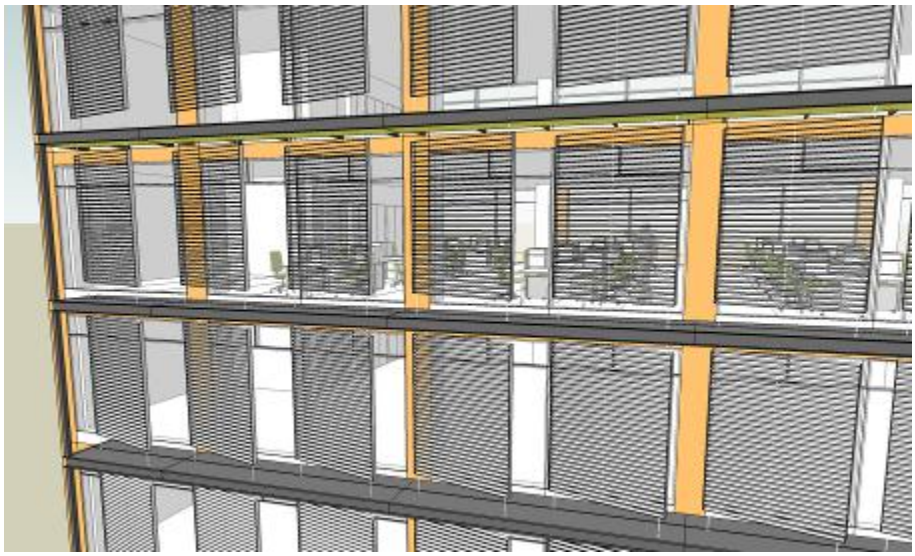
UNIVERSITÀ DEGLI STUDI DI UDINE

DOTTORATO DI RICERCA IN
TECNOLOGIE CHIMICHE ED ENERGETICHE

OPTIMISATION OF OFFICE BUILDING FAÇADES BY MEANS OF GENETIC ALGORITHMS

Dott. Gianluca RAPONE



COMMISSIONE

Prof. Giulio Lorenzini	REVISORE
Dott. Matthias SCHUSS	REVISORE
Prof.	COMMISSARIO
Prof.	COMMISSARIO
Prof.	COMMISSARIO
Prof. Onorio SARO	SUPERVISORE

Prof. Alfredo Soldati	COORDINATORE DEL DOTTORATO

Author’s e-mail: gianluca.rapone@uniud.it

Author’s address:

Dipartimento di Ingegneria Elettrica
Gestionale e Meccanica
Università degli Studi di Udine
Via delle Scienze, 106
33100 Udine – Italia
tel. +39 0432 5580xx
fax. +39 0432 55xxxx
web. http://www.diegm.uniud.it

Contents

List of Figures

List of Tables

Introduction

The importance of considering the environmental performance of buildings during conceptual stages of the design process is growing as a consequence of the restrictive requirements of building regulations and energy certification. This also reflects a more global concern to develop a new strategy to limit the effects of climate change. As the building industry is responsible for an extensive amount of energy being consumed and, as a result, of carbon emissions being produced, today the design of buildings is required to respond to climate change in more than one way. In the first place a reduction of carbon dioxide emissions is necessary in order to respond to the growing need of *mitigation* of their effects on the environment; secondarily the concept of *integrated design* is being increasingly acknowledged as a key issue to address *sustainability* in order to deal with the adaptation of building design to the actual and future conditions [1].

The façade plays a key role in the design of buildings that need to meet strict requirements of energy efficiency and at the same time provide internal comfort conditions. In air-conditioned buildings, and especially in office buildings that have highly glazed curtain wall façades, the energy consumption levels for heating, cooling and artificial lighting strongly depend on solar exposure and on the performance of the building envelope, since the latter is responsible for heat losses, solar heat gains and it allows for daylighting. The complicated interactions between the various parameters involved make the design of a good façade a very challenging task. It is extremely important to take into account all aspects that affect the energy and comfort performance of the building at the same time and to find the best balance between the resultant contradictory requirements.

In the published work the importance of building envelope design in ensuring thermal and visual comfort conditions is a widely discussed topic [2,3], and its implications in office buildings are analyzed in depth both on a theoretical point of view and with practical applications [4,5]. An interesting example is given by a toolkit developed by CIBSE (The Chartered Institution of Building Services Engineers, UK) for the selection of suitable façade systems based on prior investigation of the performance of 37 different highly glazed curtain wall façades [6]. Extensive analysis of different façade parameters and how they influence office internal conditions and energy consumption is provided by Hausladen et al. [7]. The integration of photovoltaic technologies in the building envelope is also a topic worthy of note, and for example the electricity benefits of daylight and building integrated photovoltaics are studied by Vartiainen for various façade layouts in office buildings [8].

The traditional techniques used to study the impacts of design variables on building operation usually involve parametric studies to compare different alternatives, often employing dynamic energy simulation programs. While these can be valuable tools to assess the performance of different envelope configurations in early stages, yet the exploration of the design space is usually not complete and thus ineffective in determining the sought-after solutions. Additional design methods and tools to help with decision-making in the concept phase are needed to cope with the complexity of the problem and to be able to generate and compare many potential design alternatives.

Simulation based optimisation is a procedure that couples an optimisation program to a simulation program whose function is to calculate a certain performance of a model. By means of an optimisation algorithm it is possible to perform an automated search for one or more optimal solutions exploring the design space in depth but without having to consider all possible solutions, which would be an extremely time consuming task. In the field of building design, the simulation can be carried out by any program that can evaluate a model of the object under study. This includes available dynamic energy simulation programs (*EnergyPlus*, *TRNSYS*, etc.) or lighting analysis programs (*Radiance*), as well as custom made programs or routines.

Different types of algorithms can be used for the optimisation process, and they can usually be classified as belonging to two main groups: deterministic gradient based algorithms and probabilistic ones. *Evolutionary algorithms* have been employed for a number of building related optimisation problems, including building envelope design, because of their capability in handling big amounts of variables and potential solutions. They are a family of probabilistic algorithms that base their search for optimal solutions on the principles of evolution of the species or the behaviour of groups of animals; among others they comprehend *Genetic Algorithms (GA)*, *Evolutionary Neural Networks (ENN)* and *Particle Swarm Optimisation (PSO).*

There is a wide selection of works where evolutionary algorithms are employed to optimize both building envelopes and HVAC systems parameters [9,10,11]. The most outstanding is the one by Magnier et al. [12], where a multi-objective optimisation is performed by means of both genetic algorithm and artificial neural network, and with the use of TRNSYS simulation software. Sticking to envelopes only, some authors focused on residential buildings optimising the size of windows [13], or all characteristics of the envelope [14], in terms of overall energy performance. Other authors involved also the shape of the building into the optimisation process, studying the impact of different climates [15], or searching for the trade-off between construction costs and environmental impact [16]. Hasan et al. [17] used PSO algorithms to optimise life-cycle cost of a single detached house in Finland, while Stephan et al. [18] used a hybrid PSO-Hooks Jeeves algorithm for the optimisation of envelope openings in order to ensure desired natural ventilation. In more recent times, the design of energy efficient façades was optimised via ENN for both single and multi-objectives in the work of Zemella et al. [19]. Moreover, in a

previous effort the author of the present thesis applied PSO algorithms to optimise the curtain wall of office buildings in terms of carbon emissions [20].
Although the optimisation techniques employed are not regarded as evolutionary algorithms, another notable and interesting work in the field of simulation based envelope design optimisation is the one by Mahdavi and Mahattanatawe [21,22]: the authors developed a computational environment that derives the optimal basic properties of a "virtual" enclosure given a set of indoor climate requirements, and then uses the found values to identify an actual building enclosure construction from a linked database.
Both free and commercial programs exist to support engineers or designers in using simulation based optimisation techniques in a wide range of applications. However, their operation is usually quite complicated and arranging an optimisation process can require a lot of effort and knowledge of the tool itself. Moreover, as they are typically general optimisation tools, they do not focus specifically on simulation programs used in the field of building design.
For the work of this thesis, a simulation-optimisation tool was developed in *Matlab* environment to automate the coupling of the free energy simulation program *EnergyPlus* to the optimisation capabilities of the genetic algorithms included in *Matlab*'s *Optimisation Toolbox*. Besides the routines and functions that make up the structure of the program, a graphical user interface was also created to make the process of setting up a simulation based optimisation more straightforward and quick. The program was then employed in different case studies to create and perform automated optimisation processes to search for the optimal values of selected envelope parameters.
In the first chapter the role of the building envelope as an interface is described and the challenges of designing an energy efficient façade are underlined. In the second chapter simulation based optimisation methods are thoroughly explained, along with the theory of genetic algorithms and the operation of the energy simulation program *EnergyPlus*. In the third chapter the custom made program *ePlusOpt* is presented, illustrating both the underlying processes and programming and its function as a graphical user interface. The program was then employed for different optimisation processes in three case studies, whose results are reported in the following chapters.

1

Building envelope design

1.1 The façade as interface

The façade of a building is responsible for the interaction between the interior and the external environment: it has to provide thermal insulation, acoustic insulation, weather tightness, allow for daylight and for a view to the outside (figure 1.1). Moreover, it confers the building its aesthetical appearance so its design is also tightly related to the architectural concepts and ambitions. Since all these factors are strongly interrelated and they give rise to a number of functional conflicts, the design of a building envelope can be a very complex task.

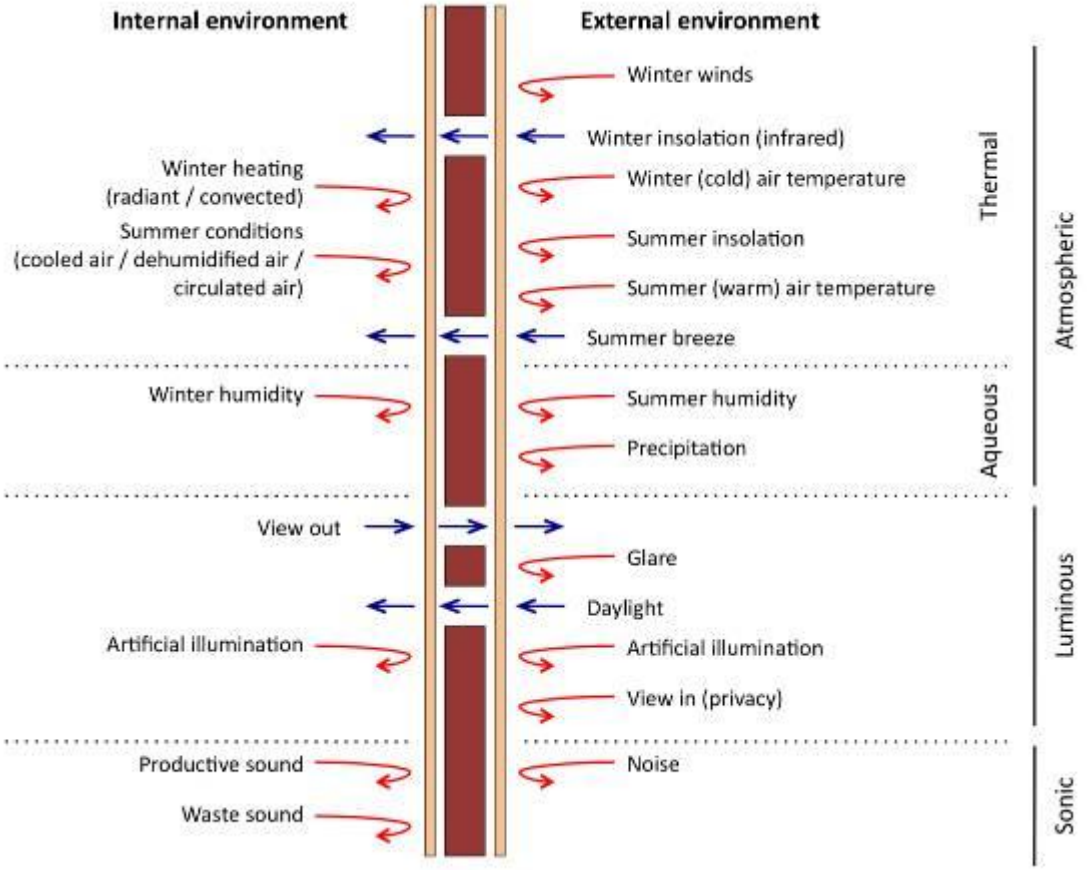


Figure 1.1 – The façade as interface

The role of the façade with respect to energy consumption (and the consequent carbon emissions) and indoor climate is to guarantee internal comfort conditions to the greatest possible extent while reducing the operation energy requirements, all under the dynamically changing conditions of the outdoor climate. To give an idea of how complicated this role of the façade as an interface is, let us consider how different requirements lead to conflicts of objectives. The envelope must maximise solar heat gains through windows in winter, but prevent the entry of too much solar radiation in summer; at the same time it has to provide daylight deep into the room during the period of building use and a good level of natural ventilation. The admission of the desired amount of solar radiation in winter and the optimum use of natural light often lead to glare problems. The solar screening needed in summer results in a reduction of daylight entering the interior. A strategy that involves strong natural ventilation of the building in city centres is usually accompanied by the entry of unwanted noise. The objective in envelope design is thus to find a compromise between the various requirements, bearing in mind the type of building use and the specific location.

The following paragraphs intend to give a general overview of the main envelope functions taking into account the energetic and indoor climatic characteristics of a general façade of an office building in winter and in summer, alongside some aspects of the use of natural light. The featured examples contribute to highlight the conflicting requirements that must be faced when designing an efficient façade from both energy and comfort points of view.

1.2 Thermal model used for the parametric studies

For all the parametric studies presented in the rest of the chapter the same model of a typical office building was used. The office room considered has a 4.5m width, a 5m depth and a 3m height as shown by figure 1.2.

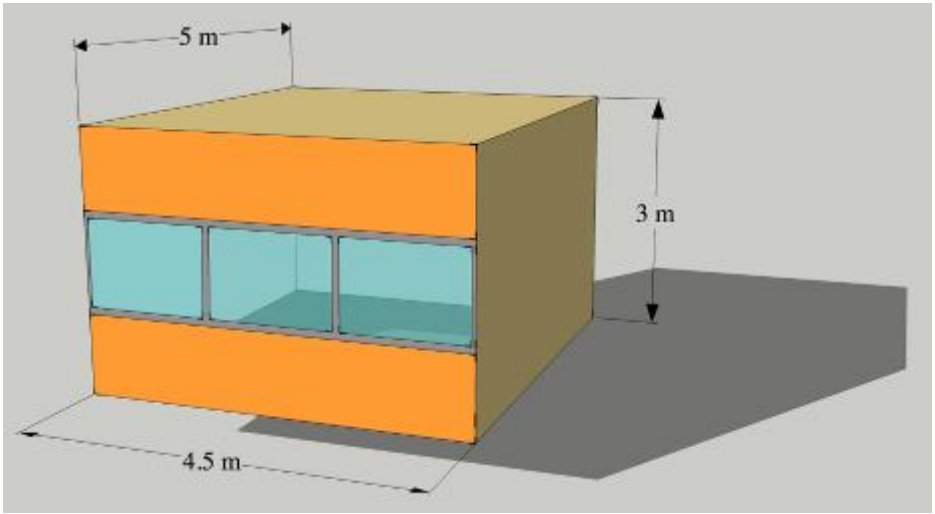


Figure 1.2 – Image of the room modelled in *EnergyPlus*

The assumptions made for internal gains, internal surfaces, ventilation and air conditioning strategies are summarized in table 1.1. All internal surfaces are considered adiabatic, no thermal mass in the floor and ceiling is present. Most of the characteristics of the façade, such as the wall thermal transmittance, the glazing type and percentage, or the presence of external shading devices, are varied throughout the different examples. To compute the energy required for the artificial lighting a daylighting control model was employed, based on two reference points positioned at desk level inside the room as pictured in figure 1.3. According to the levels of daylight coming in from the windows, the use of artificial lights in the room is regulated. The illuminance design level was set at 500 lux, the lights being turned on whenever this level is not met, working with a continuous dimming strategy.

TYPE	VALUE
Internal gains	
Electric equipment	460 W
People (62 people)	108 W/person
Lights	8 W/m^2
Surface reflectance	
Walls	60 %
Ceiling	80 %
Floor	20 %
Ventilation	
Infiltration	0.15 ach
Mechanical Ventilation	1 ach (max)
Operating strategy	
Heating setpoint / setback	22°C / 12°C
Cooling setpoint / setback	24°C / 30°C
Heating and Cooling availability	8am – 6pm on workdays

Table 1.1 – Model design assumptions

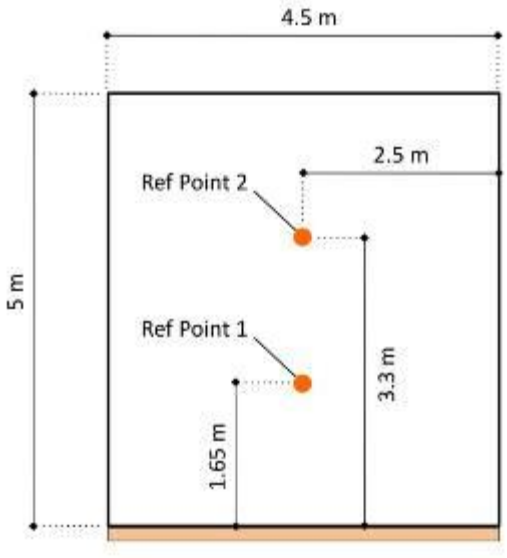


Figure 1.3 – Disposition of daylighting reference points in the model

1.3 The façade in winter

The main concern in winter is the reduction of transmission heat losses, which are influenced mainly by the insulation thickness and the quality of the glazing system. In addition, especially after the improvement of insulation standards, particular attention must be paid to thermal bridges, which can also cause surface condensation problems. Obviously the heat losses reduce with increasing insulation thickness, but

other factors are to be kept in mind because the preferred thickness can depend on the type of construction, the percentage of window area, and the amount of internal heat loads for example. The same aspects can influence also the selection of the type of glass to install. Double glazed units are the standard, with the option of applying low emissivity coatings to improve they performance. Three-pane units can lower the glazing U-value down to 0.5 W/m^{2}K but they are much more expensive and complex to build. Figures 1.4 and 1.5 show the effect of insulation standard and of percentage of glazed area respectively on the annual heating demand for the aforementioned office room under the particular conditions summarised in the tables next to them.

Insulation Standard	Wall U-value (W/m^{2}K)	Glazing U-value (W/m^{2}K)
Poor	0.5	1.4
Medium	0.3	1.1
Good	0.2	0.8

Table 1.2 – Insulation standards used in figure 1.4

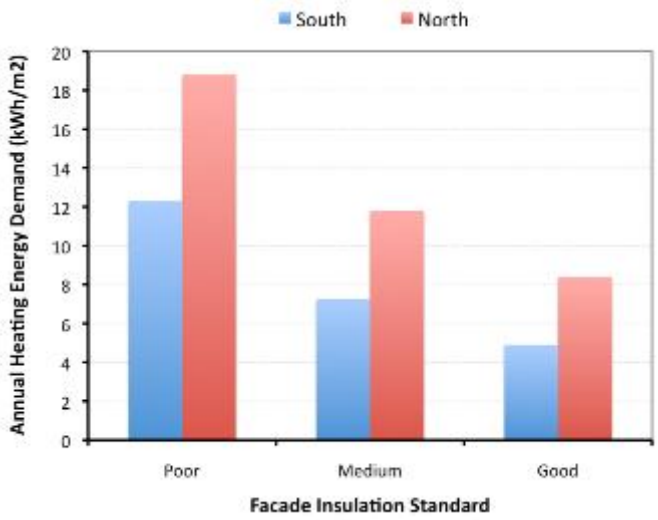


Figure 1.4 – Effect of insulation standard on heating energy demand

Climate	Paris
Glazing %	50
Glazing g-value	0.6
Blinds Strategy	Closed if: Glare I > 22 or I_{fac} > 200 W/m^2

Table 1.3 – Model assumptions applicable to figure 1.4

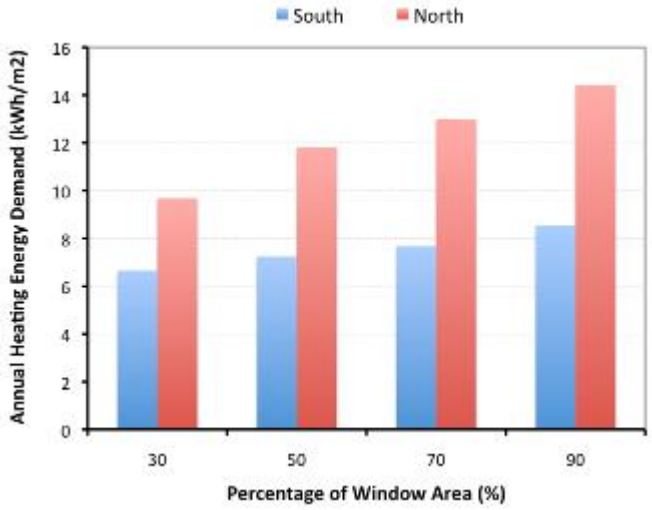


Figure 1.5 – Effect of percentage of glazed area on heating energy demand

Climate	Paris
Glazing U-value	1.1 W/m^2K
Glazing g-value	0.6
Wall U-value	0.3 W/m^2K
Blinds Strategy	Closed if: Glare I > 22 or I_{fac} > 200 W/m^2

Table 1.4 – Model assumptions applicable to figure 1.5

Solar heat gains from the windows also play an important role in reducing the heating demand during the winter months. They depend essentially on the orientation of the façade, the area of the glazed surfaces and their g-value, but also on the presence and characteristics of shading devices. However, caution must be used because the usability of these gains depends on the particular climatic conditions, on the presence of thermally exploitable storage masses and again on the internal loads.
Finally, ventilation heat losses must be taken into account: first of all a high degree of air tightness of the envelope is essential to reduce unwanted leaks, secondly the rate of air changes should be planned in order to reduce ventilation energy demand to the possible extent, and finally heat recovery systems should be considered where high flows of exhaust air are expected.

1.4 The façade in summer

Assuring a good room climate in summer can be a difficult task especially when dealing with office buildings where the internal heat loads are particularly high and the desire for transparency often leads to an increased entry of solar radiation and a reduced thermal storage mass. The internal conditions of buildings during the summer months are mainly determined by the following factors: orientation, proportion of glazed area, glazing type and solar shading strategy.
The orientation determines the azimuth and altitude angles of the sun to the façade and the intensity of the solar irradiance, which, along with the outside temperature profile, can strongly affect the comfort levels during office hours. For this reason strategies to provide appropriate solar screening and ventilation must be developed according to the characteristics of the façade arising from its orientation.

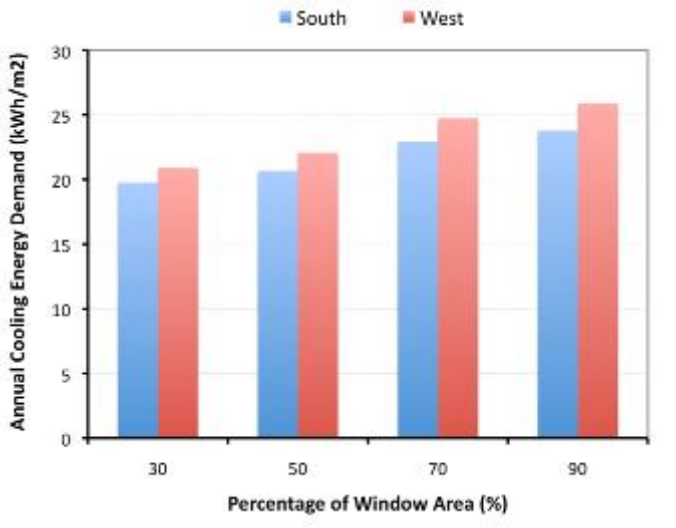


Figure 1.6 – Effect of percentage of glazed area on cooling energy demand

Climate	Rome
Wall U-value	0.3 W/m^{2}K
Glazing U-value	1.1 W/m^{2}K
Glazing g-value	0.45
Blinds Strategy	Closed if: Glare Ind > 22 or I_{fac} > 200 W/m^2
External Shading	Overhang, depth 1 m

Table 1.5 – Model assumptions applicable to figure 1.6

The percentage of glazed area achievable without exceeding internal comfort criteria also depends on the orientation: as rules of thumb north facing façades can usually have larger glazed areas, the south side allows higher transparency than the east or west sides because of the different angle of incidence of solar radiation. In general the cooling energy demand rises almost in direct proportion to the percentage of window area (figure 1.6). The design of an accurate solar screening scheme has a considerable impact on both cooling loads and comfort conditions. The total solar energy heat gains through a window are determined by the solar energy transmittance of the glazing (g-value) and by the shading factor of the solar screening provided. Solar control glass or blinds systems in the glazing cavity can influence the g-value, while the shading factor depends on the type and placement (internal or external) of system used (figure 1.7). Internal screening is inexpensive and low maintenance, but not able to keep out much of the solar radiation. External systems are more efficient but in certain cases can be affected by the weather conditions.

Shading strategy	External shading device	Glazing g-value
Type 1	Overhang, 1 m deep	0.6
Type 2	Overhang, 1 m deep	0.35
Type 3	Louvres, 0.2 m deep, 0.2 m spaced	0.6
Type 4	Louvres, 0.2 m deep, 0.2 m spaced	0.35

Table 1.6 – Shading strategies used in figure 1.7

Nevertheless, the design of solar screening must always take into account the orientation of the building and its interaction with the proportion of window area.

Furthermore, control strategies on the automation of shading devices need to be carefully thought through because they can significantly alter the overall efficiency of the whole screening system.

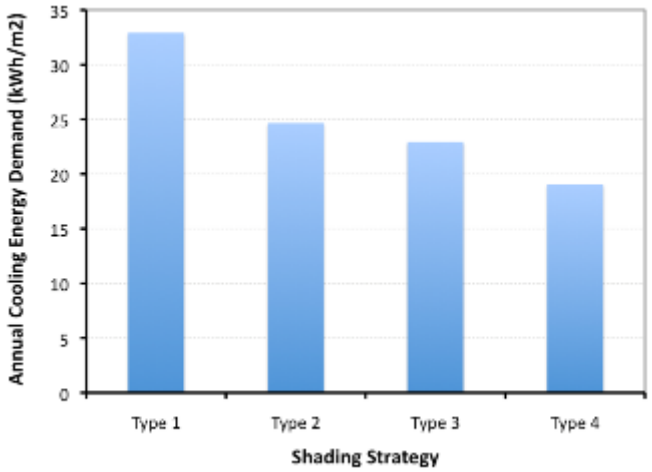


Figure 1.7 – Effect of shading strategy area on cooling energy demand

Climate	Rome
Orientation	South
Wall U-value	0.3 W/m^2K
Glazing %	70
Glazing U-value	1.1 W/m^2K
Blinds Strategy	Closed if: Glare Ind > 22 or I_{fac} > 200 W/m^2

Table 1.7 – Model assumptions applicable to figure 1.7

2

Simulation based optimisation

2.1 Concepts and process

Optimisation is a process aimed at finding the best solutions of a problem by means of minimising an objective function that somehow describes its performance. The procedure comprehends a model that describes the problem and an optimisation algorithm that minimises the objective function. The optimisation model is made up of variables, constraints and the aforementioned objective function.

In simulation based optimisation the objective function is computed by running an external program that simulates the behaviour of a particular model that represents the problem. Inside the optimisation algorithm there must be a call for the simulation solver every time the performance of a potential solution needs to be evaluated. Once the simulation produces its output, this is retrieved by the algorithm and used to calculate the objective function. This procedure is repeated throughout the whole progression of the optimisation until the algorithm reaches its stopping criterion; at this point the process stops and the optimum solution found is returned.

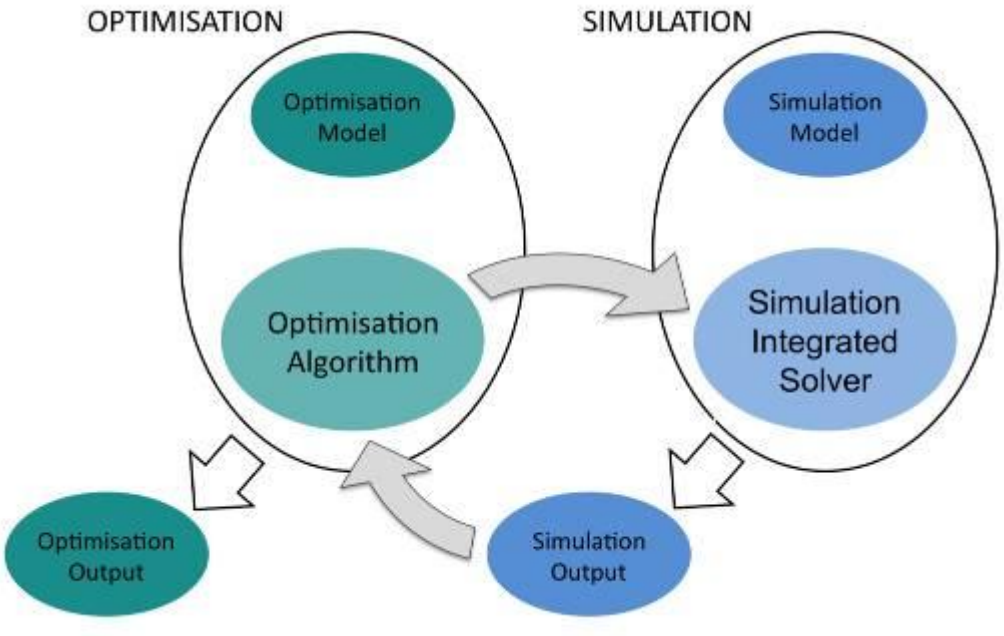


Figure 2.1 – Scheme of simulation based optimisation process

In the field of building design, the simulation side consists of dynamic simulations of thermal models run by proper software tools (see paragraph 5 of this chapter). The coupling of the two programs is not straightforward and to be achieved requires a rather large amount of programming. There are some optimisation programs that provide tools to assist the user in setting up this coupling. The next chapter will explain how the self made program *ePlusOpt* is automatically programmed to perform this task.

2.2 Mathematical background of optimisation problems

2.2.1 Basic optimisation problems

In the most general form, the optimisation problems can be stated as follows [23]: let $X \subset \Re^n$ be a user-specified constraint set, and let $f : \Re^n \to \Re$ be a user-defined cost function that is bounded from below. The constraint set *X* consists of all possible design options, and the cost function $f(\bullet)$ measures the system performance.
The aim of the optimisation process is to find a solution to the problem:

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

This problem is usually solved by iterative methods, which construct infinite sequences of progressively better approximations to a "solution", i.e. a point that satisfies an optimality condition.

2.2.2 Problems with continuous and discrete variables

It is possible to distinguish between problems whose design parameters are continuous variables, discrete variables, or both. In the latter case, the constraint set that comprises all the possible solutions becomes the following:

$$X \triangleq X_c \times X_d \quad \text{with} \quad x \triangleq (x_c, x_d) \in \Re^{n_c} \times Z^{n_d}$$

The constraints on the continuous variables are:

$$X_c \triangleq \{x \in \Re^{n_c} \mid l^i \le x^i \le u^i, i \in \{1,....,n_c\}\}$$

where the limits are $-\infty \le l^i \le u^i \le \infty, \forall i \in \{1,...,n_c\}$.

The constraints for the discrete variables are:

$$X_d \subset Z^{n_d}$$

that is a set with a finite number of integers for each variable.

The cost function becomes: $f : \Re^{n_c} \times Z^{n_d} \to \Re$

2.2.3 Problems where the cost function is computed by a simulation software

When the value of the cost function derives from simulations run by proper software tools the function $f:\Re^n \to \Re$ cannot be mathematically evaluated, but can be approximated numerically by approximating cost functions $f^*:\Re_+^p \times \Re^n \to \Re$, where the first argument is the precision parameter of the numerical solvers. In such programs, computing the cost involves solving a system of partial and ordinary differential equations that are coupled to algebraic equations. In general, it is only possible to obtain approximate numerical solutions. Hence, the cost function $f(x)$ can only be estimated by an approximating cost function $f^*(\varepsilon,x)$, where $\varepsilon \in \Re_+^p$ is a vector that contains precision parameters of the numerical solvers. Consequently, the optimisation algorithm can only be applied to $f^*(\varepsilon,x)$ and not to $f(x)$.

In such thermal building simulation programs it is common that the termination criteria of the solvers that are used to solve the partial differential equations, ordinary differential equations, and algebraic equations depend on the independent variable *x*. Therefore, a perturbation of *x* can cause a change in the sequence of solver iterations, which causes the approximating cost functions $f^*(\varepsilon,x)$ to be discontinuous in *x*. Consequently, $f^*(\varepsilon,\bullet)$ is discontinuous, and a descent direction for $f^*(\varepsilon,\bullet)$ may not be a descent direction for $f(\bullet)$. Therefore, optimisation algorithms can terminate at points that are non-optimal. In general, even if the optimisation terminates at a point that is non-optimal for $f(\bullet)$, the obtained system performance can be expected to be better than the one that would be found without any optimisation. However choosing the proper algorithm can significantly reduce this risk.

2.2.4 Multi-objective optimisation problems

Multi-objective optimisation deals with the minimisation of a vector of objectives $F(x)$ that can be the subject of a number of constraints or bounds [24]:

$$\min_{x \in \Re^n} F(x)$$

subject to: $G_i(x) = 0, i = 1,\mathrm{K}\ ,k_e;$

$$G_i(x) \le 0, i = k_e + 1,\mathrm{K}\ ,k;$$

$$l \le x \le u$$

Since F(x) is a vector of competing components, there is no unique solution to the problem and the concept of non-inferiority [25] (also called Pareto optimality [26]) must be introduced to characterize the objectives. A non-inferior solution is one in which an improvement in one objective requires a degradation of another. To better understand this concept, consider an element $\boldsymbol{x}$ in the feasible region $\boldsymbol{\Omega}$ of the parameter space:

$$\Omega = \{x \in \Re^n\}$$

subject to: $G_i(x) = 0, i = 1,\mathrm{K}\ ,k_e;\quad G_i(x) \le 0, i = k_e + 1,\mathrm{K}\ ,k;\quad l \le x \le u$

The corresponding feasible region for the objective function space **Λ** is defined as follows:

$$\Lambda = \left\{ y \in \Re^m : y = F(x), x \in \Omega \right\}$$

As pictured in figure 2.2, the performance vector F(x) maps the parameter space into the objective function space.

