process involves a number of steps, and often parts of the process may need to be repeated many times before manufacturing of a final design can begin. These steps can include research, conceptualization, feasibility assessment, establishing design requirements, preliminary design, detailed design, production planning, and finally production. Based on (1) most of these steps can be categorized in three principal stages: conceptual design, preliminary design and detailed design. A summarized illustration of the three stages is depicted in Fig. 1 below:

Fig. 1 – Product design process (1)



Amongst these three stages, preliminary design bridges the gap between concepts and the detailed design blueprints. In this stage, the overall system configuration is defined as precise as possible, and schematics, diagrams, and layouts of the project will provide a framework to build the project on. This stage is also particularly important as it conditions 80% to 90% of the lifecycle cost breakdown (2). The output of this stage is an analytical model that represents the achieved framework and is used to propose a first quantification of design parameters.

One part of the aforementioned analytical models is the mathematical formulations that describe the underlying physical laws associated with the product and often are the result of the convergence of different engineering fields. It almost remains intact during the rest of the steps and is referred to as the *physico-mathematical* part. The other part, referred to as *specifications*, is more dynamic compared to part one and consists of requirements set by customer or marketing departments to ensure end-user satisfaction and can include characteristics such as performance, cost, life-cycle, etc. Computer scientists and applied mathematics experts combine the physico-mathematical and specification parts to propose a unified analytical model. An illustration of how the steps of reaching this unified model come about is depicted in Fig. 2.

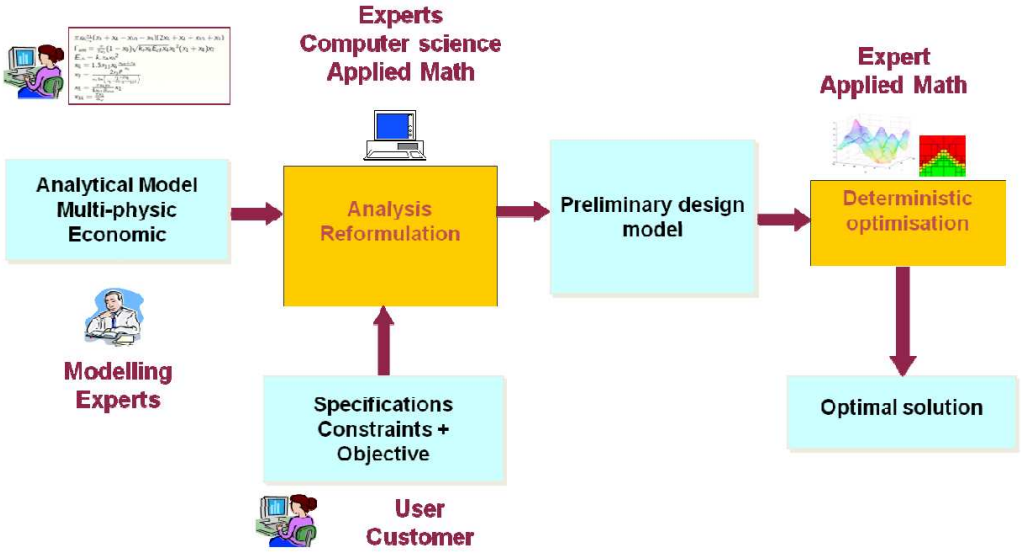
The analytical models of the preliminary design phase are often formulated as follows:

$$\begin{cases} \min\limits_{x \in X \subseteq \mathbb{R}^n} f(x) \\ g_i(x) \leq 0 \ \ \forall i \in \{1, \dots, m\} \\ h_j(x) = 0 \ \ \forall j \in \{1, \dots, q\} \end{cases}$$

Where X is a hypercube representing the bound constraints for parameters, x includes all of the variables belonging to X, f is the function that is subject to optimization (objective

function), $G = \{g_i, i \in \{1, \dots, m\}\}$ is the set of inequality constraints, and $H = \{h_j,\ j \in \{1, \dots, q\}\}$ is the set of equality constraints. The optimization problems discussed in this report are also formulated in this fashion.

Fig. 2 – The approach to make the preliminary design's analytical model (1)



In order to reach adequate results in the design of the product, analytical models generated in the preliminary design phase are often improves by undergoing numerical optimization techniques. There are many choices amongst available optimization approaches and many companies and research facilities are constantly working on creating better methods and algorithms that could improve upon previous ones or tackle special problem types that cannot be solved with previous solutions. In the same context, a number of tools, algorithms and software packages have been developed at G-SCOP to help designers with the task of optimizing their models[3] under design constraints. These solutions have been created using different optimization methods depending on the engineering problem types faced at the aforementioned laboratory. As most engineering problems and as a result analytical models are often nonlinear, non-convex, non-smooth and sometimes in a black-box format, the optimization group at G-SCOP intends to investigate a family of methods known as derivative-free methods that do not require the presence of derivative information of the objective function to operate with the focus on one particular solution called NOMAD[4].

During this internship and in accordance to the proposed objectives[5] the intern has conducted a series of bibliographical research on the subject of derivative-free methods and the NOMAD package which are available in the second and third chapter of this report. The coherence of choosing NOMAD has been tested against other available resources in a

[3] The main focus in the lab is on nonlinear black-box engineering models that are mostly obtained from industrial partners
[4] Nonlinear optimization using MADS algorithm
[5] Available as an appendix

number of benchmark problems and case studies in chapter four. And the fifth chapter is dedicated to final conclusions and suggestions.

It is worth mentioning that as the intern was new to numerical optimization's research field, a number of reports on different stochastic and deterministic numerical optimization algorithms were asked to be generated during the first one and a half month of the internship period and as they are available in appendix format, for the sake of brevity and also complying with maximum page count limits, an introduction to numerical optimization itself is removed from this report and will only be available during the presentation of the study.

2 - Derivative-free Global Optimization

2.1 - History and timeline of derivative-free optimization

Optimization problems, in which the derivatives cannot or are very costly to be computed, are on the rise in modern complex physical, chemical, econometric measurements and engineering applications and under such circumstances a class of nonlinear optimization techniques named derivative-free optimization methods (DFO) has always been needed. Although it is not exactly known when the idea of derivative free methods for minimization was first introduced, we can say possibly that the approach of using DFO methods arose in 1960's in the form of direct search methods.

Borrowing from (3), a timeline in the history of innovations in the context of DFO algorithms is provided in Fig. 3. As it can be seen in this figure, early works appeared between 1960 and 1990. The Hooke-Jeeves and Nelder-Mead algorithms were the dominant approaches in the 1960s and 1970s, and continue to be popular. Stochastic algorithms were introduced in the 1970s and 1980s and there was relatively little theory behind the deterministic algorithms until the 1990s. A more in-depth detailed review on the historical developments of derivative free optimization methods can be found in (4).

Fig. 3 - Historical illustration of important innovations in DFO (3)

(1961) Hooke & Jeeves algorithm (5)
(1962) First simplex-based optimization algorithm (6)
(1965) Nelder-Mead simplex algorithm (7)
(1969) First use of trust-region quadratic-based models (8)

(1973) First published monograph (9)
(1975) Genetic algorithms are proposed (10)
(1979) Hit-and-run algorithms are proposed (11)

(1983) First use of simulated annealing in optimization (12)
(1989) DACE stochastic model is proposed (13)

(1991) Convergence of multi-directional search algorithms is demonstrated (14)
(1991) Implicit filtering is proposed (15)
(1993) Ideas from Lipschitzian optimization are introduced (16)
(1994) Geometry considerations for points in trust-region method (17)
(1995) Particle swarm algorithm is proposed (18), (19)
(1997) DACE surrogate model introduced (20)
(1998) First use of radial basis functions in surrogate models (21)
(1999) Introduction of multi-level coordinate search (22)

(2002) First use of augmented Lagrangian in pattern search methods (23)
(2003) Generating set nomenclature is introduced (24)
(2004) Filters (25) and simplex derivatives (26) are incorporated in pattern search
(2009) First textbook on derivative-free optimization (27)

2.2 - General properties of DFO approaches and target problems profiles

Several paths can be considered when we are faced with the problems which do not allow utilization of derivatives of the objective function which can generally be summarized in two strategies:

The first strategy is to use automatic differentiation tools (as opposed to manually coding them). Automatic differentiation is utilized to define computer programs which calculate the derivatives of a function by some procedures. These computer programs calculate the Jacobian of vector-valued functions which are from n dimensional to m dimensional Euclidean space, i.e., from $\mathbb{R}^n$ to $\mathbb{R}^m$. (If the function is scalar-valued, i.e., from $\mathbb{R}^n$ to $\mathbb{R}$, then the computer program should calculate the gradient (and Hessian) of the function.) The limitation here is that the function to be differentiated is required to be the result of a callable program which cannot be treated as a black-box. (In example, objective functions in complex engineering problems are often obtained from some simulation procedures and thus the users cannot take advantage from automatic differentiation tools.)

The second strategy is to make use of finite difference approximation of the derivatives (gradients and possibly Hessian matrices). However, in general, given the cost of evaluating the objective function, evaluating its Hessian by finite differences is much too expensive. As a result, hybrid techniques which incorporate finite differences for computing gradients in conjunction with the quasi-Newton Hessian approximation (28) have been created and proved to be helpful. Some optimization software packages perform the finite difference gradient evaluation normally. The limitation of this strategy is the additional function evaluations required in the calculation of the derivatives that may be very costly and, most importantly, finite differencing can be unreliable in the presence of noise if the differentiation step is not adapted according to the noise level.

Due to the limitations that were mentioned there are always situations where none of these strategies for obtaining the derivatives works and thus a DFO approach should be selected by the users.

If we want to discuss the general properties of a DFO approach, in the case of unconstrained optimization, derivative-free methods build a linear or quadratic model of the objective function and apply one of the fundamental approaches of continuous optimization, i.e., a trust-region or a line-search, to optimize this model (29). (While derivative based methods typically use a Taylor -based model which is an approximation of the objective function, derivative free methods use interpolation, regression or other sample-based models.)

In the case of constrained optimization which is the case with examined models in this report, the strategy of derivative-free methods is usually to apply sequential quadratic programming methods for the linearization of the constraints (30).

Based on the topics brought up in (31) and (27), one can expect to successfully address problems with DFO methods which have the following characteristics (-sometimes a limitation):

- We do not have more than, say, a hundred variables (in serial computation)
- Objective functions are reasonably smooth (~not excessively non-smooth)
- Evaluation of the function is expensive and/or computed with noise (and for which accurate finite-difference derivative estimation is prohibitive and automatic differentiation is ruled out)
- Rapid asymptotic convergence is not of primary importance
- Only a few digits of accuracy are required

Additionally, other characteristics of DFO methods that are worth mentioning include:

- In the case of hundreds of variables, using a parallel environment or exploiting problem information can be a solution
- It is usually hard to minimize non-convex functions (without derivatives)
- It is generally accepted that DFO methods have the ability to find *good* local optima
- They have a tendency to (i) go to generally low regions in the early iterations; (ii) *smooth* the function in later iterations
- Stopping criteria can be a challenge in the absence of derivatives, when the function evaluations are noisy and/or expensive

However, as a general rule suggested by many, if one can obtain clean derivatives (even if it requires considerable effort) and functions defining the problem are smooth and free of noise, one should not use derivative-free methods.

