Charles Lin
Sasha Evans
Date: Tue, 10/2/18

ME 146/246 Project 1:
Solar Collector Design and Analysis

Introduction

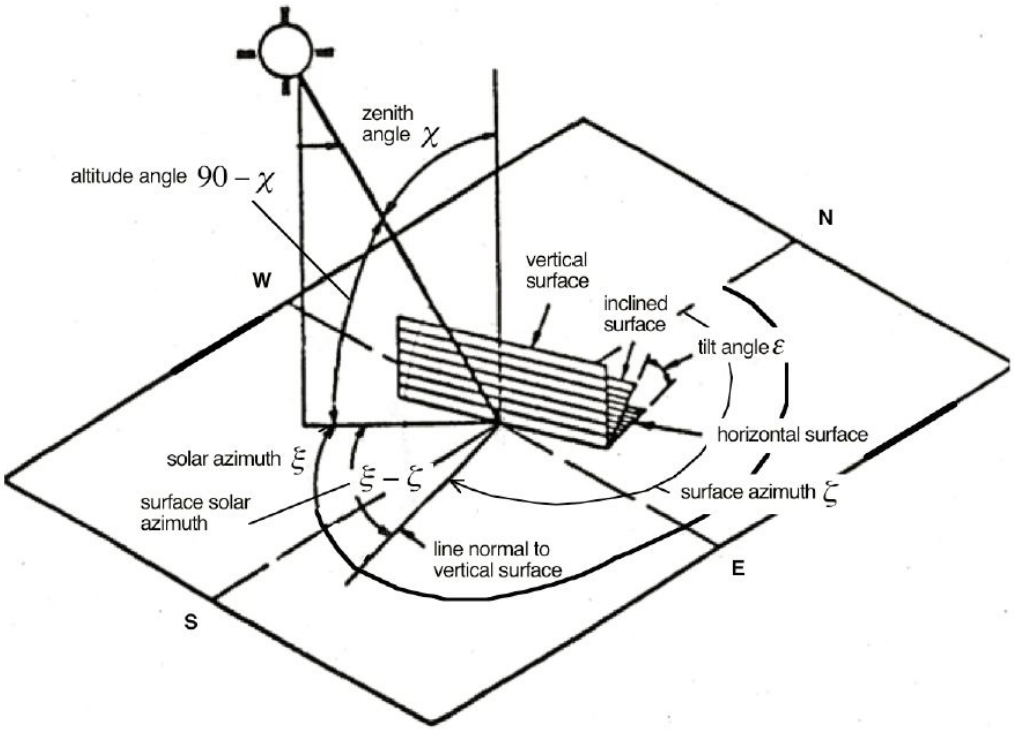

Fig. 1: Collector Model

The purpose of this project is to create a mathematical model of a solar collector and use it to provide key parameters and considerations for proposed operation in Berkeley, CA. Using MATLAB, we performed analysis according to our conceptual model, shown in Fig. 1, by implementing the following equations and constants in Tasks 1, 2, and 3:

Calculation of Incident Solar Flux

Variables

- λ: Latitude $(\lambda = 37.9^\circ N \; for \; Berkeley, \; CA)$
- ζ: Surface azimuth angle
- ε: Surface inclination from horizontal
- t: Solar time (given in decimal for a 24-hr clock)

Equations

- Hour angle: $\alpha \Rightarrow \alpha = 15(t - 12)$ [1]
- Declination: $\delta \Rightarrow \delta = 23.44 \sin[\frac{360}{365.25}(d - 80)]$, $d = number \; of \; day \; of \; the \; year$ [2]
- Solar zenith angle: $\chi \Rightarrow \cos\chi = \sin\lambda \sin\delta + \cos\lambda \cos\delta \cos\alpha$ [3]
- Solar azimuth angle: $\xi \Rightarrow \tan\xi = \frac{\sin\alpha}{\sin\lambda \cos\alpha - \cos\lambda \tan\delta}$, evaluated by chart below: [4]

Sign(α)	Sign(tanξ)	ξ
+	+	180° + arctan(tanξ)
+	−	360°+ arctan(tanξ)
−	+	arctan(tanξ)
−	−	180°+ arctan(tanξ)

- Intensity of direct normal radiation: $I_{DN} \Rightarrow I_{DN} = Ae^{\frac{-B}{\sin(90-\gamma)}}$ [5]
  - Constants: A = 1310 W/m^2, B = 0.18
- Incident direct solar flux: $I_D \Rightarrow I_D = I_{DN}[\cos\chi\cos\varepsilon + \sin\varepsilon\sin\chi\cos(\xi-\zeta)]$ [6]

Calculation of Collector Efficiency

Variables

- τ_g : collector glazing transmissivity
- α_c : collector absorber plate absorptivity
- $h_{conv,o}$: convection coefficient (outside air)
- $h_{conv,i}$: convection coefficient (inside, between absorber and glazing)
- δ_g : glazing thickness
- δ_{ins} : insulation thickness on collector backside
- k_g : glazing thermal conductivity
- k_{ins} : insulation thermal conductivity
- A_c : collector absorber area
- U_{loss} : overall heat loss
- $\dot{m}$: mass flow rate
- c_p : specific heat

Equations

Collector efficiency: $\eta_{coll} \Rightarrow \eta_{coll} = \overline{F}_R\,[\tau_g\alpha_c - \frac{U_{loss}}{I_D}(T_{inlet} - T_{ambient})]$ [7]

Heat removal factor: $\overline{F}_R = \frac{1-exp(-A_cU_{loss}/\dot{m}c_p)}{A_cU_{loss}/\dot{m}c_p}$ [8]

Total conductance: $U_{loss}A_c = [\frac{1}{h_{conv,o}A_c} + \frac{\delta_g}{k_gA_c} + \frac{1}{h_{conv,i}A_c}]^{-1} + [\frac{1}{h_{conv,o}A_c} + \frac{\delta_{ins}}{k_{ins}A_c}]^{-1}$ [9]

Responsibilities

This section provides a summary of how the project work was divided between the teammates when preparing this report.

Charles Lin:

- Introduction
- Task 1
- Task 2
- Task 3
- Task 4
    - Part (a): Determining optimal angles
    - Part (a): Analysis for parallel versus series arrangement of collectors

Sasha Evans:

- Task 4
    - Part (a): Determining optimal mass flow rate
    - Part (b)
    - Part (c)
- Task 5a
- Task 5b

Design Task Documentation

Task 1

Task Description:
We are tasked with calculating the incident direct radiation per square meter of a surface I_D, given by Eq. [6]. To set up a proof of concept for our mathematical model, we initialize all variables as constants.

Assumptions/Idealizations:
- 200° azimuth
- 36° tilt angle
- Date: April 30, 2018
- Solar time: 13.00 (1:00 PM)
- Location: Berkeley, CA

Algorithm Summary of the Program:
We utilize Eqs. [1] through [5] to find the prerequisite values for $I_{DN}, \chi, \varepsilon,$ and ξ. We use these values to calculate for I_D in Eq. [6].

Results:
Our resulting: I_D = 1048.34 W/m^2.

Task 2

Task Description:
We are tasked with refining our model to calculate total energy incident for a surface area A_c over a time interval from 10 AM - 4 PM. Finding total energy incident requires us to integrate the values of I_D result over this time interval.

Assumptions/Idealizations:
- 200° azimuth
- 36° tilt angle
- Date: April 30, 2018
- Solar time interval: [10.00, 16.00] (10:00 AM - 4:00 PM)
- Location: Berkeley, CA
- Area A_c = 1 m^2

Algorithm summary of the program:

To set up our integration, we set up our time interval by instituting a linearly spaced vector from 10.00 to 16.00 hours (solar time). Using these values for t, we find the corresponding results for I_D.

Before integrating I_D over t, we also need to verify whether our time steps were small enough to adequately capture the behavior of I_D. This means we need $dI_D/dt < 2\%$, which we test for using a for-loop that finds percent difference between one value of I_D and the next. Using this, we find that a time step of 0.01 hours for our t array gives sufficiently fine values of I_D; Table 1 below illustrates the sufficiency of our time step. We integrate I_D over t using [cumtrapz], a built-in Matlab function for cumulative trapezoidal numerical integration.

dt (hrs)	Percent change of I_D
0.0100	0.2535 %
0.0200	0.2521 %
0.0300	0.2507 %
0.0400	0.2494 %
0.0500	0.2480 %
0.0600	0.2467 %
0.0700	0.2454 %
0.0800	0.2440 %
0.0900	0.2427 %
0.1000	0.2414 %

Table 1: dI_D/dt for dt for dt from 0.01 to 0.1 hrs

Results:

Our resulting total energy is 20.036 MJ, shown in Fig. 2

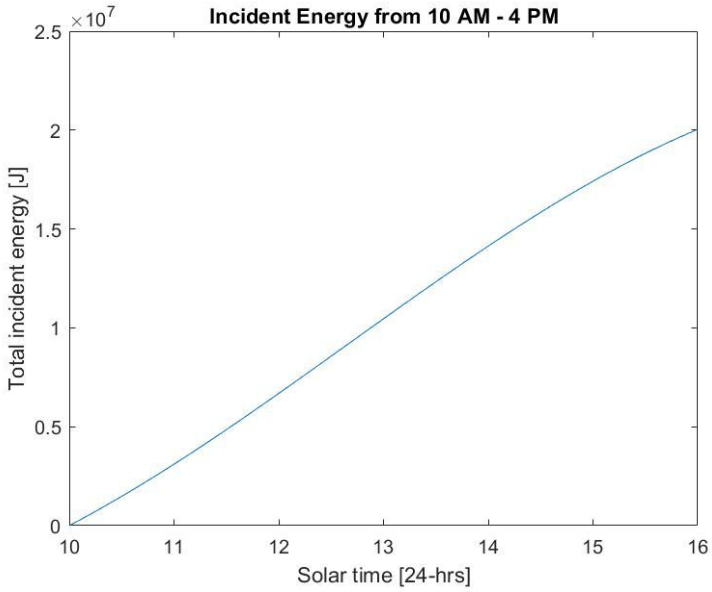

Fig. 2: Incident Energy over Time

Task 3

Task Description:

Now that we are able to calculate total energy incident on a surface over a period of time, we finalize our model by accounting for solar collector efficiency.