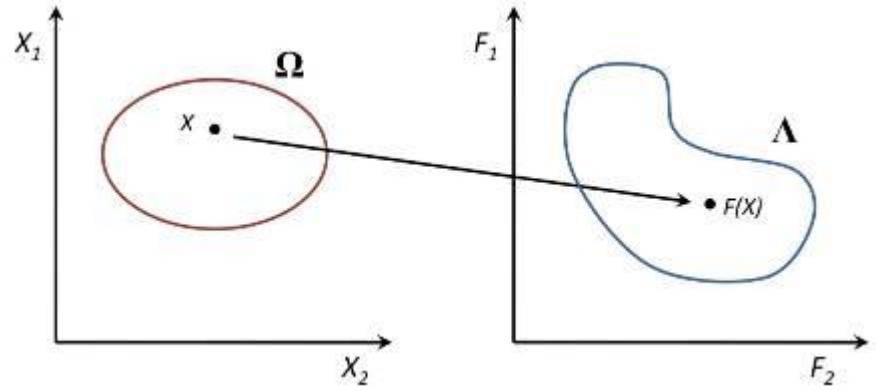


Figure 2.2 – Mapping from parameter space into objective function space

A point $x^* \in \Omega$ is defined as a non-inferior solution if for some neighbourhood of x^* there does not exist a Δx such that:

$(x^* + \Delta x) \in \Omega$ and

$F_i(x^* + \Delta x) \le F_i(x^*) i = 1,\mathrm{K}\ ,m$ and

$F_j(x^* + \Delta x) < F_j(x^*)$ for at least one *j*.

Figure 2.3 shows the set of non-inferior solutions in a two-dimensional representation: that is the curve the lies between points C and D.

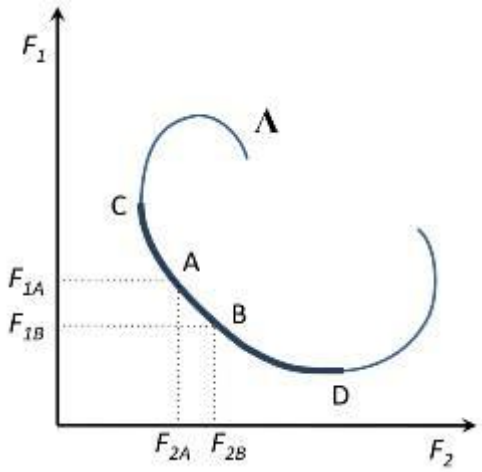


Figure 2.3 – Set of non-inferior solutions

Points A and B are examples of non-inferior points, because an improvement in one objective requires a degradation in the other: $F_{1B} < F_{1A}$, $F_{2B} > F_{2A}$.
Multi-objective optimisation is concerned with the generation and selection of non-inferior solution points, also called Pareto optima.

2.3 Single-objective vs. multi-objective optimisation

2.3.1 Multi criterion decision making (MCDM)

MCDM methods are used to support decision makers facing problems involving multiple criteria. Since there does not exist a unique optimal solution to such problems that can be obtained without incorporating preference information, it is necessary to evaluate the impact of the trade-off between the different design criteria on the design solutions.
The process has two elements [27]:

1. *Decision* as to which trade-off between the criteria has to be used;
2. *Search* for one or more solutions that reflect the desired trade-off.

The relationship between decision and search has three forms:

- *A priori*: decide first, then search; leads to one optimum design solution (the single point in figure 2.4);
- *Progressive*: decide and search concurrently; leads to various optimum design solutions (multiple colored points in figure 2.4);
- *A posteriori*: search first, then decide; leads to a complete set of optimum design solutions (the trade-off curve in figure 2.4).

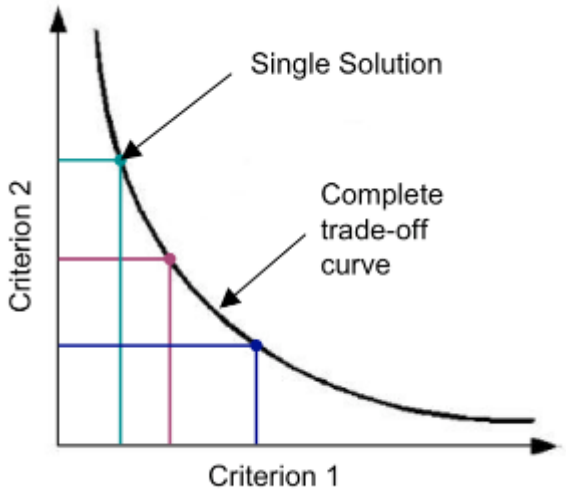


Figure 2.4 – Representation of MCDM for two criteria

The concept of an optimal solution is often replaced by that of non-dominated solutions. A non-dominated solution has the property that it is not possible to move away from it to any other solution without sacrificing in at least one criterion. Therefore, it makes sense for the decision maker to choose a solution from the non-dominated set. Otherwise, he could do better in terms of some or all of the criteria, and not do worse in any of them.

2.3.2 Comparison of the two techniques

In single-objective optimisation, there is an "*a priori"* relationship between decision and search. The objective function is made up by the weighted sum of the different criteria:

$$f(x) = w_1 \cdot f_1(x) + w_2 \cdot f_2(x) + \mathrm{K} + w_n \cdot f_n(x)$$

This means that the search is for a single optimal design solution, which represents the chosen trade-off between the different criteria.

On the other hand, multi-objective techniques involve an objective function for each criterion implicated and perform a search for a set of possible optimal solutions, each corresponding to a different trade-off between the objectives. In this case the decision as to which solution to adopt is made *"a posteriori"*.

The most used method to determine the trade-off curve is the *Pareto Theory*. A ranking scheme is employed to find the non-dominated solutions: the ranking of a solution represents the number of solutions that have a lower value in both criteria (figure 2.5). The non-dominated solutions are the ones indicated by a ranking of zero and they make up the Pareto set of solutions, also called Pareto Front (figure 2.6).

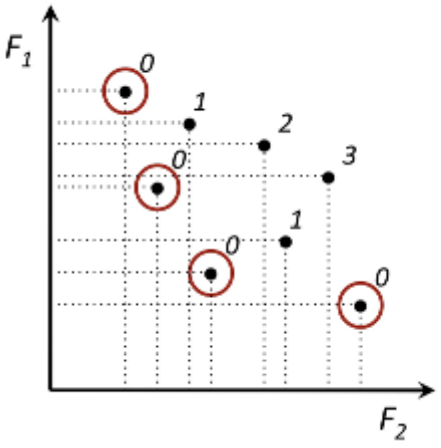


Figure 2.5 – Pareto ranking scheme

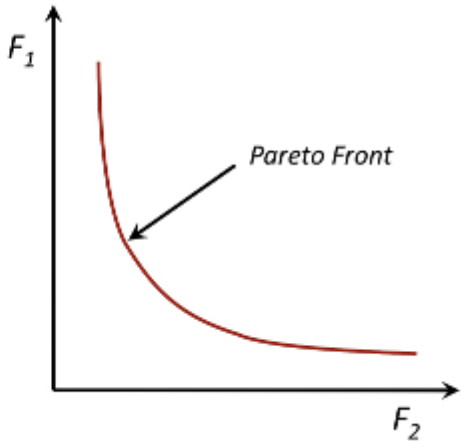


Figure 2.6 – Pareto Front

In the field of building design, where many complicated and interrelated issues are involved, it is usually not advisable to use weighting factors for the different criteria because a lot of assumptions would have to be made and only one or a small number of solution would be found. Instead, it would be much more beneficial to provide the

decision makers with various good alternatives to choose from during the early design stages. That's the reason why multi-objective optimisation techniques are more appropriate.

2.3.3 Algorithm selection

There are many available algorithms that can be used for a large variety of optimisation problems: the choice of the most suitable one to handle the optimisation process depends on the number and type (continuous and/or discrete) of variables involved, and on the way the objective function is calculated.
In simulation based optimisation, where external dynamic simulations are employed to compute the value of the objective function, the latter is highly discontinuous and non differentiable. Therefore it is essential to use algorithms that can cope with these characteristics of the objective function when solving the optimisation problem. Evolutionary algorithms have proved to be particularly suitable in this field, and they offer the additional advantage of their capability in handling huge amounts of variables and potential solutions. Genetic algorithms are the most common in this family of population based probabilistic algorithms, which includes also PSO (Particle Swarm Optimisation) and ENN (Evolutionary Neural Networks). Wetter e Wright demonstrated the suitability of probabilistic optimisation algorithms to treat non-smooth, simulation based optimisation problems [28].

2.4 Genetic Algorithms

2.4.1 Brief history

The idea to apply Darwin's theories of evolution on optimisation tools for engineering problems initiated in the 1950s and 1960s. There were several independent projects, but the common idea was to evolve a population of candidate solutions to a given problem using operators inspired by natural selection and genetic variation [29].
John Holland is recognised as the inventor of Genetic Algorithms (GAs), which he developed with his colleagues at the University of Michigan in the 1960s and 70s. His main goal was to find a general way to import the mechanisms of natural adaptation into computer systems. He explained these concepts in his book "Adaptation in Natural and Artificial Systems", where he presented the genetic algorithm as an abstraction of biological evolution.

2.4.2 Basic principles

Genetic algorithms are population based probabilistic methods based on natural selection and genetic recombination, the processes that drive biological evolution. A *population* of individuals (possible solutions) is first randomly generated and then repeatedly modified through *genetic operators*. At each step, the GA selects

individuals from the current population to be *parents* based on their *fitness function* value, and uses them to produce the *children* for the next *generation*. Over successive generations, the population "evolves" toward an optimal solution. The crucial steps of the algorithm operation are outlined in figure 2.7.
The main differences in comparison to a classical, derivative-based, optimisation algorithm are that the GA generates a population of points at each iteration and that it selects the next generation by computation using random number generators.

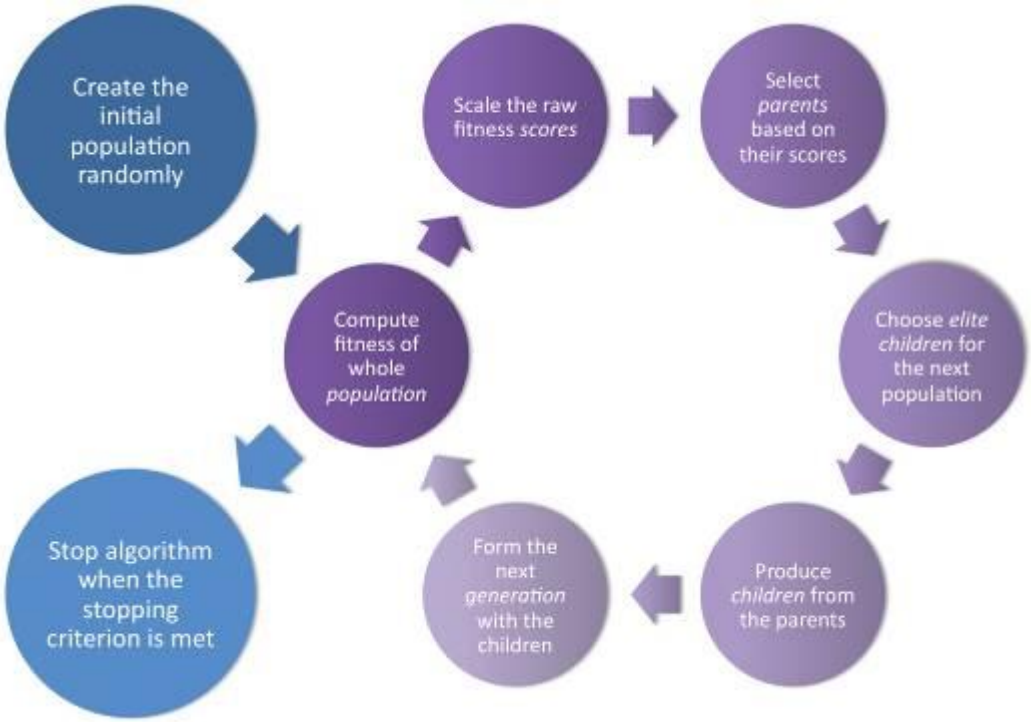


Figure 2.7 – Outline of GA operation

2.4.3 Terminology

- *Fitness function* – the objective function that has to be minimised;
- *Individual* – any point to which the fitness function can be applied, so any possible solution to the problem;
- *Score* – the value of the fitness function of a particular individual;
- *Genome* (or *Chromosome*) – the "genetic" information contained in any individual, i.e. the values of the variables;
- *Gene* – the entries of the genome, i.e. each variable encoded;
- *Population* – an array of individuals;
- *Generation* – each newly formed successive population;
- *Parents* – the individuals selected to create the next generation;
- *Children* – the individuals that will form the next generation;

- *Elite children* – individuals of the current generation with the best score;
- *Diversity* – the average distance between individuals in a population.

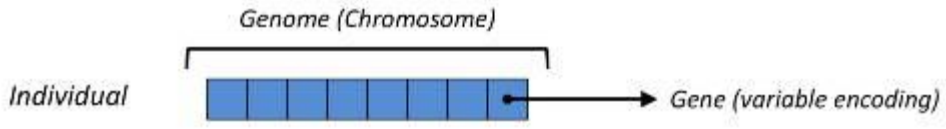


Figure 2.8 – Genome and genes

2.4.4 Genetic operators

There are three main types of rules that are brought into play at each step to create the next generation from the current population. Their procedure mimics some processes of biological evolution, this is why they are called genetic operators.

- *Selection* rules select the individuals, called parents, that will build the population of the next generation;
- *Crossover* rules combine two parents to form children for the next generation;
- *Mutation* rules apply random changes to individual parents to form children.

Each operator can implement different types of rules and can have some parameters to adjust. Which ones are the most appropriate for a specific problem is very difficult to know; however some general guidelines can be followed.

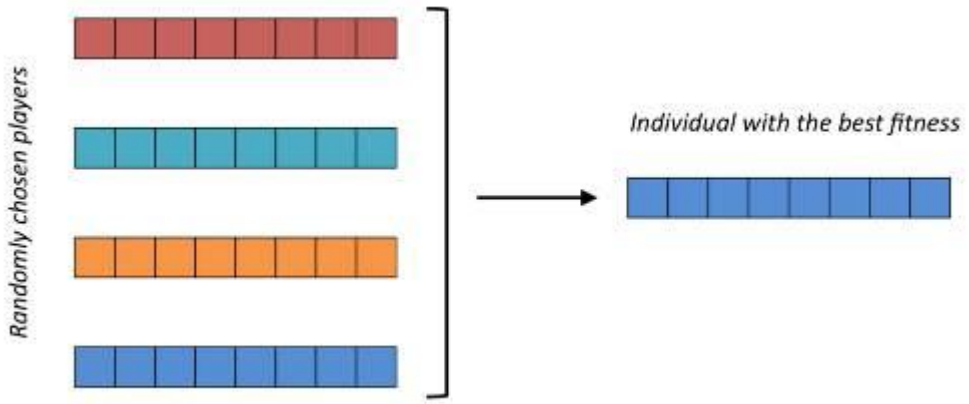


Figure 2.9 – Selection rule: Tournament with size 4

Figure 2.9 illustrates the working of a *Tournament* type selection rule, where four individuals (the size can be changed) are randomly picked from the current population and evaluated against each other in terms of their score. The one with the higher *score* will be selected as a parent for the next generation. The next figure explains how children are created by the different operators. The *crossover* and *mutation* operators can involve many different rules as how to create the children, the one pictured are just an example.

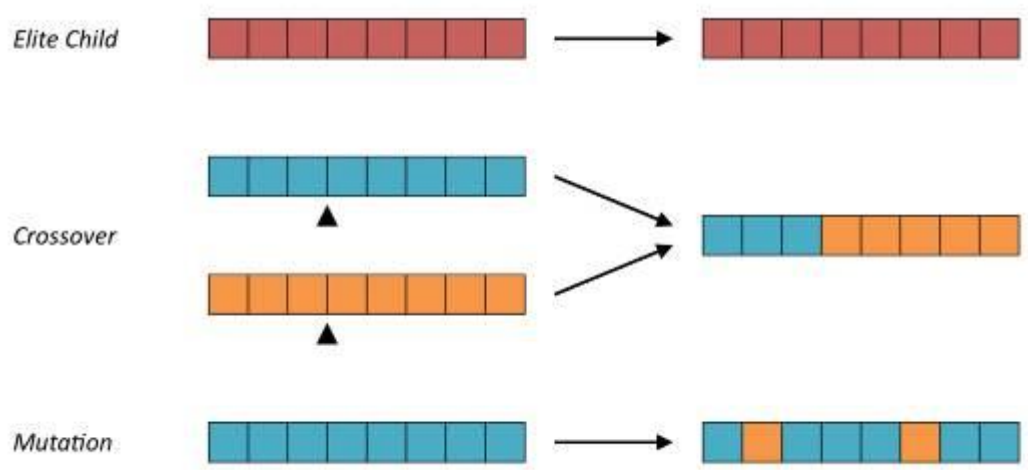


Figure 2.10 – Creation of children

2.5 The energy simulation program: *EnergyPlus*

2.5.1 How the program works

EnergyPlus [30] is an energy simulation program for the analysis of the behaviour of buildings in applications that concern heating, cooling, lighting, ventilation, and any other source that takes part in the energy balance. It is written in the Fortran 90 programming language and owes many of its peculiar characteristics to programs such as Blast and DOE-2 which laid the foundations for energy modelling since the 80's. The aim of this thesis is not to analyse in detail the way the software work, therefore only a brief description of its main features will follow.

The software is made up by a modular system where different parts interact with each other to evaluate the energy requirements of buildings, by means of dynamic simulations that take into account different environmental and operating conditions. Such parts do not interfere in the calculation, but take part in it only when the simulation demands it; this is the reason for the availability of different solution algorithms. The core of the simulation is the energy model, which is based on the fundamental principles of thermal balance. The model is assisted by a control system to handle the great amount of data required to simulate the high number of combinations of systems and plants in relation to different environmental conditions.

EnergyPlus is a simulation engine and does not incorporate any graphical user interface neither for the input of data nor for the visualization of the output. These functions can be carried out by a number of different third party programs that provide user interfaces. The input model consists basically of text files which are interpreted by the *Simulation Manager*, that can also interact with external modules to interpret data coming from different sources. This broad-spectrum architecture of the program is explained in figure 2.11.

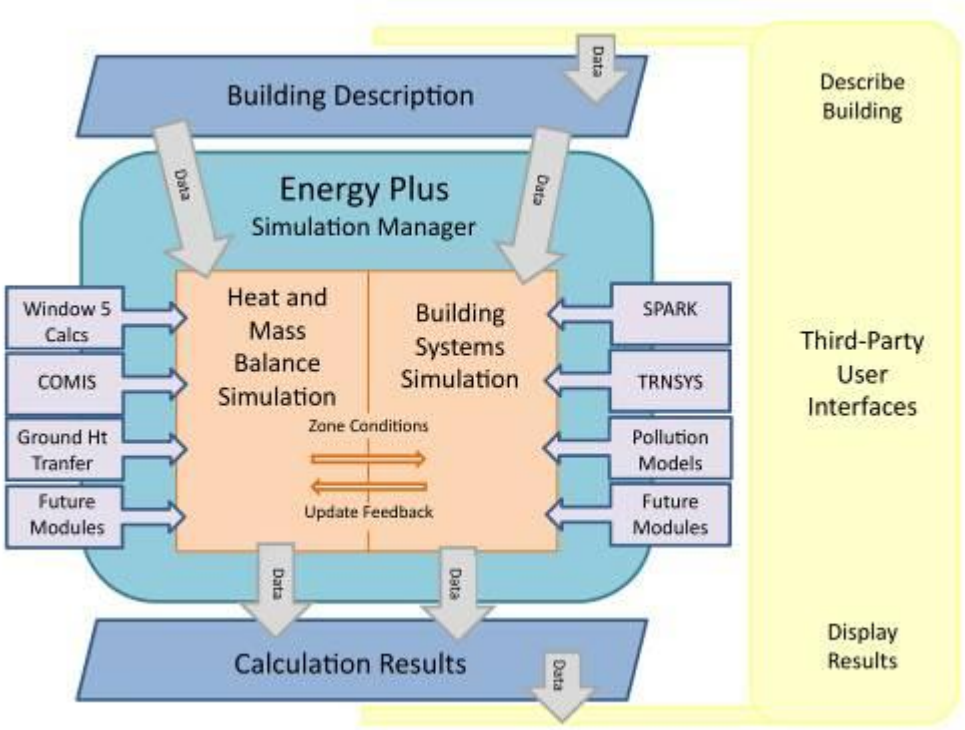


Figure 2.11 – General interaction of modules in *EnergyPlus*

Differently from other simulation programs, *EnergyPlus* doesn't calculate the required thermal loads prior to the plant simulation, but at every cycle it makes use of the values obtained from the latter to compute the new loads required for the next cycle. This basically means that it runs an integrated simulation, where all the major parts, i.e. building, system and plant, are solved simultaneously to obtain a physically realistic simulation. In programs with sequential simulations, such as BLAST or DOE-2, the building zones, air handling systems, and central plant equipment are simulated sequentially with no feedback from one to the other. The sequential solution begins with a zone heat balance that updates the zone conditions and determines the heating/cooling loads at all time steps. This information is fed to the air handling simulation to determine the system response, but that response does not affect zone conditions. Similarly, the system information is passed to the plant simulation without any feedback. This simulation technique works well when the system response is a well-defined function of the air temperature of the conditioned space.

However, in most situations the system capacity is dependent on outside conditions and/or other parameters of the conditioned space. In sequential simulation methods the lack of feedback from the system to the building can lead to non-physical results.

In order to obtain a simulation that is physically realistic, the elements have to be linked in a simultaneous solution scheme. The entire integrated program can be represented as a series of functional elements connected by fluid loops. In *EnergyPlus* all the elements are integrated and controlled by the *Integrated Solution*

Manager. The loops are divided into supply and demand sides, and the solution scheme generally relies on successive substitution iteration to reconcile supply and demand, using the Gauss-Seidell philosophy of continuous updating.

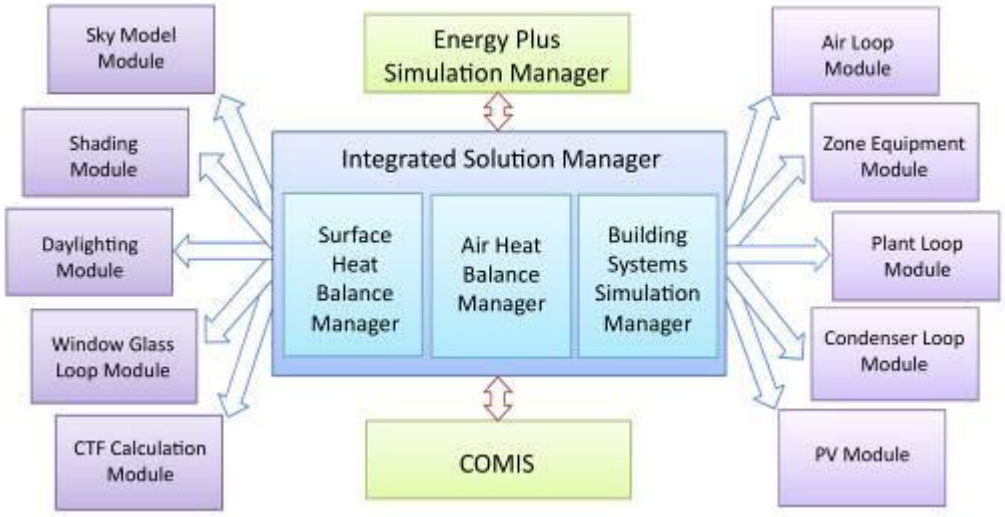


Figure 2.12 – *EnergyPlus* program schematic

The basis for the zone and system integration incorporates a shortened simulation time step, typically between 0.1 and 0.25 hours, and uses a time-marching method having the zone conditions lagged by one time step. The error associated with this approach depends significantly on the time step. The smaller the step size the smaller the error, but the longer the computation time. In order to permit increasing the time step as much as possible, while retaining stability, zone air capacity was also introduced into the heat balance.

The resulting method is called "lagging with zone capacitance". Although requiring substantially more time to execute than sequential simulation methods, the improved realism of the simultaneous solution of loads, systems and plant simulation is desirable.

The method of lagging with zone capacitance uses information from previous time steps to predict system response and update the zone temperature at the current time. The time constant, τ, for a zone is on the order of:

$$\tau \approx \frac{\rho V c_p}{\left| \dot{Q}_{load} + \dot{Q}_{sys} \right|} \qquad \text{(eq. 2.1)}$$

where the numerator is the zone air heat capacitance and the denominator is the net rate of heat energy input. The value of τ can vary because the zone load and system output change throughout the simulation. Therefore, a variable adaptive time step shorter than one hour is used for updating the system conditions. For stability reasons it was necessary to derive an equation for the zone temperature that included the

unsteady zone capacitance term and to identify methods for determining the zone conditions and system response at successive time steps. The formulation of the solution scheme starts with a heat balance on the zone.

$$C_z \frac{dT_z}{dt} = \sum_{i=1}^{N_{sl}} \dot{Q}_i + \sum_{i=1}^{N_{surfaces}} h_i A_i (T_{si} - T_z) + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p (T_{zi} - T_z) + \dot{m}_{\inf} C_p (T_\infty - T_z) + \dot{Q}_{sys}$$

(eq. 2.2)

where:

$\sum_{i=1}^{N_{sl}} \dot{Q}_i$ is the sum of the convective internal loads,

$\sum_{i=1}^{N_{surfaces}} h_i A_i (T_{si} - T_z)$ is the convective heat transfer from the zone surfaces,

$\dot{m}_{\inf} C_p (T_\infty - T_z)$ is the heat transfer due to infiltration of outside air,

$\sum_{i=1}^{N_{zones}} \dot{m}_i C_p (T_{zi} - T_z)$ is the heat transfer due to interzone air mixing,

$\dot{Q}_{sys}$ is the system output and

$C_z \frac{dT_z}{dt}$ is the energy stored in the air.

If the air capacitance is neglected, the steady state system output is:

$$-\dot{Q}_{sys} = \sum_{i=1}^{N_{sl}} \dot{Q}_i + \sum_{i=1}^{N_{Surfaces}} h_i A_i (T_{si} - T_z) + \sum_{i=1}^{N_{Zones}} \dot{m}_i C_p (T_{zi} - T_z) + \dot{m}_{\inf} C_p (T_\infty - T_z) \qquad \text{(eq. 2.3)}$$

Air systems provide hot or cold air to the zones to meet heating or cooling loads. The system energy provided to the zone, $\dot{Q}_{sys}$, can thus be formulated from the difference between the supply air enthalpy and the enthalpy of the air leaving the zone:

$$\dot{Q}_{sys} = \dot{m}_{sys} C_p (T_{\sup} - T_z) \qquad \text{(eq. 2.4)}$$

Equation 4 assumes that the zone supply air mass flow rate is exactly equal to the sum of the air flow rates leaving the zone through the system return air plenum and being exhausted directly from the zone. If equation 2.4 is substituted into equation 2.2, we have:

$$C_z \frac{dT_z}{dt} = \sum_{i=1}^{N_{sl}} \dot{Q}_i + \sum_{i=1}^{N_{surfaces}} h_i A_i (T_{si} - T_z) + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p (T_{zi} - T_z) + \dot{m}_{\inf} C_p (T_\infty - T_z) + \dot{m}_{sys} C_p (T_{\sup} - T_z)$$

(eq. 2.5)

The sum of zone loads and system output now equals the change in energy stored in the zone. In order to calculate the derivative term, a finite difference approximation may be used, such as:

$$\frac{dT}{dt} = (\delta t)^{-1}\left(T_z^t - T_z^{t-\delta t}\right) + O(\delta t) \qquad \text{(eq. 2.6)}$$

The use of numerical integration in a long time simulation is a cause for some concern due to the potential build-up of truncation error over many time steps. In this case, the finite difference approximation is of low order that further aggravates the problem. However, the cyclic nature of building energy simulations should cause truncation errors to cancel over each daily cycle, so that no net accumulation of error occurs, even over many days of simulation. All the terms containing the zone mean air temperature were then grouped on the left hand side of the equation. Since the remaining terms are not known at the current time, they were lagged by one time step and collected on the right hand side; therefore the formula for updating the zone mean air temperature becomes:

$$C_z \frac{T_z^t - T_z^{t-\delta t}}{dt} + T_z^t\left(\sum_{i=1}^{N_{surfaces}} h_i A_i + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p + \dot{m}_{\inf} C_p + \dot{m}_{sys} C_p\right) =$$
$$\sum_{i=1}^{N_{sl}} \dot{Q}_i^t + \dot{m}_{sys} C_p T_{\sup ply}^t + \left(\sum_{i=1}^{N_{surfaces}} h_i A_i T_{si} + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p T_{zi} + \dot{m}_{\inf} C_p T_\infty\right)^{t-\delta t} \qquad \text{(eq. 2.7)}$$

One final arrangement is to move the lagged temperature in the derivative approximation to the right side of the equation:

$$T_z^t = \frac{\sum_{i=1}^{N_{sl}} \dot{Q}_i^t + \dot{m}_{sys} C_p T_{Supply}^t + \left(C_z \frac{T_z}{\delta t} + \sum_{i=1}^{N_{surfaces}} h_i A_i T_{si} + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p T_{zi} + \dot{m}_{\inf} C_p T_\infty\right)^{t-\delta t}}{\frac{C_z}{\delta t} + \left(\sum_{i=1}^{N_{surfaces}} h_i A_i + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p + \dot{m}_{\inf} C_p + \dot{m}_{sys} C_p\right)}$$

(eq. 2.8)

Equation 2.8 could be used to estimate zone temperatures.
The simulation follows a Predictor/Corrector process, which can be briefly summarised in these three steps:

1) using equation 2.3, an estimate is made of the system energy required to balance the equation with the zone temperature equal to the setpoint temperature;
2) with that quantity as a demand, the system is simulated to determine its actual supply capability at the time of the simulation;

3) the actual system capability is used in equation 2.8 to calculate the resulting zone temperature.

2.5.2 The use of energy models within an optimisation process

As previously stated, an energy model to be used as input for *EnergyPlus* is essentially a text file that contains objects written with a certain syntax to allow the program to read it and decode it. The extension used to identify these input files is "**.idf**". Irrespective of the method used to build it (third party interface, "IDF editor" in *EnergyPlus*, or plain text editor), the file contains information on every component included in the model in the form of simple lines of text.
When it comes to including simulations into an optimisation process, the input file representing the model is required to provide a means to identify within its components the variables of the problem which will be iteratively changed during the procedure every time a simulation of a different model is called for. This brings forth the necessity for the model to undergo a step of manual preparation in order to replace the value of any input item that represents a variable with some kind of identifier. A function or script will then be employed every time a simulation is needed to search for this identifier inside the input text file and to replace it with the actual value that the variable assumes in that specific case.

2.5.3 The use of *Input Macros* to improve flexibility in *EnergyPlus* input files

The *Input Macros* feature provided by the program allows to increase the flexibility of the input files in different ways. These include the following capabilities:

- incorporating external files containing pieces of IDF into the main *Energy Plus* input stream;
- selectively accepting or skipping portions of the input;
- defining a block of input with parameters and later referencing this block;
- performing arithmetic and logical operations on the input.

These capabilities are invoked in the EP-MACRO program [31] by inserting macro commands in the input file. Macro commands are preceded by "##" to distinguish them from regular input commands. To let *EnergyPlus* know that the input file contains *Input Macros*, the file extension must be changed from ".idf" to "**.imf**". In this way the EP-MACRO processor is called first and its execution produces an IDF file where the macro commands are converted into regular lines of *EnergyPlus* input; at this point the solver is called and supplied with this resultant IDF input file.
For the advantage they can bring into simulation based optimisation processes, the most important of these features are the inclusion of external files into the main input stream and the option to define and later reference any block of input.

The **##include** command puts all the lines of an external file into the *EnergyPlus* input stream starting right after the command line. The name of the file that is included is the concatenation of {prefixPathName}, entered using **##fileprefix** , and {fileName.idf}. When all the lines in the external file have been read in, input reverts back to the original input file at the line following the **##include** command. Thus the use of the following commands in the main input file:

##fileprefix {prefixPathName}
##include {fileName.idf}

will incorporate in it the file whose full name is "prefixPathName/fileName.idf".

The **##def** command allows a block of input text to be defined and given a name. The block of text can then be inserted anywhere in the *EnergyPlus* input stream by simply referencing the name of the block. The block can have parameters (also called arguments) that can be given different values each time the block is referenced. The following syntax defines a macro with the name "macroName" and arguments "arg1" through "arg*n*". "MacroText" is one or more lines of *EnergyPlus* input text.

##def macroName [arg1,..,arg*n*]
MacroText
##enddef

The next command is the same as **##def** but there are no arguments and there is only one line of text so that the terminating command is not required.

