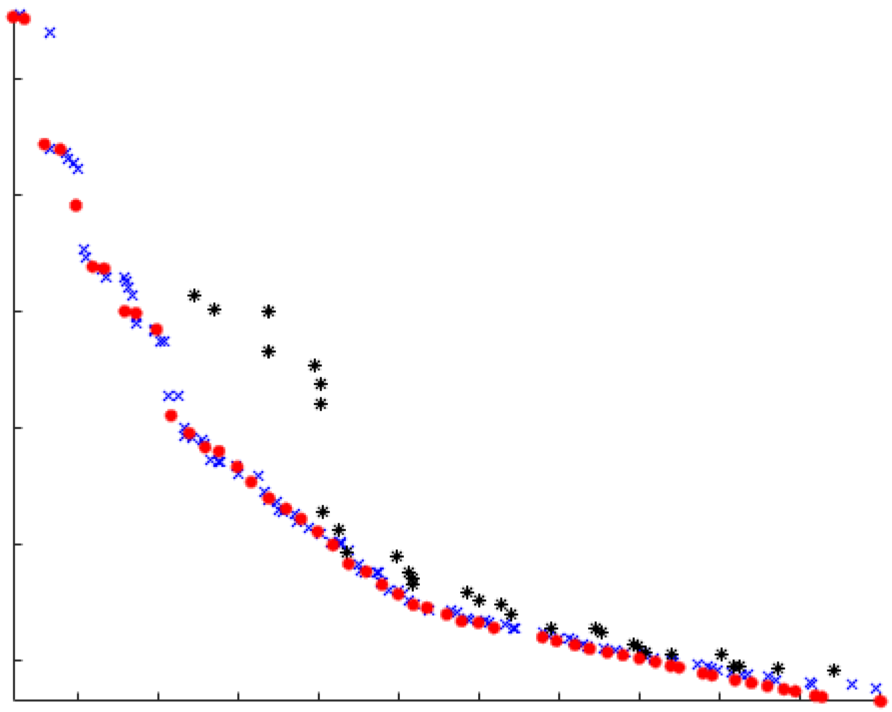
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



Retrofitting of Housing Blocks

Minimizing space heating and cost using combinatorial optimization modelling and algorithms

Master's thesis in Computer Science – algorithms, languages and logic

ANTON DANIELSSON & KRISTIAN ONSJÖ

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2018

Master's thesis 2018

Retrofitting of Housing Blocks

Minimizing space heating and cost using combinatorial optimization modelling and algorithms

ANTON DANIELSSON
KRISTIAN ONSJÖ

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2018

Retrofitting of Housing Blocks
Minimizing space heating and cost using combinatorial optimization modelling and algorithms
ANTON DANIELSSON
KRISTIAN ONSJÖ

Supervisors: Birgit Grohe, Department of Computer Science and Engineering
Ann-Brith Strömberg, Department of Mathematical Sciences
Advisor: Robert Stigsson, Consid
Examiner: Peter Damaschke, Department of Computer Science and Engineering

Master's Thesis 2018
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Optimization result constructed in MATLAB showing how the CPLEX solution (red •) compares to 1 000 (black *) and 10 000 (blue x) generations of NSGA-II. The horizontal axis shows the investment cost, while the vertical axis shows the space heating energy demand.

Typeset in LaTeX
Gothenburg, Sweden 2018

Retrofitting of Building Blocks
Minimizing space heating and cost using combinatorial optimization
ANTON DANIELSSON
KRISTIAN ONSJÖ
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This report details an approach toward finding Pareto optimal solutions to the problem of retrofitting buildings by modelling it as an integer linear optimization problem. The report discusses some background to its importance, e.g., that many buildings do not currently meet the standards set by the European Union for energy efficiency. A current prognosis shows that these standards will not be met in due time, and as such the DREEAM project aims to improve the work being done on retrofitting buildings. The optimization tool DreeamTool, currently being developed within DREEAM, uses a stochastic algorithm, NSGA-II. In the report it is shown how a method based on integer linear programming with multiple objective functions performs, as opposed to the current method. It is shown that the mathematical model and solution approach finds Pareto optimal solutions in a computation time that is considerably shorter than that used by NSGA-II for finding an approximation of the Pareto front.

Keywords: integer linear programming, mathematical modelling, linearization, combinatorial optimization, retrofitting, buildings, neighborhoods, retrofitting of building blocks, energy efficiency

Acknowledgements

We would like to thank our supervisors Birgit Grohe and Ann-Brith Strömberg for their invaluable support in designing the mathematical model, as well as sharing their collected knowledge and aiding in locating resources for us to expand our own knowledge.

We would also like to thank Robert Stigsson and Fredrik Ek at Consid for offering us the opportunity to conduct this research and their support in using DreeamTool so that we could compare our solution to their previous work.

We would like to thank Claudio Nägeli at Chalmers Department of Civil and Environmental Engineering for his informative lectures and guidance that aided us in understanding the physics and mathematics behind retrofitting buildings so that we were equipped with the theoretical foundation to develop the mathematical model.

Finally we would like to thank our examiner Peter Damaschke.

Anton Danielsson, Gothenburg, May 2018
Kristian Onsjö, Gothenburg, May 2018

Contents

1

Introduction

1.1 Background

Today, many buildings do not meet the energy requirements set by the European Union, and as such they are in need of retrofitting[1] in order to eventually meet the requirements. Due to the insufficiency of the present retrofitting rates, new methods are being developed to improve the way buildings are retrofitted. Current approaches focus on single buildings and single components, which hinders the deployment of interconnected renewable systems and may even cause financial losses. A new approach to meet these problems is to retrofit many buildings simultaneously, often with interconnected systems (such as ventilation or hot water supply). This has been shown to decrease energy demand by 10–40% as compared to standard retrofitting approaches. Furthermore, the use of computationally efficient optimization models is being employed. The DREEAM (Demonstrating an integrated Renovation approach for Energy Efficiency At the Multi-building scale) project aims to utilize these tools to further reduce the total energy demand by up to 75% (see [1]). The approach to this is to quantify the retrofitting problem into a set of objective functions and investigate how the choice of components relate to these functions. This results in a multiobjective optimization problem, which can be approached either by stochastic optimization or by deterministic modelling and solution methods. Currently, stochastic optimization methods are utilized in the DREEAM project, but this report shows a proof of concept where a mathematical model has been developed and shown to be a more suitable alternative in regards to speed and accuracy.

When dealing with multiple objective functions, the number of optimization problems that have to be solved grow larger with the number of objective functions, which rapidly leads to a large (and often too large) computational burden. One tool that deals with such problems is the DreeamTool [1] optimization engine used by the project. Today the DreeamTool optimization engine utilizes the Non-dominant Sorting Genetic Algorithm II (NSGA-II), a stochastic optimization algorithm [2]. It is currently limited to optimizing two objective functions with a near future goal to be able to optimize up to three objectives. As with many stochastic algorithms this is computationally heavy and requires a very powerful computer, and/or extensive runtime, along with poor capabilities to expand the number of objective functions.

[1] Retrofitting is a term used for adding a component which was not in the original construction.

The computational models used in the DREEAM project have been developed by a group of researchers at the Department of Civil and Environmental Engineering (CEE) at Chalmers University of Technology [3]. Together with Consid [4] they aim to develop an application which uses these models to find the optimal solutions to retrofitting problems [5].

One example of an optimization problem in the DREEAM project is to optimize retrofitting on an entire building block containing 50 buildings. Consider the case where the buildings have 200 different types of walls, 50 different types of floors, 50 types of windows, and so on. Every wall, floor, and window has 20 different retrofitting options. Furthermore, the water and heat supply can be installed in several ways to provide for several buildings, or individually. These are just some of the possible choices within the algorithm. Currently, the NSGA-II algorithm produces the data required to create a Pareto front, which contains a number of possible construction solutions based on two objective functions. The calculations to provide this data are computationally heavy, and even though the algorithm itself has been parallelized and runs on a cluster of powerful computers it still takes days to get any results on a large–scale problem.

In order to reduce the runtime, an alternative is to instead use combinatorial optimization methods. However, these methods require the formulation of a mathematical model of the retrofitting problem. To our knowledge, there does not yet exist any such model for the problem at hand. Therefore, it is required to analyze the existing equations utilized in retrofitting, (see Appendix A), and reformulate them such that they can be incorporated in a mathematical model for combinatorial optimization.

1.2 Project aim

The aim of this project is, in essence, to improve solutions to the retrofitting optimization problem. This will be done by developing a model of the problem, which is solved using combinatorial algorithms. These improvements may then be candidates for coming versions of DreeamTool, currently being developed at Consid.

The initial approach will be to formulate a mathematical model of the problem. This model will in turn be studied to give an idea of what type of algorithms could be applied, at which point an investigation of existing suitable combinatorial optimization algorithms will ensue. Following this, attempts will be made at applying an algorithm to the mathematical model of the retrofitting problem.

Furthermore, the applied algorithm will then be evaluated in three main aspects, as compared to NSGA-II; runtime, measures of optimality, and number of objective functions to be simultaneously optimized. If either runtime, measures of optimality, or the number of objective functions that can be handled is improved, without significant losses in accuracy of the result as compared to NSGA-II, the model is deemed a success and can be recommended for further research and development.

1.3 The Retrofitting Optimization Problem

As described in Section 1.1 the problem that DREEAM tries to solve is to reduce the energy consumption of the retrofitted building without significantly increasing the associated cost. This problem can be formulated as an optimization problem where the associated equations are as provided by DREEAM, and objective functions are then, in turn, constructed from the variables used in these equations.

The key to this problem is identifying suitable variables and modelling objective functions, with a preference toward linear objective functions if possible without great loss in accuracy. It is also of importance to realize what simplifications need to be done to the objective functions as well as their significance in the final output. Due to this, methods such as linearizing equations are used in the creation of the model as they provide simplification of calculations where careful consideration has been made in order to produce an accurate solution.

1.4 Delimitations

When modeling an integer optimization problem with many variables and several objective functions it is necessary to narrow it down to achieve a simpler setting. In the case of this project the focus was on the model itself, and as such a few delimitations had to be made.

Once the mathematical model was completed and reached a stage where an optimization algorithm was applied, further computational optimization methods such as GPU computations was not in the scope of the project.

There was no in-depth investigation into the performance of different optimization algorithms, but rather the CPLEX algorithm was used as a proof of concept.

Furthermore, as having more objective functions gets increasingly computationally heavy, there was an upper delimitation of three objective functions at a time. No delimitations were be placed on the input parameters, however, meaning that data regarding buildings, location, retrofitting options, and heating systems were all considered.

Lastly, the project would not include any further improvements or research on NSGA-II, the genetic algorithm currently used in the DREEAM project. When the project required comparisons with the genetic algorithm, already conducted research on NSGA-II was studied together with empirical testing.

1.5 Related work

Methods used earlier to solve refurbishment problems have mainly been focused on stochastic optimization algorithms, in particular the NSGA-II algorithm [2], which is a genetic algorithm with several special features. The algorithm sorts the solutions into several so-called fronts. The first front is non-dominated (which means that no values have been found that yield a better result), the second front is dominated by the first, and so on. It also includes an algorithm-specific parameter, referred to as crowding distance. This is a measure of how close a solution is to its neighbours and allows a quantification of the solution diversity, where larger average crowding distance is a measure of better diversity in the population. Individuals in the population with the lowest rank (which is a methodology where the results are queued such that the lowest queue number corresponds to the best result) and greatest crowding distance (solutions furthest apart) are then selected. These individuals are then used as per the simple genetic algorithm, shown in Section 2.4.

For an initial basic understanding of what combinatorial optimization is, how it can be used, and what initial steps to take, the textbook [6] provides a good theoretical foundation. It not only includes optimization, but also how to work with mathematical modelling in order to formulate an optimization problem in a neat, standardized manner. The book also provides examples written in the programming language AMPL that serve to quickly prototype initial attempts to the mathematical model of the optimization problem.

In a sense more specific to this particular type of optimization, similar work has been done in the thesis [7]. There the problem of specifying components in trucks was solved via viewing the problem as a combinatorial mathematical optimization problem where desirable parameters were found to be dependent on the components chosen. This allowed for a mathematical model of how a certain set of components would relate to these parameters. The author also discusses how real-world applications usually need to be reduced as they are too complicated for an exact evaluation. Similar to our work Pareto optimal sets and robust optimization is also discussed in detail.

Due to the complexity of the problem it will likely be important to reduce the set of objective functions such as shown in [8]. By reducing each set of objective functions with similar characteristics to a single objective function the computational burden can be significantly reduced. This is due to the fact that in real-world applications, an approximation will often yield a sufficiently accurate result. Given a brief overview of DreeamTool's available objective functions (seen in Appendix D), it seems likely that these methods may be applicable to the problem studied in this project.

A very similar idea to what is presented in this report can be found in [9] where the authors talk about the life cycle assessment when conducting site-specific retrofitting. They want to evaluate and visualize the difference between different retrofitting options. They also believe it is necessary to further understand the social aspect of retrofitting, such as requests by tenants.

2

Theoretical background

2.1 Multiple Linear Regression

Multiple linear regression refers to linear regression of a vector $\boldsymbol{y}$ of a single dependent variable's values and a matrix $\boldsymbol{X}$ of multiple independent variables' values (see [10]). The aim is to find a vector $\boldsymbol{\beta}$ of coefficients and a residual vector of the function observations $\boldsymbol{\epsilon}$, such that $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ needs to be satisfactorily small.

