Using Solar Energy to Address Global Warming and Sustainability

New Mexico

Supercomputing Challenge

April 3rd, 2021

Team 9

Capital High School

Team Members:

Christopher Mendoza Castillo

Zachariah Burch

Joshua Mari Tamarra

Teacher:

Barbara Teterycz

Mentors:

Barbara Teterycz and David Ritter

Table of Contents

Abstract/Executive Summary ……………………..……………………………… 2

Methodology/Research/Results ………………………...……………………… 3

Conclusion ………………………………………………………………………. 12

Future Work ……………………………………………………………………... 13

Acknowledgements ……………………………………………………….... 14

References …………………………………………………………...….. 15

Appendix 1 ………………………………………………………………… 17

Appendix 2 ………………………………………………………………… 20

Appendix 3 ………………………………………………………………… 26

Abstract/Executive Summary

Since burning fossil fuels causes global warming that results in many damages to the environment (such as wildfires, coastal flooding, loss of sea ice and increased sea level, as well as constantly increasing average annual temperatures), and because many experts believe that the world has already used oil reserves at an unsustainable rate, the main focus of this project was to learn how to address and solve these problems.

Based on the research as well as the learning experience gained from working with the professional mentors, such as a retired electrical engineer from Silicon Valley, Mr. David Ritter as well as our Computer Science Principles teacher and a former office manager of the PV business, Barbara Teterycz, the solution to the problems mentioned above is renewable energy. It is because this type of energy is considered to be clean, which means that it does not contribute to the carbon footprint, but instead it actually helps decrease it. In addition, as its name indicates, this type of energy is unlimited and has a variety of forms, such as hydropower, wind power, and solar power.

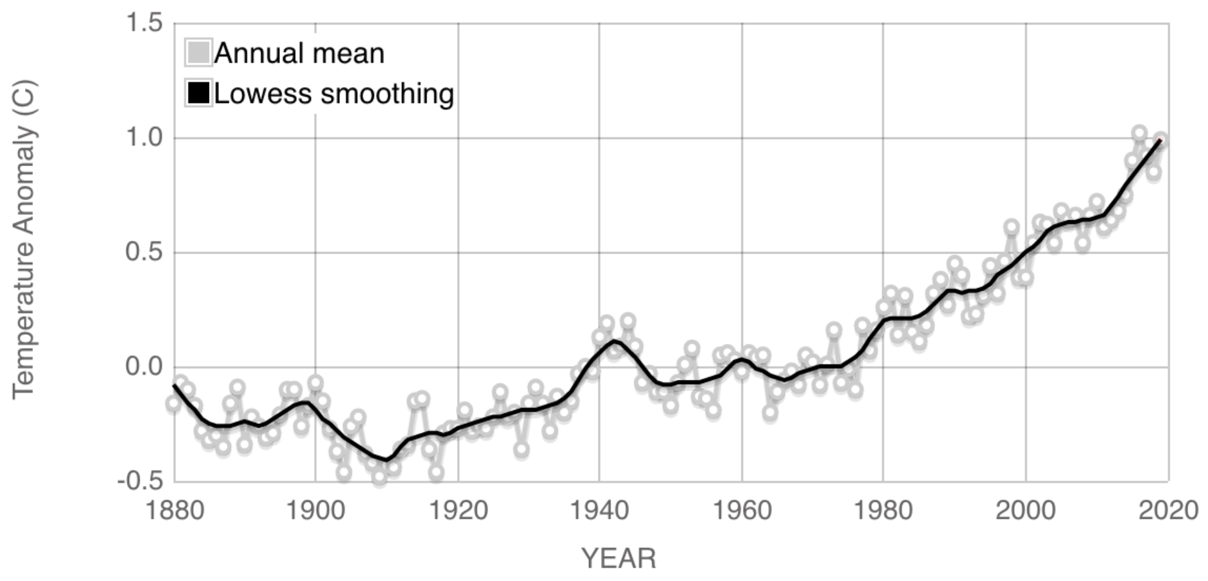
Because of the natural resources which are present here, in New Mexico state (such as a lot of sunny days and open space) this last (but not least) type of renewable energy became the main point of interest. While working on this project, we've learned how to determine the size, annual and monthly production, price, and payoff time of the PV systems, and how to write both JavaScript and Python programs that output all these details. We've also learned how the efficiency of the solar panels change depending on the degree of the angle, at which they are tilted towards the sun. Following this idea, we wrote a mathematical formula and a computational model to graphically represent the dependency of the panel's efficiency on the tilt angle in the form of a line chart.

The purpose of this project is to help our local community better understand the importance, benefits, types, and functionality of the PV systems, and how their overall efficiency

can be improved by their correct installation. After submitting our first computational artifact (a mobile app written using JavaScript) to the Congressional App Challenge, it was recognized by the senator Ben Lujan, who awarded it with the first place.

Methodology/Research/Results

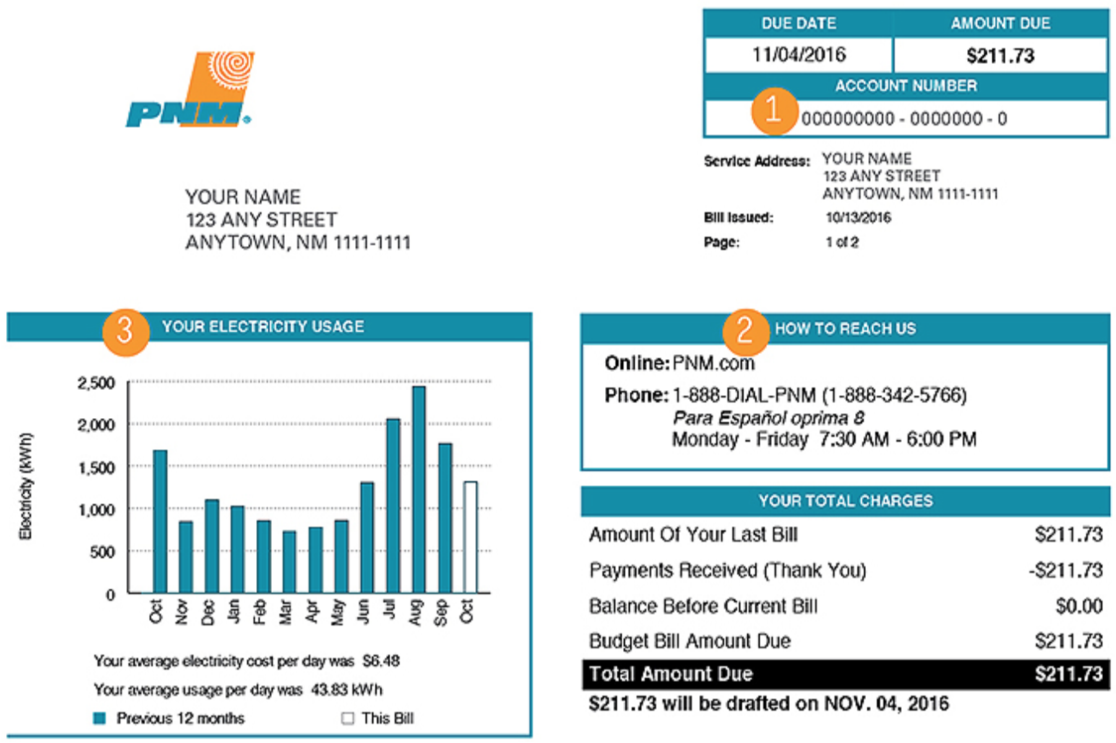
Fossil fuels were formed over millions of years by the decomposing remains of plants and animals under extreme heat and pressure. In order to release their stored energy, fossil fuels had to be burned, and during this combustion process a variety of emissions and particulates have been released into the atmosphere, where they combined with water vapor and caused acid rain. In addition to this pollution, burning fossil fuels has also released carbon dioxide (known as greenhouse gas) that has greatly contributed to global climate change. It's because this greenhouse gas has absorbed the heat from the sun causing the temperature of the Earth's surface and its lower atmosphere to keep increasing (The Environmental Literacy Council). As shown on the graph from NASA, nineteen of the twenty warmest years all have occurred since 2001, with the exception of 1998 (NASA's Jet Propulsion Laboratory):



