SCIENTIFIC COMPUTING FOR BUILDING ENGINEERING

Final Presentation G14
01 September 2025

POLITECNICO DI MILANO – SCIENTIFIC COMPUTING FOR BUILDING ENGINEERING
G14: FARNAZ VAFA – JACOPO GUERRINI – MOHAMMAD SHOJAEE – SALMA YANUAR

Team 14

Farnaz Ahmadi Vafa (FV)
Civil Engineer

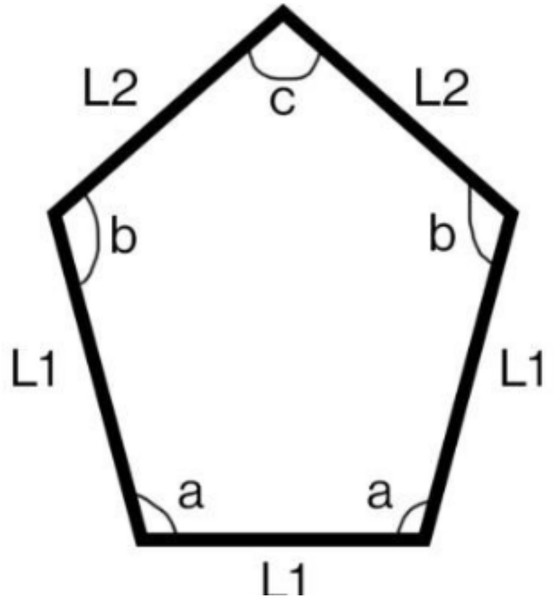
Mohammed Amin Shojaee (MS)
Civil Engineer

Salma Nabilah Yanuar (SY)
Architect

Turning Torso

This project focuses on optimizing the solar performance of twisted high-rise towers. Using the Turning Torso in Malmö as a case study, we applied parametric design, environmental simulation, and emulator modeling tooptimize the rotation and L1 in order to get close as possible to the design Value of 6000 kWh

Background and Problem



- Twisted towers = unique solar patterns.
- Fixed perimeter floor plans are common in real design.
- Gap: No combined analysis of rotation + proportions.
- Turning Torso is an ideal test case.

Twisted towers interact with sunlight differently from straight towers. In real projects, perimeter constraints often limit design freedom. Previous research hasn't fully explored how floor rotation and geometric proportions interact to affect solar gain. Turning Torso gives us a real-world, iconic example for testing.

Objectives

TEST IMPACT OF FLOOR ROTATION AND PROPORTIONS

OPTIMIZE FOR 6000 KWH ON 21 DEC (WORST DAY)

COMBINE SIMULATION + EMULATOR FOR EFFICIENCY

CREATE A FLEXIBLE WORKFLOW FOR OTHER TOWERS

Our goals were to understand how geometry affects solar performance, optimize the design for a specific energy target, and build a workflow that can be reused for other twisted towers.

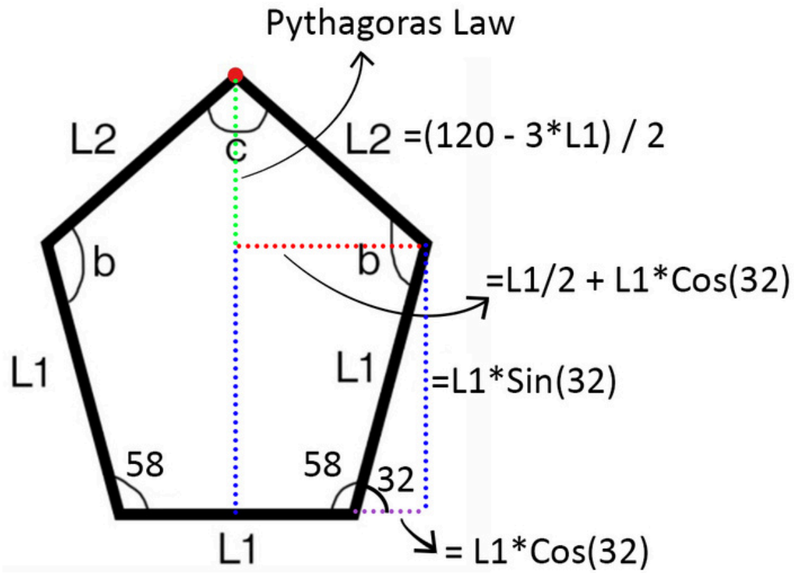
Methodology Overview

1. PARAMETRIC MODELLING IN GRASSHOPPER
2. SIMULATION WITH LADYBUG TOOLS (MALMÖ EPW)
3. EMULATOR IN PYTHON
4. OPTIMIZATION WITH POWELL METHOD AND MINIMIZE FUNCTION
5. AUTOMATED GEOMETRY CONTROL AND DATA COLLECTION

We built a fully parametric model in Grasshopper, used Ladybug Tools with Malmö climate data for simulations, created a polynomial emulator in Python to speed up predictions, and ran an optimization algorithm to find the best design.

1 PARAMETRIC MODELLING IN GRASSHOPPER

- A single floor plan was abstracted as a pentagon with alternating sides L1 and L2.
- L1 was manually varied, L2 was auto-calculated to keep perimeter P constant.
- Floor rotation (θ) was applied to simulate twisting across tower height.
- Internal angles (b, c) updated automatically to maintain valid geometry.



L1 values (meters):
16, 17, 18, 19, 20, 21, 22, 23
Rotation angles (°):
10 to 120 (step of 10°)
Total configurations tested:
96 (in original case study)

Model built in Grasshopper for Rhino; phyton code controlled L1 and rotation angle to generate valid and diverse floor geometries for solar simulation.

2 FORMALIZATION OF THE ADDRESSED PROBLEM

- The goal is to find the best L1 and rotation (θ) that near to to the target value of solar irradiation on the façade.
- The floor is a five-sided polygon with alternating sides L1 and L2.
- L2 is calculated to keep perimeter constant using:

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

- Solar irradiation is represented as $I(L_1, \theta)$, estimated using emulator.
- The optimization domain is:
  - $L_1 \in [16, 23]$ meters
  - $\theta \in [10°, 120°]$
- Initial Optimization Objective:

$$Maximize\ I(L_1, \theta)$$

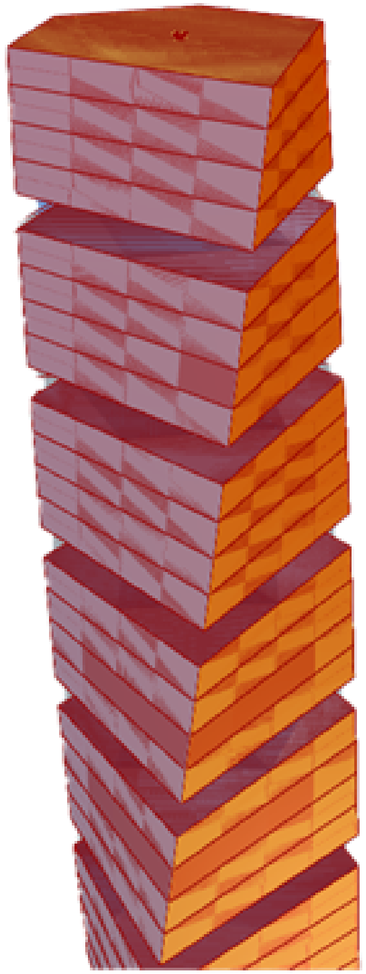
- Loss Function (for emulator training):

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

- Optimization type:
  - Simulation-based with emulator approximation
- Total configurations simulated: 96

This problem is approached as a black-box optimization task where L1 and θ are inputs. Solar irradiation is predicted using a trained emulator, and the best combination is selected based on the minimum error.