There are several assumptions made to allow for standard linear regression:

- The independent variables $\boldsymbol{X}$ are assumed to be deterministic. This means that the values are assumed to be fixed, error-free, values. For our case, this property is dependent on the retrofitting model developed by CEE and also holds true for our developed optimization model, as there is no stochasticity in the equations (see Appendix A).
- The resulting approximation $\boldsymbol{y}$ is assumed to be linear when the finding the linearization coefficient vector $\boldsymbol{\beta}$. Often a nonlinear function can be modelled via piecewise linear functions, which is the case in the optimization model developed in this thesis.
- A constant variance is assumed in the error vector $\boldsymbol{\epsilon}$. In practice this is not possible, since larger independent variable values will have a larger variance as compared to smaller values. However, the variance may be scaled for each variable to be placed on a comparable scale.
- Independence of errors is assumed in that, whenever $i \neq j$, the error ϵ_i does not affect ϵ_j, i.e. errors do not propagate.
- The data set of independent variables $\boldsymbol{X}$ and corresponding dependent variables $\boldsymbol{y}$ are assumed to be one to one, meaning that every row of independent variables is complete and has a corresponding dependent variable. One method to handle this assumption is to ignore all rows with incomplete independent variables and their associated dependent variable. It is important that this does not reduce the number of observations below that of the number of independent variables. If the number of observations were to be lower than the number of independent variables it would yield an underdetermined system of equations, in which case this method is unusable. However, if there are thousands of observations and only a few variables, as is the case in this thesis, it is deemed an unlikely scenario.

Ordinary least squares (see Chapter 1 in [11]) was chosen as the method to solve the multiple linear regression model developed in this thesis, shown in (2.1), due to being the most commonly used method:

$$\underset{\beta\in\mathbb{R}^5,\epsilon\in\mathbb{R}^5}{\text{minimize}}(\boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{\epsilon}),$$
$$\text{subject to } \boldsymbol{\epsilon} = \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}.$$

2.2 Multi-objective optimization

Multi-objective optimization is a type of optimization where there are several, possibly conflicting objective function to optimize [12]. As the objective functions are conflicting they can not be optimized in a separate instance for each objective but instead share most of their variables and therefore have to be considered simultaneously. Genetic algorithms attempt to solve these problems using all the different objective functions as is, as described in Section 2.4. When using a method such as linear programming, the optimization model is modified such that all but one objective is modelled as a constraint in the model. This leads to a model which may be considerably more complicated than the original optimization model, and which needs to be solved for several settings of each of the added constraints. One way to simplify a multi-objective problem into a single objective is with a method called the ϵ-constraint method [13].

The **ϵ-constraint** method is a method where one of the objective functions is chosen to be optimized. The other objective functions are then described as constraints. These constraints are then updated to achieve optimal solutions to the new problem instance. Combining the results of these problem instances can give a good idea of how the optimal solutions change depending on the constraints. A good representation of these results is a Pareto front where each axis represents one objective function and the resulting points on the front represent the values of the different solutions, where each solution is optimal in a trade-off between the objective functions.

2.3 Integer Linear Programming

Linear Programming (LP) is a method used both for modelling, and for finding the optimal value in a mathematical model, where the constraints and objective function are linear. They can be expressed in canonical form as Equation 2.1.

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}, \\ &\text{subject to} \\ &\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \\ &\boldsymbol{x} \geq 0. \end{aligned} \tag{2.1}$$

There exist several different algorithms for solving LPs, the most common being the simplex method and interior point methods [6].

Integer Linear Programming (ILP) can be used for optimization problems where the domains of the variables are restricted to integers only. An ILP can be expressed as in Equation 2.2.

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{Z}^n}{\text{minimize}} \quad & \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}, \\ \text{subject to} \quad & \\ \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}&, \\ \boldsymbol{x} \geq 0&. \end{aligned} \tag{2.2}$$

Solution methods for ILP usually consist of a top level search, such as branch-and-bound or branch-and-cut, and a subroutine for solving LPs (see [14], Session 1). The simplex method was developed by Dantzig in 1947 [15]. It is often explained by visualizing the problem as a polytope formed by the constraint where the algorithm "walks" along the edges of the polytope, visiting a sequence of corner points; see Figure 2.1. The corner points represent different solutions of the LP and it can be proven that optimal solutions always can be found at corner points (see [16]). The algorithm makes sure to always move to a corner point that is not worse than the current corner point and it will thus, in a finite number of steps, end up in an optimal solution. The algorithm runs in exponential time in the worst case, but behaves polynomially in practice (see [17] Chapter 13, [18] and [19]). In fact, for many applications the simplex method outperforms interior point methods even if the latter has a running time that is polynomial in the worst case (see [20] Chaper 1).

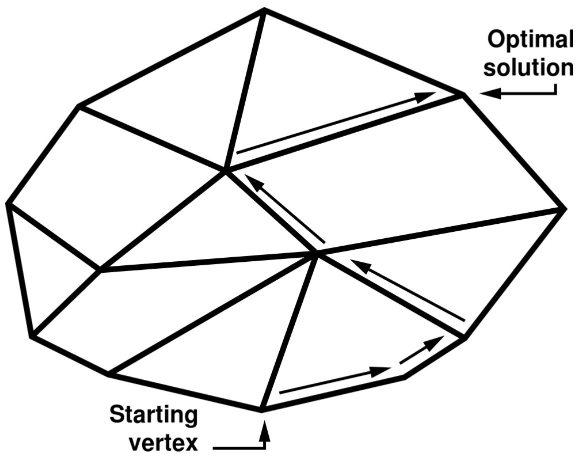


Figure 2.1: An illustration of how simplex searches for the optimal value in a polytope during each iteration until an optimal solution is found. (Illustration from `https://commons.wikimedia.org/wiki/File:Simplex_description.png`)

There exist a number of software packages for solving ILPs. One of them is CPLEX [21], which is an ILP optimizer by IBM. For our computations we chose the simplex method in the CPLEX ILP-solver settings.

2.4 Genetic Algorithms

Genetic algorithms are based on the theory of evolution and include features such as mutations, populations, and survival of the fittest. They are generally used to search for near-optimal values and are often employed in complex scenarios where algebraic simplifications of the problem are difficult to make. Below follows a description of a basic genetic algorithm [22] as well as the pseudocode representation in Listing 2.1.

Step 1: Initialize a population of N chromosomes as binary number representations of the natural number variables in a function f of N inputs. Each chromosome is of length M, where M is the number of digits required for the binary number representation of the largest possible variable.

Step 2: Evaluation

- Step 2.1: The N chromosomes are decoded to the variables' decimal values as variables $\boldsymbol{x}_n$, $n = 1, \ldots, N$.
- Step 2.2: The objective function f is evaluated and a fitness value F_n is assigned to individual n as $F_n := f(\boldsymbol{x}_n)$, $n = 1, \ldots, N$.

Step 3: Next generation of the population

- Step 3.1: Two chromosomes are selected at random; a higher fitness value increases the likelihood of being selected. Once selected, they are removed from the population.
- Step 3.2: **Crossover:** Two new chromosomes are created by combining parts of the two chromosomes selected in Step 3.1. This is done by splitting the chromosomes at either a random or previously defined index m, where $1 \leq m \leq M$, and combining the resulting pieces.
- Step 3.3: **Mutation:** Every bit in the two chromosomes created in Step 3.2 has a small probability to mutate and flip value.
- Step 3.4: The steps 3.1–3.3 are repeated until all chromosomes in the population have been evaluated and a new generation has been created.

Step 4: Repeat from **Step 2** with the newly created generation until a termination criterion has been met.

```
Inputs: instance Π,
        size n of population ,
        rate β of elitism ,
        rate γ of mutation ,
        number δ of iterations
Output: solution x
//  Initialization
    Generate n feasible solution randomly;
    Save them in the population pop;
    Loop until the terminal condition
    for i=1 to δ do
//  Elitism based selection
        number of elitism ne = n · β;
        select the best ne solutions in pop and save them in pop_1;
//  Crossover
        number of crossover nc = (n − ne)/2;
        for j=1 to nc do
            randomly select two solutions x_A and x_B from pop_1;
            generate x_C and x_D by one−point crossover to x_A and x_B;
            save x_C and x_D to pop_2;
        endfor
//  Mutation
        for j=1 to nc do
            select a solution x_j from pop_2;
            mutate each bit of x_j under the rate γ and generate a new
                solution x'_j;
            if x'_j is unfeasible
                update x_j with a feasible solution by repairing x'_j;
            endif
            update x_j with x'_j in pop2;
        endfor
// Updating
        update pop = pop_1 + pop_2;
    endfor
// Returning the best solution
    return the best solution x in pop;
```

Listing 2.1: Algorithm example of a simple genetic algorithm[23].

3

Methods

3.1 A mathematical model of the retrofitting optimization problem

In this section the ILP model of the building retrofitting optimization problem is presented. This model consist of variables, a linear objective function, and linear constraints. In our model we will use binary variables which will be called decision variables in the remainder of this chapter. The following subsections will motivate the design choices, such as which parameters to use as independent variables, in order to formulate an ILP model of the retrofitting optimization problem.

Decision variables for the different combinations of *spaces* and *component types* (types of windows, walls, ceilings, floors, and ventilation systems) will be introduced along with constraints and the objective functions for costs and space heating in order to describe the model [24, 25]. Since the space heating objective function is nonlinear, linearization is used to be able to construct an ILP model.

It should also be noted here that the different model parts are described twice. First a general explanation of the reasoning behind the modelling follows in Sections 3.1.1–3.1.2, and in Section 3.4 the complete model is presented. The term *space* used in this chapter refers to the physical space where a specific component is to be placed, for example the hole in a wall where a window is to be placed.

3.1.1 Decision variables

For the space heating equations described in Appendix A it seemed to suit well to let the decision variables be a combination of spaces and component types. For example, for windows the decision variable x_{ij} equals 1 if the type of window $j \in \{1, \dots, n\}$ is used on window space $i \in \{1, \dots, N\}$, and 0 otherwise. This concept applies to all decision variables. The component types are in turn associated with a set of parameters which relate to the space heating equations, such as insulation, and ventilation effectiveness. An example follows based on the window decision variables as the other components are handled analogously, with the exception of the ventilation system.

The Matrix (3.1) represents the decision variables for windows; each row corresponds to a specific space to be retrofitted, and each column corresponds to the type of component chosen. This general approach facilitates future developments, allowing different types of components on different levels of the buildings. Currently, for all spaces on any wall in a certain cardinal direction, the same component type is used; hence all such spaces are in the mathematical model considered as a single space in the mathematical model. The ventilation system is assumed to be chosen for the entire building, which means that it can be represented by a vector, where each element corresponds to one type of ventilation system. Further details regarding decision variable notations are covered in the matrices in (3.5).

$$\mathbf{x} := \begin{bmatrix} x_{11} & \dots & x_{1n} \\ \vdots & \vdots & \vdots \\ x_{N1} & \dots & x_{Nn} \end{bmatrix} \tag{3.1}$$

By also including the current component in the decision variable matrix, one can choose *not* to retrofit the component, but instead leave the original in place. Should it be absolutely necessary to retrofit the component the original component type can simply be excluded from the decision variable matrix, and thus not be considered as an option.

3.1.2 Constraints

The first constraint is general and enforced on each space. This constraint expresses that only one component type may be chosen for each space; it is described for windows as,

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad \forall i = 1, \dots, N. \tag{3.2}$$

An analogous reasoning is made for all components, including ventilation which possesses only one space. The user specified constraint allows limitations on which component types are allowed in which spaces and is represented by a 0-1 matrix $\boldsymbol{\lambda}$ of the same dimensions as the corresponding decision variable matrix. An example of a zero in this matrix, based on the conditions of the northern hemisphere, could be that a certain window type may not be allowed to face north, which is a generally colder direction, because that window type may have a too low insulation value, and as such would diffuse too much cold air. This is formulated for window type i, j as

$$x_{ij} \leq \lambda_{ij}, \qquad \forall i = 1, \dots, N, \quad \forall j = 1, \dots, n, \tag{3.3}$$

where $\lambda_{ij} = 1$ means that component type j is a feasible option for space i and $\lambda_{ij} = 0$ means that it is not. Analogous restriction matrices follow for all components.

More detailed explanations, including the notation used for the other components, will follow in Section 3.4.

3.2 Simulations of the space heating equations

Efforts were made in the project towards analyzing the space heating function algebraically, but as the complexity of the function proved this to be too hard, efforts were instead shifted toward a numerical simulation of the function. The simulation was based on Excel spreadsheet data provided by CEE and Consid, in order to explore what further steps could be taken towards creating the mathematical model. The data was created through measurements made every hour for an entire year at an undisclosed location. The cost functions proved far more manageable and could be used without any significant alteration. A note should be made here that the component type parameters in the optimization are variables in the simulation and linearization models to be described in this section.

The numerical simulations were made in MATLAB via the equations given in formulas (A.1)–(A.38) as provided by the DREEAM project. Values for all the constants were extracted from the provided data samples. Via a thorough analysis of what parameters the different components affect we concluded that heat transfer coefficients for windows U_{w} and opaque surfaces U_{op}, ventilation effectiveness q, and glazed surface ratio multiplied by solar transmittance $F_{\mathrm{g}}g$ were the core parameters being altered by the choice of components. Aided by discussions with experts at the DREEAM project these variables were set to intervals of feasible values [26]. A graphical representation of how the space heating changes, as illustrated in Figure 3.1, depending on U_{w} and U_{op} was created. This served as a foundation to determine which parameter would be best suited for a piecewise linearization. All permutations of variable pairs were tested and the heat transfer pair was found to result in the best fit. The fit was determined by reviewing the maximum error of the linearization as compared to the measurement data. This was complemented with a graphical analysis to give a rough confirmation that the linearization produced a model similar to the measurement data. The figures in Appendix C.1 show the dependency of the space heating function on $F_{\mathrm{g}}g$ where a linear relationship can be imagined. In the linearization $F_{\mathrm{g}}g$ is considered as a single variable, as every window includes both an F_{g} and g value, and these are precomputed to a single parameter $F_{\mathrm{g}}g$ for every window component.

