Optimizing Solar Irradiation in Twisted Towers: A Parametric and Emulator-Based Study of the Turning Torso

GROUP 14

Farnaz Ahmadi Vafa – farnaz.ahmadi@mail.polimi.it – 274387
Jacopo Guerinni – jacopo.guerinni@mail.polimi.it – 278492
Mohammad Amin Shojaee – mohammadamin.shojaeechahragh@mail.polimi.it – 270608
Salma Nabilah Yanuar – salmanabilah.yanuar@mail.polimi.it – 273563

SCIENTIFIC COMPUTING FOR BUILDING ENGINEERING **062367**
2024/2025

Table of Contents

Table of Figure

1. Introduction

This project investigates how parametric design and computational modelling can support the optimization of floor rotation and base surface length to maximize solar irradiation on the façade of a twisted high-rise tower. The main objective is to identify which combinations of floor geometry and rotation maximize solar exposure on the shortest solar exposure of the year, which is on 21st of December. Using the Turning Torso in Malmö as a case study, we simulate and analyse geometric variations to apply scientific computing methods to a real architectural challenge. Our approach combines both direct simulation and emulator modelling to inform performance-driven design decisions.

Figure 1 Turning Torso

In recent decades, parametric design has become central to architecture and building engineering, enabling designers to define geometry through adjustable parameters and systematically explore a range of design options. When linked with environmental simulation tools, parametric methods offer powerful strategies to optimize buildings for daylighting, energy efficiency, and solar performance [1] [2]. Among early examples of iconic parametric buildings, the Turning Torso, designed by Santiago Calatrava and completed in 2005, stands out as the first twisted skyscraper. Its 90-degree rotational geometry makes it ideal for studying how floor-by-floor orientation impacts solar access.

Numerous studies have explored parametric optimization in architectural performance, particularly in daylighting and thermal comfort. For instance, Klimczak et al. (2018) used solar radiation mapping to inform adaptive façade design, while Salama

and Gadelha (2020) applied evolutionary algorithms to optimize twisted high-rise forms for environmental efficiency. However, few have systematically analysed how the combination of geometric proportions and rotation—under a fixed perimeter constraint—affects solar performance. This constraint reflects real-world design conditions, where total floor perimeter is often dictated by structural, programmatic, or cost limitations.

To address this gap, we abstract the Turning Torso's floor plan into a simplified pentagonal shape with alternating side lengths, labelled L1 and L2. We simulate combinations of L1–L2 ratios and floor rotations (ranging from 10° to 120°) to evaluate how these parameters influence solar exposure. Solar irradiation is calculated using Ladybug Tools within Rhino-Grasshopper, using weather data for Malmö. To extend the analysis beyond discrete simulations, a predictive emulator model is built using Python enabling interpolation across a wider design space.

Figure below illustrates the floor plan logic used in the parametric model. The pentagon maintains a constant perimeter while allowing L1 and L2 to vary, with internal angles (a, b, c) that are fixed. This setup provides a controlled yet flexible framework for exploring how geometry and orientation impact solar exposure.

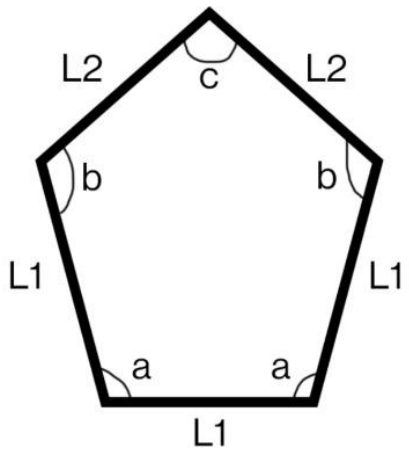

Figure 2 Floor Plan Optimization Scheme a five-sided polygonal plan with alternating edge lengths L1 and L2 and corresponding internal angles a, b, and c.

1.1. Objectives

This project aims to achieve the following goals:

- Investigate the impact of floor rotation on solar irradiation levels across the façade of the Turning Torso.
- Evaluate how the proportions of L1 and L2, while keeping the perimeter constant, affect the building's ability to receive solar energy.
- Simulate various geometric configurations of the pentagonal floor plan using parametric tools to identify patterns in solar performance.
- Determine the optimal combination of floor rotation and geometry that maximizes solar gain in the context of Malmö's environmental conditions.

- Apply mathematical reasoning and computational design to support architectural decisions with scientific evidence.
- Create a polynomial surface that best fits the solar irradiation dataset to accurately model the relationship between geometry and solar performance.
- Optimize the design parameters (rotation angle and L1) to achieve a specific target solar irradiation level (6000 kWh), using the SciPy's minimize with Powell algorithm to find the most accurate and feasible configuration.

2. Methodology

2.1. Parametric Model Creation in Rhino-Grasshopper

The parametric model was developed in Grasshopper for Rhino, a visual scripting environment ideal for architectural geometry manipulation. A single floor plan of the Turning Torso was abstracted as a regular pentagon composed of alternating edge lengths, labelled L1 and L2.

The following key parameters were defined and controlled:

- L1 (m): One of the alternating side lengths of the pentagon
- L2 (m): The second alternating side length, calculated to maintain constant perimeter
- P (m): The total perimeter (constant across all configurations)
- Θ : Floor rotation angle in degrees, simulating the twisting form

For each configuration, Grasshopper recalculated internal angles (a, b, c) to maintain valid polygonal geometry. The model was designed to generate multiple iterations of the floor plan by adjusting L1 and θ through sliders. The parametric flexibility allowed the testing of:

8 values of L1: [16, 17, 18, 19, 20, 21, 22, 23]

12 rotation angles: [10°, 20°, 30°, 40°, 50°, 60°, 70°, 80°, 90°, 100°, 110°, 120°]

2.2. Formalization of the Addressed Problem

The goal of this project is to determine which combination of floor shape and rotation results in the highest solar irradiation during lowest solar exposure in the year on the façade of a floor in a twisted high-rise structure. The problem is approached as a parametric optimization task, where both the floor's geometric proportions and its rotation are treated as variables.

The geometry is based on a five-sided polygon (pentagon) with alternating edge lengths, labelled L1 and L2. The total perimeter of the shape remains constant, so when L1 changes, L2 is recalculated accordingly using the formula:

$$L_2 = \frac{P - 3L_1}{2}$$

where P is the fixed total perimeter. Each floor is then rotated by an angle θ, which ranges from 10° to 120° in our tests.

The objective is to find the values of L1 (measured in meters) and θ (in degrees) that maximize the amount of solar irradiation $I(L_1, \theta)$ received on the façade. This leads to the following initial optimization formulation:

$$Maximize\ I(L_1, \theta)\ where\ L_1 \in [16{,}23]\ m, \theta\ \in [10°, 120°]$$

Since $I(L_1, \theta)$ is calculated from simulation data and not available as an explicit formula, the optimization relies on simulation outputs and an emulator model trained to estimate irradiation values for any input combination.

To train the emulator, we used Sum of Squared Errors to measure the discrepancy between predicted values and actual data in regression and emulator models and we used a mean squared error (MSE) loss , which compares the predicted irradiation values to the actual simulation results. The loss function is defined as:

$$Loss = \frac{1}{N}\sum_{i=1}^{N}(I_{sim}^{(i)} - I_{Pred}^{(i)})^2$$

where N is the number of data points, I_{sim} is the irradiation from the Ladybug simulation, and $I_{Pred}$is the value predicted by the emulator.

2.3. Emulator Model Development

In computational design and building performance analysis, simulation-based optimization often becomes computationally expensive when exploring a wide range of geometric configurations. To address this limitation, an emulator—also known as a surrogate model—is employed to approximate the results of detailed simulations using a fast, predictive model trained on a limited dataset.

In this project, the emulator is designed to predict total solar irradiation [kWh] received by a twisted tower configuration on 21 December. Unlike typical studies that focus only on vertical façades, this analysis includes solar energy incident on both the vertical envelope and the horizontal openings and roof of the building.

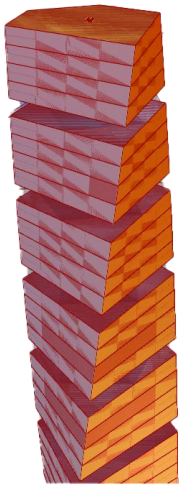

Figure 3 Floor Blocks

The emulator takes two inputs:

- L1: the length of one of the alternating sides in the pentagonal floor plan,
- θ: the rotation angle of each floor simulating the tower's twist

The use of the emulator in this context serves three key purposes:

1. Prediction of irradiation for new configurations

   The emulator allows for the estimation of solar irradiation values for combinations of L1 and θ that were not explicitly simulated in Ladybug, thus expanding the analytical scope of the study.

2. Support for parametric design exploration

   Designers can quickly evaluate how changes in floor geometry and rotation affect solar performance across a wide design space, without re-running resource-heavy simulations for each case.

3. Acceleration of optimization processes

   By replacing simulation loops with instant predictions, the emulator facilitates rapid feedback during optimization routines, making it feasible to search for solar-optimal designs under perimeter constraints.

   Through these functions, the emulator becomes a valuable component in the computational workflow—bridging parametric modelling, environmental simulation, and data-driven prediction.

2.4. Optimization and Python Scripting

As the emulator analysis demonstrated, a best polynomial surface provides to our dataset. Having determined the surface's coefficients and explicit equation, we now advance to the optimization phase.

Originally, the project's primary objective was to identify the rotation angle and L_1 value that yield the maximum radiation. Both the initial data visualization and the polynomial surrogate model confirmed that this global maximum occurs at a single, fixed combination of angle and L_1. To render the optimization task more meaningful, we instead specify a target irradiation level 6000 kWh—and seek the rotation angle (r) and L_1

length (L) that minimize the discrepancy between the surrogate model's predicted radiation and this target.

In other words, we optimize r and L to bring the polynomial model's output as close as possible to 6000 kWh. Mathematically, this amounts to minimizing the absolute error between the surrogate equation's radiation estimate and 6000 kWh. A smaller error thus corresponds to a configuration of r and L capable of delivering the desired 6000 kWh.

It's important to note that this value represents the total incident solar energy over a full 24-hour period on December 21st—the shortest day of the year—making it the worst-case scenario. The calculated 6,000 kWh is the sum of solar radiation on all input geometry. At the end the possibility to change this value to an ideal value is provided so the building can be optimized according to the average daily need of the building.

To find the Optimization of the Target Irradiation, we find the angle r and length L that produce a desired irradiation level of 6000 kWh, we set up and solved the following optimization problem.

