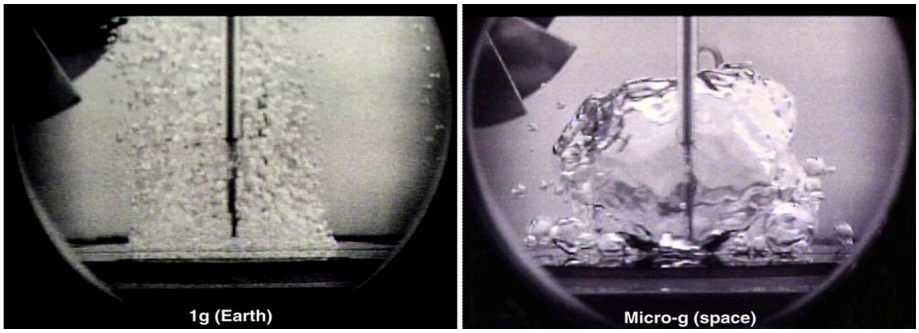
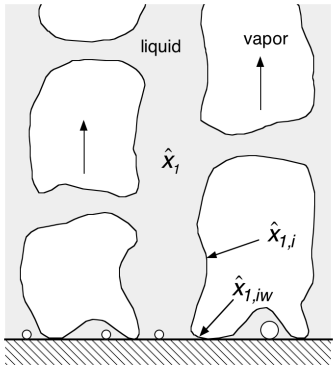
Project 1. See updates in red text. **Due February 16, 2021**

You may team up with a partner for this project. Do not share information or results with other groups.

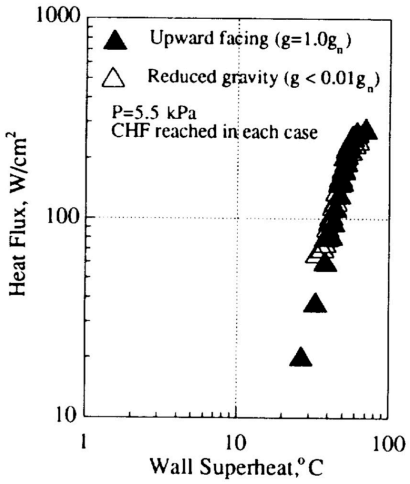
Background Information

Boiling heat transfer can be a useful, efficient heat transfer process in two-phase loops used for thermal management systems in terrestrial and aerospace applications. It is centrally important in, for example, Rankine cycle boiling and air-conditioning evaporators. In terrestrial applications, gravity buoyancy is a important mechanism that moves vapor away from the surface, allowing liquid to replace it to sustain the process. In microgravity or zero gravity, this mechanism is eliminated, and vapor tends to accumulate near the surface. If the surface is partially or fully blanketed with a vapor layer (transition or film boiling) the heat transfer rate can become so low that the system performance may be unacceptably low and/or the accumulation of heat may raise the wall temperature to a level that damages equipment. Boiling vaporization of cryogenic fuels and liquid oxygen is also a concern during transfer into and out of storage tanks during space missions. Note that high g levels can be encountered during acceleration of a vehicle, and very low g levels result during travel in deep space or in orbit.





(a)



(b)

Figure 1. (a) Concentration differences due to preferential vaporization of 2-propanol at the interface during nucleate boiling of water/2-propanol mixtures. (c) Nucleate boiling data for a 2-propanol/water mixture at 5.5 kPa pressure at earth normal gravity and reduced gravity (1.2 cm diameter, upward facing copper heated surface, 0.015 mole fraction of 2-propanol in water).

Figure 2. NASA parabolic aircraft microgravity boiling tests.

Boiling in high or low gravity conditions is important to space power, propulsion and thermal management systems. NASA has test facilities in the International Space Station to study boiling processes. They have also been studied boiling in parabolic flight aircraft, rockets, and drop towers to achieve reduced.