Assumptions/Idealizations:

We rely on Eqs. [7] through [9], assuming the following conditions:

- All previous assumptions from Task 2 hold.
- collector glazing transmissivity $\tau_g = 0.89$
- collector absorber plate absorptivity $\alpha_c = 0.85$
- $h_{conv,o} = 7\ \text{W/m}^2$
- $h_{conv,i} = 3.1\ \text{W/m}^2$
- $\delta_g = 0.007\ \text{m}$ (0.7 cm)
- $\delta_{ins} = 0.06\ \text{m}$ (6 cm)
- $k_g = 1.3\ \text{W/mK}$
- $k_{ins} = 0.045,\ 0.018\ \text{W/mK}$
- $A_c = 1\ \text{m}^2$
- $\dot{m} = 0.0267\ \text{kg/s}$
- $c_p = 4186\ \text{J/kg}^\circ\text{C}$
- $T_{inlet} = 12\ ^\circ\text{C}$
- $T_{ambient} = 16\ ^\circ\text{C}$

Results:

Under these assumptions, we solve for total energy while accounting for collector efficiency. To do so, we find the corresponding efficiency values η_{coll} for each value of I_D and perform element-by-element multiplication of the 2 vectors such that $I_{D,actual} = I_D \eta_{coll}$. We then integrate $I_{D,actual}$ over time t.

To find the overall efficiency over this time period, we take the ratio of cumulative E_{actual} value to our ideal E_{ideal}. We calculate this for k_{ins} = 0.045 and 0.018 W/mK.

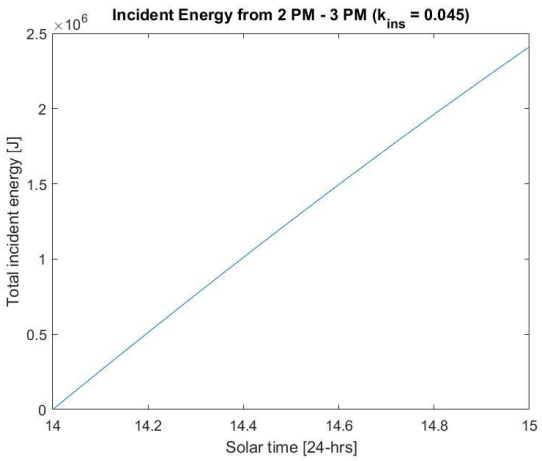

Fig. 3(a): k_{ins} = 0.045

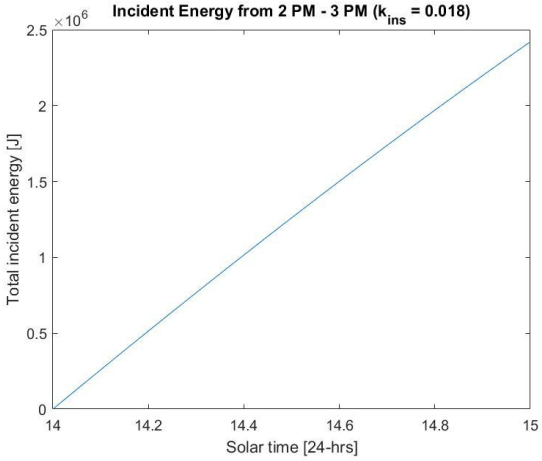

Fig. 3(b): k_{ins} = 0.045

```
For k_ins = 0.045...
   Total ideal energy (w/out eff.) = 3279986.097 J
   Total actual energy = 2410633.0218 J
   Efficiency = 0.73495
>> task3
For k_ins = 0.018...
   Total ideal energy (w/out eff.) = 3279986.097 J
   Total actual energy = 2420375.7269 J
   Efficiency = 0.73792
```

Fig. 3(c)

After decreasing insulation thermal conductivity to 0.018 W/mK, total energy increased by 9742.71 J and efficiency increased by 0.00297.

Task 4(a): Optimal Collector Angles & Arrangement

Task description:

We are tasked with determining the optimal angles that would maximize the amount of water that can be heated from 16°C to 65°C by the system of three solar collectors, each of area A_c = 3.25 m^2 from 10AM to 4PM (solar time) on April 30th in Berkeley, CA. The ambient temperature varies linearly between 9°C and 16°C from 10AM to 4PM.

Assumptions/Idealizations:

- Time interval is 10:00 AM to 4:00 PM.
- Water must be heated in one pass.
- Perfect insulation of the tank where water is collected; water does not lose heat.
- Goal is to have the average temperature of all the water collected in the tank at the end of the day (by 4PM) at or above 65ºC. This means that at every time step the water temperature may be higher or lower than 65ºC, but once all the contributions are added up, the average temperature has to be at or above 65ºC.
- Water pressure loss from series arrangement is negligible.
- Constant mass flow rate of water throughout the day.
- Time step size for incident direct radiation integration = 0.01 hrs.

Algorithm summary of the program:

1. Initializing solar incident flux calculation constants
2. Performing intermediate calculations up to I_{DN} using Eqs. [1] to [5]
3. Use nested for-loop to select optimal ζ and ε for I_D using Eq. [6].
    a. We use the nested for-loop to solve for maximum I_D by using arrays of ζ and ε. Based off of physical constraints seen in Fig. 1, we bound surface azimuth angle ζ from $[0^\circ, 360^\circ]$ and tilt angle ε from $[0^\circ, 90^\circ]$.
4. Initialize collector efficiency constants and calculate for efficiency η_{coll} using Eqs. [7] to [9]. We set up conditions to find differences, if any, between a series and parallel configuration.
    a. Series:
        i. Model the 3 collectors in series as 1 large collector with combined area $A_c = 9.75\ \text{m}^2$.
        ii. Set $\dot{m}_{series} = 0.0317$ kg/s. This selection is arbitrary and based off of previous code.
        iii. Set $T_{ambient}$ as a linearly spaced vector from 9ºC to 16ºC.
    b. Parallel:
        i. Model the 3 collectors in parallel by calculating energy of each collector individually with $A_c = 3.25\ \text{m}^2$; multiply the resulting energy by 3 to obtain system total energy.
        ii. Set $\dot{m}_{parallel} = \frac{1}{3}(\dot{m}_{series})$, assuming evenly divided mass flow through each collector.
        iii. Set $T_{ambient}$ as a linearly spaced vector from 9ºC to 16ºC.

Results:

Through our program analysis, we find that the optimal pair of ζ and ε values that yield the maximum I_D are $\zeta = 180^\circ$ and $\varepsilon = 22.95^\circ$. Furthermore, we find that the total energy incident for series and parallel

configurations is the same; therefore, we conclude that the configuration does not impact our energy collection. These results are supported by Figs. 4(a) and 4(b) below.

In order to select between a parallel or series configuration, we must turn to other design criteria. In general, series collectors are easier to implement than collectors in parallel, which also lowers project costs. As a result, our system design will feature collectors in a series arrangement.

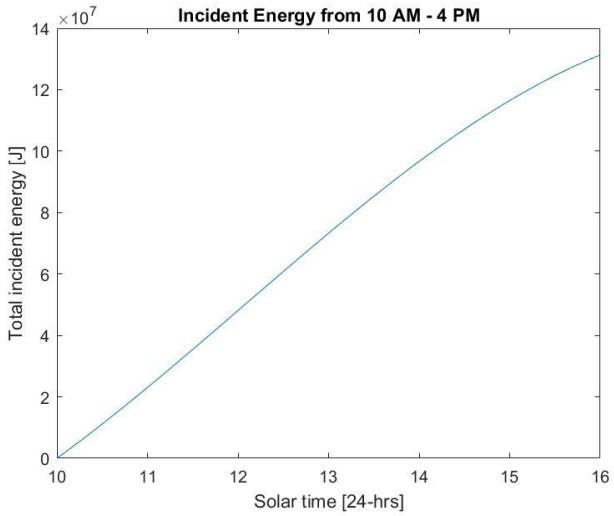

Fig. 4(a): Identical Series & Parallel Results

```
>> task4_a_series
Series configuration:
   Total energy = 131333303.5512 J
   Max efficiency = 0.68372
   Min efficiency = 0.66445
   Optimal surface tilt angle epsilon: 22.95
   Optimal surface azimuth angle zeta: 180
>> task4_a_parallel
Parallel configuration:
   Total energy = 131333303.5512 J
   Max efficiency = 0.68372
   Min efficiency = 0.66445
   Optimal surface tilt angle epsilon: 22.95
   Optimal surface azimuth angle zeta: 180
```

Fig. 4(b): Calculated Result Values

Task 4(a): Optimal Mass Flow Rate

Task description:

The purpose of this task is to determine the optimal mass flow rate of water that would maximize the amount of water that can be heated from 16ºC to 65ºC by the system of three solar collectors, each of area $A_C = 3.25m^2$, from 10AM to 4PM (solar time) on April 30th in Berkeley, CA. The ambient temperature varies linearly between 9ºC and 16ºC from 10AM to 4PM. Once determined, the optimal mass flow rate will be reported and the total amount of water heated as well as the average temperature of water in the tank will be calculated.

Assumptions/Idealizations:

- Assuming the time interval to be from 10AM to 4PM before the shadow is introduced (Task 5a), and from 11AM to 4PM after the shadow is introduced.
- Water must be heated in one pass.
- Assuming the tank where water is collected is perfectly insulated, so that the water in the tank does not lose heat.
- Assuming that our goal is to have the average temperature of all the water collected in the tank at the end of the day (by 4PM) at or above 65ºC. This means that at every time step the water temperature may be higher or lower than 65ºC, but once all the contributions are added up, the average temperature has to be at or above 65ºC.

- Assuming the water pressure loss from arranging the collectors in series is negligible.
- Assuming that the collectors are arranged in series.
- Assuming the collector needs to have a constant mass flow rate of water throughout the day.
- Assuming the azimuth angle of 180° and inclination angle 23°, based on the design optimizations performed in the previous part of the task.
- Assuming the time step size for the integration to be 0.025 hrs.
- Assuming water density at 65°C is 980.55 kg/m^3. This density number is used in gallons-to-kg conversions.

Parameters:

- Collector glazing transmissivity $\tau_g = 0.89$
- Collector absorber plate absorptivity $\alpha_c = 0.85$
- $h_{conv,o} = 7$ W/m^{2}K
- $h_{conv,i} = 3.1$ W/m^{2}K
- $\delta_g = 0.7$ cm
- $\delta_{ins} = 6$ cm
- $k_g = 1.3$ W/mK
- $k_{ins} = 0.045$ W/mK
- $A_c = 3.25$ m^2 * 3 - since there are three collectors, each of the same area
- Specific heat of water: $c_p = 4186$ J/kg°C
- $\lambda = 37.9$°N (Berkeley, CA)
- d = 120 (April 30th)

Additional equations:

$\dot{q} = \dot{m} c_p \Delta T$ - rate of heat (per second) required to increase the temperature of water flowing at mass flow rate $\dot{m}$ by ΔT.