3 EMULATOR IN PYTHON



- Emulator replaces slow simulations with fast prediction.
- Trained on 96 configurations (L1, θ) with Ladybug results.
- Predicts total solar irradiation (including walls, roof, and floor).
- Key inputs:
  - L1 = pentagon edge (m)
  - θ = floor rotation (°)

THE MAIN PURPOSE

to find a polynomial surface (which has an equation) that fitsthe best to the original dataset so the values between the original dataset that was not calulated can be predicted and interpolated with the higher precision and ther is no need for iterating and modeling the builidng for all the values

Purpose

- Fast prediction for new L1, θ combinations
- Design exploration without re-running simulations
- Speed up optimization routines
- Find the best polynominal that fits well based on the original data

The emulator bridges parametric modeling, simulation, and data-driven prediction, supporting real-time performance evaluation.

4 OPTIMIZATION WITH POWELL ALGORITHM

- Goal: Find L1 and θ that get close to 6000 kWh on Dec 21
- Polynomial surface from emulator used in optimization
- Error function:

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

Solver: Powell's algorithm (SciPy)

- L1 ∈ [16, 23] meters
- θ ∈ [10°, 120°]

Instead of maximizing a global peak, the optimizer finds the closest match to the required energy target, improving design realism

5 AUTOMATED GEOMETRY CONTROL AND DATA COLLECTION

- Python replaced manual Grasshopper inputs
- L1 and top vertex coordinates calculated using math formulas
- Geometry auto-updates to satisfy fixed perimeter
- α kept fixed; L2 recalculated from L1
- Script 1:
  - Controls L1, θ, floors, and location
  - Run and control by a trigger
  - Iterates over full range with delay
- Script 2:
  - Collects total incident irradiation + surface area
  - Stores full dataset in CSV (6 values per row)

Python scripting reduced manual work, ensured consistency, and enabled large-scale testing of geometric and environmental combinations.

Results : Parametric Modelling

1

Generate Tower Geometry

- Use the Series component to define Z-heights for each floor (every 4 meters).
- Apply Move to vertically stack identical pentagonal floor plates.

2

Apply Progressive Floor Rotation

- Multiply base angle θ by floor index to simulate twist.

$$\theta_{\text{floor}} = \theta \times \text{floor index}$$

- Use Rotate and DegToRad to apply rotation

3

Create Tower Shell

- Apply Loft to connect all floors into one continuous surface.
- Use Cap Holes to close the geometry, forming a solid for analysis.

4

Calculate Surface Area

- Use the Area component to calculate each face.
- Apply Mass Addition to sum values and get the total façade surface area.

5

Simulate Solar Irradiation

- Import EPW climate data (Malmö) via Ladybug.
- Convert geometry to mesh for per-face solar evaluation.
- Visualize cumulative irradiation using Ladybug irradiation analysis, with color mapping

Results : Radiation Analysis

Performance Insight Through Climate-Based Simulation

1 Climate Setup

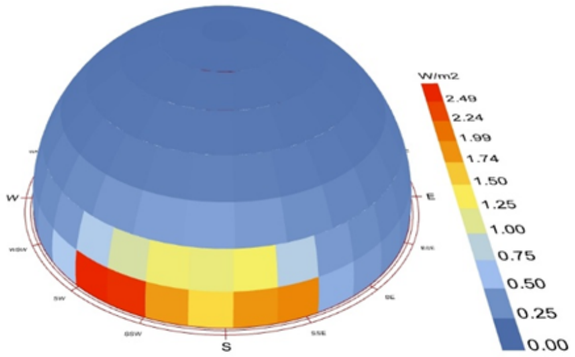
- Used EPW weather data for Malmö
- Simulated worst-case solar day: 21 December
- Target: 6000 kWh irradiation to meet energy needs year-round

2 Solar Context & Temperature

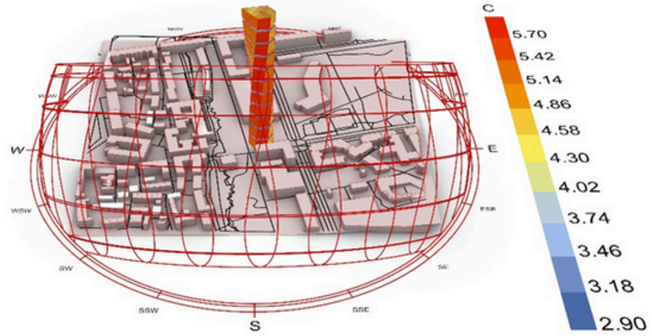
- Analyzed dry bulb temperature distribution
- Coldest zones: North/Northeast
- Warmest exposure: South-facing

3 Radiation Contour Plot

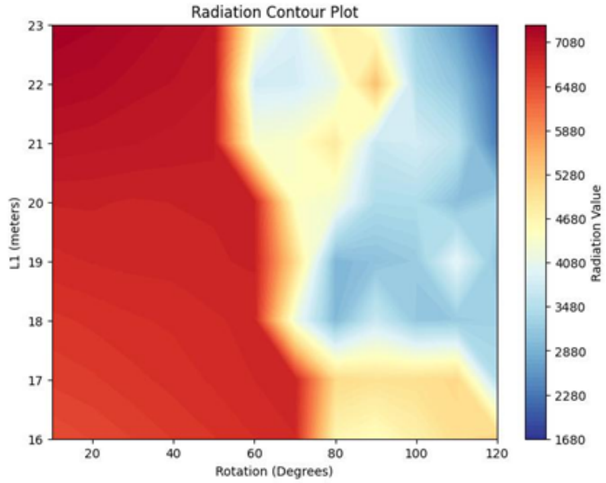
- Highest irradiation at low rotation (10°–50°) and longer L1
- L1 > 20 m gives consistently higher values



Total Irradiance
21 Dec 01:00 - 21 Dec 23:00
country : SWE
city : Malmo
time-zone : 1.0
source : SRC-TMYx



Dry Bulb Temperature (C)
country: SWE
city: Malmo
time-zone: 1.0
source: SRC-TMYx