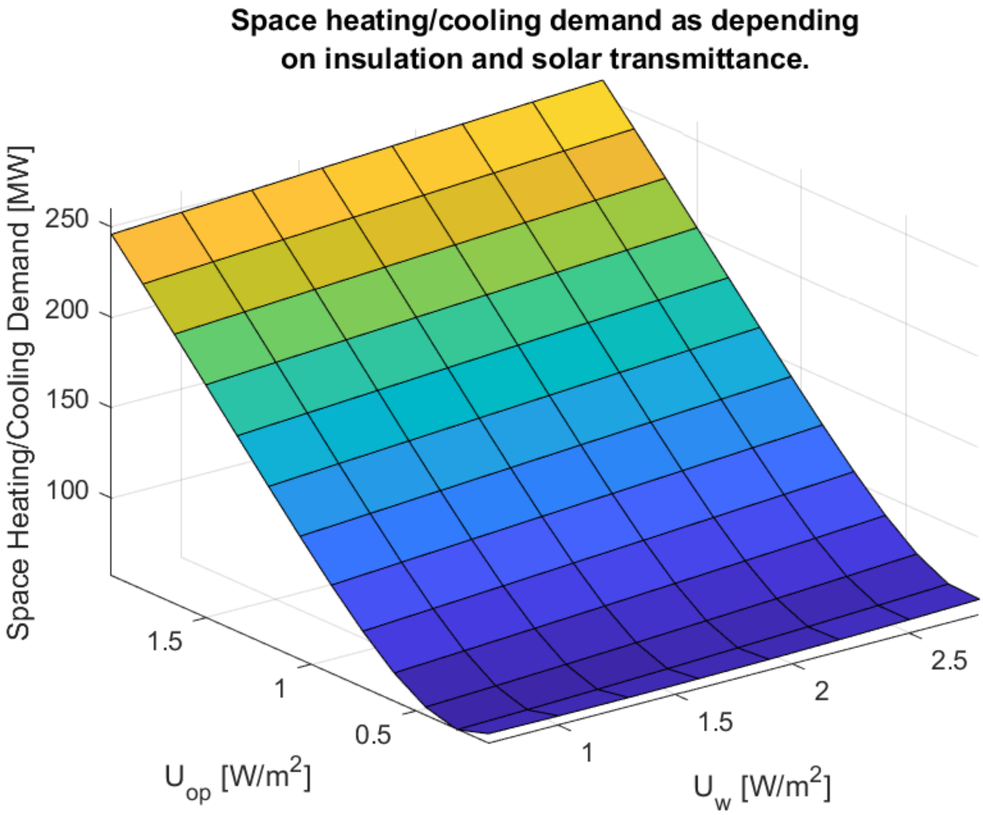


Figure 3.1: Graphical illustration of the space heating function as depending on U_{w} and U_{op}.

3.3 Linearization of the space heating function

Given the nonlinearity of the space heating function it is necessary to linearize the results given by the simulations to achieve functions usable for linear programming. The space heating function was piecewise linearized utilizing multiple linear regression in several partitions of the domain of U_{op}, which was deemed a suitable choice for partitioning via the graphical analysis described in Section 3.2. There were no further research into this via methods such as analyzing integrals of the errors, as the graphical analysis provided enough ground to work on for a proof-of-concept. Figure 3.2 shows a plot of the space heating function along with the resulting piecewise linearization. A generalized equation for the piecewise linearization can be seen in Equation (3.4), where each linear function a_p is defined by the variables U_{w}, U_{op}, q, and $F_{\text{g}}g$, and the piecewise linear and convex function f is defined as the maximum value of the functions a_p, where $p = 1, \ldots, P$. The coefficients $\beta^1, \ldots, \beta^P$ are chosen as described in Section 2.1 using multiple linear regression and solving an ordinary least squares problem. The linearization is then used in the complete model (see Section 3.4) by choosing decision variables (retrofitting components) such that the value of the decision variable parameters U_{w}, U_{op}, q, and $F_{\text{g}}g$ minimize f.

$$
\begin{gathered}
f = \max\{a_1, ..., a_P\}, \\
\text{where} \\
a_1 = \beta_1^0 U_{\text{w}} + \beta_1^1 U_{\text{op}} + \beta_1^2 q + \beta_1^3 F_{\text{g}} g + \beta_1^4, \\
\vdots \\
a_P = \beta_P^0 U_{\text{w}} + \beta_P^1 U_{\text{op}} + \beta_P^2 q + \beta_P^3 F_{\text{g}} g + \beta_P^4.
\end{gathered} \tag{3.4}
$$

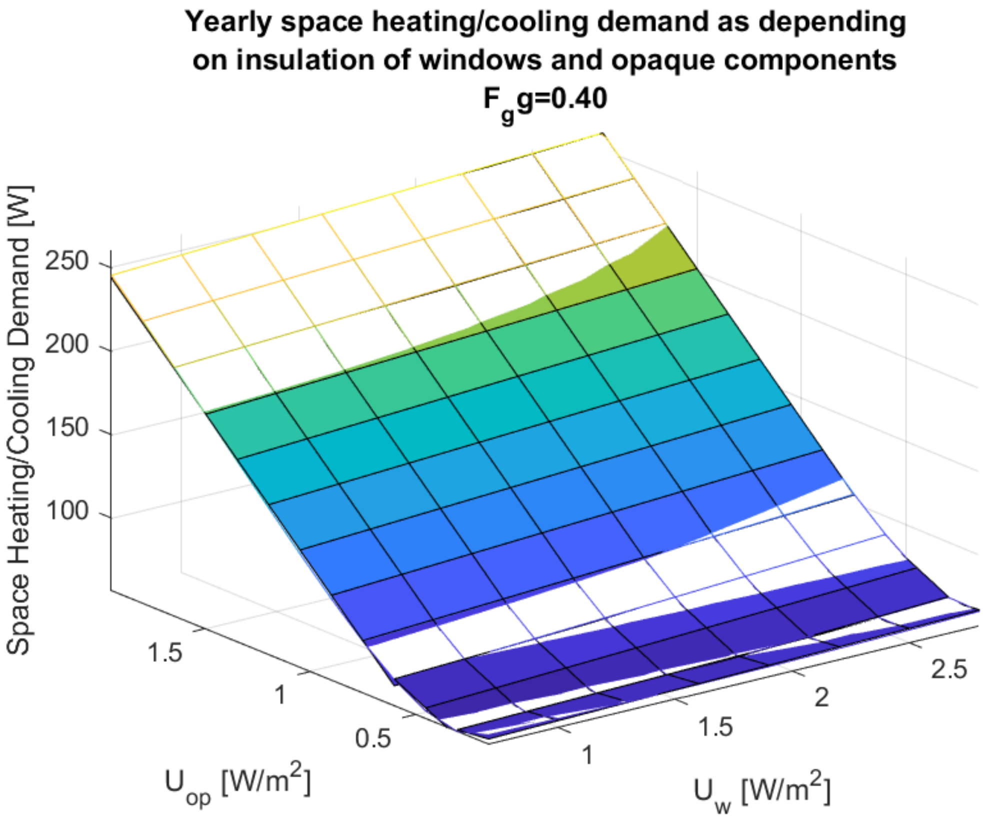


Figure 3.2: Graphical representation of the space heating function along with the resulting piecewise linearized model where $P = 3$. The colored areas show where the linearization is greater than or equal to the original space heating function value. Both $F_{\mathrm{g}}g$ and q are fixed in order to visualize the two factors with most impact, U_{w} and U_{op}.

3.4 Developing a complete mathematical optimization model

This section describes the entire mathematical optimization model with regards to the complete notation utilized, how the decision variables relate to the objective functions, as well as the constraints placed upon the decision variables.

3.4.1 Decision Variables

The decision variables are as described in Section 3.1.1, where each variable matrix corresponds to types of components, their location, and whether or not they are chosen. The naming convention was chosen to reduce clutter with large variable names. Explanations can be seen in Tables 3.1, 3.2, 3.3, and 3.4. The complete notation for all decision variables in consideration is shown in (3.5). Each matrix has its own index with regard to size, as each component has independent numbers of spaces as well as of types. The relationship between variable names and components can be seen in Table 3.1.

$$\mathbf{x} = \begin{bmatrix} x_{11} & \dots & x_{1n} \\ \vdots & \vdots & \vdots \\ x_{N1} & \dots & x_{Nn} \end{bmatrix} \tag{3.5a}$$

$$\mathbf{y} = \begin{bmatrix} y_{11} & \dots & y_{1m} \\ \vdots & \vdots & \vdots \\ y_{M1} & \dots & y_{Mm} \end{bmatrix} \tag{3.5b}$$

$$\mathbf{z} = \begin{bmatrix} z_{11} & \dots & z_{1w} \\ \vdots & \vdots & \vdots \\ z_{W1} & \dots & z_{Ww} \end{bmatrix} \tag{3.5c}$$

$$\mathbf{r} = \begin{bmatrix} r_{11} & \dots & r_{1k} \\ \vdots & \vdots & \vdots \\ r_{K1} & \dots & r_{Kk} \end{bmatrix} \tag{3.5d}$$

$$\mathbf{v} = \begin{bmatrix} v_1 & \dots & v_c \end{bmatrix} \tag{3.5e}$$

Mathematical notation	**Components**
$\mathbf{x}$	Windows
$\mathbf{y}$	Walls
$\mathbf{z}$	Floors
$\mathbf{r}$	Ceilings
$\mathbf{v}$	Ventilation system

Table 3.1: Variables used and their corresponding components.

3.4.2 Constraints

The reasoning behind the constraints follow from Section 3.1.2, whereas this section will focus on going into greater detail of the notation used for the components' constraints. A complete notation for restrictions regarding component placement can be seen in (3.6). Constraints corresponding to the notation can be seen in Table 3.2. The placement constraints follow the same indexing as the decision variable matrices, since every element is individually subject to if it is allowed on space i if it is of type j.

$$\lambda = \begin{bmatrix} \lambda_{11} & \dots & \lambda_{1n} \\ \vdots & \vdots & \vdots \\ \lambda_{N1} & \dots & \lambda_{Nn} \end{bmatrix} \in \mathbb{B}^{N \times n} \tag{3.6a}$$

$$\mu = \begin{bmatrix} \mu_{11} & \dots & \mu_{1m} \\ \vdots & \vdots & \vdots \\ \mu_{M1} & \dots & \mu_{Mm} \end{bmatrix} \in \mathbb{B}^{M \times m} \tag{3.6b}$$

$$\xi = \begin{bmatrix} \xi_{11} & \dots & \xi_{1k} \\ \vdots & \vdots & \vdots \\ \xi_{K1} & \dots & \xi_{Kk} \end{bmatrix} \in \mathbb{B}^{K \times k} \tag{3.6c}$$

$$\gamma = \begin{bmatrix} \gamma_{11} & \dots & \gamma_{1w} \\ \vdots & \vdots & \vdots \\ \gamma_{W1} & \dots & \gamma_{Ww} \end{bmatrix} \in \mathbb{B}^{W \times w} \tag{3.6d}$$

Parameter name	**Component**
λ	Windows.
μ	Walls.
ξ	Floors.
γ	Ceilings.

Table 3.2: Parameters used for restrictions regarding component placement.

The 0-1 nature of the decision variables are restricted to binary values, according to

$$x_{Nn} \in \{0,1\} \qquad \forall N \forall n \tag{3.7a}$$

$$y_{Mm} \in \{0,1\} \qquad \forall M \forall m \tag{3.7b}$$

$$r_{Kk} \in \{0,1\} \qquad \forall K \forall k \tag{3.7c}$$

$$z_{Ww} \in \{0,1\} \qquad \forall W \forall w \tag{3.7d}$$

$$v_c \in \{0,1\} \qquad \forall c \tag{3.7e}$$

The admissibility constraints are then formulated as

$$x_{Nn} \leq \lambda_{Nn} \qquad \forall N \forall n \tag{3.8a}$$
$$y_{Mm} \leq \mu_{Mm} \qquad \forall M \forall m \tag{3.8b}$$
$$r_{Kk} \leq \xi_{Kk} \qquad \forall K \forall k \tag{3.8c}$$
$$z_{Ww} \leq \gamma_{Ww} \qquad \forall W \forall w \tag{3.8d}$$

It is assumed that all ventilation systems are allowed for the building in question. It is possible that not all ventilation systems will be admissible for placement in the case where several buildings are considered, but as it is also possible that each building can be optimized individually in this regard, simplifying the problem formulation to allow for all ventilation systems. Even if this would not be the case, it is a minor alteration to include a similar constraint to ventilation systems as well, and as such it will not be regarded in this model.

In order to impose that only a single component type is placed in each available space, the following constraints are formulated:

$$\sum_{j=1}^{n} x_{Nj} = 1, \qquad \forall N, \tag{3.9a}$$
$$\sum_{j=1}^{m} y_{Mj} = 1, \qquad \forall M, \tag{3.9b}$$
$$\sum_{j=1}^{k} r_{Kj} = 1, \qquad \forall K, \tag{3.9c}$$
$$\sum_{j=1}^{w} z_{Wj} = 1, \qquad \forall W, \tag{3.9d}$$
$$\sum_{j=1}^{c} v_{j} = 1. \tag{3.9e}$$

3.4.3 Objective functions

The model contains several **cost** functions, and while they are all included in this subsection, only the investment cost function is used as a proof of concept for the ILP model in Section 4.4.