But global warming and pollution have not been the only problems. Even though they are very harmful to the environment, the world continues using fossil fuels, because it needs energy. However, they are not unlimited. According to the research, many experts believe that the world has already reached the peak for oil extraction (The Environmental Literacy Council). At this stage of the research, we identified a more complex problem that the world faces in reference to the current sources of energy, which appeared to be both detrimental and limited.

Since there are about 325 sunny days here in Santa Fe, the solution to this complex problem is the use of renewable sources of energy (such as solar PV power) on a much bigger scale (Santa Fe Convention Center). However, in order to accomplish this goal, both the community leaders and members have to be aware of their importance, benefits and their functionality.

In order to determine the size of the PV solar system for a given household, professional PV installers ask their clients for their electricity usage, which can be found on their utility bill (PNM):



PNM

YOUR NAME
123 ANY STREET
ANYTOWN, NM 1111-1111

DUE DATE	AMOUNT DUE
11/04/2016	$211.73

ACCOUNT NUMBER
1 000000000 - 0000000 - 0

Service Address:	YOUR NAME 123 ANY STREET ANYTOWN, NM 1111-1111
Bill Issued:	10/13/2016
Page:	1 of 2

3 YOUR ELECTRICITY USAGE

Electricity (kWh)
2,500
2,000
1,500
1,000
500
0
Oct
Nov
Dec
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct

Your average electricity cost per day was $6.48
Your average usage per day was 43.83 kWh
■ Previous 12 months ☐ This Bill

2 HOW TO REACH US

Online: PNM.com
Phone: 1-888-DIAL-PNM (1-888-342-5766)
Para Español oprima 8
Monday - Friday 7:30 AM - 6:00 PM

YOUR TOTAL CHARGES

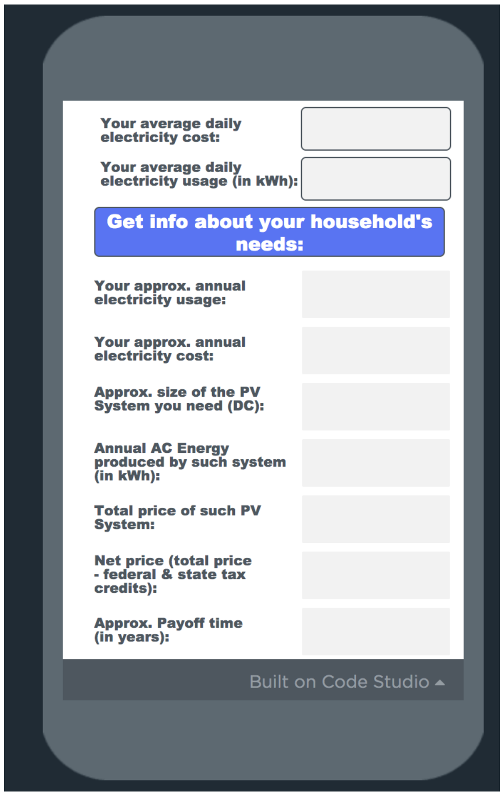
Amount Of Your Last Bill	$211.73
Payments Received (Thank You)	-$211.73
Balance Before Current Bill	$0.00
Budget Bill Amount Due	$211.73
Total Amount Due	**$211.73**

$211.73 will be drafted on NOV. 04, 2016

Next, they use PV Watts Calculator (an online tool created by the National Renewable Energy Lab) to find out what size of the PV system produces enough AC energy to cover the electricity needs of a given household in a specific geographical area and for each of these two types of the system: roof and ground mount (NREL). Also, since there are various types of the PV systems, in order to help homeowners pick the best type for their house, PV installers propose roof mount systems for the houses with small or no yards, whereas the households with larger yards are advised to choose open rack systems because they do not interfere with the maintenance of the roof and are easier to get access to (if there is a need). And finally, since a standard residential polycrystalline solar panel produces 275 watts energy, in order to determine the approximate number of panels needed, PV installers divide the size of the PV system by 275 (SolarReviews).

Since the average cost of the solar system in the U.S. is $2.91 per watt (according to EnergySage that connects homeowners with solar installers), multiplying the size of the PV system by this rate would result in an approximate total cost of the system (EnergySage). However, since there is a federal (26%) and a state (10%) income tax credit available, the net cost of the PV system is the total cost minus these tax credits (U.S. Department of Energy). And finally, in order to determine the payoff time of the system, which means the number of years needed to pay off the system from what the homeowner will save on the utility bills, the net cost of the system has to be divided by the annual electricity cost. In addition, since utilities are obligated by law to prove that a certain percentage of their energy comes from renewable sources, many of them purchase Renewable Energy Certificates (REC) for each kilowatt-hour produced by the PV system (PNM). Also, connecting the PV system to the utility grid (called interconnection) results in net metering, which causes the utility meter to spin backwards when the excess of energy produced by the system is sent to the grid (U.S. Department of Energy).

Most of these details and facts related to the PV systems mentioned above have been included in two computer programs. The first one, a mobile app written using JavaScript is a useful and handy tool that informs the local community members about the best size of the PV system for their house as well as its price and payoff time (Appendix 1).



Your average daily electricity cost:

Your average daily electricity usage (in kWh):

Get info about your household's needs:

Your approx. annual electricity usage:

Your approx. annual electricity cost:

Approx. size of the PV System you need (DC):

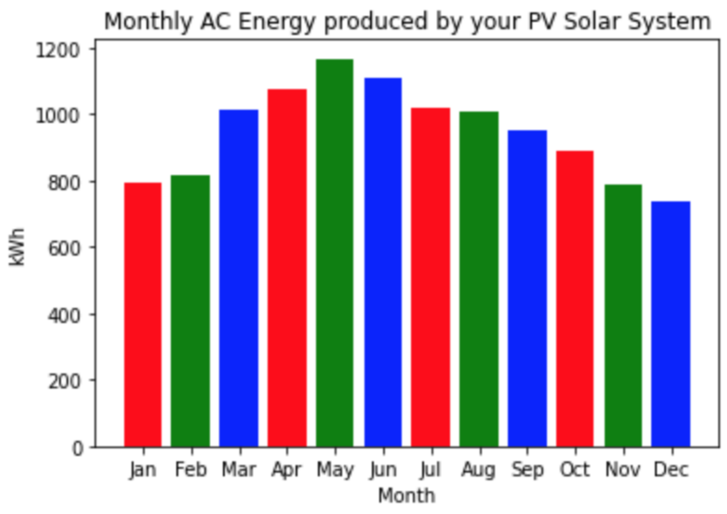
Annual AC Energy produced by such system (in kWh):