2.4.1. Goal and Error Measure

We use our surrogate model $\hat{f}(r, L)$ from Section 3.1 to measure how far its prediction is from 6000 kWh. We define the error function:

$$E(r, L) = (\hat{f}(r, L) - 6000)^2$$

Minimizing $E(r, L)$ ensures that the model's output moves as close as possible to the target value

2.4.2. Method and Practical Limits

To solve this problem, we employ the Powell algorithm via SciPy's `minimize` routine. Powell's method was chosen because it does not rely on gradient information, an advantage when working with a complex polynomial surrogate whose derivatives are not readily available. We also impose realistic bounds on the variables, restricting r between 10° and 120° and L between 16 m and 23 m. These limits reflect the range of our original dataset and ensure the solution remains within physically meaningful parameters.

$$10° \leq r \leq 120° \quad , \quad 16 \leq L_1 \leq 23$$

In summary :

- Algorithm: Powell method, a derivative-free solver, as our surrogate surface does not provide an easy way to compute gradients.
- Variable ranges: To keep the solution within realistic settings.
- Implementation: SciPy's minimize function (with method='Powell') was used. It explores different (r, L) combinations and refines them until the error E can no longer be reduced.

2.4.3. Automated Geometry Control and Data Collection

The overall logic, workflow, and objectives of the methodology remain consistent with earlier stages of the project. However, several improvements have been introduced to enhance efficiency, precision, and automation. The primary modification concerns the use of Python scripting in place of multiple Grasshopper components to control the building's base floor geometry.

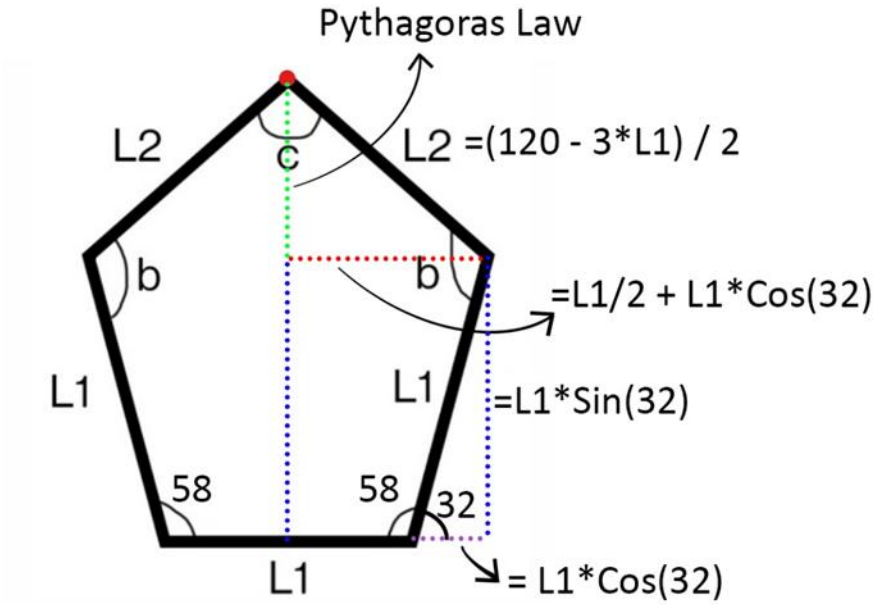

Figure 4 Mathematical Configurations of Floor Plan

As illustrated in Figure above, the three lower edges of the pentagonal base share the same length, and the angle α is fixed for all design alternatives. The total perimeter is also kept constant across all configurations. Using geometric relationships and trigonometric functions, implemented via Python's math library, it is possible to calculate the coordinates of the top vertex directly for varying values of L1. This approach eliminates the need to determine intermediate angles β and γ explicitly. Once the top vertex coordinates are obtained, they are connected to the two ends of the middle L1 segment to define the complete building footprint.

The schematic in Figure above demonstrates the mathematical procedure used to determine the top vertex coordinates. By varying the parameter L1 within a defined range (16–23 m), the script automatically recalculates the corresponding L2 values to satisfy the fixed 120 m perimeter constraint, and updates the building geometry accordingly in real time.

A major enhancement to the Grasshopper workflow lies in the automation of the emulator process, input parameter assignment, and result data collection through the use of two dedicated Python scripts. The first script controls the assignment of input parameters, which include L1, rotation angle, number of floors, and building location. These parameters are stored in number components, allowing manual adjustment if desired. For automated runs, the script iterates through all possible parameter

combinations within user-defined limits, applying a controlled delay between iterations to allow the model to update. This ensures comprehensive coverage of the design space.

The second Python script is responsible for data extraction and storage. It receives six inputs: the four initial model parameters (L1, rotation, number of floors, location) and the two main simulation outputs (total incident irradiation and total building surface area). Using Python's csv library, the script appends each complete set of inputs and outputs as a new row in a CSV file. Consequently, every iteration generates a unique record, producing a comprehensive dataset that can be used for further optimization, analysis, and visualization.

Through the integration of parametric modelling, Python scripting, and automated data handling, the workflow achieves a higher level of precision, reduces manual intervention, and supports systematic exploration of the design space in alignment with the project's optimization goals.

3. Results

3.1. Parametric Modelling

To construct the vertical form of the tower, a Series component is used to generate a list of Z-values that represent the height of each floor, typically spaced at regular intervals (4 meters per floor). These Z-values are then used to vertically move each pentagonal floor plate upward, using the Move component combined with Z-axis vectors to determine the direction and distance of translation.

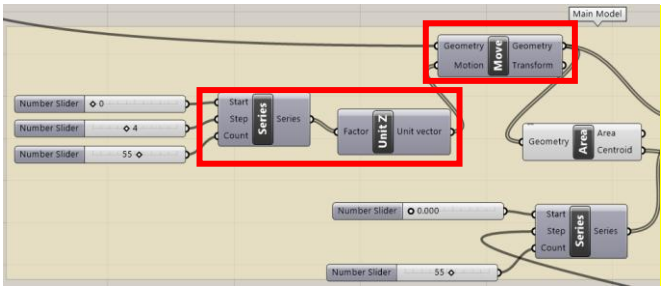

Figure 5 Parametric Modelling Move Component

To simulate the characteristic twist of the Turning Torso, a progressive rotation is applied to each successive floor. This is achieved by multiplying a base rotation angle, θ, by the index of the floor in the stack, such that each level is rotated slightly more than the one below it (see 12 rotations angles above). The resulting angle for each floor, $\theta_{(floor)}$ = θ × floor index, is then applied using the Rotate component. Since Grasshopper's rotation operations work in radians, a DegToRad component is used to convert degrees into the appropriate format before the transformation.

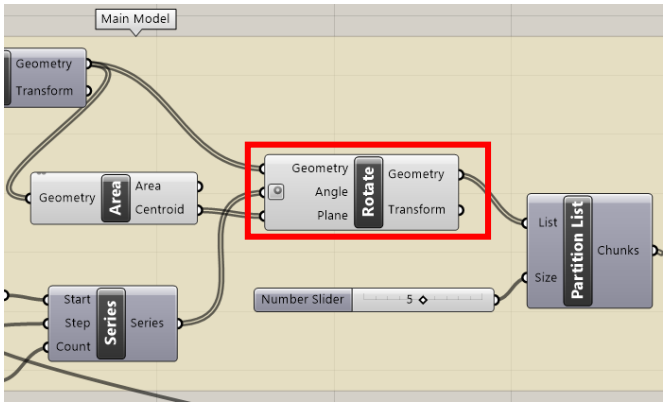

Figure 6 Parametric Modelling Rotate Component

Once all the pentagons have been positioned and rotated correctly, they are passed into a Loft component, which generates a continuous surface that connects each floor, forming the twisted shell of the tower. Last to capped the geometry, Cap Holes components it's added to close off the top and bottom faces of the lofted form, resulting in a clean, watertight model.

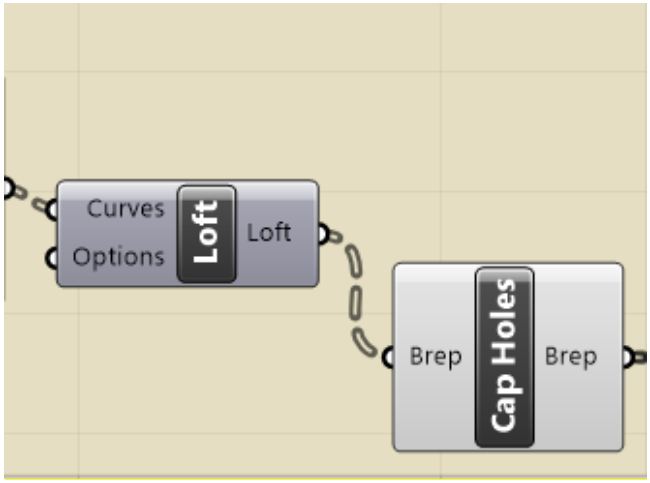

Figure 7 Parametric Modelling Loft and Cap Holes Component

After generating the geometry of the tower the Cap Holes component it's crucial for converting open breps into closed solids, which is necessary for accurate volumetric and area analysis. Once the geometry is capped and forms a closed brep, it is possible to extract useful geometric data, such as surface area.

To analyse the total area of the façade it's necessary add the Area component to compute the area of each face.

After calculating the area for each relevant surface, the Mass Addition component is applied to sum all the individual values, yielding the total surface area of interest. This total it's used then for the analysis of solar gain.

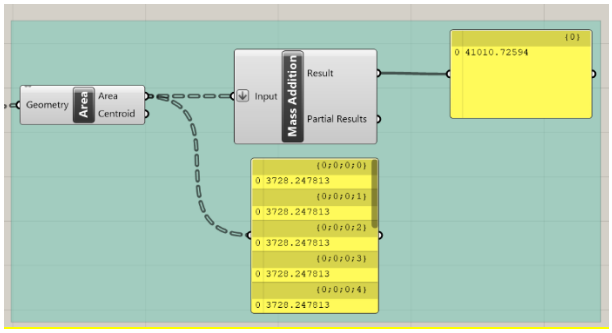

Figure 8 Parametric Modelling Total Surface

To enhance the environmental performance of the Turning Torso model, Ladybug Tools was integrated into the Grasshopper workflow to simulate and optimize solar irradiation across the building's facade. This process was aimed at informing geometric adjustments—particularly the rotation angle of each floor—based on how sunlight interacts with the twisting surface throughout the year.The analysis began with the incorporation of location-specific climate data. An EPW (EnergyPlus Weather) Malmo file was imported using Ladybug's weather data components, providing essential information such as solar radiation levels, temperature, and daylight hours. This data served as the foundation for accurate solar exposure simulations.

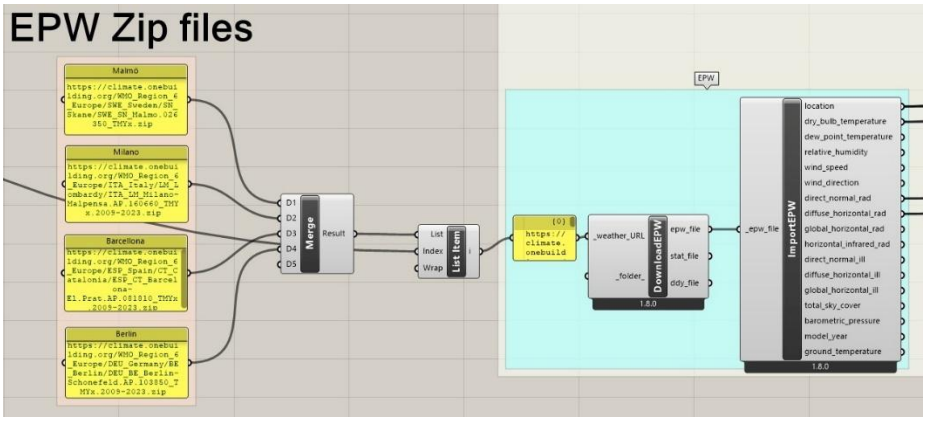

Figure 9 Parametric Modelling EPW File Component

Once the weather data was in place, the twisted geometry of the tower—constructed from stacked and rotated pentagonal floor plates—was converted into a mesh. This step was necessary because Ladybug evaluates solar exposure on a per-face basis, and mesh geometry is required for this type of simulation. The mesh surface was then connected to Ladybug's solar irradiation analysis components, which calculated the total cumulative radiation each part of the facade would receive over a designated time period, typically spanning an entire year.