The investment cost equation is arguably one of the more straightforward of the objective functions. Each reduction in cost results in a linear reduction of total costs.

The component costs are divided into material, labour, and replacement costs, with the added costs of operation and maintenance for the ventilation system. In the model, several vectors are constructed for the different costs of each component type, with indices as per their associated component, as

$$\mathbf{C}^{\text{Mat,Win}} = \begin{bmatrix} C_1^{\text{Mat,Win}} & \dots & C_n^{\text{Mat,Win}} \end{bmatrix}, \tag{3.10a}$$
$$\mathbf{C}^{\text{Mat,Wal}} = \begin{bmatrix} C_1^{\text{Mat,Wal}} & \dots & C_m^{\text{Mat,Wal}} \end{bmatrix}, \tag{3.10b}$$
$$\mathbf{C}^{\text{Mat,Flo}} = \begin{bmatrix} C_1^{\text{Mat,Flo}} & \dots & C_w^{\text{Mat,Flo}} \end{bmatrix}, \tag{3.10c}$$
$$\mathbf{C}^{\text{Mat,Cei}} = \begin{bmatrix} C_1^{\text{Mat,Cei}} & \dots & C_k^{\text{Mat,Cei}} \end{bmatrix}, \tag{3.10d}$$
$$\mathbf{C}^{\text{Mat,Ven}} = \begin{bmatrix} C_1^{\text{Mat,Ven}} & \dots & C_c^{\text{Mat,Ven}} \end{bmatrix}, \tag{3.10e}$$

$$\mathbf{C}^{\text{Lab,Win}} = \begin{bmatrix} C_1^{\text{Lab,Win}} & \dots & C_n^{\text{Lab,Win}} \end{bmatrix}, \tag{3.11a}$$
$$\mathbf{C}^{\text{Lab,Wal}} = \begin{bmatrix} C_1^{\text{Lab,Wal}} & \dots & C_m^{\text{Lab,Wal}} \end{bmatrix}, \tag{3.11b}$$
$$\mathbf{C}^{\text{Lab,Flo}} = \begin{bmatrix} C_1^{\text{Lab,Flo}} & \dots & C_w^{\text{Lab,Flo}} \end{bmatrix}, \tag{3.11c}$$
$$\mathbf{C}^{\text{Lab,Cei}} = \begin{bmatrix} C_1^{\text{Lab,Cei}} & \dots & C_k^{\text{Lab,Cei}} \end{bmatrix}, \tag{3.11d}$$
$$\mathbf{C}^{\text{Lab,Ven}} = \begin{bmatrix} C_1^{\text{Lab,Ven}} & \dots & C_c^{\text{Lab,Ven}} \end{bmatrix}, \tag{3.11e}$$

$$\mathbf{C}^{\text{Rep,Win}} = \begin{bmatrix} C_1^{\text{Rep,Win}} & \dots & C_n^{\text{Rep,Win}} \end{bmatrix}, \tag{3.12a}$$
$$\mathbf{C}^{\text{Rep,Wal}} = \begin{bmatrix} C_1^{\text{Rep,Wal}} & \dots & C_m^{\text{Rep,Wal}} \end{bmatrix}, \tag{3.12b}$$
$$\mathbf{C}^{\text{Rep,Flo}} = \begin{bmatrix} C_1^{\text{Rep,Flo}} & \dots & C_w^{\text{Rep,Flo}} \end{bmatrix}, \tag{3.12c}$$
$$\mathbf{C}^{\text{Rep,Cei}} = \begin{bmatrix} C_1^{\text{Rep,Cei}} & \dots & C_k^{\text{Rep,Cei}} \end{bmatrix}, \tag{3.12d}$$
$$\mathbf{C}^{\text{Rep,Ven}} = \begin{bmatrix} C_1^{\text{Rep,Ven}} & \dots & C_c^{\text{Rep,Ven}} \end{bmatrix}, \tag{3.12e}$$

$$\mathbf{C}^{\text{Ope,Ven}} = \begin{bmatrix} C_1^{\text{Ope,Ven}} & \dots & C_c^{\text{Ope,Ven}} \end{bmatrix}, \tag{3.13}$$

$$\mathbf{C}^{\text{Mai,Ven}} = \begin{bmatrix} C_1^{\text{Mai,Ven}} & \dots & C_c^{\text{Mai,Ven}} \end{bmatrix}. \tag{3.14}$$

Next the total cost for the selection of components is calculated for each component, using the introduced decision variables:

$$T^{\text{Mat,Win}} = \sum_N \sum_n x_{Nn} \mathbf{C}_n^{\text{Mat,Win}} \tag{3.15a}$$
$$T^{\text{Mat,Wal}} = \sum_M \sum_m y_{Mm} \mathbf{C}_m^{\text{Mat,Wal}} \tag{3.15b}$$
$$T^{\text{Mat,Flo}} = \sum_K \sum_k z_{Kk} \mathbf{C}_k^{\text{Mat,Flo}} \tag{3.15c}$$
$$T^{\text{Mat,Cei}} = \sum_W \sum_w r_{Ww} \mathbf{C}_w^{\text{Mat,Cei}} \tag{3.15d}$$
$$T^{\text{Mat,Ven}} = \sum_c v_c \mathbf{C}_c^{\text{Mat,Ven}} \tag{3.15e}$$

$$T^{\text{Lab,Win}} = \sum_N \sum_n x_{Nn} \mathbf{C}_n^{\text{Lab,Win}} \tag{3.16a}$$
$$T^{\text{Lab,Wal}} = \sum_M \sum_m y_{Mm} \mathbf{C}_m^{\text{Lab,Wal}} \tag{3.16b}$$
$$T^{\text{Lab,Flo}} = \sum_K \sum_k z_{Kk} \mathbf{C}_k^{\text{Lab,Flo}} \tag{3.16c}$$
$$T^{\text{Lab,Cei}} = \sum_W \sum_w r_{Ww} \mathbf{C}_w^{\text{Lab,Cei}} \tag{3.16d}$$
$$T^{\text{Lab,Ven}} = \sum_c v_c \mathbf{C}_c^{\text{Lab,Ven}} \tag{3.16e}$$

$$T^{\text{Rep,Win}} = \sum_N \sum_n x_{Nn} \mathbf{C}_n^{\text{Rep,Win}} \tag{3.17a}$$
$$T^{\text{Rep,Wal}} = \sum_M \sum_m y_{Mm} \mathbf{C}_m^{\text{Rep,Wal}} \tag{3.17b}$$
$$T^{\text{Rep,Flo}} = \sum_K \sum_k z_{Kk} \mathbf{C}_k^{\text{Rep,Flo}} \tag{3.17c}$$
$$T^{\text{Rep,Cei}} = \sum_W \sum_w r_{Ww} \mathbf{C}_w^{\text{Rep,Cei}} \tag{3.17d}$$
$$T^{\text{Rep,Ven}} = \sum_c v_c \mathbf{C}_c^{\text{Rep,Ven}} \tag{3.17e}$$

$$T^{\text{Ope,Ven}} = \sum_c v_c \mathbf{C}_c^{\text{Ope,Ven}} \tag{3.18}$$

$$T^{\text{Mai}} = \sum_c v_c \mathbf{C}_c^{\text{Mai}} \tag{3.19}$$

The resulting costs are then summed up into totals for materials, labour, replacement, maintenance, and operation according to

$$T^{\text{Mat}} = T^{\text{Mat,Win}} + T^{\text{Mat,Wal}} + T^{\text{Mat,Flo}} + T^{\text{Mat,Cei}} + T^{\text{Mat,Ven}} \tag{3.20}$$

$$T^{\text{Lab}} = T^{\text{Lab,Win}} + T^{\text{Lab,Wal}} + T^{\text{Lab,Flo}} + T^{\text{Lab,Cei}} + T^{\text{Lab,Ven}} \tag{3.21}$$

$$T^{\text{Rep}} = T^{\text{Rep,Win}} + T^{\text{Rep,Wal}} + T^{\text{Rep,Flo}} + T^{\text{Rep,Cei}} + T^{\text{Rep,Ven}} \tag{3.22}$$

$$T^{\text{Inv}} = T^{\text{Mat}} + T^{\text{Lab}} + T^{\text{Add}} + T^{\text{Fee}} + T^{\text{Tax}} + T^{\text{Mar}} \tag{3.23}$$

The expression in Equation (3.23) is then admissible for use as an objective function with added constants for additional costs, fees, taxes, and margins. With minor additional information, such as changes in taxes and fees over time, further objective functions could also be defined; see Appendix B. $T^{\text{Ope,Ven}}$ and T^{Mai} are not used in this objective function, but the mathematical models for these are included due to their use in other cost related objective functions; see Appendix B.

The **space heating** objective function is calculated by using the parameters given by a selection of decision variables in the linearization performed in Section 3.3. For every component, except the ventilation system, two parameters are determined by their placement: The placement area, A, as well as the adjustment factor, b. Vectors (3.24)–(3.25) show these parameters in vector form. Note that the indexing used in the vectors correspond to the same indexing as used in the decision variable matrices for each component as indicated by the parameters' superindex. The window components have a unique set of parameters as shown in (3.26)–(3.28). The physical counterpart of the parameters are listed in Tables 3.3 and 3.4.

$$\mathbf{A}^{\text{x}} = \begin{bmatrix} A_1^{\text{x}} & \dots & A_N^{\text{x}} \end{bmatrix}, \tag{3.24a}$$
$$\mathbf{A}^{\text{y}} = \begin{bmatrix} A_1^{\text{y}} & \dots & A_M^{\text{y}} \end{bmatrix}, \tag{3.24b}$$
$$\mathbf{A}^{\text{z}} = \begin{bmatrix} A_1^{\text{z}} & \dots & A_W^{\text{z}} \end{bmatrix}, \tag{3.24c}$$
$$\mathbf{A}^{\text{r}} = \begin{bmatrix} A_1^{\text{r}} & \dots & A_K^{\text{r}} \end{bmatrix}, \tag{3.24d}$$

$$\mathbf{b}^{\text{x}} = \begin{bmatrix} b_1^{\text{x}} & \dots & b_N^{\text{x}} \end{bmatrix}, \tag{3.25a}$$
$$\mathbf{b}^{\text{y}} = \begin{bmatrix} b_1^{\text{y}} & \dots & b_M^{\text{y}} \end{bmatrix}, \tag{3.25b}$$
$$\mathbf{b}^{\text{z}} = \begin{bmatrix} b_1^{\text{z}} & \dots & b_W^{\text{z}} \end{bmatrix}, \tag{3.25c}$$
$$\mathbf{b}^{\text{r}} = \begin{bmatrix} b_1^{\text{r}} & \dots & b_K^{\text{r}} \end{bmatrix}, \tag{3.25d}$$

$$\mathbf{I}^{\text{global}} = \begin{bmatrix} I_1^{\text{global}} & \dots & I_N^{\text{global}} \end{bmatrix}, \tag{3.26}$$

$$\mathbf{F}^{\text{sh}} = \begin{bmatrix} F_1^{\text{sh}} & \dots & F_N^{\text{sh}} \end{bmatrix}, \tag{3.27}$$

$$\mathbf{F}^{\text{sh,ob}} = \begin{bmatrix} F_1^{\text{sh,ob}} & \dots & F_N^{\text{sh,ob}} \end{bmatrix}. \tag{3.28}$$

Parameter	**Description**
$\mathbf{A}^*$	Area of component placement location for component type $*$
$\mathbf{b}^*$	Adjustment factor if temperature outside placement location differs from outside temperature for component type $*$

Table 3.3: Parameters associated with the placement of components chosen by the decision variables and their descriptions.

Parameter	**Description**
$\mathbf{U}^*$	Insulation value for component for component type $*$
$\boldsymbol{\Delta}\mathbf{U}^*$	Reductions in insulation, e.g. due to holes for fastenings, for component type $*$
$\mathbf{q}$	Strength of ventilation system
$\mathbf{I}^{\text{global}}$	Global solar irradiation on window
$\mathbf{F}^{\text{g}}$	Ratio of glazed surface area of total window area
$\mathbf{g}$	Solar transmittance of window component
$\mathbf{F}^{\text{sh}}$	Shading factor imposed by movable shading devices
$\mathbf{F}^{\text{sh,ob}}$	Shading factor imposed by obstacles

Table 3.4: Parameters associated with the components chosen by the decision variables and their descriptions.