Algorithm summary of the program:

1) Declare variables for collector's location, design parameters, and other constants relevant to the system's operation.
2) Perform intermediate calculations for variables that do not depend on time of day or mass flow rate:
    - Conductance (determined by collector's design parameters)
    - U_{loss} (determined by collector's design parameters)
    - Declination (determined by day of the year)
3) Express variables that are dependent on time of day as functions of time of day.
4) Set up integration inputs and declare the arrays required to store calculation outputs.
5) Set up an outer loop to iterate through potential mass flow rates to find the optimal one.
6) Set up a nested inner loop to iterate through the time steps throughout the day. <u>For each time step</u>:
    - Calculate incident direct radiation values.
    - Determine heat removal factor $\overline{F_R}$ and the corresponding collector efficiency η_{coll}.

- Multiply Incident direct radiation by η_{coll} and by total collector area. This becomes the $\dot{q}$ term for the $\dot{q} = \dot{m}c_p\Delta T$ equation for this time step.
- At each time step for a given mass flow rate value, the ΔT term of the $\dot{q} = \dot{m}c_p\Delta T$ equation will vary depending on the amount of heat ($\dot{q}$) received during that time step. ΔT is calculated as: $\Delta T = \dot{q}/\dot{m}c_p$.
- Increment the total change in temperature by the ΔT amount calculated in the previous step. The sum of ΔT amounts for all time steps will later be divided by the number of time steps to find the average increase in temperature for all the water in the tank. This average increase needs to be at or above 49ºC to meet the requirement of the problem.
- Increment the ambient temperature by the appropriate amount as it is changing linearly from 9ºC to 16ºC between 10AM and 4PM.
- Repeat loop operations for the next time step.

7) For each mass flow rate value that is being tested by the loop:
   - Calculate the total energy collected during the day by integrating $\dot{q}$ values over the time steps, in seconds. The answer will be in Joules, since the units of $\dot{q}$ are J/s.
   - Record the current mass flow value, corresponding total amount of the water heated, and average temperature of the water in the tank.
   - Repeat loop operations for the next mass flow rate.
8) Summarize the results from the above calculations in a matrix showing mass flow rates (column 1), corresponding average temperature increases (column 2), and total amounts of water heated (column 3). Find the first row with average temperature increase above 49ºC. This is the row with the largest total mass of water that can be heated from 16ºC to 65ºC. Display the corresponding mass flow rate, which is the optimal value for this design.
9) The final section of the code plots the total volume of water heated and the corresponding temperature increases against potential mass flow rate values. This plot is presented and discussed below in the Results section.

Results:

Optimal mass flow rate: 0.0304 kg/sec.
Total amount of water heated: 656.64 kg, or 176.91 gallons.
Average temperature of water in the tank at the end of the day: 65.09 ºC.

Observations:

- Even though higher mass flow rate leads to greater total volume of water going through the system, it also pushes the water through the collector faster, allowing for less time for it to get heated. Therefore, there is a competing relationship between heating the water to a certain temperature and heating a maximum amount of water possible.
- Efficiency term η_{coll} (Eqn [7]) tends to increase as the mass flow rate increases, but the mass flow rates at which η_{coll} approaches 1 (the maximum possible value) are not practical for heating the water sufficiently to meet the requirements of this problem.

Table 2 below shows the relationship between mass flow rate, average total change in temperature of water collected, and total amount of water heated. Highlighted in yellow is the row with the highest mass flow rate for which the average change in temperature at the end of the day exceeds 49ºC (i.e. the water is heated from 16ºC to 65ºC). This mass flow rate results in heating 656.64 kg of water. Highlighted in red for reference is the row where the approximate target amount of water (350 gallons, or 1299.125 kg) would be heated by the collector system. It is clear from the table that this volume of water can only be heated to a temperature incerase of 25.79ºC, which would not be sufficient to heat the water from 16ºC to 65ºC.

Mass flow rate (kg/s)	Average change in temp (C)	Total amount of water heated (kg)
0.0200	71.63	432.00
0.0300	49.70	648.00
0.0301	49.54	650.16
0.0302	49.39	652.32
0.0303	49.24	654.48
0.0304	49.09	656.64
0.0305	48.95	658.80
0.0400	38.03	864.00
0.0500	30.80	1080.00
0.06	25.88	1296.00
0.0601	25.83	1298.16
0.0602	25.79	1300.32
0.0603	25.75	1302.48
0.0604	25.71	1304.64
0.0605	25.67	1306.80
0.0700	22.31	1512.00

Table 2: Average Change in Temperature and Total Mass of Water by Mass Flow Rate

Fig.5 below shows the average temperature of water in the tank at the end of the day, as well as the number of gallons of water in the tank, plotted against various mass flow rates. The red stars on the graph indicate the point where the average water temperature at the end of the day reaches 65ºC (on the blue line), and the corresponding number of gallons of water heated (on the red line), which is 176.91 gallons. The blue point on the graph indicates the target volume of water to be heated (350 gallons), which will not be possible to achieve with this collector design.

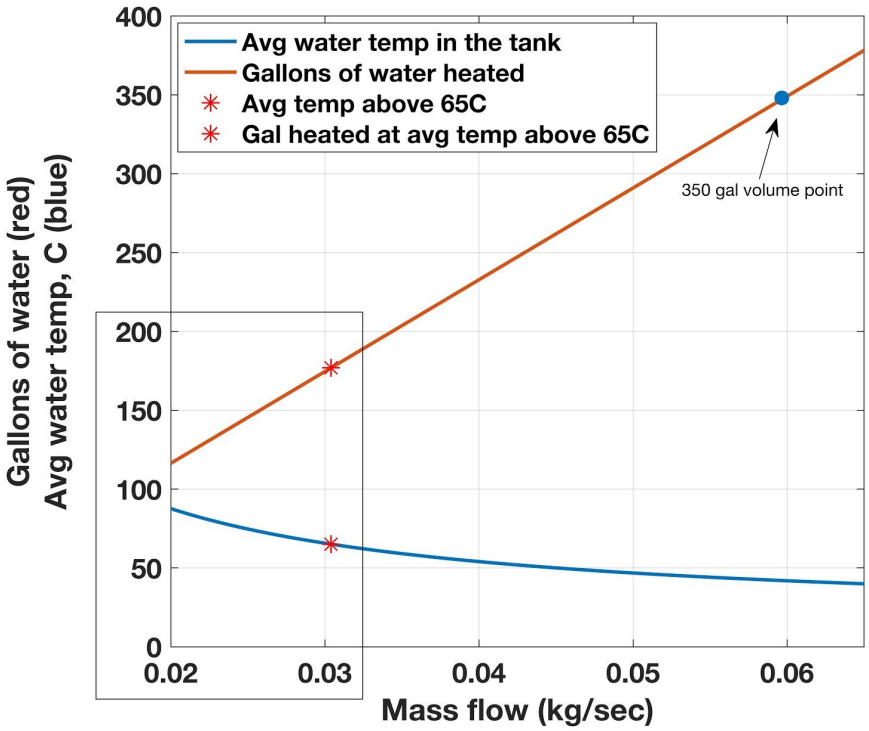

Fig. 5: Volume and Average Temperature of Water by Mass Flow Rate

Task 4(b)

Task description:

The purpose of this task is to determine whether the current collector system will be sufficient to heat 350 gallons of water from 16ºC to 65ºC, and to determine the amount (in kg) of additional natural gas required (if any) to heat the remaining water in order to meet this goal.

Assumptions/Idealizations:

- Same assumptions as in "Task 4(a): Optimal mass flow rate" section.
- Assuming water density at 65ºC is 980.55 kg/m^3.
- Assuming natural gas will be burned at 90% efficiency.
- Assuming the lower heating value of natural gas to be: LHV(CH_4) = 50,050 kJ/kg.

Additional equations:

$Q = mc_p\Delta T$ - total heat required to increase the temperature of water of mass m by ΔT.

Algorithm summary of the program:

It was shown in the previous analysis ("Task 4(a): Optimal mass flow rate" section) that the current collector design would not be able to heat 350 gallons of water from 16ºC to 65ºC during the day for the design parameters and inputs specified. The optimal collector design would only allow to heat 656.64 kg

of water, whereas the equivalent mass of 350 gallons of water at 65°C is 1299.125 kg. This leaves an additional 642.485 kg of water to be heated by natural gas.

1) Convert the volume of water required to heat (350 gallons) into kg, using the density of water at 65°C. Calculate the remaining amount of water (in kg) to be heated.
2) Calculate the total amount of heat (Q) required to heat the remaining mass of water using this formula: $Q = mc_p\Delta T$.
3) To calculate the amount of natural gas required to produce this amount of heat (Q), divide Q by LHV (lower heating value) of natural gas.
4) Since the answer above assumes the efficiency of 100%, divide the result by 0.9 to accommodate for the process with 90% efficiency.

Results:
Total amount of water to be heated: 1299.125 kg
Amount of water heated by collector energy: 656.64 kg
Amount of water that remains to be heated by natural gas: 642.485 kg
Amount of heat required to heat the remaining water: 131782707.5 J = 131.78 MJ
Natural gas required to heat remaining water: 2.93 kg

Task 4(c)

Task description:
The goal of this task is to determine how much CO_2 (in kg) would be prevented from being released into the atmosphere each year if the solar collector system designed above is used instead of natural gas to heat water.

Assumptions/Idealizations:
- Same assumptions as in "Task 4(a): Optimal mass flow rate" section.
- Assume the system designed above operates and collects energy at the same rate on 200 days per year.
- Assume the goal is no longer to heat 350 gallons of water per day, but rather to heat the maximum amount of water possible from 16°C to 65°C. Based on the calculations above, that amount is 656.64 kg of water per day.
- Assume natural gas can be presented as 100% CH_4 for the purposes of this problem.

Algorithm summary of the program:
1) Calculate the total amount of water that can be heated over 200 days using the daily amount calculated in previous analysis ("Task 4(a): Optimal mass flow rate" section).
2) Calculate the total amount of heat per year (200 days) required to heat the amount of water calculated in the previous step using the formula $Q = mc_p\Delta T$.