##set1 macroName MacroText

To reference, and thus insert, any block of input anywhere in the input stream, it is enough to write its name followed by the arguments (if any) in square brackets.
The benefits that can be brought by these features to the exchange of data between the simulation side and the optimisation side of the process are obvious. For example the model can be broken down into different parts, or "data sets", that will then be included into the main input file, allowing for a better organisation of the components of the model. Blocks of input, or just a single parameter, can be defined at the beginning of the file and then referenced anywhere. In this way, changing the definition of an item causes the same change to happen wherever that item is referenced throughout the whole input stream. Combining these two features it is possible to create a data set with the definition of all those input parameters that have been chosen as variables, and then tell the main input file to include it. The parameters can then be referenced in the objects of the model that represent the variables. To explain the procedure better, an example is reported in the following

figures. The first one pictures an excerpt of a data set file named “ExampleFile.idf” that contains the definition of both a block of input and a single line parameter. The second one represents the main input file where this data set is included with the appropriate command, and where the two defined items are referenced.

```
! ExampleFile.idf
! This is a DATA SET containing some definitions of input text

! This is a definition of a block of input
##def INSUL [x, y]
Material,
   Insulation,               !- Name
   Rough,                    !- Roughness
   x,                     !- Thickness {m}
   y,                     !- Conductivity {W/m-K}
   80,                       !- Density {kg/m3}
   840,                      !- Specific Heat {J/kg-K}
   0.9,                      !- Thermal Absorptance
   0.7,                      !- Solar Absorptance
   0.7;                      !- Visible Absorptance
##enddef

! This is the definition of a single parameter
##set1 SetPointTEMP  22
```

Figure 2.13 – Use of Input Macros: example of the definition of a block of input

```
! This is the main Energy Plus input stream

... OTHER ITEMS OF THE MODEL ...

! These commands incorporate the data set
##fileprefix ../../examples/dataSets/
##include ExampleFile.idf

... MORE ITEMS OF THE MODEL ...

! Reference to the block of input called "INSUL"
! This creates the item as defined and assigns the specified values to its arguments
INSUL [0.12, 0.04]

! The next item includes the reference to the single parameter
Schedule:Compact,
   Heating Setpoints Sch,
   Temperature,
   Through: 12/31,
   For: Weekdays,
   Until: 06:00,  12,
   Until: 19:00,  SetPointTEMP,
   Until: 24:00,  12;

... MORE ITEMS OF THE MODEL ...
```

Figure 2.14 – Use of Input Macros: example of inclusion of an external file

3

ePlusOpt: a *Matlab* program for *EnergyPlus* simulation based optimisation

3.1 Introduction

In optimisation problems where the objective function is calculated by an external simulation program, there is the need to configure the correct communication between the latter and the optimisation solver. The program *ePlusOpt* was developed through Matlab programming language with the specific aim to automate as much as possible the interaction between the simulation program *EnergyPlus* and Matlab's optimisation tools. Given that the user has already prepared the energy simulation model to be employed for the *EnergyPlus* simulations, the program's graphical interface allows the user to easily set up an optimisation problem, to launch the optimisation process and to automatically save the results at its end.

On the optimisation side, the program makes use of the Genetic Algorithm included in *Matlab*'s *Global Optimisation Toolbox*, which implement genetic algorithms to minimise single-objective or multi-objective functions. These objective functions are calculated for each possible solution by running the energy simulation of the corresponding model in *EnergyPlus*.

Figure 3.1 illustrates on a program level the mutual interactions between the different elements that are involved when the program is run. An in depth description of the program's structure and operation will be presented in the next paragraphs.

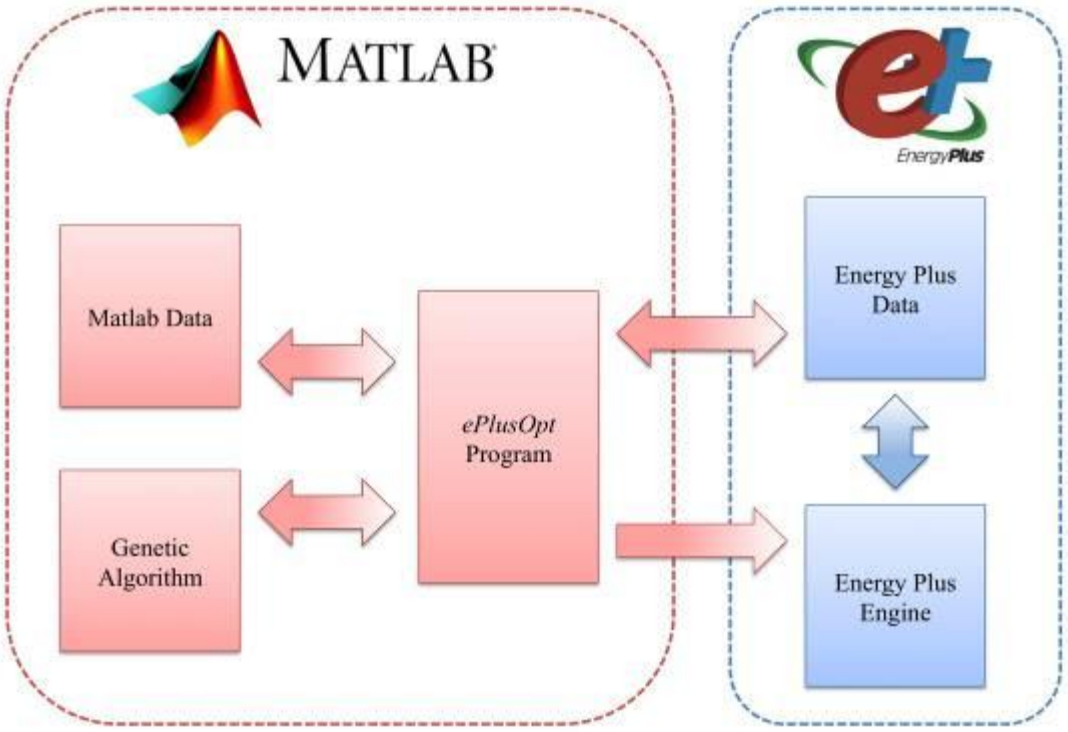


Figure 3.1 – General interactions on a program level

3.2 *ePlusOpt* as interface

The main purpose of *ePlusOpt* is to serve in simulation based optimisation problems as an interface between the optimisation side, characterized by the genetic algorithms in *Matlab*, and the simulation side, constituted by the thermal simulation engine *EnergyPlus*. By means of a graphical user interface (GUI), it allows the user to define the optimisation problem, to save its definition, and to run it within *Matlab* environment. At the end of the optimisation process, the output is automatically saved in the form of *Matlab* data but also in other formats to make it available for post-processing.

Figure 3.2 shows how the program interacts with the different agents and components involved in the process. According to inputs coming from the user through the graphical interface, it builds the needed items to set up and run an optimisation problem in *Matlab*, and to configure the user-supplied energy model to work correctly during all steps of the process. On the simulation side, it can write into *EnergyPlus* input files, start the simulation and read from the output files. On the optimisation side, it can call the Genetic Algorithm, control its progression and get the results.

The next paragraph explains how the program configures the coupling of optimisation and simulation and assures the correct execution of their interaction.

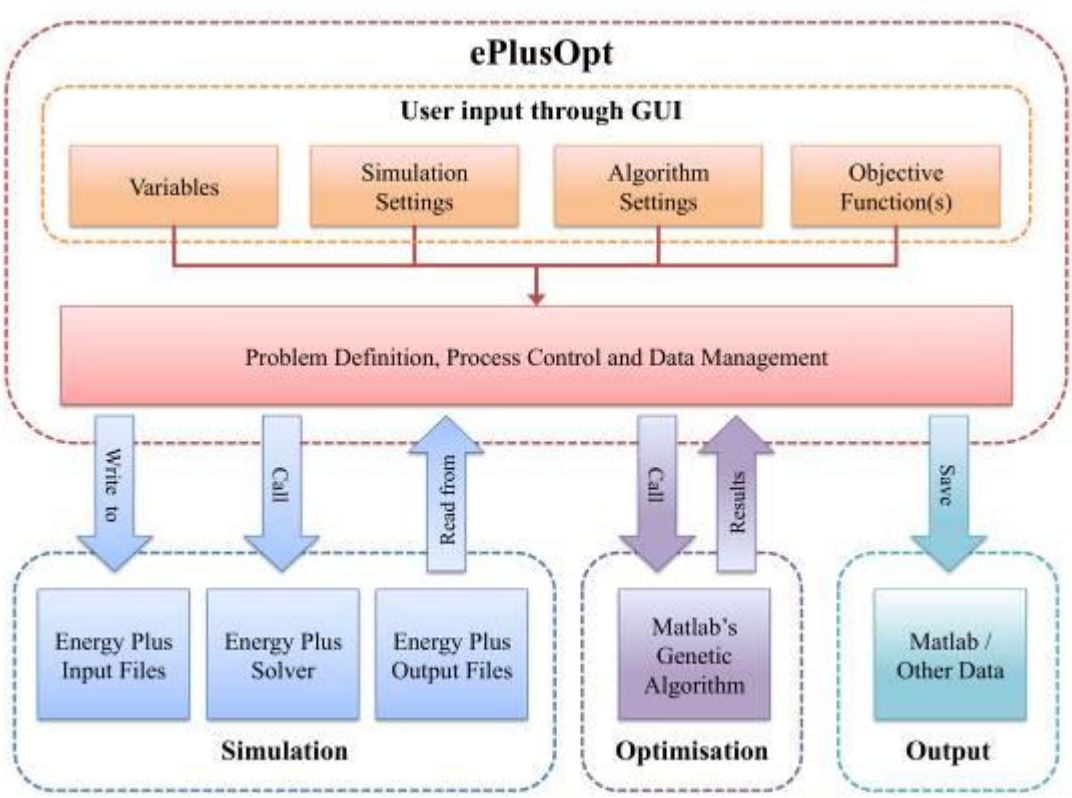


Figure 3.2 – Detailed interactions between *ePlusOpt* and the other components

3.3 The coupling of *Matlab*'s GA and *EnergyPlus*

The coupling of simulation and optimisation is managed through a series of functions and scripts that configure the communication between the GA in *Matlab* and the *EnergyPlus* software. The core of this interaction happens inside the *fitness function* which is called by the GA to compute the objectives of the optimisation. For every possible solution to the problem, hence for every individual in the population, a simulation of the corresponding energy model is necessary to obtain the values of the parameters required for the evaluation of the objective function. To achieve this, the process outlined in figure 3.3 is carried out in the *fitness function* for every individual in the population. The combination of the variables encoded in the *chromosome* of each individual is passed to a function that writes them inside a data set that is part of the energy model used for the simulations. Subsequently another *Matlab* function starts the simulation of the updated input file by calling the *EnergyPlus* executable file. This is accomplished by means of executing a system command line that was previously and automatically built by the program according to some simulation parameters entered by the user. When the simulation ends and the output files are produced, a third function retrieves from them the values of the output variables that were requested by the user. The *fitness function* finally uses these values to compute the objective(s).

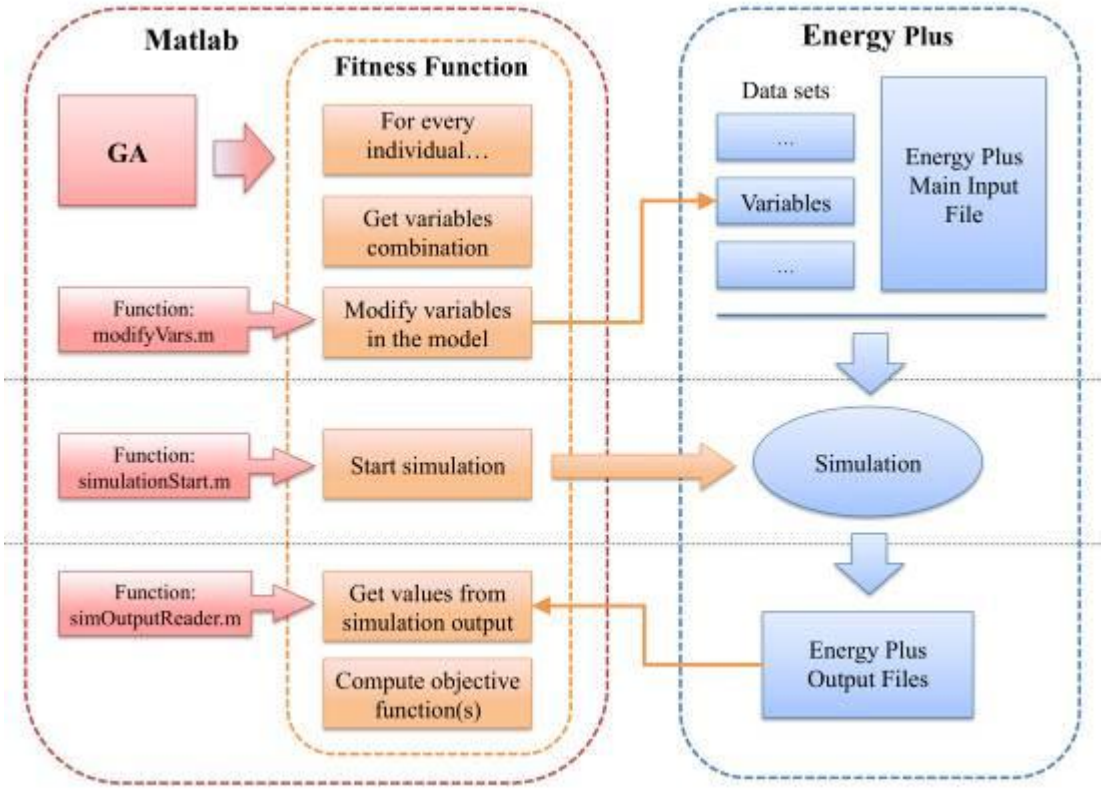


Figure 3.3 – Coupling of the GA and EnergyPlus

3.4 The object-oriented programming at the base of the program

Matlab programming language allows to work with object oriented programming, which is a very valuable tool when developing programs that need to be flexible and re-usable. The main concepts of object oriented programming are the ones of *class* and *object*. A *class* is a definition that specifies certain characteristics that all instances of the class share. These characteristics are determined by the *properties*, *methods*, and *events* that define the class and the values of *attributes* that modify the behaviour of each of these class components. Class definitions describe how objects of the class are created, what data the objects contain, and how you can manipulate this data. A class is like a template for the creation of a specific instance of the class. This instance or *object* contains actual data for a particular entity that is represented by the class. Objects are not just passive data containers. Objects actively manage the data contained by allowing certain operations to be performed. An important aspect of objects is that you can write software that accesses the information stored in the object via its properties and methods without knowing anything about how that information is stored, or even whether it is stored or calculated when queried. The object isolates code that accesses the object from the internal implementation of methods and properties. This characteristic of objects is called *encapsulation*.

The following is some basic terminology of object oriented programming and related concepts in *Matlab*:

- Class definition - Description of what is common to every instance of a class
- Properties - Data storage for class instances
- Methods - Special functions that implement operations that are usually performed only on instances of the class
- Events - Messages that are defined by classes and broadcast by class instances when some specific action occurs
- Attributes - Values that modify the behaviour of properties, methods, events, and classes
- Objects - Instances of classes, which contain actual data values stored in the objects' properties

To exploit the features of object oriented programming, some *classes* were created to be used as the base for storing and managing data in *ePlusOpt*. Figure 3.4 pictures a UML class diagram that describes these classes and their relationship. UML stands for Unified Modelling Language and it is a representation standard that comprises concepts and notations used for creating models of object oriented computer software.

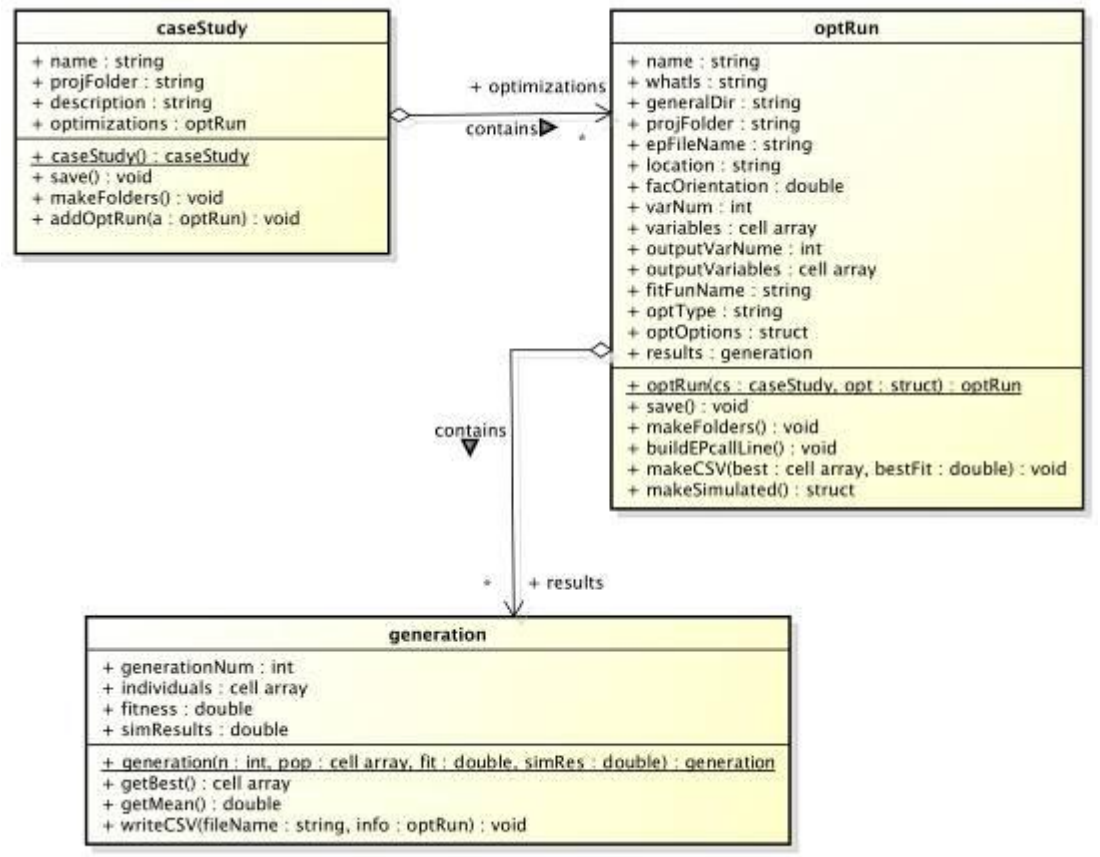


Figure 3.4 – UML class diagram

Basically, the three classes can generate objects that are designed to do the following:

- *caseStudy*: hold general information on the case study and reference any optimisation run that has been performed;
- *optRun*: store complete information on an optimisation run, comprehending simulation parameters, GA settings, variables, and, once the optimisation has been performed, the results;
- *generation*: store the results of each iteration of the optimisation process to be available for later inspection and to build the program outputs.

The hierarchy of the objects is further explained graphically in figure 3.5.
The user does not need to know how the objects are created or how the data is stored or manipulated because everything is managed automatically by the code that stands behind the graphical user interface. In case the user wants to directly access the information saved in the objects, this can be easily done in *Matlab* by loading the corresponding variables in the workspace.

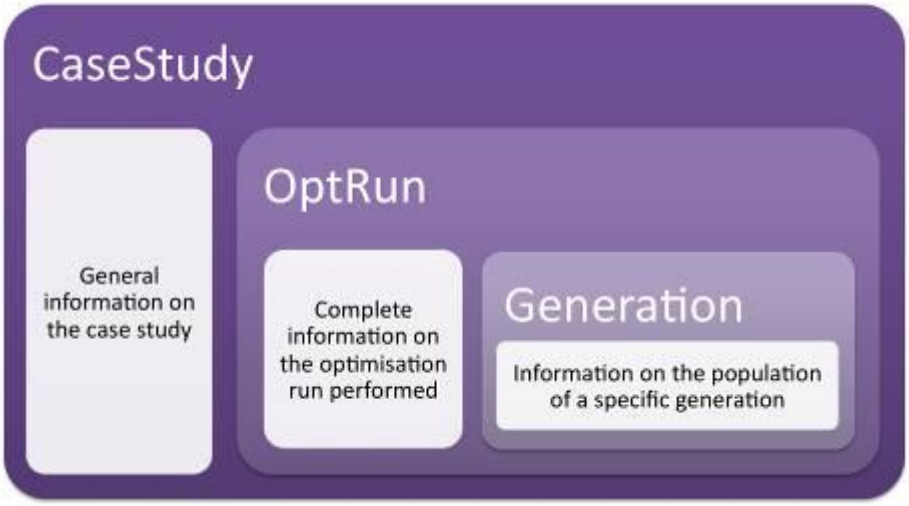


Figure 3.5 – Hierarchy of objects

3.5 Folders and files structure

The program is designed to work with a specific arrangement of folders and files contained by a main project folder. The user must be aware of this structure in order to know how to interact with the program correctly. The main folder is laid out in figure 3.6, while a detailed account of the "caseStudies" and the "energyPlus" folders is displayed in the subsequent figures.

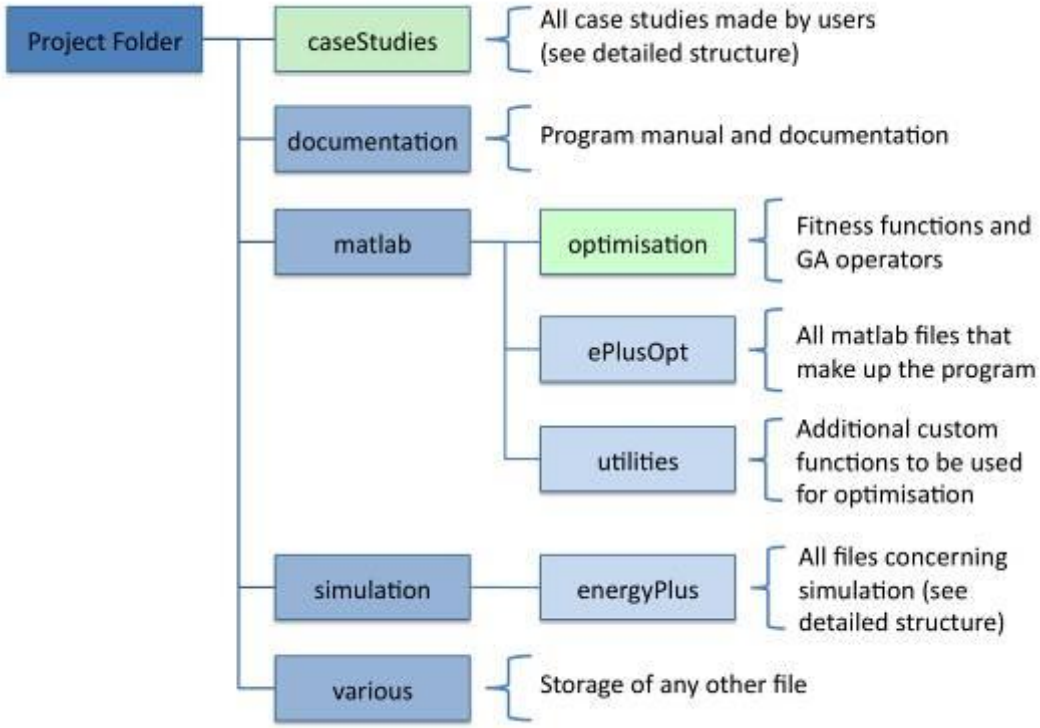


Figure 3.6 – Structure of project folder

The folders highlighted in light green colour are the ones where some kind of user interaction happens. For example the "optimisation" folder is where the *fitness functions* must be placed after having modified a template found in the same folder. Every time a new case study is made from the GUI (see paragraph 3.7), a new folder with the user supplied name is automatically created inside the "caseStudies" folder. As shown in figure 3.7 this will hold the information about the case study and all the results of the optimisations performed. Hence, once an optimisation run ends, its output files can be found inside a subfolder of the main case study folder. In particular, three types of output are generated:

- a *Matlab* variable file with extension ".m" with the results;
- a comma separated (extension ".csv") file with information about the whole optimisation process and its results;
- one or more graphs in form of a *Matlab* figure file that depend on the kind of optimisation performed (single or multi objective).

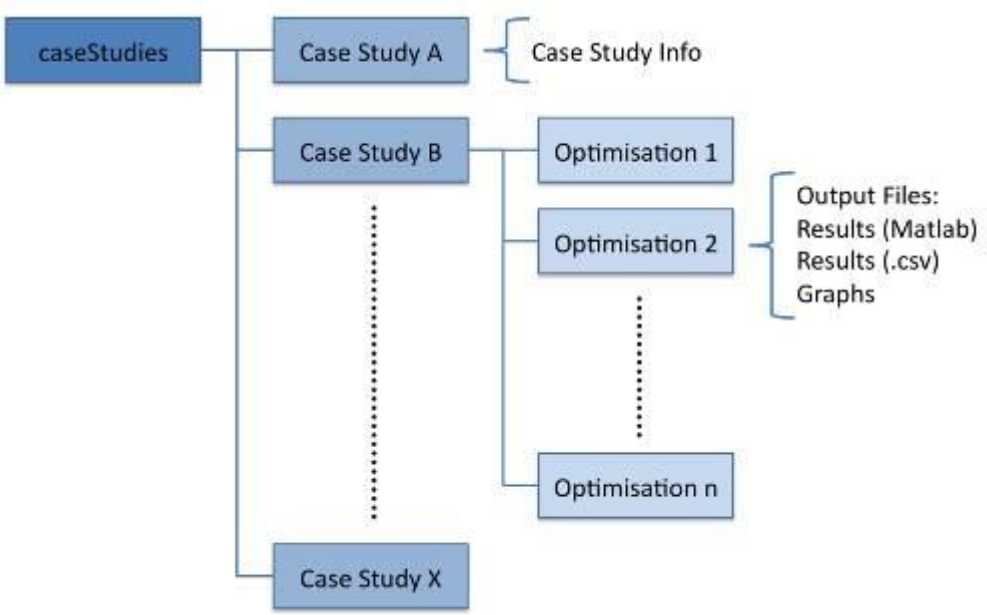


Figure 3.7 – Structure of "caseStudies" folder

At case study creation, another folder with the same name is also automatically added to the "dataSets" folder to accommodate the data set files associated with any Energy Plus model that will be employed for the specific case study. However, the main input files for the Energy Plus model must be placed inside the "inputFiles" folder, because the program will look for them in this location. The next paragraph informs on how these files must be prepared to work within the optimisation process.

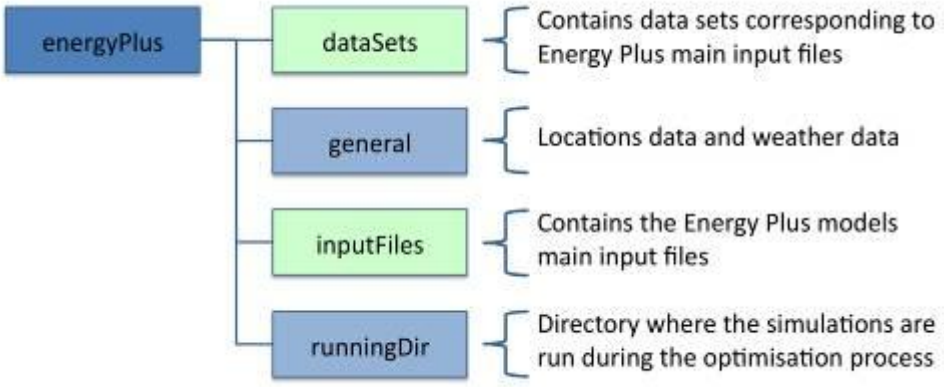


Figure 3.8 – Structure of "energyPlus" folder

3.6 Preparation of *EnergyPlus* input files to use with *ePlusOpt*

The energy model used for the *EnergyPlus* simulations is made up by one or more input text files. The users of *ePlusOpt* are required to write and arrange these text files in a specific way in order to work properly within the program. This is necessary because the read and write operations that involve these files take place by

means of low level input/output functions that are programmed to work with specific files and a defined syntax.
As seen in chapter 2, it is possible to split a basic *EnergyPlus* input file into parts (also called data sets) that get included into the main file once the application is run. Taking advantage of this feature, it is possible to declare in different files all the parameters and variables of a model that will have to be iteratively changed during an optimisation process. The benefits of doing so are an improvement in the model clarity and the fact that the function that iteratively writes the value of the variables does not have to search through the entire input text file. This is managed in *ePlusOpt* in the following way: all energy models main input files added to the previously mentioned "inputFiles" folder must reference the following two data sets contained in the case study folder that was created in the "dataSets" folder:

- 0_Parameters.idf, that will include the definition of any parameter that the user might want to change frequently before starting an optimisation;
- 0_Variables.idf, that will include the definition of all the variables of the optimisation problem.

These data set files are automatically copied from templates into the directory when the case study folder is created, and initially contain only the instructions for how they should be filled out. The "Parameters" file actually includes also the definition of two default items that can be regulated from the program GUI: the location to use for the simulations and the orientation of the building. Any other parameter and its value can be added by the user. The "Variables" file is the one that will be modified by the "modifyVars.m" function before running every simulation according to the combination of variables coming from the GA (see figure 3.3). An excerpt of both files is displayed in figures 3.9 and 3.10, along with an example of variable declaration.

```
! /////// GENERAL PARAMETERS ///////

! General Info
##set1 loc[ ]                    ! Name of the Location
##set1 bOR[ ]                    ! Building Orientation


! /////// ADDITIONAL PARAMETERS ///////

! ----- Add additional parameters below -----
! Use exclusively the following sintax:
! (for each parameter copy and paste the entire next line,
! delete the "!" at the beginning and edit "parameterName")
! ##set1 parameterName[ ]                ! Description of the parameter goes here
```

Figure 3.9 – excerpt of "0_Parameters.idf" data set file

```
! /////// VARIABLES ///////

! ----- Define the variables of the problem below -----
! Use exclusively the following sintax:
! (for each variable copy and paste the entire next line,
! delete the "!" at the beginning and edit "variableName")
! ##set1 variableName[ ]          ! Description of the variable goes here

##set1 Var1[ ]            ! This is the first variable
##set1 Var2[ ]            ! This is the second variable
```

Figure 3.10 – Excerpt of "0_Variables.idf" data set file

Please note that all the parameters declared in these files must obviously be referenced in the *EnergyPlus* objects that make use of them. For instruction on how to include data sets into the main input file and how to set and reference parameters inside *EnergyPlus* files, refer to chapter 2, end of section 2.5.

3.7 The graphical user interface (GUI)

ePlusOpt works mainly through a graphical user interface that allows users to set up the optimisation problem after having previously fulfilled some tasks. These essentially consist in writing a *Matlab* function that computes the objectives, preparing the energy simulation model and copying it into the appropriate folder. When the program is launched, a start up message (figure 3.11) warns the user about taking care of these tasks before proceeding.

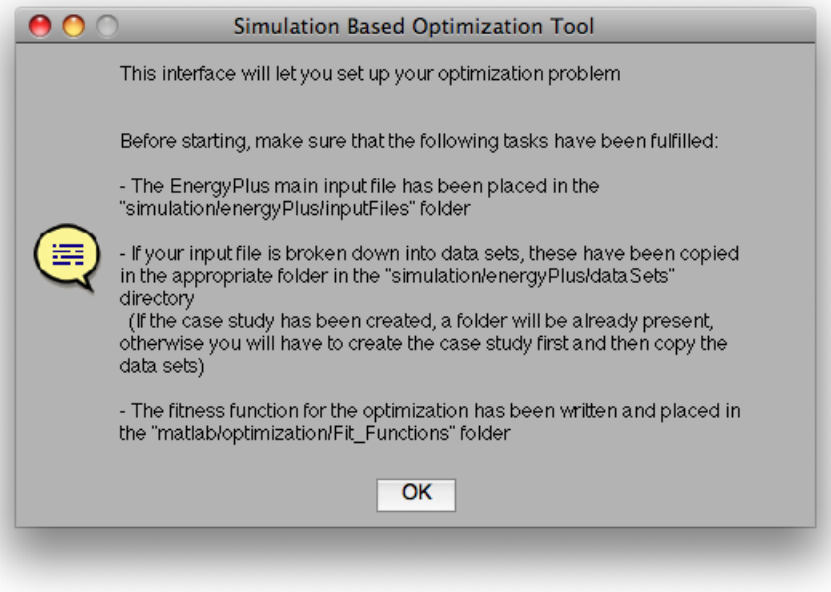


Figure 3.11 – GUI start up message

The main panel of the GUI is divided in five parts, or boxes, that group together settings of different kinds. In the following section, the features of each of these boxes are explained in detail. Once all fields have been completed, pressing the "Run Optimisation" button causes the program to perform the following tasks:

- create a new subfolder for the optimisation in the folder of the selected case study;
- save an "optRun" object with all the information entered inside this folder;
- call the GA and run the optimisation;
- at optimisation end save the results.

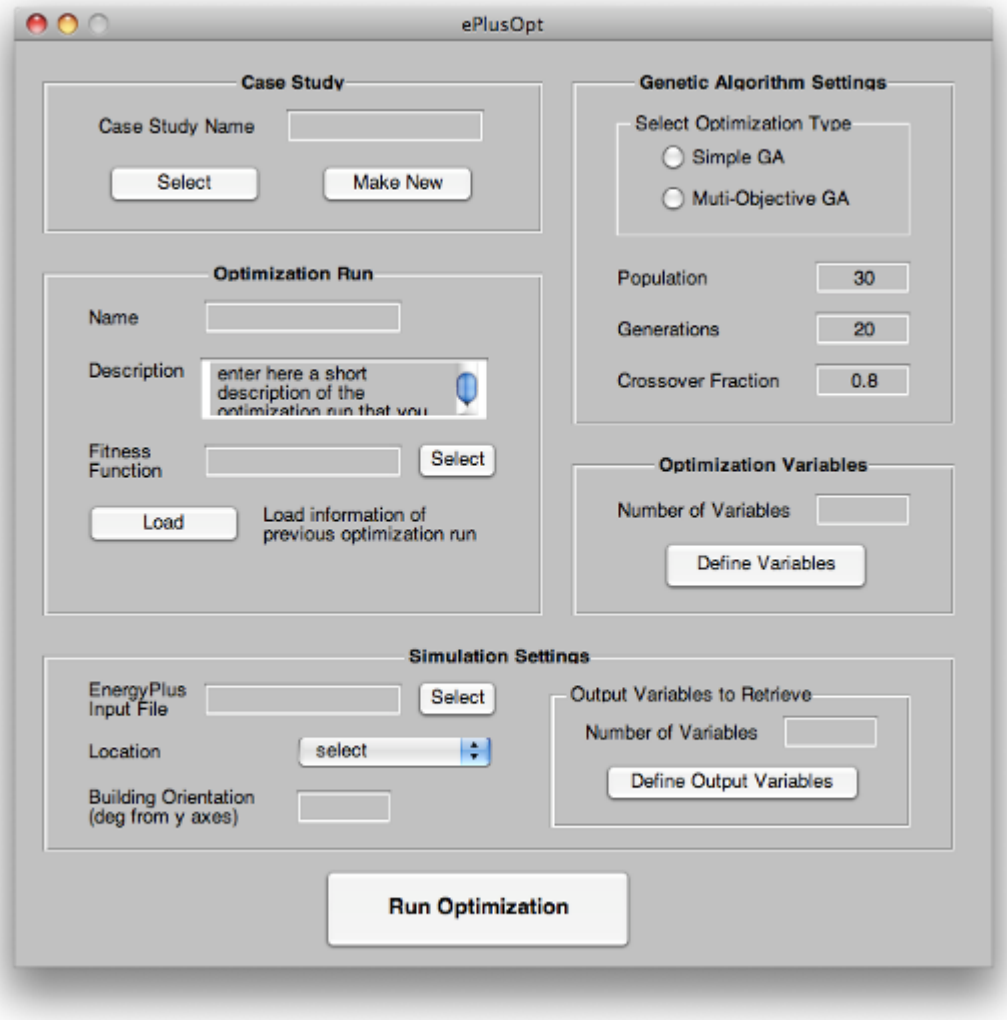


Figure 3.12 – GUI main panel

3.7.1 Case Study box

As pointed out earlier, any optimisation is part of a case study, whose folder will accommodate all the subfolders corresponding to the optimisation runs performed. In

the first box of the interface an existent case study must be selected or a new one can be created. The "Select" button opens a dialog box that lets the user choose any "caseStudy" object contained inside the main case studies folder; its contents will be automatically loaded. The "Make New" button opens an edit box (figure 3.13) that prompts the user for the information needed to create a new "caseStudy" object.