Total price of such PV System:

Net price (total price - federal & state tax credits):

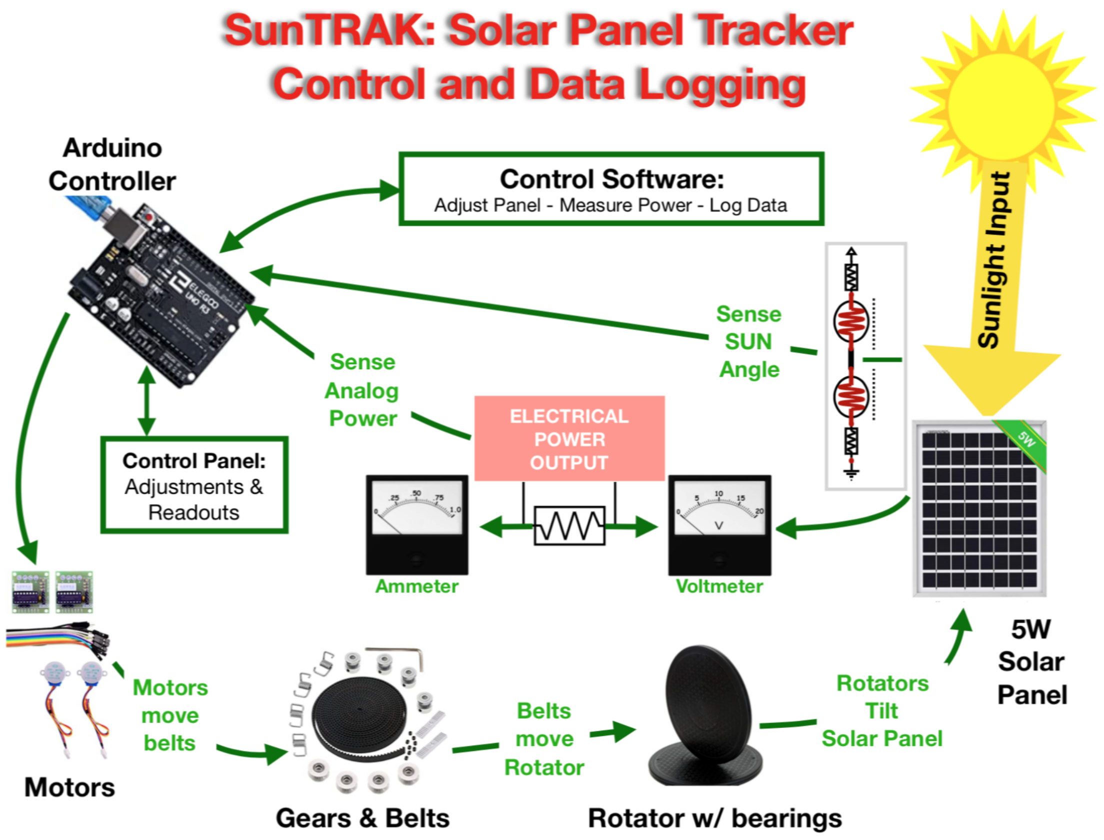
Approx. Payoff time (in years):

The second program, written using Python, presents monthly AC energy production in the form of a bar chart (Appendix 2).

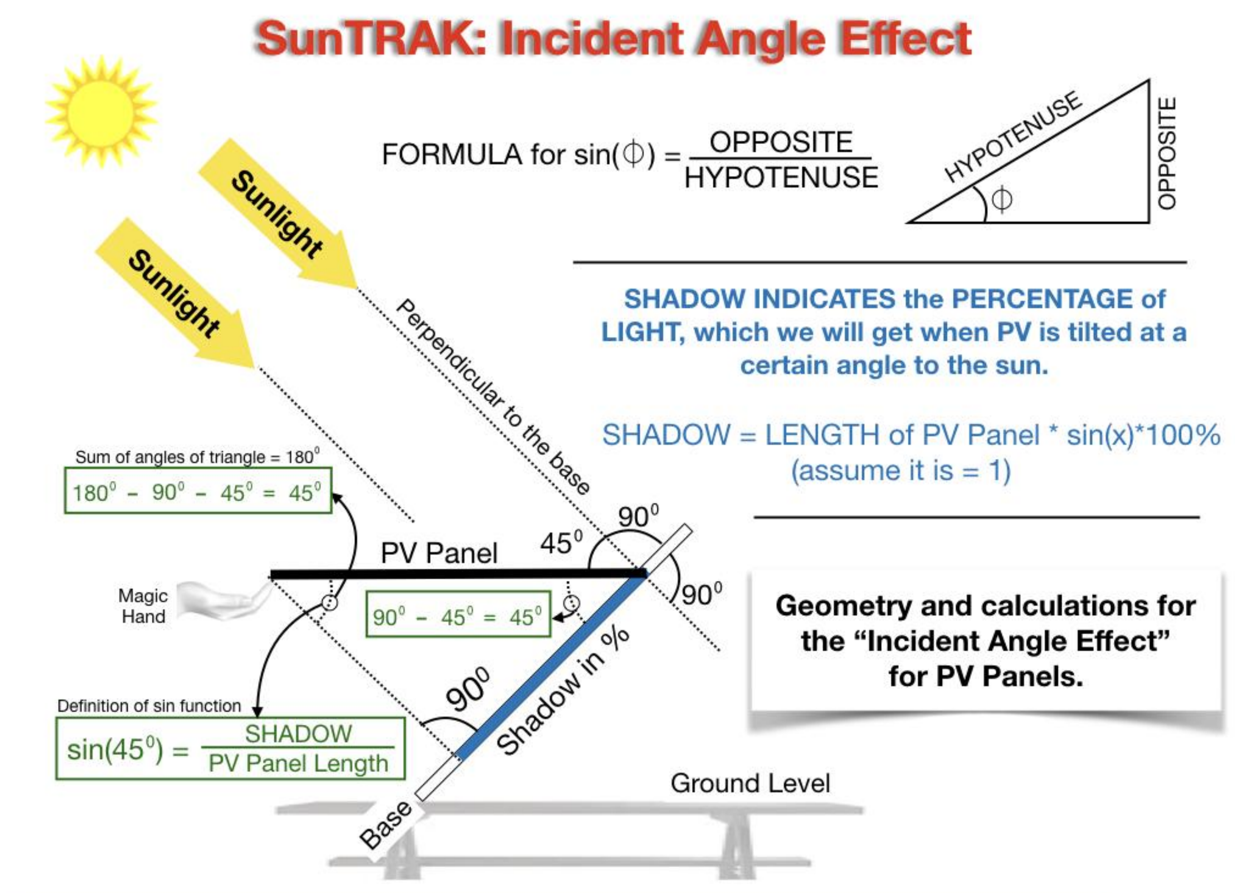


Based on this chart, the maximum production of electricity takes place in the month of May here, in Santa Fe. Since the longest days with the shortest shadows occur at the end of June, this outcome was very surprising. This fact required making additional research, which led to the conclusion that the monsoon season starts here in the middle of June each year resulting in many cloudy days. If this is the case, it would definitely explain the drop in the production of AC energy in June.