As the simulation ran, it produced both numerical output and visual feedback. The tower's facade was mapped with a colour gradient, ranging from blue to red, indicating zones of low to high solar exposure, respectively. These visualizations made it immediately clear how different parts of the tower were affected by sun orientation, self-shading, and seasonal variation.

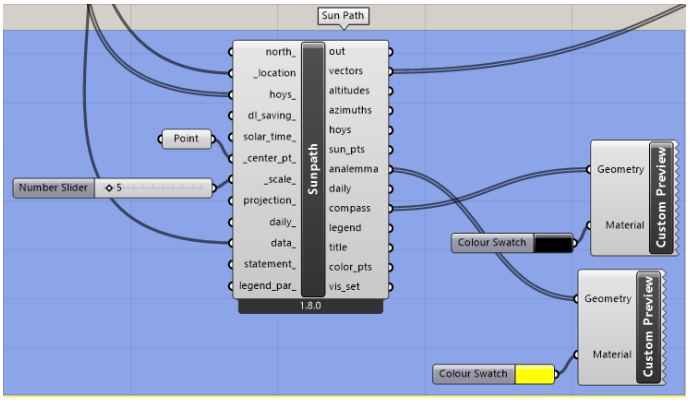

Figure 10 Parametric Modelling Sunpath Component

3.2. Radiation Analysis

3.2.1. Climate-Based Solar Simulation Overview

Using the LadyBug, Lunchbox, and HB-Radiance extensions in Grasshopper, it is possible to conduct detailed climate-based analyses and visualize the building's and surrounding environmental conditions with high accuracy. For this study, the EPW climate file for Malmö was employed to ensure realistic simulation results.

By using the Period component, it is possible to analyse building performance under worst-case seasonal conditions. In this case, 21 December was selected as it is the day with the shortest daylight duration and the lowest solar irradiation of the year. This approach allows for the optimization of the building geometry to meet a predicted daily energy requirement of 6000 kWh under the most challenging solar conditions. If the building achieves this target during the worst-case scenario, it can reliably meet energy demands throughout the year.

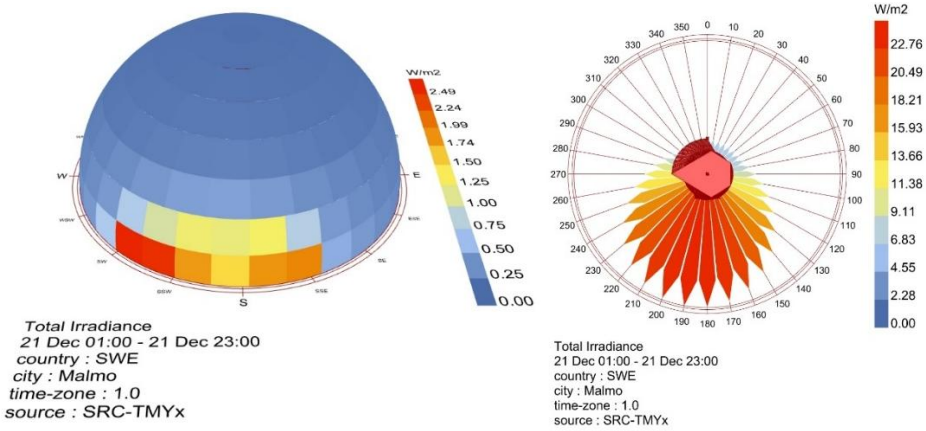

Figure 11 LadyBug Total Irradiance

The simulations focused on parameters most relevant to the project, particularly incident solar irradiation. The use of an EPW file enables precise calculation of these values, facilitating informed design decisions. The analysis outputs include the distribution of dry bulb temperature, annual solar position diagrams, irradiance rose and

dome plots with corresponding legends, incident irradiance and irradiation values for the building façades, as well as other relevant solar gain visualizations.

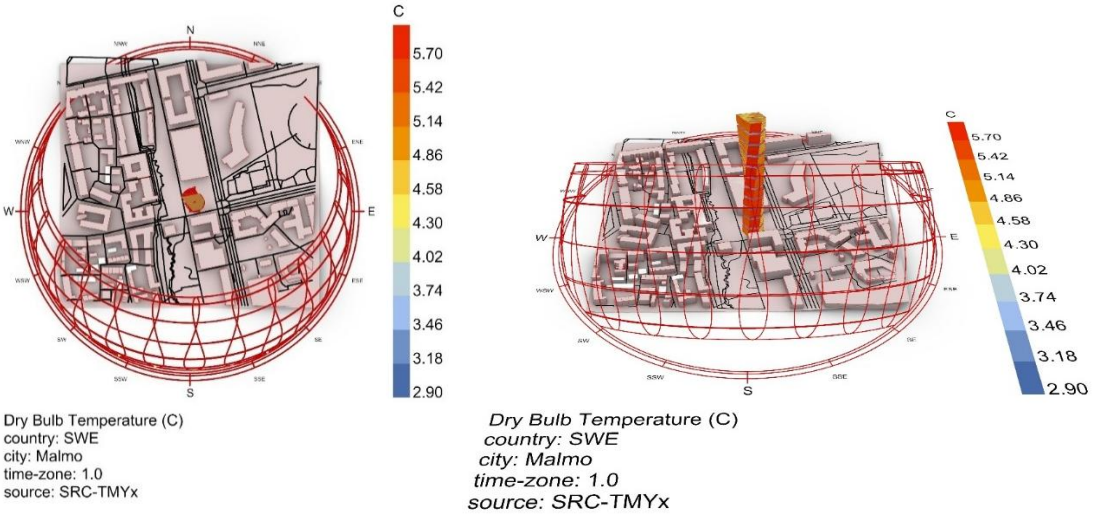

Figure 12 LadyBug Dry Bulb Temperature

The dry bulb temperature visualizations presented in the diagrams illustrate the average temperature distribution around the site in Malmö based on the EPW climate file. The spherical and cylindrical representations demonstrate that the lowest ambient temperatures occur predominantly from the north and northeast directions, which correlates with seasonal wind and solar patterns in this region. The warmest exposures are concentrated toward the south, as expected, with maximum dry bulb temperatures reaching around 5.7°C. These visual insights help contextualize the need for maximizing solar gain in winter, particularly on southern façades. By understanding the temperature gradients and their orientation, design strategies such as optimal rotation and façade surface area allocation can be informed not only by solar availability but also by ambient thermal conditions. This reinforces the relevance of the selected date, December 21st, and the importance of performance-driven design even in low-temperature, low-irradiation environments.

These visualizations demonstrate the influence of building geometry, orientation, rotation, and height on solar energy gains. The results highlight that a well-optimized design, when considering the environmental context, design constraints, and the potential of the building envelope, can significantly reduce reliance on non-renewable energy sources while maximizing solar energy utilization.

This methodology shows the importance of integrating climate-based simulation tools into early-stage design to ensure sustainable energy performance.

3.2.2. Radiation Contour Analysis

The radiation contour plots provide a visual overview of the 21st December solar irradiation values across various combinations of floor rotation angles and L1 values. These plots offer insight into how the two main design parameters—rotation and floor plan geometry—interact to influence solar exposure. The X-axis represents the rotation

of the floor plate (in degrees), while the Y-axis corresponds to the edge length L1 (in meters). The colour gradients denote the total irradiation, ranging from approximately 1600 to over 7000 kWh in 21st of December.

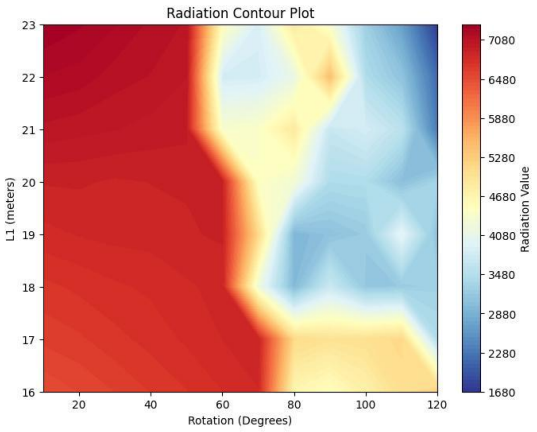

Figure 13 Radiation Contour Plot showing 21st December irradiation across parameter combinations.

In Figure above, we observe that solar irradiation is significantly higher in configurations with lower rotation angles, especially in the range of 10° to 50°. Configurations with L1 values above 20 meters perform consistently better in terms of irradiation capture. This indicates a positive correlation between longer floor edge and solar exposure—likely due to the increased surface area facing the sun.

3.2.3. Rotation Sensitivity at Maximum L1

To isolate the effect of rotation on performance, Figure below presents a line plot of radiation values at L1 = 23 meters for all tested rotation angles. The data shows a consistent decline in radiation as rotation increases. From 10° to 50°, the reduction is relatively modest, but a sharp drop occurs between 50° and 60°, suggesting a critical threshold where performance is significantly compromised.

Beyond 60°, solar exposure continues to decline, with minor fluctuations likely caused by angular changes in façade orientation and its relation to solar azimuth angles in Malmö's geographic context. This plot validates the assumption that excessive rotation induces self-shading and misalignment with the sun path, significantly reducing solar gain.

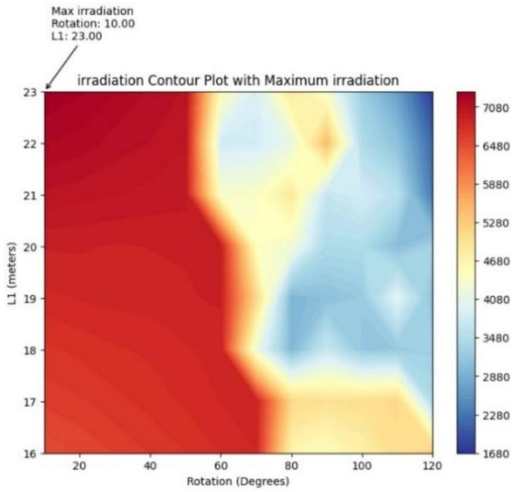

Figure 14 Solar irradiation as a function of rotation angle for fixed L1 = 23 m.

