MathWorks Math Modeling Challenge 2025

Flint Hill School

Team #17839, Oakton, Virginia
Coach: Elizabeth Van Lieshout
Students: Jack Dunn, Kevin Harvey, Billy Ho, Divya Kudva, Neev Patel

M3 Challenge FINALIST—$5,000 Team Award

JUDGE COMMENTS

Specifically for Team #17839—Submitted at the close of triage judging

COMMENT 1: Extremely well-written. Assumptions are clear. Modeling is thorough. Outputs are analyzed.

COMMENT 2: Nice job in addressing Q1. Did a good job in citing resources.

COMMENT 3: Check that the units reported for the variables and other quantities are compatible and used consistently. Some sections of the report are repeated indicating that a general proofreading is in order. Use consistent rounding in the reporting of numbers.

COMMENT 4: You have a well-written and clear report that justifies your choice of models and provides results. You take into account several factors in the development of your models to make them more predictive. Well done! The results of your models are reasonable and show great effort!

Hot Button Issue:
It's Getting Hot In Here

March 2, 2025

1. Executive Summary

As the years go by, climate change becomes increasingly worrisome. With global temperatures on the rise and with multiple record-breaking temperatures being reached worldwide, heat waves have grown in frequency and intensity. Cities around the world experience power outages due to persistent heat waves. We seek to assess the effects of the increasingly distressing heat waves in Memphis, Tennessee.

We first developed a model to determine the effect of heat waves on individual housing units across Memphis. Using a General Energy Balance Equation model, we estimated the change of the given houses' interior temperatures throughout a day during a heat wave. We took into account factors like heat from conduction, residents to determine internal heat, and air change per hour to determine heat loss and gain from ventilation. Our results from the modified Energy Balance Equation model for Problem 1 were incredibly defensible. For example, Home 4 with more surface area and exposure to sun experience the most drastic temperature changes as well as the highest peak temperature, whilst Home 1 was the least susceptible, barely reaching the peak outdoor temperature around sunset due to retention of heat from conduction and internal heating.

We then developed a model to predict the peak demand that Memphis, Tennessee's power grid will need to accommodate fore during increasingly hot summer months in the next 20 years. Using our model from Problem 1, we determined vulnerability for communities experiencing heat waves. We used a thermal equilibrium model to determine power demand during extreme heat events by determining the amount of energy required to keep one house at the desired temperature, 22 C°, then calculating the entire city of Memphis' power consumption based on the demands of different housing types. After fine tuning the baseline demand for power to account for variance, we determined that the peak demand, 32.5% of which came from air conditioning. Accounting for population growth and rising climate change, we determined peak power in the coming years will grow upwards of 4000 MW to 5000 MW; however, legislation and technological advancements in the next decade are predicted to reduce cooling requirements by 40% to 50%. These future changes can be utilized to offset the drastic increase in power consumption from population growth and effects of climate change.

Finally, we developed a third model to determine the vulnerability of certain neighborhoods in Memphis, Tennessee using a Principle Component Analysis (PCA). The PCA determined the weights of each factor by maximizing variance of each factor and basing its weight on how much it effects the output. Using the proportions calculated by this model, we determined the vulnerability scores of each neighborhood, with East Memphis at the most risk, with a score of 100, and Rossville with the lowest score of 9.99.

The 3 models we have developed will be vital to ameliorate possible damages from dangerous heat events worsened by climate change in Memphis, and the world as a whole.

Contents

2. Part 1: Hot to Go

2.1. Problem Statement

Problem 1 asks us to develop a model to predict the indoor temperature of any non-air-conditioned dwelling during a heat wave over a 24-hour period in either Memphis, Tennessee, or Birmingham, United Kingdom. For our model, we are assessing the effect of a heat wave on housing in Memphis, Tennessee. We are asked to explain our choices in determining our model. We are asked to use the M3 dateset as part of our solution.[1]

2.2. Assumptions and Justifications

- **Assumption: Function Shade (f_{shade}) is defined as the following:**
  - Very shady = 0
  - Not very shady = 0.5
  - Not at all shady = 1

  - Justification: The classification of "very, not very, and not at all shady" within the data set provides us with 3 different levels of shadiness. This allow us to split the three different level as 0, 0.5 and 1, respectively.

- **Assumption: $\delta(t)$ is equal to 3 C°from 6 AM to 6 PM. At any other point in time, $\delta(t)$ is equal to 0.**
  - Justification: Due to our research and acquired knowledge about homes in Memphis, Tennessee, we have established 3 C°to be an accurate average change in $\delta(t)$ from time 6AM to 6PM. As for why it is from 6AM to 6PM, that is when the sun rises and sets. When the sun rises, the sunlight hits the house with nothing in the way of the sun, so the sunlight instantly starts heating the house. Similarly, when the sun sets, the sunlight no longer hits the house all, so the sunlight instantly stops heating the house.

- **Assumption: The height of each floor of a residence is approximately 10 feet.**
  - Justification: In the state of Tennessee, floors of buildings have an average height of 10 ft. [2]

- **Assumption: Older houses have worse insulation.**

[1] Hot Button Issue, MathWorks Math Modeling Challenge 2025, curated data, https://m3challenge.siam.org/897bjhb54cgfc/.

[2] https://www.thempc.org/eagenda/x/mpc/2018/october-9-2018-regular-mpc-meeting/table-2-tn-2-dimensional-standards.pdf

- Justification: Because of innovation in construction techniques, older houses, particularly ones build before 1960s, do not retain heat as well as modern housing. [3]

- **Homes in Memphis have an air change rate(ACH) of 3 or 0.5 per hour depending on the temperature difference.**
  - Justification: When the temperature outside is cooler then the temperature inside, people will open the windows to cool off the house. This circulation that we are measuring is called Air Change per Hour (ACH). Due to the few house options we have available, we have decided to use an ACH of 3 for our modeling. This is because 3 ACH is a minimum standard that most buildings should be in accordance with when windows are opened. However, the ACH dramatically decreases when the temperature outside is hotter than within, as people will then close the window in an attempt to prevent the heat from building up. This causes our ACH to be reduced to 0.5, as there will still be some circulation from cracks and other areas. [4].

- **The sun has an immediate effect on internal heating when it rises or sets.**
  - Justification: We do not have any addition information on how other external factors (number of obstruction, percentage of sunlight, etc) will effect the amount of solar radiation on heating from the sun. Therefore, the moment that the sun rises/set below the horizon, we will treat it as having full contact with house.

- **All residents of a household are within the residence during the heat wave.**
  - Justification: In order to account for internal heat, we must assume the presence of all residents.

- **A singular person within a home generates 100 Watts of power each day.**
  - Justification: It can be approximated that a human being on average generates 100 Watts just by existing. [5]

- **Assumption: Building thermal capacity (C) is approximately 75,000 J/K.**
  - Justification: This value represents typical residential thermal mass derived from building physics research. Studies of residential buildings show thermal capacities ranging from 40,000-100,000 J/K depending on construction type, with wood-frame homes (common in Memphis) averaging around 75,000 J/K. [6]

[3] https://www.insulatekansascity.com/insulation-blog/inherited-an-old-house-heres-how-to-check-if-its-under-insulated/#:~:text=According%20to%20experts%20from%20Realtor,built%20with%20insulation%20in%20mind.
[4] https://www.sanalifewellness.com/blog/air-changes-per-hour-ach
[5] https://www.fst.com/news-stories/magazine/renewable-energy/human-power-plant/
[6] https://codes.iccsafe.org/content/IRC2024P2/chapter-11-re-energy-efficiency

2.3. Developing the Model

For Problem 1, we decided to use a General Energy Balance Equation. This equation takes into account sources of thermal energy that affect the house. Our components include capacitance, heat from conduction, heat from internal sources within the house, and heat from the ventilation or airflow of the house. We can use the differential equation to solve for $\frac{dT_{in}}{dt}$, which will give us rate of change of T_{in}, the internal temperature.

$$C \cdot \frac{dT_{in}}{dt} = Q_{conduction} + Q_{internal} + Q_{solar} + Q_{ventilation} \quad (1)$$

Variables of the Energy Balance Equation		
Variables	**Definition**	**Units**
C	Heat Capacitance	Joules/Kelvin
$Q_{conduction}$	Newton's Law of Cooling	W
$Q_{internal}$	Heat generated from people inside	W
Q_{solar}	Heat from solar radiation	W
$Q_{ventilation}$	Heat being circulated	$\frac{J}{h}$

Table 1: Summary of Variables in the General Energy Balance Equation

The General Energy Balance Equation considers Q_{solar} in order to include heat imposed onto a house due to the sun's radiation; however, we determined that our $Q_{conduction}$ accounts for the this heat factor due to solar radiation. Therefore, we can safely disregard Q_{solar}, and we can eschew it from our model.

$Q_{conduction} = UA \cdot (T_{sa} - T_{in})$		
Variables	**Definition**	**Units**
$Q_{conduction}$	Newton's Law of Cooling	N/A
UA	Constant of heat transfer	W/K
T_{sa}	$T_{out} + f_{shade} \cdot \delta t$; solar radiation onto the house	C°
T_{in}	Temperature in	C°
T_{out}	Temperature outside	C°
f_{shade}	Shade constant	N/A
δt	$\delta t = \begin{cases} 6am < x < 6pm & 3 \\ 6pm < x < 6am & 0 \end{cases} \quad (2)$	C°

Table 2: Variables for $Q_{conduction}$

The equation for $Q_{conduction}$ uses Newton's Law of Cooling to estimate the heat transfer of the outside temperature to the interior of the residence.[7] Newton's Law of Cooling states

[7]https://mathresearch.utsa.edu

that the rate at which an object will cool is proportional to the difference in temperatures between the object and its surroundings. In the context of the problem, this means that the rate at which the house gains heat due to conduction is proportional to the temperature outside minus the temperature within the house.