Furthermore every component, except the ventilation system, also have the component specific parameters insulation value, U, and adjustment factor, ΔU. The ventilation system has no placement parameters, but does have the component specific parameter ventilation effectiveness, q. Windows have a special set of component specific parameters in their ratio between glazed and total window area, F_{g}, as well as their solar transmittance, g. The parameters in vector form are given by

$$\mathbf{U}^{\text{x}} = \begin{bmatrix} U_1^{\text{x}} & \dots & U_n^{\text{x}} \end{bmatrix}, \tag{3.29a}$$

$$\mathbf{U}^{\text{y}} = \begin{bmatrix} U_1^{\text{y}} & \dots & U_m^{\text{y}} \end{bmatrix}, \tag{3.29b}$$

$$\mathbf{U}^{\text{z}} = \begin{bmatrix} U_1^{\text{z}} & \dots & U_w^{\text{z}} \end{bmatrix}, \tag{3.29c}$$

$$\mathbf{U}^{\text{r}} = \begin{bmatrix} U_1^{\text{r}} & \dots & U_k^{\text{r}} \end{bmatrix}, \tag{3.29d}$$

$$
\begin{aligned}
\mathbf{\Delta U}^{\mathrm{x}} &= \begin{bmatrix}\Delta U_1^{\mathrm{x}} & \dots & \Delta U_n^{\mathrm{x}}\end{bmatrix}, && (3.30a)\\
\mathbf{\Delta U}^{\mathrm{y}} &= \begin{bmatrix}\Delta U_1^{\mathrm{y}} & \dots & \Delta U_m^{\mathrm{y}}\end{bmatrix}, && (3.30b)\\
\mathbf{\Delta U}^{\mathrm{z}} &= \begin{bmatrix}\Delta U_1^{\mathrm{z}} & \dots & \Delta U_w^{\mathrm{z}}\end{bmatrix}, && (3.30c)\\
\mathbf{\Delta U}^{\mathrm{r}} &= \begin{bmatrix}\Delta U_1^{\mathrm{r}} & \dots & \Delta U_k^{\mathrm{r}}\end{bmatrix}, && (3.30d)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{q} &= \begin{bmatrix}q_1 & \dots & q_V\end{bmatrix}, && (3.31)\\
\mathbf{F}^{\mathrm{g}} &= \begin{bmatrix}F_1^{\mathrm{g}} & \dots & F_n^{\mathrm{g}}\end{bmatrix}, && (3.32)\\
\mathbf{g} &= \begin{bmatrix}g_1 & \dots & g_n\end{bmatrix}. && (3.33)
\end{aligned}
$$

In order to facilitate the presentation of the resulting constants from the component parameters two new constants are introduced, one for the component type (see (3.35)) and one for the component placement location (see (3.34)). Note that both (3.35a) and (3.35e) are determined by the chosen window components, but utilized differently in the linearization, and as such must be separated into two constants.

$$
\begin{aligned}
\mathbf{C}^{\mathrm{xp}} &= \begin{bmatrix}A_1^{\mathrm{x}} b_1^{\mathrm{x}} & \dots & A_N^{\mathrm{x}} b_N^{\mathrm{x}}\end{bmatrix}^T && (3.34a)\\
\mathbf{C}^{\mathrm{yp}} &= \begin{bmatrix}A_1^{\mathrm{y}} b_1^{\mathrm{y}} & \dots & A_M^{\mathrm{y}} b_M^{\mathrm{y}}\end{bmatrix}^T && (3.34b)\\
\mathbf{C}^{\mathrm{zp}} &= \begin{bmatrix}A_1^{\mathrm{z}} b_1^{\mathrm{z}} & \dots & A_W^{\mathrm{z}} b_W^{\mathrm{z}}\end{bmatrix}^T && (3.34c)\\
\mathbf{C}^{\mathrm{rp}} &= \begin{bmatrix}A_1^{\mathrm{r}} b_1^{\mathrm{r}} & \dots & A_K^{\mathrm{r}} b_K^{\mathrm{r}}\end{bmatrix}^T && (3.34d)\\
\mathbf{C}^{\mathrm{gxp}} &= \begin{bmatrix}I_1^{\mathrm{global}} A_1^{\mathrm{x}} F_1^{\mathrm{sh}} F_1^{\mathrm{sh,ob}} & \dots & I_N^{\mathrm{global}} A_N^{\mathrm{x}} F_N^{\mathrm{sh}} F_N^{\mathrm{sh,ob}}\end{bmatrix}^T && (3.34e)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{C}^{\mathrm{x}} &= \begin{bmatrix}U_1^{\mathrm{x}} + \Delta U_1^{\mathrm{x}}, & \dots, & U_n^{\mathrm{x}} + \Delta U_n^{\mathrm{x}}\end{bmatrix} && (3.35a)\\
\mathbf{C}^{\mathrm{y}} &= \begin{bmatrix}U_1^{\mathrm{y}} + \Delta U_1^{\mathrm{y}}, & \dots, & U_m^{\mathrm{y}} + \Delta U_m^{\mathrm{y}}\end{bmatrix} && (3.35b)\\
\mathbf{C}^{\mathrm{z}} &= \begin{bmatrix}U_1^{\mathrm{z}} + \Delta U_1^{\mathrm{z}}, & \dots, & U_w^{\mathrm{z}} + \Delta U_w^{\mathrm{z}}\end{bmatrix} && (3.35c)\\
\mathbf{C}^{\mathrm{r}} &= \begin{bmatrix}U_1^{\mathrm{r}} + \Delta U_1^{\mathrm{r}}, & \dots, & U_k^{\mathrm{r}} + \Delta U_k^{\mathrm{r}}\end{bmatrix} && (3.35d)\\
\mathbf{C}^{\mathrm{gx}} &= \begin{bmatrix}F_1^{\mathrm{g}} g_1 & \dots & F_n^{\mathrm{g}} g_n\end{bmatrix} && (3.35e)
\end{aligned}
$$

The constant vectors associated with every component's location and type are multiplied to form matrices that correspond to the thermal transmittance for every component in every position. This is to include as general a model as possible since, for example, a large window space may not only be replaced by a large window, but also by several smaller windows. Whether this is an admissible solution or not is left to the user constructing the admissibility matrices for each component.

The resulting objective function to be minimized will be called ω, and is subject to a constraint imposed by the piecewise linearization of the space heating equation (see Section 3.3). Due to the space heating function being convex over the area of interest, it is possible to always choose the greatest value of the piecewise equations as this will shape a curve tracing under the actual space heating equation. By in turn searching for the minimum of ω, utilizing the piecewise space heating linearization as a constraint, the optimization will be pressed between the linearization and an approximation of the original space heating equation. By increasing the number of linear piecewise linearizations, the edges between the partitions will be sufficiently small to not cause any significant error.

$$\min\ \omega_{\omega\in\mathbb{R},x\in\mathbb{B}^{N\times n},y\in\mathbb{B}^{M\times m},z\in\mathbb{B}^{W\times w},r\in\mathbb{B}^{K\times k},v\in\mathbb{B}^{c}} \tag{3.36a}$$

$$\text{subject to}$$

$$\begin{aligned}\omega \geq \Bigg\{&\beta_l^0\Bigg[\sum_{i=1}^{N}\sum_{j=1}^{n} x_{ij}C_j^{\mathrm{x}}C_i^{\mathrm{xp}}\Bigg] + \beta_l^1\Bigg[\sum_{i=1}^{M}\sum_{j=1}^{m} y_{ij}C_j^{\mathrm{y}}C_i^{\mathrm{yp}} + \sum_{i=1}^{K}\sum_{j=1}^{k} r_{ij}C_j^{\mathrm{r}}C_i^{\mathrm{rp}} + \sum_{i=1}^{A}\sum_{j=1}^{a} z_{ij}C_j^{\mathrm{z}}C_i^{\mathrm{zp}}\Bigg]\\ &+\beta_l^2\Bigg[\sum_{j=1}^{V} v_jC_j\Bigg] + \beta_l^3\Bigg[\sum_{i=1}^{N}\sum_{j=1}^{n} x_{ij}C_j^{\mathrm{gx}}C_i^{\mathrm{gxp}}\Bigg] + \beta_l^4\Bigg\}, \qquad l = 1,\dots,L,\end{aligned} \tag{3.36b}$$

$$x_{Nn} \leq \lambda_{Nn}, \qquad \forall N \forall n, \tag{3.36c}$$

$$y_{Mm} \leq \mu_{Mm}, \qquad \forall M \forall m, \tag{3.36d}$$

$$r_{Kk} \leq \xi_{Kk}, \qquad \forall K \forall k, \tag{3.36e}$$

$$z_{Ww} \leq \gamma_{Ww}, \qquad \forall W \forall w, \tag{3.36f}$$

$$\sum_{j=1}^{n} x_{Nj} = 1, \qquad \forall N, \tag{3.36g}$$

$$\sum_{j=1}^{m} y_{Mj} = 1, \qquad \forall M, \tag{3.36h}$$

$$\sum_{j=1}^{k} r_{Kj} = 1, \qquad \forall K, \tag{3.36i}$$

$$\sum_{j=1}^{w} z_{Wj} = 1, \qquad \forall W, \tag{3.36j}$$

$$\sum_{j=1}^{c} v_j = 1, \tag{3.36k}$$

$$T^{\mathrm{Inv}} \leq \epsilon. \tag{3.36l}$$

The constraint modelled by the inequality (3.36b) is shown as depending on the decision variables who in turn choose the U^{w}, U^{op}, q, and $F^{\mathrm{g}}g$ values. The number L of partitions, where $l = \{1, ..., L\}$, are typically in the range of 3 to 5 for this application, each partition corresponding to some subinterval of U^{op}, U_l^{op}. Where β_l^0, β_l^1, β_l^2 and β_l^3 are vectors of coefficients from the linearization, and β_l^4 is a vector of constant offsets that are set during linearization. Further constraints are added in (3.36c)-(3.36k) limiting the choice of components as shown in Section 3.4.2. Finally the ϵ-constraint (3.36l) (see Section 2.2) is added to the complete model, limiting the investment cost.

4

Tests and results

In this chapter, several different results will be presented without much regard to analysis or discussion, but rather to display the results along with explanations where needed. Further thoughts regarding what the results mean will follow in the discussion and conclusions section. First the data set will be explained in greater detail, next the results given by the linearization and optimization will be presented. It should be noted that the results from the simulation are not presented as it is just an intermediate step toward the linearization. The main result in this chapter is found in Section 4.4, where the results from testing the ILP model against NSGA-II are presented.

4.1 Data set

The data set used was from an example building previously tested by the NSGA-II algorithm. A summary of the components available in the data set can be seen in Appendix E. This example building has four walls, one unheated bottom floor, one unheated top ceiling and 126 windows. Further, the building is oblong and rectangular, as sketched in Figure 4.1, which is important to know when calculating the impact of the sun. In the data set there also exist sample data for the amount of heating and cooling, respectively, for every hour throughout one year.

Figure 4.1: A rough sketch of the oblong building showing the compass direction for each side of the building.

The admissibility data for each component is written, individually for each renovation, in the form of a table where each component which is unavailable at a certain space is marked by a ’0’.

Data regarding the weather such as temperature, azimuth, diffuse horizontal radiation, global horizontal radiation, and height of the sun has also been sampled throughout a year and are used when simulating the total space heating demand.

4.2 Linearization

The most important result regarding the linearization, apart from the coefficients, is the magnitude of the maximum error over each partition of the piecewise linearization. It was found that the largest resulting error was around 6%. The magnitude of the error depends on the number of piecewise partitions used in the linearization, and can be further reduced by increasing the number of partitions P, shown in Section 3.3. For the proof of concept this margin of error was accepted and kept in mind when comparing the solutions computed by CPLEX and NSGA-II, respectively.

Partition(P)	Average error(%)	Maximum error(%)
1	1.52	4.28
2	1.97	6.16
3	0.99	5.71

Table 4.1: This table shows the average and maximum errors found for each partition in the piecewise linearization during the testing of the model.

An important note about the linearization process is that the simulation step adds a significant amount of time to precompute the problem data, but also that the entire process does not scale with the amount of variables (more windows, walls, buildings, etc). The simulation which it is based upon does however scale with the level of resolution chosen, and has a linear time complexity of $O(n^5)$, where n is equal to the largest number of simulated results for a variable, imposed by looping over a number of points in a feasible interval of the four variables being simulated as well as looping over every hour for the measured year.

4.3 Optimization solutions

The optimization model was implemented in AMPL (see [27]), which is a modeling language for optimization. It also includes the optimization solver CPLEX, which was briefly described in Section 2.3. Once the model was linearized it was implemented in AMPL. As a first step, the binary requirements on the variables in the model in Chapter 3 was relaxed and the resulting LP-relaxation was solved using the Simplex method (within the solver CPLEX). As expected, the solutions were highly fractional and thus focus was shifted to using CPLEX' ILP solver. For the data set provided, CPLEX produced optimal integer solutions, which are compared with the solution provided by NSGA-II in Section 4.4.

When comparing solutions to different problem instances given by AMPL, there are a lot of differences even when the costs of the solutions are relatively close to each other. One example of this can be seen when comparing the solutions given by a maximum investment cost of 5.1 Mkr with the solution given by a maximum of 6.1 Mkr as shown in Tables 4.2 and 4.3. The two solutions have different types of components in all component spaces, except two original windows and a heat recovery system. Differences in the choice of components seem to be consistent between most different solutions at different investment costs, except from the heat recovery system which is immediately included as soon as there is enough money to afford it. One other thing that is consistent is that the solutions for lower investment cost usually includes insulation to the floor and to the ceiling. This is due to the original floor and ceiling being of poor quality with respect to insulation, and that they are covering such a large area relative to the total area of the building.

If the constraint on investment cost is removed, the optimal solution is found at around 8.5 MKr. This solution is presented in Table 4.4. It should be noted that the reason that different walls are chosen are due to the fact that sandwich walls are not admissible on Wall 2 and 4. Comparing the optimal solution to the solution which applies no retrofitting, a nearly ten times lower space heating energy demand is achieved. It was also found that when no limits were placed on the investment cost, both NSGA-II and CPLEX found the same component setup to be the optimal solution.

Component space	**Option**
Ceiling	150 mm external insulation above the roof slab
Floor	100 mm insulation on the outside (unheated heated side) of the floor slab
Wall 1	Sandwich insulation 80 mm
Wall 2	Original
Wall 3	Sandwich insulation 120 mm
Wall 4	Original (no replacement)
Window 1	Original (no replacement)
Window 2	Original (no replacement)
Window 3	Original (no replacement)
Window 4	Original (no replacement)
Ventilation	Heat recovery

Table 4.2: Components chosen by CPLEX for a maximum investment cost of 5.1 Mkr.