3.2.4. Heatmap Visualization

Figure below presents a heatmap representation, offering a grid-based visualization of the same parameter space. The red zone in the upper left (low rotation, high L1) clearly indicates the most favourable configuration for maximizing solar performance. The sharp contrast between red and blue regions suggests a nonlinear relationship—where small increases in rotation beyond 60° result in disproportionately large performance penalties.

This graphical approach helps reinforce the contour observations and reveals the steepness of the performance gradient across the parameter space.

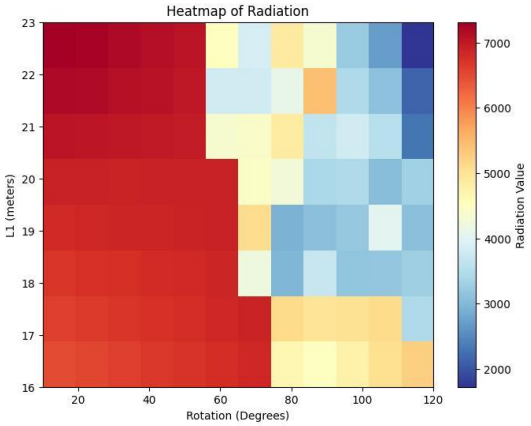

Figure 15 Heatmap showing solar radiation values across L1 and rotation combinations.

3.2.5. 3D Surface Plot

The 3D surface plot in Figure below offers a spatial visualization of how solar performance changes with both variables. The elevated corner at low rotation and high L1 further confirms the optimal configuration. The steep decline along the rotation axis reinforces earlier findings—rotation plays a dominant role in decreasing solar access.

This perspective makes it visually apparent that maximizing solar irradiation is not merely a function of geometry size (L1), but is heavily influenced by angular positioning.

The plot surface shows irregularities at high rotations, which may represent complex solar interactions, including diffused light and shadow overlaps not immediately intuitive from 2D views.

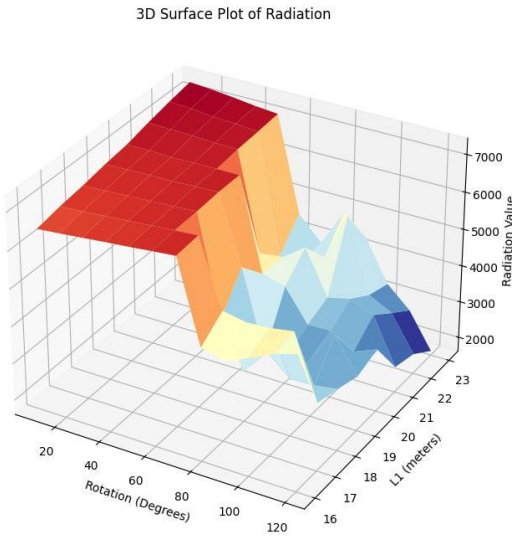

Figure 16 3D surface plot of solar irradiation values based on parametric input space

3.2.6. Conclusion of the Analysis

This radiation analysis reveals several key insights for performance-driven design of twisted towers:

- The optimal configuration for solar gain is found at L1 = 23 meters and a minimal rotation of 10°.
- Rotation has a significantly stronger influence on solar performance than floor edge length, with steep losses beyond 60°.
- Larger L1 values help increase surface area, but their benefit is overridden by the negative impact of excessive twist.

From a design standpoint, these findings suggest that in climates with limited sunlight—like Malmö—it is more beneficial to limit rotational distortion when solar performance is a priority. While twisting may provide visual interest and aerodynamic advantages, it compromises solar efficiency unless carefully optimized. This supports the use of parametric and emulator tools early in the design process, helping to balance aesthetics with performance goals.

3.3. Emulator Model Development

As a first step in emulator development, a second-degree polynomial regression model was implemented to approximate the relationship between geometric parameters and solar irradiation.

The emulator was trained on 96 data samples (8 values of L1 × 12 values of θ), generated through simulations in Grasshopper using Ladybug Tools. The two input

parameters—rotation angle (r) and side length (L1)—were expanded using polynomial feature transformation to include non-linear and interaction terms:

$$\{1, r, L_1, r^2, r.L_1, L_1^2\}$$

This transformation enabled the linear regression model to fit a curved response surface, capturing the essential non-linear behaviour in the data while maintaining model interpretability and computational efficiency. The resulting emulator provided a strong balance between prediction accuracy, speed, and clarity, making it well-suited for rapid evaluation of solar performance in parametric design. By enabling fast and reliable prediction of total solar irradiation, it supports early-stage decision-making in the context of solar-responsive architectural form-finding.

The general form of a second-degree polynomial with two variables r and L_1 is:

$$f(r, L_1) = a_1.r^2 + a_2\,.r\,L_1 + a_3\,.{L_1}^2 + a_4\,.\,r + a_5\,.L_1 + a_6$$

where each coefficient a_i is learned from the data

The final fitted equation representing the emulator surface is:

$$f(r, L_1) = 15043.6 + 60.5121\,r - 946.506\,L_1 - 0.162797\,r^2 - 4.08447\,r\,L_1 + 28.2463\,{L_1}^2$$

Where:

- r is the floor rotation angle in degrees,
- L_1 is the length of the alternating side in the pentagonal floor plan (in meters),
- $f(r, L_1)$ is the predicted total solar irradiation in kilowatt-hours (kWh) for 21 December.

To assess the accuracy of the emulator, a scatter plot was generated

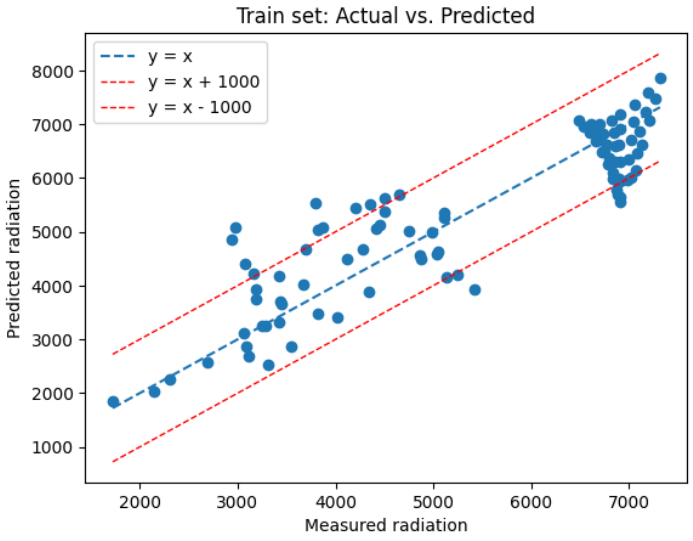

Figure 17 Train Set : Actual vs Predicted Emulator

Comparing the predicted radiation values from the second-degree polynomial model with the actual simulation results from the training dataset. In this plot, the diagonal

dashed line represents the ideal case where predicted values exactly match the measured ones (i.e., the line y=x). The closer the data points are to this line, the better the model's predictive performance. To better evaluate prediction accuracy, a ±1000 kWh error threshold was visualized using parallel dashed lines around the ideal line

Visual inspection reveals that the model performs reasonably well for lower irradiation values (below 3500 kWh), where predictions closely follow the reference line. However, for higher irradiation values, the distribution of points becomes more dispersed. While a cluster of points still lies near the ideal line, many are scattered significantly above or below it and there are Several data points falling outside this band,. This indicates that the second-degree polynomial model lacks the flexibility needed to accurately capture the variation in higher radiation zones, which are more sensitive to geometric flexibility. As a result, although the model is useful for capturing general trends, it exhibits clear limitations in predictive precision, particularly for high-performance configurations.

To quantitatively assess the performance of the second-degree polynomial regression model, the Sum of Squared Errors (SSE) was computed both before and after model fitting.

SSE measures the total squared difference between predicted values and actual observations. It is a widely used metric in regression analysis because it directly reflects the cumulative magnitude of prediction errors: the lower the SSE, the closer the model's predictions are to the real data.

In this project, the initial SSE—obtained from a baseline model that only predicts the mean of the output—was 25,735,518.19. After fitting the polynomial surface to the data, the SSE dropped significantly to 5,097,169.48, representing a reduction of 80.2%. This strong decrease demonstrates that the emulator model captures a substantial portion of the variance in the simulation results and is effective at approximating the general solar irradiation trend across the parameter space.

SSE is especially useful here because it allows for a clear, numerical comparison between the baseline prediction and the emulator's performance, providing evidence of improvement gained through modelling.

However, while SSE confirms overall error reduction, visual identified localized inaccuracies, particularly in high-irradiation configurations.

To better understand the spatial distribution of prediction errors across the design space, a heatmap of absolute error was generated by comparing the emulator's predictions with the actual simulation results figure below. The heatmap visualizes the absolute difference in solar irradiation [kWh] over the full range of input parameters. Areas in lighter shades (yellow white) indicate low prediction error, while darker regions (purple-black) represent zones where the emulator deviates more significantly from the simulation data.

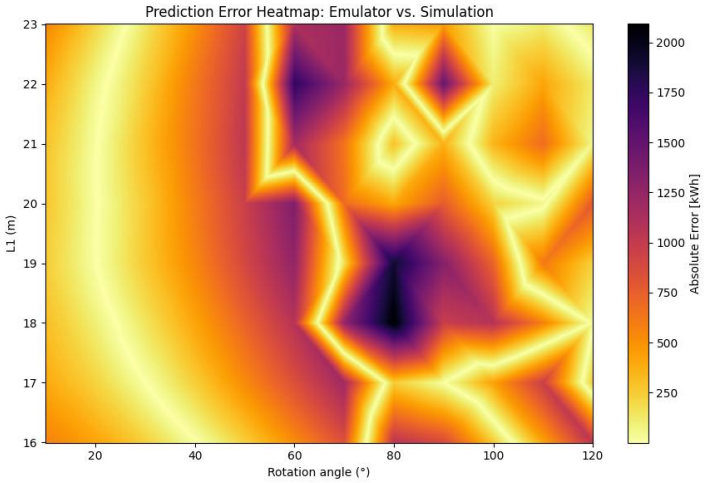

Figure 18 Prediction Error Heatmap: Emulator vs Simulation

The results show that the emulator performs relatively well for low rotation angles and smaller L1 values, where errors remain under 250 kWh. However, in more geometrically complex configurations—especially around rotation angles between 60° and 90° and L1 values between 18 m and 22 m—the error increases noticeably, reaching over 2000 kWh in some regions. This visualization confirms that while the second-degree polynomial model captures overall trends, it struggles to generalize across the full design space, particularly in higher-performance or edge configurations.

To test the emulator, a known configuration from the dataset (rotation = 20°, L1 = 20 m) was used as input. The model predicted a solar irradiation value of 6923.34 kWh, while the actual simulated value was 6909.38 kWh, showing a close match. This confirms that the emulator can generalize well and provide valid predictions for any combination of input parameters within the trained range.