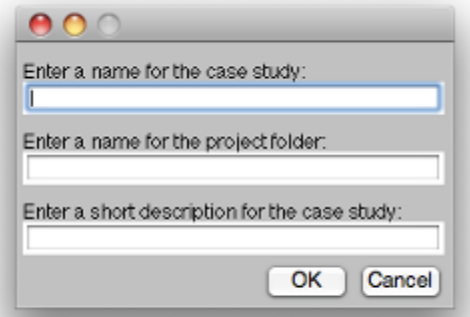


Figure 3.13 – Edit box to create a new case study

3.7.2 Optimisation Run box

Every optimisation process must be characterized by a name and a brief description. The name entered in this box will be used to identify the subfolder where the optimisation output files will be placed, and a *Matlab* variable containing the corresponding *"optRun"* object that will be saved inside this folder. The user must then select the *fitness function* to be used for the optimisation, which should have been previously prepared and placed in the appropriate folder. The "Load" button allows to load information on an optimisation run that was previously run or just defined. This functionality comes to hand when the user wants to re-run a previous optimisation with one or more altered parameters, because he doesn't have to define a new one from scratch but he can load and the modify an already existent one.

3.7.3 Simulation Settings box

In this section the energy model to be used for the simulations is chosen by selecting the main *EnergyPlus* input file prepared by the user. Moreover, two general parameters for the simulations can be indicated, namely the location and the orientation of the building. The selection of the location occurs by means of a drop-down menu that is connected to a list of the weather files that are present in the "simulation/energyPlus/general" folder. If the desired location is not in this list, it means that the corresponding weather file is missing. To provide a new location, the user must manually add the weather file to the folder and update the aforementioned list.

A further section of the box is dedicated to the definition of the output variables that the user wants to retrieve from the *EnergyPlus* output files. Entering the number of

desired output variables and pushing the "Define Output Variables" button opens the new panel pictured in figure 3.14. Any output variables that the energy models produces can be defined just by entering its complete name as reported in *EnergyPlus* output files, i.e. the ".eso", ".mtr" and ".csv" files. The type of the requested variables shall be chosen among three options ("Timestep", "Hourly" or "Run Period"), and a unique name shall be assigned to each of them. *ePlusOpt* will automatically read the values of these output variables from the simulation output files and make them available inside the *fitness function* for any processing needed to compute the objectives of the optimisation. Output variables definitions can be saved into a *Matlab* variable and loaded in any successive optimisation session.

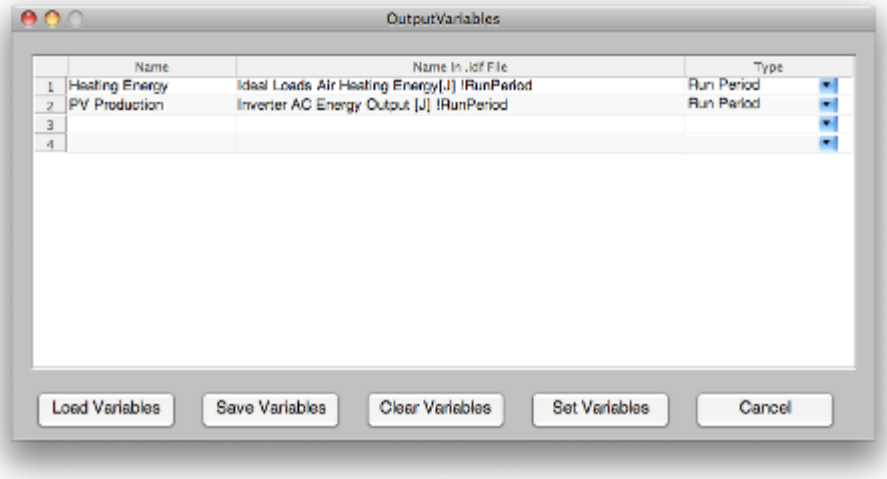


Figure 3.14 – Output variables definition panel

3.7.4 Algorithm Options box

The next chapter talks about the implementation of the GA algorithm in Matlab and describes its parameters. Although most of them are fixed for the use in *ePlusOpt*, the main characteristics can be edited directly from the user interface. These include the number of individuals in the *Population*, the number of *Generations* that are reckoned necessary to find the sought after minimum, and the *Crossover Fraction*, that is the percentage of children that will be generated through *crossover* (the remaining children will be generated through *mutation*). Another essential choice that can be made in this box is the one between a simple GA to deal with single objective optimisation problems and a multi-objective GA to manage multi objective problems. When the multi-objective GA is selected, an additional parameter can be set: the *Pareto Fraction*, or the percentage of points in the population that will be used to form the Pareto front.

In case the user wants to edit more of the parameters that define the GA behaviour, this can be done by modifying the "setOptOptions.m" function in the "optimisation" folder. The future development of the program will include the opportunity to edit more GA characteristics directly from the GUI.

3.7.5 Variables box

An additional panel that opens after entering the number of desired variables and clicking on the "Define variables" button serves as interface to define the variables of the optimisation problem. A name ought to be specified for every variable, along with the name it has in the previously described "0_Variables.idf" data set text file. This is essential to build the link between the variables in *Matlab* and the corresponding ones in the *EnergyPlus* model. Furthermore, a drop-down menu lets the user choose the type between continuous, discrete numeric, and discrete letter based. Based on this choice, the next fields shall be completed by entering a lower bound, upper bound and a step for a continuous variable, or a set of comma separated values for a discrete one.

An example of variables definition can be see in figure 3.15, where the rwo entries match the ones defined as an example in the excerpt of the "0_Variables.idf" data set file (figure X). Variable definitions can be saved in a *Matlab* file to be re-used later.

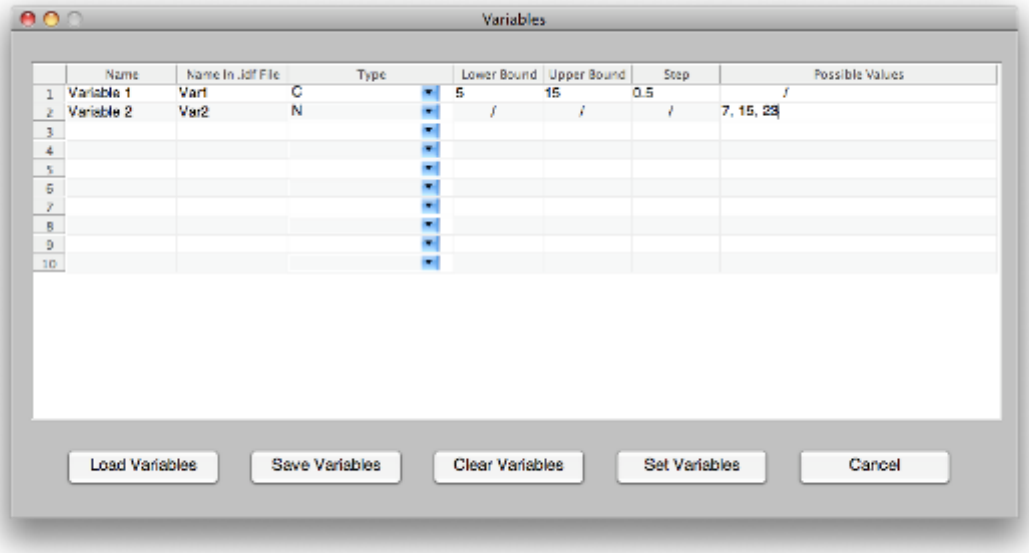


Figure 3.15 – Variables definition panel

3.8 Implementation of *Matlab*'s Genetic Algorithm

The *Global Optimisation Toolbox* comprehends a simple genetic algorithm and a multi-objective genetic algorithm. Both can be run either from a GUI or directly from the command line. In *ePlusOpt* everything is automated so the algorithms are called from the command line inside specific functions.

3.8.1 Customisation of the GA

Standard genetic algorithms in *Matlab* work only with numeric or logical data type, which means that the variables of the problem (the *genes*) can be represented only by numbers. However, the user has the option to supply the algorithm with some self

written functions in order to make it work with a custom-built data type. In particular, the following three functions, which deal with the management of the variables within the population inside the GA, must be provided:

- a *Creation Function* that generates the initial population;
- a *Mutation Function* that determines how mutation children are created;
- a *Crossover Function* that determines how cross-over children are created.

This process was carried on in the development of the *ePlusOpt* tool since the maximum flexibility in the definition of the problem's variables was sought after. The data type that could best provide this flexibility was *Matlab*'s *cell array*, a type which can hold any kind of other data types in its cells. In this way, when declaring the variables of the problem, the user can choose freely between numeric, logical, character or string data types. Using the typical expressions of evolutionary algorithms, each *gene* can be represented by any of the basic data types, and a combination of these inside a cell array makes up the *chromosome*. The custom GA operators that were implemented are the following:

- *myCreation*: a creation function that randomly generates the initial population, with the possibility to apply a "rule" on the creation;
- *myMutation*: a mutation function;
- *myXover*: a crossover function.

3.8.2 Fixed parameters of the GA

The GA has several options than can be set to control the optimisation process. In the development of *ePlusOpt*, it was decided to let the user modify only some of these options while keeping the others fixed.
In the *Population* section, the fixed parameters are the following:

- *Population Type*, specifies the data type for the variables that will be used in the fitness function: as mentioned earlier, there was the need to define a bespoke data type, so the *Custom* option is selected;
- *Creation Function*, specifies the function that creates the initial population for the GA: the custom written function "myCreation" is used, which randomly generates the population based on the user-supplied definition of the variables;
- *Initial Population*, makes it possible to supply an initial population for the GA: no initial population is provided in the current settings;
- *Initial Scores*, as there is no initial population, no initial scores are provided;
- *Initial Range*, as there is no initial population, no initial range is provided;

- *Stopping Criteria*, tells the algorithm when to stop its execution: the *Generations* option is selected, meaning that the algorithm stops only when the maximum number of generations is reached.

The following options deal with the properties of the genetic operators:

- *Selection*, identifies the method used to choose among the current population the individuals which will generate the following generation based on their "scores": the *Tournament* selection function is chosen with a value of 2 (this means that 2 individuals are randomly selected and the best one gets picked);
- *Reproduction*, determines the scheme used by the algorithm to generate the children at each iteration: the number of *elite children*, the individuals that live on to the next generation, is fixed at 2, while the *crossover fraction* has a default value of 0.8 (meaning that 80% of the new generation will be populated by children generated from cross-over operations) but is one of the parameters that the user can change (see afterwards);
- *Mutation*, specifies the function that defines how to perform the mutation of an individual to create its mutated child: the custom written function "myMutation" is used;
- *Crossover*, specifies the function that defines how to perform the cross-over between two parents to create their children: the custom written function "myXover" is used;
- *Migration*, describes the movements of the individuals among sub-populations: not active in the current settings.

3.8.3 How the GA is called inside *ePlusOpt*

When the "Run Optimisation" button is pressed in the GUI (see figure 3.12) after having entered all the needed information, the program builds an *optRun* object where it stores this information and saves it inside the automatically generated corresponding folder inside the case study main folder. Then the function *"runGAopt"* is run (see flowchart in figure 3.16), inside which the *Matlab* variables needed for the genetic algorithm are created and a number of side processes are handled. Among these, storage for the output of all simulations performed is created in order to add the capability to re-use this information instead of running another simulation when the same model needs to be evaluated for a second time during the optimisation. Then the genetic algorithm is finally called with the optimisation options and the handle to the fitness function as arguments.

Figure 3.17 illustrates the progression of the algorithm, which is the typical GA process with the only difference that the fitness function is called with the introduction of the aforementioned "*simulated*" storage. The way this new element is exploited is better understood by looking at the flowchart unfolding the fitness

function process in figure 3.18. From the same flowchart is also possible to better comprehend how the evaluation of each individual is achieved through an energy simulation (the "*simulationStart*" function) and the post-processing of the output variables retrieved from it by the "*simOutputReader*" function.
When the GA stops execution because the stopping criteria has been met, it returns its output to the "*runGAopt*" function which in turn returns it to the main process of the *ePlusOpt* program.

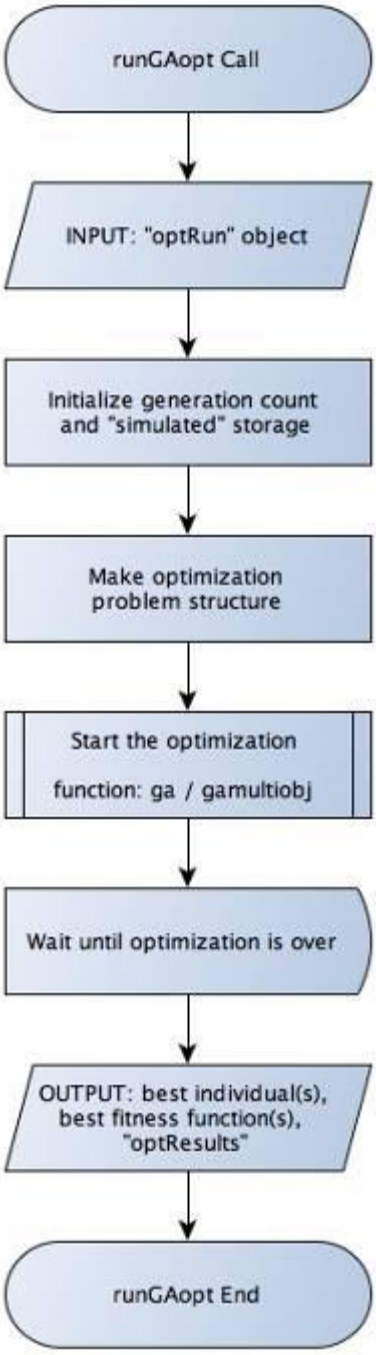


Figure 3.16 – Flowchart of "runGAopt" function inside *ePlusOpt*

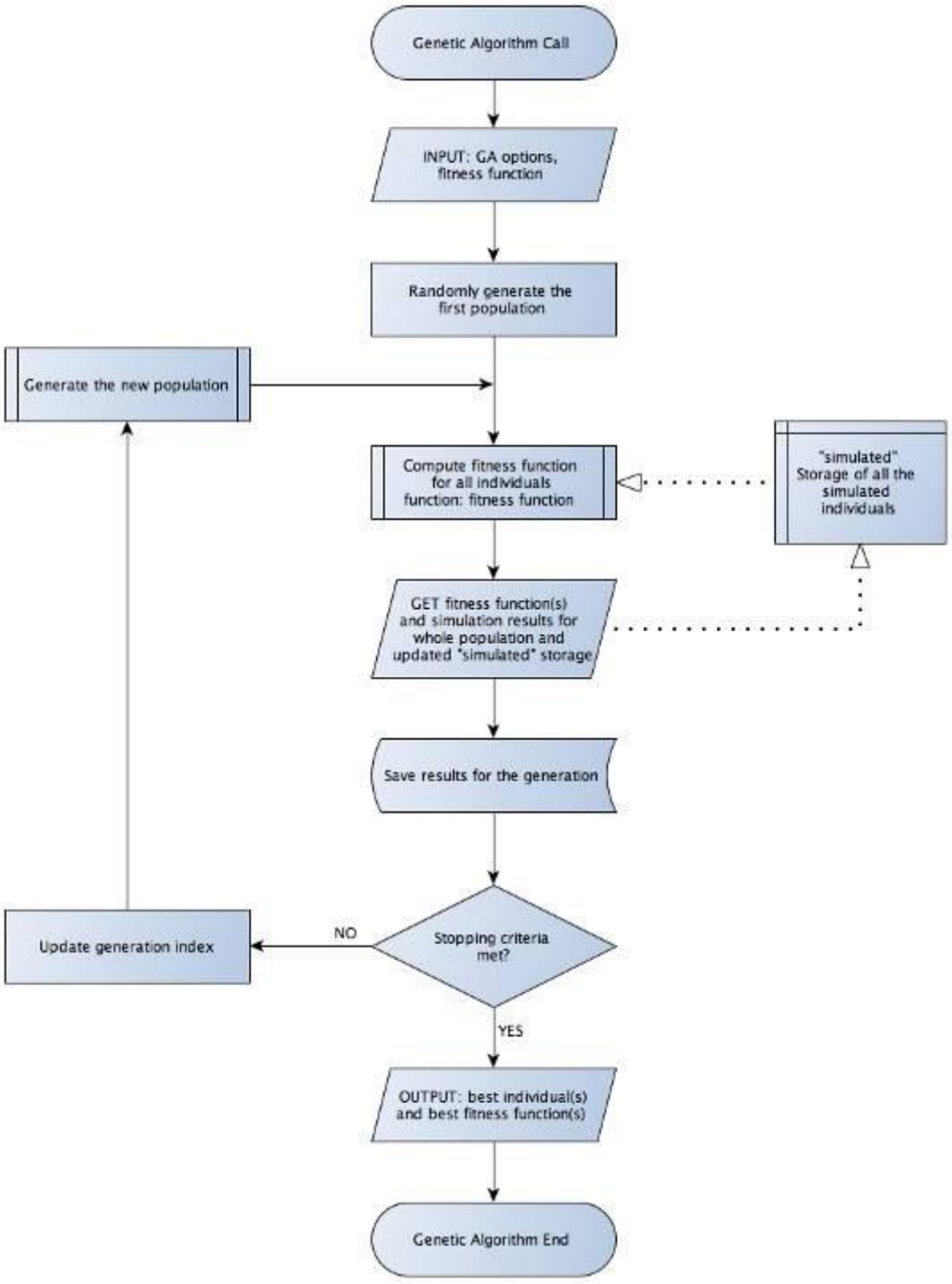


Figure 3.17 – Flowchart of genetic algorithm process

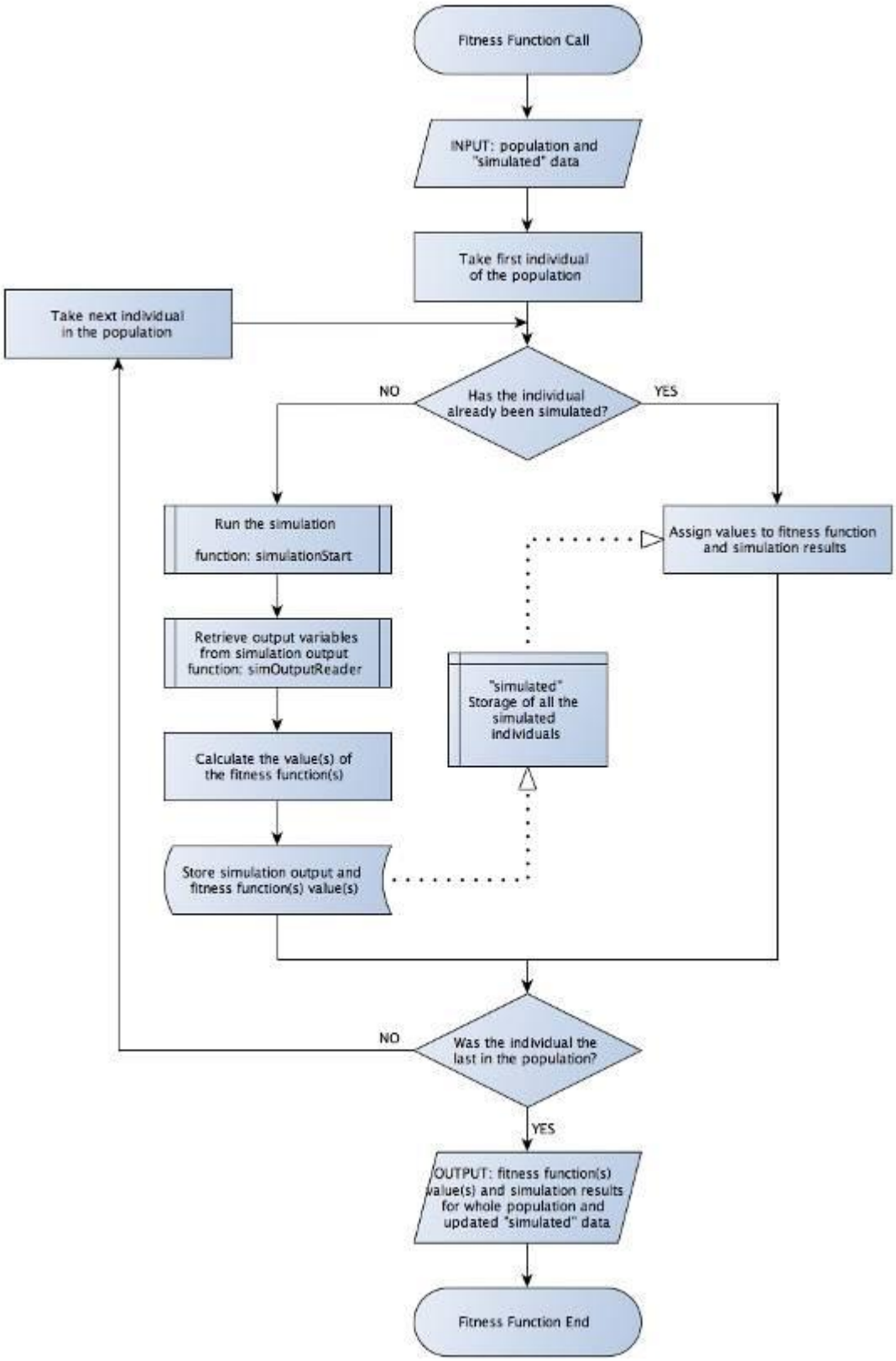


Figure 3.18 – Flowchart of the general fitness function process

4

Case Study I
General façade of an office building

4.1 Description

The model of the office room that was used for the parametric studies in the first chapter was employed in this first case study to optimise the design of a general façade under different conditions. Most of the characteristics defining the façade were considered as variables, while all other design assumptions correspond to the ones described in the first chapter. The aim of this first case study is primarily to give evidence that the developed optimisation tool is functional and that the algorithm correctly finds the optima of the problem.

First a single-objective optimisation was carried out for a base case and a sensitivity analysis on the genetic algorithm parameters was performed in order to choose the ones that can find the optimal solution in the shortest amount of time without loosing accuracy. Then a double-objective optimisation was run in order to examine the trade-off between two criteria and to compare this kind of analysis to the single-objective one.

4.2 Variables

The variables characterising the composition of the façade that were chosen to be optimised are described in detail in the following table and figures.

VARIABLE	TYPE	Range/Step (if continuous) Possible Values (if discrete)
Glazing percentage	Continuous	33 – 100 % / 7.5 %
Type of glazing	Discrete	Triple, LowE, Sel1b, Sel2b, Sel3
Insulation thickness	Continuous	4 – 20 cm / 4 cm
Overhang depth	Continuous	0.5 – 1.2 m / 0.2 m
Fins depth	Continuous	0.3 – 1 m / 0.2 m

Table 4.1 – Variables of the problem

The basic window area consists of a strip of one meter height that spans the whole width of the façade corresponding to 33% of the total surface. The height can increase by steps of 0.25 m, initially in the upper part of the façade and then, when this one is completely glazed, also in the lower part. The result is 9 different possible glazing percentages that will be explored. The external shading is provided by an overhang and four louvres arranged as shown in figure 4.1. The depth of each type of device can vary as defined in table 4.1. The solar screening strategy also comprehends automated internal blinds on the windows that close when the glare index exceeds 22 or when the solar radiation incident on the façade is greater than 200 W/m^2.

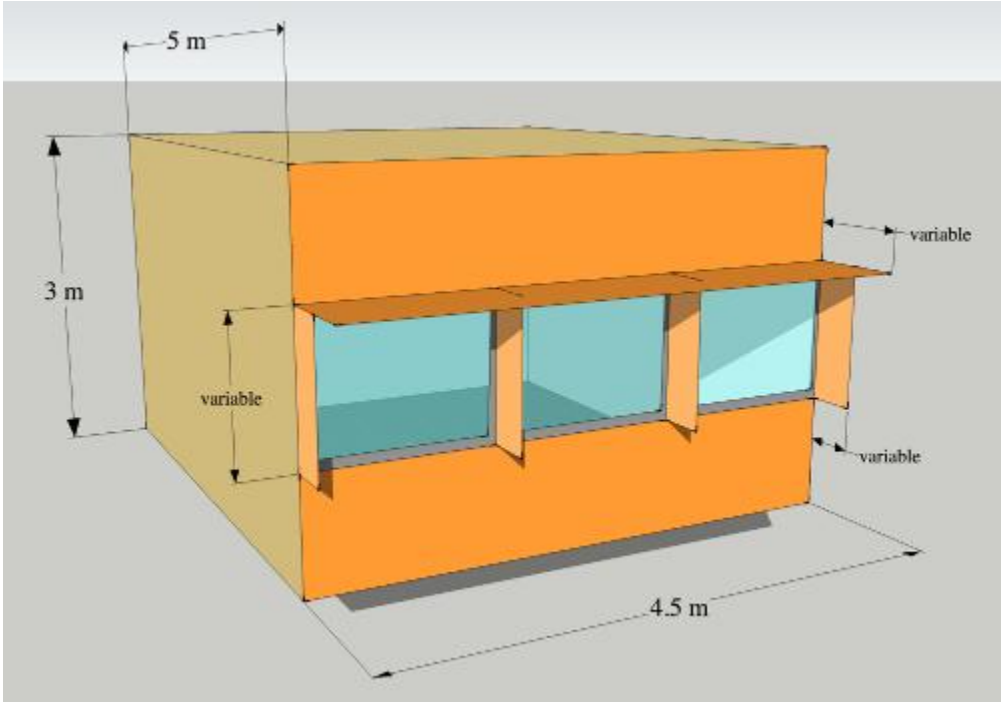


Figure 4.1 – Variables that define the geometry of the façade

The thermal and solar parameters of the six types of glass modelled are displayed in table 4.2. They range from a triple glazing to a normal low emissivity one, to four different types of solar control glass with different combinations of g-value and lighting transmittance. The last variable to be optimised is the thickness of the insulation layer, which is made of fibreglass with a thermal conductivity of 0,45 W/mK.

Variable Name	Glazing Type	External glass pane used	g-value	LT (%)	U-value (W/m^2K)
Triple	Clear Triple Glazing	Planibel TopN	0.50	72	0.8
LowE	Simple Low Emissivity	Planibel TopN	0.58	75	1.1
Sel1b	Selective type 1b	Iplus Sun	0.43	69	1.1
Sel2b	Selective type 2b	Stopray Safir 61/32	0.35	58	1.1
Sel3	Selective type 3	Ipasol 50/25	0.28	48	1.1
Sel4	Selective type 4	SunGuard SN40	0.24	38	1.1

Table 4.2 – Solar and thermal performance of glass types modelled

4.3 Single-objective optimisation

4.3.1 Objective function

The aim of the first optimisation process is to find out the façade configuration that guarantees the best environmental performance, in terms of minimum emission of CO_2. Therefore the considered objective function is the annual amount of carbon dioxide emitted due to the operational energy consumption for heating, cooling and artificial lighting.

$$F = \frac{f_{gas}}{\eta_H} \cdot Q_H + f_{el} \cdot \left(Q_L + \frac{Q_C}{COP} \right) \quad [\text{kg } CO_2]$$

where:

- Q_H , Q_L and Q_C [kWh] are the annual energy consumptions due to heating, artificial lighting and cooling respectively, which are calculated by the simulation program;
- f_{gas} and f_{el} [kg_{CO2}/kWh] are the carbon intensity factors for gas and electricity, which relate the amount of carbon emissions to the energy consumption of gas (for heating) and electricity (for cooling and artificial lights); these values have been assumed to be f_{gas} = 0.194 kg_{CO2}/kWh and f_{el} = 0.422 kg_{CO2}/kWh [1];
- η_H [-] is the overall annual efficiency of the heating system, which has been assumed to be 0.89 considering heat production only [32];
- *COP* [-] is the coefficient of performance of the cooling system; normally it depends on the type of system installed and on the climatic conditions, but for this study an average value of 3 was assumed.

4.3.2 Results

For the main study the model was set with the façade exposed to the south in the climate of Paris. Before performing the optimisation, the whole design space was evaluated. The chosen variables give rise to 5625 possible combinations. This is not a very big design space, but it was intentionally kept to a limited size in order to be able to run the simulations corresponding to all the possible solutions in a reasonable time stretch. The procedure took around 27 hours on a 2.5 Giga Hertz MacBookPro. In this way the performance of the optimisation algorithm can be assessed because it is possible to identify the real minimum of the problem.

The optimisation was run initially with a population of 20 individuals for 20 generations. The length of the process was around 40 minutes, during which 147 simulation were carried out. Figure 4.2 illustrates the evolution of the algorithm through the generations. It can be observed that the minimum is found already at the sixth generation. As it can be inferred from the progress of the mean objective function value, the convergence is quite fast in the first generations, where the exploration of the initially spread out solutions quickly concentrates in the area of the minimum. The number of simulations performed to reach the optimal point represents only the 2.6 % of the total possible solutions.

The configuration of the façade that corresponds to the value of 174 $kgCO_2$ found as the minimum presents the characteristics reported in table 4.3.

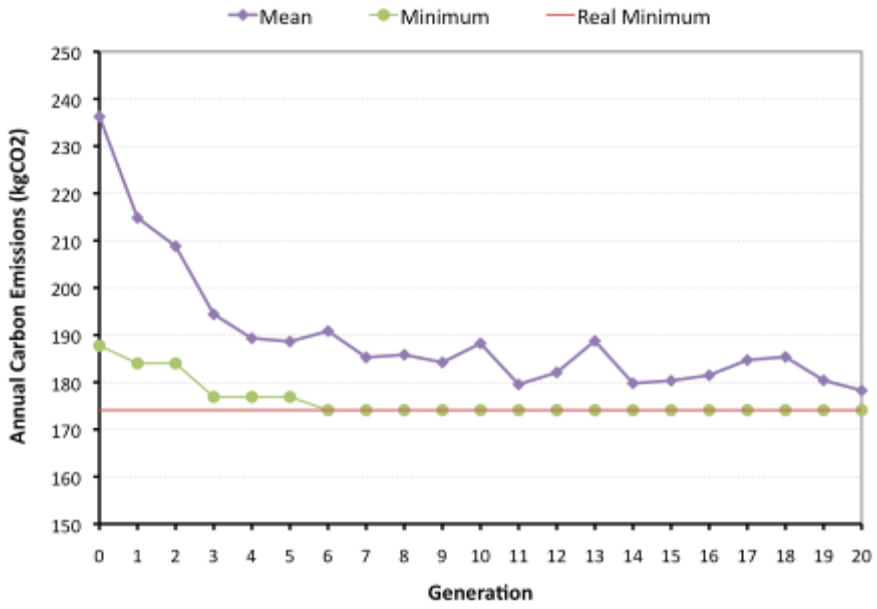


Figure 4.2 – Algorithm progress for single-objective optimisation

VARIABLE	VALUE
Glazing percentage	33 %
Type of glazing	Sel1b
Insulation thickness	20 cm
Overhang depth	1 m
Fins depth	0.6 m

Table 4.3 – Optimal façade configuration

4.3.3 Sensitivity analysis on algorithm parameters

The optimisation was repeated with different population sizes and with different numbers of generations. Table 4.4 reports the different settings for the seven cases analysed and the corresponding results found. Images depicting the progress of the algorithm for each case are reported in Appendix B. In two optimisations the real minimum was not reached, because the population was not big enough to guarantee a proper exploration of the design space (case 5) or because the number of generations was too small for the algorithm to achieve a complete convergence (case 1). Nevertheless it is encouraging to observe that most of the times the real minimum can be found with a very limited number of model evaluations (or simulation performed). It is interesting to notice that in the cases where the minimum is achieved, it can actually be found in different stages of the algorithm. The reason for this lays in the probabilistic nature of genetic algorithms, as every time the initial population is randomly generated and the also the genetic operators work with a certain amount of probability when shaping the new generation.

Case number	Population	Generations	Minimum ($kgCO_2$)	Minimum found at generation	Number of simulations performed
1	20	10	178	-	108
2	20	15	174	12	125
3	20	20	174	6	147
4	20	25	174	7	163
5	15	15	176	-	102
6	25	15	174	9	179
7	30	15	174	9	193

Table 4.4 – Results with different settings of the algorithm

4.4 Double-objective optimisation

4.4.1 Objective function

When dealing with office buildings one of the most difficult challenges is to find a proper compromise between the cooling loads and the artificial lights usage, as they are two extremely opposing criteria. Therefore the Pareto front in this double-objective optimisation will consist in the trade-off curve between the annual energy demand for cooling and for lighting.

$$F_1 = \frac{Q_C}{COP} \quad \text{[kWh]}$$

Where a value of 3 was assumed for the coefficient of performance of the cooling system as in the previous case.

$$F_2 = Q_L \quad \text{[kWh]}$$

The lights energy demand is based on the daylighting strategy described in the first chapter, which assumes that the artificial lights are controlled by the two sensors placed inside the room and that their intensity is regulated according to the levels of daylight that enter through the windows.

4.4.2 Results

The results of the simulations performed for the evaluation of the whole design space in the single-objective optimisation were used to build the whole design space also for this double-objective problem. Next the points on the real *pareto front* were identified in order to later compare them to the ones obtained from the optimisation process. The fact that the two objectives are completely opposing helps in generating a very well defined pareto front composed by 163 points. All the points in the objective space and on the front are displayed in figure 4.3.