In addition to all these important details about PV systems, it’s also important to know how they work. PV panels are made from silicon that absorbs sunlight, which then knocks electrons loose allowing them to flow freely (HowStuffWorks). Since they all flow in one direction, they are called direct current (DC). However, since utility grids and wall sockets deliver an alternating current (AC) that changes direction at regular intervals, each PV system needs an inverter to convert DC into AC (Wester). The image below presents the prototype of the PV system that was used for the purpose of this project.



In order for the solar panel to produce the maximum amount of energy, it should be tilted at 90 degrees towards the sun. It's because the shadow it makes, is the largest then, meaning that it receives the most possible amount of the sunlight.


SunTRAK: Incident Angle Effect
FORMULA for sin(ϕ) = OPPOSITE / HYPOTENUSE
HYPOTENUSE
OPPOSITE
ϕ
Sunlight
Sunlight
Perpendicular to the base
SHADOW INDICATES the PERCENTAGE of LIGHT, which we will get when PV is tilted at a certain angle to the sun.
SHADOW = LENGTH of PV Panel * sin(x)*100%
(assume it is = 1)
Sum of angles of triangle = 180°
180° - 90° - 45° = 45°
PV Panel
45°
90°
90°
Magic Hand
90° - 45° = 45°
Geometry and calculations for the "Incident Angle Effect" for PV Panels.
90°
Shadow in %
Definition of sin function
sin(45°) = SHADOW / PV Panel Length
Ground Level
Base

As shown in the picture above, the percentage of the panel's shadow (the opposite side in the right triangle) can be calculated by multiplying the length of the panel (hypotenuse), which for simplicity has been assigned a default length of 1, by the sine of the angle that is opposite to the shadow, and then by 100%. Since only a 90 degrees angle results with the highest value of sine (1) and any other angle results with the value of sine below 1, the highest percentage of the shadow (which also represents the production of the PV panel) will be obtained for this exact angle. In order to model this in a computer simulation, the degrees had to be converted into radians using the following formula: degrees * pi/180. Since pi represents the number of times the radius (the length from the center of the circle to its edge) makes a half-circle, dividing it by

180 degrees (which is a half circle as well) results in the numerical unit of an angle (radian). This conversion was necessary because these are the numbers (not the degrees) that can be easily translated to bits. These two math concepts were used in the following functions in the model:

```
def Deg2Rad(x):  return (math.pi*x/180.0)
def Sine(x): return math.sin(Deg2Rad(x))
```

The above Sine(x) function was then used in the IdealModel variable in the following way:

```
    Angle = [(i*5) for i in range(37)]
    IdealModel = [100*Sine(a) for a in Angle]
```

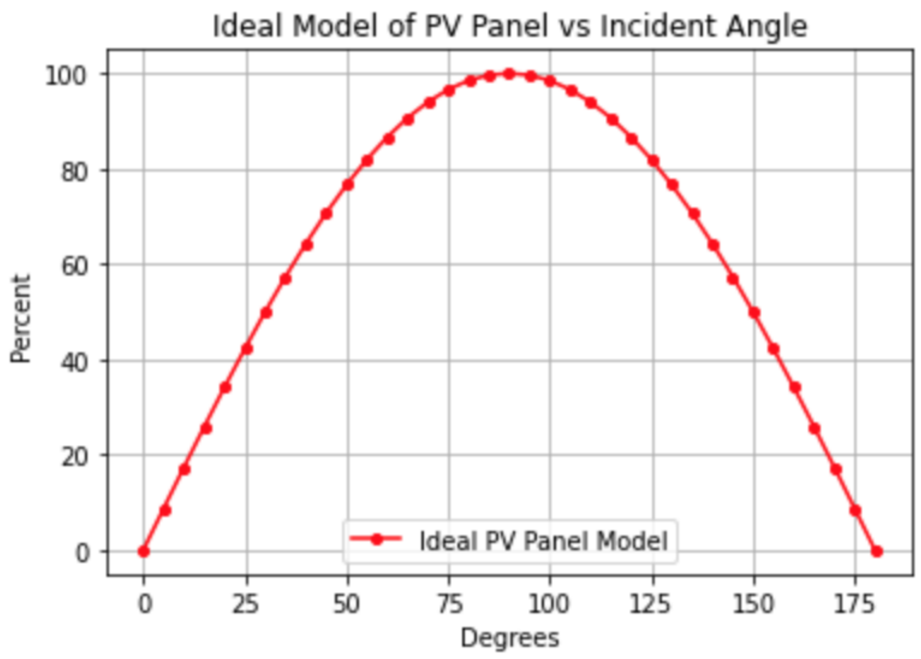
Since the value of the loop control variable i ranges from 0 to 36 and in each iteration it is multiplied by 5, the list comprehension used by these two variables results in the following arrays:

```
Angle = [0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70,
75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145,
150, 155, 160, 165, 170, 175, 180]
IdealModel = [0.0, 8.715574274765817, 17.36481776669303, 25.88190451025207,
34.20201433256687, 42.26182617406995, 49.99999999999999, 57.35764363510461,
64.27876096865393, 70.71067811865474, 76.60444431189781, 81.91520442889917,
86.60254037844386, 90.63077870366499, 93.96926207859083, 96.59258262890683,
98.4807753012208, 99.61946980917456, 100.0, 99.61946980917456,
98.4807753012208, 96.59258262890683, 93.96926207859084, 90.630778703665,
86.60254037844388, 81.91520442889917, 76.60444431189781, 70.71067811865476,
64.27876096865394, 57.35764363510459, 49.99999999999999, 42.26182617406995,
34.20201433256689, 25.88190451025206, 17.36481776669303, 8.715574274765864,
1.2246467991473532e-14]
```

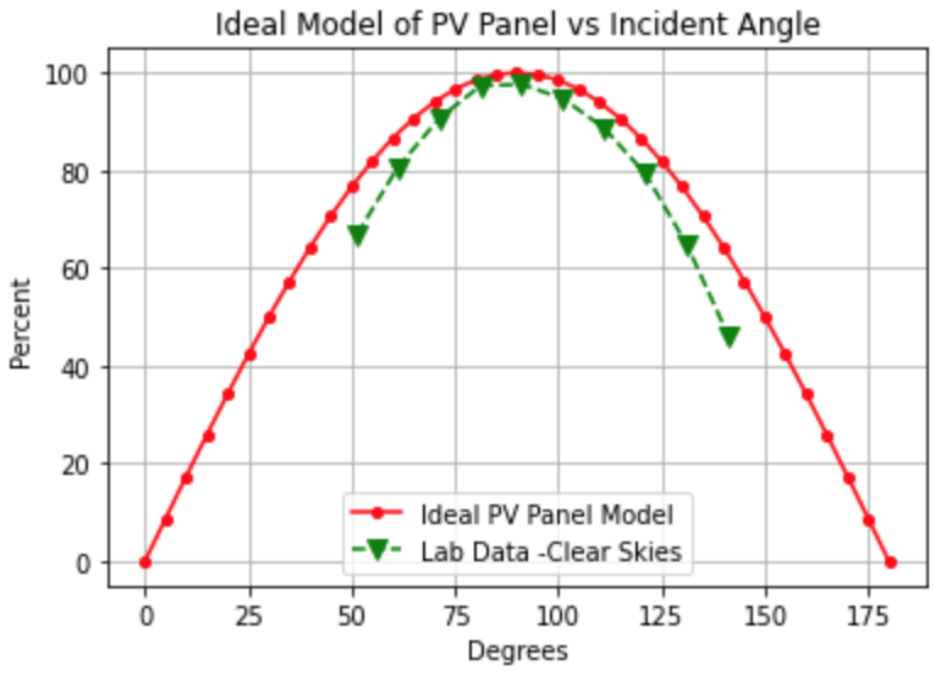
After printing the middle value in these two arrays, what we get is 100% for 90 degrees angle:

```
print(Angle[int(len(Angle)/2)]) → 90
print(IdealModel[int(len(IdealModel)/2)]) → 100.0
```