This equation utilizes UA, temperature due to solar radiation on the house and temperature inside the house. UA is a constant of insulation that determines how well a building retains heat. We considered the building type, with apartments having a better heat transfer capacity, and houses having a worse one. This means apartment-style buildings will gain or lose heat at a much faster rate than house-style buildings do. In addition to building type, we took into account building age, in which the older the building is, the worse the heat transfer capacity will be, as stated by one of our assumptions. T_{sa} is the sol-air, which is composed of the outside temperature plus the temperature gained from solar radiation projected onto the house.[8] Our model also takes into account the surface area, and larger surface areas will result in a higher T_{sa}. The level of shade a house has is determined by the given data on Memphis, TN, which qualifies a houses' shadiness on a level of "Very Shady" to "Not Shady at All." We can roughly use the shadiness and constant of insulation to estimate the amount of heat added to the internal temperature of a house due to this direct solar radiation.

$Q_{internal} = p \cdot \frac{100}{V_h}$		
Variables	**Definition**	**Units**
p	Number of people inside	Peeople
100	Heat generated per person	W
V_h	Volume of the house	m^3

Table 3: Variables for $Q_{internal}$

The equation for $Q_{internal}$ is undoubtedly refreshing after seeing the monstrosities of our previous variables. Internal heat is the heat generated within the residence. For our purposes, we can simply calculate internal heat by considering the number of people within a residence. The average human generates 100 watts of energy at rest [9]. However, $Q_{internal}$ has a small overall impact on the overall temperature of the house because it mostly just impacts the temperature that the people within experience, which we are not considering.

$Q_{ventilation} = ACH \cdot 0.9 \cdot A \cdot (T_{out} - T_{in})$		
Variables	**Definition**	**Units**
ACH	Air change per hour	N/A
0.9	Ventilation coefficient	$W\frac{m^2}{K}$
A	Area of floor	m^2
$T_{out} - T_{in}$	Change in temperature	C°

Table 4: Variables for $Q_{ventilation}$

Finally, $Q_{ventilation}$ is the heat due to the circulation of air. This component determines

[8] https://www.sciencedirect.com/science/article/abs/pii/0306261977900174
[9] https://www.fst.com/news-stories/magazine/renewable-energy/human-power-plant/

the amount of heat gained or lost from ventilation. The ACH is the amount of air changes per hour, which tells us at what rate the indoor air is replaced. [10]. This factor is important to consider in order to assess how the circulation of air within a home affects its internal temperature. For most homes, the ACH is 3 air changes per hour [11].

$$C \cdot \frac{dT_{in}}{dt} = UA \cdot (T_{sa} - T_{in}) + p \cdot 100 + ACH \cdot 0.9 \cdot A \cdot (T_{out} - T_{in}) \tag{3}$$

Alas, here is the final equation for our model. Each component has been described above. We can utilize this equation to determine the effect of a heat wave on internal temperatures in Memphis, Tennessee.

2.4. Analyzing the Results

Using the given data from the Memphis, Tennessee data set, our model predicted the heat flow throughout the 4 given homes. Figure 3.1 shows the change in heat, in Watts, throughout a day. All 4 houses experienced a rapid influx of heat around 5 to 6 AM in the morning, and a rapid decrease in heat around 6 to 7 pm. This occurrence is due to the sun. Because our model greatly depends on temperature outside, it is reasonable to see these two time-frames as points of drastic changes due to the sun rising and setting.

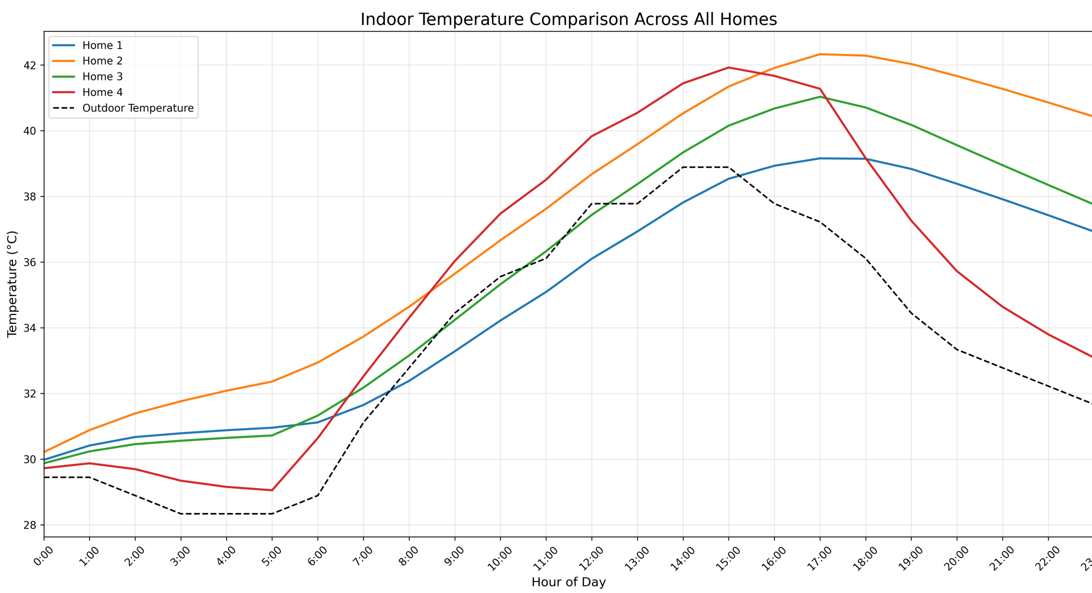

Figure 2.1: Indoor Temperature Comparison Across All Homes

Figure 3.2 displays the internal temperature throughout a day. All four homes get increasingly warm throughout the day and then slowly cool off after sunset. Homes 3 and 4 eventually exceed the highest outdoor temperature, reaching over 40 C°. Homes 1 and 2 stay below the highest outdoor temperature throughout the day, although they do retain heat, even hours after sunset.

[10]https://cleanair.camfil.us/2021/10/22/how-to-calculate-air-changes-per-hour
[11]https://www.sanalifewellness.com/blog/air-changes-per-hour-ach.

Home 1		
Time	**Out (°C)**	**In (°C)**
00:00	29.4	30.0
01:00	29.4	30.4
02:00	28.9	30.7
03:00	28.3	30.8
04:00	28.3	30.9
05:00	28.3	31.0
06:00	28.9	31.1
07:00	31.1	31.7
08:00	32.8	32.4
09:00	34.4	33.3
10:00	35.6	34.2
11:00	36.1	35.1
12:00	37.8	36.1
13:00	37.8	36.9
14:00	38.9	37.8
15:00	38.9	38.5
16:00	37.8	38.9
17:00	37.2	39.2
18:00	36.1	39.1
19:00	34.4	38.8
20:00	33.3	38.4
21:00	32.8	37.9
22:00	32.2	37.4
23:00	31.7	36.9

Figure 2.2: Home 1 Temperatures

Home 2		
Time	**Out (°C)**	**In (°C)**
00:00	29.4	30.2
01:00	29.4	30.9
02:00	28.9	31.4
03:00	28.3	31.8
04:00	28.3	32.1
05:00	28.3	32.4
06:00	28.9	32.9
07:00	31.1	33.7
08:00	32.8	34.6
09:00	34.4	35.6
10:00	35.6	36.7
11:00	36.1	37.6
12:00	37.8	38.7
13:00	37.8	39.6
14:00	38.9	40.5
15:00	38.9	41.3
16:00	37.8	41.9
17:00	37.2	42.3
18:00	36.1	42.3
19:00	34.4	42.0
20:00	33.3	41.7
21:00	32.8	41.3
22:00	32.2	40.9
23:00	31.7	40.4

Figure 2.3: Home 2 Temperatures

Home 3		
Time	**Out (°C)**	**In (°C)**
00:00	29.4	29.9
01:00	29.4	30.2
02:00	28.9	30.5
03:00	28.3	30.6
04:00	28.3	30.6
05:00	28.3	30.7
06:00	28.9	31.3
07:00	31.1	32.2
08:00	32.8	33.2
09:00	34.4	34.2
10:00	35.6	35.3
11:00	36.1	36.3
12:00	37.8	37.4
13:00	37.8	38.4
14:00	38.9	39.3
15:00	38.9	40.2
16:00	37.8	40.7
17:00	37.2	41.0
18:00	36.1	40.7
19:00	34.4	40.2
20:00	33.3	39.6
21:00	32.8	39.0
22:00	32.2	38.4
23:00	31.7	37.8

Figure 2.4: Home 3 Temperatures

Home 4		
Time	**Out (°C)**	**In (°C)**
00:00	29.4	29.7
01:00	29.4	29.9
02:00	28.9	29.7
03:00	28.3	29.3
04:00	28.3	29.2
05:00	28.3	29.1
06:00	28.9	30.6
07:00	31.1	32.5
08:00	32.8	34.3
09:00	34.4	36.0
10:00	35.6	37.5
11:00	36.1	38.5
12:00	37.8	39.8
13:00	37.8	40.5
14:00	38.9	41.4
15:00	38.9	41.9
16:00	37.8	41.7
17:00	37.2	41.3
18:00	36.1	39.2
19:00	34.4	37.3
20:00	33.3	35.7
21:00	32.8	34.6
22:00	32.2	33.8
23:00	31.7	33.1

Figure 2.5: Home 4 Temperatures

According to our model, the home most susceptible to overheating due to heat waves is Home 4. Home 4 is a single-family residence made in 1990, and it is not located in a shaded area. It is also the largest of the four homes, with a square-meterage of 278 m^2, and 6 people living in it. The house that is least susceptible is Home 1, which is a 3 person, single-family residence made in 1953, located in shade with a square-meterage of 88 m^2. However, all houses reach or surpass the max temperature of the day, 39 C°.

There are 3 glaring differences between Home 1 and Home 4. Firstly, the size of the homes are different. Despite the residences both being single-family homes, Home 4 is drastically bigger than Home 1, being over 3 times the size of Home 1. Secondly, the shadiness of the homes proved to be important. Houses 3 and 4, the most susceptible homes, were both classified as "Not at all in shade," and House 2 was classified as "Not very shady," whilst House 1 was "Very shady." The major factors that made Home 1 the least susceptible was the square-meterage, with it being 88 m^2, as well as the fact it had only 3 residents.