4 Rotation Sensitivity (L1 = 23 m)

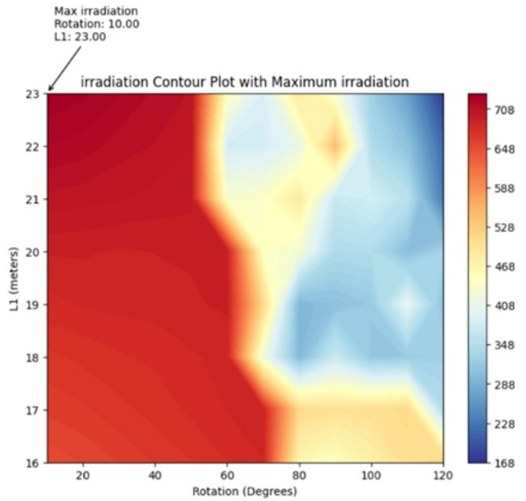
- Sharp drop in performance after 50° rotation
- Confirms self-shading and misalignment at high angles

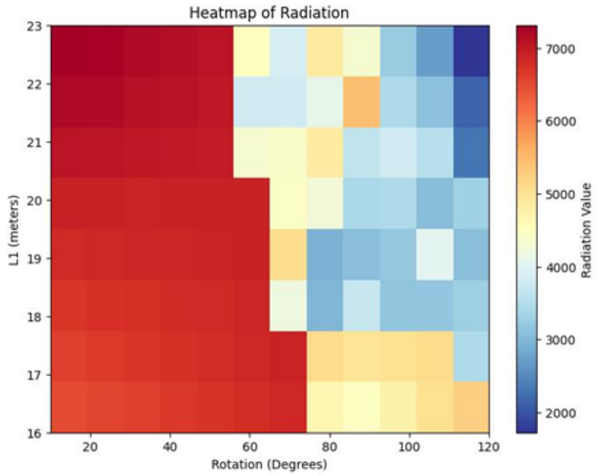
5 Heatmap Visualization

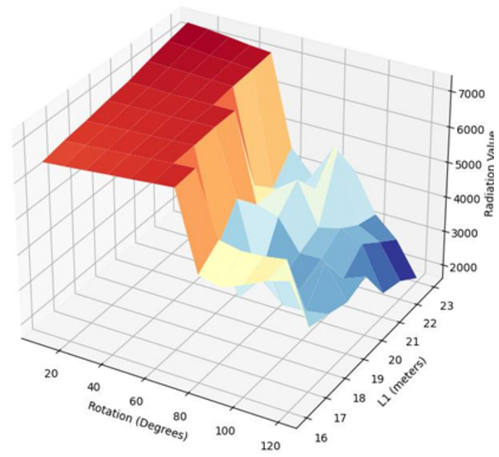
- Visualizes performance gradient
- Red zone (top-left) = best config (low rotation, high L1)
- Steep drop-off beyond 60°

6 3D Surface Plot

- Clear peak at L1 = 23 m, θ = 10°
- Decline along rotation axis = high impact of twisting







Rotation impacts solar gain more than L1 **Best config:** L1 = 23 m, θ = 10°

Performance-focused geometry requires early parametric evaluation

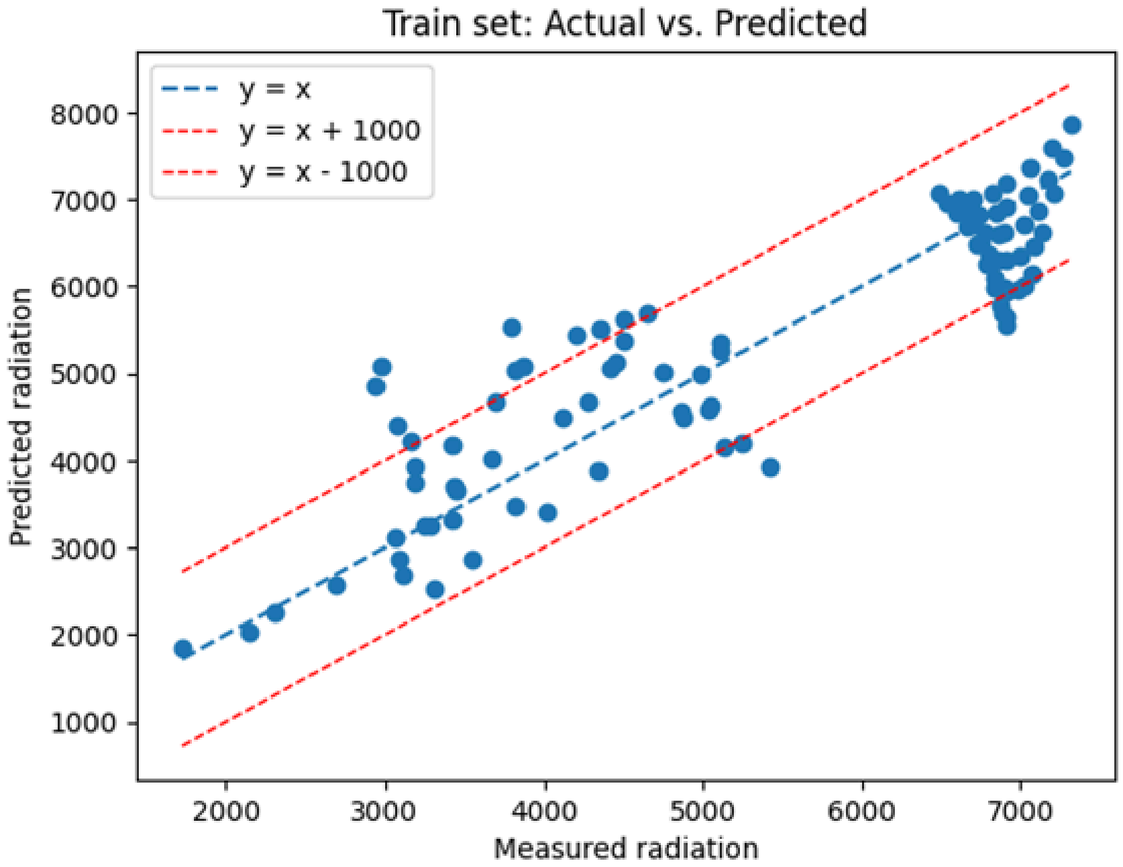
Results : Emulator Model Development

From Simulation to Prediction

1

Polynomial Regression Setup (second degree)

1. Starting from the simpleased surface equation with 2 degree
2. valuate the error between the original data and the corrisponding point on polynomial surface
3. Try to reduce this error by using higher degree surfaces; this, was done by evaluating the R2 and MSE of the model and org. dataset
4. At the end by cross validation we found the the equation degree that has the highest R2 and and lowest MSE