3) Calculate how much natural gas the would have been required to provide this amount of heat (Q): $m(CH4) = (Q/LHV(CH4))/0.9$ (kg). This formula also accounts for the process efficiency of 90%.
4) Using the molar mass of CH_4, calculate the number of moles equivalent to the mass of CH4 calculated above: $\nu(CH4) = m(CH4)\ in\ grams/M(CH4)$ (mol).
5) Based on the chemical reaction of burning CH_4, the number of moles of CO_2 is the same as CH_4: $\nu(CO2) = \nu(CH4)$. Calculate the mass of CO_2 as follows:
   $m(CO2)\ in\ grams = \nu(CO2) * M(CO2)$, where M(CO_2) is the molar mass of CO_2.
6) Convert the result from grams back to kilograms for the final answer.

Results:
Amount of CH4 that would be saved yearly: 598 kg
Amount of CO2 that would be prevented from being released yearly: 1644.52 kg

Task 5(a)

Task description:
The shadow from a neighbouring building effectively changes the daily integration time interval from 10AM-4PM to 11AM-4PM, since the shadow is blocking the sunlight to the collector before 11AM at all times of the year. This task also incorporates the following additional assumption: during the year, on average, 46% of the hours between 10AM and 4PM are cloudy and provide no energy input. The purpose of this task is to determine the following:

a) Impact of a shadow from a neighbouring building on the amount of energy collected by the solar collector system per year.
b) Amount of natural gas required to compensate for the loss in energy collected.
c) Cost of natural gas required to compensate for the loss in energy collected.

Assumptions/Idealizations:

- Same assumptions as in "Task 4(a): Optimal mass flow rate" section.
- Assume that during the year, on average, 46% of the hours between 10AM and 4PM are cloudy and provide no energy input. For the purposes of this problem, we are also assuming that this percentage remains the same for the time interval between 11AM and 4PM.
- Assume all collector parameters (such as azimuth angle ζ and inclination angle ε) remain the same throughout the year.
- The "loss due to shadow" is defined as the difference in yearly energy output between the system that can collect energy from 10AM to 4PM and the system that can only collect energy from 11AM to 4PM, both under the assumed average cloudy conditions of 46% per day mentioned above.
- Assume the price of natural gas to be $12.36 per thousand cubic feet. Source: www.eia.gov (price of natural gas for CA residents as of June 2018).

- Assume the density of natural gas to be 0.81 kg/m^3. Density of natural gas in kg/m^3 is between 0.7 and 0.92 (source: www.engineeringtoolbox.com). For this problem, let us use the value from the middle of this range: 0.81 kg/m^3.

Algorithm summary of the program:

a) Yearly loss in energy collection due to the shadow:
   1. Create a function for yearlyEnergy that can accept times of day as inputs. The output of this function is the yearly energy amount collected by the system. The logic of this function is using the code from the previous steps of the problem with one modification: it also calculates daily energy amounts at two week intervals for the entire year, and then integrates those daily energy contributions over the entire year to calculate the total yearly energy amount. This is achieved by using an additional loop with the "day of the year" (d) as it's variable. The increment for this loop is 14, and it runs from zero to 365 to cover the entire year.
   2. Call the yearlyEnergy function twice: First time, using the original energy collection start time (10AM); Second time, using the new energy collection start time (11AM). The difference between the two values represents the yearly loss in energy collection due to the shadow.

b) Amount of natural gas required to compensate for the loss in energy collection:
   Divide the yearly energy loss value calculated in the previous step by the LHV (lower heating value) of natural gas, and divide the answer by 0.9 to account for 90% efficiency of the process. The answer is in kg (Joules divided by Joules/kg).

c) Yearly cost of natural gas required to compensate for the loss in energy collection:
   Since the prices are typically in dollars per thousand cubic feet, first convert the mass of natural gas calculated in the previous step into volume, in cubic feet. Next, multiply the answer by the current available residential prices for natural gas.

Results:

Total energy collection before the shadow from the new building: 26197.9037 MJ per year.
Total energy collection after the shadow from the new building: 21544.8758 MJ per year.
Energy collection is reduced by 4653.0279 MJ per year due to shadow from the new building.
Natural gas required per year to compensate: 103.30 kg.
Yearly cost of additional natural gas: 55.66 dollars.

Task 5(b) (ME 246 students only)

Task description:

In this task, a motorized frame is incorporated into the collector design, allowing for the azimuth angle ζ of the collector to always match the solar azimuth angle ξ. The goal is to determine if this design modification will produce enough extra energy to compensate for the energy loss due to shadow from the neighbouring building. The following will be calculated:

a) Total energy collected with and without the motorized frame over the year.

b) Percentage increase in yearly collected energy compared to the original fixed frame design.
c) Comparison between the energy gained from the motorized frame design and the energy lost due to shadow for the original fixed frame design over a full year.

Assumptions/Idealizations:

- Same assumptions as in "Task 4(a): Optimal mass flow rate" section.
- Assume that during the year, on average, 46% of the hours between 10AM and 4PM are cloudy and provide no energy input. For the purposes of this problem, we are also assuming that this percentage remains the same for the time interval between 11AM and 4PM.
- Assume all collector parameters (such as inclination angle ε) remain the same throughout the year, unless otherwise specified.
- The "loss due to shadow" is defined as the difference in yearly energy output between the system that can collect energy from 10AM to 4PM and the system that can only collect energy from 11AM to 4PM, both under the assumed average cloudy conditions of 46% per day mentioned above.
- Assume that at each time step, the azimuth angle ζ of the collector is set to match the solar azimuth angle ξ by the motorized frame design modification.

Algorithm summary of the program:

a) Total energy collected with and without the motorized frame over the year:
    1. In addition to the yearlyEnergy function created for the previous task, create another function for yearlyEnergy_motorFrame. Both functions will accept times of day as inputs and provide the yearly energy amount collected by the system as outputs. The logic of the new yearlyEnergy_motorFrame function is very similar to yearlyEnergy, but the main difference is that azimuth angle ζ is set to equal solar azimuth angle ξ at every time step for the motorized frame design.
    2. Call the yearlyEnergy and yearlyEnergy_motorFrame functions using the new energy collection start time (11AM) for both. The difference between the two values represents the yearly increase in energy collected due to the motorized frame design..
b) Percentage increase in yearly collected energy compared to the original fixed frame design.
    Use the difference between the two values calculated above and express it as a percentage of the original energy collected with the fixed frame design.
c) Comparison between the energy gained from the motorized frame design and the energy lost due to shadow for the original fixed frame design over a full year.
    Compare the amount of energy lost calculated in Task 5a to the additional energy produced by motorized frame design.

Results:

Total energy collected <u>without motorized frame:</u> 21.5449 GJ per year.
Total energy collected <u>with motorized frame:</u> 23.8606 GJ per year.
<u>Percentage increase</u> in yearly collected energy: 10.7482 %.
Energy collection is reduced by 4.653 GJ per year due to shadow from the new building.

Energy collection is increased by 2.3157 GJ per year due to motorized frame design.

Motorized frame design allows to offset only ~50 % of the energy loss due to shadow, so motorized frame collector would not produce enough extra energy to compensate for the loss due to new building's shadow. A potential design improvement suggestion would be to also modify the tilt angle ε throughout the year to maximize the amount of incident direct radiation on the collector. Having ε set to lower values (more flat) in the summer and higher values (more tilted) in the winter would accommodate for the fact that the sun is higher in the sky during the summer and lower in the winter.

Appendix A: MATLAB Code

Task 1

```
%% Task 1
% Problem statement:
%     surface azimuth, zeta = 200 deg.
%     tilt angle, eps = 36 deg.
%     day = 4/30/2018
%     time = 1 PM = 13.00 hrs
%     location = Berkeley, CA; lamb = 37.9 deg.
% *All sin/cos/tan and angle entries/calcs are in degrees

%% Initializing constants
lamb = 37.9;
zeta = 200;
eps = 36;
t = 13; %Solar time, [hrs (decimal)]
d = 120;
A = 1310; %[W/m^2]
B = 0.18; %Unitless

%% Intermediate Calculations
a = 15*(t-12); %Hour Angle, [deg]
delta = 23.44*sind(360/365.25*(d-80));
X = acosd(sind(lamb)*sind(delta)+cosd(lamb)*cosd(delta)*cosd(a));

tand_E = sind(a)/(sind(lamb)*cosd(a)-cosd(lamb)*tand(delta));
if a>=0
   if tand_E >= 0
      E = 180 + atand(tand_E);
   elseif tand_E < 0
      E = 360 + atand(tand_E);
   end
elseif a<0
   if tand_E >= 0
      E = atand(tand_E);
   elseif tand_E < 0
      E = 180 + atand(tand_E);
   end
end

%% Final Calculations
% Units should be [W/m^2]

I_dn = A*exp(-B/sind(90-X));
I_d = I_dn*(cosd(X)*cosd(eps)+ sind(eps)*sind(X)*cosd(E-zeta));
disp("Direct solar incident heat flux: " +I_d + " W");

%% Results

% Direct solar incident heat flux: 1048.3376 W
```

Task 2

```
%% Task 2
% Problem statement:
%     surface azimuth, zeta = 200 deg.
%     tilt angle, eps = 36 deg.
%     day = 4/30/2018
%     time =10 AM to 4:00PM = 10.00 - 16.00 hrs
% *All sin/cos/tan and angle entries/calcs are in degrees

%% Initializing constants
lamb = 37.9;
zeta = 200;
eps = 36;
t = linspace(10.00, 16.00, 601); %Solar time, [hrs (decimal)]
d = 120;
A = 1310; %[W/m^2]
B = 0.18; %Unitless

%% Intermediate Calculations
a = 15*(t-12); %Hour Angle, [deg] (array)
delta = 23.44*sind(360/365.25*(d-80));
X = acosd(sind(lamb)*sind(delta)+cosd(lamb)*cosd(delta)*cosd(a));

for i = 1:length(a)
   tand_E(i) = sind(a(i))/(sind(lamb)*cosd(a(i))-cosd(lamb)*tand(delta));
   if a(i)>=0
      if tand_E(i) >= 0
         E(i) = 180 + atand(tand_E(i));
      elseif tand_E(i) < 0
         E(i) = 360 + atand(tand_E(i));
      end
   elseif a(i)<0
      if tand_E(i) >= 0
         E(i) = atand(tand_E(i));
      elseif tand_E(i) <= 0
         E(i) = 180 + atand(tand_E(i));
      end
   end
end

%% Final Calculations
% Units should be [W/m^2]

I_dn = A*exp(-B./sind(90-X));
I_d = I_dn.*(cosd(X)*cosd(eps)+ sind(eps)*sind(X).*cosd(E-zeta));

%% Time Step Verification and Integration
for j = 1:length(I_d)-1
   new(j)=abs(100*(I_d(j+1)-I_d(j))/I_d(j+1))>2;
end

Z = cumtrapz(t*3600,I_d); %Converting t from [hrs] to [s]
plot(t,Z);
xlabel("Solar time [24-hrs]");
ylabel("Total incident energy [J]");
title("Incident Energy from 10 AM - 4 PM");
```

```
disp("Total energy: " + Z(length(Z)) + " J");

%% Results

% Total incident energy: 20036071.2047 J
```