An important observation is the rapid rate at which the internal temperature of House 4 decreases after sunset. This is due to the fact it is incredibly large, making its Q conduction very, very large. Thus, when the sun goes down, the major source of heat practically vanishes and allows for rapid cooling. The Homes 2 and 3, however, have 2 to 3 people living in significantly smaller, unshaded homes, leading high retention of heat gained during the day, even after sunset.

From our results, it can be determined that the 3 major factors to overheating in homes are the square-meterage, the number of people inside, as well as the degree of shade. These results are logical given the nature of our model and the factors it takes into account.

2.5. Strengths and Weaknesses

2.5.1 Strengths

- We chose a differential equations model, which is effective in capturing the rate of change. This is useful in the context of determining inside temperature because the inside temperature can be determined from the rate of change of the initial temperature. The rate of cooling depends on the instantaneous temperature, so we have to use a differential equation because it takes into account the instantaneous temperature.

- This model is not overly complicated, every part have a logical explanation and the math doesn't rely on neural networks or simulation.

2.5.2 Weaknesses

- Our model fails to take into account wind into our internal temperature. Since the wind speed can affect the air changes per hour (ACH), this lack of the wind speed can lead our predicted temperature to vary from the actual value depending on the wind speed.

- Our model has variable constants which vary from house to house. Some factors including the function for shade and capacitance of the house vary from house to house. We did not account for each individual house and instead generalized to houses in Memphis, Tennessee.

- Our model always assumes that it is sunny outside, leading to inaccurate predictions on days in which it is cloudy. Although this effect is drastically reduces as during a heatwave it is almost always sunny.
- We solved this differential equation using Euler's method, which leads our data to not be continuous and is only close approximate to the actual solution. However, in order for us to solve Euler's method, we had to take small steps, seconds instead of hours, which led us to interpolate our given temperature in hours to temperature in seconds. This additional interpolation could potentially make our model less accurate.

3. Part 2: Power Hungry

3.1. Problem Statement

Problem 2 asks us to develop a model that predicts the peak demand that Memphis, Tennessee's power grid should be prepared to handle during the summer months. We are asked if we foresee any changes in the maximum demand 20 years from now.

3.2. Assumptions and Justifications

- **Assumption: The average temperature that people in Memphis keep their house at is between 70-73 degrees Fahrenheit or 22 degrees Celsius**
  - Justification: This is the average that an American keeps their home temperature at. [12]
- **Assumption: Older homes have reduced AC capacity and efficiency.**
  - Justification: We implemented capacity factors (0.7-1.0) based on home age. This reflects real-world conditions where older AC systems experience reduced capacity due to wear, outdated technology, and improper sizing. [13]
- **Assumption: Buildings can be represented by a single thermal zone with uniform temperature.**
  - Justification: While real buildings have temperature variations between rooms, the single-zone approach is standard in building energy modeling for city-scale simulations. For predicting overall energy consumption patterns, this simplification introduces minimal error while greatly reducing complexity.
- **Assumption: Memphis housing stock can be reasonably represented by four archetypal building models.**

[12] https://www.adt.com/resources/average-room-temperature: :text=That%20being%20said%2C%20the%20average,68%2

[13] "https://www.aceee.org/research-report/a2001"

– Justification: While there may be other type of building models, considering all possible models increases complexity while have four types does not drastically increase the error.

3.3. Developing the Model

We chose a thermal equilibrium model based on our previous model from Problem 1 to determine the power demand during extreme heat events. This approach allowed us to measure the heat exchange between indoor and outdoor environments. A physics-based approach is more appropriate than others because power demand during extreme heat events are largely driven by cooling needs by integrating the thermodynamics of buildings to create an accurate representation of citywide power demand. We considered other approaches, such as statistical regression based on historical patterns of temperature and power consumption, but those models often fail to capture non-linear relationships between temperature and power consumption. They also cannot account for building-specific factors such as the insulation, population, shade, and AC-efficiency of each house.

We first solved the Energy Balance Equation 3 in Question 1 for the amount of energy required to keep one house at the desired temperature, giving us:

$$P_{AC} = \min\left(\frac{UA(T_{sa} - T_{setpoint}) + P \times H_p + ACH \times 0.9 \times A \times (T_{out} - T_{setpoint})}{COP}, P_{rated} \cdot CF\right) \tag{4}$$

Energy Balance Equation Solved for Energy Required

Symbol	**Variables**	**Unit**
UA	Building envelope heat transfer coefficient	$\frac{W}{K}$
T_{sa}	Sol-air temperature	K
T_{in}	Indoor temperature	K
T_{out}	Outdoor temperature	K
PP	Number of persons	people
H_p	Heat gain per person	W
ACH	Air changes per hour	$\frac{1}{hr}$
AA	Floor area	m^2
P_{AC}	AC power consumption	W
COP	Coefficient of Performance	N/A
CF	Capacity factor	N/A
$Q_{required}$	Required cooling power	W

Table 5: Parameter definitions and typical values.

In order to calculate the entire city of Memphis' power consumption during an extreme heat event, we first summed up the various categories of residence types: detached houses, townhouses, apartments, and mobile or other types. We then calculate the proportion of each housing type. In order to factor in the age of the houses, we created a distribution for homes, breaking down the housing stock into categories. By combining the style-type percentages with these age brackets, we mapped the data provided–being the number of dwellings–to four housing model categories, similar to the four provided example dwelling types. In order to determine the baseline of power that the city of Memphis requires outside of an extreme heating event, we first calculated the region's average power demand. In order to do so, we divided the annual consumption into hourly watts. We then adjusted the baseline to reflect seasonal variations, such as increased appliance and lighting demands to accommodate for increased demand for A/c during summer. In order to account for systemic factors that are not residential heating, such as commercial or industrial electrical consumption, we then further refined the baseline.

To align the results with realistic peak demand scenarios, further refinements are applied to account for systemic factors not explicitly modeled, such as commercial or industrial consumption. These adjustments ensure that the baseline serves as a foundational reference point, allowing cooling demands—calculated separately through thermodynamic simulations—to be superimposed while maintaining plausible total grid demand levels. The approach emphasizes practical alignment with established utility benchmarks. We chose a proportionality constant of 1.5 for the baseline to help align it better with real data from Memphis' power grid.

We then summed the power required to cool all the different housing categories:

$$P_{total} = P_{baseline} + \sum_{i=1}^{4} P_{AC,i} \cdot N_i \cdot SF \tag{5}$$

Equation: Peak Energy Consumption Required

Symbol	**Variables**	**Unit**
P_{total}	Total power consumption	W
$P_{baseline}$	Baseline power consumption	W
$P_{AC,i}$	Air conditioning power consumption	W
N_i	Number of dwellings in housing category i	(count)
SF	Scaling factor	N/A

Table 6: Parameter definitions for the peak energy consumption equation.

The total peak power demand for Memphis is calculated by adding the baseline power consumption of the city with the aggregated air conditioning of the different housing categories. For each of the housing models(i), we multiply the air conditioning power consumption (P_AC,i) by the number of households in that category (N_i), and then apply a scaling factor (SF) of 70 to account for system-wide effects that are not captured in the individual building simulations.

3.4. Analyzing the Results

Using the model we created, we predict that at any random heatwave, the peak outdoor temperature is going to be 38.89 C°. This peak will occur at around 2 PM, as it follows the pattern of most heat waves. From this information, we determined that the Peak Power Demanded is 2850.51 MW, which 32.5% of comes from air conditioning. This results in an AC contribution of 926.96 MW at the peak of the heatwave. By 2045, Memphis's electrical consumptions patterns will likely have changed drastically due to climate change, population growth, and technological evolution. Increasing global temperatures will bring hotter average temperatures and harsher extreme heat events, such as heat waves. [14] These changes in consumer population and frequency of climate events will increase the demands for power as many more people will want A/C to cool their houses in a hotter climate. Even with a moderate population growth of the city, the peak power required by the city could grow upwards of 4000 MW: a substantial increase from current usage in 2024. Additionally, commercial and industrial sectors will increase in their power usage during potential heat events as they expand their businesses.

Building codes implemented in the 2020s and 2030s will result in approximately 30% of Memphis housing stock being constructed to much higher efficiency standards by 2044, reducing cooling requirements by 40-50% in these newer developments.[15] Retrofitting programs for existing buildings could improve another 25% of the housing stock, yielding 15-25% efficiency gains in these structures.[16] AC technology improvements should increase average system efficiency by 30-40%. [17] These technological and zoning changes could help offset the gain in power consumption from projected population growth and effects from climate change.

3.5. Sensitivity Analysis

We tested the sensitivity of our power demand model by systematically varying its value while holding all other inputs constant at their baseline values. This allowed us to isolate the impacts that each variable had on the power demand. We strategically selected four critical parameters: AC setpoint temperature (ranging from 19-25°C), AC efficiency (COP from 1.9-2.7), AC load scaling factor (50-90), and capacity factor scenarios (representing different distributions of AC system efficiencies across housing stock).

The analysis revealed that each 1°C increase in thermostat settings reduces peak power demand by approximately 39 MW (1.4%), with higher temperatures yielding progressively smaller AC contributions. It also showed that each 1°C increase in thermostat settings reduces peak power demand by approximately 39 MW (1.4%), with higher temperatures yielding progressively smaller AC contributions. The scalefactor highlights the substantial impact of the scaling factor, which accounts for commercial buildings and other effects not directly modeled, with each 10-point increase adding approximately 132 MW (4.6%) to peak demand. The capacity factor highlights the substantial impact of the scaling

[14] https://www.vox.com/a/weather-climate-change-us-cities-global-warming

[15] "https://www.tn.gov/commerce.html"

[16] "https://www.energy.gov/eere/wap/weatherization-assistance-program

[17] "https://www.energy.gov/eere/buildings/appliance-and-equipment-standards-program"

AC Setpoint Temperature (°C)	**Peak Power** (MW)	**AC Contribution** (%)
19	2928.24	34.3
20	2889.38	33.4
21	2850.51	32.5
22	2811.65	31.6
23	2772.79	30.6
24	2733.93	29.6
25	2695.06	28.6

Table 7: Effect of AC setpoint temperature on peak power demand.