2.3 - Listing of available resources

Before starting this section it needs to be said that the idea of creating a precise taxonomy of optimization methods makes no sense mainly because there are many hybrid approaches which combine several different techniques; and partly because the area of Global Optimization is a living and breathing research discipline where many contributions come from scientists from various research areas and numerous ideas are developed by these practitioners. Taxonomies should thus be considered as just a rough sketch, as one possible way to classify optimizers. However, if required two taxonomies are proposed at Pg. 23- (32) and Pg. 32- (33) that can present a general idea of optimization algorithms. From a practical of view, an online decision three is available at (34) which can be of use when deciding to pick a solution pathway for a particular problem Furthermore, a listing of available DFO resources, inspired by (3), (34), (35) and (36) is completed and presented in Tbl. 1 below:

Tbl. 1 – A list of available DFO resources as of June 2013

Solver	URL	Lang.	Bounds	Constraints	Global / Local Opt.
ASA	ingber.com	C	Required	No	Global
BARON	minlp.com i2c2.aut.ac.nz/Wiki/OPTI	GAMS AIMMS	Required	Yes	Global
BOBYQA	Available by email: mjdp@cam.ac.uk i2c2.aut.ac.nz/Wiki/OPTI	Fortran Matlab	Required	No	Local
CMA-ES	lri.fr/~hansen/cmaesintro.html	Matlab	Optional	No	Global
COBYLA	jeannot.org/~js/code/index.en.html i2c2.aut.ac.nz/Wiki/OPTI	C Matlab	Required	Yes	Local
DAKOTA/DIRECT	cs.sandia.gov/dakota/	C++	Required	Yes	Global
DAKOTA/EA	cs.sandia.gov/dakota/	C++	Required	Yes	Global
DAKOTA/PATTERN	cs.sandia.gov/dakota/	C++	Required	Yes	Global
DAKOTA/SOLIS-WETS	cs.sandia.gov/dakota/	C++	Required	Yes	Global
DFO	projects.coin-or.org/Dfo	Fortran	Required	Yes	Global
FMINSEARCH	mathworks.com	Matlab	No	No	Local
GLOBAL	inf.u-szeged.hu/~csendes	Matlab	Required	No	Global
HOPSPACK	software.sandia.gov/trac/hopspack	C++	Optional	Yes	Global
IMFIL	ncsu.edu/~ctk/imfil.html	Matlab	Required	Yes	Global
ISRES	www.notendur.hi.is/~tpr ab-initio.mit.edu/wiki/index.php/NLopt i2c2.aut.ac.nz/Wiki/OPTI	C Matlab	Required	Yes	Global
MCS	mat.univie.ac.at/~neum/software/mcs/	Matlab	Required	No	Global
NEWUOA	Available by email: mjdp@cam.ac.uk	Fortran	No	No	Global
NEWUOA+bound constraints	ab-initio.mit.edu/wiki/index.php/NLopt i2c2.aut.ac.nz/Wiki/OPTI	Matlab	Required	Yes	Local
NOMAD	gerad.ca/nomad i2c2.aut.ac.nz/Wiki/OPTI	C++ Matlab	Optional	Yes	Global
NOMADm	gerad.ca/nomad/Abramson/nomadm.html	Matlab	Optional	Yes	Global
PRAXIS	netlib.org/opt/praxis ab-initio.mit.edu/wiki/index.php/NLopt i2c2.aut.ac.nz/Wiki/OPTI	Fortran Matlab	No Yes	No Yes	Local
PSWARM	norg.uminho.pt/aivaz/pswarm/ i2c2.aut.ac.nz/Wiki/OPTI	Matlab	Required	Linear	Global
SCIP*	scip.zib.de i2c2.aut.ac.nz/Wiki/OPTI	C++ Matlab	Optional	Yes	Global
SID-PSM	mat.uc.pt/sid-psm/	Matlab	Optional	Linear	Global
SNOBFIT	mat.univie.ac.at/~neum/software/snobfit	Matlab	Required	No	Global
TOMLAB/GLCCLUSTER	tomopt.com	Matlab	Required	Yes	Global
TOMLAB/LGO	pinterconsulting.com	Matlab	Required	Yes	Global
TOMLAB/MULTIMIN	tomopt.com	Matlab	Required	Yes	Global
TOMLAB/OQNLP	tomopt.com	Matlab	Required	Yes	Global
* SCIP internally generates first and second derivatives using CppAD					

3 - NOMAD, a Derivative-free Software Package

3.1 - Introduction

NOMAD is an open source software package which implements the Mesh Adaptive Direct Search algorithm (MADS) for deterministic optimization of black-box objective functions (with no derivative information) under general nonlinear constraints. It is coded in C++ and is available for all major operating systems. A MATLAB version is also available through OPTI Toolbox (36). More information about the history of the NOMAD project from its birth in 2000 to date can be found at (37). Apart from academia, NOMAD has also gained popularity in the industrial sector; amongst them are notable companies such as Boeing, Airbus, GM and ExxonMobil. A number of examples and applications where NOMAD has been used can also be cited in (38), (39) and (40).

3.2 – Basics of the algorithm

NOMAD is designed to solve problems that have the following form:

$$\min_{x \in \Omega} f(x)$$

Where:

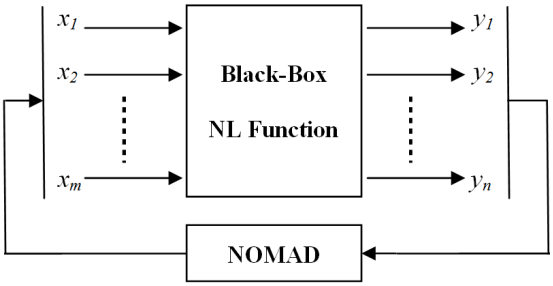
$$\begin{cases} \Omega = \{x \in X : \ c_j \leq 0, j \in J\} \subset \mathbb{R}^n \\ f, c_j : X \to \mathbb{R} \cup \{\infty\} \ \text{ for all } \ j \in J = \{1, 2, \dots, m\} \\ X \ \text{ is a subset of } \ \mathbb{R}^n \end{cases}$$

The functions f and c_j, respectively the objective function(s) and constraint(s), are defining the problem type. They usually possess no exploitable features such as derivative information, it's not possible to evaluate the function value at some trial points, they can be noisy functions and costly to evaluate. In fact this is what a black-box problem class means when using NOMAD.

NOMAD can handle bi-objective and multi-objective optimization as it will be explained later. Constraints can be in one or more of the following forms: black-boxes, nonlinear inequalities, yes/no or hidden. Variables x can be integer, binary, categorical or a mixture of them.

To illustrate what different configurations can be solved with the software package from a user's point of view, the three most common problem cases can be introduced as follows with the assumption that we are doing multi-objective optimization on a nonlinear black-box function under general nonlinear constraints in some cases:

Fig. 4 – General configuration of Form A to C



Form A:
- Bounds:
  - *lb* on x_1 to x_m: $[lbx_1, lbx_2, \ldots, lbx_m]$
  - *ub* on x_1 to x_m: $[ubx_1, ubx_2, \ldots, ubx_m]$
- NL Constraints:
  - Function(s) of a number of inputs:
    e.g. $x_1^2 + x_3^4 - x_7/x_4 * x_8 \leq 24$ or a black-box

Form B:
- Bounds:
  - *lb* on x_1 to x_m: $[lbx_1, lbx_2, \ldots, lbx_m]$
  - *ub* on x_1 to x_m: $[ubx_1, ubx_2, \ldots, ubx_m]$
  - *lb* on y_1 to y_n: $[lby_1, lby_2, \ldots, lby_m]$
  - *ub* on y_1 to y_n: $[uby_1, uby_2, \ldots, uby_m]$
- NL Constraints:
  - No constraints similar to Form A

Form C:
- Bounds:
  - *lb* on x_1 to x_m: $[lbx_1, lbx_2, \ldots, lbx_m]$
  - *ub* on x_1 to x_m: $[ubx_1, ubx_2, \ldots, ubx_m]$
  - *lb* on y_1 to y_n: $[lby_1, lby_2, \ldots, lby_m]$
  - *ub* on y_1 to y_n: $[uby_1, uby_2, \ldots, uby_m]$
- NL Constraints:
  - Function(s) of a number of inputs:
    e.g. $x_1^2 + x_3^4 - x_7/x_4 * x_8 \leq 24$ or a black-box
  - Function(s) of a number of outputs:
    e.g. $y_1^2 + y_3^4 - y_7/y_4 * y_8 \leq 26$ or a black-box

Because NOMAD implements the MADS algorithm (41) that is an extended version of Generalized Pattern Search algorithm (GPS) (42) which is itself an extension of the Coordinate Search (or Compass Search) (43), we will continue with explanation of the MADS algorithm.

MADS is an iterative method where at each iteration black-box functions are evaluated on trial points picked from a discrete mesh whose structure is defined at that iteration on the search domain by:

$$M_k = \bigcup_{x \in V_k} \{x + \Delta_k^m D_z : z \in \mathbb{N}^{n_D}\}$$

Where:

- k is the iteration number, n is the number of inputs, n_D is the number of directions
- $\Delta_k^m \in \mathbb{R}^+$ is the mesh size parameter
- V_k is the cache (previously evaluated points, V_0 ~ starting point)
- D is a $n \times n_D$ matrix representing a fixed finite set of n_D directions in $\mathbb{R}^n$

  * D is called the set of mesh directions and is constructed so that $D = GZ$ where G is a nonsingular $n \times n$ matrix and Z is a $n \times n_D$ integer matrix. D is often taken as $D = [I_n - I_n]$ where I_n is the identical matrix in dimension n.

Each MADS iteration is composed of three steps: the search, the poll and updates. During the search step values of any point on the underlying mesh can be returned, but of course, it is trying to identify a point that improves the current best solution. Since the convergence analysis does not depend on this step, a fair amount of flexibility exists at this step which can be exploited by the user to define his/her own search strategies. NOMAD has taken advantage of this fact and has introduced a Variable Neighborhood Search (VNS) (39) at this step which can help in escaping from falling into a local optima.

At the second step which is more strictly defined, the poll step, new trial mesh points in the vicinity of the best current solution are generated. In other words, it explores the mesh near the current x_k with the following set of poll trial points:

$$P_k = \{x_k + \Delta_k^m d : d \in D_k\} \subset M_k$$

Where:

- D_k is the set of poll direction

  * Each column of D_k is an integer combination of the columns of D and D_k such that its columns form a positive spanning set.

  ** Points of P_k are generated such that their distance to the poll center x_k upper bounded by the poll size parameter $\Delta_k^p \in \mathbb{R}^+$.

  *** In this algorithm we will always have $\Delta_k^m < \Delta_k^p$ and the reduction rate of Δ_k^m is greater than Δ_k^p when a failure in finding better solutions occurs.

To summarize the first two steps, trial points are generated on the mesh. Black-box functions are evaluated at these points and the algorithm determines if each evaluation is a success or not (meaning whether we have found points with lower function values that also satisfy the constraints or not) and if a success hasn't happened the evaluation process will continue until the last trial point (all of these points are stored in cache).

At the last step, updates, the cumulative success or failure of all the evaluations of that iteration is determined by checking whether a better solution was found or not and then the next iterate x_{k+1} is chosen. Based on this success or failure, the mesh size parameter is also updated with:

$$\Delta_{k+1}^{m} = \tau^{\omega_k} \Delta_k^m$$

Where:

- $\tau > 1$ is a fixed rational number
- ω_k is a finite integer, positive or null if iteration k is success, otherwise strictly negative

The poll size parameter is also updated in this step based on the implemented predefined rules which will be described later. However, as an example if the LT-MADS method is chosen we would have $\Delta_k^m \leq \Delta_k^p \leq 1$ and $\Delta_k^p = \sqrt{\Delta_k^m}$.

In the end of step three the cache (V_{k+1}) and iteration counter are also updated.

A graphical summary of the algorithm is presented in Fig. 5 and Fig. 6 illustrates the relation between mesh size parameter and poll size parameter during three consecutive iterations.