The nucleate boiling data used here was obtained aboard the "vomit comet" NASA aircraft (see photos in Fig. 2) for boiling of a water/2-propanol mixture. The results of earlier studies indicated that using mixtures of this type can substantially enhance nucleate boiling heat transfer and the critical heat flux. For certain binary fluid mixtures, there is clear evidence that surface tension gradients resulting from concentration differences act to enhance the fluid motion towards the heated surface. As depicted in Fig. 1.a, because the lower vapor pressure alcohol preferentially evaporates at the interface, the lower 2-propanol concentration at the interface near the wall reduces the interfacial concentration of the alcohol there ($\hat{x}_{1,iw} < \hat{x}_{1,i}$) virtually to zero. This, in turn, lowers the surface tension close to the contact line at the wall to essentially that for pure water. The resulting gradient in surface tension along the interface produces a Marangoni driven flow of liquid towards the contact line at the wall. In experiments at $1g_{en}$, the Marangoni effect supplements the effects of buoyancy. However, for binary mixture nucleate boiling experiments under $0.01g_{en}$ reduced gravity conditions, the buoyancy effect is essentially removed, and the ability of Marangoni forces to induce liquid motion towards the surface becomes the dominant liquid delivery mechanism at the surface.

The data from these studies definitively indicate that variation of gravitational acceleration affects the boiling process in water/2-propanol mixtures much differently than in pure water boiling. The objective here is to apply machine learning tools to the data to explore the interplay of gravity and Marangoni effects during boiling of a water/2-propanol mixture for a range of gravity acceleration conditions.

The experiments corresponded to the following parameter values:

Circular copper heated surface diameter = 12.5 mm
Moderately rough and wetted heater surface
Pressure at heated surface = 5.5 kPa to 9.5 kPa
Gravitational acceleration = $0.01 g_{en}$ to $2 g_{en}$
Wall superheat = 20 to 80 °C
2-propanol bulk liquid mole fraction ($\hat{x}_{1,b}$) = 0.0 to 0.025
Fractional change in surface tension due to interface evaporation ($\gamma = (\sigma_w - \sigma_{mix}) / \sigma_{mix}$) = 0.0 to 1.74

The resulting data can be envisioned as an array in which the rows are associated data for one test data point: $[q'', T_w - T_{sat}, g, \gamma, P]$. Consistent with this, in the example code provided, the raw data is organized in an array of the form:

```
[[44.1, 32.5, 0.098, 1.79, 5.5], ]
 [47.4, 33.2, 0.098, 1.79, 5.5]
 [49.4, 34.2, 0.098, 1.79, 5.5]
                ⋮
 [59.2, 34.8, 0.098, 1.79, 5.5]
 [67.8, 36.3, 0.098, 1.79, 5.5]
```

Nomenclature

$c_{p,l}$	liquid specific heat
g	gravitational acceleration, m/s^2
g_{en}	earth-normal gravitational acceleration, 9.8 m/s^2
h_{lv}	latent heat of vaporization, kJ/kg
k_l	liquid thermal conductivity, W/mK
L_h	heated surface characteristic size, m
q''	heat flux, W/cm^2
q''_{en}	heat flux under earth-normal gravity
Pr_l	liquid Prandtl number
T_{sat}	saturation temperature, °C
$T_w - T_{sat}$	wall superheat, °C
$\hat{x}_{1,b}$	bulk mole fraction of 2-propanol in water/2-propanol solution
$\hat{x}_{1,i}$	interface mole fraction of 2-propanol in water/2-propanol solution
α_l	liquid thermal diffusivity
γ	surface tension parameter $= (\sigma_w - \sigma_{mix}) / \sigma_{mix}$
μ_l	liquid viscosity
ρ_l	liquid density, kg/m^3
ρ_v	vapor density, kg/m^3
σ	surface tension, N/m

Task 1.
The purposes of Task 1 are to install the first portion of the genetic algorithm program **CodeP1.1** and to become familiar with package installation and data prep for the program.