AC Efficiency (COP)	**Peak Power** (MW)	**AC Contribution** (%)
1.9	3045.66	36.8
2.1	2938.80	34.5
2.3	2850.51	32.5
2.5	2776.36	30.7
2.7	2713.19	29.1

Table 8: Effect of AC efficiency (COP) on peak power demand.

AC Load Scale Factor	**Peak Power** (MW)	**AC Contribution** (%)
50	2585.67	25.6
60	2718.09	29.2
70	2850.51	32.5
80	2982.94	35.5
90	3115.36	38.3

Table 9: Effect of AC load scaling factor on peak power demand.

Capacity Factor Scenario	**Peak Power** (MW)	**AC Contribution** (%)
All high efficiency	2850.51	32.5
Baseline	2850.51	32.5
All low efficiency	2850.51	32.5
High variation	2850.51	32.5

Table 10: Effect of capacity factor scenarios on peak power demand.

factor, which accounts for commercial buildings and other effects not directly modeled, with each 10-point increase adding approximately 132 MW (4.6%) to peak demand.

3.6. Strengths and Weaknesses

3.6.1 Strengths

- We opted for a thermal equilibrium model, which effectively captures the essential physics of heat transfer and building thermodynamics. This approach allows for explicit modeling of conduction, ventilation, and cooling processes, setting it apart from simpler statistical models.
- The model accurately reflects the non-linear relationship between outdoor temperature and power consumption. It takes into account the rate of temperature change, which is crucial for understanding how buildings react to heat.
- This model is tailored to the region of Mephis, Tennessee, as it incorporates housing distribution data and electricity consumption patterns specific to the city. This tailoring ensures that local building characteristics and climate conditions are considered.
- By integrating Newton's Cooling Law and varying capacity factors for different types of homes, the model mirrors real-world scenarios where older buildings find it challenging to maintain comfort during heat waves, resulting in more realistic power demand forecasts.

3.6.2 Weaknesses

- Our model simplifies the diverse housing stock in Memphis into just four categories of buildings. This feature likely under-represents the true variety in building performance, especially for unique housing types or those with unusual characteristics.
- We apply a scaling factor of 70.0 to extend our residential model to citywide demand. This introduces potential inaccuracies by assuming proportional scaling relationships across different sectors and does not explicitly account for commercial and industrial buildings.
- The model assumes uniform thermostat settings and occupant behaviors across all households. In reality, preferences and behaviors can vary significantly, which can greatly influence power consumption.
- Additionally, our model lacks thorough validation against Memphis-specific power consumption data during similar heat events, relying instead on general patterns observed in other cities.

4. Part 3: Beat the Heat

4.1. Problem Statement

We are asked to develop a vulnerability score for various neighborhoods in Memphis to help them equitably allocate resources for minimizing the effects of a heat wave or a power grid failure. We are asked to justify all factors we choose to include in our vulnerability scores.

In addition, we must propose a single approach for how Memphis officials can incorporate our vulnerability scores into their management of heat waves.

4.2. Assumptions and Justifications

- **Assumption: All houses created in each decade was made out of the same material**
  - Justification: The majority of houses created in the same decade were constructed using the same material, which each have different properties that affect the neighborhoods vulnerability to heat.
- **Assumption: The results of our model reveal accurate vulnerabilities in Memphis, Tennessee.**
  - Justification: In order for our solutions and analysis to address Problem 3, we must assume, dare I say safely, that our model from Problem 1 accurately reveals flaws in certain housing types.

4.3. Developing the Model

To create our model, we first determined the factors relating to the vulnerability score. We have compiled these factors into the following table.

Symbol	**Variable**	**Unit**
N_H	Number of households	#
P_N	Proportion of developed, open space in neighborhood	N/A
H_{2010}	Homes built 2010 or later	#
H_{1990}	Homes built 1990 to 2009	#
H_{1970}	Homes built 1970 to 1989	#
H_{1950}	Homes built 1950 to 1969	#
$H_{<1950}$	Homes built 1950 or earlier	#
D_H	Detached whole house	#
T_H	Townhouse	#
P_H	Apartments	#
M_H	Mobile Homes/Other	#

Table 11: Individual factors which determine the vulnerability score

We choose the number of households as a factor because the more households there are the more demand on the power grid, and the more buildings are at risk of high levels of heat. The proportion of developed, open space in neighborhood is important because it helps determine the density of houses, and the amount of free space in the neighborhood. The more free space, the less vulnerable homes are to high levels of heat or the effects of power outages. The type of house and date of house construction is important because each type of house is differently effected by heat, and the date of house construction helps us determine the material the house was created. This is important as different materials have different constants for heat absorption.

After we complied all of these factors, we needed to decide the weights of each factor. To do this, we wanted to avoid subjectivity, and we opted to do a Principle Component Analysis or PCA. PCA is used to reduce variable data to its Principle Components. Yet even though this is helpful, this was not the main reason we decided to use this statistical method. PCA is a method which helps us determine the weights of each factor in a non-biased way. It does this by maximizing the variance of each of the factors, and basing its weight on how much that effects the output. And by normalizing our data, we ensure that our weights are not skewed towards the greater numbers. To perform a PCA, we first had to create a covariance matrix using the normalized values (z-scores) for each value in our data table. After this, we computed the eigenvalues and vectors. This is shown below.

N_H	P_N	H_{2010}	H_{1999}	H_{1970}	H_{1950}	$H_{<1950}$	D_H	T_H	P_H&M_H*
5.2920	1.5067	2.7740	3.1703	3.5748	2.0020	1.1228	4.2394	3.500544	4.128
0.9387	4.9542	7.4997	8.1032	1.7743	7.2830	8.3998	1.7124	3.5100	2.579
0.5348	1.1817	0.5243	0.2659	1.0218	1.0562	0.7529	1.4103	0.2509	1.494
0.8717	2.0741	1.4610	0.5200	2.9344	1.9662	2.1205	1.8057	0.9839	0.877
0.1816	1.9229	1.0498	0.5600	1.2783	1.0134	0.6253	0.1933	1.6470	0.688
0.1065	1.4253	0.3648	0.2003	0.4644	0.0927	0.0402	0.1031	1.8432	0.491
0.2632	0.0621	0.6004	0.2952	0.6125	1.1442	1.3542	0.3687	0.4984	0.219
0.0644	0.0560	1.3271	1.2179	0.7021	0.1166	0.3918	0.1455	0.1602	0.113
0.0296	0.0014	0.0036	0.0077	0.0062	0.0048	0.0044	0.0135	0.0003	0.003

Table 12: Absolue value of the vulnerability eigenvectors scaled by the eigenvalues, P_H and M_H are combined as they have the same eigenvalues

Then we multiplied each of the vectors by its corresponding eigenvalue and summed each of the rows. The eigenvalues represent the importance of each of the eigenvalues. By taking the proportion of each weight (Initial weight divided by total weights) we have the percentage values of each weight. All the components in the eigenvectors represent a factor that affects the vulnerability.

After this we determined the proportions of each factor by dividing the initial value given above by the total. Below is our final equation for the vulnerability score.

$S_i = \frac{v_i}{v_{max}} * 50 + 50$ where S_i is the vulnerability score for the ith neighborhood and v_i is the value for the ith neighborhood and v_{max} is the highest value out of the neighborhoods. A higher vulnerability score means that the neighborhood is more vulnerable.

After creating our equation we wanted to create a resource score, which can help the city determine which neighborhood needs the most resources diverted to it. We accomplished this by using the same method as described above, except adding factors relating to demographics in the respective neighborhoods.

After computing the PCA values we have the following equation for resource score:

$S_i = \frac{v_i}{v_{max}} * 50 + 50$ where S_i is the resource score for the ith neighborhood and v_i is the value for the ith neighborhood and v_{max} is the highest value out of the neighborhoods. A higher resource score means that the neighborhood has more resources, so it needs less

Variables	**Eigenvectors multiplied by Eigenvalues summed up**	**Proportional weights**
N_H	8.282963	0.060527
P_N	13.18483	0.096347
H_{2010}	15.60522	0.114034
H_{1990}	14.34078	0.104794
H_{1970}	12.36929	0.090388
H_{1950}	14.67964	0.107271
$H_{<1950}$	14.81246	0.108241
D_H	9.992278	0.073018
T_H	12.39486	0.090575
P_H	10.592	0.077400
M_H	10.592	0.077400

Table 13: Summation of scaled eigenvectors their proportional rate

Neightborhood	**Resource Score**
Downtown/South Main Arts District/South Bluffs	28.2831
Lakeland/Arlington/Brunswick	74.6408
Collierville/Piperton	100
⋮	⋮
Germantown, Zipcode 1	65.6563
Germantown, Zipcode 2	48.9859
South Riverdale	32.9456

Table 14: Resource scores for the neighborhoods

support.

4.4. Analyzing the Results

Based on our results for our vulnerability score, we have determined that East Memphis, Tennessee, is the at the most risk with a vulnerability score of 100. Contrary to this, Rossville has a vulnerability score of 9.99853204. These results are consistent with the data, as Rossville is a small town in Memphis which has a low number of buildings in proportion to open space and most of its homes built more recently. East Memphis, on the other hand, is much larger with a much higher proportion of developed, open space in neighborhood.

Based on the results for our resource score we can see that South Forum / Washington Heights needs the most resources diverted to it with a resource score of 14.406 and Collierville / Piperton needs the least with a resource score of 100. This is consistent with our results and data, along with our previous vulnerability score.

During heat waves, the city should prioritize resource allocation based on our given resource score, with a higher resource score meaning that the city has more resources and needs less resources. This would include deploying cooling centers and mobile cooling units to the most vulnerable areas, targeting public health messaging and outreach to vulnerable populations in those areas, prioritizing power restoration in those areas in case of outages, and increasing the number of emergency services in those areas.

4.5. Strengths and Weaknesses

4.5.1 Strengths

- Principal component analysis allows us to find the importance of the various factors without needing previous data of vulnerability scores and resource scores. Without PCA, we wouldn't have been able determine the importance factors since we were unable to run a regression.