The above arrays were then plotted in the model as shown in the image below:



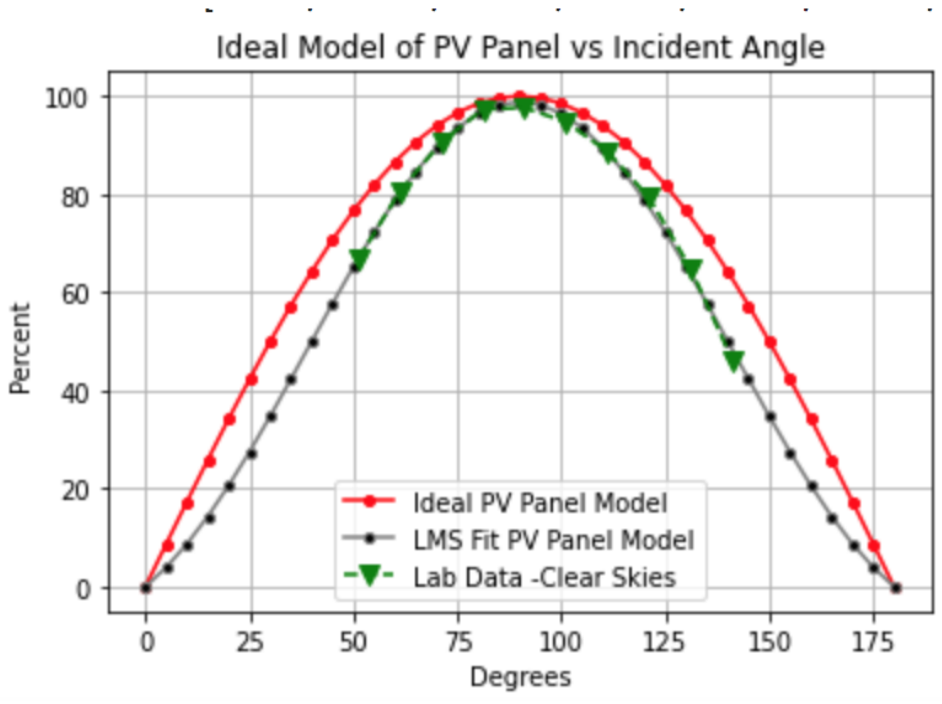
Unfortunately the lab data collected from the prototype of the PV system were slightly different from our model:



This was caused by the fact that the tilt angle of the prototype (as measured from the flat surface) was not necessarily the same as its angle towards the sun. It's because the sun is almost never directly overhead (except at the equator on the spring and fall equinoxes). In addition, for any degree different than 90 some of the sunlight reflects off of the panel and is lost. This is called “scattering”. All of this required adjusting the model to fit the lab data. In order to do so, it was necessary to use the curve_fit() method from the scipy.optimize module (Appendix 3). The values of the parameters returned by this method were then used to adjust the original shadow formula, so it included any potential differences between the tilt angle of the PV panel and its actual angle towards the sun as well as the scattering of the light hitting the panel. This was accomplished by adding the following parameters (coefficients) to the original shadow formula: Efficiency, Scatter, and AngleShift:

```
def pFitModel(x, Efficiency, Scatter, AngleShift):
 return Efficiency*Sine(x + AngleShift)*((Scatter*Sine(x + 
AngleShift) + 1.0)/(Scatter + 1.0))*100
```

Using these parameters was necessary to adjust the theoretical model to the lab data. The curve_fit() function works in the way that it looks for the least mean squared (LMS) error (or the smallest sum of the squared differences between the functions) by adjusting the values of these parameters. Once such values are found, they are then applied to the model so it fits the lab data (grey line):



Such an adjusted model perfectly simulates the productions of the PV system based on its tilt angle towards the sun when the sky is clear. It clearly shows that in order for the actual solar trackers to constantly chase the sun, they must be controlled by a programmable microcontroller, such as Arduino that constantly adjusts the tilt angle of the panel, for which it produces the maximum amount of electricity.

Conclusion

Since solar PV systems do not rely on fossil fuels, they can definitely help decrease greenhouse gas footprint a lot. A residential solar PV system has the capability of providing for the electricity needs of an entire home with about 80% lower carbon emissions than fossil fuels (Arif). And the reason why it's not 100% is because the house still needs electricity during the night, and if it does not have the battery, it then relies on the power from the utility grid, which depends on fossil fuels as one of its energy sources.

Also, since the California governor signed an order banning sale of gasoline-powered cars by 2035, investing in PV systems seems to be the most adequate solution and a significant source of energy that will be used to charge electric cars. It's only a matter of time when the government of New Mexico declares a similar legal restriction. That is why sooner or later this project may become a very useful educational tool used to inform students, but also a local community about this type of renewable energy. More specifically, this project and all its computational artifacts could be used to explain the purpose, types, size, and the annual and monthly production of the PV systems, as well as their price, and payoff time. In addition, both the computational model and a prototype could also be used to demonstrate how PV systems work, and how important the proper tilt angle of the solar panels is.

In addition, taking data and creating a model for a solar panel array tells how well any particular array will perform before it is installed. That way, the most cost effective arrangement can be predicted. While it would be much easier to do with the ground mount systems, it would be much more challenging (but still doable) to install the solar trackers on the roofs. In order to properly set and control the tilt angle, the PV panels should be installed on a construction attached to the roof that would enable rotation and movement of such panels at the angle that would maximize the production of AC energy. However, if such solution would be too expensive and/or too hard to implement, simply lifting and tilting the panels, as shown in the image below, could also increase the efficiency of the system:

Thus in short, our model allows studying the effectiveness of the systems that give customers the ability to manually change angles to maximize their output.

Future Work

Due to COVID-19 and the necessity to keep social distance, it was impossible to work closely with our engineering mentor on building the prototype of the PV system. We still learned from his lectures, Google presentation, and photos, but it would be more beneficial to observe him at work as well as to build such a prototype together. We would then have a great opportunity to learn some electrical engineering too.

Thus, our plan for the future is to learn about designing electrical circuits and assembling and programming a variety of prototypes, and for this reason we plan to continue meeting with our mentor in order to learn how to program Arduino to control such prototypes, like this PV panel. In addition, we also would like to learn more about how, depending on each season in a year, should the PV panels be titled in order to produce the most energy. And finally, since this project involved some college level math, like least mean squared error and calculus, we would also like to start learning and practicing these kinds of math concepts in order to better understand the functions from the scientific libraries.

Acknowledgments

This project required a lot of time, hard work and determination from our teacher and mentor. We would like to thank Ms. Teterycz for all her time and knowledge, which she did not hesitate sharing with us.

Lastly, we would like to also thank Mr. David Ritter for his kindness and willingness to mentor us throughout this project. Mr. David has provided us with many resources and his impressive and very rich expertise.

This project was a fascinating opportunity which would not have been possible without the support from both our teacher and our engineering mentor.