Task 3

```
%% Task 3
% Problem statement:
%       Include effect of solar collector efficiency
%       Determine efficiency of 200 deg. azimuth, tilt angle 36 deg., solar
%       time is between 2-3 PM [14.00-15.00 hrs]
%
%% Initializing constants
lamb = 37.9;
zeta = 200;
eps = 36;
t = linspace(14.00, 15, 101); %Solar time, [hrs (decimal)]
d = 120;
A = 1310; %[W/m^2]
B = 0.18; %Unitless

%% Intermediate Calculations
a = 15*(t-12); %Hour Angle, [deg] (array)
delta = 23.44*sind(360/365.25*(d-80));
X = acosd(sind(lamb)*sind(delta)+cosd(lamb)*cosd(delta)*cosd(a));

for i = 1:length(a)
   tand_E(i) = sind(a(i))/(sind(lamb)*cosd(a(i))-cosd(lamb)*tand(delta));
   if a(i)>=0
      if tand_E(i) >= 0
         E(i) = 180 + atand(tand_E(i));
      elseif tand_E(i) < 0
         E(i) = 360 + atand(tand_E(i));
      end
   elseif a(i)<0
      if tand_E(i) >= 0
         E(i) = atand(tand_E(i));
      elseif tand_E(i) <= 0
         E(i) = 180 + atand(tand_E(i));
      end
   end
end

%% Incident Direct Solar Flux
% Units should be [W/m^2]

I_dn = A*exp(-B./sind(90-X));
I_d = I_dn.*(cosd(X)*cosd(eps)+ sind(eps)*sind(X).*cosd(E-zeta));

%% Collector Efficiency
ho = 7; %[W/(m^2*K)]
hi = 3.1; %[W/(m^2*K)]
dg = 0.007; %[cm]
kg = 1.3; %[W/mK]
d_ins = 0.06; %[m], from 6 cm
k_ins = 0.018; %[W/mK]
tau = 0.89; %[unitless]
ac = 0.85; %[unitless]
cp = 4186; %[J/(kg*C)]
mdot = 0.0267; %[0.0267 kg/s]

T_amb = 12; %[Celsius]
```

```
T_in = 16; %[Celsius]
Area = 1; %assume 1 m^2

COND = 1/(1/(ho*Area) + dg/(kg*Area) + 1/(hi*Area)) + 1/(1/(ho*Area) + d_ins/(k_ins*Area)); %constant
U_loss = COND/Area;
FR = (1-exp(-COND/(mdot*cp)))/(COND/(mdot*cp));

eff = FR*(tau*ac - U_loss./I_d*(T_in-T_amb)); %eff is array

%% Trapezoidal Integration with Efficiency

ID_actual = I_d.*eff; %[array of W/m^2]
ID_total = cumtrapz(t*3600,ID_actual); %[W/m^2] *3600 for [hrs] to [s]
E_total = ID_total*Area; %[W], TOTAL ENERGY

ID_total_noEff = cumtrapz(t*3600,I_d);
E_total_noEff = ID_total_noEff*Area;

plot(t, E_total);
xlabel("Solar time [24-hrs]");
ylabel("Total incident energy [J]");
title("Incident Energy from 2 PM - 3 PM (k_i_n_s = 0.018)");

disp("For k_ins = " + k_ins + "...");
disp("   Total ideal energy (w/out eff.) = " + E_total_noEff(length(E_total_noEff)) + " J");
disp("   Total actual energy = " + E_total(length(E_total)) + " J");
disp("   Efficiency = " + E_total(length(E_total))/E_total_noEff(length(E_total_noEff)));

%% RESULTS

% For k_ins = 0.045...
%    Total ideal energy (w/out eff.) = 3279986.097 J
%    Total actual energy = 2410633.0218 J
%    Efficiency = 0.73495
%
% For k_ins = 0.018...
%    Total ideal energy (w/out eff.) = 3279986.097 J
%    Total actual energy = 2420375.7269 J
%    Efficiency = 0.73792

%% CONCLUSION

%   After decreasing insulation thermal conductivity to 0.018 W/mK, total
%   energy increased by 9742.71 J and efficiency increased by 0.00297.
```

Task 4(a): Optimal Collector Angles & Arrangement

Series Arrangement

```
%% Task 4 - SERIES
% Problem statement:
%   * Design params: zeta (surf. az); eps (surf. tilt); mdot (mass flow
%         rate)
%   * T_out: 65 oC
%   * Solar time: 10.00 -> 16.00 (10 AM - 4 PM)
%   * qdot [W] = mdot*cp*dT = E_tot
%       * cp*dT = 4186*(65-16) = 205114
%       * mdot = E_tot/205114

%% Initializing constants
% zeta and eps will be initialized before solar flux calculations

lamb = 37.9;
t = linspace(10.00, 16, 601); %Solar time, [hrs (decimal)]
d = 120;
A = 1310; %[W/m^2]
B = 0.18; %Unitless

%% Intermediate Calculations
a = 15*(t-12); %Hour Angle, [deg] (array)
delta = 23.44*sind(360/365.25*(d-80));
X = acosd(sind(lamb)*sind(delta)+cosd(lamb)*cosd(delta)*cosd(a));

for i = 1:length(a)
   tand_E(i) = sind(a(i))/(sind(lamb)*cosd(a(i))-cosd(lamb)*tand(delta));
   if a(i)>=0
      if tand_E(i) >= 0
         E(i) = 180 + atand(tand_E(i));
      elseif tand_E(i) < 0
         E(i) = 360 + atand(tand_E(i));
      end
   elseif a(i)<0
      if tand_E(i) >= 0
         E(i) = atand(tand_E(i));
      elseif tand_E(i) <= 0
         E(i) = 180 + atand(tand_E(i));
      end
   end
end

%% Incident Direct Solar Flux; Eps, Zeta Optimization
% Units should be [W/m^2]

I_dn = A*exp(-B./sind(90-X));
eps = linspace(0,90,601); %DESIGN PARAM: SURFACE TILT ANGLE
zeta = linspace(0,360,601); %DESIGN PARAM: SURFACE AZIMUTH ANGLE

maxI_d = 0;
OPT_eps = 0;
OPT_zeta = 0;

for i=1:601
   for j=1:601
      I_d = I_dn.*(cosd(X).*cosd(eps(i))+ sind(eps(i)).*sind(X).*cosd(E-zeta(j)));
      if max(I_d)>maxI_d
         maxI_d = max(I_d);
         OPT_eps = eps(i);
```

```
            OPT_zeta = zeta(j);
        end
    end
end

% Maximized I_d
I_d = I_dn.*(cosd(X).*cosd(OPT_eps)+ sind(OPT_eps).*sind(X).*cosd(E-OPT_zeta));

%% Collector Efficiency
ho = 7; %[W/(m^2*K)]
hi = 3.1; %[W/(m^2*K)]
dg = 0.007; %[cm]
kg = 1.3; %[W/mK]
d_ins = 0.06; %[m], from 6 cm
k_ins = 0.045; %[W/mK]
tau = 0.89; %[unitless]
ac = 0.85; %[unitless]
cp = 4186; %[J/(kg*C)]
mdot = 0.0317; %DESIGN PARAM: MASS FLOW RATE

T_amb = linspace(9, 16, 601); %[Celsius] VARIES LINEARLY between 9-16 oC
T_in = 16; %[Celsius]
Area = 9.75; %assume 9.75 m^2, ALL COLLECTORS

COND = 1/(1/(ho*Area) + dg/(kg*Area) + 1/(hi*Area)) + 1/(1/(ho*Area) + d_ins/(k_ins*Area)); %constant
U_loss = COND/Area;
FR = (1-exp(-COND/(mdot*cp)))/(COND/(mdot*cp));

eff = FR*(tau*ac - U_loss./I_d.*(T_in-T_amb)); %eff is array, includes I_d

%% Trapezoidal Integration with Efficiency

ID_actual = I_d.*eff; %[array of W/m^2]
ID_total = cumtrapz(t*3600,ID_actual); %[J/m^2], integration
E_coll = ID_total*Area; %[J], TOTAL ENERGY
E_total = E_coll; % SERIES - TOTAL AREA

%% Results

plot(t, E_total);
xlabel("Solar time [24-hrs]");
ylabel("Total incident energy [J]");
title("Incident Energy from 10 AM - 4 PM");

disp("Series configuration: ");
disp("   Total energy = " + E_total(length(E_total)) + " J");
disp("   Max efficiency = " + max(eff));
disp("   Min efficiency = " + min(eff));
disp("   Optimal surface tilt angle epsilon: " + OPT_eps);
disp("   Optimal surface azimuth angle zeta: " + OPT_zeta);
```