- The PCA relies on data to determine vulnerability, making the assessment more objective and less prone to bias.

4.5.2 Weaknesses

- PCA does not give the most accurate values as it is a way to manipulate data and reduce data dimensions. Using PCA to determine feature importance isn't the most effective because the principal components in PCA are linear combinations of the original factors and not the factors themselves, which means we have to extract the factor importance from the eigenvectors.

- While the report provides vulnerability scores, it could offer a more in-depth interpretation of the principal components themselves. Understanding which factors contribute most to each component would provide valuable insights.

5. Conclusions

5.1. Further Studies

We have focused our model on Memphis as discussed in the initial problem. Future studies could expand this by developing similar models for other cities with varying climate and building stock characteristics, allowing for comparative analyses and broader applicability. We could also add more granular and niche data related to specific neighborhoods, which would increase the accuracy of the model. In addition to this, future research could assist us in incorporating dynamic occupancy models that account for variations in household schedules, thermostat preferences, and adaptive behaviors during heat waves. This would lead to more realistic power demand projections.

5.2. Summary

This report explores how heat waves are impacting households in Memphis, Tennessee. It employs various analyses to examine indoor temperatures, forecast power demand, and pinpoint neighborhoods that are most at risk. The use of a General Energy Balance Equation highlights how building features impact indoor heat levels during heat waves, with some designs exacerbating the situation. Additionally, the report employs a thermal equilibrium model to project a notable rise in future electricity demand due to heat waves. This underscores the necessity for new technologies and policies to address the anticipated increase in power requirements. Moreover, a Principal Component Analysis is used to identify neighborhoods that face the greatest risk during heat waves. This allows for targeted interventions, such as improving building designs or implementing community programs. Together, these models provide a comprehensive understanding of heat wave impacts and suggest ways to provide resources to impacted community. The report stresses the need for enhancing building efficiency, modernizing the power grid, and reducing neighborhood vulnerability to strengthen urban resilience against extreme heat.

6. References

[1] Hot Button Issue, MathWorks Math Modeling Challenge 2025, curated data, https://m3challenge.siam.o
[2] https://www.thempc.org/eagenda/x/mpc/2018/october-9-2018-regular-mpc-meeting/table-2-tn-2-dimensional-standards.pdf
[3] https://www.insulatekansascity.com/insulation-blog/inherited-an-old-house-heres-how-to-check-if-its-under-insulated/#:~:text=According%20to%20experts%20from%20Realtor,built%20with%20ins
[4] https://www.sanalifewellness.com/blog/air-changes-per-hour-ach
[5] https://www.fst.com/news-stories/magazine/renewable-energy/human-power-plant/
[6] https://codes.iccsafe.org/content/IRC2024P2/chapter-11-re-energy-efficiency
[7] https://mathresearch.utsa.edu
[8]https://www.sciencedirect.com/science/article/abs/pii/0306261977900174
[9] https://www.fst.com/news-stories/magazine/renewable-energy/human-power-plant/
[10] https://cleanair.camfil.us/2021/10/22/how-to-calculate-air-changes-per-hour
[11] https://www.sanalifewellness.com/blog/air-changes-per-hour-ach
[12] https://www.adt.com/resources/average-room-temperature: :text=That%20being%20said%2C%20the
[13] https://www.aceee.org/research-report/a2001
[14] https://www.vox.com/a/weather-climate-change-us-cities-global-warming
[15] https://www.tn.gov/commerce.html
[16] https://www.energy.gov/eere/wap/weatherization-assistance-program
[17] https://www.energy.gov/eere/buildings/appliance-and-equipment-standards-program

7. Appendix: Code Listed for Technical Computing Consideration

7.1. Question I

Question I

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.dates import DateFormatter
import datetime

# Load data from Excel files
accommodation_data = pd.read_excel("Accomadation_Memphis.xlsx", sheet_name="Sheet1")
weather_data = pd.read_excel("weather.xlsx", sheet_name="Sheet1")

# Process accommodation data
homes = []
for i in range(1, 5):  # Columns B to E correspond to Home 1 to Home 4
    home = {
        'name': f'Home {i}',
        'type': accommodation_data.iloc[1, i],  # Row 1: Accommodation type
        'year_built': accommodation_data.iloc[7, i],  # Row 7: Year structure built
        'shade': accommodation_data.iloc[9, i],  # Row 9: Shade assessment
        'size_sq_ft': accommodation_data.iloc[5, i],  # Row 5: Size of unit (square feet)
        'persons': accommodation_data.iloc[8, i],  # Row 8: Persons living in unit
    }
    homes.append(home)
```

```python
# Shading factor mapping
shade_map = {
    'Very shady': 0,
    'Not very shady': 0.67,
    'Not at all shady': 1,
}

# Calculate additional parameters for each home
for home in homes:
    home['size_sq_m'] = home['size_sq_ft'] / 10.764  # Convert square feet to square meters
    if 'Single-family' in home['type']:
        if home['year_built'] < 1970:
            a = 1
        elif 1970 <= home['year_built'] <= 2000:
            a = 0.7
        else:
            a = 0.25
    else:  # Apartment
        if home['year_built'] < 1970:
            a = 0.5
        elif 1970 <= home['year_built'] <= 2000:
            a = 0.35
        else:
            a = 0.25
    home['UA'] = a * home['size_sq_m']
    home['f_shade'] = shade_map[home['shade']]
    home['C'] = 75000
    home['Q_internal'] = home['persons'] * 100  # Internal heat gains in watts (W)

# Process weather data
weather_data['Temp_C'] = (weather_data['Temperature (F)'] - 32) * 5/9
weather_data['Temp_K'] = weather_data['Temp_C'] + 273.15  # Convert to Kelvin

# Extract hour from the Time column
if isinstance(weather_data['Time'].iloc[0], str):
    weather_data['hour'] = weather_data['Time'].apply(lambda x: int(x.split(':')[0]) if ':' in x else int(x))
else:
    weather_data['hour'] = weather_data['Time'].apply(lambda x: x.hour if hasattr(x, 'hour') else int(x))

weather_data['delta_t'] = weather_data['hour'].apply(lambda x: 3 if 6 <= x < 18 else 0)

# Simulate indoor temperature for each home (per-second steps)
for home in homes:
    UA = home['UA']
    f_shade = home['f_shade']
    C = home['C']
    Q_internal = home['Q_internal']  # In watts (W)
    A = home['size_sq_m']

    T_in = weather_data['Temp_K'].iloc[0]  # Initial indoor temperature in Kelvin
    results = []  # Store all relevant values for output

    print(home)
    for hour_idx in range(len(weather_data)):
        current_weather = weather_data.iloc[hour_idx]
        T_out = current_weather['Temp_K']
        delta_t = current_weather['delta_t']
        T_sa = T_out + f_shade * delta_t  # Sol-air temperature in Kelvin

        # Simulate 3600 seconds (1 hour)
        for _ in range(3600):
            ACH = 2 if T_out < T_in else 0.3  # Air changes per hour (ACH)
            ACH_per_second = ACH / 3600  # Convert ACH to per-second
```

```
            # Heat flows (W)
            Q_conduction = UA * (T_sa - T_in)
            Q_ventilation = ACH_per_second * 0.9 * A * (T_out - T_in)


            # Temperature change (K)
            dT = (Q_conduction + Q_internal/(home['size_sq_m']*3) + Q_ventilation) * 1 / C  # t = 1 second
            T_in += dT  # Update indoor temperature

        # Append hourly results
        results.append({
            "Time": current_weather["Time"],
            "Hour": hour_idx,  # Add numeric hour index for plotting
            "Outdoor_Temp_C": T_out - 273.15,
            "Sol_Air_Temp_C": T_sa - 273.15,
            "Q_Conduction_W": Q_conduction,
            "Q_Ventilation_W": Q_ventilation,
            "Q_Internal_W": Q_internal,
            "ACH": ACH,
            "Indoor_Temp_C": T_in - 273.15
        })

    home["results"] = results  # Store results in each home

# Output results
for home in homes:
    print(f"\n{home['name']} - Temperature Simulation Results:")
    print(f"{'Time':<10} {'Outdoor ( C )':<15} {'Sol-Air ( C )':<15} {'Q_Cond (W)':<15} {'Q_Vent (W)':<15} {'Q_Int (W)':<15} {'ACH':<10} {'Indoor ( C )':<15}")
    print("-" * 120)

    for entry in home["results"]:
        print(f"{entry['Time']} {entry['Outdoor_Temp_C']:<15.2f} {entry['Sol_Air_Temp_C']:<15.2f} {entry['Q_Conduction_W']:<15.2f} {entry['Q_Ventilation_W']:<15.2f} {entry['Q_Internal_W']:<15.2f} {entry['ACH']:<10.2f} {entry['Indoor_Temp_C']:<15.2f}")


def generate_temperature_graphs(homes):
    """
    Generate and save temperature graphs for each home and a combined comparison graph.