Fig. 5 - High-level presentation of the MADS algorithm (44)

Initialization:

Set the iteration counter, mesh and poll size parameter to zero $\Delta_k^m, \Delta_k^p, k \leftarrow 0$
Load the initial point $x_0 \in V_0$
Let G and Z be the matrices used to define D and τ the rational number used to update the mesh size parameter

Mein Loop:

repeat

SEARCH on the mesh to find a better solution than x_k

if the *SEARCH* failed then
  POLL on the mesh to find a better solution than x_k

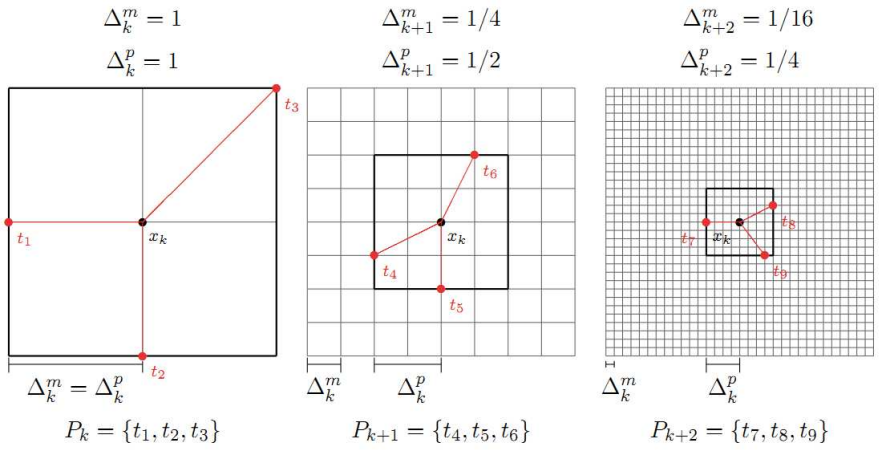
if a better solution than x_k was found by either *SEARCH* or *POLL* then
  call it x_{k+1} and coarsen the mesh

else
  set $x_{k+1} = x_k$ and refine the mesh

UPDATE parameters and set $k \leftarrow k + 1$

until stopping criteria is satisfied

Fig. 6 – Relation between Δ_k^m and Δ_k^p during three consecutive iterations (45)



* In this example $n = 2$. Thin lines represent the mesh of size Δ_k^m and thick lines the points at distance Δ_k^p from x_k in norm L_∞. Poll trial points are generated by random LT-MADS $n + 1$ directions. Poll trial points lie at the intersection between the thin and thick lines.

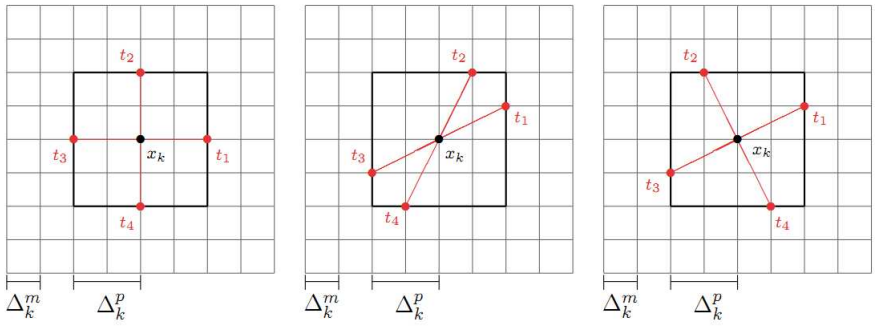
A number of studies have been conducted on the convergence analysis of the MADS algorithm which can be reached at (46), (41), (47) and (48). Details on this subject will not be further discussed in this section.

3.3 - NOMAD's extensions to the MADS algorithm

Extensions to the MADS algorithm that exists in NOMAD can be summed up in the following categories: Polling strategies, Treating constraints, Multi-objective optimization, Parallelizing, Surrogate functions and Other features. Each of these will be shortly introduced below.

Polling strategies: At the polling step of the algorithm, the poll direction is chosen based on the method predefined by default or the user. As the NOMAD software implements both GPS and MADS algorithms, the underlying methods of choosing directions in these two algorithms is also coded and available within the NOMAD source. The method of choosing directions in GPS is simply referred to as GPS and MADS offers two methods called LT-MADS directions defined in detail at (41) and OrthoMADS directions introduced in (49). The differences between the three methods is illustrated in Fig. 7 below. The efficiency of OrthoMADS has made this method the default direction type in the NOMAD package. However, the user can still switch back to the other two methods should it be required.

Fig. 7 – Different choices of direction types in NOMAD (45)



* In this example $n = 2$. Thin lines represent the mesh of size Δ_k^m and thick lines the points at distance Δ_k^p from x_k in norm L_∞. Poll trial points lie at the intersection between the thin and thick lines. In these three cases $\Delta_k^p = 2\Delta_k^m$ and $2n$ directions are considered. GPS directions are orthogonal but always the coordinate directions are chosen. In the case of the other two, directions can be chosen from any of the intersections between thin and thick lines. LT-MADS directions are based on random generation of lower-triangular matrices. OrthoMADS directions are orthogonal and no randomness is involved (~ truly deterministic).

Treating constraints: As mentioned before, NOMAD can handle three types of constraints: black-boxes, nonlinear inequalities, yes/no or hidden constraints. There are a number of strategies coded in NOMAD to do this such as the Extreme Barrier approach (EB) (41), filter technique (50), Progressive Barrier approach (PB) (51), and a hybrid method consisted of PB and EB called Progressive-to-Extreme Barrier approach (PEB) (40). The latter is currently the default setting in the software.

Multi-objective optimization: Currently NOMAD can perform bi-objective optimization and based on (44) multi-objective optimization is going to be available in future versions. Bi-objective optimization in NOMAD has been enabled by using the BiMADS algorithm (52) which in basic terms consist of initiating a series of normal single-objective MADS evaluations

Parallelizing: NOMAD is at its best capable of handling problems with less than 50 variables. However, with parallelizing techniques (53) this limitation can be tackled. Currently there are three different methods available in the NOMAD package for parallelizing: p-MADS which is in essence evaluating in parallel lists of trial points that can be generated during the MADS algorithm. COOP-MADS which consists of running several MADS algorithms on the same problem. And PSD-MADS (54) in which random subsections of the main problem are reformulated by fixing a certain number of their variables and then distributed to different processes all of which are managed over by a master process.

In order to just increase the efficiency of solving normal problems one can use p-MADS, to deal with small to medium sized problems one can choose COOP-MADS, and to be able to solve large-scale problems (up to ≅500 variables) PSD-MADS should be implemented.

Surrogates: Surrogates are functions that are relatively similar to the target black-box function of the optimization problem but are less costly to optimize. Surrogates can be defined before the optimization process and stay rigid during the process (non-adaptive type) or can be autonomously created (or updated) before the start of the process based on past evaluations (adaptive type).

NOMAD implements non-adaptive surrogates in number ways. For example to increase the chance of finding better solutions in minimum time at the poll step (second step of the MADS algorithm), NOMAD can first evaluate the trial points on a surrogate model and sort the points based on their surrogate values before evaluating them on the actual black-box function. More information on surrogates can be found in (27) and (55).

Other features: Amongst other features that are included in the NOMAD package we can point to support for categorical variables (variables that can have only a finite and predetermined set of values) and periodic (or cyclic) variables, search strategies such as VNS, and a Mixed Variable Programming algorithm (MVP) for optimization under a mixture of different variable types.

3.4 - Using NOMAD

The NOMAD software package is available for all of the popular operating systems (Microsoft Windows XP and 7, Unix, Linux and Mac OS X) (37) and there is also a MATLAB version of it available through OPTI Toolbox (36). For the conduct of this internship the latter was chosen to enable the intern to compare the results with other solvers that are available for MATLAB.

4 - Numerical Results and Case Studies

4.1 - Benchmarking test cases

In this section the capabilities of NOMAD are tested against a number of other derivative-free solutions by trying to find the global minima of a number of well-known mathematical functions often used in benchmarking scenarios. Since at the moment OPTI Toolbox does not provide particular tools for benchmarking purposes[6], we will demonstrate the results in tabular format. A cumulative conclusion will be made in chapter five.

All of the tests have been conducted on the same machine and predefined shared parameters between different solvers have been set to values of Tbl. 2 below. Tbl 3 also provides a very short listing of the solvers that have been used in the benchmarking effort.

Tbl. 2 - Default settings of the optimization process (test cases)

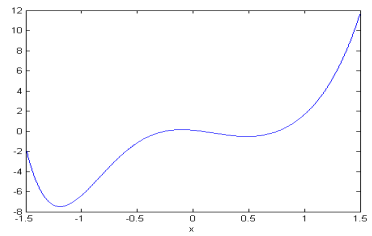
Name	Definition	Limit
maxiter	Max. number of iterations the solver can run.	1500
maxfeval	Max. number of nonlinear objective function evaluations.	10000
maxnodes	Max. nodes a mixed integer solver can explore.	10000
maxtime	Max. execution time of a solver.	1000^{Sec}
tolrfun	Relative convergence tolerance of the solver.	1.0e-07
tolafun	Absolute convergence tolerance of the solver.	1.0e-07
tolint	Absolute tolerance used to define whether a solution is an integer value.	1.0e-05

Tbl. 3 - List of the used algorithms in engineering case studies

OPTI Toolbox: www.i2c2.aut.ac.nz/Wiki/OPTI	
NOMAD	NOMAD: *Nonlinear Optimization using the MADS Algorithm* Details: www.gerad.ca/nomad
BARON	BARON: *Branch-And-Reduce Optimization Navigator* Details: www.minlp.com
SCIP	SCIP: *Spatial Branch and Bound using IPOPT and SoPlex* Details: www.scip.zib.de
ISRES	ISRES: *Improved Stochastic Ranking Evolution Strategy* One of the algorithms available in the NLOPT package within OPTI Toolbox Details: www.ab-initio.mit.edu/nlopt
Patternsearch / Global Direct Search	One of the algorithms available in MathWork's Global Optimization Toolbox (GMATLAB), also reachable through OPTI Toolbox Details: www.mathworks.com/products/global-optimization

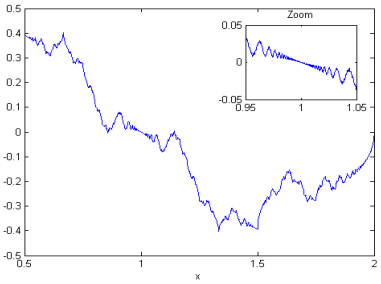
[6] There exists some new tools for benchmarking in the beta version of the next release of OPTI that the intern has received privately from the developer in late June, but existence of a number of bugs still makes it impossible to get practical dependable results out of the package

Tbl. 4 – Bench. prob., non-convex polynomial

Name: *Non-convex Polynomial*	Description of the Function
	▪ 6th order non-convex polynomial ▪ Nonlinear ▪ Bounded: $[-1.5] \times [1.5]$ ▪ Unconstrained ▪ Random initial guess

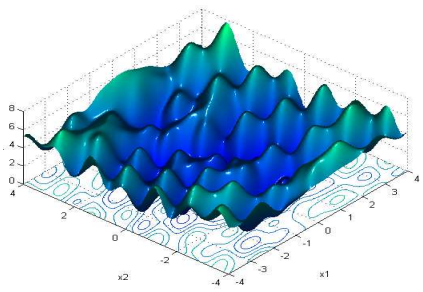
Results				
Solver / Algorithm	Status	Func. Evaluations	Time sec	Found Minima
NOMAD	Solved	50	0.205	-7.487312
BARON	Solved	NA	1.276	-7.487312
ISRES	Solved	1862	15.129	-7.487311
GMATLAB	Solved	61	1.005	-7.487312
SCIP	Solved	NA	0.651	-7.487328

Tbl. 5 - Bench. prob., single dimension Reimann

Name: *Single dimension Riemann*	Description of the Function
	▪ Non-convex ▪ Nonlinear ▪ Non-differentiable ▪ Noisy ▪ Bounded: $[0.5] \times [2]$ ▪ Unconstrained ▪ Random initial guess

Results				
Solver / Algorithm	Status	Func. Evaluations	Time sec	Found Minima
NOMAD	Solved	76	1.678	-0.392699
BARON	Solver Error	-	-	-
ISRES	Limit Exceeded	Exceeded (1e+4)	364.559	-0.399791
GMATLAB	Solved	42	2.105	-0.392699
SCIP	Solver Error	-	-	-

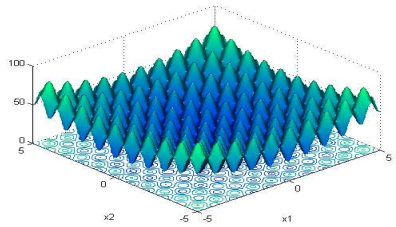
Tbl. 6 – Bench. prob., Wolfram global function

Name: *Wolfram Global Function*	Description of the Function
	▪ Non-convex ▪ Nonlinear ▪ Multiple global minima ▪ Multiple local minima ▪ Bounded: $[-4; -4] \times [4; 4]$ ▪ Arbitrary NL constraints ▪ Random initial guess