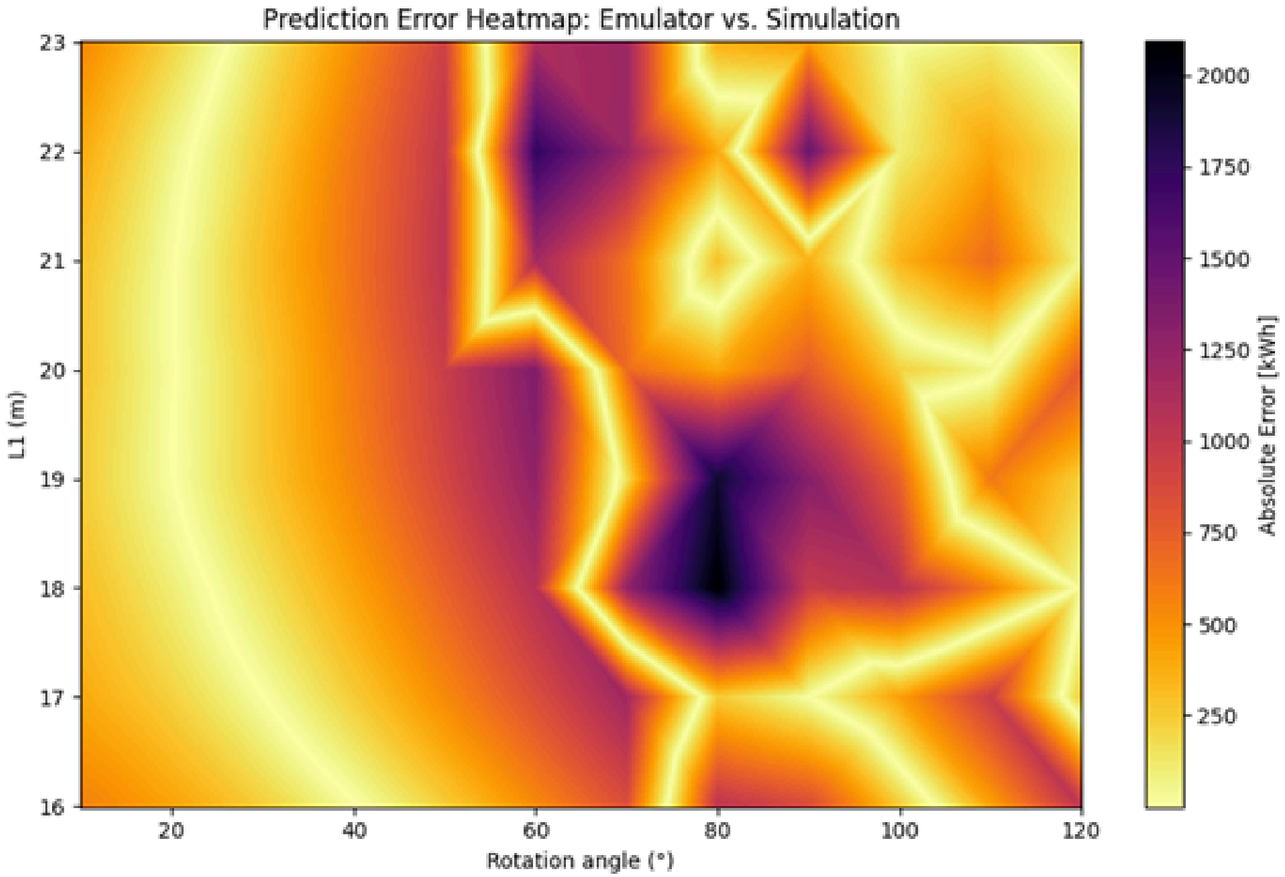
Results : Emulator Model Development

From Simulation to Prediction

2

Evaluate 2nd-Degree Model

- SSE reduced by 80.2%, but error remained in high-radiation zones
- Error heatmap revealed performance drop in complex zones



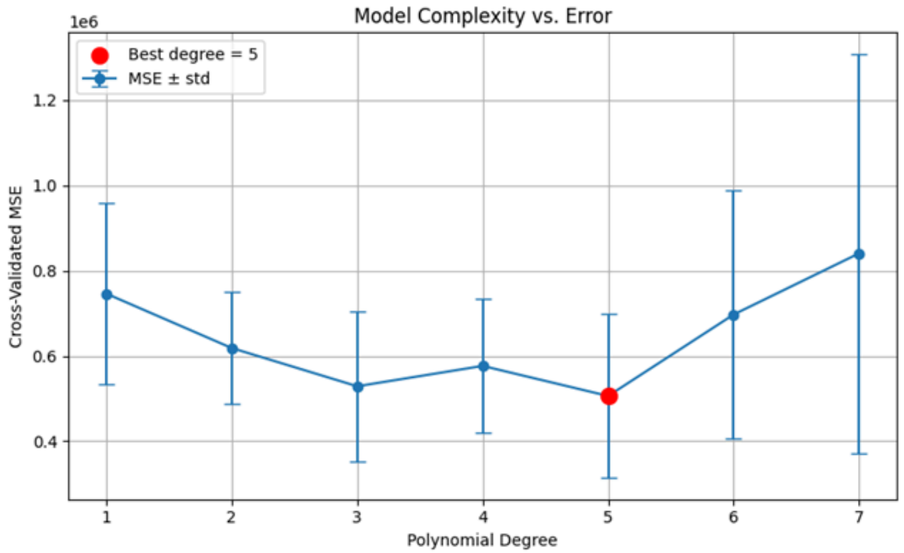
Results : Emulator Model Development

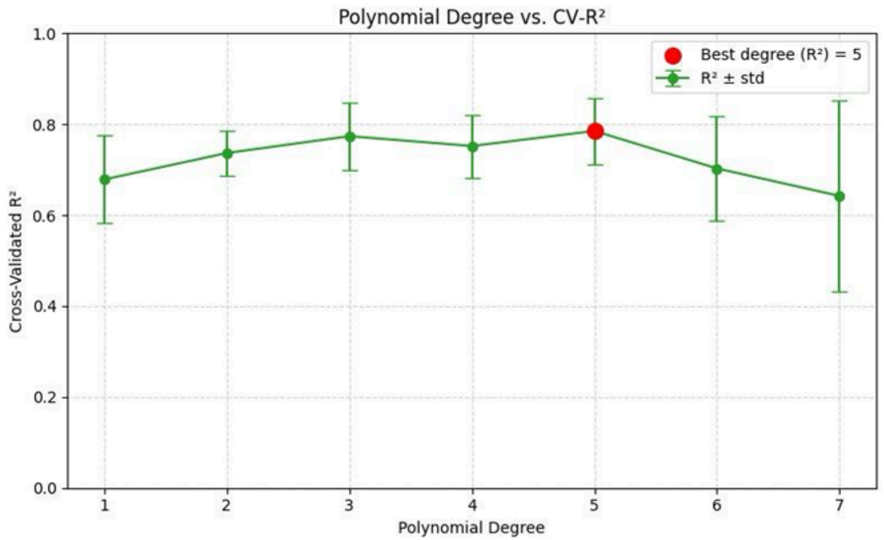
From Simulation to Prediction

3

Upgrade to 5th-Degree Model

- Added full polynomial terms up to degree 5 (21 features)
- 7-fold cross-validation used to avoid overfitting
- Best fit at degree = 5:
  - MSE: 506,130 ± 192k
  - $R^2 = 0.785$