Parallel Arrangement

```
%% Task 4 - PARALLEL
% Problem statement:
%   * Design params: zeta (surf. az); eps (surf. tilt); mdot (mass flow
%         rate)
%   * T_out: 65 oC
%   * Solar time: 10.00 -> 16.00 (10 AM - 4 PM)
%   * qdot [W] = mdot*cp*dT = E_tot
%      * cp*dT = 4186*(65-16) = 205114
%      * mdot = E_tot/205114

%% Initializing constants
% zeta and eps will be initialized before solar flux calculations

lamb = 37.9;
t = linspace(10.00, 16, 601); %Solar time, [hrs (decimal)]
d = 120;
A = 1310; %[W/m^2]
B = 0.18; %Unitless

%% Intermediate Calculations
a = 15*(t-12); %Hour Angle, [deg] (array)
delta = 23.44*sind(360/365.25*(d-80));
X = acosd(sind(lamb)*sind(delta)+cosd(lamb)*cosd(delta)*cosd(a));

for i = 1:length(a)
   tand_E(i) = sind(a(i))/(sind(lamb)*cosd(a(i))-cosd(lamb)*tand(delta));
   if a(i)>=0
      if tand_E(i) >= 0
         E(i) = 180 + atand(tand_E(i));
      elseif tand_E(i) < 0
         E(i) = 360 + atand(tand_E(i));
      end
   elseif a(i)<0
      if tand_E(i) >= 0
         E(i) = atand(tand_E(i));
      elseif tand_E(i) <= 0
         E(i) = 180 + atand(tand_E(i));
      end
   end
end

%% Incident Direct Solar Flux
% Units should be [W/m^2]

I_dn = A*exp(-B./sind(90-X));
eps = linspace(0,90,601); %DESIGN PARAM: SURFACE TILT ANGLE
zeta = linspace(0,360,601); %DESIGN PARAM: SURFACE AZIMUTH ANGLE

maxI_d = 0;
OPT_eps = 0;
OPT_zeta = 0;

for i=1:601
   for j=1:601
      I_d = I_dn.*(cosd(X).*cosd(eps(i))+ sind(eps(i)).*sind(X).*cosd(E-zeta(j)));
      if max(I_d)>maxI_d
         maxI_d = max(I_d);
         OPT_eps = eps(i);
         OPT_zeta = zeta(j);
      end
   end
end
```

```
% Maximized I_d
I_d = I_dn.*(cosd(X).*cosd(OPT_eps)+ sind(OPT_eps).*sind(X).*cosd(E-OPT_zeta));

%% Collector Efficiency
ho = 7; %[W/(m^2*K)]
hi = 3.1; %[W/(m^2*K)]
dg = 0.007; %[cm]
kg = 1.3; %[W/mK]
d_ins = 0.06; %[m], from 6 cm
k_ins = 0.045; %[W/mK]
tau = 0.89; %[unitless]
ac = 0.85; %[unitless]
cp = 4186; %[J/(kg*C)]
mdot = 0.0317/3; %DESIGN PARAM: MASS FLOW RATE, MUST BE DIVIDED BY 3

T_amb = linspace(9, 16, 601); %[Celsius] VARIES LINEARLY between 9-16 oC
T_in = 16; %[Celsius]
Area = 3.25; %assume 3.25 m^2, ONE COLLECTOR

COND = 1/(1/(ho*Area) + dg/(kg*Area) + 1/(hi*Area)) + 1/(1/(ho*Area) + d_ins/(k_ins*Area)); %constant
U_loss = COND/Area;
FR = (1-exp(-COND/(mdot*cp)))/(COND/(mdot*cp));

eff = FR*(tau*ac - U_loss./I_d.*(T_in-T_amb)); %eff is array, includes I_d

%% Trapezoidal Integration with Efficiency

ID_actual = I_d.*eff; %[array of W/m^2]
ID_total = cumtrapz(t*3600,ID_actual); %[J/m^2], *3600 to convert from [hrs] to [s]
E_coll = ID_total*Area; %[J], TOTAL ENERGY
E_total = 3*E_coll; % PARALLEL - 3*individual collectors

%% Results

plot(t, E_total);
xlabel("Solar time [24-hrs]");
ylabel("Total incident energy [J]");
title("Incident Energy from 10 AM - 4 PM");

disp("Parallel configuration: ");
disp("   Total energy = " + E_total(length(E_total)) + " J");
disp("   Max efficiency = " + max(eff));
disp("   Min efficiency = " + min(eff));
disp("   Optimal surface tilt angle epsilon: " + OPT_eps);
disp("   Optimal surface azimuth angle zeta: " + OPT_zeta);
```

Task 4(a): Optimal Mass Flow Rate

```
clc
close all

% constants, inputs, collector's characteristics
A = 1310; % W/m2
B = 0.18;
lambda = 37.9; % latitude of Berkeley, CA
d = 120; % day of the year; Day 120 is Apr 30, 2018
tau_g = 0.89; % collector glazing transmissivity
alpha_c = 0.85; % collector absorptivity
Cp = 4186; % J/kg*K
h_conv_o = 7; % W/m2*K, outside air convective heat transfer coefficient
delta_g = 0.7*10^-2; % cm converted to m; glazing thickness
Kg = 1.3; % W/m*K; glazing thermal conductivity
h_conv_i = 3.1; % W/m2*K, inside convection coefficient (between absorber and glazing)
delta_ins = 6*10^-2; % cm converted to m; insulation thickness
Kins = 0.045; % W/m*K, insulation thermal conductivity
Ac = 3.25*3; % 3 solar collectors, each has area of 3.25 m^2

% Formulas for efficiency calculation
% Total conductance is Uloss times area or the collector:
conductance = ((1/(h_conv_o*Ac))+(delta_g/(Kg*Ac))+1/(h_conv_i*Ac))^-1+((1/(h_conv_o*Ac))+(delta_ins/Kins*Ac))^-1;
Uloss = conductance/Ac;

% Intermediate formulas
dec = 23.44*sind((360/365.25)*(d-80)); % declination
alpha = @(t) ((15*(t-12))); % hour angle (function of time of day)
zen_angle = @(t) rad2deg(acos(sind(lambda)*sind(dec) + cosd(lambda)*cosd(dec)*cosd(alpha(t)))); % zenith angle
sol_azimuth_tan = @(t) sind(alpha(t))/(sind(lambda)*cosd(alpha(t))-cosd(lambda)*tand(dec)); % tangent of solar azimuth
Int = @(t) A*exp((-B/sind((90)-zen_angle(t)))); % Intensity

% Integration inputs
t0 = 10; % 10AM
tfinal = 16; %4PM
h = 0.025; % time step size
num_steps = (tfinal - t0)/h; % number of steps given step size
Ti = 16; % water inlet temperature
% Ambient temperature varies from 9C to 16C during the time interval
T0 = 9;
Tfinal = 16;
% Using the number of steps to determine change in T for each step
T_step = (Tfinal - T0)/num_steps;

tot_energy = []; % array for Total Energy values
element=1; % first array index as 1
m_dot_values = []; % array for mass flow values
m_total = []; % array for total mass of water heated
dT_avg = []; % array for average change in water temperature

for m_dot = 0.02:0.0001:0.07
  surf_azimuth = 180;
  surf_inc = 23;
  q_dot = []; % array for q-dot values
  index=1; % first array index as 1
  Ta = T0; % ambient temperature at the beginning of the process is set to T0
  dT = []; % change in temperature of water at each time step
  dT_sum = 0; % total change in temperature of water
```

```
  for t = t0:h:tfinal %from time t0 to tfinal, in 0.025 hr increments
  % Determine solar azimuth angle given alpha and tan of solar azimuth angle.
      if (alpha(t) > 0)
        if(sol_azimuth_tan(t) > 0)
          sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
          sol_azimuth = @(t) 360 + rad2deg(atan(sol_azimuth_tan(t)));
        end
      elseif (alpha(t) < 0)
        if(sol_azimuth_tan(t) > 0)
          sol_azimuth = @(t) rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
          sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        end
      end
      % Incident Direct Radiation (W/m2)
      Id = @(t) Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth));
      % Heat removal factor:
      Fr = (1 - exp(-conductance/(m_dot*Cp)))/(conductance/(m_dot*Cp));
      efficiency =  @(t) Fr*((tau_g*alpha_c) - (Uloss/Id(t))*(Ti - Ta));
      % Output is Incident Direct Radiation adjusted for collector
      % efficiency and multiplied by collector area.
      q_dot(index) = Id(t)*efficiency(t)*Ac; % Units: J/s
      % Calculating change in temperature based on q = mCp(dT) equation:
      dT (index) = (q_dot(index)/(m_dot*Cp));
      % Adding the change in temperature to the total to find average
      % change in temperature later:
      dT_sum = dT_sum + dT(index);
      % Increment array indexes and change in ambient temperature:
      index = index + 1;
      Ta = Ta + T_step;
  end
      % Use trapz MATLAB function (time_steps as x and q_dot(t) as y)
      time_steps = [t0:h:tfinal]*3600; % Multiply by 3600 to convert to seconds
      % Total energy collected during the day (in Joules):
      tot_energy (element) = trapz(time_steps,q_dot);

      m_dot_values (element) = m_dot; % current mass flow value
      m_total (element) = m_dot*(tfinal-t0)*3600; % total mass of water that can be heated
      dT_avg (element) = dT_sum/num_steps; % average change in water temperature in the tank
      element = element + 1; % increment the index for m_values, m_total, dT_avg

end

% Transpose vectors for total energy, azimuth, and inclination to columns:
tot_energy_transposed = tot_energy';
m_dot_values_transposed = m_dot_values';
m_total_transposed = m_total';
dT_avg_transposed = dT_avg';

% Concatenate the column vectors into a results matrix:
results = (horzcat(m_dot_values_transposed,dT_avg_transposed,m_total_transposed));
disp(results);

format long g

% Create a new results matrix with change in temperature greater than 49C:
new = results(:,2) > 49;
results_new = results(new,:);
[~,ind] = max(results_new(:,3)); % find the index of the row with maximum total mass
```

```
optimal_mass_flow = results_new(ind,1); % display the corresponding mass flow
disp('The optimal mass flow for this design is: ');
disp(optimal_mass_flow);

% Plot the average water temperature and volume of water heated (in gallons)
% for each mass flow value:
avg_temp = dT_avg + 16; % convert from delta T to total T of water
density = 980.55; % kg/m3; water density at 65C
gallons_water = vol*264.172;

plot(m_dot_values,avg_temp,m_dot_values,gallons_water)
xlabel('mass flow')
hold on
plot(m_dot_values(ind),avg_temp(ind),'r*',m_dot_values(ind),gallons_water(ind),'r*')
```