References

EnergySage. “How Much Does a 4.5 kW Solar System Cost in 2020?” *energysage*, 2020, https://news.energysage.com/4-5-kw-solar-system-cost/. Accessed 16 September 2020.

The Environmental Literacy Council. “Fossil Fuels.” *The Environmental Literacy Council*, 2015, https://enviroliteracy.org/energy/fossil-fuels/. Accessed 10 September 2020.

HowStuffWorks. “How Solar Cells Work.” *science.howstuffworks*, 2020, https://science.howstuffworks.com/environmental/energy/solar-cell.htm. Accessed 25 September 2020.

Arif, Marshall S. “Residential Solar Panels and Their Impact on the Reduction of Carbon Emissions.” nature.berkeley.edu, April 2013, https://nature.berkeley.edu/classes/es196/projects/2013final/ArifM_2013.pdf. Accessed 13 January 2020

NASA's Jet Propulsion Laboratory. “Global Temperature.” *Global Climate Change*, 2020, https://climate.nasa.gov/vital-signs/global-temperature/. Accessed 5 October 2020.

NREL. “PVWatts Calculator.” *PVWatts Calculator*, 2020, https://pvwatts.nrel.gov/. Accessed 15 September 2020.

PNM. “How To Read Your Bill.” *PNM*, 2014, https://www.pnm.com/definitions. Accessed 20 September 2020.

PNM. “PNM Customer Solar Energy Program.” *PNM*, 2018, https://www.pnm.com/solar. Accessed 20 September 2020.

Santa Fe Convention Center. “Santa Fe Weather.” *Santa Fe*, 2020, https://santafe.org/Weather/index.html. Accessed 3 November 2020.

SolarReviews. “How Much Energy Does A Solar Panel Produce?” *SolarReviews*, 2020, https://www.solarreviews.com/blog/how-much-electricity-does-a-solar-panel-produce. Accessed 15 September 2020.

U.S. Department of Energy. “Homeowner's Guide to the Federal Tax Credit for Solar Photovoltaics.” *ENERGY.GOV*, 2020, https://www.energy.gov/sites/prod/files/2020/01/f70/Guide%20to%20Federal%20Tax%20Credit%20for%20Residential%20Solar%20PV.pdf. Accessed 20 September 2020.

U.S. Department of Energy. “New Mexico.” *NREL*, 2020, https://www.nrel.gov/solar/rps/nm.html. Accessed 20 September 2020.

Wester, Catharine. “Why Do I Need an Inverter for My Home Solar Energy System.” *Ezine Articles*, 2011, https://ezinearticles.com/?Why-Do-I-Need-an-Inverter-for-My-Home-Solar-Energy-System&id=6328627. Accessed 25 September 2020.

Appendix 1

JavaScript app:

```
function pricePayoff(size) {
  var netPVCost = Math.round(size*2.91 - size*2.91*36/100);
  setText("totalPrice", '$'+Math.round(size*2.91));
  setText("netPrice", '$'+netPVCost);
  setText("payoffTime", Math.round(10*netPVCost/totalElectrCost)/10);
  //Since Math.roud() rounds to the nearest integer, in order
  //to get one decimal place, multiply the value by 10, round,
  //and then devide it by 10.
}

var totalUsage = 0;
var totalElectrCost = 0;

onEvent("button", "click", function( ) {
  totalUsage = Math.round(getNumber("avgDailyUsage")*365);
  setNumber("annualUsage", totalUsage);
  totalElectrCost = Math.round(getNumber("avgDailyCost")*365);
  setText("annualCost", '$'+totalElectrCost);

  if (totalUsage <= 1800) {
    setText("sizePV", "1 kW");
    setText("annualPVEnergy", "1,745");
    pricePayoff(1000);
  } else if ((totalUsage > 1800) && (totalUsage <= 2700)) {
    setText("sizePV", "1.5 kW");
    setText("annualPVEnergy", "2,619");
    pricePayoff(1500);
  } else if ((totalUsage > 2700) && (totalUsage <= 3500)) {
    setText("sizePV", "2 kW");
    setText("annualPVEnergy", "3,492");
    pricePayoff(2000);
  } else if ((totalUsage > 3500) && (totalUsage <= 4500)) {
    setText("sizePV", "2.5 kW");
    setText("annualPVEnergy", "4,365");
    pricePayoff(2500);
  } else if ((totalUsage > 4500) && (totalUsage <= 5400)) {
    setText("sizePV", "3 kW");
    setText("annualPVEnergy", "5,238");
    pricePayoff(3000);
  } else if ((totalUsage > 5400) && (totalUsage <= 6300)) {
    setText("sizePV", "3.5 kW");
    setText("annualPVEnergy", "6,111");
    pricePayoff(3500);
```

```
} else if ((totalUsage > 6300) && (totalUsage <= 7200)) {
  setText("sizePV", "4 kW");
  setText("annualPVEnergy", "6,984");
  pricePayoff(4000);
} else if ((totalUsage > 7200) && (totalUsage <= 8100)) {
  setText("sizePV", "4.5 kW");
  setText("annualPVEnergy", "7,857");
  pricePayoff(4500);
} else if ((totalUsage > 8100) && (totalUsage <= 9000)) {
  setText("sizePV", "5 kW");
  setText("annualPVEnergy", "8,730");
  pricePayoff(5000);
} else if ((totalUsage > 9000) && (totalUsage <= 9900)) {
  setText("sizePV", "5.5 kW");
  setText("annualPVEnergy", "9,603");
  pricePayoff(5500);
} else if ((totalUsage > 9900) && (totalUsage <= 10800)) {
  setText("sizePV", "6 kW");
  setText("annualPVEnergy", "10,476");
  pricePayoff(6000);
} else if ((totalUsage > 10800) && (totalUsage <= 11700)) {
  setText("sizePV", "6.5 kW");
  setText("annualPVEnergy", "11,349");
  pricePayoff(6500);
} else if ((totalUsage > 11700) && (totalUsage <= 12800)) {
  setText("sizePV", "7 kW");
  setText("annualPVEnergy", "12,414");
  pricePayoff(7000);
} else if ((totalUsage > 12800) && (totalUsage <= 13700)) {
  setText("sizePV", "7.5 kW");
  setText("annualPVEnergy", "13,300");
  pricePayoff(7500);
} else if ((totalUsage > 13700) && (totalUsage <= 14600)) {
  setText("sizePV", "8 kW");
  setText("annualPVEnergy", "14,187");
  pricePayoff(8000);
} else if ((totalUsage > 14600) && (totalUsage <= 15500)) {
  setText("sizePV", "8.5 kW");
  setText("annualPVEnergy", "15,074");
  pricePayoff(8500);
} else if ((totalUsage > 15500) && (totalUsage <= 16400)) {
  setText("sizePV", "9 kW");
  setText("annualPVEnergy", "15,960");
  pricePayoff(9000);
} else if ((totalUsage > 16400) && (totalUsage <= 17300)) {
  setText("sizePV", "9.5 kW");
  setText("annualPVEnergy", "16,847");
  pricePayoff(9500);
  } else if ((totalUsage > 17300) && (totalUsage <= 18300)) {
    setText("sizePV", "10 kW");
    setText("annualPVEnergy", "17,734");
    pricePayoff(10000);
  }
  else {
    setScreen("screen2");
    onEvent("tryAgain", "click", function( ) {
      setScreen("screen1");
    });
    setText("avgDailyUsage", "");
    setText("avgDailyCost", "");
    setText("annualUsage", "");
    setText("annualCost", "");
    setText("sizePV", "");
    setText("annualPVEnergy", "");
    setText("totalPrice", "");
    setText("netPrice", "");
    setText("payoffTime", " ");
  }
});
```