    Args:
        homes (list): List of home dictionaries containing simulation results
    """
    # Use the default style and set white backgrounds for figures and axes
    plt.style.use('default')
    plt.rcParams['figure.facecolor'] = 'white'
    plt.rcParams['axes.facecolor'] = 'white'

    # Create a figure for all homes comparison with white background
    plt.figure(figsize=(15, 8), facecolor='white')

    # Colors for each home
    colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728']

    # Use numeric hour indices for x-axis
    x_values = [entry['Hour'] for entry in homes[0]['results']]
    hour_labels = [f"{i}:00" for i in range(len(x_values))]

    # Plot indoor temperature for each home on the combined graph
    for i, home in enumerate(homes):
        indoor_temps = [entry['Indoor_Temp_C'] for entry in home['results']]
        plt.plot(x_values, indoor_temps, label=f"{home['name']}", color=colors[i], linewidth=2)

        # Create individual graph for each home with white background
```

```
        plt.figure(figsize=(12, 6), facecolor='white')

        indoor_temps = [entry['Indoor_Temp_C'] for entry in home['results']]
        outdoor_temps = [entry['Outdoor_Temp_C'] for entry in home['results']]
        sol_air_temps = [entry['Sol_Air_Temp_C'] for entry in home['results']]

        plt.plot(x_values, indoor_temps, label='Indoor Temperature', linewidth=2.5)
        plt.plot(x_values, outdoor_temps, label='Outdoor Temperature', linewidth=1.5, linestyle='--')
        plt.plot(x_values, sol_air_temps, label='Sol-Air Temperature', linewidth=1.5, linestyle=':')

        # Add home details as text annotation
        details = (f"Type: {home['type']}\n"
                   f"Year Built: {home['year_built']}\n"
                   f"Size: {home['size_sq_ft']} sq ft\n"
                   f"Shade: {home['shade']}\n"
                   f"Occupants: {home['persons']}")

        plt.annotate(details, xy=(0.02, 0.02), xycoords='axes fraction',
                     bbox=dict(boxstyle="round,pad=0.5", fc="white", alpha=0.8),
                     fontsize=9, verticalalignment='bottom')

        plt.title(f"Temperature Profile for {home['name']}", fontsize=16)
        plt.xlabel('Hour of Day', fontsize=12)
        plt.ylabel('Temperature ( C )', fontsize=12)
        plt.legend(loc='best')
        plt.grid(True, alpha=0.3)

        plt.xticks(x_values, hour_labels, rotation=45)
        plt.xlim(0, len(x_values)-1)

        plt.figtext(0.5, 0.01, f"UA: {home['UA']:.2f}, Shade Factor: {home['f_shade']}",
                   ha='center', fontsize=10, bbox=dict(boxstyle="round,pad=0.3", fc="white", alpha=0.8))

        plt.tight_layout()
        plt.savefig(f"{home['name'].replace(' ', '_')}_temperature.png", dpi=300)
        plt.close()

    # Return to the combined graph
    plt.figure(1)
    plt.title('Indoor Temperature Comparison Across All Homes', fontsize=16)
    plt.xlabel('Hour of Day', fontsize=12)
    plt.ylabel('Temperature ( C )', fontsize=12)
    plt.legend(loc='best')
    plt.grid(True, alpha=0.3)

    plt.xticks(x_values, hour_labels, rotation=45)
    plt.xlim(0, len(x_values)-1)

    # Add outdoor temperature to combined graph
    outdoor_temps = [entry['Outdoor_Temp_C'] for entry in homes[0]['results']]
    plt.plot(x_values, outdoor_temps, label='Outdoor Temperature', color='black', linewidth=1.5, linestyle='--')

    plt.legend(loc='best')
    plt.tight_layout()
    plt.savefig("All_Homes_Temperature_Comparison.png", dpi=300)
    plt.close()

    # Create a heat gain comparison chart with white background
    plt.figure(figsize=(15, 8), facecolor='white')

    for i, home in enumerate(homes):
        conduction = [entry['Q_Conduction_W'] for entry in home['results']]
        ventilation = [entry['Q_Ventilation_W'] for entry in home['results']]
        internal = [entry['Q_Internal_W'] for entry in home['results']]
```

```python

        total_heat = [c + v + i for c, v, i in zip(conduction, ventilation, internal)]

        plt.plot(x_values, total_heat, label=f"{home['name']} Total Heat Flow", color=colors[i], linewidth=2)

    plt.title('Total Heat Flow Comparison Across All Homes', fontsize=16)
    plt.xlabel('Hour of Day', fontsize=12)
    plt.ylabel('Heat Flow (W)', fontsize=12)
    plt.legend(loc='best')
    plt.grid(True, alpha=0.3)

    plt.xticks(x_values, hour_labels, rotation=45)
    plt.xlim(0, len(x_values)-1)

    plt.axhline(y=0, color='black', linestyle='--', alpha=0.3)
    plt.tight_layout()
    plt.savefig("All_Homes_Heat_Flow_Comparison.png", dpi=300)

    print("All graphs have been generated successfully!")

# Generate the graphs after running the simulation
generate_temperature_graphs(homes)
```

7.2. Question II

Question II

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from copy import deepcopy

# Load data from Excel files
try:
    accommodation_data = pd.read_excel("Accomadation_Memphis.xlsx", sheet_name="Sheet1")
    weather_data = pd.read_excel("weather.xlsx", sheet_name="Sheet1")
    print("Successfully loaded Excel files")
except Exception as e:
    print(f"Error loading Excel files: {e}")
    print("Using fallback data instead")
    # Use fallback data if Excel files can't be loaded
    weather_data = pd.DataFrame({
        'Time': [f"{hour:02d}:00:00" for hour in range(24)],
        'Temperature ( F )': [85, 85, 84, 83, 83, 83, 84, 88, 91, 94, 96, 97,
                              100, 100, 102, 102, 100, 99, 97, 94, 92, 91, 90, 89],
    })

# Process homes data
homes = []
try:
    for i in range(1, 5):  # Columns B to E correspond to Home 1 to Home 4
        home = {
            'name': f'Home {i}',
            'type': accommodation_data.iloc[1, i],  # Row 1: Accommodation type
            'year_built': accommodation_data.iloc[7, i],  # Row 7: Year structure built
            'shade': accommodation_data.iloc[9, i],  # Row 9: Shade assessment
            'size_sq_ft': accommodation_data.iloc[5, i],  # Row 5: Size of unit (square feet)
            'persons': accommodation_data.iloc[8, i],  # Row 8: Persons living in unit
        }
        homes.append(home)
except Exception as e:
    print(f"Error processing home data: {e}")
```

```
    print("Using fallback home data instead")
    # Use fallback data if accommodation data can't be processed
    homes = [
        {'name': 'Home 1', 'type': 'Single-family home', 'year_built': 2010, 'shade': 'Very shady', 'size_sq_ft': 88.26, 'persons': 3},
        {'name': 'Home 2', 'type': 'Single-family home', 'year_built': 1985, 'shade': 'Not very shady', 'size_sq_ft': 185.70, 'persons': 4},
        {'name': 'Home 3', 'type': 'Apartment', 'year_built': 2015, 'shade': 'Not at all shady', 'size_sq_ft': 88.26, 'persons': 2},
        {'name': 'Home 4', 'type': 'Apartment', 'year_built': 1960, 'shade': 'Not at all shady', 'size_sq_ft': 391.05, 'persons': 2}
    ]

# Shading factor mapping
shade_map = {
    'Very shady': 0,
    'Not very shady': 0.67,
    'Not at all shady': 1,
}

# Calculate additional parameters for each home
for home in homes:
    home['size_sq_m'] = home['size_sq_ft'] / 10.764  # Convert square feet to square meters
    if 'Single-family' in home['type']:
        if home['year_built'] < 1970:
            a = 1
        elif 1970 <= home['year_built'] <= 2000:
            a = 0.7
        else:
            a = 0.25
    else:  # Apartment
        if home['year_built'] < 1970:
            a = 0.5
        elif 1970 <= home['year_built'] <= 2000:
            a = 0.35
        else:
            a = 0.25
    home['UA'] = a * home['size_sq_m']
    home['f_shade'] = shade_map[home['shade']]
    home['C'] = 75000
    home['Q_internal'] = home['persons'] * 100  # Internal heat gains in watts (W)

# Process weather data
try:
    weather_data['Temp_C'] = (weather_data['Temperature (F)'] - 32) * 5/9
    weather_data['Temp_K'] = weather_data['Temp_C'] + 273.15  # Convert to Kelvin

    # Extract hour from the Time column
    if isinstance(weather_data['Time'].iloc[0], str):
        weather_data['hour'] = weather_data['Time'].apply(lambda x: int(x.split(':')[0]) if ':' in x else int(x))
    else:
        weather_data['hour'] = weather_data['Time'].apply(lambda x: x.hour if hasattr(x, 'hour') else int(x))

    weather_data['delta_t'] = weather_data['hour'].apply(lambda x: 5 if 6 <= x < 18 else 0)
except Exception as e:
    print(f"Error processing weather data: {e}")
    print("Recreating weather data with proper structure")
    # Create properly structured weather data if processing fails
    weather_data = pd.DataFrame({
        'Time': [f"{hour:02d}:00:00" for hour in range(24)],
        'Temp_C': [29.4, 29.4, 28.9, 28.3, 28.3, 28.3, 28.9, 31.1, 32.8, 34.4, 35.6, 36.1,
                   37.8, 37.8, 38.9, 38.9, 37.8, 37.2, 36.1, 34.4, 33.3, 32.8, 32.2, 31.7],
        'Temp_K': [t + 273.15 for t in [29.4, 29.4, 28.9, 28.3, 28.3, 28.3, 28.9, 31.1,
```

```python
        32.8, 34.4, 35.6, 36.1,
                                   37.8, 37.8, 38.9, 38.9, 37.8, 37.2, 36.1, 34.4, 33.3, 32.8, 32.2, 31.7]],
        'hour': list(range(24)),
        'delta_t': [0, 0, 0, 0, 0, 0, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 0, 0, 0, 0, 0, 0]
    })

# Memphis demographic data
MEMPHIS_POPULATION = 652236  # Population of Memphis
AVERAGE_HOUSEHOLD_SIZE = 2.5  # Average number of people per household
MEMPHIS_HOUSEHOLDS = MEMPHIS_POPULATION / AVERAGE_HOUSEHOLD_SIZE

# Housing distribution based on Memphis data
def calculate_housing_distribution():
    """Calculate housing distribution based on Memphis housing data"""
    # Distribution by housing type and age from data
    return [
        (0.321, 0),  # 32.1% mapped to Home 1
        (0.415, 1),  # 41.5% mapped to Home 2
        (0.110, 2),  # 11.0% mapped to Home 3
        (0.154, 3),  # 15.4% mapped to Home 4
    ]

HOUSING_DISTRIBUTION = calculate_housing_distribution()

def calculate_baseline_power():
    """Calculate baseline power consumption for Memphis excluding AC"""
    # From the data:
    # Memphis annual usage per household: 15,172 kWh
    annual_usage_per_household = 15172  # kWh

    # Total number of households in Memphis/Shelby County
    total_households = MEMPHIS_HOUSEHOLDS

    # Calculate total annual residential consumption
    annual_residential_kwh = annual_usage_per_household * total_households

    # From the data, Shelby County annual consumption (2022): 9,768,296,000 kWh
    total_annual_consumption = 9768296000  # kWh

    # Calculate average power in watts
    hours_in_year = 365 * 24
    avg_power_w = (total_annual_consumption * 1000) / hours_in_year

    # Summer baseline is higher than annual average
    summer_non_ac_factor = 1.15  # 15% higher baseline during summer

    # Adjust baseline to align with target power values
    summer_baseline_w = avg_power_w * summer_non_ac_factor

    return summer_baseline_w * 1.5  # Scaling factor to align with expected city-wide impact

def run_sensitivity_analysis(base_params, homes, weather_data, housing_distribution, memphis_households):
    """
    Run sensitivity analysis on key model parameters

    Args:
        base_params: Dictionary with baseline parameters
        homes: List of home dictionaries with thermal properties
        weather_data: DataFrame with weather information
        housing_distribution: List of (percentage, home_index) tuples
        memphis_households: Number of households in Memphis

    Returns:
        Dictionary with sensitivity analysis results
```

```
    """
    print("Running sensitivity analysis...")

    # Define parameter ranges to test
    param_ranges = {
        "AC_SETPOINT_TEMP_C": [19, 20, 21, 22, 23, 24, 25],
        "AC_POWER_RATING": [6000, 7000, 8000, 9000, 10000],
        "AC_EFFICIENCY": [1.9, 2.1, 2.3, 2.5, 2.7],
        "AC_LOAD_SCALE_FACTOR": [50, 60, 70, 80, 90]
    }

    # Capacity factor scenarios
    capacity_factor_scenarios = [
        {
            "name": "All high efficiency",
            "factors": {
                'Home 1': 1.0,
                'Home 2': 0.95,
                'Home 3': 0.9,
                'Home 4': 0.85
            }
        },
        {
            "name": "Baseline",
            "factors": base_params['AC_CAPACITY_FACTOR']
        },
        {
            "name": "All low efficiency",
            "factors": {
                'Home 1': 0.9,
                'Home 2': 0.8,
                'Home 3': 0.7,
                'Home 4': 0.6
            }
        },
        {
            "name": "High variation",
            "factors": {
                'Home 1': 1.0,
                'Home 2': 0.9,
                'Home 3': 0.7,
                'Home 4': 0.5
            }
        }
    ]

    # Initialize results containers
    sensitivity_results = {}

    # Run baseline model first
    baseline_results = simulate_power_model(
        base_params,
        homes,
        weather_data,
        housing_distribution,
        memphis_households
    )

    print(f"\nBaseline results:")
    print(f"  Peak Power: {baseline_results['peak_power_mw']:.2f} MW")
    print(f"  AC Contribution: {baseline_results['ac_contribution_mw']:.2f} MW ({↩
        baseline_results['ac_percentage']:.1f}%)")

    # Single parameter variations
    for param_name, param_values in param_ranges.items():
        results = []
        print(f"\nTesting {param_name}...")