Results

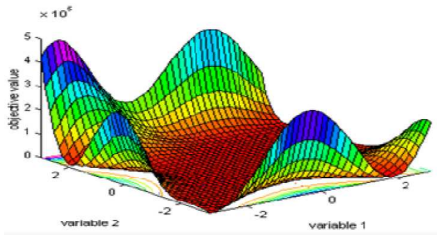
Solver / Algorithm	Status	Func. Evaluations	Time [sec]	Found Minima
NOMAD	Solved	139	0. 597	0.072875
BARON	Solver Error	-	-	-
ISRES	Limit Exceeded	Exceeded (1e+4)	82.250	0.196233
GMATLAB	Solved	261	1.421176	0.178509
SCIP	Solver Error	-	-	-

Tbl. 7 – Bench. prob., Rastrigin function

Name: *Rastrigin Function*	Description of the Function
	▪ Non-convex ▪ Nonlinear ▪ Bounded: $[5\pi; -20\pi] \times [20\pi; -4\pi]$ ▪ Arbitrary NL constraints ▪ Random initial guess

Results				
Solver / Algorithm	Status	Func. Evaluations	Time [sec]	Found Minima
NOMAD	Solved	169	0.598	4.228152
BARON	Solver Error	-	-	-
ISRES	Solved	6186	68.065	4.228154
GMATLAB	Solved	176	1.144	4.228152
SCIP	Solver Error	-	-	-

Tbl. 8 – Bench. prob., Goldstein price function

Name: *Goldstein Price Function*	Description of the Function
	▪ Non-convex ▪ Nonlinear ▪ Bounded: $[-2; -2] \times [2; 2]$ ▪ Arbitrary NL constraints ▪ Random initial guess

Results				
Solver / Algorithm	Status	Func. Evaluations	Time [sec]	Found Minima
NOMAD	Solved	169	0.598	4.228152
BARON	Solver Error	-	-	-
ISRES	Solved	6186	68.065	4.228154
GMATLAB	Solved	176	1.144	4.228152
SCIP	Solver Error	-	-	-

4.2 - Electromechanical actuator case study

4.2.1 - Model representation

The subject of this case study is a slot-less rectangular waveform permanent magnet electrical machine. The design model, established from simplified electromagnetic modeling, corresponds to various equations ensuring the flux conservation in the magnetic circuit, the electromechanical conversion and the overall heating up due to the losses in the stator by Joule effect. This model is one of the engineering case studies that have been used for benchmarking and evaluation purposes in a number of G-SCOP's scientific publications such as (56), (57) and (1). The original analytical model, extracted from (58), contains equations that sum up a number of nonlinear equality constraints from several focus fields such as electrical, mechanical, magnetic and heat. We will define a constrained optimization problem based on this machine that will include 6 nonlinear equality constraints and 19 variables in total (from which 10 will be bounds and 9 will be design parameters). This problem will then be tackled by NOMAD and a number of other solvers within OPTI Toolbox and the results will be compared both with each other and the optimized values that were obtained in (58). An illustration of the machine's structure (referred to as MAPSE) is available in Fig. 8 below:

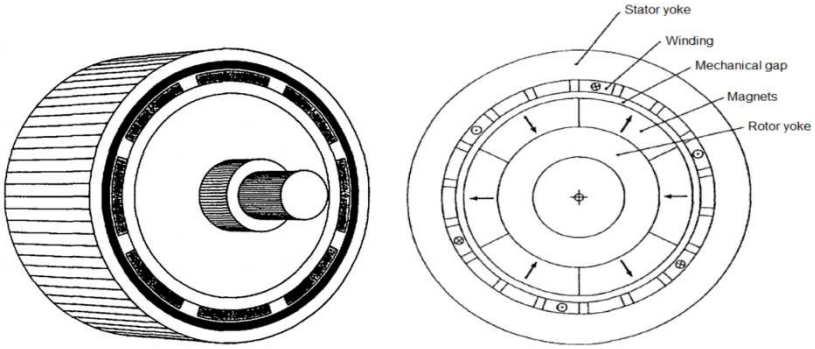


Fig. 8 - MAPSE structure section-view (58)

The definitions of the variables and parameters that are present in the analytical model of this machine are presented in Tbl. 9. The governing equations of the machine are presented in Tbl. 10.

Tbl. 9 - Variables and design parameters of the model

Definition	Type	Symbol	Range/Value
Winding thickness	Variable	E (m)	[0.001 0.05]
Thickness of the permanent magnet	Variable	l_a (m)	[0.001 0.05]
Polar arc factor	Variable	β	[0.8 1.0]
Thickness of the mechanical air gap	Variable	e (m)	[0.0001 0.005]
Length ratio	Variable	λ	[1.0 2.5]
Current areal density	Variable	J_{cu} (A/m2)	[1.0e+5 1.0e+7]
Bore diameter	Variable	D (m)	[0.01 0.5]
Magnetic field in the air gap	Variable	B_e (T)	[0.1 1.0]

Semi-empiric magnetic leakage coefficient	Variable	K_f	[0.01 0.5]
Thickness of the yoke	Variable	C (m)	[0.001 0.05]
Body length	Variable	L (m)	[0.004 0.5]
Number of pole pairs	Parameter	p	[1.0 10]
Polar step	Parameter	Δ_p (m)	= = 0.1
Coil winding filling factor	Parameter	k_r	= = 0.7
Magnetic field in the iron	Parameter	B_{fer} (T)	= = 1.5
Machine warm-up	Parameter	E_{ch} (A/m)	= = 1.0e+11
Electromagnetic torque	Parameter	Γ_{em} (N.m)	= = 10
Magnetic polarization	Parameter	M (T)	= = 0.9
Specific resistance of the copper	Parameter	ρ_{cu} (Ω.m)	= = 1.8e-8

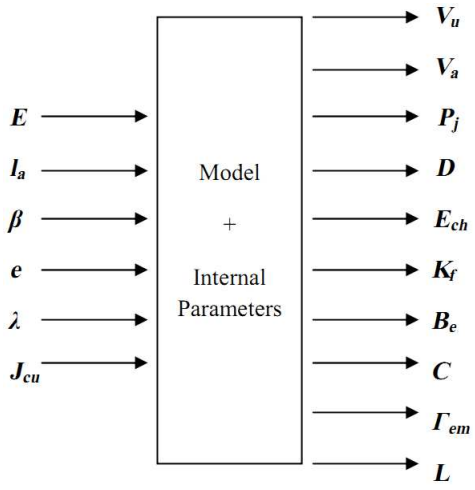
Tbl. 10 - Governing equations of the machine

Definition	Formula
Electromagnetic torque	$\Gamma_{em} = \frac{\pi}{2\lambda}(1 - K_f)\sqrt{k_r \beta E_{ch} E}\, D^2 (D + E) B_e$
Machine (global winding) warm-up	$E_{ch} = k_r E J_{cu}^2$
Semi-empiric magnetic leakage coefficient	$K_f = 1.5 p \beta \frac{e + E}{D}$
Magnetic field in the air gap	$B_e = \frac{2 l_a M}{D log[\frac{D + 2E}{D - 2(l_a + e)}]}$
Thickness of the yoke	$C = \frac{\pi \beta B_e}{4 p B_{fer}} D$
Bore diameter	$D = \frac{p \Delta_p}{\pi}$
Active parts volume	$V_u = \pi \frac{D}{\lambda}(D + E - e - l_a)(2C + E + e + l_a)$
Magnet volume	$V_a = \pi \beta l_a \frac{D}{\lambda}(D - 2e - l_a)$
Stator's Joule losses	$P_j = \pi \rho_{cu} \frac{D}{\lambda}(D + E) E_{ch}$

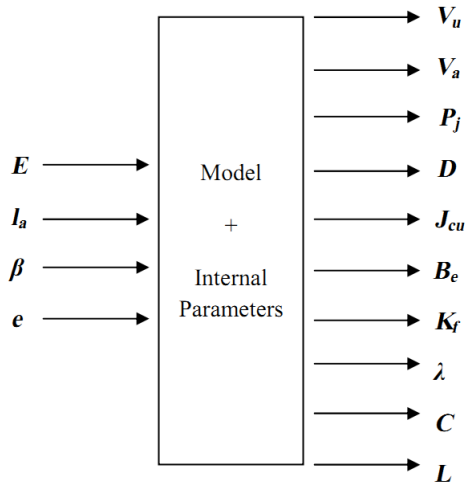
It is worth mentioning that we started the case study by the initial set of basic equations available in Tbl. 10 knowing that our objective functions for the optimization process are V_u, V_a and P_j. As the initial results were not satisfying enough a reformulation in the governing equations of the model was performed[7]. In cases where we are dealing with small scale objective functions that have an explicit form (rather than a black-box form) reformulation is a very common approach to improve the results and reduce computation times. Both final formats of the model (non-reformulated and reformulated) with explanation of inputs and outputs, boundaries and constraints are presented in Tbl. 11 and Tbl. 12.

[7] More information on effects of reformulation in optimization of an explicit problem is available as an appendix

Tbl. 11 - Non-reformulated model of the machine for optimization

Objective Function(s)	Minimize $\begin{cases} V_u = \pi \frac{D}{\lambda}(D+E-e-l_a)(2C+E+e+l_a) \\ V_a = \pi\beta l_a \frac{D}{\lambda}(D-2e-l_a) \\ P_j = \pi\rho_{cu}\frac{D}{\lambda}(D+E)E_{ch} \end{cases}$
Input / Output Scheme	
Output Constraints	With definitive values $\begin{cases} \Gamma_{em} = \frac{\pi}{2\lambda}(1-K_f)\sqrt{k_r\beta E_{ch}E}\,D^2(D+E)B_e \\ E_{ch} = k_r E J_{cu}^2 \end{cases}, \begin{cases} \Gamma_{em} == 10 \\ E_{ch} == 10^{11} \end{cases}$ Within intervals $\begin{cases} K_f = 1.5p\beta\frac{e+E}{D} \\ B_e = \frac{2l_aM}{Dlog[\frac{D+2E}{D-2(l_a+e)}]} \\ C = \frac{\pi\beta B_e}{4pB_{fer}}D \\ D = \frac{p\Delta_p}{\pi} \end{cases}, \begin{cases} K_f : [0.01\ \ 0.5] \\ B_e : [0.1\ \ 1.0] \\ C : [0.001\ \ 0.05] \\ D : [0.01\ \ 0.5] \end{cases}$
Input Constraints (Bounds)	Within intervals $\begin{cases} E : [0.001\ \ 0.05] \\ l_a : [0.001\ \ 0.5] \\ \beta : [0.8\ \ 1.0] \\ e : [0.0001\ \ 0.005] \\ \lambda : [1.0\ \ 2.5] \\ J_{cu} : [10^5\ \ 10^7] \end{cases}$
Internal Parameters	With definitive values $\begin{cases} p == 4 \\ \Delta_p == 0.1 \\ k_r == 0.7 \\ M == 0.9 \\ B_{fer} == 1.5 \\ \rho_{cu} == 1.8\times10^{-8} \end{cases}$ Internal equation $\{ L = \frac{D}{\lambda}$

Tbl. 12 - Reformulated model of the machine for optimization

Objective Function(s)	Minimize $\begin{cases} V_u = \pi \frac{D}{\lambda}(D+E-e-l_a)(2C+E+e+l_a) \\ V_a = \pi\beta l_a \frac{D}{\lambda}(D-2e-l_a) \\ P_j = \pi\rho_{cu}\frac{D}{\lambda}(D+E)E_{ch} \end{cases}$
Input / Output Scheme	
Output Constraints	Within intervals $\begin{cases} D = \frac{p\Delta_p}{\pi} \\ J_{cu} = \sqrt{\frac{E_{ch}}{k_r E}} \\ B_e = \frac{2l_a M}{Dlog[\frac{D+2E}{D-2(l_a+e)}]} \\ K_f = 1.5p\beta\frac{e+E}{D} \\ \lambda = \frac{\pi}{2\Gamma_{em}}\left(1-K_f\right)\sqrt{k_r\beta E_{ch}E}\,D^2(D+E)B_e \\ C = \frac{\pi\beta B_e}{4pB_{fer}}D \end{cases}$, $\begin{cases} D: [0.01\ \ 0.5] \\ J_{cu}: [10^5\ \ 10^7] \\ B_e: [0.1\ \ 1.0] \\ K_f: [0.01\ \ 0.5] \\ \lambda: [1.0\ \ 2.5] \\ C: [0.001\ \ 0.05] \end{cases}$
Input Constraints (Bounds)	Within intervals $\begin{cases} E: [0.001\ \ 0.05] \\ l_a: [0.001\ \ 0.5] \\ \beta: [0.8\ \ 1.0] \\ e: [0.0001\ \ 0.005] \end{cases}$
Internal Parameters	With definitive values $\begin{cases} p == 4 \\ \Delta_p == 0.1 \\ k_r == 0.7 \\ M == 0.9 \\ B_{fer} == 1.5 \\ \rho_{cu} == 1.8\times10^{-8} \\ E_{ch} == 10^{11} \\ \Gamma_{em} == 10 \end{cases}$ Internal equation $\{ L = \frac{D}{\lambda}$

4.2.2 - Numerical results

The derivative-free global optimization algorithms and solvers that were used is this case study are the same as benchmark test cases and a short listing was available earlier in Tbl. 3. All of the tests have been conducted on the same machine and predefined shared parameters between different solvers have been set to also to the same values of Tbl. 2 as they were in benchmark test cases. In all tests five random initial guesses were used within boundaries.