Appendix 2

JavaScript app translated to Python enhanced by a bar chart:

```
import math
import matplotlib.pyplot as plt
print()
#The max daily usage: 50!
#Small house uses about 8-10 kWh at the cost of $1-2 per day
#Large household uses about 35-50 kWh at the cost of $5.5 - 7 per day
avgDailyUsage = float(input("Your average electricity usage per day in kWh
from your Utility bill: "))
avgDailyCost = float(input("Your average electricity cost per day: "))
print()
print("Your annual electricity usage is about {:,.0f}
kWh.".format(avgDailyUsage*365))   #curly brackets indicate the
placeholder for variable, comma is used to indicate the thousands place
and .0 means that there will be no decimal values, and f means that this
variable uses float as a data type
print("Your annual electricity cost is about
${:,.0f}.".format(avgDailyCost*365))
def pricePayoff(size, prod):
 print()
 panels = math.ceil(size/275) #275 watts is produced by a standard
polycrystaline solar panel
 print("Approximate NUMBER of PANELS: "+str(panels)+';')
 print()  #2.91 is the average cost of 1 watt produced by a PV system
 print("TOTAL COST: about ${:,.0f};".format(size*2.91))
 print()
 netCost = size*2.91 - size*2.91*36/100  #26% is federal income tax credit
and 10% is state, all = 36%
 print("NET COST (after deducting 36% federal & state tax credits):
${:,.0f};".format(netCost))
 print()
 print("PAYOFF: It will take about {:.1f} years to pay off such
system".format(netCost/(avgDailyCost*365)))
 print("from what you will save on your utility bills each year.")
 print()
 months = ['Jan:', 'Feb:', 'Mar:', 'Apr:', 'May:', 'Jun:', 'Jul:', 'Aug:',
'Sep:', 'Oct:','Nov:', 'Dec:']
 print("Such PV System will produce the following amount of AC energy each
month:")
 for i in range(len(prod)):
   print(months[i],prod[i])
 print("Total:",sum(prod),'kWh')
 print()

if avgDailyUsage*365 <= 18300:
 print()
 print("Specifications of the PV System for area code = 87507:")
 print("*"*54)
 print()
 #Use PVWatts Calculator to get annual production made by 1 - 10 kW
DC(direct current) PV Systems:
 if avgDailyUsage*365 <= 1800:        #max 1 kW DC System outpt
   print("SIZE: 1 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 1,745 kWh of AC")
   production = [122, 125, 156, 165, 179, 170, 157, 155, 146, 136, 121,
113]
   pricePayoff(1000, production)
 elif avgDailyUsage*365 > 1800 and  avgDailyUsage*365 <= 2700:   #max 1.5
kW DC System output
   print("SIZE: 1.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 2,619 kWh of AC;")
   production = [183, 188, 233, 248, 269, 256, 235, 232, 219, 205, 182,
169]
   pricePayoff(1500, production)
 elif avgDailyUsage*365 > 2700 and avgDailyUsage*365 <= 3500:
   print("SIZE: 2 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 3,492 kWh of AC;")
   production = [244, 251, 311, 330, 359, 341, 313, 310, 292, 273, 242,
226]
   pricePayoff(2000, production)
 elif avgDailyUsage*365 > 3500 and avgDailyUsage*365 <= 4500:
   print("SIZE: 2.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 4,365 kWh of AC;")
   production = [305, 314, 389, 413, 448, 426, 391, 387, 365, 341, 303,
282]
   pricePayoff(2500, production)
 elif avgDailyUsage*365 > 4500 and avgDailyUsage*365 <= 5400:
   print("SIZE: 3 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 5,238 kWh of AC;")
   production = [367, 376, 467, 495, 538, 511, 470, 465, 438, 409, 363,
339]
   pricePayoff(3000, production)
 elif avgDailyUsage*365 > 5400 and avgDailyUsage*365 <= 6300:
   print("SIZE: 3.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 6,111 kWh of AC;")
   production = [428, 439, 544, 578, 628, 597, 548, 542, 511, 477, 424,
395]
   pricePayoff(3500, production)
 elif avgDailyUsage*365 > 6300 and avgDailyUsage*365 <= 7200:
   print("SIZE: 4 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 6,984 kWh of AC;")
   production = [489, 502, 622, 660, 717, 682, 626, 620, 584, 545, 484,
452]
   pricePayoff(4000, production)
 elif avgDailyUsage*365 > 7200 and avgDailyUsage*365 <= 8100:
   print("SIZE: 4.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 7,857 kWh of AC;")
   production = [550, 565, 700, 743, 807, 767, 705, 697, 657, 614, 545,
508]
   pricePayoff(4500, production)
 elif avgDailyUsage*365 > 8100 and avgDailyUsage*365 <= 9000:
   print("SIZE: 5 kW DC System;")
```

```
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 8,730 kWh of AC;")
   production = [611, 627, 778, 826, 897, 852, 783, 774, 730, 682, 605,
565]
   pricePayoff(5000, production)
 elif avgDailyUsage*365 > 9000 and avgDailyUsage*365 <= 9900:
   print("SIZE: 5.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 9,603 kWh of AC;")
   production = [672, 690, 856, 908, 986, 938, 861, 852, 803, 750, 666,
621]
   pricePayoff(5500, production)
 elif avgDailyUsage*365 > 9900 and avgDailyUsage*365 <= 10800:
   print("SIZE: 6 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 10,476 kWh of AC;")
   production = [733, 753, 933, 991, 1076, 1023, 940, 929, 876, 818, 726,
678]
   pricePayoff(6000, production)
 elif avgDailyUsage*365 > 10800 and avgDailyUsage*365 <= 11700:
   print("SIZE: 6.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (roof mount);")
   print()
   print("ANNUAL PRODUCTION: about 11,349 kWh of AC;")
   production = [794, 816, 1011, 1073, 1166, 1108, 1018, 1007, 949, 886,
787, 734]
   pricePayoff(6500, production)
 elif avgDailyUsage*365 > 11700 and avgDailyUsage*365 <= 12800:
   print("SIZE: 7 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 12,414 kWh of AC;")
   production = [867, 889, 1107, 1172, 1274, 1214, 1117, 1104, 1040, 969,
859, 802]
   pricePayoff(7000, production)
 elif avgDailyUsage*365 > 12800 and avgDailyUsage*365 <= 13700:
   print("SIZE: 7.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 13,300 kWh of AC;")
   production = [928, 953, 1186, 1255, 1365, 1300, 1197, 1183, 1114, 1038,
921, 859]
   pricePayoff(7500, production)
 elif avgDailyUsage*365 > 13700 and avgDailyUsage*365 <= 14600:
   print("SIZE: 8 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 14,187 kWh of AC;")
   production = [990, 1016, 1265, 1339, 1456, 1387, 1276, 1262, 1189,
1108, 982, 916]
   pricePayoff(8000, production)
 elif avgDailyUsage*365 > 14600 and avgDailyUsage*365 <= 15500:
   print("SIZE: 8.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 15,074 kWh of AC;")
   production = [1052, 1080, 1345, 1423, 1547, 1474, 1356, 1341, 1263,
1177, 1043, 974]
   pricePayoff(8500, production)
 elif avgDailyUsage*365 > 15500 and avgDailyUsage*365 <= 16400:
   print("SIZE: 9 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 15,960 kWh of AC;")
   production = [1114, 1143, 1424, 1507, 1638, 1561, 1436, 1420, 1337,
1246, 1105, 1031]
   pricePayoff(9000, production)
 elif avgDailyUsage*365 > 16400 and avgDailyUsage*365 <= 17300:
   print("SIZE: 9.5 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 16,847 kWh of AC;")
   production = [1176, 1207, 1503, 1590, 1729, 1647, 1516, 1499, 1411,
1315, 1166, 1088]
   pricePayoff(9500, production)
 elif avgDailyUsage*365 > 17300 and avgDailyUsage*365 <= 18300:
   print("SIZE: 10 kW DC System;")
   print()
   print("RECOMMENDED TYPE: Fixed (open rack);")
   print()
   print("ANNUAL PRODUCTION: about 17,734 kWh of AC;")
   production = [1238, 1270, 1582, 1674, 1819, 1734, 1595, 1577, 1486,
1385, 1228, 1145]
   pricePayoff(10000, production)
 #Plotting a bar chart to display monthly production of AC energy by PV
Solar System:
 #List comprehension used for x coordinates
 x = [i for i in range(1, 13)]
 months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep',
'Oct', 'Nov', 'Dec']
 plt.bar(x, production, tick_label = months, width = 0.8, color = ['red',
'green', 'blue'])
 #Labels for x and y axis:
 plt.xlabel('Month')
 plt.ylabel('kWh')
 #Title of the graph:
 plt.title('Monthly AC Energy produced by your PV Solar System')
 plt.show()

else:
 print()
 print("This app was designed for residential houses that use up to 50kWh
electricity a day")
```