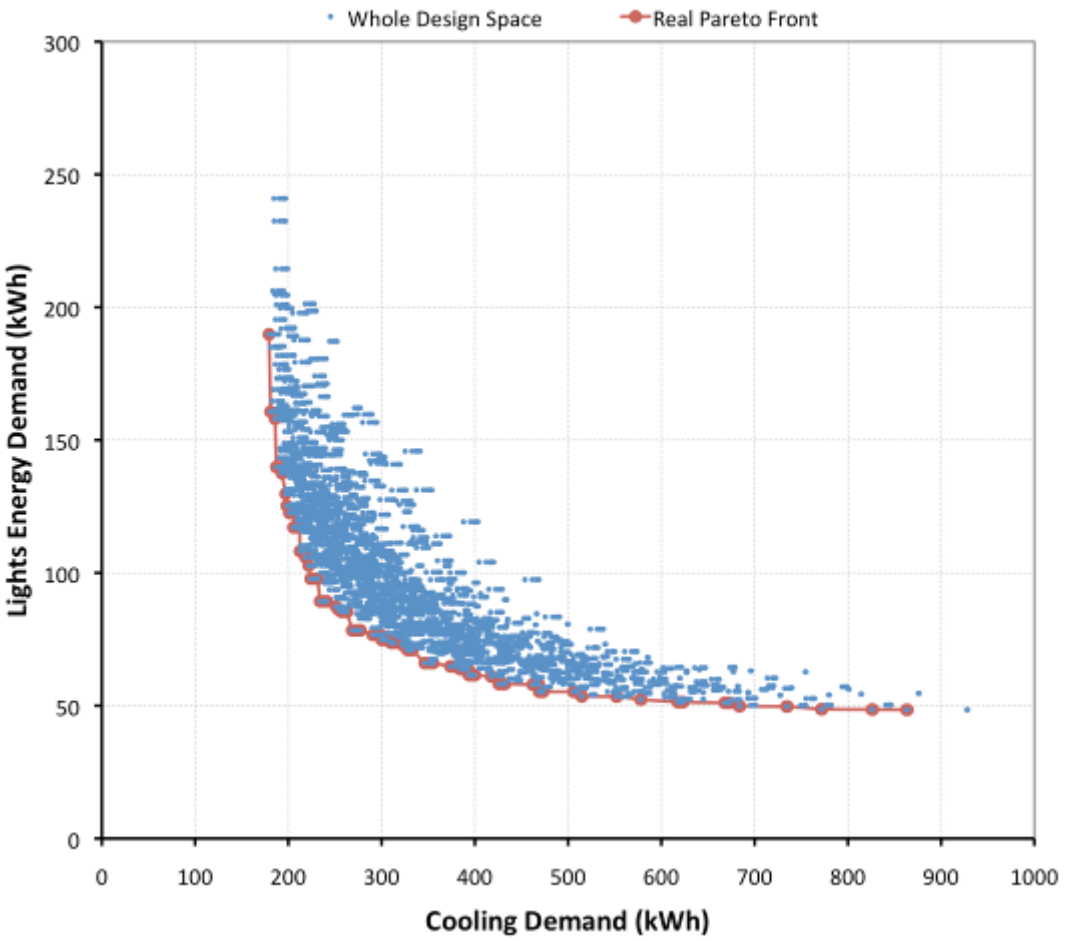


Figure 4.3 – Complete design space

Two different double-objective optimisations were performed, with the options listed in table 4.5. The corresponding pareto fronts found are displayed in figure 4.4 and as predictable the one achieved with a bigger population is more accurate than the other. It is interesting to notice how the points towards the two extremes are found with good precision while the ones in the central area are more far away from the real front. In the first case the points are not evenly distributed and there are some evident discontinuities, hence the number of individuals and of generations are clearly not enough to reach a satisfactory solution. In the second case the resulting front is more balanced and the solutions uniformly spread out.

Case number	Population	Generations	Points on Pareto Front	Simulations performed
1	30	15	12	315
2	40	20	16	500

Table 4.5 – Different options of the algorithm investigated

The number of points on the front is limited, but enough to correspond to groups of solutions with significantly different characteristics. In fact, among the 163 points on the real pareto front the ones that fall very close to each other are likely to be almost identical solutions, with only small variations in one or two variables at a time. When conducting a search of this kind in building related problems, what is most important is to be presented with a range of solutions that can represent different trade-offs of the objectives. This is achieved with an acceptable accuracy in the second case. If a bigger number of solutions were required, the number of individuals in the population should be increased; this would mean that a bigger number of simulations were to be run. In the second case this number summed up to 500, which is about 9 % of the total possible solutions, so increasing it would mean more than 10 % of the design space were to be evaluated. In a problem of this level, where the space is rather small and the simulations are fast (around 20 seconds each), it could still be feasible to do it, but in problems with a larger design space it would be time consuming and the concept of the optimisation process would lose its value.

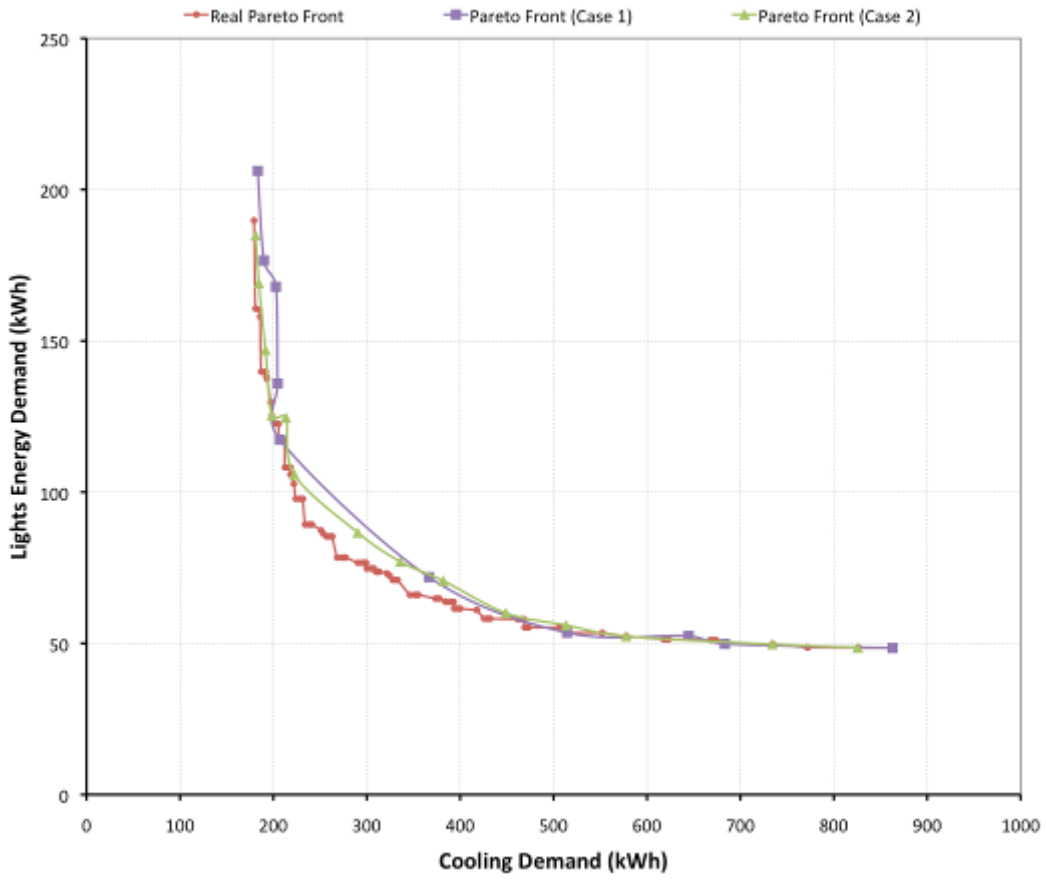


Figure 4.4 – Pareto fronts for different cases compared to real front

5

Case Study II
BiPV on test building in Austria

5.1 Description

During a period of study at the “Department of building physics and building ecology” of the “Vienna Technical University” in 2011, a collaboration for an optimisation project was established. The object of the project was a test building by an Austrian private research centre named Fibag with whom the department has a partnership for the development of joint research projects. The centre is located in Styria and it operates in the field of "integral construction engineering", in particular offering solutions for the integration of energy technologies into the building envelope or the optimisation of the entire energy balance of a building through the combination of the technologies used in the envelope with the technical facilities of the building itself [1].

Figure 5.1 – External view of the Fibag test building

The building contains eight main rooms on two different floors and some of them were made available to the department to be used for experimental studies. Since the building envelope consists in a continuous curtain wall façade, the idea was to try to optimise its design from the point of view of investment and operation costs including the use of building integrated photovoltaics. Only one of the rooms was picked out to be analysed, with a floor area of 50 m^2 and the main façade facing south. The basic principle was to create a modular façade where each of the modules could be either glazed, opaque, or equipped with a photovoltaic panel.

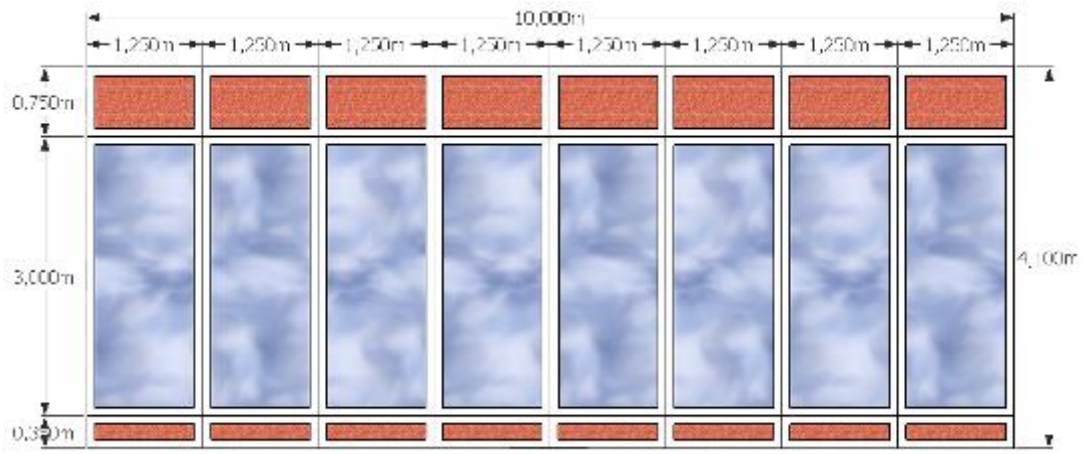


Figure 5.2 – Sketch of the original façade

The original assembly of the façade, sketched out in figure 5.2, consists in eight curtain wall units fully glazed from floor level to ceiling level, and with spandrel panels covering the service spaces above and below. It was decided to change its composition in order to attain a higher degree of flexibility in the possible layouts. Consequently the key area, ignoring the spandrel panels, was subdivided in three rows and four columns, thus generating a sort of grid of twelve equal modules with an area of 2.5 m^2 each. The two central ones were assumed to be fixed and glazed in order to provide a minimum window area which corresponds to 10% of the floor area, while the other ten modules were assumed to be variable and each of them could be any of the three types.

A base configuration where all the variable modules were chosen to be simple spandrel panels was taken as reference case. The energy consumption levels for space heating, cooling and artificial lights stemming from the energy simulation of this case were recorded to be later compared with the ones arising from all other possible compositions generated during the optimisation process.

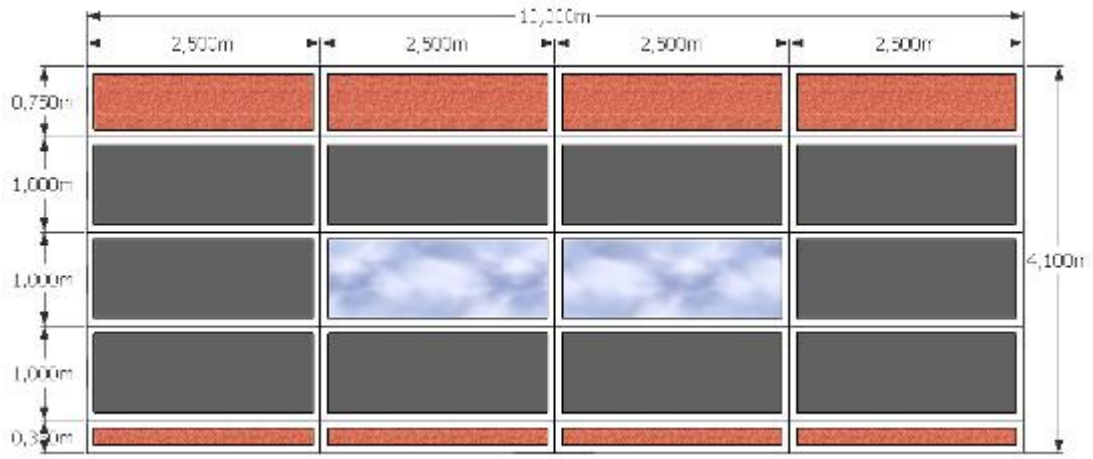


Figure 5.3 – Layout of the façade used in the study

5.2 Thermal model

An *EnergyPlus* thermal model of the room was built: all internal partitions are considered adiabatic, the east façade facing outdoors is completely opaque with a thermal transmittance of 0,25 W/m^{2}K. The main façade is modelled with a backing surface that embodies the behaviour of the aluminium frames of the curtain wall units. The thermal transmittance of this part is a mean value of the real transmittance of the frames, calculated as 4.2 W/m^{2}K. All modules are sub-surfaces "cut-out" in this backing surface, with characteristics described in the "Variables" section below.
Two reference points positioned at desk level inside the room as pictured in figure 5.5 are used to control the daylighting in the zone. Based on the levels of daylight coming in from the windows, they trigger the use of artificial lights in the room.

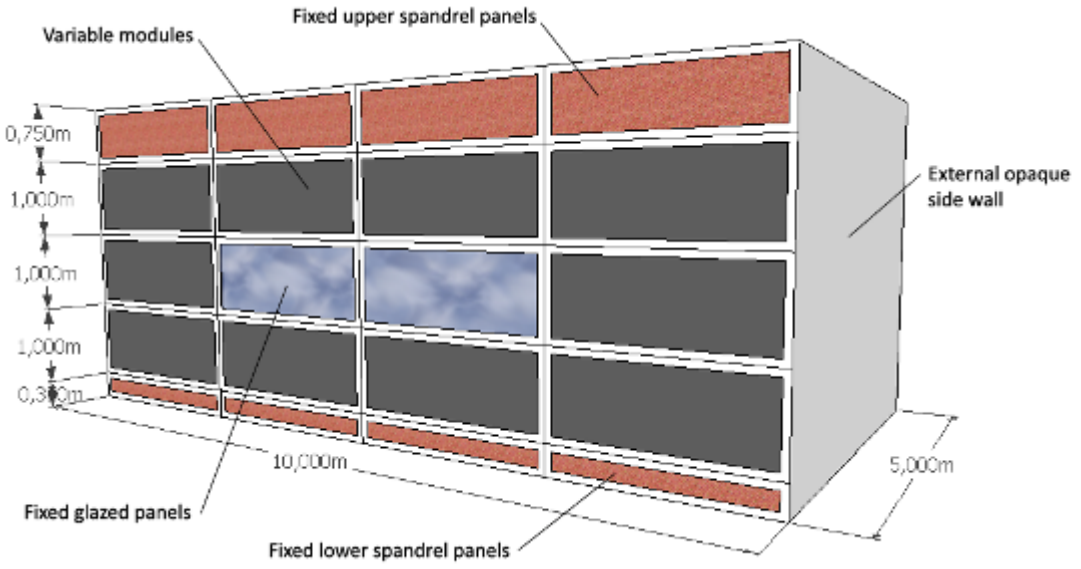


Figure 5.4 – Image of the *EnergyPlus* model

The illuminance design level was set at 500 lux, the lights being turned on whenever this level is not met and working with a continuous dimming strategy. Besides, all the windows have internal blinds equipped with automatic glare control: the blinds are deployed whenever the glare index calculated in the reference points exceeds the target value of 22.
All other design assumptions are summarised in table 5.1. The plant system is considered to have unlimited capacity and can thus always maintain the required setpoint temperatures during the periods of peak heating and cooling loads.

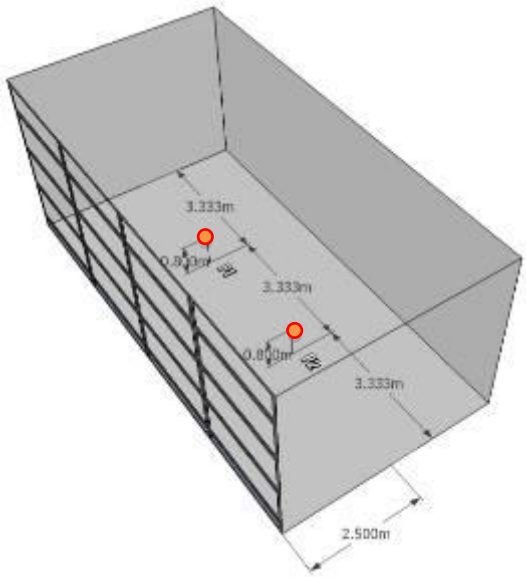


Figure 5.5 – Disposition of daylighting reference points in the model

The overall annual efficiency of the heating system is assumed to be 0.8, while for the coefficient of performance of the cooling system a value of 2.5 is taken. The simulations were carried out for a year time period using the Vienna Schwechat weather file.

TYPE	VALUE
Internal gains	
Electric equipment	15 W/m^2
People (max 6 people)	126 W/person
Lights	8 W/m^2
Surface reflectance	
Walls	50 %
Ceiling	70 %
Floor	30 %
Ventilation	
Infiltration	0.15 ach
Mechanical Ventilation	1 ach (max)
Operating strategy	
Heating setpoint / setback	20°C / 12°C
Cooling setpoint / setback	25°C / 32°C
Heating and Cooling availability	7am – 7pm on workdays

Table 5.1 – Model design assumptions

5.3 Variables

Except the fixed upper and lower spandrel parts and the two central glazed panels, all the other ten modules of the façade can vary. Each of them can be either a spandrel panel, a glazed panel, or a BiPV panel. This gives rise to 3^{10} = 59049 possible combinations. Moreover, a "rule" was implemented in the algorithm so that when glazed panels are present, they are first positioned in the two upper rows, and then in the lower row only if the former are completely filled with windows.

- Spandrel Panel

The construction is that of a simple opaque panel made up of an outer cladding layer, an insulation layer and a vapour barrier followed by an internal gypsum board. The thermal transmittance adds up to 0,25 W/m^{2}K.

- Glazed Panel

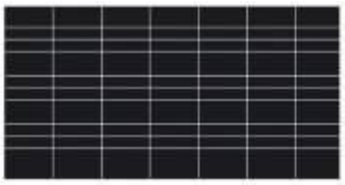
It is constituted of a double glazing unit with a clear internal glass pane and a low emissivity external pane. The thermal, solar and visual characteristics are reported in the next table.

U-value	g-value	LT
1.14 W/m^{2}K	0.58	75 %

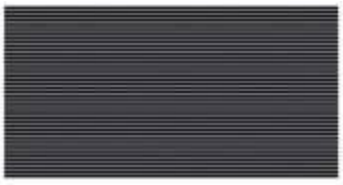
Table 5.2 – Characteristics of the glazing modelled

- Photovoltaic Panel

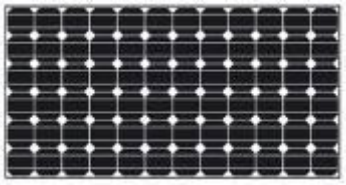
The BiPV panels construction is the same as the spandrel panel with the addition of the layer containing the photovoltaic cells on the external surface. Although in the main case the panels feature polycrystalline silicon photovoltaic cells, other two types of photovoltaic panels are considered later in the study, so all three kinds are described.



Polycrystalline PV
Efficiency: 14 %



Thin Film PV
Efficiency: 7 %



Monocrystalline PV
Efficiency: 18 %

The PV panels are modelled in *EnergyPlus* with objects that simply apply their overall energy conversion efficiency to the incident solar radiation. They are then connected to an *inverter* object with an efficiency of 0.9 that transforms the direct current produced by the panels into alternate current that can be fed to the grid.

5.4 Objective functions

The double-objective optimisation aim is to investigate the trade-off between the costs related to the improvement of the façade and the resultant savings associated with the operation of the building. The first criterion is therefore defined by the investment costs to be faced when upgrading the reference façade with more glazed panels or with building integrated photovoltaics:

$$F_1 = n_G \cdot p_G + n_{PV} \cdot p_{PV} \quad [€]$$

where: n_G is the number of glazed panels (in addition to the two fixed ones)

p_G is the price to replace a spandrel panel with a glazed panel

n_{PV} is the number of photovoltaic panels

p_{PV} is the price to replace a spandrel panel with a photovoltaic panel

The following assumptions were made on the prices:

Replacement of a spandrel panel	**PRICE**
with a glazed panel	200 €
with a polycrystalline-Si PV panel	600 €
with a thin film PV	200 €
with a monocrystalline-Si PV	1000 €

Table 5.3 – Prices assumptions

The savings in energy costs related to the building operation are calculated in comparison to the base case and are defined as follows:

$$s_{OC} = UPV^* \cdot \left[dE_{heat} \cdot t_{GAS} + \left(dE_{cool} + dE_{lights} \right) \cdot t_{EL} \right] + 13 \cdot E_{PV} \cdot t_{FEED-IN} \quad [€]$$

where the differential energy consumptions for heating, cooling and artificial lights are respectively:

$$dE_{heat} = E_{heat,ref} - E_{heat} \quad [\text{kWh}]$$

$$dE_{cool} = E_{cool,ref} - E_{cool} \quad [\text{kWh}]$$

$$dE_{lights} = E_{lights,ref} - E_{lights} \quad [\text{kWh}]$$

It is clear from these expressions that a decrease in energy consumption compared to the base case yields a positive value for the savings, and vice versa.

E_{PV} is the energy produced by the photovoltaic panels, in kWh, and if present is obviously always positive.

The energy tariffs for the supply of natural gas, used for space heating, and of electricity, used for space cooling and artificial lighting, were taken directly from the main Austrian energy supplier[1] and are displayed in table 5.4.

Energy type	Tariff [€/kWh]
Natural gas, t_{GAS}	0.065
Electricity, t_{EL}	0.172

Table 5.4 – Energy tariffs

In Austria the system of feed-in tariffs, better known as "*Einspeisepreis-Verordnung"*, is active since 2001. The actual tariffs for 2011, reported in table 5.5, are determined by the ÖSVO 2011[2] and are applied to thirteen years long contracts. The value for building integrated photovoltaic systems with a power greater than 20 kW was assumed in the study.

Peak power [kW]	BiPV [€/kWh]	Stand-alone PV [€/kWh]
$5 < P \leq 20$	0.38	0.35
$P > 20$	**0.33**	0.25

Table 5.5 – Feed-in tariffs fixed by the *Ökostromverordnung* 2011 in Austria

The operating costs were considered for a period of thirteen years in order to correspond to the length of the feed-in contracts for PV production prefigured by the Austrian law. Since the feed-in tariff value is guaranteed during this period, the yearly savings arising from PV energy production are just multiplied by thirteen. On the other hand, the yearly operation costs cannot be considered constant over the years because the energy prices are likely to rise during a thirteen year period. Besides, since the evaluation of the costs is done at present time, there's the need to calculate the present value of these non-uniform amounts recurring over the period considered. Hence, using life-cycle cost analysis concepts, the modified uniform present value factor is calculated with the following formula:

$$UPV* = \frac{1+e}{d-e} \cdot \left[1 - \left(\frac{1+e}{1+d} \right)^{n} \right] = 12.15$$

in which the following values for the parameters were assumed:

[1] http://www.wienenergie.at/

[2] Ökostromverordnung 2011, published on the Bundesgesetzblatt 28/01/2011. The full text can be found at the following address:
http://www.ris.bka.gv.at/Dokumente/BgblAuth/BGBLA_2011_II_25/BGBLA_2011_II_25.pdf

n = 13 for the number of years;

d = 3% for the real interest rate;

e = 2% for the escalation in energy price.

This factor represents the present value of recurring annual amounts that change from year to year at a constant escalation rate over a certain number of years, given a fixed interest rate. It the present problem, the *UPV** factor is applied to the annual savings arising from the operation of the building.
Finally, since the energy cost savings need to be maximized and not minimized, but the optimisation algorithm works only by minimizing the functions, the second objective must be expressed as the energy cost savings with a minus sign in front:

$$F_2 = -s_{OC} \quad [€]$$

5.5 Results for the base case

The double-objective optimisation was run for the south facing façade, using polycrystalline silicon PVs for the building integrated photovoltaics. The resultant optimised solutions are displayed in figure 5.6. For visual clarity the second objective is displayed as the maximised operation savings, hence with positive values. That's the reason why the points in the graph appear reversed compared to the usual *Pareto Front* representation. One additional solution with zero investment costs and zero savings was excluded since it corresponded to the reference case. The dotted blue line corresponds to the equivalence in investments and savings, thus marking the separation between the zones of overall losses and savings. As it can be observed, all optimised solutions fall in the savings zone. The red line is a polynomial tendency line that interpolates the solutions, giving a more clear graphic idea of how they denote the *Pareto Front*. The fact that the solution points do not actually fall on an exact exponential curve is to be ascribed to the heavy discontinuity of the objective functions.

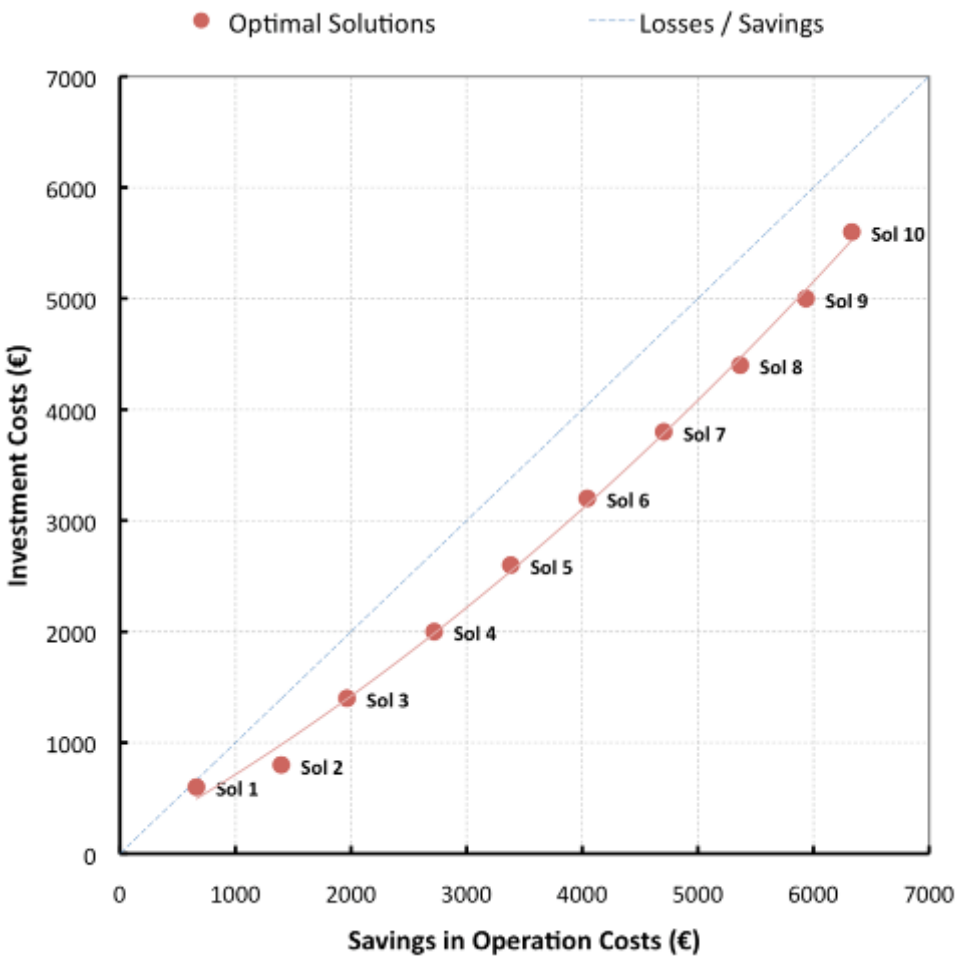


Figure 5.6 – Optimal solutions for base case

It is straightforward that as the investments increase, so do the savings in operation costs. However, when analysing the total savings (savings in operation costs minus investment) compared to the reference case, it can be noted that the solution that maximises the total savings is not the one the maximises the savings in operation costs. This is revealed in the next graph (figure 5.7), where it can be observed that the difference in total costs (or savings since they are negative) increases to a maximum corresponding to solution 8 and then starts decreasing.

As illustrated afterwards, the increase in investment costs essentially corresponds to a gradually bigger number of PV panels in the solutions, as the number of glazed panels present in the solutions is always limited to one. In the light of this fact, the decrease in total savings that happens in the solution with the highest investment costs means that after integrating a certain number of PV panels in the façade it is not profitable to add more of them because the extra investment wouldn't be balanced by the monetary gain received from selling the resultant higher electricity produced.

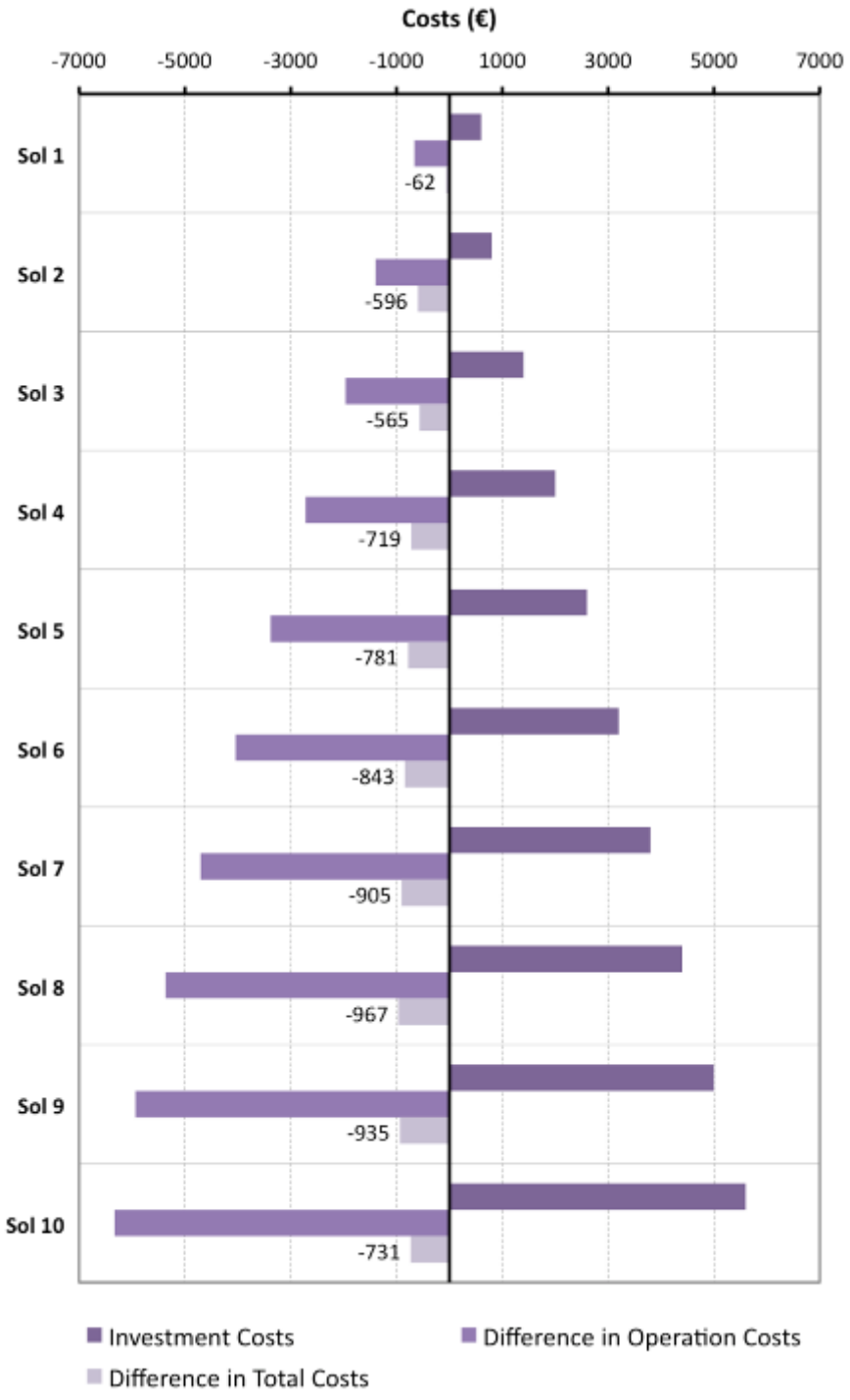
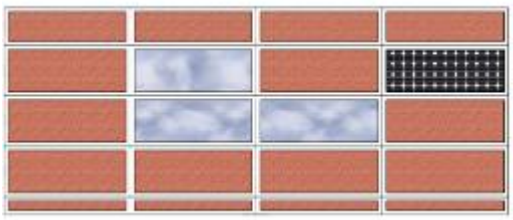


Figure 5.7 – Differences in costs compared to reference case for all solutions

Some of the solutions are now analysed in depth by showing a scheme of the façade configuration and a graph with the breakdown of the savings in operation costs in terms of differential costs compared to the reference case. The first solution presents the same façade configuration of the reference case with the addition of one PV panel: the savings are exclusively due to the PV production and the difference in total costs is negligible.

Solution 2 – Total savings = 596 €



The additional glazed panel brings forth significant savings in artificial lights energy consumption as it increases the daylighting levels in the room. The small losses in cost for heating and cooling are abundantly outnumbered by the income resulting from the selling of the energy produced by the PV panel.

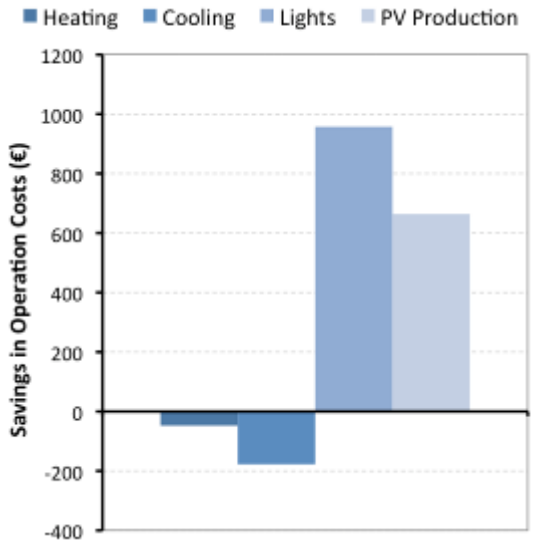
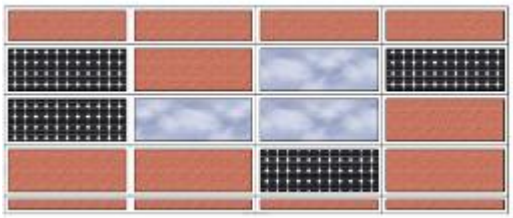


Figure 5.8 – Savings / Losses for Solution 2

Solution 5 – Total savings = 781 €



Like in the previous solution, savings in the cost for artificial lighting are guaranteed by the additional window. The losses in cost for heating and cooling remain essentially constant while the four PV panels generate significantly higher savings due to the selling of the electricity produced.