The trained polynomial model was used to predict solar irradiation over this grid, generating a smooth surface of expected values. The optimal point on this surface(the configuration yielding maximum predicted solar gain) was identified at:

$$r = 10.00\circ, \text{L1} = 23.00 \text{ m} \Rightarrow \text{predicted irradiation} \approx 7865.7 \text{ kWh}$$

However, a comparison with the actual simulation data revealed that the true irradiation value for this same configuration is 7315.26 kWh, while the model predicted 7865.7 kWh, indicating a noticeable prediction error. This discrepancy highlights a limitation of the second-degree polynomial regression model used. While this model effectively captures general performance trends, its capacity to approximate complex or sharply varying responses is limited. This suggests that a more flexible function form may be required for improved accuracy. Therefore, in the next step, higher-degree polynomial models are explored to assess whether they offer a better fit and more reliable predictions across the full design space.

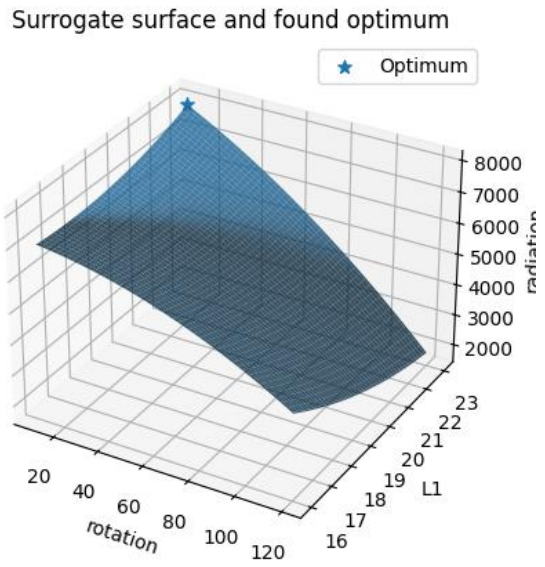

Figure 19 Surrogate Surface and Found Optimum

To ensure a reliable evaluation of model performance, 7-fold cross-validation was applied. This method allows the model to be tested on different subsets of the data, helping to reduce bias and variance in the error estimation (especially valuable with a limited dataset like this dataset). To identify the optimal model complexity, polynomial regression models with degrees from 1 to 7 were evaluated using this approach. The metric used for comparison was the mean squared error (MSE).

```
degree= 1 | MSE=745959.2 ±213191.9
degree= 2 | MSE=618460.7 ±131119.3
degree= 3 | MSE=528870.2 ±176464.7
degree= 4 | MSE=576594.5 ±157588.9
degree= 5 | MSE=506130.3 ±192328.0
degree= 6 | MSE=696998.2 ±290002.7
degree= 7 | MSE=840371.1 ±467558.7
```

The results showed that degree 5 achieved the lowest average MSE of approximately 506,130 ± 192,328, outperforming both lower and higher-degree models. This suggests that degree 5 provides the best trade-off between underfitting and overfitting, capturing complex patterns in the data while maintaining generalization capability. Therefore, this degree was selected for further analysis and final model fitting.

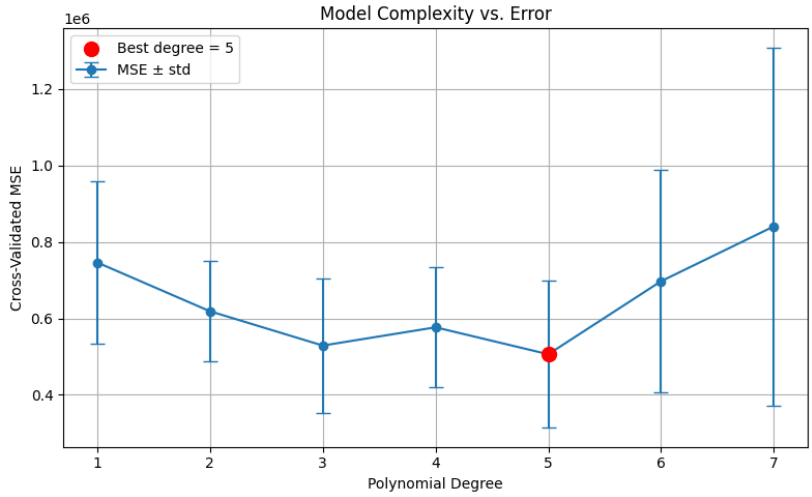

Figure 20 Model Complexity vs Error

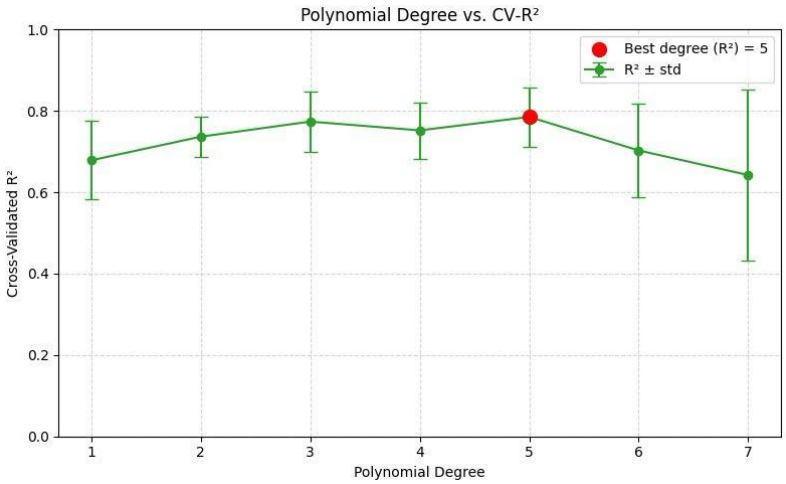

Figure 21 Polynomial Degree vs CV-R^2

Figures above shows the cross-validated mean squared error (MSE) for polynomial models of degree 1 through 7, with error bars indicating the standard deviation across folds. The red marker highlights degree 5, which achieved the lowest average MSE, confirming it as the most balanced model in terms of bias and variance. While it is often expected that increasing the degree of a polynomial will lead to lower error, this was not consistently observed. Beyond a certain point, higher-degree models began to overfit the training data, capturing noise rather than meaningful patterns—resulting in worse generalization performance. This behaviour confirms the importance of cross-validation in selecting a degree that balances accuracy and robustness.

To better evaluate and compare the accuracy of the emulator, since relying solely on MSE can be somewhat misleading, the chart also presents the R^2 values of each surrogate polynomial surface. As shown, the fifth-degree polynomial exhibits the highest performance and accuracy, with an R^2 of 0.785. This indicates a very good predictive capability of the surface model.

After identifying degree 5 as the optimal model based on cross-validated MSE, the dataset was reprocessed using a 5th-degree polynomial transformation, followed by training a new linear regression model.

The general form of a 5th-degree polynomial with two input variables(rotation (r) and L1 (L)) includes all terms up to the fifth power and their interactions. It can be expressed as:

$$f(r, L_1) = a_1 . r^5 + a_2 . r^4 L_1 + a_3 . r^3 {L_1}^2 + a_4 . r^2 {L_1}^3 + a_5 . r {L_1}^4 + {L_1}^5 + \cdots + a_{21}$$

where :

1. each a_i represents a learned coefficient for one of the polynomial terms, including pure powers (e.g., r^2, ${L_1}^3$) and mixed terms (e.g., $r {L_1}^4$, $r^3 {L_1}^2$). Using the Polynomial Features(degree=5) transformation, the model generates all 21 features automatically. These terms were then matched with their learned coefficients.

The resulting coefficients were used to construct the final equation:

$$f(r, L_1) = 177502.4349 - 21661.8742 \cdot r - 403.1559 \cdot L_1 + 144.61 \cdot r^2 + 3881.9884 \cdot r L_1 - 4124.0603 \cdot {L_1}^2 - 1.0456 . r^3 - 13.7902 \cdot r^2 L_1 - 271.1835 . r {L_1}^2 + 414.8053 \cdot {L_1}^3 + 0.0046 \cdot r^4 + 0.0523 \cdot r^3 L_1 + 0.4720 . r^2 {L_1}^2 + 8.6358 \cdot r {L_1}^3 - 15.6265 \cdot {L_1}^4 + -0.0001 \cdot r^4 L_1 - 0.0008 \cdot r^3 {L_1}^2 - 0.0056 \cdot r^2 {L_1}^3 - 0.1050 \cdot r {L_1}^4 + 0.2094.15$$

This expression represents the final emulator and allows fast, flexible prediction of solar irradiation for any combination of rr and ll within the trained design space.

For the 5th-degree emulator, the same visualization approach was adopted to evaluate prediction accuracy. A scatter plot was generated comparing predicted values to the actual simulation data, with the ±1000 kWh threshold retained from the 2nd-degree analysis for consistent comparison.

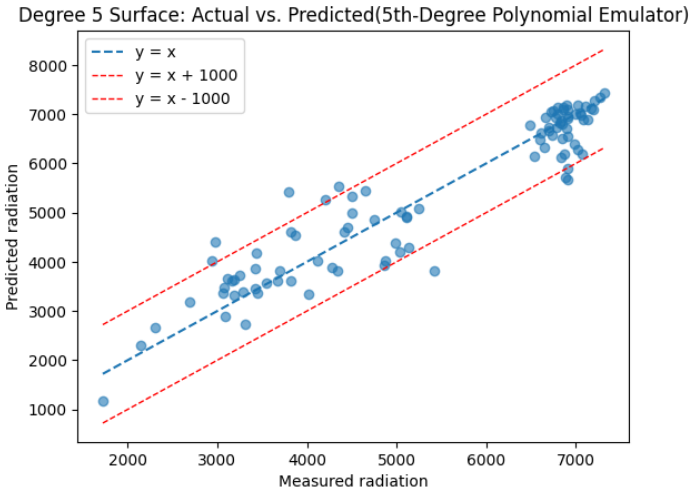

Figure 22 Degree 5 Surface: Actual vs Predicted Polynomial Emulator

The results show a clear improvement: a greater concentration of points falls within the defined error bounds, and a noticeably higher number of data points cluster tightly around the ideal line y=x. This visual trend indicates that the 5th-degree polynomial model offers higher predictive accuracy and more reliable performance across a broader range of irradiation values when compared to the 2nd-degree model. The

reduced dispersion and tighter alignment with the reference line validate the improved fit and confirm that the added complexity in the 5th-degree model effectively captures the underlying non-linear behaviour in the dataset.

To evaluate the predictive accuracy of the 5th-degree emulator, the sum of squared errors (SSE) was calculated and compared to both the baseline and the earlier 2nd-degree model. The initial SSE, corresponding to a baseline model that predicts the mean radiation value for all configurations, was 25,735,518.19 in both cases. After fitting the 5th-degree polynomial model, the optimized SSE dropped to 2,927,390.38, resulting in an 88.6% reduction.

This reduction is notably greater than that of the 2nd-degree model, which achieved an 80.2% reduction. While SSE reduction alone does not guarantee a significant improvement—since it may depend on the baseline's sensitivity or dataset characteristics—the fact that the initial SSE is identical across both models confirms a fair comparison. Therefore, the increased reduction achieved by the 5th-degree model is meaningful and reflects a true enhancement in prediction performance.