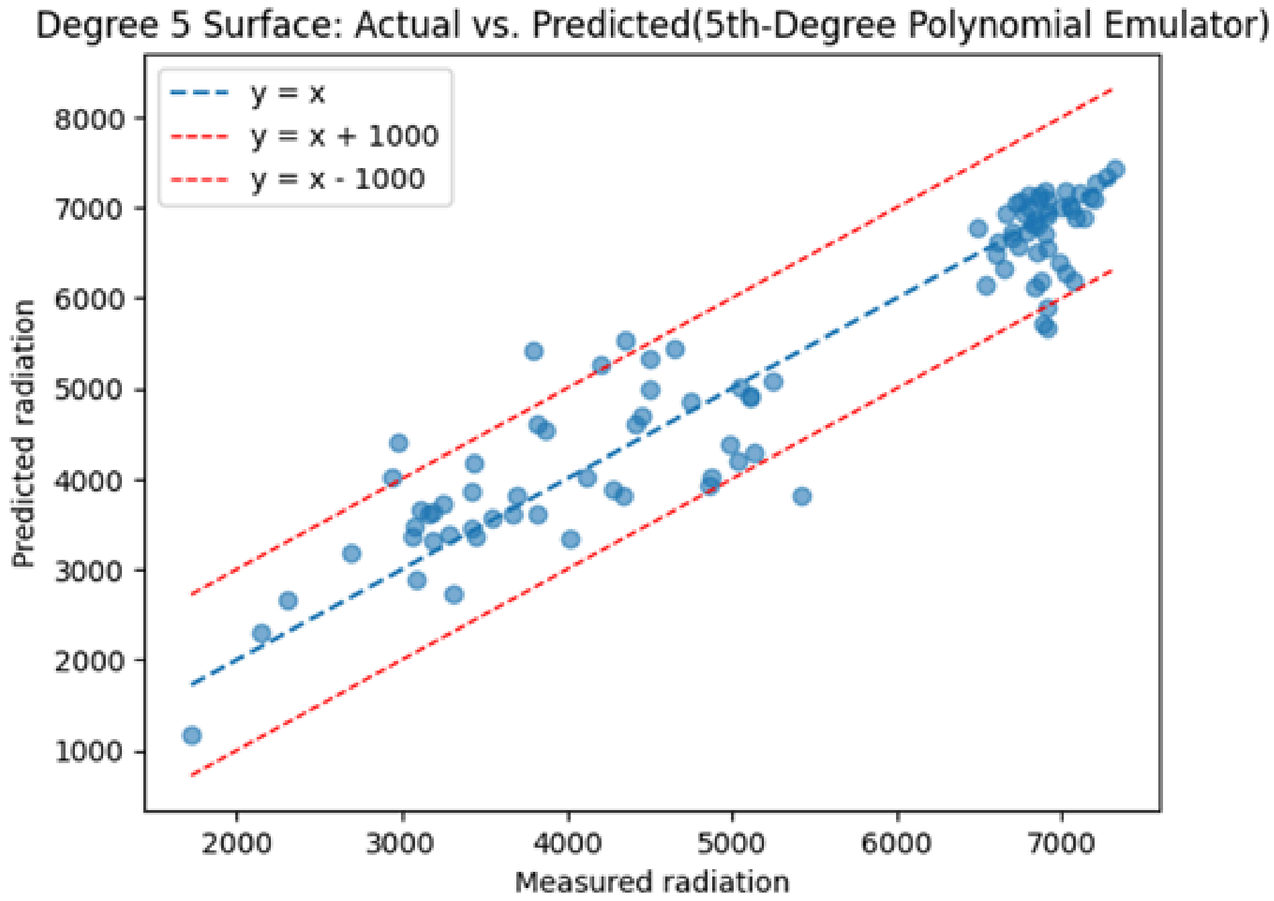
Results : Emulator Model Development

From Simulation to Prediction

4

Accuracy Improvement

- SSE dropped by 88.6% → better than 2nd-degree
- Visual results: tighter cluster along y=x



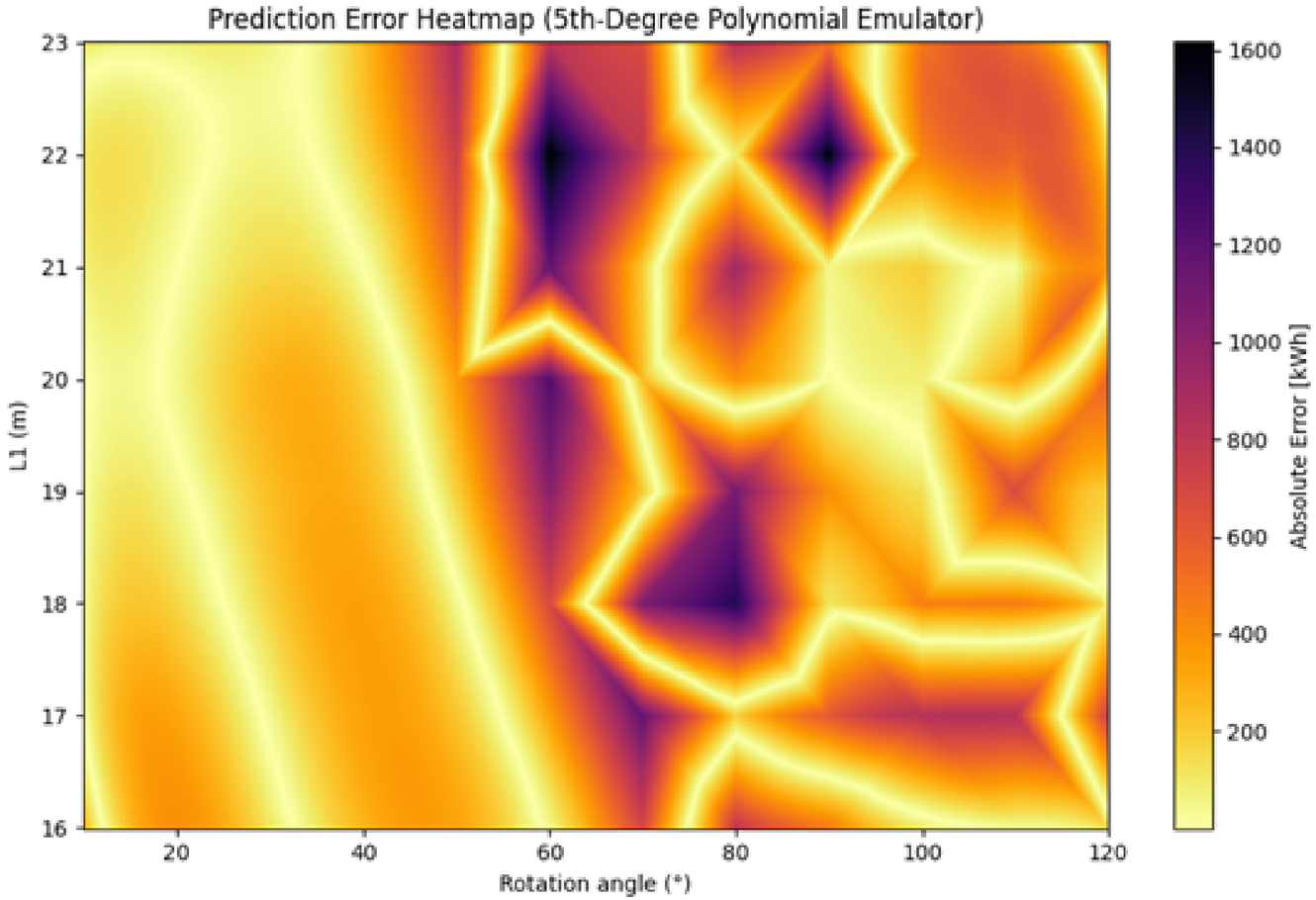
Results : Emulator Model Development

From Simulation to Prediction

5

Local Error Heatmap

- Lower error in low rotation zones
- Some localized overfitting in high rotation/high L1 areas