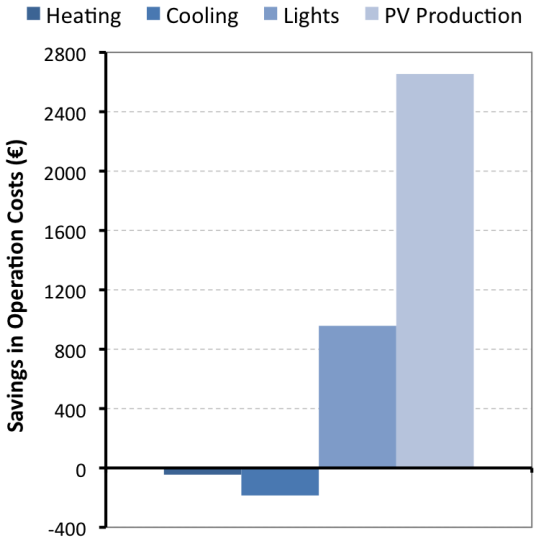
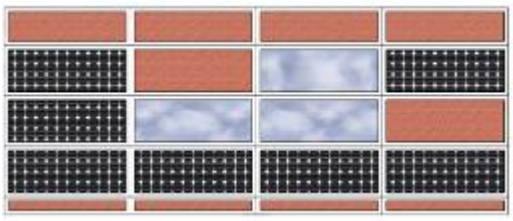


Figure 5.9 – Savings / Losses for Solution 5

The other solutions follow the same trend, presenting one additional glazed panel and an increasing number of PV panels as the investment costs grow. Hence the savings or losses in operation costs remain basically constant, while the money gains from the selling of the electricity produced boost as the number of PVs increases.
It is interesting to point out that the additional window is always placed in the upper row, meaning that is spans from a two to a three metres height. The reason is that this configuration allows more light into the room without causing additional glare,

which would have an influence on the deployment of the automated blinds and would thus lower the amount of daylight from the window itself.

Solution 8 – Total savings = 967 €



The savings in artificial lights costs and the losses in heating and cooling costs remain the same as the previously analysed case. The only difference is the bigger contribution arising from the selling of the PV produced electricity due to the bigger number of panels installed.

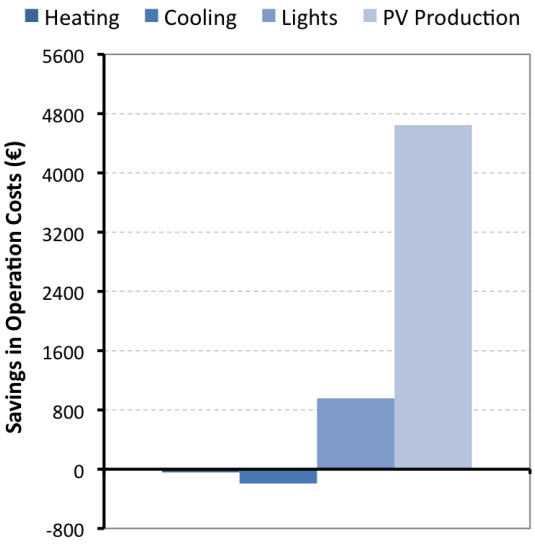
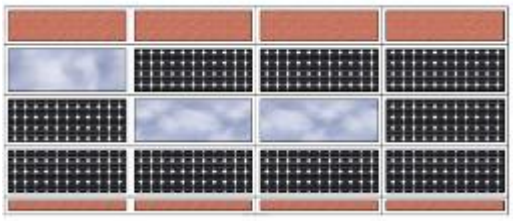


Figure 5.10 – Savings / Losses for Solution 8

Solution 10 – Total savings = 731 €



The same thing happens in the last solution, where the number of PV panels integrated in the façade is maximum. The savings deriving from the selling of the electricity produced rise further, but as explain earlier in this case they do not overcome the investment costs, hence the total savings are lower then the previous solution.

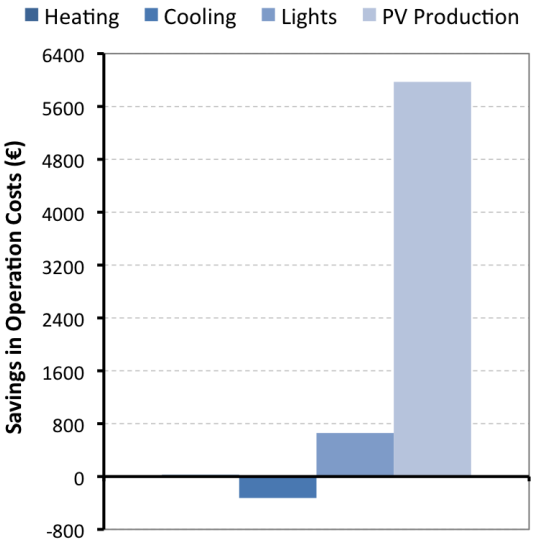


Figure 5.11 – Savings / Losses for Solution 10

All solutions are reported in detail in appendix C.

5.6 Additional results and sensitivity analyses

5.6.1 Results for different types of PV panels

The same optimisation was run using different technologies for the BiPV panels: monocrystalline-Si cells first, and thin film in a second time. The different relationship between the efficiency and the investment cost of these different kinds of PVs led to quite diverse optimised solutions compared to the base case. As it can be observed in figure 5.12, the higher efficiency of monocrystalline PV can produce higher savings in operation costs but the elevated investment cost assumed for this technology makes it less cost-effective than polycrystalline PV. With higher investments, so when a bigger number of panels is employed, the solutions fall in the overall losses area, thus becoming unprofitable from a total cost point of view. On the other hand, thin film panels provide a much better correlation between investment and savings. Although the savings in operation costs are generally lower compared to the other cases due to the minor efficiency of thin film, their small investment cost makes them very profitable, generating higher total savings.

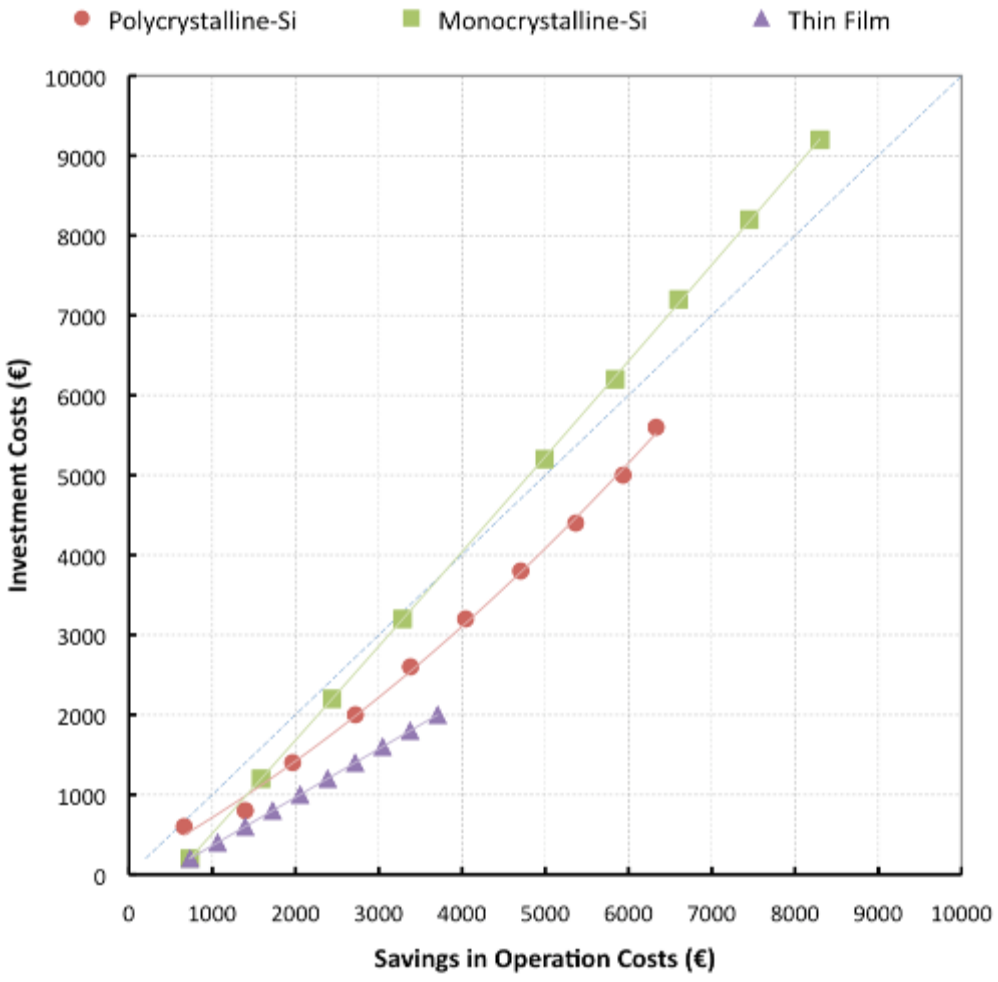


Figure 5.12 – Optimal solutions for different types of BiPV panels

The next three graphs show the average distribution of panels types in all solutions that yield positive total savings. In the case of monocrystalline-Si PVs, only a limited number of BiPV panels lead to cost-effective solutions, while with polycrystalline-Si PVs more panels can be employed; the number grows further in the case of thin film technology.

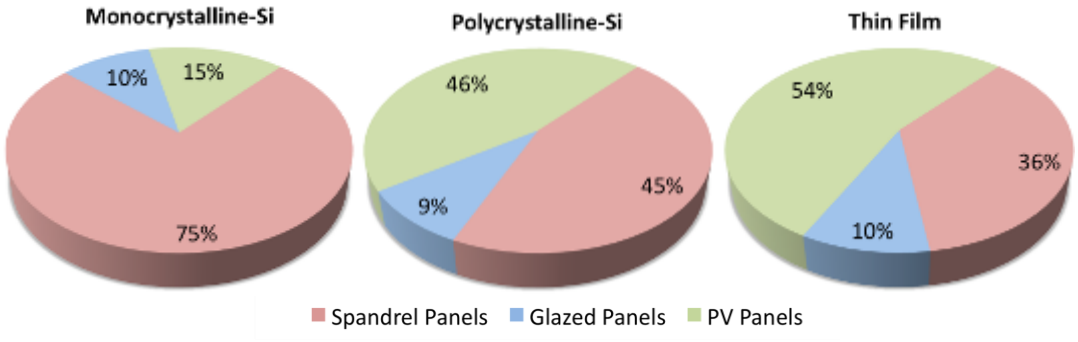


Figure 5.13 – Percentage of each type of panel in all solutions that yield positive savings

From inspection of all solutions it can be noticed that almost all of them comprehend only one additional glazed panel, while the ratio of spandrel to PV panels changes. Therefore an analysis on the differential total costs in relation to the number of BiPV panels installed on the façade can be carried out for each of the panel types. The results are displayed in figure 5.14, from which the advantage of using a large number of thin film PV panels emerges clearly.

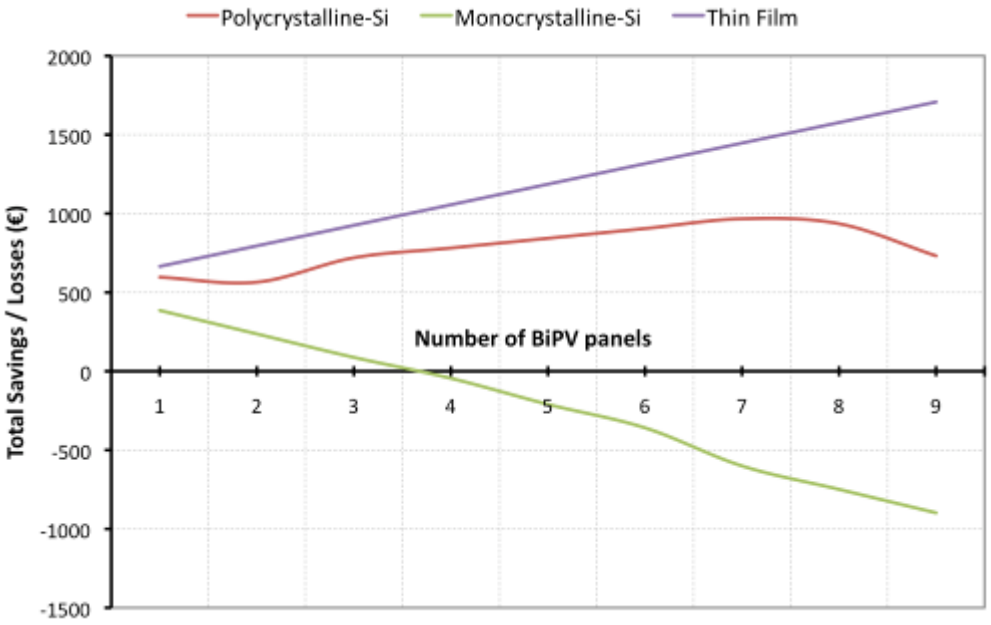


Figure 5.14 – Differential total costs in relation to number of BiPV installed

As already seen in the analysis of the base case, Polycrystalline-Si PVs are profitable in all found solutions but their benefits tend to diminish when the panels installed exceed a certain number. Finally, under the cost assumptions made, monocrystalline PVs are much less effective in producing actual savings, and when installed in large number can actually generate big losses in the overall differential costs.

5.6.2 Sensitivity analysis on *feed-in* tariffs

A major role in the search for the optimal solutions is played by the money gains arising from the energy produced by the photovoltaic system that is sold to the grid. A study was conducted to assess the influence of the *feed-in* tariffs on the solutions. Besides the base tariff of 0,33 €/kWh taken from the Austrian legislation, two different figures were considered: a lower one of 0,25 €/kWh and a higher one of 0,40 €/kWh. The results of the new optimisations runs, along with the base case ones, are displayed in the next figure.

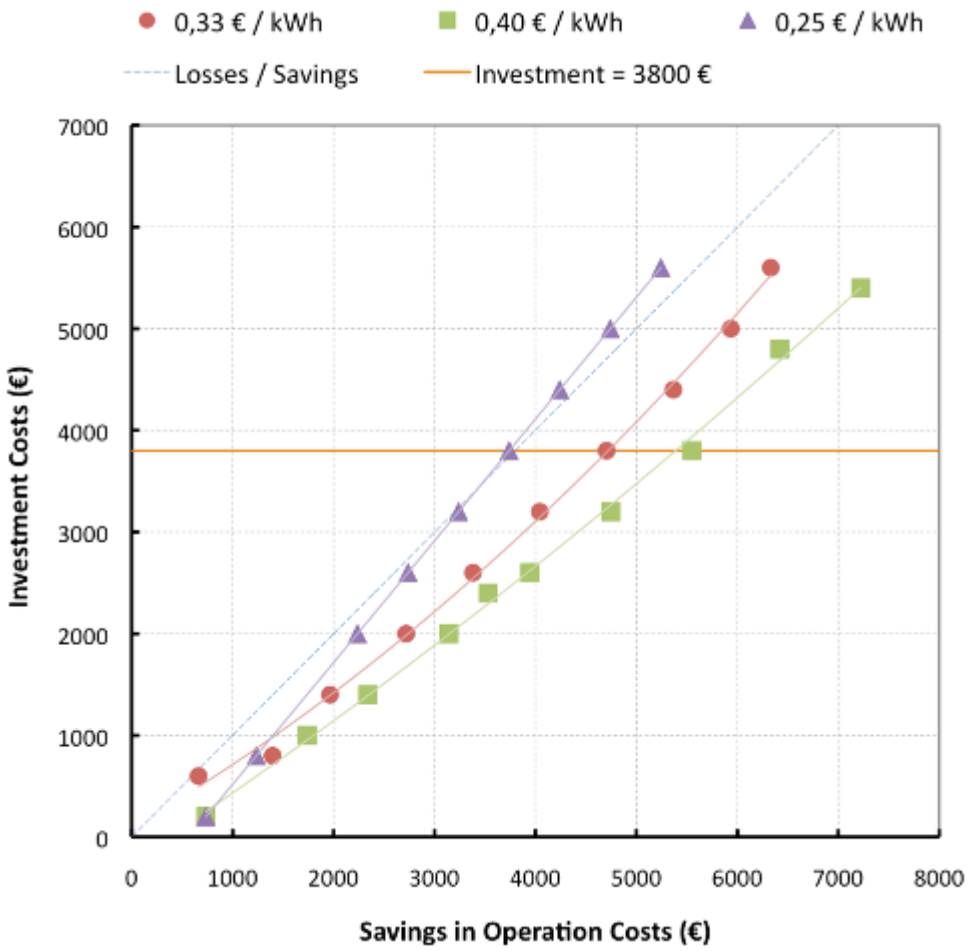


Figure 5.15 – Comparison of optimal solutions for different feed-in tariffs

As predictable, the bigger gains resulting from a higher *feed-in* tariff make the front of the optimised solutions shift towards the area of bigger savings, while the opposite

happens for the lower tariff. The difference is less significant for small investments (small number of PV panels involved) and it grows for bigger ones.
The horizontal orange line in the graph marks a constant investment of 3800 €, corresponding in all cases to a solution with one additional glazed panel, and six BiPV panels. The graph in figure 5.16 shows the savings that are achieved in this solution depending on the *feed-in* tariff considered: the ones stemming from PV energy production increase with the tariff itself, and the total differential costs follow this trend as the other savings remain constant since the thermal and visual characteristics of the façade are the same. However, for the 0,25 €/kWh tariff the total differential costs actually result in losses because the money paid for the PV energy was not enough to meet the investment costs. This proves the influence of the price paid out for the energy produced by the PVs on the final solutions and therefore it's a parameter that should affect the choices in the design stages.

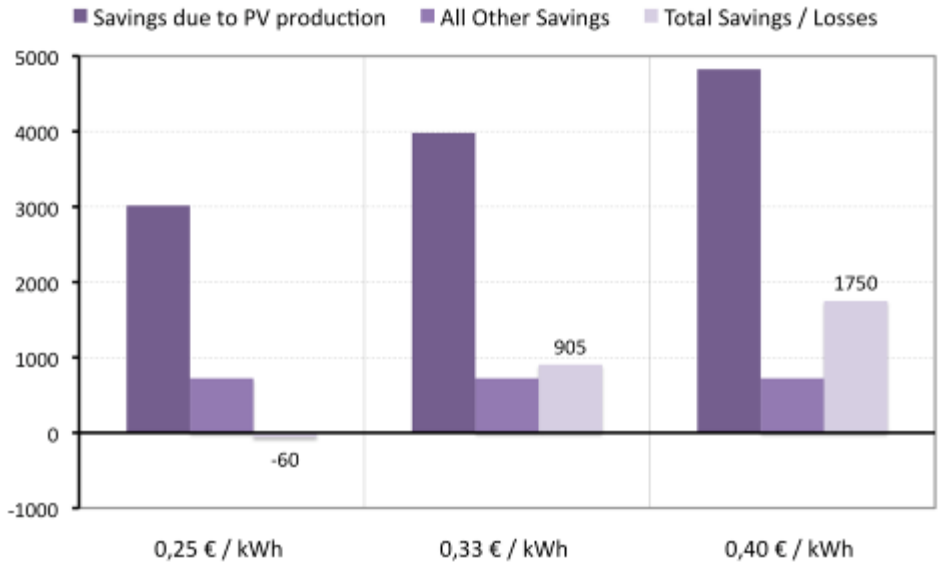


Figure 5.16 – Savings resulting from an investment of 3800 € for different feed-in tariffs

It is interesting to compare solutions that lead to the same amount of total savings in different scenarios of tariffs. In figure 5.17 the solution that in the base case yields the maximum total savings is considered, and starting from it a line that denotes equivalent savings is traced. The solution found for the case where the *feed-in* tariff is higher that falls on this line is obtained with a significantly lower investment.

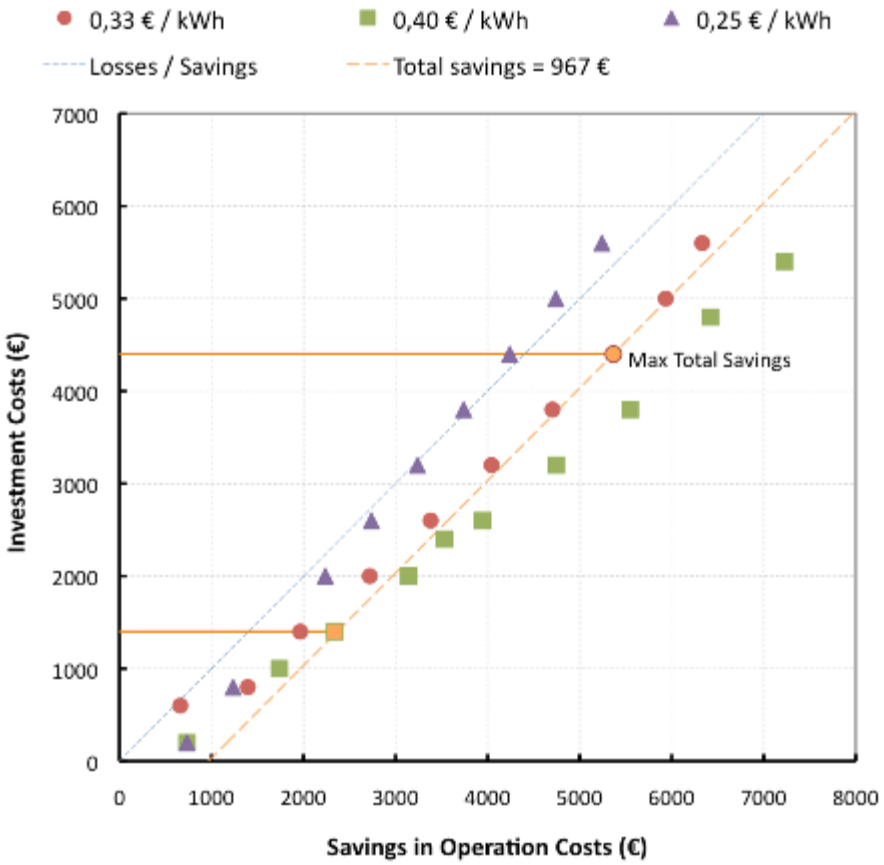


Figure 5.17 – Solutions with equivalent total savings for different feed-in tariffs

The configurations of the façade for these two solutions are displayed in the following figures: obviously the one corresponding to the higher feed-in tariff can reach the same level of savings with a smaller number of BiPV panels.

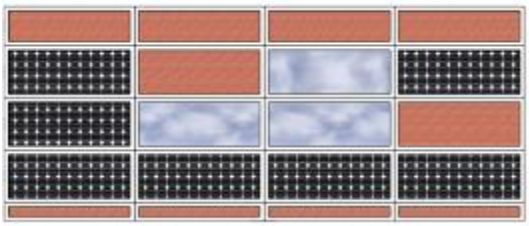


Figure 5.18 – Solution for base case

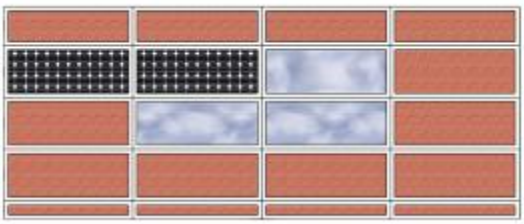


Figure 5.19 – Solution for 0,40 €/kWh feed-in

5.6.3 Influence of façade orientation

It was thought of interest to estimate the behaviour of the façade also for different orientations. Hence the model was rotated in order to carry out optimisations with the façade facing the other three main cardinal points. The influence of the orientation on the optimised solutions found can be observed in figure 5.20: as it shifts from south, through west and east, to north, the curves representing the solutions grow steeper, with the same investment generating gradually smaller savings in operation costs.

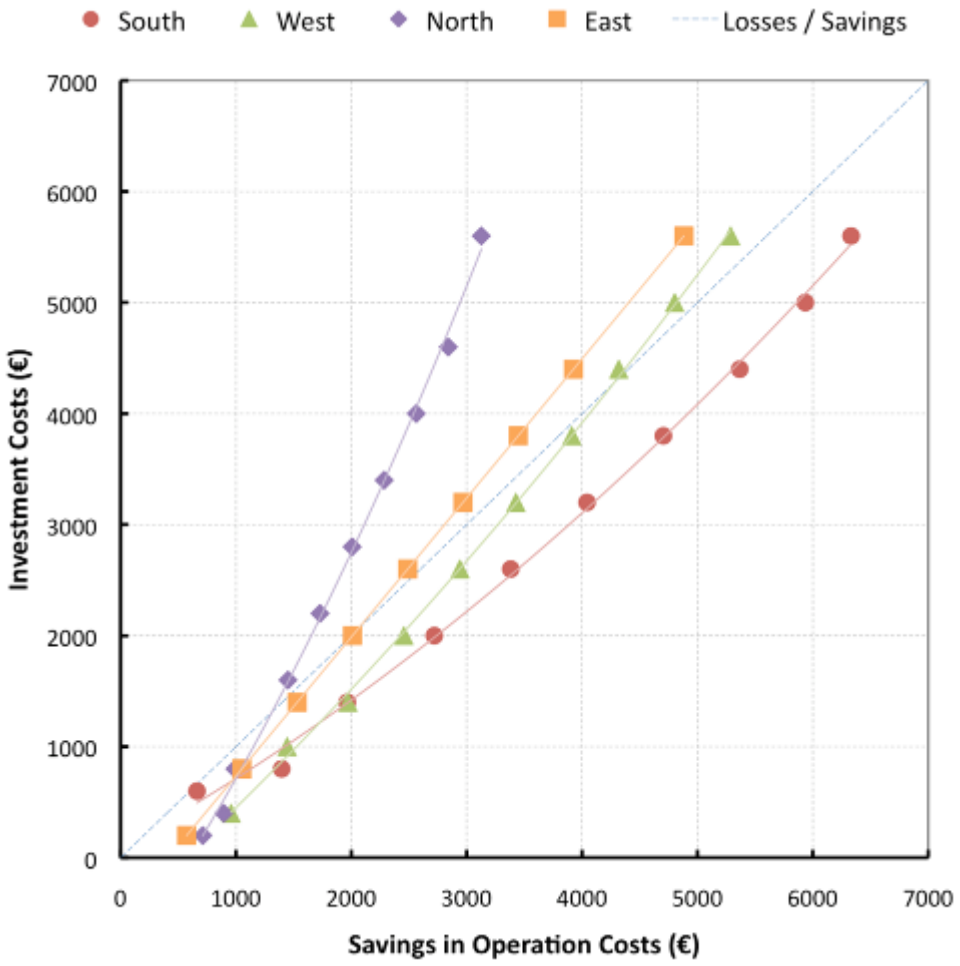


Figure 5.20 – Comparison of optimal solutions for different orientations of the façade

For little investment costs the optimal solutions yield positive total savings in all orientations; these solutions are generally characterised by one or two additional glazed panels in the upper row that guarantee a better daylight distribution and hence smaller costs for the artificial lights. This is more beneficial for the north, west and east facing cases than it is for the south facing one. Eventually, the addition of one PV panel can add a contribution to the total savings, the south orientation being the one that benefits the most from it.

As the investments increase, the number of glazed panels in the solutions remains more or less constant while progressively more PV panels are used; the influence of the savings resulting from the selling of the energy produced on the total savings is greater and thus the orientation that guarantees the maximum exposition to the sun (south) is privileged, while the one that catches little direct solar radiation (north) leads to considerable losses in total costs. The west and east orientation follow a similar in-between trend, with solutions providing overall savings up to a certain level of investment, which is bigger in the case of a west facing façade. This can be more clearly detected from the graph in figure 5.21, where the difference in total costs compared to the reference case is shown in direct relation to investment costs.

It is evident that when searching for the optimal configurations of the façade the orientation should be carefully taken into consideration. As a rule of thumb resulting from this study, it can be said that increasing the glazing percentage in the upper part of the façade is advantageous in all cases, but as the number of BiPV to install can be large in a south facing façade, it should be restrained in a west facing one, only a few PVs should be installed in an east facing façade, and preferably none on a north facing one.

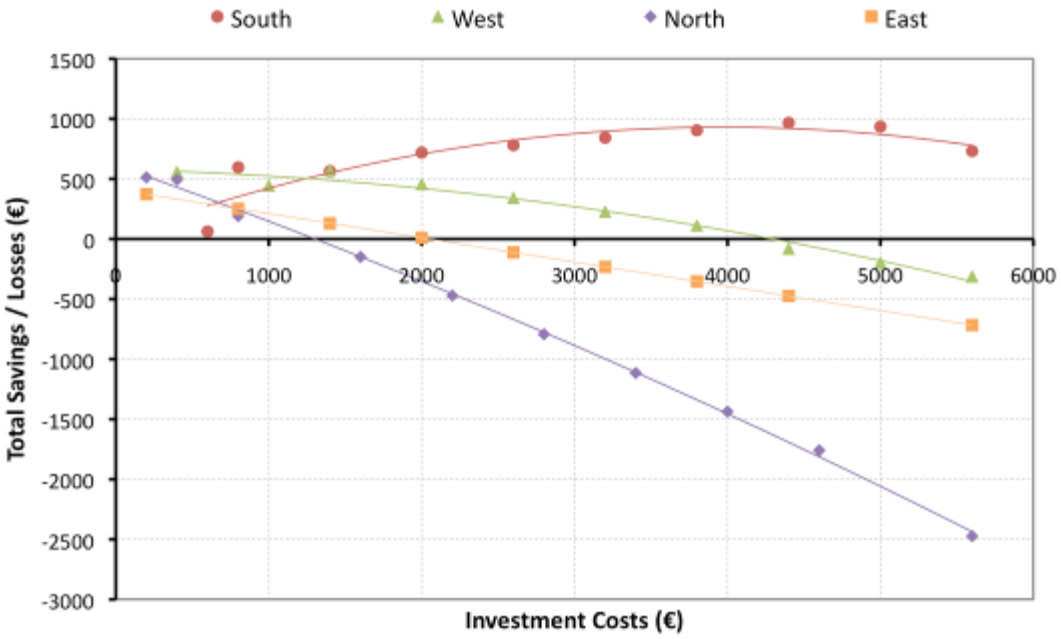


Figure 5.21 – Differential total costs in relation to orientation of the façade

6

Case Study III
Real office building in London

6.1 Description

As the outcome of a collaboration with Arup Façade Engineering (AFE) London, this case study deals with an architectural project that was commissioned to AFE in march 2011 for consultancy. The building in question is a seven storeys building that for the most part accommodates open plan offices; it is located in London and is designed by an English architectural firm.

Figure 6.1 – Rendering of the office building in London

The original enquiry of the architects was for the evaluation of the solar performance of the main glazed façade. In particular the solar heat gains through the windows were to be assessed in order to comply with two different UK standards: "Criterion 3 of Part L2A 2010" and the "BCO (British Council for Offices) Guide to Specification 2009". The first one limits the amount of solar gains from beginning of April to end of September: it sets a limit that is equivalent to the cumulative solar gains through a 1-metre high east-facing window along the width of the façade with a g-value of 0.68. The resulting solar gains through the reference window are 230 kWh/m (per linear meter of facade). The latter recommends a maximum range for solar gains as 50-65 W/m² (of floor area for the first 4.5m of the perimeter) and it is generally referred to when the comfort criterion is considered.
The work carried out by the building physics engineers at AFE consisted in studying a module of the longer façade (pointed out in figure 6.2 and seen from the inside in figure 6.3) in which different percentages of glazing and different shading options were investigated. A thermal model comprehending the chosen façade module and the 4.5m deep corresponding part of the office plan (considered as area of influence of the windows) was build to be used with the simulation software *EnergyPlus*. For each possible configuration an iterative process was employed to derive the *g-value* of the glazed area required to meet the different criteria.

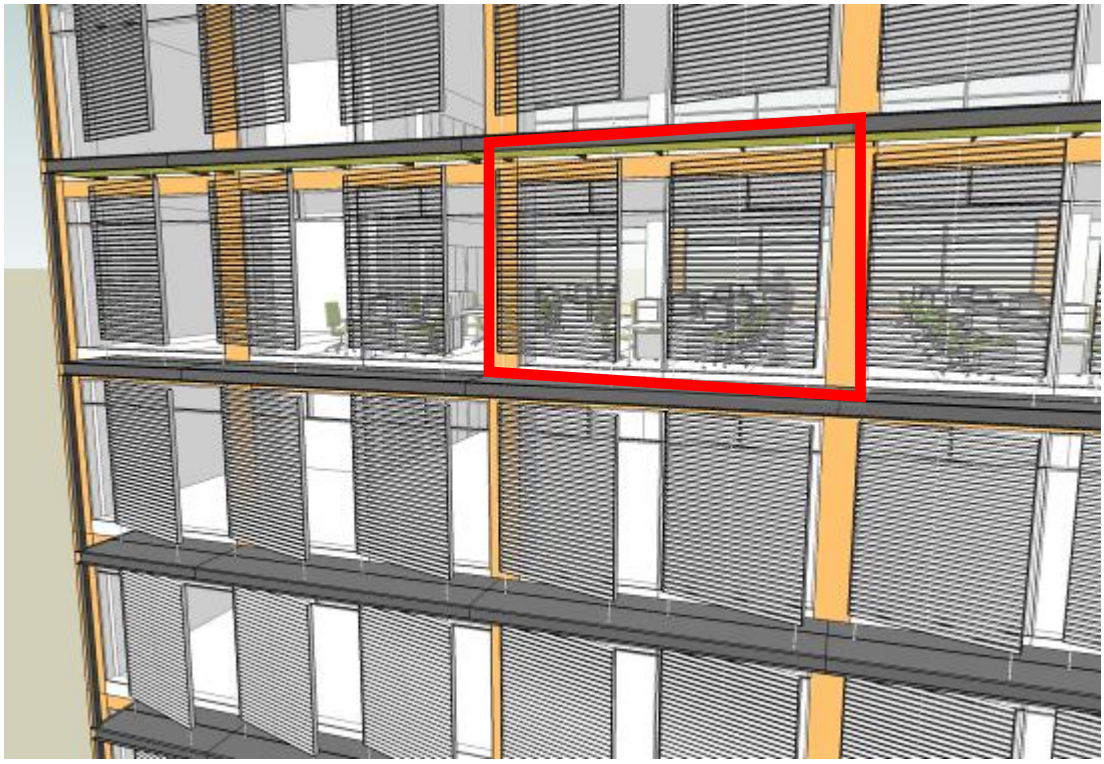


Figure 6.2 – The module of the façade under study (taken from the architectural model)

As a result of the analysis, the maximum *g-values* necessary to meet the different criteria were found for each of the twenty options considered, thus presenting the

architects with a set of possible solutions. However, the solutions could not be said to be optimised, and they didn't take into account other important performance indexes, such as the amount and distribution of daylight in the room for example.
It was thus decided to carry on a new study that involved a wider range of variables, hence expanding the design space, and that implemented an optimisation process to search for the optimal solutions among all the possible ones. It was also decided to add a new design criterion to assess the daylight behaviour of the façade, consequently aiming to solve a double-objective optimisation problem.