This result supports the decision to adopt the 5th-degree emulator, as it not only reduces error more effectively but also demonstrates a better fit to the complex, non-linear patterns observed in the solar irradiation response surface.

To gain a better understanding of the emulator's accuracy, a visualization of the absolute error was presented using a heatmap, following the same approach used for the 2nd-degree model. The heatmap illustrates the

absolute difference between the predicted and simulated irradiation values across the full design space, with respect to rotation angle and L1.

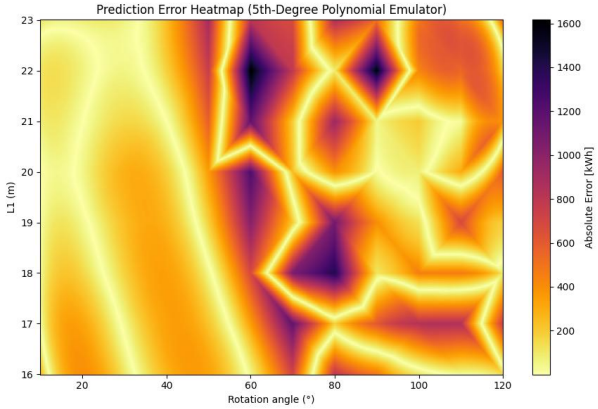

Figure 23 Prediction Error Heatmap Polynomial Emulator

The heatmap of the 5th-degree polynomial emulator confirms that the overall error has decreased compared to the 2nd-degree model, aligning with the SSE reduction from 80.2% to 88.6%. However, despite the improved overall accuracy, the visualization also reveals that in some regions—particularly at intermediate rotation angles and higher L1 values—the 5th-degree model exhibits greater localized error than the 2nd-degree version. Conversely, in lower rotation angle regions, a clear improvement can be observed, especially where sharp yellow contours appear, indicating zones of lower error.

This behaviour is a result of the increased flexibility of higher-degree polynomials, which, while better at capturing complex trends, can also lead to overfitting or exaggerated error in areas with sparse data. This reinforces the importance of not relying solely on global metrics like SSE, but also visual inspection of error distributions to evaluate model robustness.

Following the same approach used in the 2nd-degree emulator, a known configuration from the dataset (rotation = 20°and L1 = 20 m) was used to test the accuracy of the 5th-degree model. For this input, the emulator predicted a solar irradiation value of 6900.63 kWh, which is remarkably close to the actual measured value of 6909.38 kWh. This small deviation indicates strong predictive performance. Notably, the 2nd-degree model had previously predicted 6923.34 kWhfor the same configuration, showing a larger error. This comparison further confirms that the 5th-degree polynomial emulator provides better accuracy, particularly in matching known values from the simulation dataset.

To determine the configuration that yields the maximum predicted solar irradiation, a fine mesh grid was created over the design space, and the emulator was evaluated at each point to locate the global maximum. This procedure identified the optimal configuration for the 5th-degree model as rotation = 10.00° and L1 = 23.00 m, with a predicted radiation value of approximately 7435.3 kWh.

Interestingly, this optimal configuration is identical to that found using the 2nd-degree model, which also reported its maximum at r = 10°, L1 = 23 m. However, the key difference lies in the accuracy of the predicted values. The 2nd-degree model overestimated the radiation at this point, predicting 7865.7 kWh, while the 5th-degree model produced a value much closer to the actual measured irradiation of 7315.27 kWh from the simulation dataset. This comparison confirms that, although both models identified the same optimal geometry, the 5th-degree polynomial emulator provides a more accurate approximation of the true solar performance.

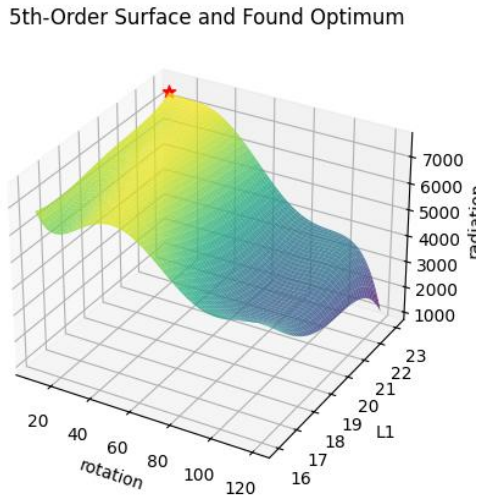

Figure 24 Order Surface and Found Optimum

3.4. Optimization

3.4.1. Objective Function

To set up our optimization, we first import SciPy's general-purpose minimizer and then define an objective function that quantifies how far our surrogate's prediction deviates from the target 6000 kWh:

```
from scipy.optimize import minimize

def obj_to_target(x):
    """
    x[0] = rotation angle r, x[1] = window length L1.
    Returns (predicted_radiation - 6000)^2.
    """
    r, L1 = x
    # evaluate the surrogate model at (r, L1)
    val = surface5(np.array([[r]]), np.array([[L1]]))[0, 0]
    # return the squared error relative to our 6000 kWh goal
    return (val - 6000.0)**2
```

In this code, the function obj_to_target receives a two-element vector `x`, where `x[0]` is the rotation angle and `x[1]` is the panel length. These values are passed into the previously trained fifth-degree surrogate model (`surface5`), which returns a predicted irradiation. By computing the squared difference between this prediction and our desired 6000 W/m^2, the function produces a single scalar error measure.

3.4.2. SciPy's minimize

To initialize the search, we begin from the previously identified global maximum (r_{opt}, L_{opt})and enforce the same practical variable limits. We then call SciPy's minimize with the Powell algorithm, specifying tolerances for both the design variables (xtol) and the objective function (ftol) to control convergence. Here is the annotated code used in the report:

```
# start from the global maximum found earlier (or any feasible point)
x0 = np.array([r_opt, L_opt])

bounds = [(10.0, 120.0),  # rotation angle must lie between 10° and 120°
          (16.0,  23.0)]  # L1 length must lie between 16 m and 23 m

# Perform the optimization using the Powell method
result_target = minimize(
    fun=obj_to_target,
    x0=x0,                  # initial guess for [r, L1]
    method="Powell",        # derivative-free optimization algorithm
    bounds=bounds,          # enforce variable limits
    options={"xtol": 1e-3, "ftol": 1e-3, "disp": True}  # tolerance for changes
in r, L1 and the objective value
)
```

In practice, this setup ensures that the optimizer explores only realistic configurations and stops once changes in both the decision variables and the error measure fall below

the specified thresholds. The result target object then contains the optimal $(r*, L*)$, the corresponding objective value, and a flag indicating successful convergence.

3.4.3. Reporting the results

To conclude the optimization and report the results, we check whether the solver converged successfully. If it did, we extract the optimal rotation and length, compute the predicted radiation at that point, and print a concise summary. If not, we display an error message. The annotated code and its explanation are provided below:

```
# Check if the optimizer converged successfully
if result_target.success:
    # Extract the optimal rotation angle and L1 length
    r_target, L1_target = result_target.x

    # Evaluate the surrogate model at the optimal point
    rad_target = surface5(
        np.array([[r_target]]),
        np.array([[L1_target]])
    )[0, 0]

    # Print the results in a readable format
    print(f"Converged: rotation = {r_target:.2f}°, L1 = {L1_target:.2f}")
    print(f"  Predicted radiation = {rad_target:.2f} kWh")
    print(f"  Error squared      = {obj_to_target([r_target, L1_target]):.4f}")
else:
    # Report failure and the solver's message
    print("Optimization failed:", result_target.message)
```

After running the optimization, the code first checks result_target.success, a Boolean flag set by SciPy that indicates whether the Powell algorithm met its convergence criteria. If the flag is True, the array result_target.x contains the optimized rotation angle (r_target) and length (L1_target). The surrogate model surface5 is then evaluated once more at this optimal point to obtain rad_target, the predicted irradiation. The script prints a readable summary showing the converged values of r_target and L1_target, the corresponding radiation, and the squared error computed by calling obj_to_target again for consistency.

If the solver did not converge successfully, the else branch prints "Optimization failed:" followed by result_target.message, which contains details about why the optimization terminated.

The result of the following code is presented below:

```
Converged: rotation = 53.15°, L1 = 22.13
Predicted radiation = 6000.00 kWh
Error squared      = 0.0000
```

The optimization converged to a rotation angle of 53.15° and an L_1 length of 22.13 m, yielding a predicted irradiation of 6000 kWh with a near-zero squared error. This result demonstrates that the target irradiation of 6000 kWh can be achieved precisely by applying the calculated r and L1; in other words, the two design variables have been optimized so that our fifth-degree equation (polynomial surrogate) result a value as close as possible to target irradiation, applying those two values for r and L.

It should be noted that such a low sum of squared errors (SSE) is not guaranteed for arbitrary combinations of r and $L1$. The model's accuracy can be further assessed by comparing its prediction at a fixed existing (r, L) in the dataset and its real irradiation (as in the emulator validation stage, one may compute the SSE between the surrogate's prediction and the true irradiation value). A worked example of this comparison is provided at the end of this section.

Finally, the convergence flag returned by SciPy's Powell algorithm confirms that both the adjustments in the decision variables and the change in the objective function fell below the specified tolerances, ensuring that the solution is numerically reliable.

3.4.4. 3D Visualization

This 3D heat-map combines the fitted polynomial surface and a color-coded error metric to reveal immediately both the overall shape of the radiation response and the precise "ridge" where the model prediction gets close as possible to 6000 kWh indicating: $min\ (\hat{f}(r, L) - \ 6000)^{\ 2}$

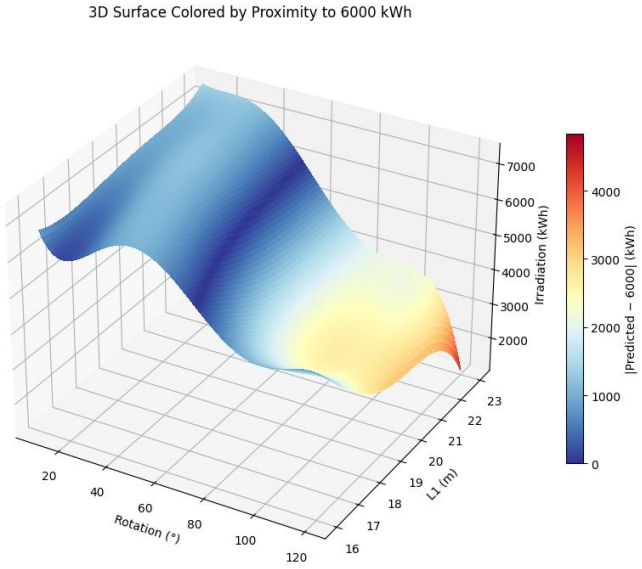

Figure 25 3D Surface Colored by Proximity to 6000 kWh

3.4.5. Top-Down Error Map contour plots

Additionally, the Top-Down Error Map contour plots illustrate how combinations of rotation r and length $L1$ affect the error: $E(r, L) = (\hat{f}(r, L) - \ 6000)^{\ 2}$.