Tasks 4(b) and 4(c)

```
clc
close all

% constants, inputs, collector's characteristics
A = 1310; % W/m2
B = 0.18;
lambda = 37.9; % latitude of Berkeley, CA
d = 120; % day of the year; Day 120 is Apr 30, 2018
tau_g = 0.89; % collector glazing transmissivity
alpha_c = 0.85; % collector absorptivity
Cp = 4186; % J/kg*K
h_conv_o = 7; % W/m2*K, outside air convective heat transfer coefficient
delta_g = 0.7*10^-2; % cm converted to m; glazing thickness
Kg = 1.3; % W/m*K; glazing thermal conductivity
h_conv_i = 3.1; % W/m2*K, inside convection coefficient (between absorber and glazing)
delta_ins = 6*10^-2; % cm converted to m; insulation thickness
Kins = 0.045; % W/m*K, insulation thermal conductivity
Ac = 3.25*3; % 3 solar collectors, each has area of 3.25 m^2

% Formulas for efficiency calculation
% Formulas for efficiency calculation
% Total conductance is Uloss times area or the collector:
conductance = ((1/(h_conv_o*Ac))+(delta_g/(Kg*Ac))+1/(h_conv_i*Ac))^-1+((1/(h_conv_o*Ac))+(delta_ins/Kins*Ac))^-1;
Uloss = conductance/Ac;

% Intermediate formulas
dec = 23.44*sind((360/365.25)*(d-80)); % declination
alpha = @(t) ((15*(t-12))); % hour angle (function of time of day)
zen_angle = @(t) rad2deg(acos(sind(lambda)*sind(dec) + cosd(lambda)*cosd(dec)*cosd(alpha(t)))); % zenith angle
sol_azimuth_tan = @(t) sind(alpha(t))/(sind(lambda)*cosd(alpha(t))-cosd(lambda)*tand(dec)); % tangent of solar azimuth
Int = @(t) A*exp((-B/sind((90)-zen_angle(t)))); % Intensity

% Integration inputs
t0 = 10; % 10AM
tfinal = 16; %4PM
h = 0.025; % time step size
num_steps = (tfinal - t0)/h; % number of steps given step size
Ti = 16; % water inlet temperature
% Ambient temperature varies from 9C to 16C during the time interval
T0 = 9;
Tfinal = 16;
% Using the number of steps to determine change in T for each step
T_step = (Tfinal - T0)/num_steps;

tot_energy = []; % array for Total Energy values
element=1; % first array index as 1
m_dot_values = []; % array for mass flow values
m_total = []; % array for total mass of water heated
dT_avg = []; % array for average change in water temperature

m_dot =  0.0304; % kg/s; optimal value determined from previous optimizations
surf_azimuth = 180;
surf_inc = 23;
q_dot = []; % array for q-dot values
index=1; % first array index as 1
```

```
Ta = T0; % ambient temperature at the beginning of the process is set to T0
dT = []; % change in temperature of water at each time step
dT_sum = 0; % total change in temperature of water

  for t = t0:h:tfinal %from time t0 to tfinal, in 0.025 hr increments
  % Determine solar azimuth angle given alpha and tan of solar azimuth angle.
     if (alpha(t) > 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 360 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     elseif (alpha(t) < 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     end
     % Incident Direct Radiation (W/m2)
     Id = @(t) Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth));
     % Heat removal factor:
     Fr = (1 - exp(-conductance/(m_dot*Cp)))/(conductance/(m_dot*Cp));
     efficiency =  @(t) Fr*((tau_g*alpha_c) - (Uloss/Id(t))*(Ti - Ta));
     % Output is Incident Direct Radiation adjusted for collector
     % efficiency and multiplied by collector area.
     q_dot(index) = Id(t)*efficiency(t)*Ac; % Units: J/s
     % Calculating change in temperature based on q = mCp(dT) equation:
     dT (index) = (q_dot(index)/(m_dot*Cp));
     % Adding the change in temperature to the total to find average
     % change in temperature later:
     dT_sum = dT_sum + dT(index);
     % Increment array indexes and change in ambient temperature:
     index = index + 1;
     Ta = Ta + T_step;
  end
     % Use trapz MATLAB function (time_steps as x and q_dot(t) as y)
     time_steps = [t0:h:tfinal]*3600; % Multiply by 3600 to convert to seconds
     % Total energy collected during the day (in Joules):
     tot_energy (element) = trapz(time_steps,q_dot);

     m_dot_values (element) = m_dot; % current mass flow value
     m_total (element) = m_dot*(tfinal-t0)*3600; % total mass of water that can be heated
     dT_avg (element) = dT_sum/num_steps; % average change in water temperature in the tank
     element = element + 1; % increment the index for m_values, m_total, dT_avg

% Transpose vectors for total energy, azimuth, and inclination to columns:
tot_energy_transposed = tot_energy';
m_dot_values_transposed = m_dot_values';
m_total_transposed = m_total';
dT_avg_transposed = dT_avg';

% Concatenate the column vectors into a results matrix:
results = (horzcat(m_dot_values_transposed,dT_avg_transposed,m_total_transposed));
format long g

%PART B
Tstart = 16; % starting temperature of water, in C
Tfinish = 65; % target temperature of water, in C
m3_water = 350/264.172; % m3; volume of water to heat, converted from 350 gallons to m3
density = 980.55; % kg/m3; water density at 65C
```

```
m_req = m3_water*density; % converting 350 gallons of water to kilograms
m_left = m_req - m_total; % how much water still needs to be heated by CH4
Q = m_left*Cp*(Tfinish-Tstart); % heat required to heat the remaining water
CH4_LHV = 50050000; % in J/kg; lower heating value of CH4
CH4_kg_ideal = Q/CH4_LHV; % how much CH4 it would take if the process was 100% efficient
CH4_kg_90eff = CH4_kg_ideal/0.9; % how much CH4 it would take for the 90% efficient process

disp("-----PART B-----");
disp("Total amount of water to be heated:   " + m_req + " kg");
disp("Amount of water heated by collector energy:   " + m_total + " kg");
disp("Amount of water that remains to be heated by natural gas:   " + m_left + " kg");
disp("Amount of heat required to heat the remaining water:   " + Q + " J");
disp("Natural gas required to heat remaining water:   " + CH4_kg_90eff + " kg");

%PART C
num_days = 200;
m_yearly = num_days*m_total; % total mass of water heated in 200 days
% Heat required is mass of water times Cp times change in temperature:
Q_yearly = m_yearly*Cp*(Tfinish-Tstart);
% Calculating how much CH4 would be required to generate this heat
% (assuming 90% efficiency):
CH4kg_saved = (Q_yearly/CH4_heating)/0.9;
% Molar mass of CH4 and CO2 required for these calculations:
M_CH4 = 16; % grams per mol
M_CO2 = 44; % grams per mol
% Number of moles is equal to mass divided by molar mass:
mol_CH4 = (CH4kg_saved*1000)/M_CH4; % moles, with mass converted to grams first
% Number of moles of CH4 and CO2 is the same,
% from coefficients of chemical reaction:
% CH4 + O2 = CO2 + 2H2O
mol_CO2 = mol_CH4;
% Mass of CO2 is equal to molar mass of CO2 times the number of moles:
CO2kg_saved = (M_CO2*mol_CO2)/1000;

disp(" ");
disp("-----PART C-----");
disp("Amount of CH4 that would be saved yearly:   " + CH4kg_saved + " kg");
disp("Amount of CO2 that would be prevented from being released yearly:   " + CO2kg_saved + " kg");
```

Task 5a (main code)

```
clc
close all

% Question (a): Energy lost due to shadow.
t0_old = 10; % starting time before shadow was introduced
t0_shadow = 11; % starting time after shadow was introduced
tfinal = 16; % same end time for both cases
% Call function yearlyEnergy for both before and after shadow scenarios:
yearlyEnergy_before_shadow = yearlyEnergy(t0_old, tfinal);
yearlyEnergy_after_shadow = yearlyEnergy(t0_shadow, tfinal);
EnergyLost = yearlyEnergy_before_shadow - yearlyEnergy_after_shadow;
disp("Total energy collection before the shadow from the new building: " + yearlyEnergy_before_shadow/10^9 + " GJ per year.");
disp("Total energy collection after the shadow from the new building: " + yearlyEnergy_after_shadow/10^9 + " GJ per year.");
disp("Energy collection is reduced by " + EnergyLost/10^9 + " GJ per year due to shadow from the new building.");

% Question (b): How much natural gas to compensate for the loss?
LHV_CH4 = 50050000; % J; lower heating value of natural gas
CH4_kg_90eff = (EnergyLost/LHV_CH4)/0.9; % Efficiency of 90%
CH4_kg_90eff_formatted = sprintf('%.2f', CH4_kg_90eff);
disp("Natural gas required per year to compensate: " + CH4_kg_90eff_formatted + " kg.");

% Question (c): Total yearly cost of natural gas to compensate for the shadow.

% Price of natural gas in dollars per thousand cubic feet as of Jul 2018.
% Source: www.eia.gov (price of natural gas for CA residents)
CH4_price = 12.36/1000; % Converted to dollars per cubic foot.

% Density of natural gas in kg/m3: 0.7 - 0.92.
% Source: https://www.engineeringtoolbox.com/gas-density-d_158.html
% For this problem, let us use the value from the middle of this range: 0.81 kg/m3.

density = 0.81; % kg/m3
volume_m3 = CH4_kg_90eff/density; % m3
volume_ft3 = volume_m3*35.3147; % ft3, assuming conversion factor 35.3147
CH4_total_cost = CH4_price*volume_ft3;
CH4_total_cost_formatted = sprintf('%.2f', CH4_total_cost);
disp("Yearly cost of additional natural gas: " + CH4_total_cost_formatted + " dollars.");
```