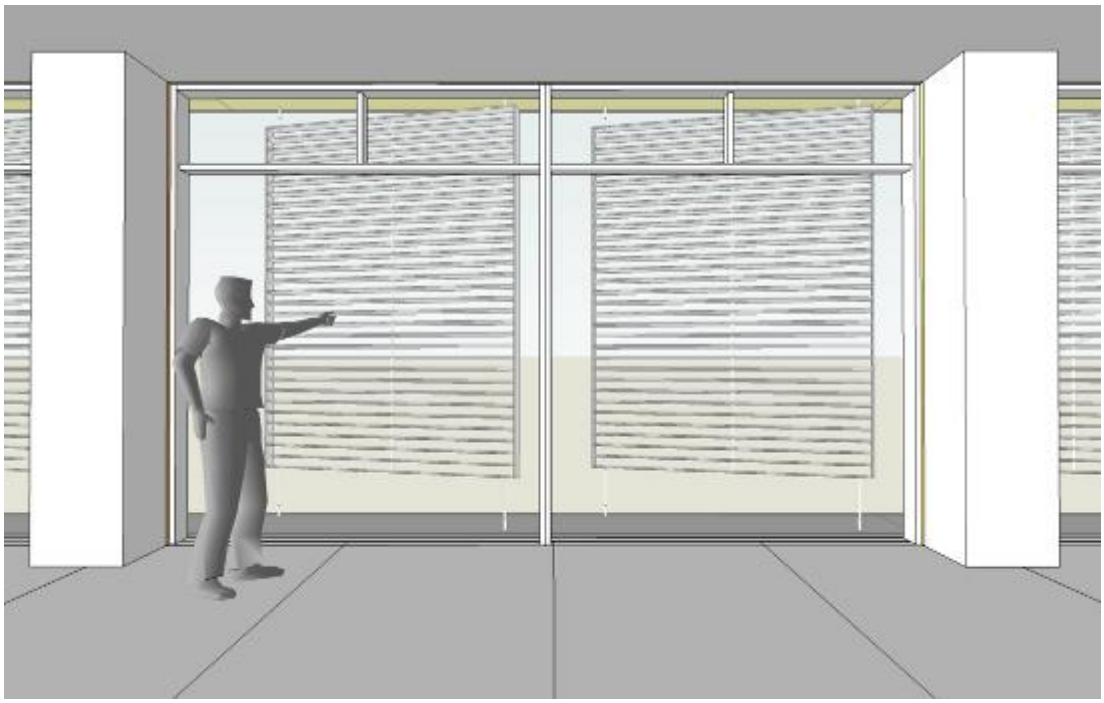


Figure 6.3 – Inside view of the façade module (taken from the architectural model)

6.2 Thermal model

The *EnergyPlus* model developed by AFE building physics engineers was used; only some minor modifications were necessary to make it work with the optimisation program *ePlusOpt*. As mentioned in the previous paragraph, it consists of a module of the main façade, with a width of 6m and a height of 4m; the depth of the room lying behind is fixed at 4.5m to represent the area of influence of windows.

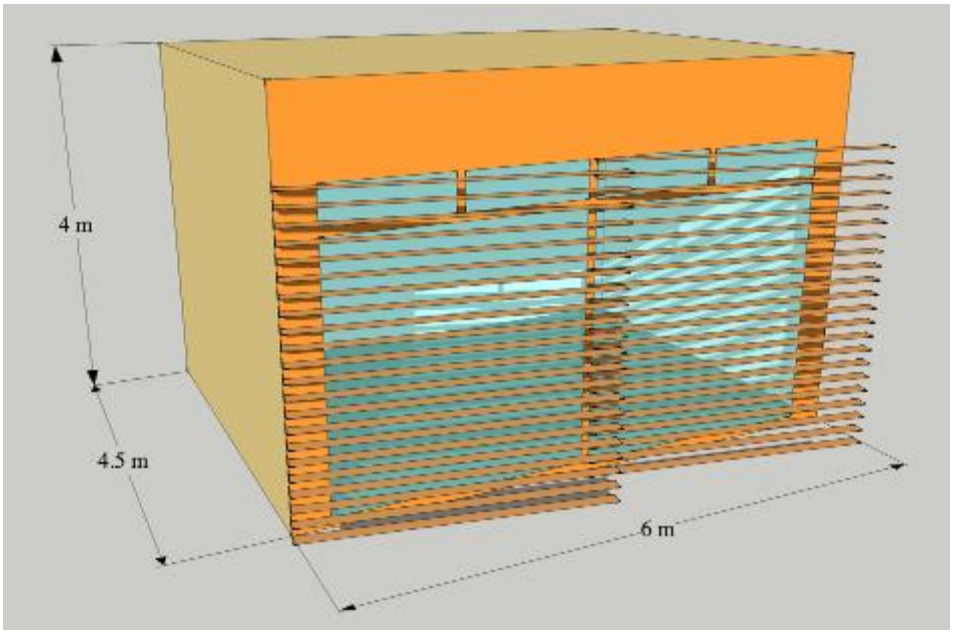


Figure 6.4 – Image of the room modelled in *EnergyPlus*

Since the partitions do not exist in reality because the office is an open plan, they are modelled with a very low surface reflectance (10%) to account for the fact that no reflections of the light rays should occur on these surfaces.

TYPE	VALUE
Internal gains	
Electric equipment	15 W/m^2
People (max 3 people)	120 W/person
Lights	11 W/m^2
Surface reflectance	
Partitions	10 %
Ceiling	90 %
Floor	30 %
Ventilation	
Infiltration	0.2 ach
Mechanical Ventilation	0.8 ach (max)
Operating strategy	
Heating setpoint / setback	22°C / 12°C
Cooling setpoint / setback	24°C / 28°C

Table 6.1 – Model design assumptions

All other design assumptions are summarized in table 6.1. The simulations carried out were one year long and they were based on London Gatwick's weather file.

To control the daylighting in the zone, two reference points were placed in the centre of the room at desk level and at different distances from the façade as pictured in figure 6.5. Besides monitoring the levels of daylight from the windows, they also act as sensors triggering the use of artificial lights in the room. The illuminance design level was set at 500 lux, the lights being turned on whenever this level is not met.

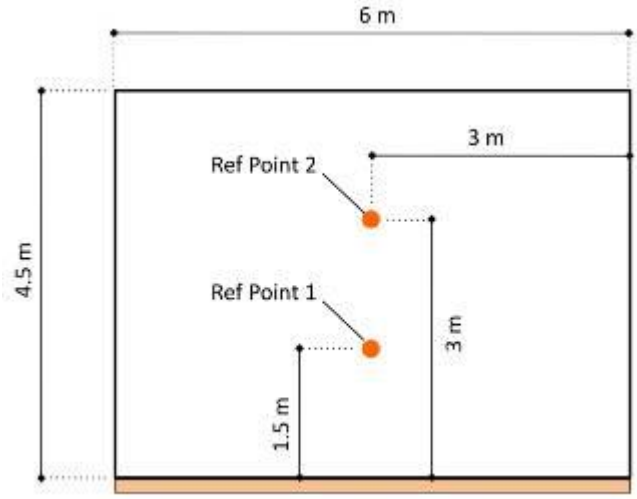


Figure 6.5 – Disposition of daylighting reference points in the model

Different scenarios for the use of automated blinds on the windows were investigated in the study, they will be explained more accurately in the results section. The blinds modelled were opaque slats with a tilt of 45° degrees and a reflectance of 50%.

6.3 Variables

Five variables characterising the façade and the shading devices were selected to be optimised. They are described in detail in the following table and figures. The possible combinations of the variables can give rise to 5040 different solutions.
The glazing can be selected among the five types already described in detail in the first case study. The percentage of glazing of the façade is regulated by the variable that defines the height of the spandrel panel added at the bottom (see next figure): considering only the area that can actually be glazed, it can vary from a maximum of 100 % (no spandrel) to a minimum of 72 %.

VARIABLE	**TYPE**	**Range/Step (if continuous) Possible Values (if discrete)**
Type of glazing	Discrete	LowE, Sel1b, Sel2b, Sel3, Sel4
Spandrel height (H)	Continuous	0 – 900mm / 150mm
Louvres depth (D)	Continuous	100 – 150mm / 10mm
Louvres spacing (s)	Continuous	100 – 150mm / 10mm
Louvres reflectance	Continuous	30 – 60 % / 10 %

Table 6.2 – Variables of the problem

Figure 6.6 – Scheme of façade module with variable spandrel height

The shading device designed by the architects consists in two sets of thin louvres positioned slightly detached from the windows and with a tilt angle of approximately 17.5 degrees from the plane of the façade, used to allow the direct view on a nearby park. Each louvre is 2.4 m long, while the depth and spacing can vary between the bounds identified in table 6.2. The last variables considered is the surface reflectance of the louvres, which can control to a certain extent the amount of daylight reaching the shading device that is reflected inside the room.

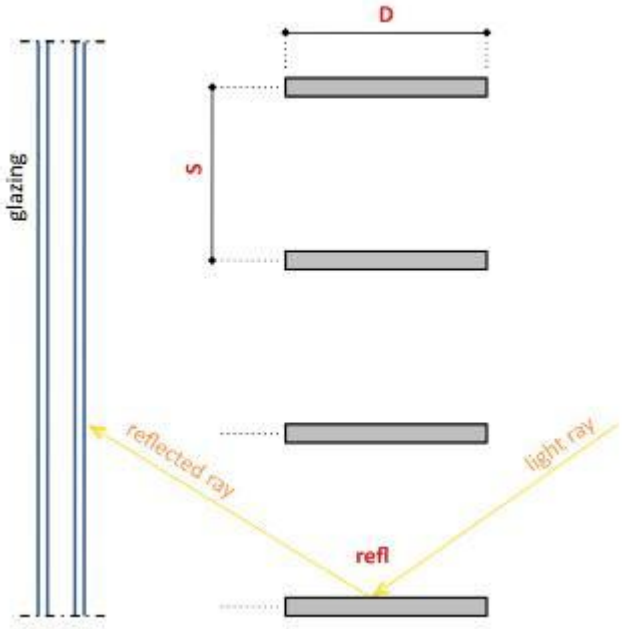


Figure 6.7 – Variable louvres characteristics

6.4 Objective functions

Having to decide which of the standards to base the research upon, the BCO standard was chosen because the study by AFE had shown that it represented a significantly more restrictive requirement than Part L, leading to lower g-values in all cases. Therefore, the first criterion for the optimisation corresponds to the solar gains per floor area for the first 4.5m of the perimeter. More precisely the objective function is defined as the annual fifth higher value of hourly solar radiation transmitted through the windows. The fifth value and not the first one was used to account for possible singularities arising from the simulation of the model.

$$F_1 = SHG_{peak,5th} \quad [\mathrm{W}]$$

For the second criterion, an index of daylighting performance was necessary. The choice fell on the yearly average daylight availability, in percentage of the lighting requirement. Daylight availability (DA) is defined as the amount of available light at a given point and time within a building interior and it can be expressed in percentage of the design lighting level:

$$DA_{refP} = \frac{ill_{refP}}{500} \quad [\%]$$

Only the hours of the day when the office is occupied (7am – 7pm) are considered, and only the working days throughout the whole year of simulation.

$$avg_{year}\left(DA_{refP}\right) = \frac{\sum_{h_{occ}} DA_{refP}}{h_{occ}} \quad [\%]$$

where h_{occ} are the number of hours when occupation is not zero.
The average over the two reference points makes up the second objective function:

$$F_2 = -\frac{avg_{year}(DA_{refP,1}) + avg_{year}(DA_{refP,2})}{2} \quad [\%]$$

The minus at the beginning is needed because the wish is to maximize this function, not to minimize it, in order to have a better daylighting performance.
The two criteria are obviously contrasting, as lower solar gains imply a higher global shading coefficient, which causes less natural light to be let inside the building, and consequently lower daylight availability.

6.5 Results and discussion

No indication about the presence of automated blinds on the window was given by the architects, hence two different models were prepared and two optimisation processes were carried out. The first model had no blinds on the windows, while in the second one the windows were equipped with automated blinds with glare control. The limit on the glare index was fixed at a value of 22 as suggested for offices by

most standards, and glare was evaluated in the model for an occupants view parallel to the façade.

The optimisation processes produced the results displayed in figure 6.8, where the pareto fronts are quite visibly defined. As in the previous case study, the front appears reversed because the second objective (the annual average DA) was maximised and not minimised.

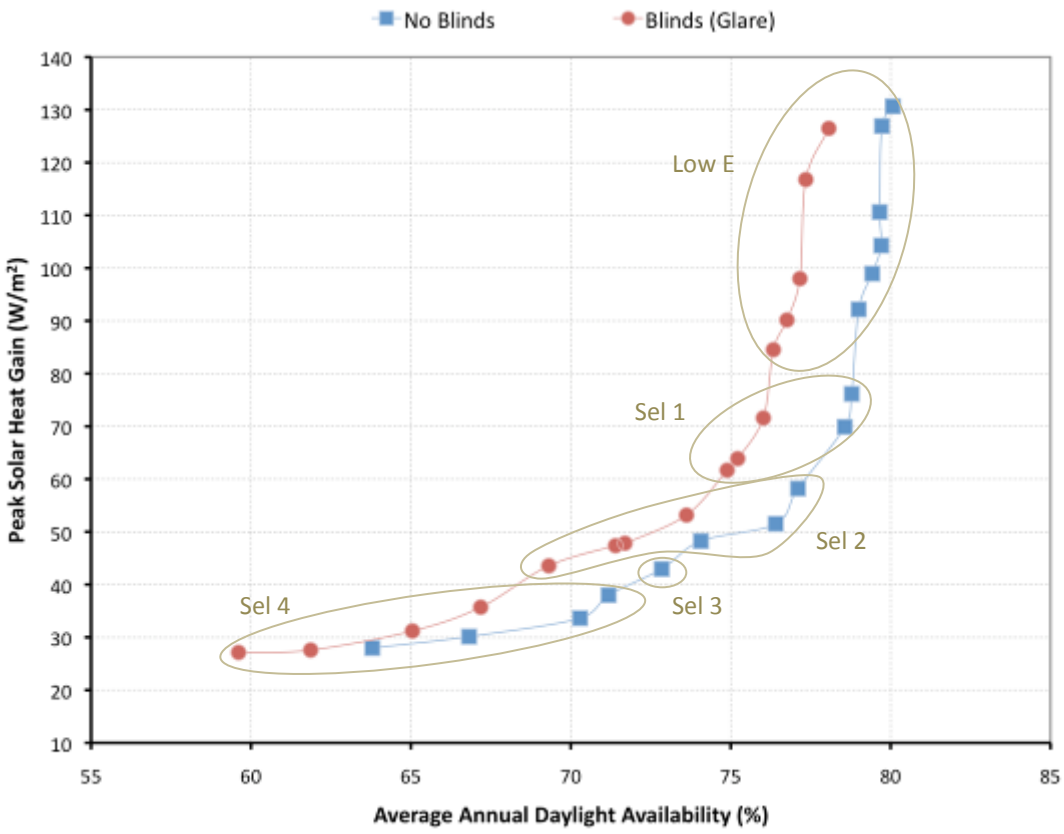


Figure 6.8 – Results of optimisation

Obviously the case with no blinds yielded higher values of annual average daylight availability, as the only shading is provided by the external louvres. However, these solutions would probably cause glare problems, the entity of which will be analysed later. It is interesting to note that with the shading system designed by the architects, it is in any case impossible to achieve a daylight availability higher than 80 % in the case with no blinds and 78 % when automated blinds with glare control are installed.

In both pareto fronts a subdivision of the optimum points based on the type of glass chosen can be observed to follow a straightforward trend. The normal low emissivity glass has a high solar and visible transmittance, so it generates solutions with good daylight availability but high solar gains. As the glass gets darker, the solutions found present lower values for both DA and peak solar gains. Among the points of each of these categories the distinctions in objective function values are the result of the different percentages of glazing and configurations of the louvres selected by the

optimisation algorithm. This demonstrates that generally the shading power of the glazing is stronger than the one of the external louvres.
Before proceeding to analyse the single solutions, it is useful to apply the restrictions dictated by the BCO standard, which, as previously stated, specifies a maximum range for solar gains as 50-65 W/m². This limitations cut out a good number of the solutions found, leaving only the ones in the lower part of the pareto front to be considered as feasible. Two regions have been identified in figure 6.9, corresponding to the peak solar heat gains exceeding the upper and lower bounds of the range set by the standard. The solutions that fall in the first region must be discarded, and these include all the ones featuring simple low emissivity glass.

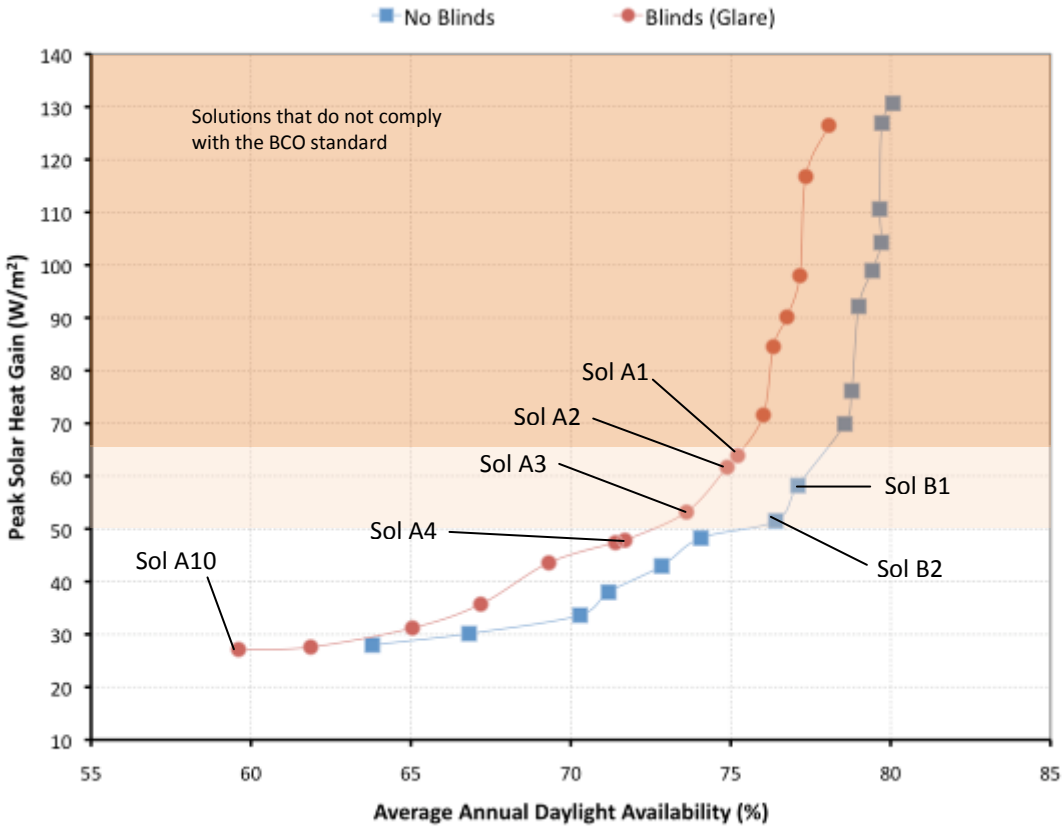


Figure 6.9 – Application of BCO standard to optimal solutions

The façade characteristics of the solutions falling in the second region are reported in the table 6.3 in decreasing order of daylight availability achievable. They comprehend three points on the curve representing the case with the automated blinds and two points on the one representing the case without blinds.
Solutions B1 and A3 present the same façade configuration, except for a slight discrepancy in the depth of louvres. Their diversity in terms of objective functions is caused exclusively by the presence of the automated blinds. From a deeper examination, it can be gathered that in solution B1 the glare index exceeds the maximum limit set for visual comfort 248 hours in the course of a year. It happens

exclusively during the central hours of the day in winter or autumn months, so when the sun's direct radiation can penetrate inside the room more easily. The same happens to various extents for all solutions found for the case with no blinds.

	Spandrel Height (mm)	Glazing Type	Louvres Depth (mm)	Louvres Spacing (mm)	Louvres Reflectance (%)
Sol B1	600	Sel 2B	100	140	60
Sol B2	750	Sel 2B	120	120	60
Sol A1	750	Sel 1B	100	120	30
Sol A2	750	Sel 1B	120	120	30
Sol A3	600	Sel 2B	110	140	60

Table 6.3 – Solutions falling within the range: 50 W/m^2 < peak SHG < 65 W/m^2

Therefore, if a comfort criterion based on the presence of glare is to be considered, all these solutions should be discarded. Nevertheless, they give a good idea of the extent to which the optimal solutions can change if the internal blinds are not considered. For this reason, when optimising the layout of the façade and the external shading device the presence of internal blinds and eventually the strategy for their automation should be known and taken into account. This concept will be further investigated in the next section.

The remaining solutions retain a peak solar heat gain below the threshold of 50 W/m^2, and the corresponding annual average daylight availabilities span from a maximum achievable value of 72 % (Sol A4) to a minimum of 59 % (Sol A10). The configurations of the façade representing all solutions are reported in the following table, once again in decreasing order of DA achievable.

	Spandrel Height (mm)	Glazing Type	Louvres Depth (mm)	Louvres Spacing (mm)	Louvres Reflectance (%)
Sol A4	900	Sel 2B	110	150	30
Sol A5	900	Sel 2B	120	150	30
Sol A6	900	Sel 2B	140	130	30
Sol A7	600	Sel 4	100	140	60
Sol A8	900	Sel 4	100	140	60
Sol A9	900	Sel 4	130	100	60
Sol A10	900	Sel 4	140	100	30

Table 6.4 – Solutions with peak SHG < 50 W/m^2

The resulting façade compositions can be divided in two groups according to the selected type of glazing, however they all present minimum or near minimum glazing percentage. Solutions A4 to A6 have windows with a g-value of 0.35 while

in A7 to A10 the chosen glazing is the darkest, with a g-value of 0.24. Within each group, the louvres change from a more "open" arrangement to a more "closed" one as the peak SHG decreases.

Since all the optimal configurations found have a reduced glazed area and dark or very dark glass, thus being quite penalizing for daylight penetration, the effectiveness of the strategy used for the automation of the blinds was questioned. The visual performance of three meaningful solutions (table 6.5) was compared by calculating the lighting levels inside the room for each of them during a winter day, a spring day and a summer day. The results of this analysis are shown in the following graphs.

	Peak SHG (W/m^2)	Annual average DA (%)
Sol A1	64	75.2
Sol A4	48	71.7
Sol A8	31	65

Table 6.5 – Solutions chosen for daylighting comparison

It's easy to observe that the daylight levels in the office room are very high during summer and spring for all three analysed cases, largely exceeding the design level of 500 lux. However, these big amounts of daylight entering the room most of the times don't cause glare problems, because the external shading is efficient in blocking the direct radiation coming from the sun at high or middle altitudes.

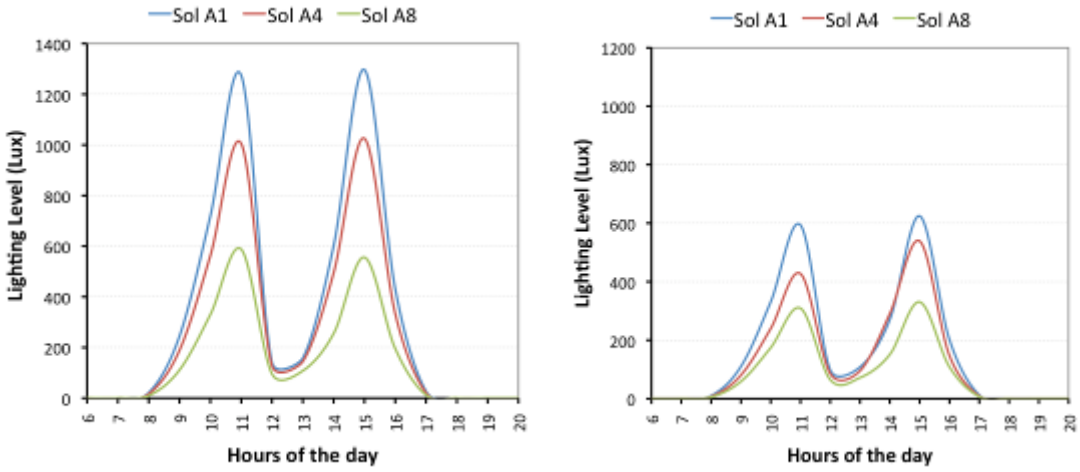


Figure 6.10 – Lighting levels in reference points 1 and 2 for a winter day

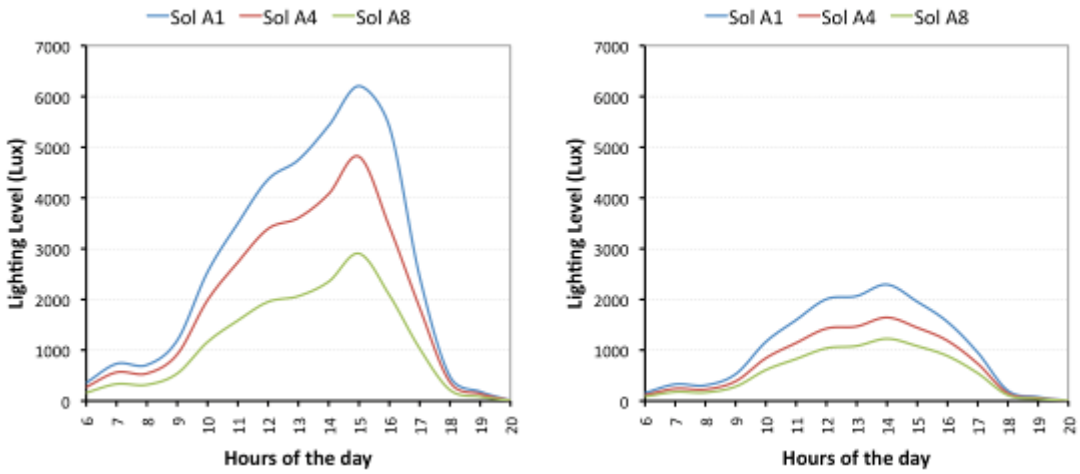


Figure 6.11 – Lighting levels in reference points 1 and 2 for a spring day

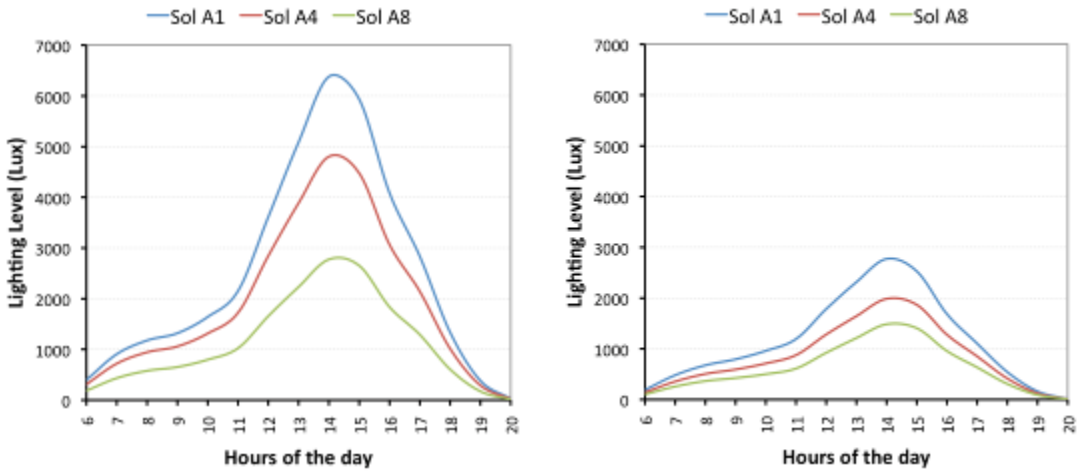


Figure 6.12 – Lighting levels in reference points 1 and 2 for a summer day

On the other hand, in winter the lower position of the sun makes the louvres inefficient in blocking direct rays, causing the blinds to be shut down in the central part of the day to avoid glare problems. The effect of the closed blinds can be clearly seen in the first figure.

When trying to cut down the peak solar heat gain, this strategy proves to be very bad, as the higher solar heat gains are likely to be registered during the hot season, and that's when the blinds operating with glare control are never shut. As a result, the daylight levels are pointlessly very high during these months while no additional shading is provided to avoid high heat gains through the windows. A different strategy for the closure of the blinds based on the actual level of solar radiation that reaches the façade would probably be more efficient in this case.

Consequently, a new automation strategy based on incident solar radiation on the windows was modelled, with a threshold of 300 W/m^2 fixed for the closure of the blinds. An additional strategy that combined both types of automations (glare and

solar radiation) was also implemented and two new optimisation runs were carried out. As expected, the results, displayed in figure 6.13, are much better than the previous ones. The peak solar heat gains are significantly lower and all solutions comply with the upper limit of 65 W/m^2 set by the BCO standard, while the limit of 50 W/m^2 is exceeded only by two points in each case. All this while retaining high levels of annual average daylight availability.

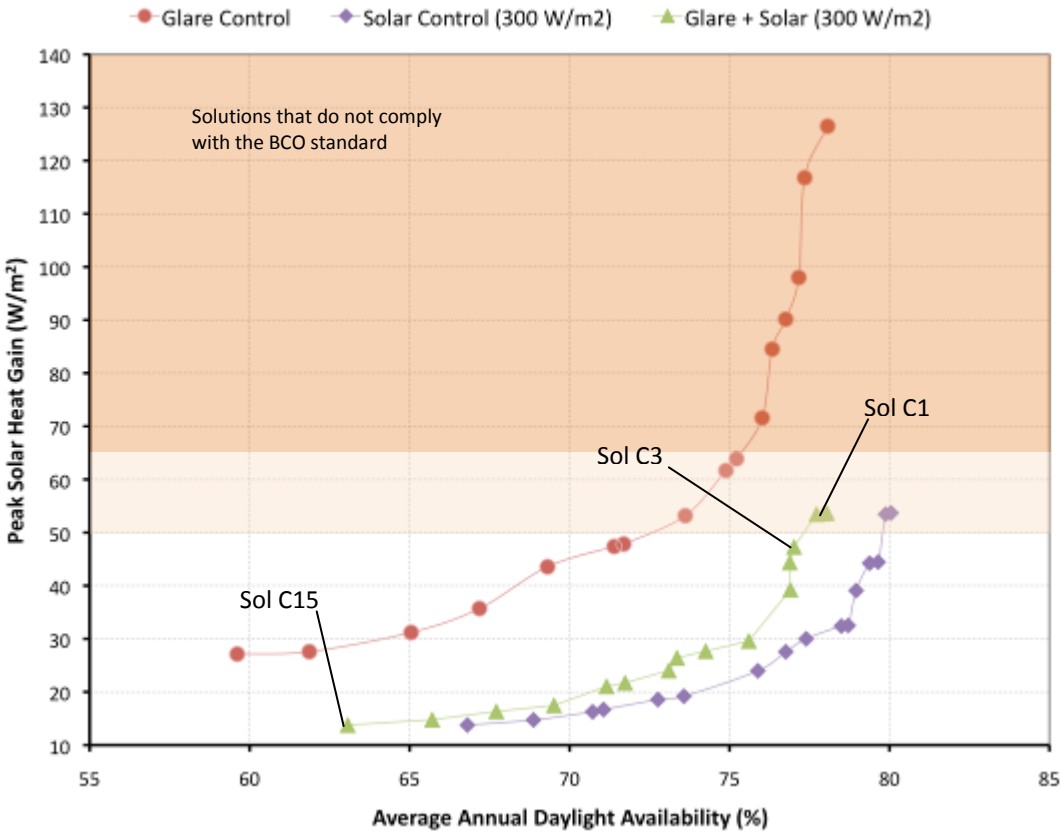


Figure 6.13 – Results of optimisations for different internal shading strategies

The case with just solar control leads to even higher levels of DA, but from inspection of the simulation outputs it can be gathered that in all cases the glare index is exceed for a large number of hours during the year. Hence, although they provide a great performance in both objectives, all these solution should not be taken into account. Table 6.6 reports the variable values for all points of the curve obtained with the combined glare and solar strategy for the blinds. Undoubtedly this is the case that produced the best solutions in terms of trade-off between the two objectives without causing any visual discomfort.

It is interesting to point out that all the optimal configurations found present a very "open" composition of the external louvres, with the minimum depth, the maximum or near maximum spacing, and mostly high reflectance. The reason for this is tightly connected to the specific strategy used for the blinds automation: the criterion on

incident solar radiation causes the blinds to be shut much more frequently than in the case with glare control only (see figure 6.14), thus providing a more constant additional shading to both visual and thermal radiation. Consequently it is preferable to have an external shading that allows more light to pass through, otherwise the daylight in the room would be poor.