Component space	Option
Ceiling	200 mm external insulation above the roof slab
Floor	Insulation on the outside (unheated heated side) of the floor slab - 300 mm on ground
Wall 1	200 mm sandwich add insulation
Wall 2	170 mm brickwall add insulation
Wall 3	200 mm sandwich add insulation
Wall 4	170 mm brickwall add insulation
Window 1	Original
Window 2	Window U-value 0.8 - Wood/aluminium
Window 3	Original
Window 4	Window U-value 0.8 - Wood/aluminium
Ventilation	Heat recovery

Table 4.3: Components chosen by CPLEX for a maximum investment cost of 6.1 Mkr.

Component space	Option
Ceiling	2*130 mm TRP external insulation above the roof slab
Floor	300 mm on ground insulation on the outside(unheated heated side) of the floor slab
Wall 1	200 mm sandwich add insulation
Wall 2	170 mm brickwall add insulation
Wall 3	200 mm sandwich add insulation
Wall 4	170 mm brickwall add insulation
Window 1	Window U-Value 0.8 - Wood/aluminium
Window 2	Window U-Value 0.8 - Wood/aluminium
Window 3	Window U-Value 0.8 - Wood/aluminium
Window 4	Window U-Value 0.8 - Wood/aluminium
Ventilation	Heat recovery

Table 4.4: Components chosen by both CPLEX and NSGA-II without any restriction on the investment cost.

4.4 Computing Pareto-optimal solutions

Figure 4.2 shows the the plots of the different solutions found by CPLEX and NSGA-II. In the figure it can be seen that the algorithm found that the most dramatic reductions in energy demand come from the initial increase in investment cost, whilst the difference for higher investment cost has far less impact on energy consumption. Examples of how the component sets for these solutions may look can be seen in Tables 4.2–4.4. By checking the factor between the highest energy consumption (when no component is replaced) and the lowest, it can be found that the greatest possible relative reduction in energy demand for this test case building is $\frac{326.11-32.04}{326.11}\frac{[\mathrm{MW}]}{[\mathrm{MW}]} \approx 90\%$.

Figure 4.3 shows the difference in the number of generations in NSGA-II. As the algorithm progresses more solutions are discovered with each iteration, slowly converging toward the Pareto optimal front. It can be seen that there are still better solutions found after 100 000 generations as compared to 10 000 generations, but that the improvement is minor.

It is important to realize that the runtime difference between increasing generations is vast, taking around 20 seconds for 1 000 generations, 3 minutes for 10 000, and 36 minutes for 100 000 generations on an average user laptop. This compares to the ILP solver CPLEX, which runs in about 4 seconds on the same hardware. The reason for this fast runtime is the simple mathematical structure in the ILP-model (3.36). It should also be noted that NSGA-II does not manage to produce satisfactory solutions in 1 000 generations, as shown in Figure 4.2. The points where NSGA-II seems to have found solutions which dominate the CPLEX solutions are just artifacts from the errors in the linearization process, as no difference in the choice of components occurs.

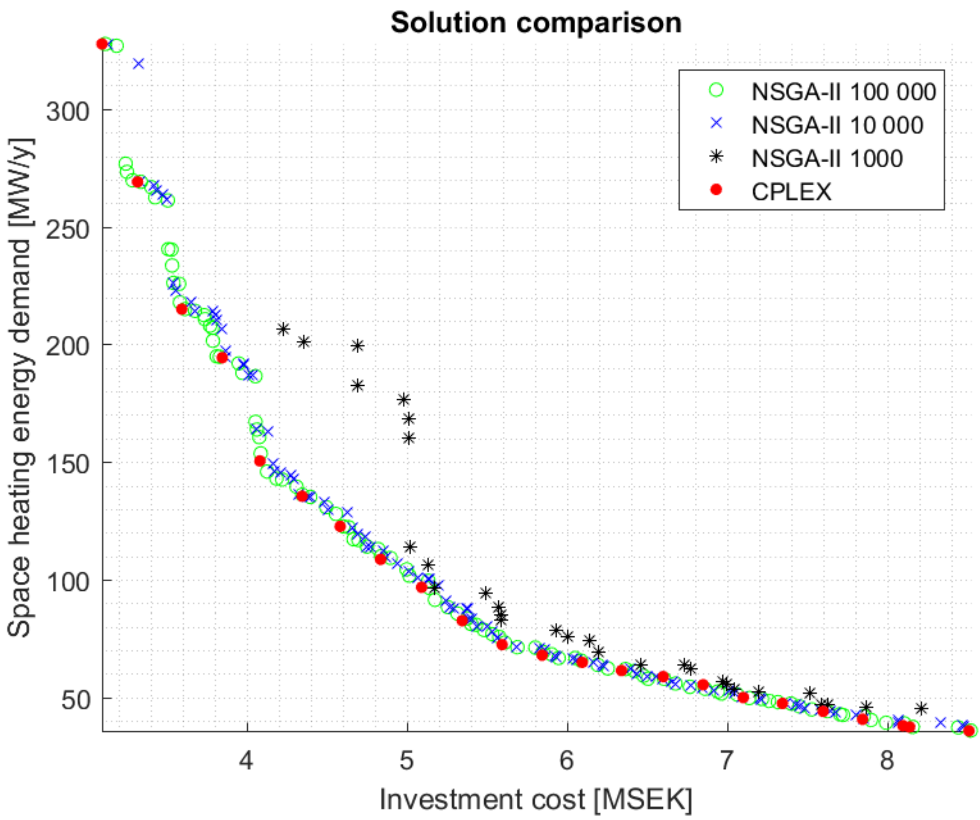


Figure 4.2: Plots of the different solutions produced by the CPLEX algorithm and NSGA-II for different numbers of generations.

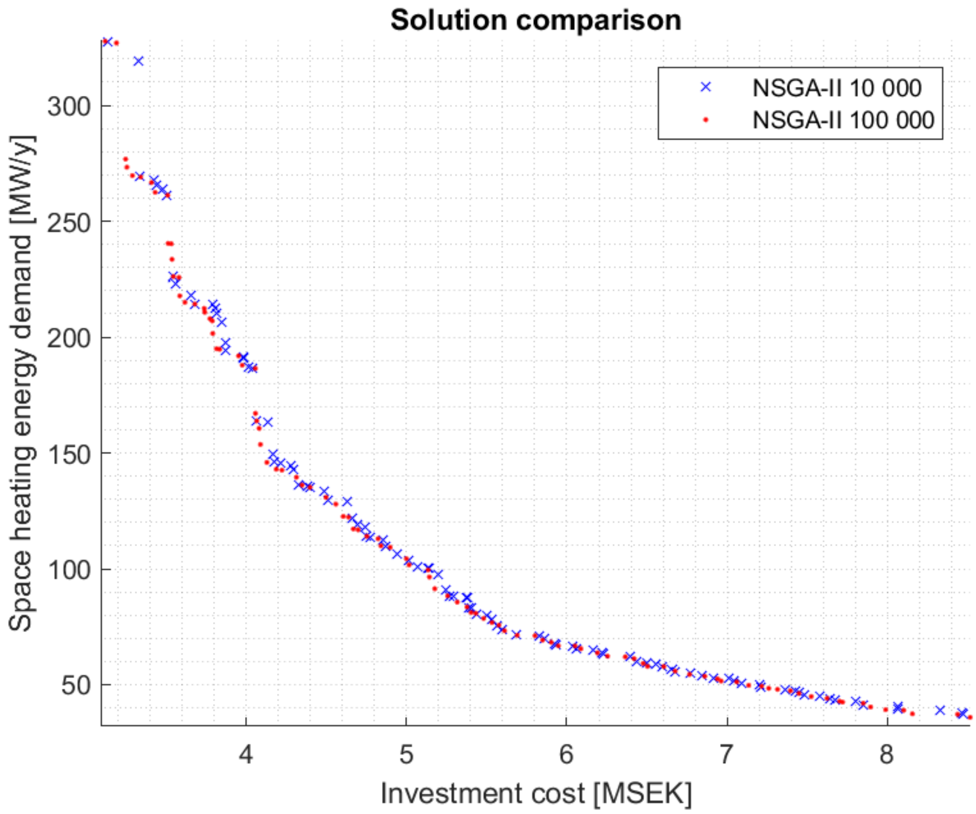


Figure 4.3: Plot showing the solutions found by NSGA-II. The figure shows the difference between solutions found by allowing the algorithm to run for 10 000 and 100 000 generations respectively.

5
Discussion and Conclusion

It has been found that the equations utilized in planning retrofitting projects are, in many cases, mathematically complex and hard to analyze algebraically. In the case of the space heating demand function, however, it was also found that, despite the apparent complexity, the equations yield a convex function. The linearization provided acceptable maximum errors, and could be further improved by increasing the number of linear pieces. It should also be noted that despite the linearization errors the same components were chosen in the optimization with no limit on investment cost, which indicate that at least for this example the errors could not be found to cause any severe consequences. Since any further retrofitting problem instance can be reduced to a simple scaling of the tested building this shows promise that even when applied to other buildings the linearization errors could possibly be handled in the same way.

When tested it was found that the ILP solver indeed found Pareto optimal solutions that dominated many of the solutions found by the previous algorithm, the stochastic NSGA-II. This would be reasonable given that the nature of NSGA-II is to choose the best solution over the set of solutions that it has found, iterating over those, and searching their neighborhood for even better solutions. This means that if NSGA-II has not found Pareto optimal solutions, it will simply choose "the best it knows". This contrasts to the ILP model which, in this proof of concept, utilizes CPLEX to search for optimal solutions (to the corresponding problem where the space heating function is linearized) in a controlled and consistent manner. Most notably, this means that not only is the runtime far lower than when using NSGA-II, it is also unnecessary to run this solver more than once for the same problem instance, unlike NSGA-II which may produce different solutions for different runs depending on initial guesses, selection, and mutation as described in Section 2.4. While the solutions may be a bit skewed due to the errors, the comparison against 1 000 generations of NSGA-II, seen in Figure 4.2 is the most telling for how this may scale in a larger neighborhood of buildings where NSGA-II will require a far greater amount of generations in order to find a near-Pareto optimal front.

As mentioned in Section 4.4 the runtime of the ILP model is vastly lower than that of NSGA-II. The value of this does not lie solely in the reduced runtime, but in the increased capabilities for the solver in solving a larger multiobjective problem with more decision variables and more component types within a reasonable time frame, for example by adding objective functions. It also means that the solution scales better to larger building blocks, resulting in possibilities that the

ILP model and solution can be used on an office laptop without need for greater computational power, even for large neighborhoods of multistory buildings. As previously mentioned, NSGA-II needs to iterate over a number of generations in order to find satisfactory results, increasing as the problem instances grows larger. This means that the 36 minute runtime for 100 000 generations is an underestimation when applied to large neighborhoods, whereas the ILP model shows promise in not having as drastic increase in runtime when dealing with larger scales of the retrofitting problem.

5.1 Future work

This report focuses mainly on space heating demand and investment cost (with further cost objective functions also supported) as these are the objectives with most use as of now. There are, however, a lot more objective functions to work on, as shown in Appendix D. However, the same type of decision variables are used in all objectives with the addition of some new components. This means that the work to include more objective functions should not be as time consuming as implementing the space heating demand and the cost. To then use a similar method on a third simultaneous objective will likely not increase the running time considerably as it can be viewed as another constraint in the current model (see Section 2.2). However, the amount of iterations will need to be increased in order to find many enough points on the Pareto front.

The immediate next step after implementation of the missing objective functions would be to attempt implementation of the complete method (simulation, linearization, and optimization) in C#/.NET in order to merge the results from this research into DreeamTool, mentioned in Section 1.1.

Another development which could further decrease the energy use would be to allow retrofitting different components at different times over the consideration period (which is of the magnitude 20 to 40 years). For this to work one would need to include the average lifetime of a component and the various costs which are included in maintaining an older component as opposed to the labour of retrofitting a new component. This means that a lot more options, with regard to both choice of replacements and timing of these actions, for retrofitting the components must be included in the mathematical model, which then will grow larger.

Further work also includes a method to dynamically determine how many linear pieces the model of the space heating demand function should use for the evaluated building as determined by the maximum allowed error in the linearization. This could be done by iterating over the maximum errors of all linear pieces, and while a partition yields a maximum error above the accepted threshold, that partition is further partitioned until the maximum error is satisfactorily small. This type of solution would yield high amounts of partitions in sensitive areas, whilst maintaining a low amount in areas where it is not needed.

Bibliography

[1] "DREEAM: Demonstrating an integrated Renovation approach for Energy Efficiency At the Multi-building scale." `http://dreeam.eu/about-dreeam/`. Accessed: 2017-01-09.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[3] "CEE to coordinate H2020-project." `http://www.chalmers.se/en/departments/ace/news/Pages/CEE-to-coordinate-H2020-project.aspx`. Accessed: 2017-01-09.

[4] "Consid AB." `http://www.consid.se`. Accessed: 2017-01-19.

[5] "Chalmers i tätt samarbete med göteborgskt it-bolag revolutionerar byggbranschen." `http://www.consid.se/newsroom/pressreleaser/chalmers-i-tatt-samarbete-med-goteborgskt-it-bolag-revolutionerar-byggbranschen/`. Accessed: 2017-01-10.

[6] J. Lundgren, M. Rönnqvist, and P. Värbrand, *Optimization*. Lund: Studentlitteratur, 2010.

[7] P. Lindroth, *Product Configuration from a Mathematical Optimization Perspective*. Doktorsavhandlingar vid Chalmers tekniska högskola. Ny serie, no: 3215, Institutionen för matematiska vetenskaper, matematik, Chalmers tekniska högskola, 2011.