```

```
        for value in param_values:
            # Update just this parameter
            params = base_params.copy()
            params[param_name] = value

            # Run model
            model_results = simulate_power_model(
                params,
                homes,
                weather_data,
                housing_distribution,
                memphis_households
            )

            # Store results
            results.append({
                "param_value": value,
                "peak_power_mw": model_results["peak_power_mw"],
                "ac_percentage": model_results["ac_percentage"]
            })

            print(f"  {param_name} = {value}: Peak Power = {model_results['peak_power_mw']:.2f} MW, AC = {model_results['ac_percentage']:.1f}%")

        sensitivity_results[param_name] = results

    # Capacity factor scenarios
    capacity_results = []
    print("\nTesting capacity factor scenarios...")

    for scenario in capacity_factor_scenarios:
        # Update capacity factors
        params = base_params.copy()
        params["AC_CAPACITY_FACTOR"] = scenario["factors"]

        # Run model
        model_results = simulate_power_model(
            params,
            homes,
            weather_data,
            housing_distribution,
            memphis_households
        )

        # Store results
        capacity_results.append({
            "scenario": scenario["name"],
            "peak_power_mw": model_results["peak_power_mw"],
            "ac_percentage": model_results["ac_percentage"]
        })

        print(f"  {scenario['name']}: Peak Power = {model_results['peak_power_mw']:.2f} MW, AC = {model_results['ac_percentage']:.1f}%")

    sensitivity_results["capacity_factor"] = capacity_results

    # Plot results
    plot_sensitivity_results(sensitivity_results, baseline_results)

    return sensitivity_results

def simulate_power_model(params, original_homes, original_weather, housing_distribution, memphis_households, temp_increase=0):
    """
    Run the Memphis power model with specific parameters

    Args:
        params: Dictionary of model parameters
```

```
        original_homes: List of home dictionaries
        original_weather: DataFrame with weather information
        housing_distribution: List of (percentage, home_index) tuples
        memphis_households: Number of households in Memphis
        temp_increase: Optional temperature increase in  C

    Returns:
        Dictionary with simulation results
    """
    # Get parameters
    ac_power_rating = params.get('AC_POWER_RATING')
    ac_setpoint_temp_c = params.get('AC_SETPOINT_TEMP_C')
    ac_efficiency = params.get('AC_EFFICIENCY')
    ac_capacity_factor = params.get('AC_CAPACITY_FACTOR')
    ac_load_scale_factor = params.get('AC_LOAD_SCALE_FACTOR')

    # Make deep copies to avoid modifying originals
    homes = deepcopy(original_homes)
    weather_data = original_weather.copy()

    # Apply temperature increase if specified
    if temp_increase > 0:
        weather_data = weather_data.copy()
        weather_data["Temp_C"] += temp_increase
        weather_data["Temp_K"] = weather_data["Temp_C"] + 273.15

    # Simulate indoor temperature for each home
    for home in homes:
        UA = home['UA']
        f_shade = home['f_shade']
        C = home['C']
        Q_internal = home['Q_internal']
        A = home['size_sq_m']

        T_in = ac_setpoint_temp_c + 273.15
        results = []

        # Get capacity factor for this home type
        capacity_factor = ac_capacity_factor[home['name']]

        for hour_idx in range(len(weather_data)):
            current_weather = weather_data.iloc[hour_idx]
            T_out = current_weather['Temp_K']
            delta_t = current_weather['delta_t']
            T_sa = T_out + f_shade * delta_t

            # AC parameters
            AC_setpoint_K = ac_setpoint_temp_c + 273.15

            # Calculate heat gains
            Q_conduction_gain = UA * (T_sa - AC_setpoint_K)
            Q_internal_gain = Q_internal / (home['size_sq_m'] * 3)
            ACH = 0.3
            ACH_per_second = ACH / 3600
            Q_ventilation_gain = ACH_per_second * 0.9 * A * (T_out - AC_setpoint_K)

            # Total heat gain that AC must remove
            Q_total_gain = Q_conduction_gain + Q_internal_gain + Q_ventilation_gain

            # Required cooling power
            required_cooling_power = max(0, Q_total_gain)

            # Max cooling capacity
            max_cooling_capacity = ac_power_rating * ac_efficiency * capacity_factor

            # Required electrical power
            required_electrical_power = required_cooling_power / ac_efficiency