Results : Emulator Model Development

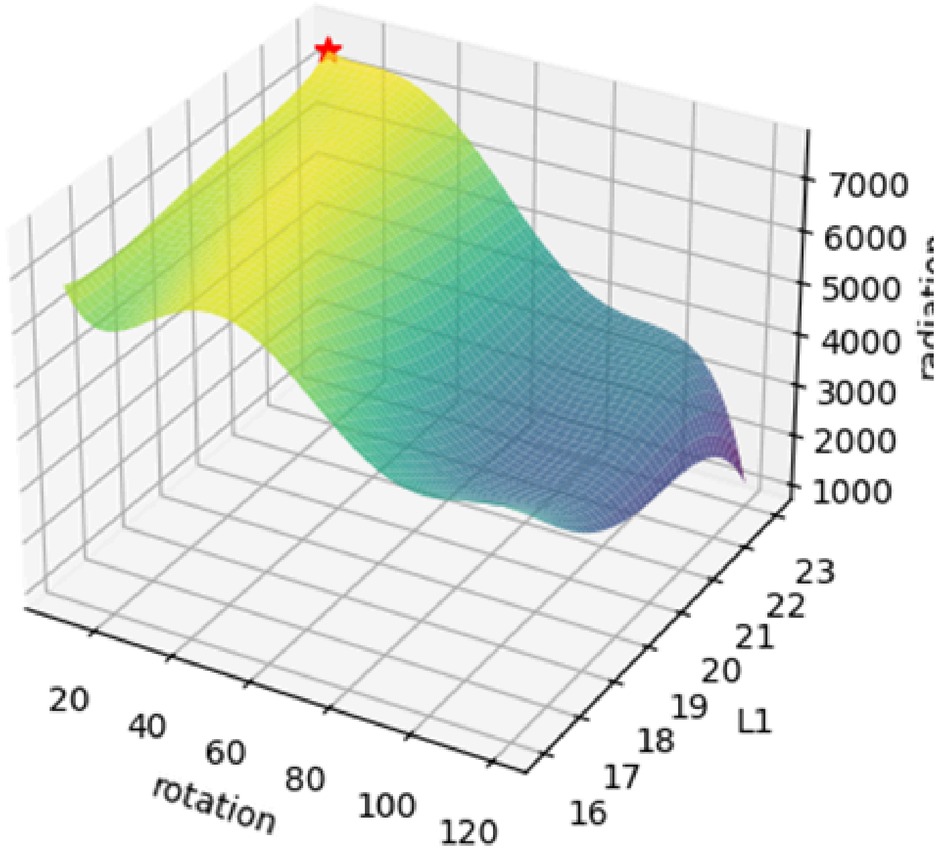
From Simulation to Prediction

5

Test Predictions & Global Optimum

- Known input (20°, 20m):
  - 5th-degree: 6900.63 kWh
  - Actual: 6909.38 kWh
- Best config:

$$r = 10^{\circ}, L_1 = 23m \rightarrow \text{Predicted } 7435.3 \text{ kWh}$$



Conclusion

- 5th-degree emulator captures non-linear behavior
- Enables fast and accurate prediction without simulation
- Supports design space exploration and optimization

Results : Optimization

Finding the Geometry That Achieves 6000 kWh

1 Define the Objective Function

- Goal: minimize squared error from target radiation (6000 kWh)
- Error function: $E(r, L_1) = (\hat{f}(r, L_1) - 6000)^2$
- $\hat{f}$ is the 5th-degree emulator predicting radiation

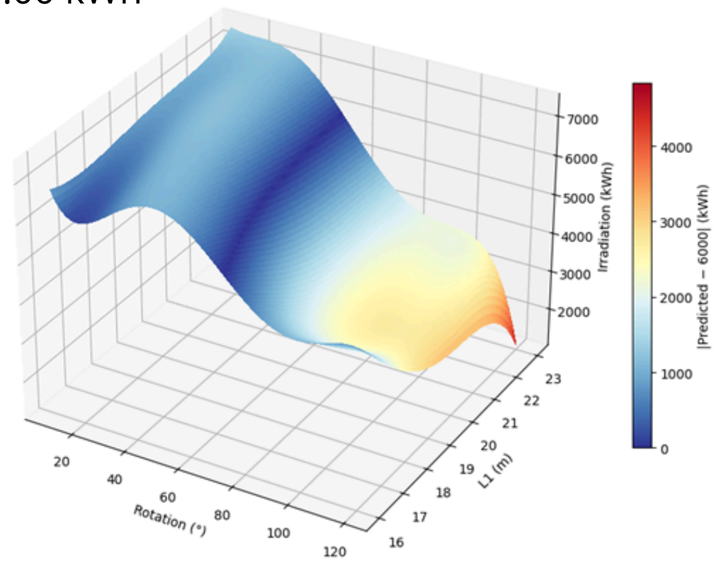
2 Set Up SciPy Optimizer

- Optimizer: scipy.optimize.minimize using Powell method
- Bounds:
  - r ∈ [10°, 120°]
  - L_1 ∈ [16 m, 23 m]
- Start point: previously found maximum (10°, 23 m)

3 Optimization Results

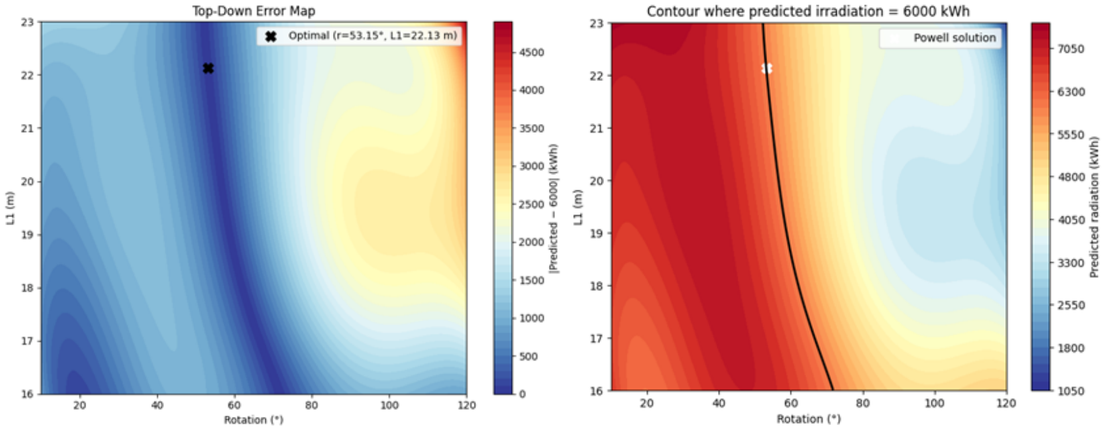
- Converged at:
  - Rotation: 53.15°
  - L_1 length: 22.13 m
  - Radiation: 6000.00 kWh
  - Error²: 0.0000