[8] P. Lindroth, M. Patriksson, and A.-B. Strömberg, "Approximating the Pareto optimal set using a reduced set of objective functions," *European Journal of Operational Research*, vol. 207, no. 3, pp. 1519–1534, 2010.

[9] Y. Ostermeyer, H. Wallbaum, and F. Reuter, "Multidimensional Pareto optimization as an approach for site-specific building refurbishment solutions applicable for life cycle sustainability assessment," *The International Journal of Life Cycle Assessment*, vol. 18, no. 9, pp. 1762–1779, 2013.

[10] G. Grégoire, "Multiple linear regression," *European Astronomical Society Publications Series*, vol. 66, pp. 45–72, 2014.

[11] F. Hayashi, *Econometrics*. Princeton University Press, 2000.

[12] K. Deb, "Multi-objective optimization," in *Search methodologies*, pp. 403–449, Springer, 2014.

[13] G. Mavrotas, "Effective implementation of the ε-constraint method in multiobjective mathematical programming problems," *Applied mathematics and computation*, vol. 213, no. 2, pp. 455–465, 2009.

[14] G. L. Nemhauser and L. A. Wolsey, "Integer programming and combinatorial optimization," *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS*

Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin, vol. 20, pp. 8–12, 1988.

[15] D. Van Dantzig, "On the principles of intuitionistic and affirmative mathematics," *Indagationes Mathematicae*, vol. 9, pp. 429–440, 1947.

[16] "Convex sets and simplex method." `http://www.trentu.ca/academic/math/bz/341-l05.pdf`.

[17] J. Nocedal and S. J. Wright, *Numerical Optimization.* New York: Springer-Verlag, 1999.

[18] A. Forsgren, P. E. Gill, and M. H. Wright, "Interior methods for nonlinear optimization," *SIAM review*, vol. 44, no. 4, pp. 525–597, 2002.

[19] V. Klee and G. J. Minty, "How good is the simplex algorithm," tech. rep., Seattle: Washington university, department of mathematics, 1970.

[20] K. C. Border, *Fixed Point Theorems with Applications to Economics and Game Theory.* Cambridge: Cambridge University Press, 1985.

[21] "Cplex optimizer." `https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/`. Accessed: 2017-03-20.

[22] M. Wahde, *Biologically Inspired Optimization Methods: An Introduction.* U.K., Southampton: WIT Press, 2008.

[23] Z. R. Jifeng Xuan, He Jiang, *Pseudo Code of Genetic Algorithm and Multi-Start Strategy Based Simulated Annealing Algorithm for Large Scale Next Release Problem.* School of Software, Dalian University of Technology, China.

[24] C. Nägeli, "Dreeam tool documentation." Classified document.

[25] I. K. Claudio Nägeli, York Ostermeyer, "Concept of the dreeam approach for the multiple building energy renovation." Classified document.

[26] Discussions with Claudio Nägeli.

[27] R. Fourer, D. M. Gay, and B. Kernighan, *Ampl*, vol. 117. Boyd & Fraser Danvers, MA, 1993.

[28] E. ISO, "13790: 2008 energy performance of buildings–calculation of energy use for space heating and cooling," *International Standard Organisation*, 2008.

A
Space heating demand equations

These functions are listed as referenced from the DREEAM Project documentation [24]. The variable descriptions and relationships can be seen in the ISO 13790:2008 standard [28].

$$H_{tr,is} = h_{is} A_f A_{at} \tag{A.1}$$

$$H_{tr,w} = \sum_c A_c b_c (U_c + \Delta U_c) \tag{A.2}$$

$$H_{tr,op} = \sum_c A_c b_c (U_c + \Delta U_c) \tag{A.3}$$

$$H_{tr,ms} = h_{ms} A_m \tag{A.4}$$

$$H_{tr,em} = \frac{1}{\frac{1}{H_{tr,op}} - \frac{1}{H_{tr,ms}}} \tag{A.5}$$

$$H_{ve,inf} = \rho_a c_a V_a ACH_{inf} \tag{A.6}$$

$$H_{ve,act} = \rho_a c_a q_{ve,act} A_f f_{ve} b_{ve} \tag{A.7}$$

$$H_{ve} = H_{ve,inf} + H_{ve,act} \tag{A.8}$$

$$I_{direct,c} = I_{direct,h} \frac{1}{\sin(\alpha)} (\cos(\alpha)\sin(\beta)\cos(\Psi + \Theta + F_{cor}) + \sin(\alpha)\cos(\beta)) \tag{A.9}$$

$$I_{diffuse,c} = I_{diffuse,h} \frac{1}{2} (1 + \cos(\beta)) \tag{A.10}$$

$$I_{global,c} = I_{direct,c} + I_{diffuse,c} \tag{A.11}$$

$$\phi_{sol,w} = \sum_c I_{global,c} A_c g_c F_{g,c} F_{sh,c} F_{sh,ob,c} \tag{A.12}$$

$$\phi_{int,Oc} = n_{Oc} \phi_{Oc} f_{Oc} \tag{A.13}$$

$$\phi_{int,App} = A_f \phi_{App} f_{App} \tag{A.14}$$

$$\phi_{int} = \phi_{int,App} + \phi_{int,Oc} \tag{A.15}$$

$$\phi_{ia} = \frac{1}{2}\phi_{int} \tag{A.16}$$

$$\phi_m = \frac{A_m}{A_{at}A_f}(\frac{1}{2}\phi_{int} + \phi_{sol,w}) \tag{A.17}$$

$$\phi_{st} = (1 - \frac{A_m}{A_{at}A_f} - \frac{H_{tr,w}}{9.1A_{at}A_f})(\frac{1}{2}\phi_{int} + \phi_{sol,w}) \tag{A.18}$$

$$H_{tr,1} = \frac{1}{\frac{1}{H_{ve}} + \frac{1}{H_{tr,is}}} \tag{A.19}$$

$$H_{tr,2} = H_{tr,1} + H_{tr,w} \tag{A.20}$$

$$H_{tr,3} = \frac{1}{\frac{1}{H_{tr,2}} + \frac{1}{H_{tr,ms}}} \tag{A.21}$$

$$\phi_{mtot,HC} = \phi_m + H_{tr,ms}\Theta_e + \frac{H_{tr,3}}{H_{tr,2}}\left(\phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left[\frac{\phi_{ia} + \phi_{HC,nd}}{H_{ve}} + \Theta_{sup}\right]\right) \tag{A.22}$$

$$\phi_{mtot,H} = \phi_m + H_{tr,ms}\Theta_e + \frac{H_{tr,3}}{H_{tr,2}}\left(\phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left[\frac{\phi_{ia} + \phi_{H,nd}}{H_{ve}} + \Theta_{sup}\right]\right) \tag{A.23}$$

$$\phi_{mtot,C} = \phi_m + H_{tr,ms}\Theta_e + \frac{H_{tr,3}}{H_{tr,2}}\left(\phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left[\frac{\phi_{ia} + \phi_{C,nd}}{H_{ve}} + \Theta_{sup}\right]\right) \tag{A.24}$$

$$\Theta_{m,t,HC} = \frac{\Theta_{m,t-1,HC}\left(\frac{C_m}{3600} - \frac{1}{2}[H_{tr,3} + H_{tr,em}]\right) + \phi_{mtot,HC}}{\frac{C_m}{3600} + \frac{1}{2}(H_{tr,3} + H_{tr,em})} \tag{A.25}$$

$$\Theta_{m,t,H} = \frac{\Theta_{m,t-1,H}\left(\frac{C_m}{3600} - \frac{1}{2}[H_{tr,3} + H_{tr,em}]\right) + \phi_{mtot,H}}{\frac{C_m}{3600} + \frac{1}{2}(H_{tr,3} + H_{tr,em})} \tag{A.26}$$

$$\Theta_{m,t,C} = \frac{\Theta_{m,t-1,C}\left(\dfrac{C_m}{3600} - \dfrac{1}{2}[H_{tr,3} + H_{tr,em}]\right) + \phi_{mtot,C}}{\dfrac{C_m}{3600} + \dfrac{1}{2}(H_{tr,3} + H_{tr,em})} \tag{A.27}$$

$$\Theta_{m,HC} = \frac{\Theta_{m,t,HC} + \Theta_{m,t-1,HC}}{2} \tag{A.28}$$

$$\Theta_{m,H} = \frac{\Theta_{m,t,H} + \Theta_{m,t-1,H}}{2} \tag{A.29}$$

$$\Theta_{m,C} = \frac{\Theta_{m,t,C} + \Theta_{m,t-1,C}}{2} \tag{A.30}$$

$$\Theta_{s,HC} = \frac{H_{tr,ms}\Theta_{m,HC} + \phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left(\Theta_{sup} + \dfrac{\phi_{ia} + \phi_{HC,nd}}{H_{ve}}\right)}{H_{tr,ms} + H_{tr,w} + H_{tr,1}} \tag{A.31}$$

$$\Theta_{s,H} = \frac{H_{tr,ms}\Theta_{m,H} + \phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left(\Theta_{sup} + \dfrac{\phi_{ia} + \phi_{H,nd}}{H_{ve}}\right)}{H_{tr,ms} + H_{tr,w} + H_{tr,1}} \tag{A.32}$$

$$\Theta_{s,C} = \frac{H_{tr,ms}\Theta_{m,C} + \phi_{st} + H_{tr,w}\Theta_e + H_{tr,1}\left(\Theta_{sup} + \dfrac{\phi_{ia} + \phi_{C,nd}}{H_{ve}}\right)}{H_{tr,ms} + H_{tr,w} + H_{tr,1}} \tag{A.33}$$

$$\Theta_{air,HC} = \frac{H_{tr,is}\Theta_{s,HC} + H_{ve}\Theta_{sup} + \phi_{ia} + \phi_{HC,nd}}{H_{tr,is} + H_{ve}} \tag{A.34}$$

$$\Theta_{air,H} = \frac{H_{tr,is}\Theta_{s,H} + H_{ve}\Theta_{sup} + \phi_{ia} + \phi_{H,nd}}{H_{tr,is} + H_{ve}} \tag{A.35}$$

$$\Theta_{air,C} = \frac{H_{tr,is}\Theta_{s,C} + H_{ve}\Theta_{sup} + \phi_{ia} + \phi_{C,nd}}{H_{tr,is} + H_{ve}} \tag{A.36}$$

$$\phi_H = \begin{cases} \phi_{H,nd}\dfrac{\Theta_{air,H,set} - \Theta_{air,HC}}{\Theta_{air,H} - \Theta_{air,HC}} & \text{, for } \Theta_{air,HC} \leq \Theta_{air,H,set} \\ 0 & \text{, for } \Theta_{air,HC} > \Theta_{air,H,set} \end{cases} \tag{A.37}$$

$$\phi_C = \begin{cases} \phi_{C,nd}\dfrac{\Theta_{air,C,set} - \Theta_{air,HC}}{\Theta_{air,C} - \Theta_{air,HC}} & \text{, for } \Theta_{air,HC} \geq \Theta_{air,C,set} \\ 0 & \text{, for } \Theta_{air,HC} < \Theta_{air,C,set} \end{cases} \tag{A.38}$$

B
Additional cost functions

These functions are listed as referenced from the DREEAM Project documentation [25].

$$C_{Running} = C_{Maintenance} + C_{Operational} + C_{Energy} \tag{B.1}$$

$$LCC = C_{Investment} + \sum_{t=1}^{T} \left(\frac{C_{Running} - C_{Replacement}}{(1+r)^t} \right) \tag{B.2}$$

$$NPV = \sum_{t=1}^{T} \left(\frac{C_{Earnings} - C_{Running} - C_{Replacement}}{(1+r)^t} \right) - C_{Investment} \tag{B.3}$$

$$NPV = 0 = \sum_{t=1}^{T} \left(\frac{C_{Earnings} - C_{Running} - C_{Replacement}}{(1+IRR)^t} \right) - C_{Investment} \tag{B.4}$$

$$ROI = \frac{(C_{Earnings,SQ} - C_{Running,SQ}) - (C_{Earnings,RC} - C_{Running,RC})}{C_{Investment}} \tag{B.5}$$

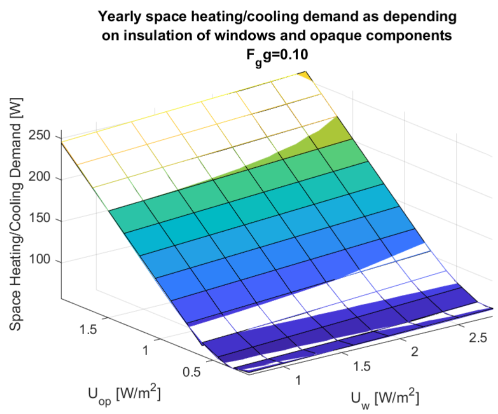
$$C_{Savings} = \frac{C_{Running,SQ} - C_{Running,RC}}{C_{Running,SQ}} \tag{B.6}$$

C

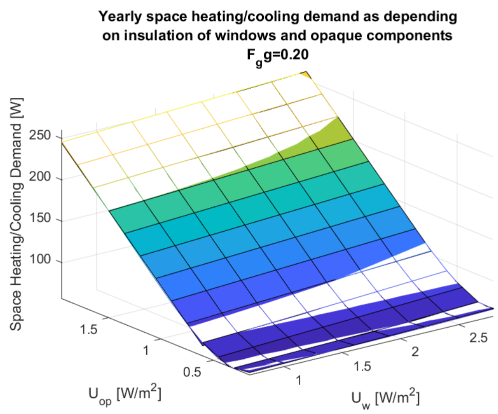
Supplementary figures

C.1 Piecewise linearizations of the space heating demand function

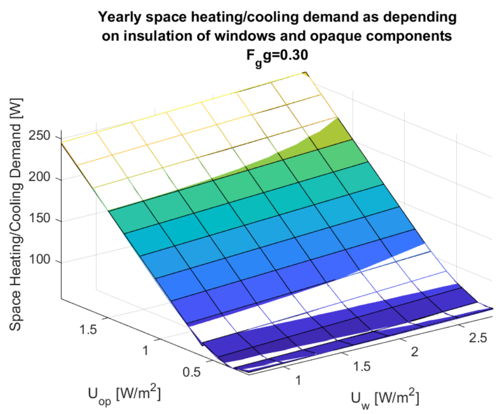
These figures show how the equation changes depending on $F_g g$. A linear relationship can be hinted by the figures.