Task 5a (yearlyEnergy function)

```
function [yearly_energy] = yearlyEnergy(t0,tfinal)

% constants, inputs, collector's characteristics
A = 1310; % W/m2
B = 0.18;
lambda = 37.9; % latitude of Berkeley, CA
d = 120; % day of the year; Day 120 is Apr 30, 2018
tau_g = 0.89; % collector glazing transmissivity
alpha_c = 0.85; % collector absorptivity
Cp = 4186; % J/kg*K
h_conv_o = 7; % W/m2*K, outside air convective heat transfer coefficient
delta_g = 0.7*10^-2; % cm converted to m; glazing thickness
Kg = 1.3; % W/m*K; glazing thermal conductivity
h_conv_i = 3.1; % W/m2*K, inside convection coefficient (between absorber and glazing)
delta_ins = 6*10^-2; % cm converted to m; insulation thickness
Kins = 0.045; % W/m*K, insulation thermal conductivity
Ac = 3.25*3; % 3 solar collectors, each has area of 3.25 m^2

% Formulas for efficiency calculation
% Formulas for efficiency calculation
% Total conductance is Uloss times area or the collector:
conductance = ((1/(h_conv_o*Ac))+(delta_g/(Kg*Ac))+1/(h_conv_i*Ac))^-1+((1/(h_conv_o*Ac))+(delta_ins/Kins*Ac))^-1;
Uloss = conductance/Ac;

% Intermediate formulas
dec = 23.44*sind((360/365.25)*(d-80)); % declination
alpha = @(t) ((15*(t-12))); % hour angle (function of time of day)
zen_angle = @(t) rad2deg(acos(sind(lambda)*sind(dec) + cosd(lambda)*cosd(dec)*cosd(alpha(t)))); % zenith angle
sol_azimuth_tan = @(t) sind(alpha(t))/(sind(lambda)*cosd(alpha(t))-cosd(lambda)*tand(dec)); % tangent of solar azimuth
Int = @(t) A*exp((-B/sind((90)-zen_angle(t)))); % Intensity

% Integration inputs
%t0 = 10; % 10AM
%tfinal = 16; %4PM
h = 0.025; % time step size
num_steps = (tfinal - t0)/h; % number of steps given step size
Ti = 16; % water inlet temperature
% Ambient temperature varies from 9C to 16C during the time interval
T0 = 9;
Tfinal = 16;
% Using the number of steps to determine change in T for each step
T_step = (Tfinal - T0)/num_steps;

tot_energy = []; % array for Total Energy values
tot_energy_no_clouds = []; % array for Total Energy values (no clouds)
element=1; % first array index as 1

m_dot =  0.0304; % kg/s; optimal value determined from previous optimizations
surf_azimuth = 180;
surf_inc = 23;

% Calculate the daily total at two-week intervals, starting with d = 1 and
% finishing at the end of the year.
for d = 1:14:366

q_dot = []; % array for q-dot values
```

```
q_dot_no_clouds = []; % array for q-dot values (no clouds)
index=1; % first array index as 1
Ta = T0; % ambient temperature at the beginning of the process is set to T0
dT = []; % change in temperature of water at each time step
dT_sum = 0; % total change in temperature of water

  for t = t0:h:tfinal %from time t0 to tfinal, in 0.025 hr increments
  % Determine solar azimuth angle given alpha and tan of solar azimuth angle.
     if (alpha(t) > 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 360 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     elseif (alpha(t) < 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     end
     % Incident Direct Radiation (W/m2)
     Id = @(t) 0.54*Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth));
     %Id_no_clouds = @(t) Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth));

     % Heat removal factor:
     Fr = (1 - exp(-conductance/(m_dot*Cp)))/(conductance/(m_dot*Cp));
     efficiency =  @(t) Fr*((tau_g*alpha_c) - (Uloss/Id(t))*(Ti - Ta));
     % Output is Incident Direct Radiation adjusted for collector
     % efficiency and multiplied by collector area.
     q_dot(index) = Id(t)*efficiency(t)*Ac; % Units: J/s
     %q_dot_no_clouds(index) = Id_no_clouds(t)*efficiency(t)*Ac; % Units: J/s
     % Increment array indexes and change in ambient temperature:
     index = index + 1;
     Ta = Ta + T_step;
  end
  % Use trapz MATLAB function, using time_steps as x and outputs(t) as y
  time_steps = [t0:h:tfinal]*3600; % Convert time steps from hours to
  % seconds for integration purposes (since W = J/s)
  tot_energy (element) = trapz(time_steps,q_dot);
  %tot_energy_no_clouds (element) = trapz(time_steps,q_dot_no_clouds);
  day_num (element) = d;
  element = element + 1;
end

% Integrate over the entire year to find the amount of energy collected for
% the year
yearly_energy = trapz(day_num,tot_energy);
%yearly_energy_no_clouds = trapz(day_num,tot_energy_no_clouds);
%disp("Yearly energy adjusted for cloudy conditions: " + yearly_energy/10^6 + " MJ per year");
%disp("Yearly energy without cloudy conditions: " + yearly_energy_no_clouds/10^6 + " MJ per year");
```

Task 5b (main code)

```
clc
close all

% Question (a): Total heat collected with and without the motorized frame.
t0_old = 10; % starting time before shadow was introduced
t0_shadow = 11; % starting time after shadow was introduced
tfinal = 16; % same end time for both cases
% Call functions yearlyEnergy and yearlyEnergy_motorFrame:
before_motorFrame = yearlyEnergy(t0_shadow, tfinal);
after_motorFrame = yearlyEnergy_motorFrame(t0_shadow, tfinal);
disp("Total energy collected without motorized frame: " + before_motorFrame/10^9 + " GJ per year.");
disp("Total energy collected with motorized frame: " + after_motorFrame/10^9 + " GJ per year.");

% Question (b): Percentage increase in yearly collected energy over the
% original fixed frame design.
energyIncrease = after_motorFrame-before_motorFrame;
percentIncrease = energyIncrease/before_motorFrame;
disp("Percentage increase in yearly collected energy: " + percentIncrease*100 + " %.");

% Question (c): Compare extra energy from motorized frame to energy loss
% due to shadow over the full year.
% Call function yearlyEnergy for both before and after shadow scenarios:
yearlyEnergy_before_shadow = yearlyEnergy(t0_old, tfinal);
yearlyEnergy_after_shadow = yearlyEnergy(t0_shadow, tfinal);
EnergyLost = yearlyEnergy_before_shadow - yearlyEnergy_after_shadow;
disp("Energy collection is reduced by " + EnergyLost/10^9 + " GJ per year due to shadow from the new building.");
disp("Energy collection is increased by " + energyIncrease/10^9 + " GJ per year due to motorized frame design.");
disp("Motorized frame design allows to offset " + ((EnergyLost-energyIncrease)/EnergyLost)*100 + " % of the energy loss due to shadow.");
```

Task 5b (yearlyEnergy_motorFrame function)

```
function [yearly_energy] = yearlyEnergy_motorFrame(t0,tfinal)

% constants, inputs, collector's characteristics
A = 1310; % W/m2
B = 0.18;
lambda = 37.9; % latitude of Berkeley, CA
d = 120; % day of the year; Day 120 is Apr 30, 2018
tau_g = 0.89; % collector glazing transmissivity
alpha_c = 0.85; % collector absorptivity
Cp = 4186; % J/kg*K
h_conv_o = 7; % W/m2*K, outside air convective heat transfer coefficient
delta_g = 0.7*10^-2; % cm converted to m; glazing thickness
Kg = 1.3; % W/m*K; glazing thermal conductivity
h_conv_i = 3.1; % W/m2*K, inside convection coefficient (between absorber and glazing)
delta_ins = 6*10^-2; % cm converted to m; insulation thickness
Kins = 0.045; % W/m*K, insulation thermal conductivity
Ac = 3.25*3; % 3 solar collectors, each has area of 3.25 m^2

% Formulas for efficiency calculation
% Formulas for efficiency calculation
% Total conductance is Uloss times area or the collector:
conductance = ((1/(h_conv_o*Ac))+(delta_g/(Kg*Ac))+1/(h_conv_i*Ac))^-1+((1/(h_conv_o*Ac))+(delta_ins/Kins*Ac))^-1;
Uloss = conductance/Ac;

% Intermediate formulas
dec = 23.44*sind((360/365.25)*(d-80)); % declination
alpha = @(t) ((15*(t-12))); % hour angle (function of time of day)
zen_angle = @(t) rad2deg(acos(sind(lambda)*sind(dec) + cosd(lambda)*cosd(dec)*cosd(alpha(t)))); % zenith angle
sol_azimuth_tan = @(t) sind(alpha(t))/(sind(lambda)*cosd(alpha(t))-cosd(lambda)*tand(dec)); % tangent of solar azimuth
Int = @(t) A*exp((-B/sind((90)-zen_angle(t)))); % Intensity

% Integration inputs
%t0 = 10; % 10AM
%tfinal = 16; %4PM
h = 0.025; % time step size
num_steps = (tfinal - t0)/h; % number of steps given step size
Ti = 16; % water inlet temperature
% Ambient temperature varies from 9C to 16C during the time interval
T0 = 9;
Tfinal = 16;
% Using the number of steps to determine change in T for each step
T_step = (Tfinal - T0)/num_steps;

tot_energy = []; % array for Total Energy values
tot_energy_no_clouds = []; % array for Total Energy values (no clouds)
element=1; % first array index as 1

m_dot =  0.0304; % kg/s; optimal value determined from previous optimizations
surf_inc = 23;

% Calculate the daily total at two-week intervals, starting with d = 1 and
% finishing at the end of the year.
for d = 1:14:366

q_dot = []; % array for q-dot values
q_dot_no_clouds = []; % array for q-dot values (no clouds)
```

```
index=1; % first array index as 1
Ta = T0; % ambient temperature at the beginning of the process is set to T0
dT = []; % change in temperature of water at each time step
dT_sum = 0; % total change in temperature of water

  for t = t0:h:tfinal %from time t0 to tfinal, in 0.025 hr increments
  % Determine solar azimuth angle given alpha and tan of solar azimuth angle.
     if (alpha(t) > 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 360 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     elseif (alpha(t) < 0)
        if(sol_azimuth_tan(t) > 0)
           sol_azimuth = @(t) rad2deg(atan(sol_azimuth_tan(t)));
        elseif (sol_azimuth_tan(t) < 0)
           sol_azimuth = @(t) 180 + rad2deg(atan(sol_azimuth_tan(t)));
        end
     end
     % Motorized frame collector design: Set azimuth angle to match solar azimuth
     surf_azimuth = sol_azimuth;
     % Incident Direct Radiation (W/m2)
     Id = @(t) 0.54*Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth(t)));
     %Id_no_clouds = @(t) Int(t)*(cosd(zen_angle(t))*cosd(surf_inc)+sind(surf_inc)*sind(zen_angle(t))*cosd(sol_azimuth(t)-surf_azimuth));

     % Heat removal factor:
     Fr = (1 - exp(-conductance/(m_dot*Cp)))/(conductance/(m_dot*Cp));
     efficiency =  @(t) Fr*((tau_g*alpha_c) - (Uloss/Id(t))*(Ti - Ta));
     % Output is Incident Direct Radiation adjusted for collector
     % efficiency and multiplied by collector area.
     q_dot(index) = Id(t)*efficiency(t)*Ac; % Units: J/s
     %q_dot_no_clouds(index) = Id_no_clouds(t)*efficiency(t)*Ac; % Units: J/s
     % Increment array indexes and change in ambient temperature:
     index = index + 1;
     Ta = Ta + T_step;
  end
  % Use trapz MATLAB function, using time_steps as x and outputs(t) as y
  time_steps = [t0:h:tfinal]*3600; % Convert time steps from hours to
  % seconds for integration purposes (since W = J/s)
  tot_energy (element) = trapz(time_steps,q_dot);
  %tot_energy_no_clouds (element) = trapz(time_steps,q_dot_no_clouds);
  day_num (element) = d;
  element = element + 1;
end

% Integrate over the entire year to find the amount of energy collected for
% the year
yearly_energy = trapz(day_num,tot_energy);
%yearly_energy_no_clouds = trapz(day_num,tot_energy_no_clouds);
%disp("Yearly energy adjusted for cloudy conditions: " + yearly_energy/10^6 + " MJ per year");
%disp("Yearly energy without cloudy conditions: " + yearly_energy_no_clouds/10^6 + " MJ per year");
```