Tbl. 13 - Nominal values for comparison of results of GNLP Optimization of the case study

Nominal Comparison Table extracted from (58)*					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073509e-04**	6.713553e-04	6.966291e-04
V_a (m3)	-	-	1.104445e-04	**6.719333e-05**	2.129067e-04
P_j (watts)	-	-	5.122298e+01	9.310473e+01	**3.826091e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.908129e-02	1.272139e-01	5.097810e-02
l_a (m)	[0.001	0.05]	5.300000e-03	1.700000e-03	1.170000e-02
E (m)	[0.001	0.05]	3.800000e-03	2.100000e-03	5.400000e-03
C (m)	[0.001	0.05]	6.159892e-03	4.227949e-03	9.070501e-03
β	[0.8	1.0]	8.000000e-01	8.000000e-01	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.619919e-01	3.170961e-01	5.442301e-01
J_{cu} (A/m2)	[10^5	10^7]	6.131393e+06	8.247861e+06	5.143445e+06
K_f	[0.01	0.5]	1.809557e-01	1.168672e-01	3.015929e-01
e (m)	[0.0001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	1.843103e+00	1.000865e+00	2.497621e+00

* Since the values published in the aforementioned paper from 1993 turned out to be not accurate enough, we only used the equations of the design and recalculated the outputs based on four inputs (E, l_a, β, e) which unfortunately had to be taken from the same paper. This has only been done for benchmarking and comparing purposes with the old method/algorithm used in the said paper.

** The authors of [1] have chosen the value of parameter *p* to be equal to *4* before the start of their optimization process.

The best obtained numerical results for minimization of V_u[8] at similar starting points are demonstrated in tabular format in Tbl. 14 and Tbl. 15 for both non-reformulated and reformulated models. Full tables of numerical results with optimized values for all of the variables (and full range of targets: V_u, V_a and P_j) are available as an appendix to this document. A cumulative conclusion will be made in chapter five.

[8] V_u is often the target that is used for comparison of results in this case study, both in the lab and published papers

Tbl. 14 - Numerical results of minimization of *Vu* using the Original Non-reformulated Model

Numerical results of minimization of V_u using the *Original Non-reformulated Model*					
Solver / Algorithm	Status	Func. Evaluations	Time [sec]	Found Minima	
NOMAD	Solved	578	25.666	V_u	4.497157e-04
BARON	Solver Error	NA	59.433	V_u	3.948752e-04
ISRES	Limit Exceeded	Exceeded (1e+4)	85.369	V_u	1.130545e-03
GMATLAB	Solved	1502	2.222	V_u	9.392908e-04
SCIP	Solver Error + Violated Constraint	NA	1.962	V_u	6.073419e-04

Tbl. 15 - Numerical results of minimization of *Vu* using the Reformulated Model

Numerical results of minimization of V_u using the *Reformulated Model*					
Solver / Algorithm	Status	Func. Evaluations	Time [sec]	Found Minima	
NOMAD	Solved	445	4.694	V_u	6.073815e-04
BARON	Solved	NA	38.445	V_u	6.073466e-04
ISRES	Solved	6265	72.135	V_u	6.118778e-04
GMATLAB	Solver Error	226	3.100	V_u	1.097014e-03
SCIP	Solver Error + Violated Constraint	NA	33.057	V_u	6.092677e-04

5 - Conclusions

During this internship a series of bibliographic research were performed around derivative-free global optimization techniques. As most of the engineering problems that the host laboratory deals with are complex nonlinear models, introduction of these methods could be of great advantage. The resulting literature of the bibliographical part is intended to help the host laboratory with further decisions on their research focuses.

One particular derivative-free solution that was of interest to the host laboratory called NOMAD was also investigated both in theory and practice. NOMAD was tested against one stochastic derivative-free algorithm (ISRES) and three deterministic derivative-free algorithms (BARON, SCIP, and GMATLAB). A portion of the tests included benchmarking on a number of standard functions during which NOMAD was the most efficient solution when dealing with non-convex nonlinear models and converged to an optima on all of the tests. Since most of the industrial models available in the host laboratory have similar features this results are promising. NOMAD was also tested on an engineering case study during which the results were as promising. NOMAD managed to remain the efficient solution and converged in both non-reformulated and reformulated cases. Although in theory NOMAD does not guarantee the convergence to a global optima, it performed if not better, equally as good as solvers that guarantee this convergence given enough computing time (such as SCIP and BARON), in all of the conducted tests.

Although NOMAD has performed better than expected so far in this study, a number of future steps can be taken during the remaining weeks of the internship:

Finishing testing on one large-scale industrial model (non-linear, non-convex) and one confidential model (non-linear, black-box, non-convex) and comparing the results with previous values that have been achieved in the host laboratory using their own custom developed commercial tools.

Exploring the possibility of improving the convergence of NOMAD to better results by coupling the final evaluation results of NOMAD with a local derivative-free solution for nonlinear non-convex models and possibly introduce that as an internal extension to the NOMAD package.

Improving the convergence of NOMAD in scenarios similar to the non-reformulated model of the electromechanical actuator case-study where NOMAD has to take into account nonlinear equalities as well. Of course equality constraints can be converted to inequalities but the accuracy of convergence to adequate results is not sufficient enough currently (since in these particular cases less than moderate tolerances have caused unexpected crashes).

Bibliography

1. **Mazhoud, I.** *Techniques d'optimisation globale pour le pre-dimensionnement en conception.* Grenoble INP. 2010.

2. **Berliner, B. J. A.** *Cost management for today's advanced manufacturing.* s.l. : Harvard Business School Press, 1988.

3. *Derivative-free optimization: a review of algorithms and comparison of software implementations.* **Rios, L. M., Sahinidis, N. V.** 2012, J Glob Optim. DOI 10.1007/s10898-012-9951-y.

4. *Recent progress in unconstrained nonlinear optimization without derivatives.* **Conn, A. R., Scheinberg, K., Toint, Ph. L.** 1997, Mathematical Programming. 79, 397-414.

5. *Direct search solution of numerical and statistical problems.* **Hooke, R., Jeeves, T.A.** 1961, J. Assoc. Comput. 212–219.

6. *Sequential application for simplex designs in optimisation and evolutionary operation.* **Spendley, W., Hext, G.R., Himsworth, F.R.** 1962, Technometrics. 441–461.

7. *A simplex method for function minimization.* **Mead, J.A. Nelder and R.** 1965, Com1puter. 308-313.

8. **Winfield, D.** *Function and FunctionalOptimization by Interpolation inData Tables.* Harvard University. 1969. PhD thesis.

9. **Brent, R.P.** *Algorithms forMinimization without Derivatives.* Englewood Cliffs : Prentice-Hall, 1973.

10. **Holland, J.H.** *Adaptation in Natural and Artificial Systems.* Ann Arbor : University of Michigan Press, 1975.

11. *Constraints' redundancy and feasible region boundedness by random feasible point generator (RFPG).* **Boneh, A., Golan, A.** Amsterdam : s.n., 1979. 3rd European Congress on Operations Research (EURO III).

12. *Optimization by simulated annealing.* **Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.** 1983, Science 220. 671–680.

13. *Design and analysis of computer experiments.* **Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.** 1989, Stat. Sci. 4. 409–423.

14. *On the convergence of multidirectional search algorithms.* **Torczon, V.J.** 1991, SIAM J. Optim. 123-145.

15. *Simulated performance optimization of gaasmesfet amplifiers.* **Winslow, T.A., Trew, R.J., Gilmore, P., Kelley, C.T.** Piscataway : IEEE/Cornell Conference on Advanced Concepts in High Speed Semiconductor Devices and Circuits, 1991. pp. 393–402.

16. *Lipschitzian optimization without the Lipschitz constant.* **Jones, D.R., Perttunen, C.D., Stuckman, B.E.** 1993, J. Optim. Theory Appl. 79. 157–181.

17. *Adirect search optimizationmethod thatmodels the objective by quadratic interpolation.* **Powell, M.J.D.** Stockholm : 5th Stockholm Optimization Days, 1994.

18. *A new optimizer using particle swarm theory.* **Eberhart, R.,Kennedy, J.** Nagoya : 6th International Symposium on Micro Machine and Human Science, 1995. pp. 39–43.

19. *Particle swarm optimization.* **Kennedy, J., Eberhart, R.** Piscataway : IEEE International Conference on Neural Networks, 1995. pp. 1942–1948.

20. **Schonlau, M.** *Computer Experiments and Global Optimization.* Department of Statistics, University of Waterloo. Waterloo : s.n., 1997. PhD thesis.

21. **Powell, M.J.D.** *Recent Research at Cambridge on Radial Basis Functions.* Dept. of Applied Mathematics and Theoretical Physics, University of Cambridge. 1998.

22. *Global optimization by multilevel coordinate search.* **Huyer, W., Neumaier, A.** 1999, J. Glob. Optim. 14. 331-355 .

23. *A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds.* **Lewis, R.M., Torczon, V.J.** 2002, SIAM J. Optim. 12. 1075–1089.

24. *Optimization by direct search: new perspectives on some classical and modern methods.* **Kolda, T.G., Lewis, R.M., Torczon, V.J.** 2003, SIAM Rev. 45. 385–482.

25. *Filter pattern search algorithms for mixed variable constrained optimization problems.* **Abramson, M.A., Audet, C., Dennis, J.E.** 2007, Pac. J. Optim. 3. 477–500.

26. *Using sampling and simplex derivatives in pattern searchmethods.* **Custodio, A.L., Vicente, L.N.** SIAM J. Optim. 18. 537–555.

27. **A. R. Conn, K. Scheinberg, and L. N. Vicente.** *Introduction to Derivative-Free Optimization.* Philadelphia : SIAM, 2009.

28. **Schnabel, J.E. Dennis and R.B.** *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Englewood Cliffs, NJ : Prentice-Hall, 1983.

29. **A.R. Conn, K. Scheinberg and L.N. Vincente.** Geometry of sample sets in derivative free optimization. Departamento de Matem´atica, Universidade de Coimbra : s.n., 2004.

30. *Sequential quadratic programming.* **Boggs, P. T., & Tolle, J. W.** 1995, Acta numerica. 4(1), 1-51..

31. *Derivative-free optimization: theory and practice, SIOPT mini-tutorial.* **Audet, C., Vicente, L. N.** Boston : s.n., 2008.

32. **Weise, T.** *Global Optimization Algorithms, Theory and Application.* 2nd. 2009. www.it-weise.de.

33. —. *Global Optimization Algorithms, Theory and Application.* 3rd. 2011. www.it-weise.de.

34. **D.Mittelmann, H.** Decision tree for optimization software. [Online] 2007. http://plato.asu.edu/guide.html.

35. NEOS Guide. [Online] http://www.neos-guide.org/.

36. **I2C2.** *OPTI Toolbox Home Page.* [Online] http://www.i2c2.aut.ac.nz/Wiki/OPTI/.