In these diagrams, regions shaded in blue correspond to low error—i.e., $(r, L1)$ pairs whose surrogate prediction lies very close to the target 6000 kWh—while red-tinted areas indicate high error, where the model's output deviates more substantially from the desired value. These visualizations provide a clear, intuitive overview of the solution and the sensitivity of the predicted irradiation to changes in each design variable.

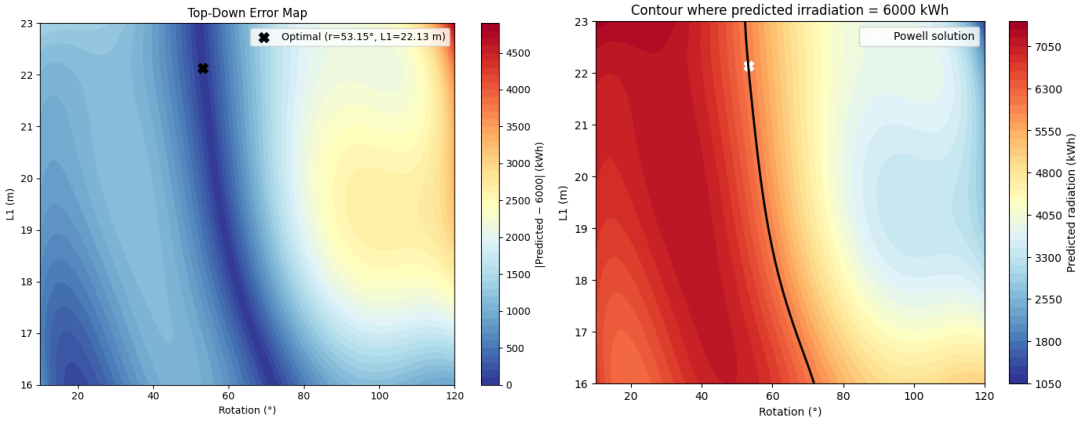

Figure 26 Top Down Error Map | Contour Predicted Irradiation

3.4.6. Critical Evaluation of Optimization Outcomes

A close inspection of the Top-Down Error Map indeed suggests the presence of several "blue" basins—regions where the squared-error falls below our tolerance and the surrogate prediction comes very close to 6000 kWh. In principle, each of these basins corresponds to a different $(r, L1)$ pair that would satisfy our target almost equally well. Yet, in every run we observe the same unique solution. So overall the discussion can be concluded in this way:

Why does the numerical optimizer return only one $(r, L1)$ pair, even though infinitely many pairs satisfy $\hat{f}(r, L) \cong 6000$) ?

When we solve:

$$\min_{r,L1} (\hat{f}(r, L) - 6000)^2$$

analytically, the solution set is the entire contour:

$$\{(r, L1) \mid \hat{f}(r, L1) \cong 6000)\}$$

a one-dimensional curve in the $(r, L1)$ plane. Yet, practically, when we call a routine like

```
result = minimize(lambda x: (f(x)-6000)**2, x0, method="Powell", …)
```

it reports one optimal pair. This arises due to several reasons:

1. The Algorithm Delivers a Single Arg min

Numerical solvers are built to return *an* $argmin$, not the full solution set. Their interface and stopping logic are designed around finding a single best point where the objective is minimal.

2. Local Convergence of the Powell Algorithm:

   Powell's method is a local, derivative-free optimizer: starting from the same initial guess (in our case, the previously found global maximum) and following the same search directions, it consistently converges into the same nearest basin. If another low-error basin lies "farther" in the parameter space or separated by a ridge of higher error, the algorithm will never cross that ridge under its standard stopping criteria. Hence, although multiple acceptable solutions exist mathematically, the solver returns the one closest to its initialization path.

3. Finite Tolerances and Early Stopping

   Real-world optimizers use convergence tolerances ($xtol, ftol$) to decide when the objective is "close enough" to zero. As soon as the squared error dips below these thresholds, the algorithm halts and returns that single pair—even though other points on the contour would also drive the error to (near) zero.

Moreover, although several regions may reach errors on the order of 10^{-6} or below, they are not perfectly equivalent in the surrogate's surface. Tiny differences—amplified by our strict tolerances ($xtol$=10^{-3}, $ftol$=10^{-3})—break the symmetry and lead to a uniquely minimal square-error. In other words, what appears visually as several equally "blue" areas may, at machine precision, differ very slightly in error value. The optimizer will favor the absolute minimum, however small the advantage.

3.4.7. Variable's Analysis

The plots below illustrate how the sum of squared errors (SSE)from 6000 changes when either r or $L1$ is held at its optimized value, demonstrating why those settings minimize the error.

1. Fixing $L1$ at the optimized value of 22.13 and varying the r from 10-120 degree, displaying how the SSE from 6000 is changing applying different value for r.

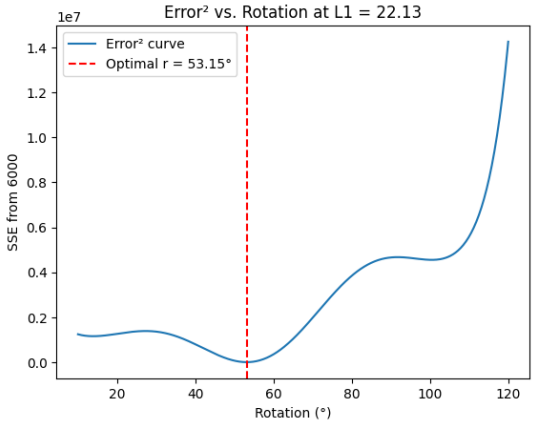

Figure 27 Error vs Rotation at L1 =22.13

2. Fixing r at the optimized value of 53.15 and varying the L from 16-23 degree, displaying how the SSE from 6000 is changing applying different value for L.

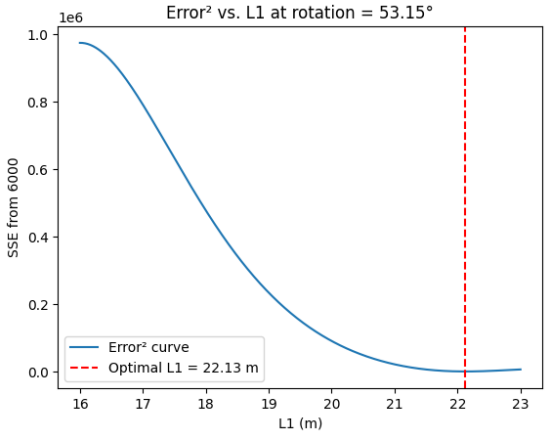

Figure 28 Error vs L1 at Rotation = 53.15 Degrees

In the final code block, the user is prompted to enter a desired rotation angle and L_1 length within the predefined bounds. The script then computes two values: the irradiation predicted by the fifth-degree polynomial surrogate model and the actual irradiation from the dataset entry whose rotation and L_1 values most closely match the user's inputs.

3.4.8. Conclusion

In summary, by employing the squared-error cost function together with our fifth-degree polynomial surrogate model—originally calibrated in the emulator phase—and by leveraging SciPy's derivative-free Powell algorithm, we have successfully determined the rotation angle "r" and L_1 length that drive the model output to the target irradiation of 6000 kWh.

$$E(r, L) = (\hat{f}(r, L) - 6000)^2$$

Specifically, the objective function quantifies the deviation from the desired value; using SciPy's `minimize` function within the practical bounds of $10° \leq r \leq 120°$ and $16m \leq L1 \leq 23m$ guided the optimizer toward the solution. At convergence, the Powell method reported negligible error, confirming that the surrogate prediction matches the specified point with high precision. This approach demonstrates how a carefully constructed polynomial emulator, when coupled with an appropriately defined cost function and a robust optimization routine, can provide clear, actionable design settings that achieve prescribed performance targets without exhaustive brute-force searches.

3.5. Optimized Floor Plan Design

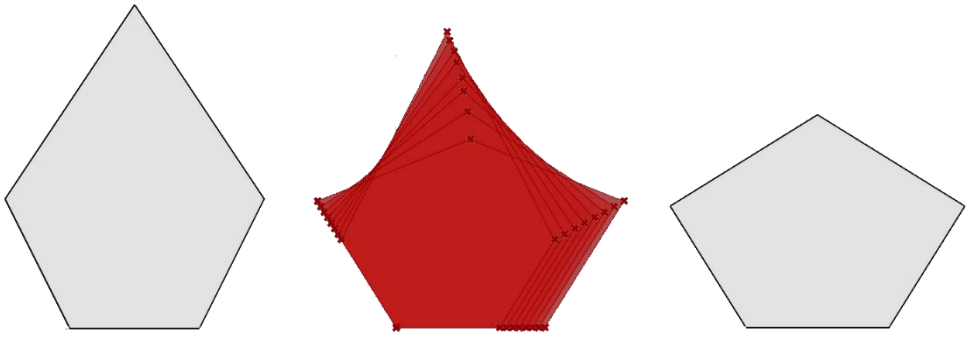

Figure 29 Optimized Floor Plan Design

Figure above shows the geometric evolution of the pentagonal floor plan throughout the full optimization process. The shape on the left represents the initial configuration, a regular five-sided polygon with equal side lengths and no rotation applied. This base geometry served as the starting point for our parametric analysis. Using Grasshopper, we generated a wide range of alternative configurations by systematically adjusting the proportions between the two alternating sides, L1 and L2, while maintaining a fixed total perimeter. The red form in the centre visualizes the collection of all these tested variations, capturing the full design space explored during the parametric modelling and simulation phase.

Each of these alternatives was evaluated using Ladybug Tools to simulate the total annual solar irradiation on the façade. The resulting dataset formed the basis for training a polynomial emulator model, which allowed us to interpolate values and estimate solar performance for unseen configurations with high computational efficiency. Based on this emulator, we then implemented a Python-based optimization routine using SciPy's Powell method, aiming to identify the specific configuration that could either maximize solar exposure or reach a defined solar target. The shape on the right represents the final optimized floor plan, which was selected because it achieved a predicted annual irradiation of 6000 kWh with minimal error according to the surrogate model.

This visualization effectively summarizes the complete methodological pipeline. It begins with a clean base form, expands into a rich set of parametric variations evaluated through simulation and modelling, and culminates in a data-driven, performance-optimized geometry. The comparison clearly highlights the geometric differences between the original design and the final solution, emphasizing how even subtle shifts in side proportions and orientation can significantly influence solar performance outcomes.

4. Discussion

The results of this study reveal important insights into how twisted tower geometry can be optimized to improve solar irradiation performance. The use of a parametric model

and emulator-based analysis has enabled a comprehensive evaluation of how geometric parameters—specifically, floor rotation and L1 length—influence solar exposure under the environmental conditions of Malmö.

4.1. Significance of the Optimized Solutions

The optimized solutions align directly with the project's goal of achieving targeted solar performance through a data-driven parametric approach. The best-performing configuration, identified at a rotation angle of approximately 10° and L1 = 23 meters, confirms that minimizing floor rotation while maximizing floor length enhances solar irradiation. However, the flexibility of the optimization framework also allowed us to redefine our objective—to find a design configuration that achieves a specific target radiation value (6000 kWh)—rather than only the global maximum. This made the design process more adaptable to real-world performance needs, such as matching energy supply with expected building demand.