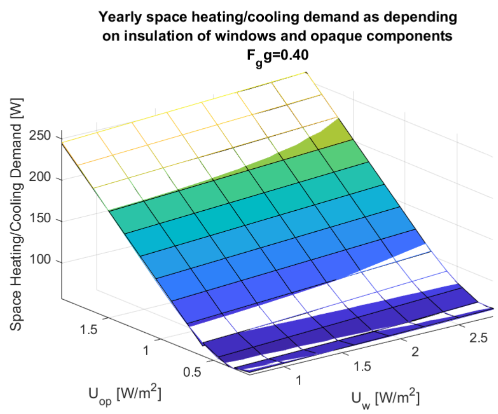
(a) $F_\mathrm{g}g = 0.072$, $q = 0.2$



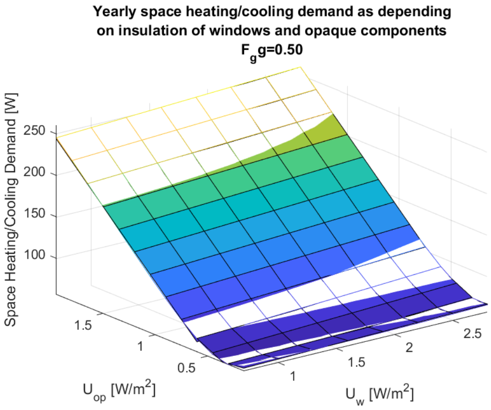
(b) $F_\mathrm{g}g = 0.072$, $q = 0.4$



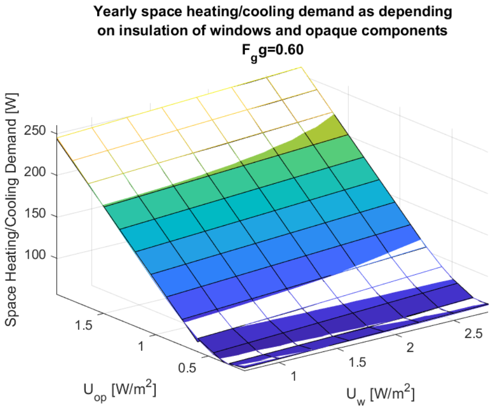
(c) $F_\mathrm{g}g = 0.072$, $q = 0.6$



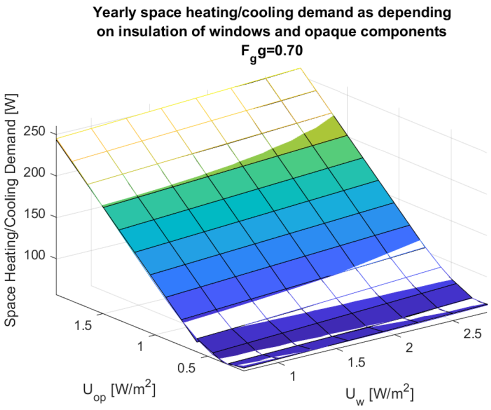
(d) $F_\mathrm{g}g = 0.072$, $q = 0.8$



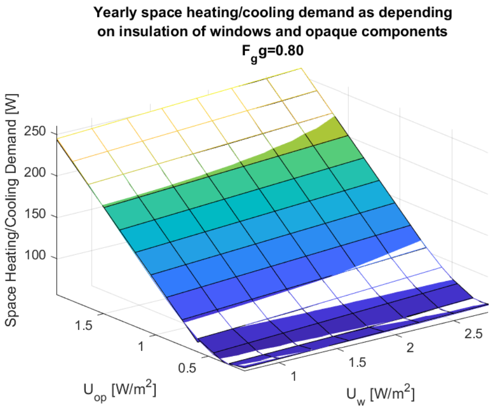
(a) $F_g g = 0.072$, $q = 1.0$



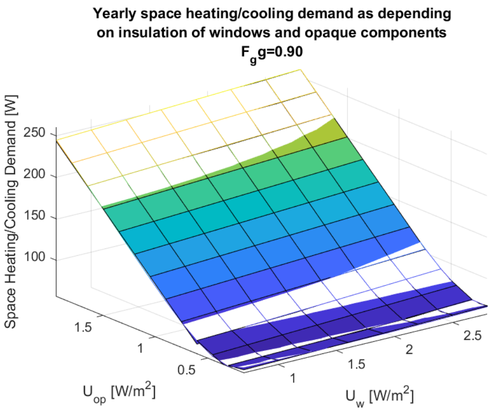
(b) $F_g g = 0.396$, $q = 0.2$



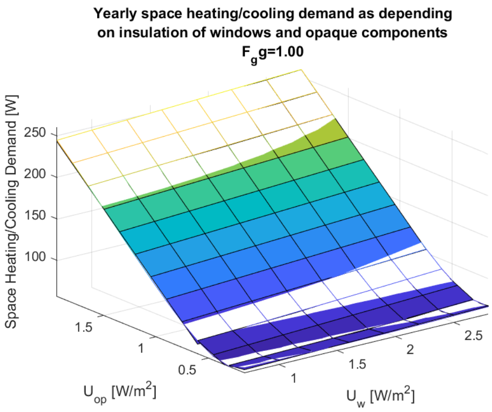
(c) $F_g g = 0.396$, $q = 0.4$



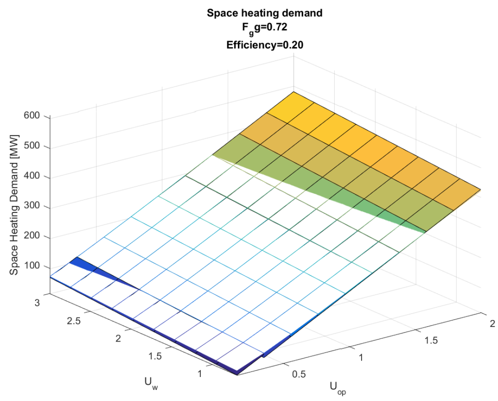
(d) $F_g g = 0.396$, $q = 0.6$



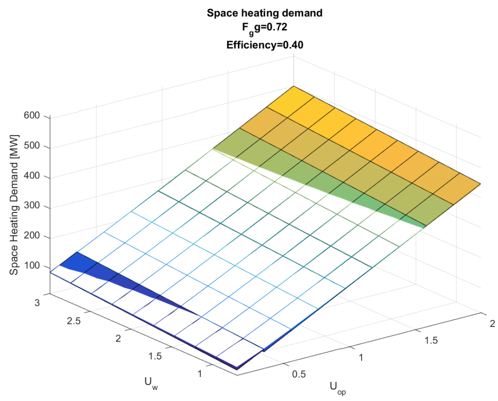
(e) $F_g g = 0.396$, $q = 0.8$



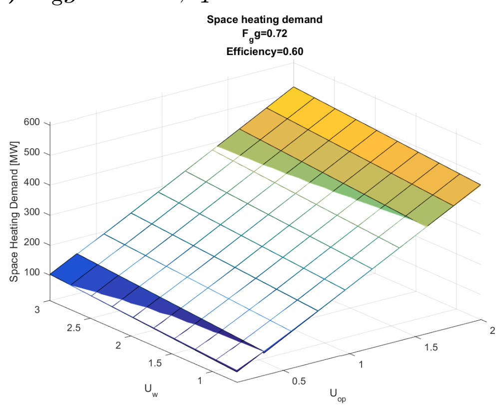
(f) $F_g g = 0.396$, $q = 1.0$



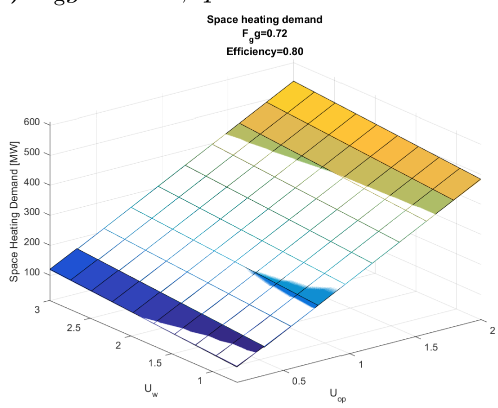
(a) $F_{\mathrm{g}}g = 0.72,\ q = 0.2$



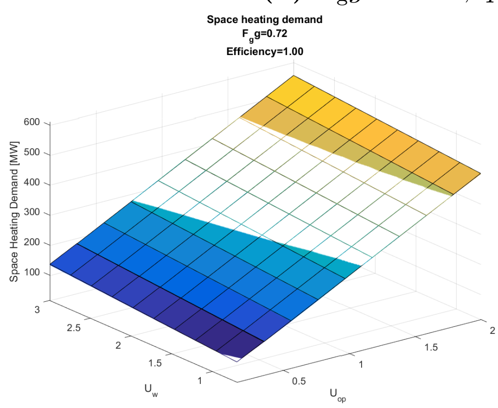
(b) $F_{\mathrm{g}}g = 0.72,\ q = 0.4$



(c) $F_{\mathrm{g}}g = 0.72,\ q = 0.6$



(d) $F_{\mathrm{g}}g = 0.72,\ q = 0.8$



(e) $F_{\mathrm{g}}g = 0.72,\ q = 1.0$

D

Objective functions

Useful Energy Demand
Useful Energy Demand of Heating
Useful Energy Demand of DHW
Final Energy Demand
Final Energy Demand of Heating
Final Energy Demand of DHW
Final Energy Demand of Building Electricty
Final Energy Demand of Household Electricity
Efficiency
Efficiency Heating
Efficiency DHW
Useful Energy Savings
Useful Energy Savings of Heating
Useful Energy Savings of DHW
Final Energy Savings
Final Energy Savings of Heating
Final Energy Savings of DHW
Final Energy Savings of Building Electricty
Final Energy Savings of Household Electricity
Electricity Production
Self Consumption of Produced Electricity
Self Production of Consumed Electricity
GHG Emission
Primary Energy Total
Primary Energy Non Renewable
Primary Energy Renewable
GHG Emission Savings
Primary Energy Total Savings
Primary Energy Non Renewable Savings
Primary Energy Renewable Savings
Investment Costs
Maintenance Costs
Operational Costs
Energy Costs
Earnings Electricity Production Feed In
Total Life Cycle Cost
Internal Rate Of Return

Modified Internal Rate Of Return
Net Present Value
Profitability Index
Cost Savings
Final Energy Savings Investment
Return On Investment

E

Component descriptions and data

Refurbishment option name (roof)	Cost/m^2 (SEK)
* 50 mm insulation TY 1523	409
* 100 mm insulation	446
* 150 mm insulation	484
* 200 mm insulation	533
* 50 mm insulation TOR	519
* 100 mm insulation TOR	565
* 160 mm insulation TOR	630
* 100 mm insulation TRP	945
* 130 mm insulation TRP	951
* 2*100 mm insulation TRP	1090
* 2*130 mm insulation TRP	1102

*External Insulation above the roof slab

Refurbishment option name (floor)	Cost/m^2 (SEK)
Insulation on the inside (heated side) of the floor slab - 45 mm insulation	1071
Insulation on the inside (heated side) of the floor slab - 70 mm insulation	1124
Insulation on the inside (heated side) of the floor slab - 95 mm insulation	1187
Insulation on the outside (unheated heated side) of the floor slab - 50 mm insulation	196
Insulation on the outside (unheated heated side) of the floor slab - 70 mm insulation	208
Insulation on the outside (unheated heated side) of the floor slab - 100 mm insulation	232
Insulation on the outside (unheated heated side) of the floor slab -100 mm on ground	361
Insulation on the outside (unheated heated side) of the floor slab - 200 mm on ground	617
Insulation on the outside (unheated heated side) of the floor slab - 300 mm on ground	874

Refurbishment option name (window)	Cost/m^2 (SEK)
9x12 Window U-Value 1.5 -Wood	4947
9x12 Window U-Value 1.2 -Wood	6779
9x12 Window U-Value 1.2 - Wood/aluminium	8402
9x12 Window U-Value 1.1 - Wood/aluminium	7705
9x12 Window U-Value 1.1 -Aluminium	7319
9x12 Window U-Value 0.8 - Wood/aluminium	9074

Refurbishment option name (wall)	Cost/m^2 (SEK)
Brickwall add insulation - 50 mm insulation	1558
Brickwall add insulation - 80 mm insulation	1572
Brickwall add insulation - 100 mm insulation	1599
Brickwall add insulation - 120 mm insulation	1651
Brickwall add insulation - 150 mm insulation	1696
Brickwall add insulation - 170 mm insulation	1758
Sandwich add insulation - 50 mm insulation	1249
Sandwich add insulation - 80 mm insulation	1269
Sandwich add insulation - 120 mm insulation	1353
Sandwich add insulation - 150 mm insulation	1398
Sandwich add insulation - 200 mm insulation	1497

Refurbishment option name (ventilation)	Cost
Exhaust air Q-value 1	541580
Heat recover Q-value 0.2	543725