Final Irradiation Value
5999.89 kWh



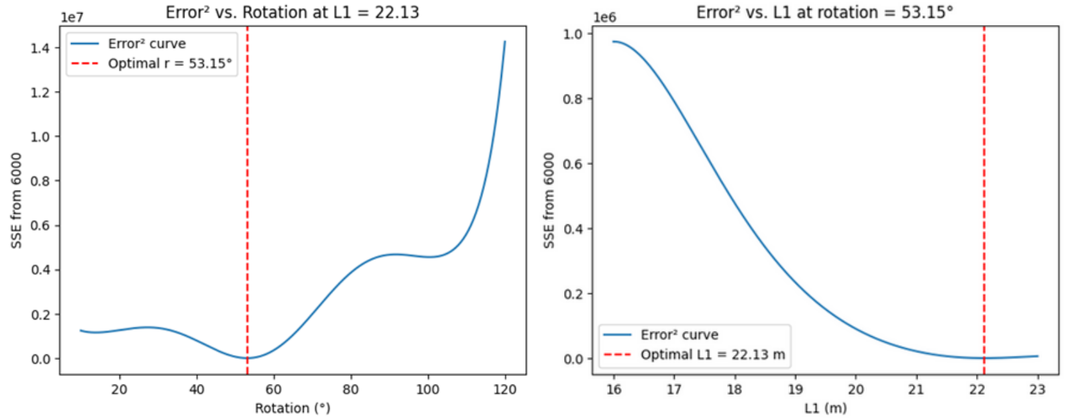
4 Visualize Error Zones

- Blue regions = (r, L1) pairs close to 6000 kWh
- Red regions = high error
- Shows optimizer's local convergence despite multiple near-optimal basins



5 Understand Variable Sensitivity

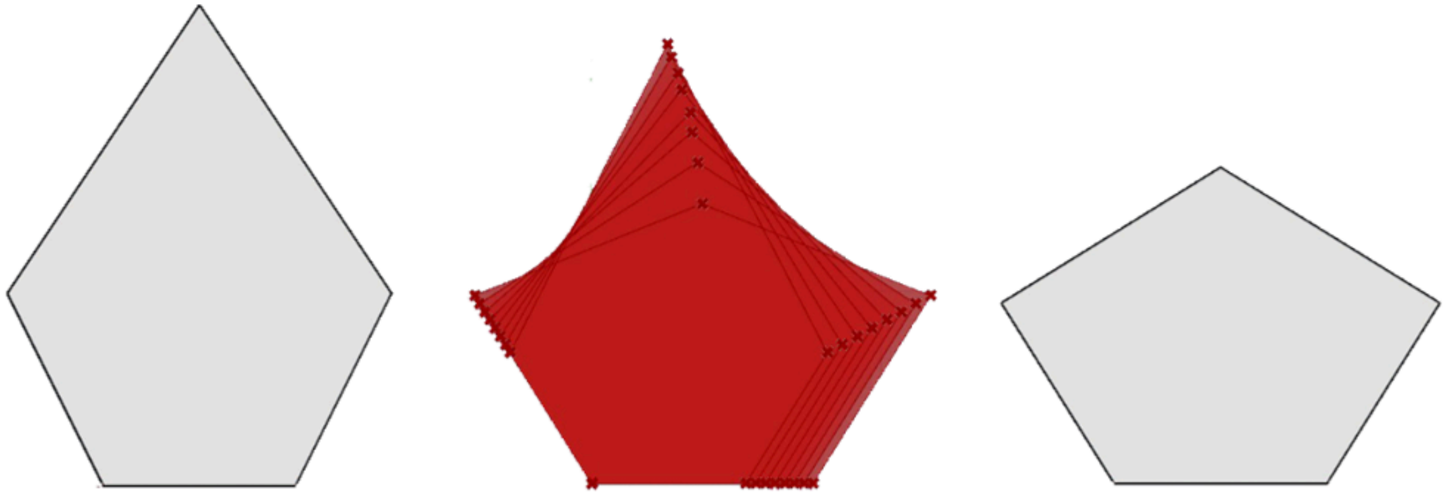
- Error vs Rotation (at fixed L_1 = 22.13)
- Error vs L_1 (at fixed r = 53.15)
- Shows how optimization "locks in" variables that minimize deviation



Using a 5th-degree emulator + Powell optimization, we precisely reached the 6000 kWh target The optimizer identifies one solution, even though multiple may exist, due to stopping criteria and local minimal, This confirms the workflow is accurate, efficient, and ready for use in performance-driven parametric design

Results : Optimized Floor Plan Design

From Initial Geometry to Solar-Optimized Solution



Initial Shape (left) : Regular pentagon with equal side lengths and no rotation

Parametric Variations (middle) : Generated by adjusting L1 and L2 while keeping perimeter constant → Explored entire design space in Grasshopper

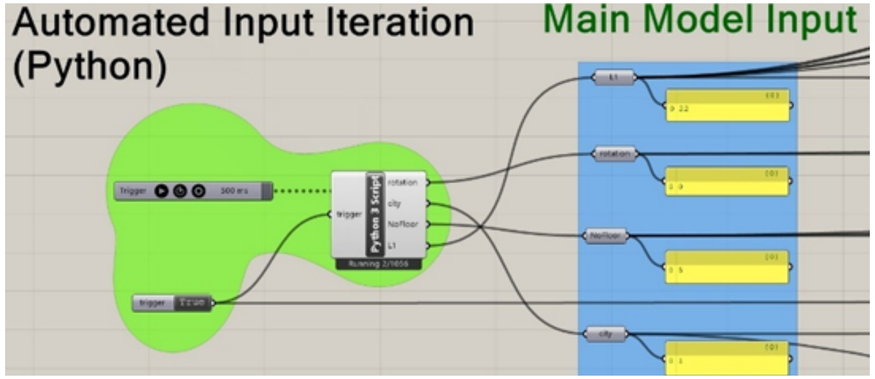
Simulation & Emulator : Each configuration evaluated with Ladybug → Trained polynomial emulator

Optimization (right) : Python + Powell method → Found floor plan with 6000 kWh irradiation

Final Geometry : Optimized proportions and rotation yield high-performance floor design