37. **Gerad.** *Gerad/NOMAD Home Page.* [Online] http://www.gerad.ca/nomad/Project/Home.html.

38. *OrthoMADS: A deterministic MADS instance with orthogonal directions.* **M. A. Abramson, C. Audet, J. E. Dennis, Jr., and S. Le Digabel.** 2009, SIAM Journal on Optimization. 20(2):948–966.

39. *Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search.* **C. Audet, V. Bechard, and S. Le Digabel.** 2008, Journal of Global Optimization. 41(2):299–318.

40. *Globalization strategies for mesh adaptive direct search.* **C. Audet, J. E. Dennis, Jr. Erratum, and S. Le Digabel.** 2010, Computational Optimization and Applications. 46(2):193–215.

41. *Mesh adaptive direct search algorithms for constrained optimization.* **Dennis, C. Audet and J. E. Dennis.** 2006, SIAM Journal on Optimization. 17(1):188–217.

42. *Analysis of generalized pattern searches.* **Dennis, C. Audet and J. E. Dennis.** 2003, SIAM Journal on Optimization. 13(3):889–903.

43. *Variable metric method for minimization.* **Davidon, W. C.** 1991, SIAM Journal on Optimization. 1(1):1–17.

44. **S. Le Digabel, C. Tribes and C. Audet.** *NOMAD User Guide v3.6.0.* 2013.

45. *Algorithm 909: NOMAD: Nonlinear Optimization with the MADS Algorithm.* **Digabel, S. Le.** New York : s.n., 2010, ACM Transactions on Mathematical Software.

46. *Convergence of mesh adaptive direct search to second-order stationary points.* **Audet, M. A. Abramson and C.** 2006, SIAM Journal on Optimization. 17(2):606–619.

47. *Mesh adaptive direct search algorithms for constrained optimization.* **C. Audet, A. L. Custodio, and J. E. Dennis.** 2008, SIAM Journal on Optimization. 18(4):1501–1503.

48. **Custodio, L. N. Vicente and A. L.** *Analysis of direct searches for discontinuous functions.* Dept. of Mathematics, Univ. Coimbra. 2010.

49. *OrthoMADS: A deterministic MADS instance with orthogonal directions.* **M. A. Abramson, C. Audet, J. E. Dennis, and S. Le Digabel.** 2009, SIAM Journal on Optimization. 20(2):948–966.

50. *A pattern Search Filter Method for Nonlinear Programming without Derivatives.* **Dennis, C. Audet and J. E.** 2004, SIAM Journal on Optimization. 14(4):980–1010.

51. *A Progressive Barrier for Derivative-Free Nonlinear Programming.* **Dennis, C. Audet and J. E.** 2009, SIAM Journal on Optimization. 20(4):445–472.

52. *Multiobjective optimization through a series of single-objective formulations.* **C. Audet, G. Savard, and W. Zghal.** 2008, SIAM Journal on Optimization. 19(1):188–210.

53. **S. Le Digabel, M. A. Abramson, C. Audet and J. E. Dennis.** Parallel Versions of the MADS Algorithmfor Black-Box Optimization. *JOPT Lecture.* Montreal : s.n., 2010.

54. *Parallel space decomposition of the mesh adaptive direct search algorithm.* **C. Audet, J. E. Dennis, and S. Le Digabel.** 2008, SIAM Journal on Optimization. 19(3):1150–1170.

55. *A rigorous framework for optimization of expensive functions by surrogates.* **A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset.** 1999, Structural Optimization. 17(1):1–13.

56. *Particle swarm optimization for solving engineering problems: A new constraint-handling mechanism.* **I. Mazhoud, K. Hadj-Hamou, J. Bigeon, and P. Joyeux.** 2013, Engineering Applications of Artificial Intelligence. 1263–1273.

57. *An improved Particle Swarm Optimization for solving constrained engineering design problems.* **A. Torkamani, K. Hadj-Hamou, and J. Bigeon.** Metz : International Conference on Industrial Engineering and Systems Management, 2011.

58. *Le dimensionnement des actionneurs électriques: un problem de programmation non linéaire.* **Kone, A. D., Nogarede, B., Lajoie Mazenc, M.** 1993, J. Phys. III France 3.

Appendix

A0 - Description and set objectives of the internship (Feb 2013)

Work Plan (as also specified in the pre-agreement and final agreement of the internship):

Analytical models are widely used in engineering design and particularly during preliminary design phases. So far a number of tools, methods and software have been developed at G-SCOP to help designers to optimize their devices under constraints (initial design data). We have been using different optimization methods depending on the engineering problem we are solving and we intend to test a new technique from the family of methods that do not use derivatives in the evaluation process of the objective functions. This particular technique/method is backed by a hierarchical non-smooth convergence analysis based on Clarke calculus and has been published in top-level scientific journals.

One implementation of this technique has been done by international colleagues in Canada and we want to establish some benchmark based on this particular method and compare it with other methods that we have already developed at G-SCOP (which include Interval Based Analysis, PSO, etc.).

The proposed work to be done during the internship period includes:

1. Completing the bibliography on the different derivative-free methods and establishing if possible some criteria for choosing more adequate methods based on engineering problem types
2. Collecting and listing available tools and libraries (to generate the code and to connect it to one library and implementing the optimization method)
3. Testing on several cases (G-SCOP will provide a number of different models)
4. In the case of non-convergent results (in other words: not obtaining the optimal results due to a number of possible issues like falling into a local optima), improving the method (in close discussion with and under guidance of the supervisor)
5. Proposing a benchmark (possibly composed of one industrial method and 2 academic method developed at G-SCOP)

A1 - Tabular numerical results of case studies

We had chosen five random starting points (initial guesses) for each of the two formulations as stated below:

Non-reformulated model's initial guess $x_0 = [E; l_a; \beta; e; \lambda; J_{cu}]$	Reformulated model's initial guess $x_0 = [E; l_a; \beta; e]$
1: [0.02; 0.03; 1; 0.005; 1.5; 1.2e+5] 2: [0.001; 0.004; 0.9; 0.003; 2.1; 9e+6] 3: [0.05; 0.001; 0.94; 0.0002; 1.05; 8.8e+5] 4: [0.015; 0.025; 0.86; 0.00015; 2.45; 5.6e+5] 5: [0.0025; 0.05; 0.88; 0.004; 1.25; 2.4e+6]	1: [0.02; 0.03; 1; 0.005] 2: [0.001; 0.004; 0.9; 0.003] 3: [0.05; 0.001; 0.94; 0.0002] 4: [0.015; 0.025; 0.86; 0.00015] 5: [0.0025; 0.05; 0.88; 0.004]

Some of the results of these evaluations (results for 1st and 2nd initial points) are presented in table formats. In the following pages, Tbl. 16.1 to 16.8 contains the results for the non-reformulated case and Tbl. 17.1 to 17.8 contains the results for the reformulated case.

Before going any further please consider the following color codes in viewing the tables:

	Title and process information bar
	Inputs
	Outputs
	Constraint violation
	Tolerable constraint violation

Tbl. 16.1 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**NOMAD****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.02; 0.03; 1; 0.005; 1.5; 1.2e+5]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**1.753032e-03**	1.753032e-03	1.759174e-03
V_a (m3)	-	-	5.913643e-04	**5.913643e-04**	5.994050e-04
P_j (watts)	-	-	6.467527e-03	6.467527e-03	**6.467293e-03**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	1.273240e-01	1.273240e-01	1.273194e-01
l_a (m)	[0.001	0.05]	1.318357e-02	1.318357e-02	1.338768e-02
E (m)	[0.001	0.05]	1.000000e-03	1.000000e-03	1.000000e-03
C (m)	[0.001	0.05]	1.160641e-02	1.160641e-02	1.160683e-02
β	[0.8	1.0]	1.000000e+00	1.000000e+00	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	6.963848e-01	6.963848e-01	6.964100e-01
J_{cu} (A/m2)	[10^5	10^7]	1.000000e+05	1.000000e+05	1.000000e+05
K_f	[0.01	0.5]	9.424778e-02	9.424778e-02	9.424778e-02
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	1.000000e+00	1.000000e+00	1.000036e+00
E_{ch} (A/m)	= = 10^11		7.000000e+06	7.000000e+06	7.000000e+06
Γ_{em} (N.m)	= = 10		1.442795e-01	1.442795e-01	1.442795e-01
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **NOMAD** Iterations: **78** FuncEvals: **670** Time: **0.272** Status: **'Infeasible'**			Solver: **NOMAD** Iterations: **78** FuncEvals: **670** Time: **0.170** Status: **'Infeasible'**		Solver: **NOMAD** Iterations: **57** FuncEvals: **519** Time: **0.241** Status: **'Infeasible '**

* NOMAD: Nonlinear Optimization using the MADS Algorithm.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.2 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**NOMAD****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.001; 0.004; 0.9; 0.003; 2.1; 9e+6]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**1.753032e-03**	1.753032e-03	1.755054e-03
V_a (m3)	-	-	5.913643e-04	**5.913643e-04**	5.940125e-04
P_j (watts)	-	-	6.467527e-03	6.467527e-03	**6.467404e-03**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	1.273240e-01	1.273240e-01	1.273215e-01
l_a (m)	[0.001	0.05]	1.318357e-02	1.318357e-02	1.325081e-02
E (m)	[0.001	0.05]	1.000000e-03	1.000000e-03	1.000000e-03
C (m)	[0.001	0.05]	1.160641e-02	1.160641e-02	1.160663e-02
β	[0.8	1.0]	1.000000e+00	1.000000e+00	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	6.963848e-01	6.963848e-01	6.963980e-01
J_{cu} (A/m2)	[10^5	10^7]	1.000000e+05	1.000000e+05	1.000000e+05
K_f	[0.01	0.5]	9.424778e-02	9.424778e-02	9.424778e-02
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	1.000000e+00	1.000000e+00	1.000019e+00
E_{ch}	= = 10^11		7.000000e+06	7.000000e+06	7.000000e+06
Γ_{em}	= = 10		1.442795e-01	1.442795e-01	1.442795e-01
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **NOMAD** Iterations: **74** FuncEvals: **659** Time: **0.279** Status: **'Infeasible'**			Solver: **NOMAD** Iterations: **60** FuncEvals: **549** Time: **0.254** Status: **'Infeasible'**		Solver: **NOMAD** Iterations: **46** FuncEvals: **418** Time: **0.220** Status: **'Infeasible'**

* NOMAD: Nonlinear Optimization using the MADS Algorithm.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.3 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**SCIP****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.02; 0.03; 1; 0.005; 1.5; 1.2e+5]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073419e-04**	7.585975e-04	NaN
V_a (m3)	-	-	1.101565e-04	**8.397486e-05**	NaN
P_j (watts)	-	-	5.132728e+01	9.273171e+01	**NaN**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.923367e-02	1.273237e-01	Inf
l_a (m)	[0.001	0.05]	5.273377e-03	2.144241e-03	0.000000e+00
E (m)	[0.001	0.05]	3.777814e-03	1.429029e-03	0.000000e+00
C (m)	[0.001	0.05]	6.159062e-03	5.087758e-03	NaN
β	[0.8	1.0]	8.000000e-01	8.000002e-01	0.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.619297e-01	3.815818e-01	NaN
J_{cu} (A/m2)	[10^5	10^7]	6.149371e+06	1.000000e+07	0.000000e+00
K_f	[0.01	0.5]	1.801193e-01	1.065616e-01	0.000000e+00
e (m)***	[0.001	0.005]	1.000000e-03	1.397606e-03	0.000000e+00
λ (=D/L)	[1	2.5]	1.839047e+00	1.000002e+00	0.000000e+00
E_{ch}	= = 10^11		1.000000e+11	1.000321e+11	0.000000e+00
Γ_{em}	= = 10		9.999923e+00	1.000076e+01	NaN
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **SCIP** BBNodes: **7** BBGap: **0** Time: **2.736** Status: **'Globally Optimal'**			Solver: **SCIP** BBNodes: **253** BBGap: **0** Time: **3.025** Status: **'Globally Optimal'**		Solver: **SCIP** BBNodes: **0** BBGap: **0** Time: **0.412** Status: **'Infeasible'**

* SCIP: Spatial Branch and Bound using IPOPT and SoPlex.

** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.

*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.4 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**SCIP****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.001; 0.004; 0.9; 0.003; 2.1; 9e+6]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073419e-04**	7.585975e-04	NaN
V_a (m3)	-	-	1.101565e-04	**8.397486e-05**	NaN
P_j (watts)	-	-	5.132728e+01	9.273171e+01	**NaN**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.923367e-02	1.273237e-01	Inf
l_a (m)	[0.001	0.05]	5.273377e-03	2.144241e-03	0.000000e+00
E (m)	[0.001	0.05]	3.777814e-03	1.429029e-03	0.000000e+00
C (m)	[0.001	0.05]	6.159062e-03	5.087758e-03	NaN
B	[0.8	1.0]	8.000000e-01	8.000000e-01	0.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.619297e-01	3.815818e-01	NaN
J_{cu} (A/m2)	[10^5	10^7]	6.149371e+06	1.000000e+07	0.000000e+00
K_f	[0.01	0.5]	1.801193e-01	1.065616e-01	0.000000e+00
e (m)***	[0.001	0.005]	1.000000e-03	1.397606e-03	0.000000e+00
λ (=D/L)	[1	2.5]	1.839047e+00	1.000002e+00	0.000000e+00
E_{ch}	= = 10^11		1.000000e+11	1.000321e+11	0.000000e+00
Γ_{em}	= = 10		9.999923e+00	1.000076e+01	NaN

Process Info.		
Minimizing V_u	Minimizing V_a	Minimizing P_j
Solver: **SCIP** BBNodes: **7** BBGap: **0** Time: **2.678** Status: **'Globally Optimal'**	Solver: **SCIP** BBNodes: **253** BBGap: **0** Time: **3.031** Status: **'Globally Optimal'**	Solver: **SCIP** BBNodes: **0** BBGap: **0** Time: **0.432** Status: **'Infeasible'**

* SCIP: Spatial Branch and Bound using IPOPT and SoPlex.

** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.

*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.5 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**NLOPT****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.02; 0.03; 1; 0.005; 1.5; 1.2e+5]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**4.084020e-04**	1.258234e-03	7.786491e-04
V_a (m3)	-	-	6.560956e-05	**3.306730e-04**	2.314994e-04
P_j (watts)	-	-	3.897836e+01	4.627537e+01	**4.717578e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	5.293109e-02	5.443493e-02	6.430860e-02
l_a (m)	[0.001	0.05]	3.488030e-03	1.975236e-02	1.153319e-02
E (m)	[0.001	0.05]	2.941805e-03	2.301094e-02	2.382160e-03
C (m)	[0.001	0.05]	5.359411e-03	5.982930e-03	7.996376e-03
β	[0.8	1.0]	9.540390e-01	9.617083e-01	9.180627e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	3.370561e-01	3.732689e-01	5.226033e-01
J_{cu} (A/m2)	[10^5	10^7]	6.967453e+06	2.491603e+06	7.744603e+06
K_f	[0.01	0.5]	2.507205e-01	1.173832e+00	2.668158e-01
e (m)***	[0.001	0.005]	2.634964e-03	2.890363e-03	3.785183e-03
λ (=D/L)	[1	2.5]	2.405466e+00	2.339012e+00	1.979890e+00
E_{ch}	= = 10^11		9.996779e+10	9.999772e+10	1.000155e+11
Γ_{em}	= = 10		4.880745e+00	-4.179773e+00	7.909498e+00

Process Info.		
Minimizing V_u	Minimizing V_a	Minimizing P_j
Solver: **NLOPT** FuncEvals: **10000** Time: **140.320** Status: **' Exceeded FuncEvals Limit '**	Solver: **NLOPT** FuncEvals: **10000** Time: **212.710** Status: **'Exceeded FuncEvals Limit'**	Solver: **NLOPT** FuncEvals: **10000** Time: **138.229** Status: **'Exceeded FuncEvals Limit'**

* NLOPT: Chosen Algorithm was ISRES (Improved Stochastic Ranking Evolution Strategy).
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.6 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**NLOPT****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.001; 0.004; 0.9; 0.003; 2.1; 9e+6]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**9.528931e-04**	1.594242e-03	8.317958e-04
V_a (m3)	-	-	5.791919e-04	**4.951416e-04**	3.549327e-04
P_j (watts)	-	-	4.038832e+01	5.261827e+01	**2.378292e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	5.403947e-02	6.150134e-02	5.318113e-02
l_a (m)	[0.001	0.05]	4.662216e-02	3.106490e-02	2.627788e-02
E (m)	[0.001	0.05]	4.831584e-03	2.396208e-02	4.042280e-03
C (m)	[0.001	0.05]	6.746224e-03	6.156656e-03	7.867803e-03
β	[0.8	1.0]	9.889099e-01	9.165296e-01	8.408697e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.093128e-01	4.030414e-01	5.614047e-01
J_{cu} (A/m2)	[10^5	10^7]	5.437818e+06	2.441766e+06	4.612523e+06
K_f	[0.01	0.5]	3.813907e-01	1.169937e+00	2.573095e-01
e (m)***	[0.001	0.005]	3.352541e-03	3.125793e-03	2.451326e-03
λ (=D/L)	[1	2.5]	2.356129e+00	2.070263e+00	2.394157e+00
E_{ch}	= = 10^11		1.000085e+11	1.000071e+11	6.020069e+10
Γ_{em}	= = 10		6.614375e+00	-4.997480e+00	6.972403e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **NLOPT** FuncEvals: **10000** Time: **84.934** Status: **'Exceeded FuncEvals Limit'**			Solver: **NLOPT** FuncEvals: **10000** Time: **180.600** Status: **'Exceeded FuncEvals Limit'**		Solver: **NLOPT** FuncEvals: **10000** Time: **70.808** Status: **'Exceeded FuncEvals Limit'**

* NLOPT: Chosen Algorithm was ISRES (Improved Stochastic Ranking Evolution Strategy).
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.7 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**GMATLAB****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.02; 0.03; 1; 0.005; 1.5; 1.2e+5]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**2.042719e-03**	2.042719e-03	2.042719e-03
V_a (m3)	-	-	6.985916e-04	**6.985916e-04**	6.985916e-04
P_j (watts)	-	-	1.425624e-01	1.425624e-01	**1.425624e-01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	8.488264e-02	8.488264e-02	8.488264e-02
l_a (m)	[0.001	0.05]	3.000000e-02	3.000000e-02	3.000000e-02
E (m)	[0.001	0.05]	2.000000e-02	2.000000e-02	2.000000e-02
C (m)	[0.001	0.05]	6.598671e-03	6.598671e-03	6.598671e-03
β	[0.8	1.0]	1.000000e+00	1.000000e+00	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	3.959203e-01	3.959203e-01	3.959203e-01
J_{cu} (A/m2)	[10^5	10^7]	1.200000e+05	1.200000e+05	1.200000e+05
K_f	[0.01	0.5]	1.178097e+00	1.178097e+00	1.178097e+00
e (m)***	[0.001	0.005]	5.000000e-03	5.000000e-03	5.000000e-03
λ (=D/L)	[1	2.5]	1.500000e+00	1.500000e+00	1.500000e+00
E_{ch}	= = 10^11		2.016000e+08	2.016000e+08	2.016000e+08
Γ_{em}	= = 10		-2.962760e-01	-2.962760e-01	-2.962760e-01
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **GMATLAB** Iterations: **1** FuncEvals: **178** Time: **3.869** Status: **'No Feasible Point Found'**			Solver: **GMATLAB** Iterations: **1** FuncEvals: **178** Time: **3.733** Status: **'No Feasible Point Found'**		Solver: **GMATLAB** Iterations: **1** FuncEvals: **178** Time: **3.566** Status: **'No Feasible Point Found'**

* GMATLAB: PATTERNSERACH - Global Direct Search.

** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.

*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 16.8 – Non-reformulated model results

GNLP Optimization of *Original Model* using *OPTI Toolbox/**GMATLAB****					
Initial Random Guess: x_0 = [E; l_a; β; e; λ; J_{cu}] = [0.001; 0.004; 0.9; 0.003; 2.1; 9e+6]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**9.607171e-04**	9.607171e-04	9.618725e-04
V_a (m3)	-	-	1.603661e-04	**1.603661e-04**	1.608117e-04
P_j (watts)	-	-	8.939259e+01	8.939259e+01	**8.930148e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	1.227219e-01	1.227219e-01	1.225968e-01
l_a (m)	[0.001	0.05]	4.000000e-03	4.000000e-03	4.000000e-03
E (m)	[0.001	0.05]	1.488281e-03	1.488281e-03	1.488281e-03
C (m)	[0.001	0.05]	5.984142e-03	5.984142e-03	6.006891e-03
β	[0.8	1.0]	8.863271e-01	8.863271e-01	8.896965e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.050971e-01	4.050971e-01	4.050971e-01
J_{cu} (A/m2)	[10^5	10^7]	9.797347e+06	9.797347e+06	9.797347e+06
K_f	[0.01	0.5]	1.874629e-01	1.874629e-01	1.881755e-01
e (m)***	[0.001	0.005]	3.000000e-03	3.000000e-03	3.000000e-03
λ (=D/L)	[1	2.5]	1.037500e+00	1.037500e+00	1.038558e+00
E_{ch}	= = 10^11		1.000000e+11	1.000000e+11	1.000000e+11
Γ_{em}	= = 10		1.000000e+01	1.000000e+01	1.000000e+01
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **GMATLAB** Iterations: **5** FuncEvals: **1437** Time: **2.469** Status: **'Solved'**			Solver: **GMATLAB** Iterations: **5** FuncEvals: **1437** Time: **2.481** Status: **'Solved'**		Solver: **GMATLAB** Iterations: **4** FuncEvals: **10000** Time: **9.420** Status: **'Exceeded FuncEvals Limit'**

* GMATLAB: PATTERNSERACH - Global Direct Search.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 17.1 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**NOMAD****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.02; 0.03; 1; 0.005]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073467e-04**	6.698825e-04	7.534970e-04
V_a (m3)	-	-	1.101250e-04	**6.716796e-05**	2.917886e-04
P_j (watts)	-	-	5.134028e+01	9.319748e+01	**3.760183e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.925057e-02	1.272890e-01	5.092966e-02
l_a (m)	[0.001	0.05]	5.270453e-03	1.698334e-03	1.748981e-02
E (m)	[0.001	0.05]	3.778993e-03	2.152563e-03	3.237747e-03
C (m)	[0.001	0.05]	6.156923e-03	4.181235e-03	1.014607e-02
β	[0.8	1.0]	8.000000e-01	8.000000e-01	9.669861e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.617693e-01	3.135926e-01	6.295482e-01
J_{cu} (A/m2)	[10^5	10^7]	6.148411e+06	8.146536e+06	6.642469e+06
K_f	[0.01	0.5]	1.801638e-01	1.188488e-01	1.931853e-01
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.001733e-03
λ (=D/L)	[1	2.5]	1.838598e+00	1.000275e+00	2.499996e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **NOMAD** Iterations: **73** FuncEvals: **642** Time: **6.436** Status: **'Converged'**			Solver: **NOMAD** Iterations: **105** FuncEvals: **912** Time: **8.234** Status: **'Converged'**		Solver: **NOMAD** Iterations: **117** FuncEvals: **1177** Time: **21.435** Status: **'Converged'**
* NOMAD: Nonlinear Optimization using the MADS Algorithm. ** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process. *** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.					