The ability to use a fifth-degree polynomial emulator model to reach this target irradiation with high accuracy (SSE near zero) demonstrates the robustness of the optimization strategy. Importantly, it provides architects and engineers with a tool that supports precision design without relying on exhaustive simulation loops.

4.2. Limitations and Potential Improvements

Despite the high overall accuracy and 88.6% SSE reduction achieved with the fifth-degree model, limitations remain. The emulator shows local inaccuracies in more complex regions of the design space, particularly between 60°–90° rotation and mid-range L1 values. These deviations suggest that while the fifth-degree model offers significant improvements over simpler polynomials, it still struggles with sharp non-linear behaviour in certain configurations.

Another limitation involves data sparsity. With only 96 simulation points for original database, the emulator may underperform in regions not well represented in the training data. Future work could benefit from:

- Denser sampling in high-gradient zones.
- Testing alternative machine learning models (e.g., Gaussian processes or neural networks) for smoother generalization.
- Considering more environmental variables like seasonal variation or incident angles from multiple orientations.

4.3. Innovation and Broader Implications

This study introduced a novel and generalizable approach for the analysis of high-rise buildings with rotational geometries. The parametric definition of the building, coupled with integration into a dedicated online platform, enables users to adapt and assess designs by entering key parameters directly. This approach eliminates the need for

advanced modelling expertise while providing quick and reliable estimations of building surface area and incident solar irradiation.

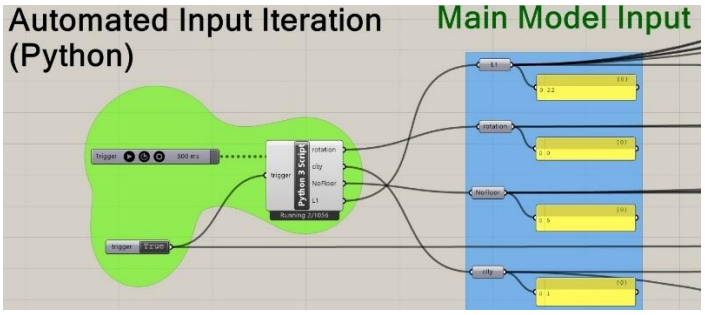

Figure 30 Automated Input Iteration

One of the most significant aspects of this innovation lies in its flexibility. The workflow allows the use of any EPW climate file for any location worldwide, with the ability to adjust building dimensions, number of floors, and other characteristics to match site-specific constraints, budgetary considerations, and environmental conditions. As a result, the method can be applied to a broad spectrum of towers that rotate around a central axis. By adjusting a small set of fixed parameters, the perimeter and proportions can be modified, effectively turning the parametric model into a generalized template for similar typologies. The online platform acts as an accessible interface, lowering the barrier to conducting such analyses.

In addition to focusing on the case study in Malmö, we also conducted a comparative radiation analysis using EPW files from four different cities with varying latitudes and climates. This allowed us to test the model's adaptability and to evaluate how solar irradiation patterns and optimal design parameters change depending on environmental context. The ability to switch between cities and re-run the analysis using the same computational framework demonstrates the geographic flexibility of the system. These comparisons revealed that while the optimal rotation and geometry vary by location, the workflow remains valid and scalable. This means that the method developed here is not limited to a single building or climate but can serve as an adaptable design tool for other high-rise projects globally, particularly those with complex or responsive geometries. By combining EPW-based climate responsiveness with a lightweight optimization model, the project sets a precedent for integrating site-specific environmental data into early-form generation strategies.

In the case study presented here, optimization targeted the real building location in Malmö, with the objective of achieving a total incident irradiation of 6000 kWh on 21 December, the day with the lowest annual solar availability. The base floor geometry and rotation were adjusted to meet this requirement while maintaining the original number of floors. However, the availability of a more comprehensive dataset now makes it possible to conduct equivalent optimization processes for other sites by simply replacing the CSV database file used in the previous iterations.

It is important to distinguish between the Visual Studio emulator, developed for fitting polynomial surfaces to three-dimensional datasets, and the Grasshopper-based iteration process used to automate model parameter assignment and data collection. While both processes involve forms of emulation, they serve different purposes within the workflow.

The development of three dedicated Python scripts—covering automated geometry generation, parameter iteration, and CSV-based result storage—represents a further step toward efficiency and reproducibility. Additionally, contextual modelling was incorporated into the climate analysis using Cadmapper and Google Maps data to represent surrounding buildings with accurate heights. Although their influence on the results was minimal due to the tower's scale, their inclusion ensured greater analytical realism.

To extend the flexibility of the parametric workflow, a user-friendly interface was developed using Python. This platform allows users to select the city (e.g., Barcelona, Malmö, Milan), floor shape (pentagon, square, hexagon), and geometric inputs like L1 and rotation angle. The application instantly calculates the expected irradiation, making the optimization process accessible without needing to open Rhino-Grasshopper or run simulations manually.

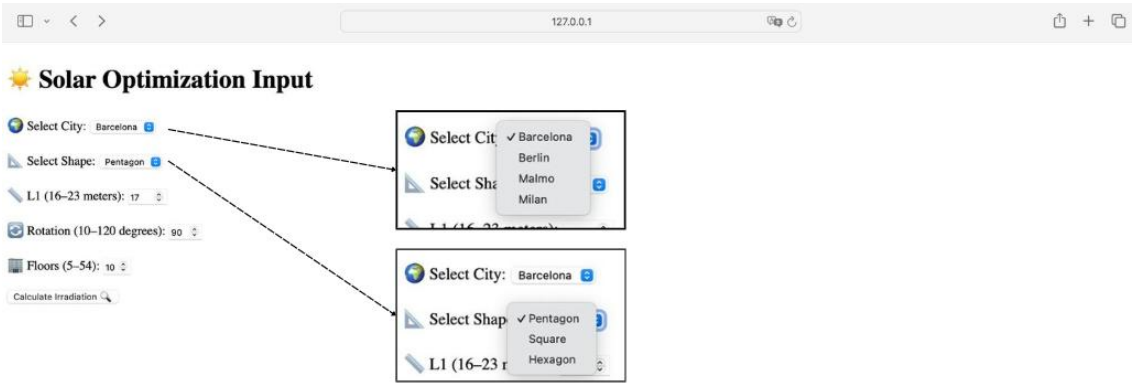

Figure 31 Website - Parametric Model Creation in Rhino-Grasshopper

Overall, this combination of parametric flexibility, automation, and user accessibility positions the workflow as a robust tool not only for this case study but also for other high-rise projects with similar geometrical and environmental characteristics. Its ability to balance technical accuracy with ease of use expands its potential applicability in both academic research and practical design processes.

4.4. Practical Applications and Future Impacts

This study offers a clear demonstration of how scientific computing can be integrated into the early design stages of environmentally responsive architecture. The emulator

model not only reduces simulation time but supports exploratory design, where architects can adjust geometry based on target solar performance goals.

Practically, this research supports the development of twisted towers optimized for northern climates—where low solar angles and diffuse radiation dominate. The methodology can be applied to other locations by simply changing the EPW file and recalibrating the emulator. This approach also lays the groundwork for integrating multi-objective optimization, where solar gain is balanced with other performance factors such as structural efficiency or daylight autonomy.

In the context of growing demand for sustainable high-rise design, our findings contribute to a deeper understanding of how form can respond to performance—not just symbolically, but quantitatively.

4.5. Originality and Courses Framework

A key aspect that distinguishes this project is the way the foundational workflow introduced in the course was not only implemented but also critically adapted and extended. While the lecture content provided a general Grasshopper-based modeling method for tower geometry and radiation analysis, our group introduced a number of original simplifications and enhancements that made the process more flexible, scalable, and data-driven.

The original Grasshopper script was redesigned to allow dynamic control over fewer parameters, specifically the L1 edge length and floor rotation, which greatly reduced complexity without sacrificing analytical depth. Instead of manually adjusting each geometry iteration, we built an automated pipeline that allowed for continuous variation across a large design space, linking geometry creation, simulation, and data export in a single loop. This not only streamlined the process but made it possible to test and compare nearly 100 unique configurations, which would have been impractical through manual operations alone.

Furthermore, our contribution lies in the transition from a purely visual parametric setup to a fully integrated Python-assisted system, which introduced emulation, error minimization, and automated optimization routines. While the lectures introduced environmental analysis using Ladybug, this project advanced those ideas by creating a predictive emulator model and coupling it with optimization logic to target specific energy goals, such as the 6000 kWh irradiation target.

In essence, this project not only demonstrates mastery of the tools provided in the course but also expands upon them to propose a generalizable and original framework that combines visual programming (Grasshopper), simulation (Ladybug), statistical modelling (polynomial regression), and algorithmic optimization (Python and SciPy). The result is a hybrid workflow that reflects both the spirit and the ambition of the course, while delivering a practical and reusable tool for real-world design contexts.

5. Conclusion

This project set out to explore how parametric geometry and computational modelling could be used to optimize solar irradiation in a twisted high-rise tower, using the Turning Torso in Malmö as a case study. Through simulation and emulator modelling, we investigated how floor rotation and geometric proportions (L1) affect annual solar gain.

The study demonstrated that lower rotation angles (especially around 10°) combined with higher L1 values (up to 23 m) lead to the most favourable solar performance. Furthermore, the application of a fifth-degree polynomial emulator proved highly effective in approximating solar radiation values, achieving an SSE reduction of 88.6%, and enabling precise design optimization without running hundreds of new simulations.

The most significant finding was the ability to target a specific irradiation value (6000 kWh) and identify the exact design configuration to achieve it. This flexibility underscores the value of integrating emulator modelling into early-stage design and optimization workflows.

While the emulator performed well overall, local inaccuracies in complex geometric zones indicate that future work should include more training data or alternative modelling techniques to improve robustness. Nevertheless, the combination of parametric modelling, simulation, and emulator-based optimization shown in this project offers a powerful workflow for performance-driven architectural design.

In conclusion, this research not only provides actionable insights for designing twisted towers in low-sunlight climates but also contributes to the broader discourse on how data and computation can meaningfully shape architectural form.

6. References

[1] Klimczak, M., & Bieniek, M. (2018). Solar radiation analysis as a basis for parametric design of building façades. E3S Web of Conferences, 49, 00062. https://doi.org/10.1051/e3sconf/20184900062

[2] Salama, A. M., & Gadelhak, M. (2020). Design optimization of twisted towers for solar performance in hot arid climates. Journal of Building Engineering, 31, 101356. https://doi.org/10.1016/j.jobe.2020.101356

[3] Ladybug Tools. (n.d.). Environmental design and simulation tools for Grasshopper. Retrieved April 2025, from https://www.ladybug.tools

[4] Calatrava, S. (n.d.). Turning Torso – Project Overview. Calatrava Architects & Engineers. Retrieved April 2025, from https://calatrava.com/projects/turning-torso-malmoe.html

[5] Woodbury, R. (2010). Elements of parametric design. Routledge.

[6] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html