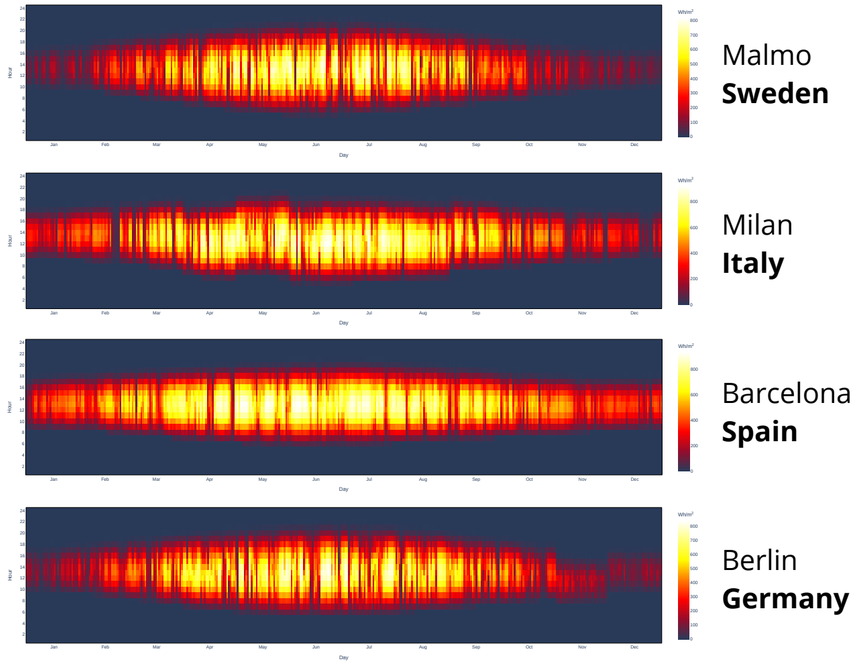
Innovation and Broader Implications

Automatic Iteration in Python



- This study presents a novel method to analyze twisted towers.
- Parametric model + online platform = no advanced modelling expertise required.
- Enables direct input of parameters to estimate solar irradiation and surface area

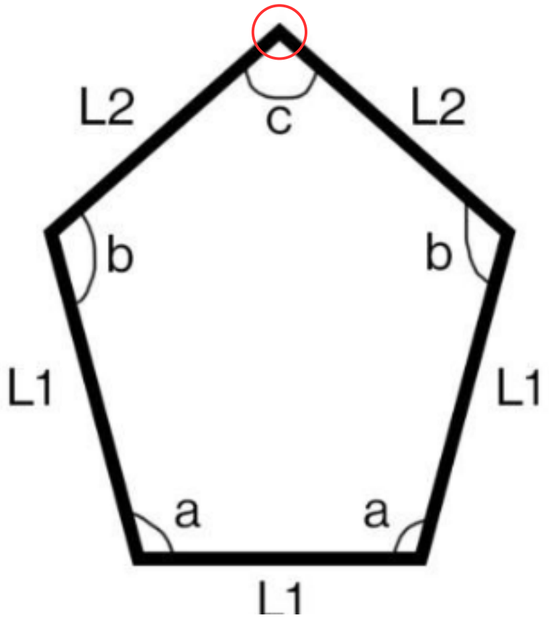
Different EPW



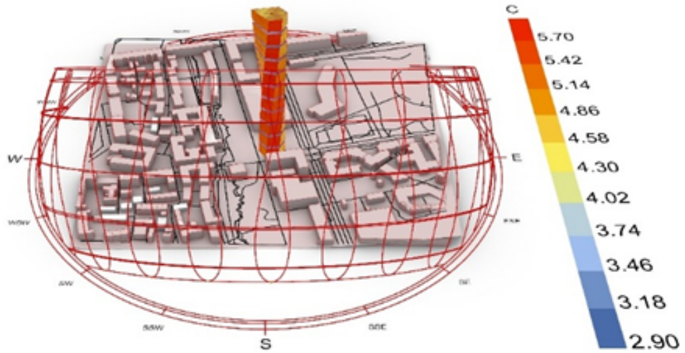
Further improvement by using different cities EPW

Innovation and Broader Implications

Base Geometry based on Phyton Code



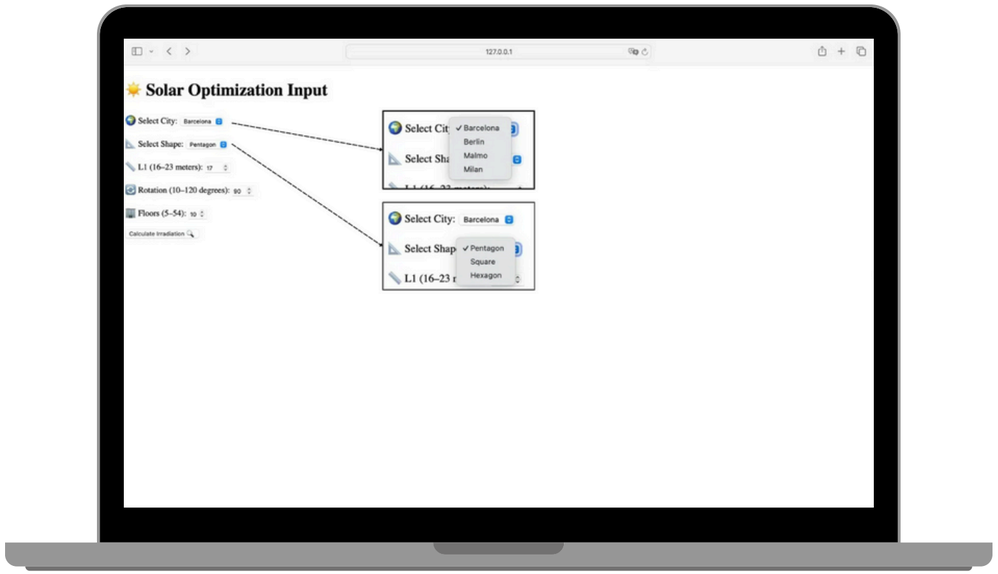
Context-Based Radiation Analysis



The local shade of nearby building might affect the total irradiation fall on the building facade, hence we assign the context for higher precision irradiation analysis

Innovation and Broader Implications

User-friendly Interface



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.

Conclusion

This project explored how parametric modeling and emulator-based optimization can guide solar-responsive design in twisted high-rise towers. Using the Turning Torso in Malmö as a case study, we found that **low rotation angles (around 10°) and longer floor edges (L1 = 23 m) maximize solar gain**. By training a fifth-degree polynomial emulator, **we achieved high predictive accuracy (88.6% SSE reduction), enabling fast and precise optimization toward a 6000 kWh radiation target.**

While some local prediction errors remain, the workflow—combining Grasshopper, Ladybug, Python, and SciPy—proves flexible, efficient, and scalable. It contributes not only to sustainable tower design in low-sun climates, but also to broader applications of data-driven geometry in architecture

Turning Torso