Tbl. 17.2 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**NOMAD****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.001; 0.004; 0.9; 0.003]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073466e-04**	6.702428e-04	7.618406e-04
V_a (m3)	-	-	1.101352e-04	**6.718393e-05**	3.016246e-04
P_j (watts)	-	-	5.133618e+01	9.314388e+01	**3.759520e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.924561e-02	1.272304e-01	5.092959e-02
l_a (m)	[0.001	0.05]	5.271356e-03	1.699536e-03	1.808481e-02
E (m)	[0.001	0.05]	3.777933e-03	2.137671e-03	3.214929e-03
C (m)	[0.001	0.05]	6.157978e-03	4.195515e-03	1.021033e-02
β	[0.8	1.0]	8.000000e-01	8.000000e-01	9.720667e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.618484e-01	3.146636e-01	6.302240e-01
J_{cu} (A/m2)	[10^5	10^7]	6.149274e+06	8.174864e+06	6.665999e+06
K_f	[0.01	0.5]	1.801238e-01	1.182874e-01	1.931696e-01
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.002051e-03
λ (=D/L)	[1	2.5]	1.838730e+00	1.000735e+00	2.500000e+00

Process Info.		
Minimizing V_u	Minimizing V_a	Minimizing P_j
Solver: **NOMAD** Iterations: **71** FuncEvals: **693** Time: **4.197** Status: **'Converged'**	Solver: **NOMAD** Iterations: **78** FuncEvals: **787** Time: **4.989** Status: **'Converged'**	Solver: **NOMAD** Iterations: **139** FuncEvals: **1428** Time: **15.000** Status: **'Converged'**

* NOMAD: Nonlinear Optimization using the MADS Algorithm.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 17.3 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**SCIP****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.02; 0.03; 1; 0.005]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073466e-04**	6.704412e-04	7.691523e-04
V_a (m3)	-	-	1.101574e-04	**6.715612e-05**	3.076617e-04
P_j (watts)	-	-	5.132766e+01	9.321088e+01	**3.756457e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.923418e-02	1.273256e-01	5.092987e-02
l_a (m)	[0.001	0.05]	5.273380e-03	1.697535e-03	1.789984e-02
E (m)	[0.001	0.05]	3.777815e-03	2.133976e-03	3.107850e-03
C (m)	[0.001	0.05]	6.159063e-03	4.195465e-03	1.054681e-02
β	[0.8	1.0]	8.000000e-01	8.000002e-01	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.619297e-01	3.146598e-01	6.328087e-01
J_{cu} (A/m2)	[10^5	10^7]	6.149370e+06	8.181939e+06	6.779864e+06
K_f	[0.01	0.5]	1.801194e-01	1.181481e-01	1.935779e-01
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	1.839033e+00	9.999874e-01	2.499986e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **SCIP** BBNodes: **105** BBGap: **0** Time: **2.889** Status: **'Globally Optimal'**			Solver: **SCIP** BBNodes: **1** BBGap: **0** Time: **1.031** Status: **'Globally Optimal'**		Solver: **SCIP** BBNodes: **415** BBGap: **0** Time: **2.427** Status: **'Globally Optimal'**
* SCIP: Spatial Branch and Bound using IPOPT and SoPlex. ** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process. *** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.					

Tbl. 17.4 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**SCIP****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.001; 0.004; 0.9; 0.003]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.073466e-04**	6.704412e-04	7.691523e-04
V_a (m3)	-	-	1.101574e-04	**6.715612e-05**	3.076617e-04
P_j (watts)	-	-	5.132766e+01	9.321088e+01	**3.756457e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.923418e-02	1.273256e-01	5.092987e-02
l_a (m)	[0.001	0.05]	5.273380e-03	1.697535e-03	1.789984e-02
E (m)	[0.001	0.05]	3.777815e-03	2.133976e-03	3.107850e-03
C (m)	[0.001	0.05]	6.159063e-03	4.195465e-03	1.054681e-02
B	[0.8	1.0]	8.000000e-01	8.000002e-01	1.000000e+00
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.619297e-01	3.146598e-01	6.328087e-01
J_{cu} (A/m2)	[10^5	10^7]	6.149370e+06	8.181939e+06	6.779864e+06
K_f	[0.01	0.5]	1.801194e-01	1.181481e-01	1.935779e-01
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	1.839033e+00	9.999874e-01	2.499986e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **SCIP** BBNodes: **25** BBGap: **0** Time: **2.429** Status: **'Globally Optimal'**			Solver: **SCIP** BBNodes: **1** BBGap: **0** Time: **0.968** Status: **'Globally Optimal'**		Solver: **SCIP** BBNodes: **415** BBGap: **0** Time: **2.349** Status: **'Globally Optimal'**
* SCIP: Spatial Branch and Bound using IPOPT and SoPlex. ** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process. *** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.					

Tbl. 17.5 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**NLOPT****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.02; 0.03; 1; 0.005]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.341988e-04**	6.813361e-04	7.708444e-04
V_a (m3)	-	-	1.411601e-04	**6.799285e-05**	3.114108e-04
P_j (watts)	-	-	4.541659e+01	9.295736e+01	**3.760063e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.150313e-02	1.271503e-01	5.095499e-02
l_a (m)	[0.001	0.05]	7.436835e-03	1.689157e-03	1.849024e-02
E (m)	[0.001	0.05]	3.261543e-03	1.959779e-03	3.168672e-03
C (m)	[0.001	0.05]	7.559451e-03	4.409907e-03	1.035890e-02
β	[0.8	1.0]	8.342989e-01	8.151211e-01	9.849202e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	5.436506e-01	3.246075e-01	6.310500e-01
J_{cu} (A/m2)	[10^5	10^7]	6.618193e+06	8.537830e+06	6.714479e+06
K_f	[0.01	0.5]	1.702704e-01	1.138923e-01	1.937953e-01
e (m)***	[0.001	0.005]	1.069339e-03	1.005265e-03	1.006757e-03
λ (=D/L)	[1	2.5]	2.070203e+00	1.001365e+00	2.498753e+00

Process Info.		
Minimizing V_u	Minimizing V_a	Minimizing P_j
Solver: **NLOPT** FuncEvals: **2153** Time: **20.003** Status: **'Solved'**	Solver: **NLOPT** FuncEvals: **10000** Time: **93.738** Status: **'Exceeded FuncEvals Limit'**	Solver: **NLOPT** FuncEvals: **10000** Time: **94.914** Status: **'Exceeded FuncEvals Limit'**

* NLOPT: Chosen Algorithm was ISRES (Improved Stochastic Ranking Evolution Strategy).
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 17.6 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**NLOPT****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.001; 0.004; 0.9; 0.003]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**6.083841e-04**	6.698542e-04	8.120983e-04
V_a (m3)	-	-	1.151772e-04	**6.895591e-05**	3.674387e-04
P_j (watts)	-	-	4.966203e+01	9.165962e+01	**3.764723e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	6.696684e-02	1.250666e-01	5.096259e-02
l_a (m)	[0.001	0.05]	5.714291e-03	1.756675e-03	2.263994e-02
E (m)	[0.001	0.05]	3.818185e-03	2.278795e-03	3.310916e-03
C (m)	[0.001	0.05]	6.351635e-03	4.189342e-03	1.022558e-02
β	[0.8	1.0]	8.010750e-01	8.088941e-01	9.873789e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.757334e-01	3.107458e-01	6.213771e-01
J_{cu} (A/m2)	[10^5	10^7]	6.116775e+06	7.917687e+06	6.568662e+06
K_f	[0.01	0.5]	1.821188e-01	1.261006e-01	2.010141e-01
e (m)***	[0.001	0.005]	1.006183e-03	1.029348e-03	1.009262e-03
λ (=D/L)	[1	2.5]	1.901298e+00	1.018049e+00	2.498381e+00

Process Info.		
Minimizing V_u	Minimizing V_a	Minimizing P_j
Solver: **NLOPT** FuncEvals: **7523** Time: **81.440** Status: **'Solved'**	Solver: **NLOPT** FuncEvals: **7376** Time: **76.206** Status: **'Solved'**	Solver: **NLOPT** FuncEvals: **10000** Time: **92.987** Status: **'Exceeded FuncEvals Limit'**

* NLOPT: Chosen Algorithm was ISRES (Improved Stochastic Ranking Evolution Strategy).
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 17.7 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**GMATLAB****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.02; 0.03; 1; 0.005]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**7.911548e-04**	7.911548e-04	8.895887e-04
V_a (m3)	-	-	3.289130e-04	**3.289130e-04**	4.756686e-04
P_j (watts)	-	-	3.795102e+01	3.795102e+01	**3.795100e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	5.092959e-02	5.092959e-02	5.092956e-02
l_a (m)	[0.001	0.05]	2.582598e-02	2.582598e-02	3.178168e-02
E (m)	[0.001	0.05]	4.450412e-03	4.450412e-03	4.450412e-03
C (m)	[0.001	0.05]	7.919656e-03	7.919656e-03	9.466492e-03
β	[0.8	1.0]	8.000000e-01	8.000000e-01	9.999998e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	5.939742e-01	5.939742e-01	5.679896e-01
J_{cu} (A/m2)	[10^5	10^7]	5.665665e+06	5.665665e+06	5.665665e+06
K_f	[0.01	0.5]	2.054757e-01	2.054757e-01	2.568446e-01
e (m)***	[0.001	0.005]	1.000000e-03	1.000000e-03	1.000000e-03
λ (=D/L)	[1	2.5]	2.500000e+00	2.500000e+00	2.500001e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **GMATLAB** Iterations: **4** FuncEvals: **521** Time: **4.414** Status: **'Solved'**			Solver: **GMATLAB** Iterations: **4** FuncEvals: **521** Time: **4.396** Status: **'Solved'**		Solver: **GMATLAB** Iterations: **4** FuncEvals: **413** Time: **4.516** Status: **'Solved'**

* GMATLAB: PATTERNSERACH - Global Direct Search.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

Tbl. 17.8 – Reformulated model results

GNLP Optimization of *Reformulated Model* using *OPTI Toolbox/**GMATLAB****					
Initial Random Guess: x_0 = [E; l_a; β; e] = [0.001; 0.004; 0.9; 0.003]					
IOs	LB	UB	Minimizing V_u	Minimizing V_a	Minimizing P_j
V_u (m3)	-	-	**1.097014e-03**	1.097014e-03	1.018780e-03
V_a (m3)	-	-	1.832101e-04	**1.832101e-04**	3.604332e-04
P_j (watts)	-	-	1.001937e+02	1.001937e+02	**4.271999e+01**
D (m)	[0.01	0.5]	1.273240e-01	1.273240e-01	1.273240e-01
L (m)	[0.004	0.5]	1.380735e-01	1.380735e-01	5.412079e-02
l_a (m)	[0.001	0.05]	4.000000e-03	4.000000e-03	2.693137e-02
E (m)	[0.001	0.05]	1.000000e-03	1.000000e-03	1.226294e-02
C (m)	[0.001	0.05]	6.422581e-03	6.422581e-03	6.735031e-03
β	[0.8	1.0]	9.000000e-01	9.000000e-01	8.000000e-01
p**	[1	10]	4	4	4
B_e (T)	[0.1	1.0]	4.281721e-01	4.281721e-01	5.051273e-01
J_{cu} (A/m2)	[10^5	10^7]	1.195229e+07	1.195229e+07	3.413137e+06
K_f	[0.01	0.5]	1.696460e-01	1.696460e-01	5.000010e-01
e (m)***	[0.001	0.005]	3.000000e-03	3.000000e-03	1.000002e-03
λ (=D/L)	[1	2.5]	9.221461e-01	9.221461e-01	2.352589e+00
Process Info.					
Minimizing V_u			Minimizing V_a		Minimizing P_j
Solver: **GMATLAB** Iterations: **1** FuncEvals: **226** Time: **2.906** Status: **'No Feasible Point Found'**			Solver: **GMATLAB** Iterations: **1** FuncEvals: **226** Time: **2.845** Status: **' No Feasible Point Found''**		Solver: **GMATLAB** Iterations: **5** FuncEvals: **1030** Time: **2.018** Status: **'Solver Error'**

* GMATLAB: PATTERNSERACH - Global Direct Search.
** We have chosen the value of parameter *p* to be equal to *4* before the start of optimization process.
*** The lower bound of *e* has been changed from *0.0001* to *0.001* to cope with the manufacturing limitations.

A2 - A short report on stochastic algorithms (particle swarm optimization)

Not included in this document, please go to:

https://www.dropbox.com/s/8vr7j67e3dwd11o/02_PSO_Apdx_02.pdf

A3 - A short summary on deterministic algorithms (state-space search)

Not included in this document, please go to:

https://www.dropbox.com/s/peahfw559us4qfb/03_DM_Apdx_01.pdf