Appendix 3

Computational Model written using Python to simulate a solar tracker:

```
from scipy.optimize import curve_fit
from numpy import vectorize
import math
import matplotlib.pyplot as plt
#===== Helper functions for manipulating data=====
def Deg2Rad(x):  return (math.pi*x/180.0) # Deg2Rad convert
degrees -> radians
def Sine(x): return math.sin(Deg2Rad(x)) # Sine function is
math.sin of degrees
def RoundN(x,n):  return ( round(x*10.0**n)/10.0**n) # Round to
N decimal places
#--------------------------------------------------
# ===== Function to find angle for max response in lab data
=====
#.   This scans the data and finds largest value
#.       then returns the index pointing to largest value
#.   Useful to find the angle closest to the maximum response in
the data
#.       "curve_fit" needs guesses for values, like the angle
for the peak
#        response at 90 degrees.  If we know this angle of
maximum in lab
#        data we can compute best guess angle offset to start
the curve_fit
def GetMaxAngle(angle,data):
  max_value = max(data)
  max_index = data.index(max_value)
  return angle[max_index]
#-------------------------------------------------
#=====        Enter the data from panel     =====
LabAngle = [i*10.0 for i in range(0,10)] #. 0 to 90 degrees in
10 degree increments
LabDataClear =
[66.7,80.6,90.6,97.3,97.6,94.7,88.6,79.2,64.7,46.1] # actual lab
data
NLabData = len(LabDataClear) # how many data points?
#-----------------------------------------------------------
#=====    Define the parameterized model and intial   =====
def pFitModel(x, Efficiency, Scatter, AngleShift):
  return Efficiency*Sine(x + AngleShift)*((Scatter*Sine(x +
AngleShift)+1.0)/(Scatter+1.0))*100
#.  We need to define our realistic function to fit to data
#.  This function has the ideal function plus a scatter function
#.  The overall Efficiency and amount of Scatter are parameters
in this function
#.  It also computes AngleShift, which is how far we have to
shift the data angles
#.     to center the lab data at 90 degrees (straight at sun)
#.  Parameters this function needs: Efficiency, Scatter,
AngleShift
#. "curve_fit" function needs the model function to be
'vectorized': the function
#.    must be able to take a list as input and generate a list
as output.
#
#  Define the vectorized version
vFitModel = vectorize(pFitModel) # this syntax came from the doc
notes
# "curve_fit" works more reliably if we have intial guesses for
parameters
#
MaxAngleGuess = GetMaxAngle(LabAngle, LabDataClear) # If we find
the angle in the
#      data that gives largest output, that is close to pointing
at sun, so we use
#      that value to compute the offset to make the lab angles
centered so that
#.     max response is near 90 degrees.  The curve fit procedure
will adjust this
#.     to make it precise.
#
guess = [ 1.0, 0.0, 90-MaxAngleGuess] # Make following guesses:
#.     1) Efficiency is 1.0 ... a perfect panel
#.     2) Scatter is zero ... panel uses all the light it gets
#.     3) AngleShift is 90-MaxAngleGuess ... this shifts data so
max is at 90 deg
#-------------------------------------------------------------
---
#=====    Invoke the curve fit procedure according to docs
=====
#.  Routine returns the parameters in the 'params' list
#.     and an error term in the 'covariance'
#.  We send the parameterized model function, the x,y data, and
the guesses
params, covariance =
curve_fit(f=vFitModel,xdata=LabAngle,ydata=LabDataClear,p0=guess
,bounds=(-10000,10000))
#.  Now we extract the parameters of interest
FM_efficiency=RoundN(params[0],3)
FM_scatter=RoundN(params[1],3)
FM_angleshift=RoundN(params[2],3)
#.  and print them out to include in plotting program
print(" FM means Fitted Model ")
print("FM_efficiency = ",FM_efficiency)
print("FM_scatter = ",FM_scatter)
print("FM_angleshift = ",FM_angleshift)

LabAngleShifted = [LabAngle[i]+FM_angleshift for i in
range(0,10)]
print("Lab Angle Shifted = ", LabAngleShifted)
print("Lab Data ", LabDataClear)
#===== Define the simplified Fit Model for plotting
#.  and sample at 5 degree angle increments
#.  Zero out the FM_angleshift parameter here because its now
included in the
#.     lab angle data.
def FitModel(x):
   return pFitModel(x,FM_efficiency,FM_scatter,0.0)
Angle = [(i*5) for i in range(37) ]
IdealModel = [100*Sine(a) for a in Angle]
FitModelSampled = [FitModel(a) for a in Angle]
#--------------------------------------------------------
#=====   Plot Ideal, Lab Data, and FitModel Data  ======
plt.plot(Angle,IdealModel,label="Ideal PV Panel Model",
color='red',linestyle='solid',marker='o',markerfacecolor='red',m
arkersize=4)# plot lab data
plt.plot(Angle,FitModelSampled,label="LMS Fit PV Panel Model",
color='grey',linestyle='solid',marker='o',markerfacecolor='black
',markersize=4)# plot lab data
plt.plot(LabAngleShifted,LabDataClear,label="Lab Data -Clear
Skies",color='green',linestyle='dashed',marker='v',markerfacecol
or='green',markersize=8)
plt.xlabel('Degrees')
plt.ylabel('Percent')
plt.grid()
plt.legend()
```

```
#Title of the graph:
plt.title('Ideal Model of PV Panel vs Incident Angle')
plt.show()
plt.savefig('PVPanelPlot.png')
```