```

```
            # Actual AC power usage
            ac_power_usage = min(required_electrical_power, ac_power_rating * ↩
                capacity_factor)
            actual_cooling_power = ac_power_usage * ac_efficiency

            # Can AC maintain setpoint?
            can_maintain_setpoint = actual_cooling_power >= required_cooling_power

            # Calculate indoor temperature
            actual_T_in = T_in

            if not can_maintain_setpoint and required_cooling_power > 0:
                # Calculate equilibrium temperature
                a = UA + (ACH_per_second * 0.9 * A)
                b = UA * T_sa + (ACH_per_second * 0.9 * A * T_out) + Q_internal_gain - ↩
                    actual_cooling_power

                equilibrium_T_in = b / a

                # Using Newton's cooling for temperature transition
                cooling_rate = 1/C
                temp_change = (equilibrium_T_in - T_in) * (1 - np.exp(-cooling_rate * ↩
                    3600))

                actual_T_in = T_in + temp_change
            else:
                actual_T_in = AC_setpoint_K

            # Update temperature for next hour
            T_in = actual_T_in

            # Store results
            results.append({
                "Time": current_weather["Time"],
                "Hour": hour_idx,
                "Outdoor_Temp_C": T_out - 273.15,
                "Indoor_Temp_C": T_in - 273.15,
                "AC_Power_W": ac_power_usage
            })

        home["results"] = results

    # Calculate baseline power
    baseline_power = calculate_baseline_power()

    # Calculate hourly power consumption
    hourly_power = []

    for hour in range(len(weather_data)):
        # Calculate total AC power for this hour
        total_ac_power = 0
        for dist_pct, home_idx in housing_distribution:
            home = homes[home_idx]
            ac_power = home["results"][hour]["AC_Power_W"]
            households_of_type = memphis_households * dist_pct
            total_ac_power += ac_power * households_of_type

        # Apply scaling factor
        total_ac_power *= ac_load_scale_factor

        # Total power
        total_power = baseline_power + total_ac_power

        hourly_power.append({
            "Hour": hour,
            "Baseline_Power_MW": baseline_power / 1e6,
            "AC_Power_MW": total_ac_power / 1e6,
            "Total_Power_MW": total_power / 1e6
```

```
        })

    # Find peak power
    power_df = pd.DataFrame(hourly_power)
    peak_power = power_df["Total_Power_MW"].max()
    peak_hour_idx = power_df["Total_Power_MW"].idxmax()
    peak_hour_data = power_df.iloc[peak_hour_idx]

    # Calculate AC percentage
    ac_percentage = peak_hour_data["AC_Power_MW"] / peak_hour_data["Total_Power_MW"] * 100

    # Return key results
    return {
        "peak_power_mw": peak_power,
        "peak_hour": peak_hour_data["Hour"],
        "ac_contribution_mw": peak_hour_data["AC_Power_MW"],
        "baseline_power_mw": peak_hour_data["Baseline_Power_MW"],
        "ac_percentage": ac_percentage
    }

def plot_sensitivity_results(results, baseline):
    """
    Create visualizations of sensitivity analysis results
    """
    # Set up the figure
    plt.figure(figsize=(20, 15))
    plt.suptitle("Sensitivity Analysis of Memphis Power Model", fontsize=16)

    # Plot continuous parameters
    continuous_params = ["AC_SETPOINT_TEMP_C", "AC_POWER_RATING", "AC_EFFICIENCY", "AC_LOAD_SCALE_FACTOR"]

    for i, param in enumerate(continuous_params):
        # Extract results for this parameter
        param_results = results[param]
        param_values = [r["param_value"] for r in param_results]
        peak_powers = [r["peak_power_mw"] for r in param_results]
        ac_percentages = [r["ac_percentage"] for r in param_results]

        # Plot peak power
        plt.subplot(3, 2, i+1)
        plt.plot(param_values, peak_powers, 'b-o', linewidth=2)
        plt.axhline(y=baseline["peak_power_mw"], color='r', linestyle='--', label='Baseline')

        # Format labels based on parameter
        if param == "AC_SETPOINT_TEMP_C":
            plt.xlabel("AC Setpoint Temperature ( C )", fontsize=12)
        elif param == "AC_POWER_RATING":
            plt.xlabel("AC Power Rating (W)", fontsize=12)
        elif param == "AC_EFFICIENCY":
            plt.xlabel("AC Coefficient of Performance", fontsize=12)
        elif param == "AC_LOAD_SCALE_FACTOR":
            plt.xlabel("AC Load Scaling Factor", fontsize=12)

        plt.ylabel("Peak Power Demand (MW)", fontsize=12)
        plt.title(f"Effect of {param} on Peak Power", fontsize=14)
        plt.grid(True, alpha=0.3)
        plt.legend()

        # Plot AC percentage
        plt.subplot(3, 2, i+3)
        plt.plot(param_values, ac_percentages, 'g-o', linewidth=2)
        plt.axhline(y=baseline["ac_percentage"], color='r', linestyle='--', label='Baseline')

        # Same x-label
        if param == "AC_SETPOINT_TEMP_C":
```

```python
            plt.xlabel("AC Setpoint Temperature ( C )", fontsize=12)
        elif param == "AC_POWER_RATING":
            plt.xlabel("AC Power Rating (W)", fontsize=12)
        elif param == "AC_EFFICIENCY":
            plt.xlabel("AC Coefficient of Performance", fontsize=12)
        elif param == "AC_LOAD_SCALE_FACTOR":
            plt.xlabel("AC Load Scaling Factor", fontsize=12)

        plt.ylabel("AC Contribution (%)", fontsize=12)
        plt.title(f"Effect of {param} on AC Percentage", fontsize=14)
        plt.grid(True, alpha=0.3)
        plt.legend()

    # Plot capacity factor scenarios
    capacity_results = results["capacity_factor"]
    scenarios = [r["scenario"] for r in capacity_results]
    peak_powers = [r["peak_power_mw"] for r in capacity_results]
    ac_percentages = [r["ac_percentage"] for r in capacity_results]

    plt.subplot(3, 1, 3)
    x = np.arange(len(scenarios))
    width = 0.35

    ax1 = plt.gca()
    bars1 = ax1.bar(x - width/2, peak_powers, width, label='Peak Power (MW)', color='b')
    ax1.set_ylabel('Peak Power (MW)', fontsize=12)
    ax1.set_xticks(x)
    ax1.set_xticklabels(scenarios, rotation=45, ha='right')

    ax2 = ax1.twinx()
    bars2 = ax2.bar(x + width/2, ac_percentages, width, label='AC Percentage (%)', color='g')
    ax2.set_ylabel('AC Contribution (%)', fontsize=12)

    ax1.set_title('Effect of Capacity Factor Scenarios', fontsize=14)
    ax1.legend(loc='upper left')
    ax2.legend(loc='upper right')

    plt.tight_layout()
    plt.savefig('memphis_power_sensitivity.png', dpi=300)
    plt.show()

def project_2044_power_demand(base_params, homes, original_weather, housing_distribution, memphis_households):
    """
    Project power demand for 2044 under different climate and technology scenarios

    Args:
        base_params: Dictionary with baseline parameters
        homes: List of home dictionaries
        original_weather: DataFrame with weather information
        housing_distribution: List of (percentage, home_index) tuples
        memphis_households: Number of households in Memphis

    Returns:
        List of dictionaries with projection results
    """
    # Define scenarios for 2044
    scenarios = [
        {
            "name": "Current Conditions (Baseline)",
            "params": base_params.copy(),
            "temp_increase": 0  # No change
        },
        {
            "name": "Climate Change Only",
            "params": {
                **base_params,
```

```
                'AC_LOAD_SCALE_FACTOR': base_params['AC_LOAD_SCALE_FACTOR'] * 1.3  # 30% increase in cooling demand
            },
            "temp_increase": 2.0  # 2 C  warmer
        },
        {
            "name": "Tech Improvement Only",
            "params": {
                **base_params,
                'AC_POWER_RATING': base_params['AC_POWER_RATING'] * 0.9,  # Smaller units needed
                'AC_EFFICIENCY': base_params['AC_EFFICIENCY'] * 1.4,  # 40% more efficient
                'AC_CAPACITY_FACTOR': {  # Better capacity across all homes
                    'Home 1': 1.0,
                    'Home 2': 0.98,
                    'Home 3': 0.95,
                    'Home 4': 0.9
                },
                'AC_LOAD_SCALE_FACTOR': base_params['AC_LOAD_SCALE_FACTOR'] * 0.7  # Reduced by 30% due to efficiency
            },
            "temp_increase": 0  # No change
        },
        {
            "name": "Climate Change + Tech Improvement",
            "params": {
                **base_params,
                'AC_POWER_RATING': base_params['AC_POWER_RATING'] * 0.9,
                'AC_EFFICIENCY': base_params['AC_EFFICIENCY'] * 1.4,
                'AC_CAPACITY_FACTOR': {
                    'Home 1': 1.0,
                    'Home 2': 0.98,
                    'Home 3': 0.95,
                    'Home 4': 0.9
                },
                'AC_LOAD_SCALE_FACTOR': base_params['AC_LOAD_SCALE_FACTOR']  # Combined effects roughly cancel out
            },
            "temp_increase": 2.0  # 2 C  warmer
        }
    ]

    results = []
    print("\nProjecting 2044 Power Demand Scenarios:")

    for scenario in scenarios:
        # Run model with scenario parameters and temperature increase
        model_results = simulate_power_model(
            scenario["params"],
            homes,
            original_weather,
            housing_distribution,
            memphis_households,
            scenario["temp_increase"]
        )

        # Store results
        results.append({
            "scenario": scenario["name"],
            "peak_power_mw": model_results["peak_power_mw"],
            "ac_contribution_mw": model_results["ac_contribution_mw"],
            "ac_percentage": model_results["ac_percentage"],
            "temp_increase": scenario["temp_increase"]
        })

        print(f"  {scenario['name']}:")
        print(f"    - Peak Power: {model_results['peak_power_mw']:.2f} MW")
        print(f"    - AC Contribution: {model_results['ac_contribution_mw']:.2f} MW ({
```

```
            model_results['ac_percentage']:.1f}%)")

    # Plot results
    plt.figure(figsize=(12, 8))

    # Extract data for plotting
    scenario_names = [r["scenario"] for r in results]
    peak_powers = [r["peak_power_mw"] for r in results]
    ac_contributions = [r["ac_contribution_mw"] for r in results]
    temp_increases = [r["temp_increase"] for r in results]

    # Create bar plot
    x = np.arange(len(scenario_names))
    width = 0.35

    fig, ax1 = plt.subplots(figsize=(14, 8))
    ax1.bar(x, peak_powers, width, label='Total Peak Power', color='blue', alpha=0.7)
    ax1.bar(x, ac_contributions, width, label='AC Contribution', color='red', alpha=0.7)

    # Add temperature increase indicators
    for i, temp in enumerate(temp_increases):
        if temp > 0:
            ax1.annotate(f"+{temp} C ", xy=(i, peak_powers[i] + 50), ha='center', va='bottom',
                        bbox=dict(boxstyle="round,pad=0.3", fc="yellow", alpha=0.7))

    # Customize plot
    ax1.set_ylabel('Power (MW)', fontsize=14)
    ax1.set_title('Projected Memphis Power Demand in 2044', fontsize=16)
    ax1.set_xticks(x)
    ax1.set_xticklabels(scenario_names, rotation=45, ha='right', fontsize=12)
    ax1.legend(loc='upper left', fontsize=12)
    ax1.grid(axis='y', alpha=0.3)

    # Add baseline line
    ax1.axhline(y=results[0]["peak_power_mw"], color='k', linestyle='--', alpha=0.5, label='Current Baseline')

    # Add percentage labels on bars
    for i, v in enumerate(peak_powers):
        ac_pct = (ac_contributions[i] / v) * 100
        ax1.text(i, v/2, f"{ac_pct:.1f}%", color='white', fontweight='bold', ha='center')

    plt.tight_layout()
    plt.savefig('memphis_power_2044_projection.png', dpi=300)
    plt.show()

    return results

# Create a parameters dictionary
base_params = {
    'AC_POWER_RATING': 8000,
    'AC_SETPOINT_TEMP_C': 21,
    'AC_EFFICIENCY': 2.3,
    'AC_CAPACITY_FACTOR': {
        'Home 1': 1.0,
        'Home 2': 0.9,
        'Home 3': 0.8,
        'Home 4': 0.7
    },
    'AC_LOAD_SCALE_FACTOR': 70.0
}

# Run the sensitivity analysis
sensitivity_results = run_sensitivity_analysis(
    base_params,
    homes,
    weather_data,
```

```
    HOUSING_DISTRIBUTION,
    MEMPHIS_HOUSEHOLDS
)

# Project power demand for 2044
projection_2044 = project_2044_power_demand(
    base_params,
    homes,
    weather_data,
    HOUSING_DISTRIBUTION,
    MEMPHIS_HOUSEHOLDS
)
```