	Spandrel Height (mm)	Glazing Type	Louvres Depth (mm)	Louvres Spacing (mm)	Louvres Reflectance (%)
Sol C1	0	Low E	100	150	60
Sol C2	0	Low E	100	150	30
Sol C3	450	Low E	100	140	60
Sol C4	600	Low E	100	140	50
Sol C5	0	Sel 1B	100	140	60
Sol C6	750	Sel 1B	100	150	60
Sol C7	900	Sel 1B	100	150	60
Sol C8	450	Sel 2B	100	140	50
Sol C9	600	Sel 2B	100	150	50
Sol C10	900	Sel 2B	100	140	60
Sol C11	900	Sel 2B	100	150	30
Sol C12	750	Sel 3	100	150	60
Sol C13	900	Sel 3	100	150	60
Sol C14	750	Sel 4	100	150	60
Sol C15	900	Sel 4	100	150	30

Table 6.6 – Solutions for the case with both glare and solar control on internal blinds

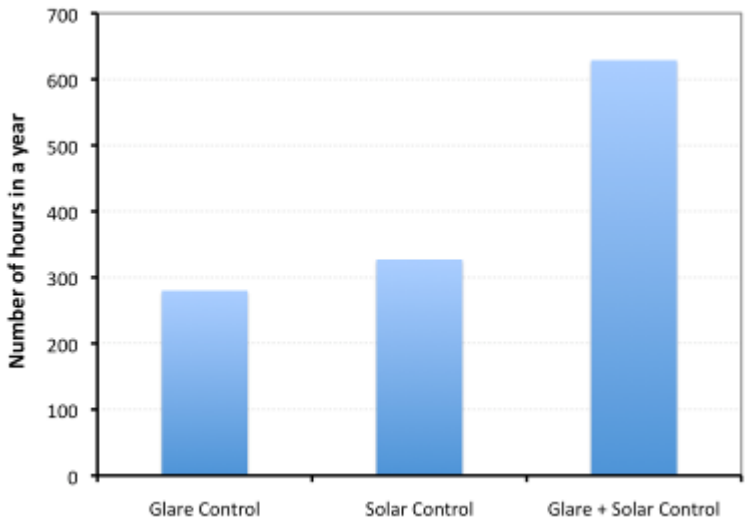


Figure 6.14 – Number of hours in a year in which the blinds are shut for different controls strategies

Thanks to this combined external and internal shading approach the annual average daylight availability still retains high values, and at the same time the solar heat gains are cut down more efficiently during the hot months mainly due to the more frequent closing of the internal blinds (see figures 6.15 to 6.17 that show daylight levels for solutions C1 and C3). In fact, the solar radiation that passes through the windows is in part reflected back by the blinds. As one would expect, this effect is stronger for the glazing types with a higher g-value than it is for the dark selective ones because the ratio of the amount of radiation blocked by the glazing to the one blocked by the blinds increases as the g-values decreases. This can be observed also from the comparison of the pareto front to the one obtained for just glare control on the blinds. The solutions that benefit the more in terms of a decrease in peak solar gains are clearly the ones on the right end of the curve, which are the ones using glazing with lower g-values. From a closer analysis of the results, it can be said that with the solar control on the blinds the peak SHG were cut down by 45-50%. As a result, this scheme for the shading devices supports the use of simple Low-E glass, a type that was completely cut out of the feasible solutions in the previous results because it led to extremely high peak solar heat gains.

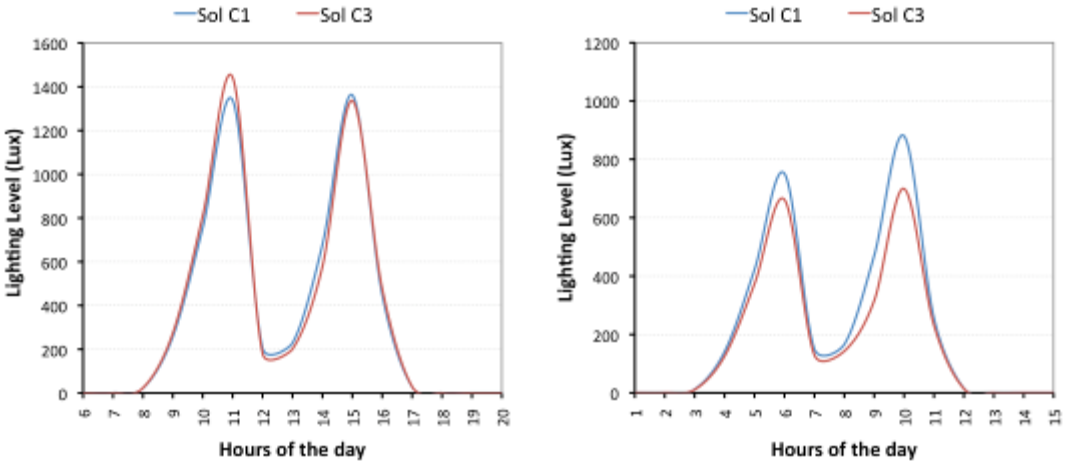


Figure 6.15 – Lighting levels in reference points 1 and 2 for a winter day

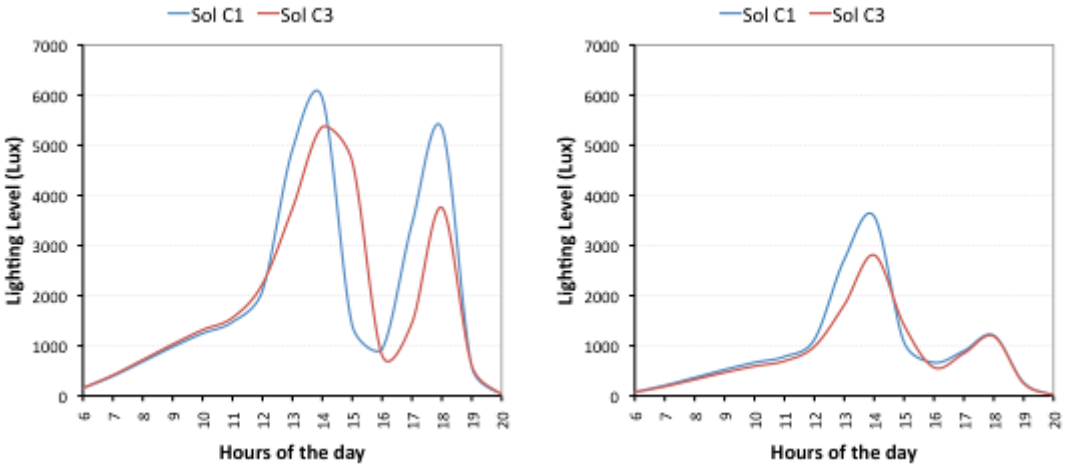


Figure 6.16 – Lighting levels in reference points 1 and 2 for a spring day

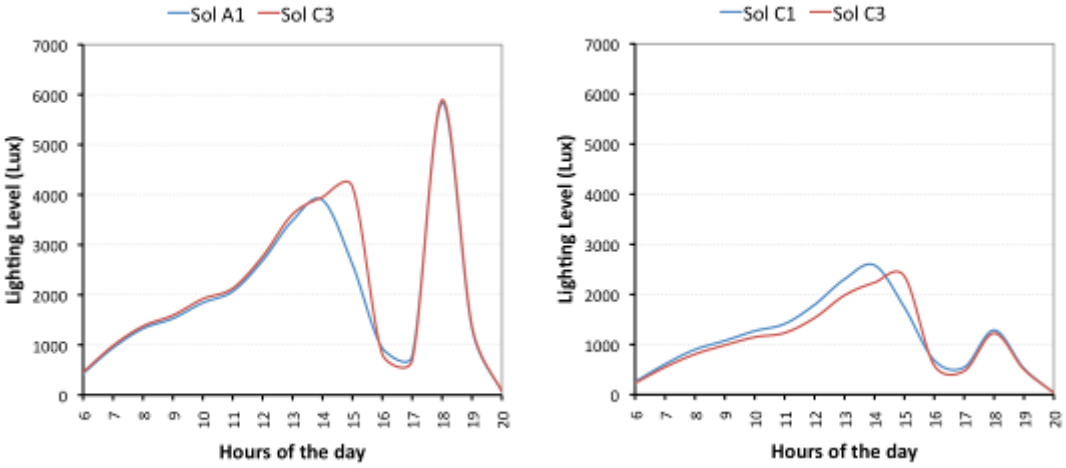


Figure 6.17 – Lighting levels in reference points 1 and 2 for a summer day

Conclusions

The role of the building envelope as interface between the interior and the exterior environment makes it a key actor in the design of buildings that need to respond to strict energy requirements and provide good internal comfort conditions. Designing an energy efficient façade can be a challenging task due to the conflicting requirements arising from its different functions, since the façade is responsible for heat losses, solar heat gains and it allows for daylighting.

Parametric studies involving dynamic energy simulation programs are traditionally used to compare different design alternatives in early stages, but in order to deal with the complexity of the problem additional methods that explore the design space to a larger extent are needed.

In this thesis, the program *ePlusOpt*, an optimisation-simulation tool developed by the author in *Matlab*, is presented and described in depth. *ePlusOpt* integrates energy simulations to be performed with *EnergyPlus* within the optimisation process; it does so by automating the exchange of information between the two programs. Genetic algorithms taken from *Matlab's Optimisation Toolbox* and properly customised are employed for the optimisation. A graphical user interface facilitates the description, arrangement and execution of optimisation problems, given that the energy model to be used for the simulations is supplied by the user. The straightforward use of the program makes it possible to overcome the otherwise complicated practice of setting up simulation based optimisation problems.

The *ePlusOpt* program is then employed to carry out optimisations of façade design for three different case studies. In the first case analysed the general façade of an office building is optimised to minimise the carbon emissions resulting from operation; subsequently the trade-off between cooling loads and artificial lights energy use is investigated through a double-objective optimisation. The comparison of the results to the whole design space which was previously prepared shows that the algorithms, with proper settings of parameters, are efficient in finding the sought-after optimal solutions.

The second case deals with the cost analysis of a façade with building integrated photovoltaics. The two objectives considered are the investment and operation costs. The optimised solutions found provide diverse design alternatives, as they represent different trade-offs between the two cost based objectives. The impact on the results of some design factors, such as the efficiency of the PV panels and the value of the feed-in tariff, is also studied in order to show how it is possible to give additional indications on the preferable solutions according to different design scenarios.

The last case study illustrates how the optimisation process can be applied to practical projects by taking into consideration the façade of a real office building in London. The variables are the type and percentage of glazing and the characteristics

of the external shading device, while the double objective optimisation takes into account the annual peak solar heat gain and the annual average daylight availability. Optimised solutions are found to be effective in finding different compromises between the two conflicting objectives. Considering the complexity of solar shading, different strategies for the automation of the window blinds are investigated, thus showing how the results are deeply influenced by the chosen shading strategy.
The encouraging results found for the case studies confirm that simulation based optimisation can be a valuable instrument in the design of energy efficient façades because it can provide a number of optimised solutions to be presented to the decision makers for the ultimate choice. Moreover, the developed program *ePlusOpt*, thanks to the user friendly graphical interface and the automatic coupling of the programs, has proven to be a good tool to set up and carry on simulation based optimisation processes.

References

[1] Kragh M, Simonella A. The missing correlation between thermal insulation and energy performance of office buildings. In: Proceedings of the International Conference on Building Envelope Systems & Technology (ICBEST), Bath, UK, 2007.

[2] Oral GK, Yener AK, Bayazit NT. Building envelope design with the objective to ensure thermal, visual and acoustic comfort conditions. Building and Environment 2004;39:281-87.

[3] Hendriks L, Van der Linden K. Building envelopes are part of a whole: reconsidering traditional apporaches. Building and Environment 2003;38:309-18.

[4] Ünver R, Akdag NY, Gedik GZ, Öztürk LD, Karabiber Z. Prediction of building envelope performance in the design stage: an application for office buildings. Building and Environment 2004;39:143-52.

[5] Poizaris H, Blomsterberg A, Wall M. Energy simulations for glazed office buildings in Sweden. Energy and Buildings 2008;40:1161-70.

[6] CIBSE, TM35: Environmental performance toolkit for glazed façades. The Chartered Institution of Building Services Engineers, 2004.

[7] Hausladen G, De Saldanha M, Liedl P. Climate Skin, building concepts that can do more with less energy. Birkhäuser, 2008.

[8] Vartiainen E. Electricity benefits of daylighting and photovoltaics for various solar facade layouts in office buildings. Energy and Buildings 2001;33:113-120.

[9] Wright J, Loosemore H, Farmani R. Optimization of building thermal design and control by multi-criterion genetic algorithm. Energy and Buildings 2002;34:959-72.

[10] Caldas LG, Norford LK. Genetic algorithms for optimization of building envelopes and the design and control of HVAC systems. Journal of Solar Energy Engineering 2003;125:343-51.

[11] Palonen M, Hasan A, Siren K. A genetic algorithm for optimization of building envelope and HVAC system parameters. Proc. Of 11th IBPSA Int. Conf. Building Simulation 2009;159-66.

[12] Magnier L, Haghighat F. Multiobjective optimization of building design using TRNSYS simulations, genetic algorithm, and Artificial Neural Network. Building and Environment 2010;45:739-746.

[13] Caldas LG, Norford LK. A design optimization tool based on a genetic algorithm. Automation in Construction 2002;11:173-84.

[14] Znouda E, Ghrab-Morcos N, Hadj-Alouane A. Optimization of Mediterranean buildings design using genetic algorithms. Energy and Buildings 2007;39:148-53.

[15] Tuhus-Dubrow D, Krarti M. Genetic algorithm based approach to optimize building envelope design for residential buildings. Building and Environment 2010;45:1574-81.

[16] Wang W, Zmeureanu R, Rivard H. Applying multi-objective genetic algorithms in green building design optimization. Building and Environment 2005;40:1512-25.

[17] Hasan A, Vuolle M, Siren K. Minimisation of life cycle cost of a detached house using combined simulation and optimisation. Building and Environment 2008;43(12):2022-34.

[18] Stephan L, Bastide A, Wurtz E, Souyri B. Ensuring desired natural ventilation rate by means of optimized openings. Proc. Of 11th IBPSA Int. Conf. Building Simulation 2009;2282-88.

[19] Zemella G, De March D, Borrotti M, Poli I. Optimised design of energy efficient building façades via Evolutionary Neural Networks. Energy and Buildings 2011;43:3297-3302.

[20] Rapone G, Saro O. Optimisation of curtain wall façades for office buildings by means of PSO algorithm. Energy and Buildings 2011; doi:10.1016/j.enbuild.2011.11.003.

[21] Mahdavi A., Mahattanatawe P. A computational environment for performance-based building enclosure design and operation. In: Research in building physics. Rotterdam, In-house publishing; 2003, p. 815-824.

[22] Mahdavi A., Mahattanatawe P. Enclosure system design and control support via dynamic simulation-assisted optimization. Proc. Of 8th IBPSA Int. Conf. Building Simulation 2003;785-792.

[23] Wetter M. GenOpt®, Generic Optimization program. User manual version 2.1.0. Simulation Research Group, Building Technologies Department, Environmental Energy Technologies Division, Lawrence Berkeley National Laboratory. <http://simulationresearch.lbl.gov/GO/ >, 2004.

[24] What is Multiobjective Optimisation. Matlab, Global Optimisation Toolbox User Manual. The MathWorks, Inc. 2010.

[25] Zadeh, L.A. Optimality and Nonscalar-Valued Performance Criteria. IEEE Trans. Automat. Contr., Vol. AC-8, p. 1, 1963.

[26] Censor, Y. Pareto Optimality in Multiobjective Problems. Appl. Math. Optimiz., Vol. 4, pp 41–59, 1977.

[27] Van Veldhuizen D, Lamont G. Multiobjective evolutionary algorithm research: a history and analysis. Report Number TR-98-03 (1998). Wright-Patterson AFB, Ohio: Department of Electrical and Computer Engineering, Air Force Institute of Technology.

[28] Wetter M, Wright J. A comparison of deterministic and probabilistic optimization algorithms for non-smooth, simulation-based optimization. Building and Environment 2004;39(8):989-99.

[29] Mitchell M. An introduction to genetic algorithms. A Bradford Book The MIT Press, 1998.

[30] EnergyPlus Energy Simulation Software, <apps1.eere.energy.gov/building/energyplus/>, 2010.

[31] Auxiliary EnergyPlus Programs, User Manual. Ernest Orlando Lawrence Berkeley National Laboratory; 2010.

[32] Parys W, Bertagnolio S, Saelens D, Hens H. Assessing overall heating efficiency of office buildings. Proc. Of 8th Int. Conf. on System Simulation in Buildings 2010;1-11.

Appendix A

List of functions written for *ePlusOpt* program.

GA Operators

FUNCTION NAME	EXPLAINATION
myCreation	Custom creation function: creates a random initial population of cell array type
myMutation	Custom “uniform” type mutation function: defines how mutation children are created
myXover	Custom “scattered” type crossover function: defines how crossover children are created

Functions For Optimisation

FUNCTION NAME	EXPLAINATION
checkIfSim	Checks whether individual has already been simulated or not. If so, returns its fitness function(s) value(s) and the simulation output variables values
setMyOptions	Sets base options for the GA
populationRule (optional)	Adds a custom written rule to be applied to the creation of the population
runGAopt	Runs a single-objective optimisation for the defined problem using the normal GA
runMOGAopt	Runs a multi-objective optimisation for the defined problem using the MOGA

Template Fitness Functions

FUNCTION NAME	EXPLAINATION
SO_fitFun_Template	Template fitness function for a single-objective optimisation
MO_fitFun_Template	Template fitness function for a multi-objective optimisation

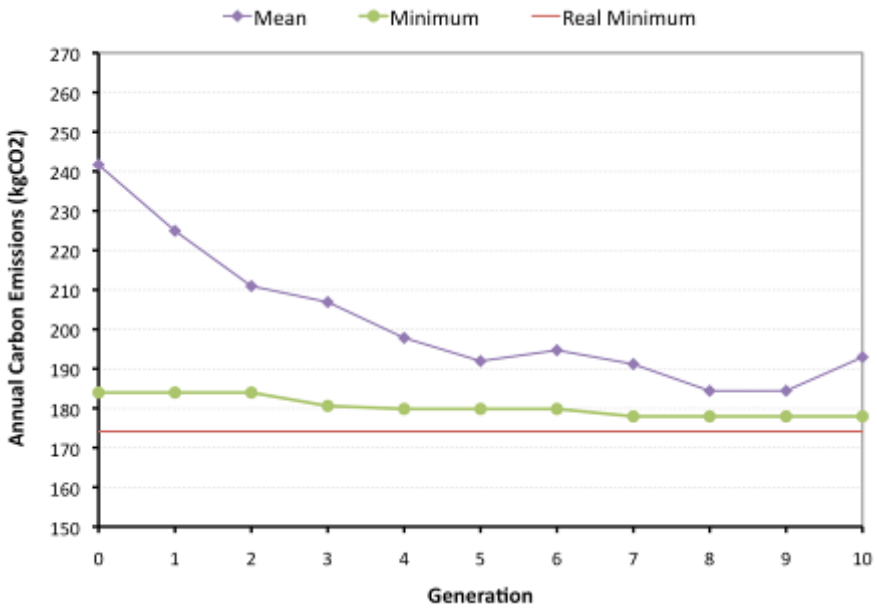
Functions For Energy Plus

FUNCTION NAME	EXPLAINATION
buildEpCall	Builds the command line to call energyPlus with the right input file and weather file
callEplus	Runs a script that contains the command line to launch energyPlus
CSVreader	Reads the ".csv" energyPlus output text file and retrieves the values of the requested "Hourly" output variables
getSimOutput	Manages the provision of the requested energyPlus output variables
meterCSVreader	Reads the "Meter.csv" energyPlus output text file and retrieves the values of the requested "RunPeriod" output variables
modifyParameters	Modifies the simulation main parameters (Location and Buiding orientation) in the "0_Parameters.idf" data set text file
simulationStart	Starts the simulation by calling "writeVariables" first, and then "callEplus"
writeAdditionalVars (optional)	Modifies the values of additional (dependent) variables in the "0_Variables.idf" data set text file
writeVariables	Modifies the values of the variables in the "0_Variables.idf" data set text file

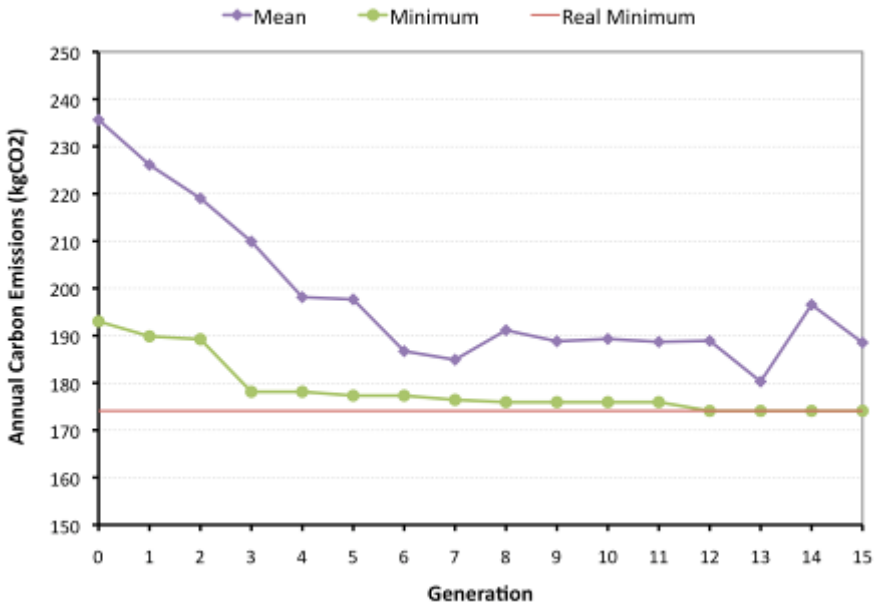
Appendix B

Case study I, single-objective optimisation: progress of the algorithm for different settings of the parameters.

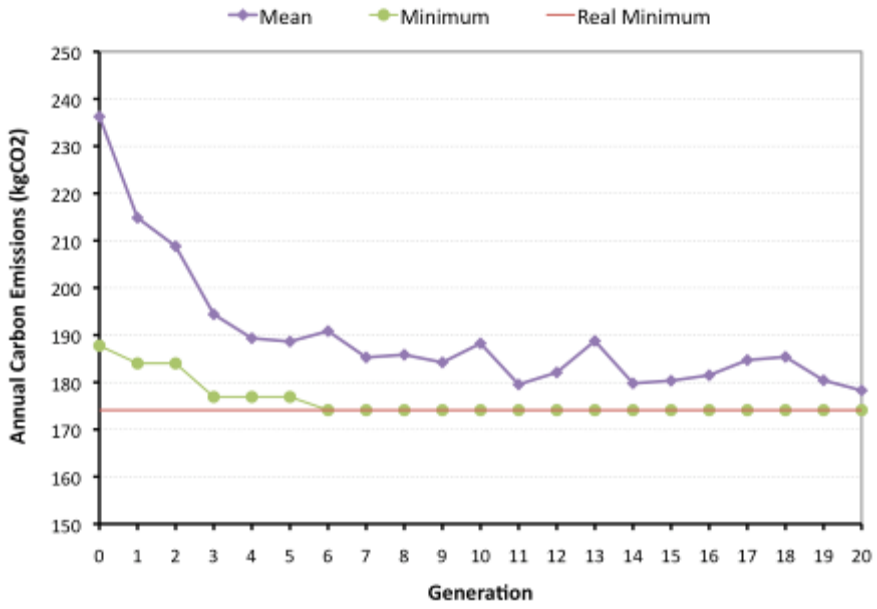
Case 1 – Population 20, Generations 10



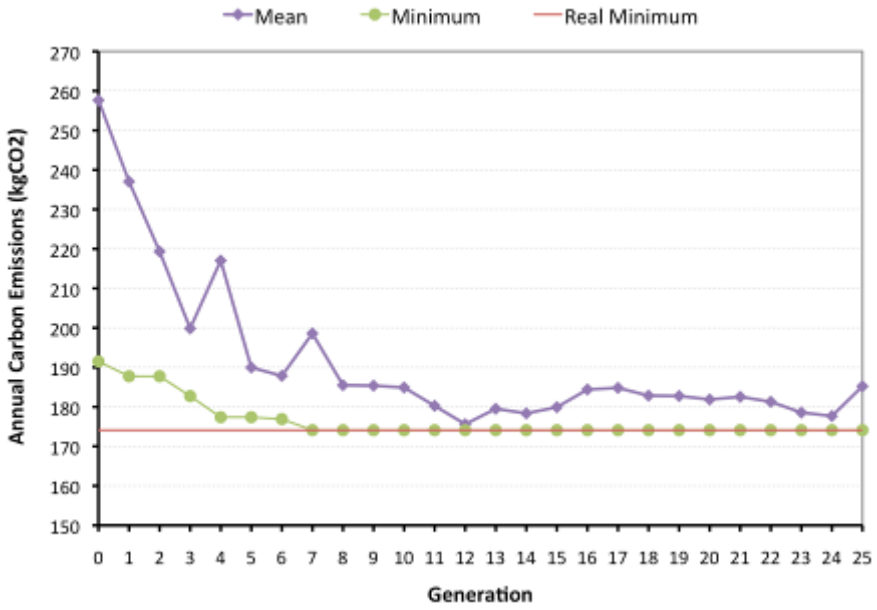
Case 2 – Population 20, Generations 15



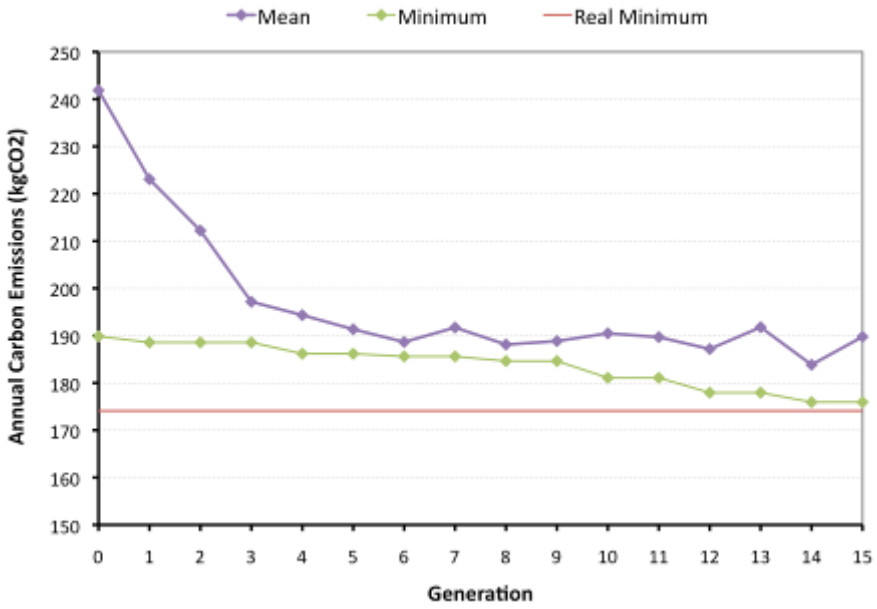
Case 3 – Population 20, Generations 20



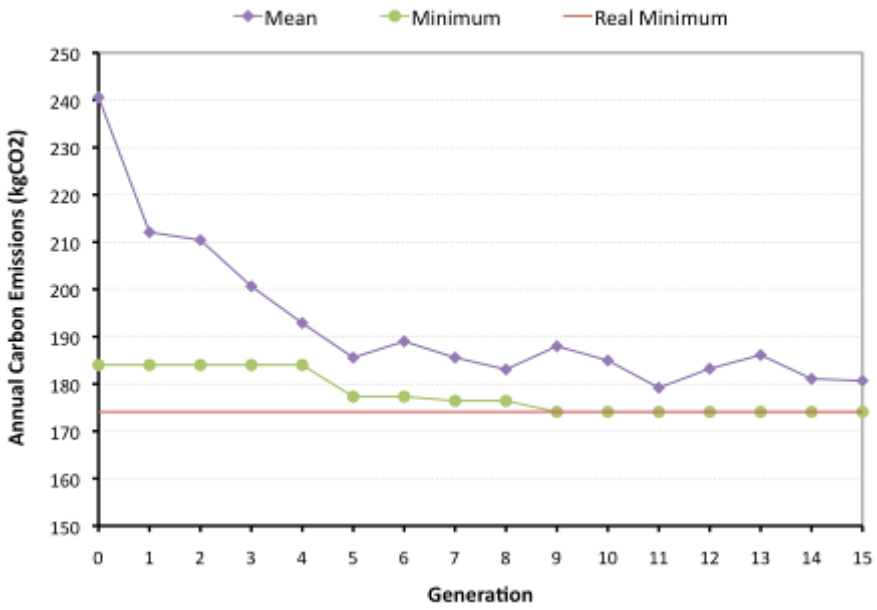
Case 4 – Population 20, Generations 25



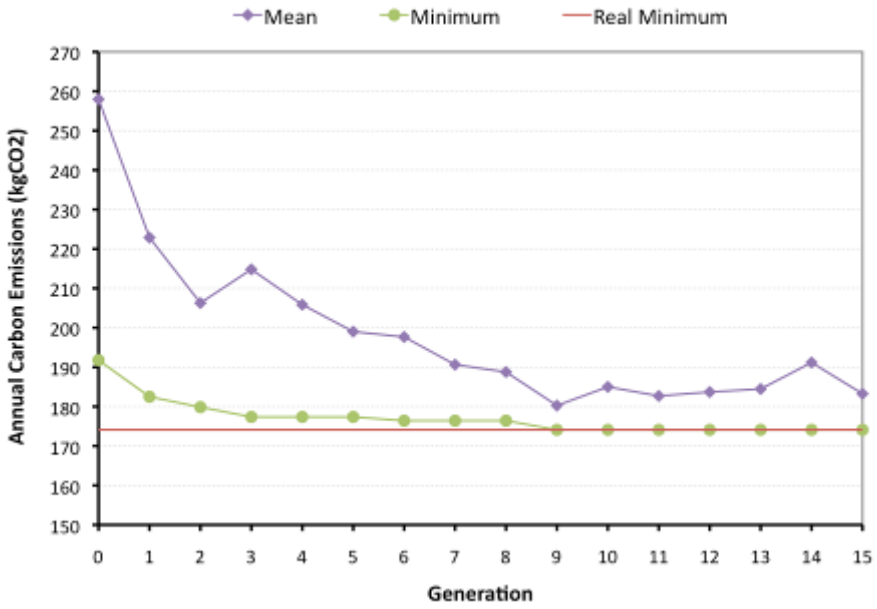
Case 5 – Population 15, Generations 15



Case 6 – Population 25, Generations 15



Case 7 – Population 30, Generations 15



Appendix C

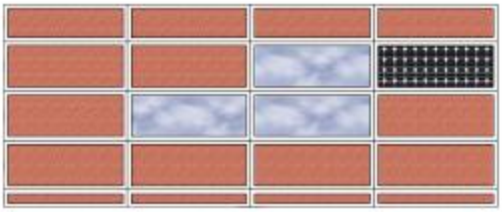
Case study II, double-objective optimisation: all solutions for base case.

Solution 1

Costs	€
Investment costs	600
Difference in operation costs	-662
Total savings	62

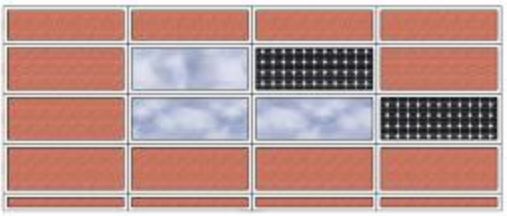
Solution 2

Costs	€
Investment costs	800
Difference in operation costs	-1396
Total savings	596



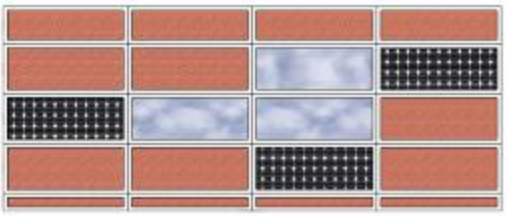
Solution 3

Costs	€
Investment costs	1400
Difference in operation costs	-1965
Total savings	565



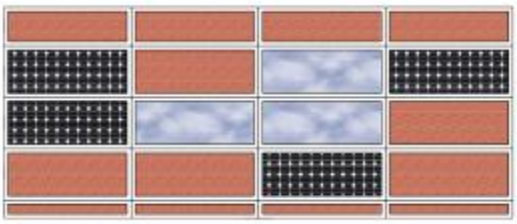
Solution 4

Costs	€
Investment costs	2000
Difference in operation costs	-2719
Total savings	719



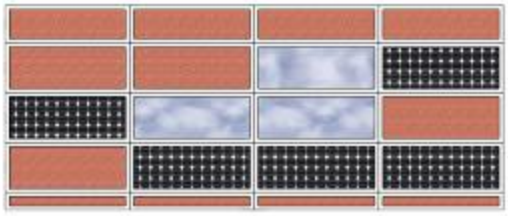
Solution 5

Costs	€
Investment costs	2600
Difference in operation costs	-3381
Total savings	781



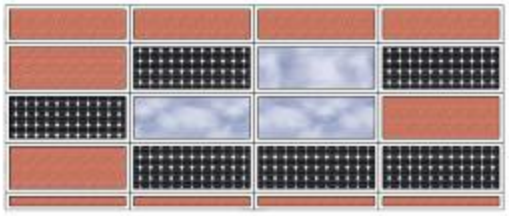
Solution 6

Costs	€
Investment costs	3200
Difference in operation costs	-4043
Total savings	843



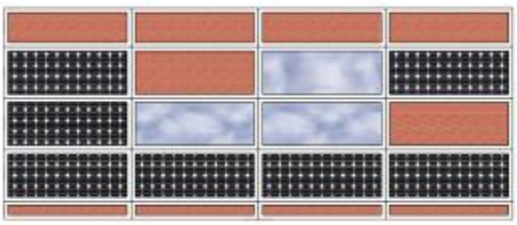
Solution 7

Costs	€
Investment costs	3800
Difference in operation costs	-4705
Total savings	905



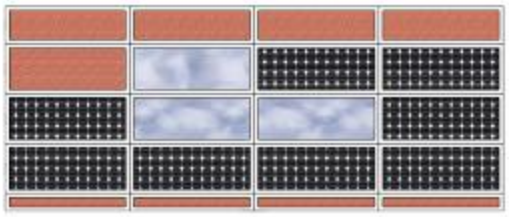
Solution 8

Costs	€
Investment costs	4400
Difference in operation costs	-5367
Total savings	967



Solution 9

Costs	€
Investment costs	5000
Difference in operation costs	-5935
Total savings	935



Solution 10

Costs	€
Investment costs	5600
Difference in operation costs	-6331
Total savings	731