Note that in the program, the data is stored explicitly in an array by using commands that first create an array with just one row of data elements

```
ydata =     [[44.1, 32.5, 0.098, 1.79, 5.5]]
```

And then uses the append function to add more rows:

```
ydata.append([47.4, 33.2, 0.098, 1.79, 5.5])
ydata.append([49.4, 34.2, 0.098, 1.79, 5.5])
```

This creates an array of the form

```
[[44.1, 32.5, 0.098, 1.79, 5.5],
[47.4, 33.2, 0.098, 1.79, 5.5],
[49.4, 34.2, 0.098, 1.79, 5.5],
        ⋮
[207.5, 50.9, 0.098, 1.71, 5.5]]
```

Note that for this format, for `ydata[i][j]:` *j* is column, *i* is row downward – both xstart at 0. So, you access an element as `ydata[row][column].` This is an array that is essentially a list of lists.

(a) Install this code into a first cell of a new Anaconda notebook. Run the code, which will simply print the array and a single array element in the output region below the cell. Inspect the output to confirm it matches the array in the Appendix on the last page of this writeup. Note that in this code I have commented-out portions of the data with pressures other than 5.5 kPa. More on that in the next Task.

(b) Using the data in the first two groups in the Appendix listing (for g = 0.098 and 9.8 m/s^2) make a log-log plot of heat flux versus wall superheat for these two gravity levels to get a sense of how strongly heat flux varies with gravity and superheat. You can do this using a separate Python program (recommended) or you can use another platform such as Excel or Matlab if you prefer. A log-log plot is done in the second python code file so you can see an example there.

Task 2.
Install **CodeP1.2** - either append it to the end of **CodeP1.1** to make one program, or copy it in the next cell in the Anaconda notebook so that running both cells in sequence will execute the genetic algorithm for simple equation model use din this Task.

The objective here is to use machine learning tools to determine how the heat flux varies with the other parameters in the data. The data reflect the combined variation with these parameters, and the objective of the machine learning analysis is to determine what the individual dependencies are. Although this is not a simple task, genetic algorithms thrive when epistasis gene interaction (suppression of one gene's effect by another) is medium to high. Note, for example, that for pure fluid, in most nucleate boiling processes, heat flux is well approximated as a power law function of superheat $q'' \sim (T_w - T_{sat})^n$ where n is about 3. Boiling heat transfer data also suggest that for pure fluids, heat flux is roughly a power law function of gravitational acceleration

$q'' \sim g^n$, where n is in the range $0.01 < n < 0.1$ for $g < 0.1\, g_{en}$, and n is on the order of 0.25 for $g/g_{en} > 0.1$. These trends indicate that heat flux approaches zero as gravitational acceleration approaches zero. For the water with small amounts of 2-propanol mixture considered here, Marangoni (variable surface tension) effects result in a finite heat flux even when $g \to 0$. Hence, the simple power law variation of q'' with g is not expected to apply for water/2-propanol mixtures. In addition, the correlations discussed above imply that the nucleate boiling heat flux exhibits a dependence on fluid properties. Since, for a given fluid, these are taken to be saturation liquid and/or vapor properties, which are a function of saturation pressure for a given fluid, the dependence on these properties could be modeled as a power law dependence on pressure.

The combined **CodeP1.1** and **CodeP1.2** program is set up to only access the black-text data in the Appendix which has a fix surface tension parameter and pressure. The heat flux tor these data are only a function of superheat and gravitational acceleration. Given these observations above about the dependence of heat flux on superheat and gravity, we therefore adopt the following postulated relation for the dependence of heat flux on superheat and gravitational acceleration:

$$q'' = n_1 (T_w - T_{sat})^{n_2} g^{n_3} \qquad (1)$$

To apply the genetic algorithm methodology describe above, a suitable error function must be defined for a given experimental data point and set of constants n_i. This could be defined to be just the difference between the experimentally measured heat flux and the right side of Eq. (1) evaluated at the measured $T_w - T_{sat}$, g and P. However, here we took the natural log of both sides to convert Eq. (1) to:

$$\ln q'' = \ln(n_1) + n_2 \ln(T_w - T_{sat}) + n_3 \ln(g) \qquad (2)$$

Working with Eq.(2) provides an equation that is more linear in the coefficients, and therefore may result in a more well behaved solution process. Based on Eq. (2), an error function $f_{err,i}$ for a given data point i can be defined as

$$f_{err,i} = -\ln q''_{data,i} + \ln(n_1) + n_2 \ln(T_w - T_{sat})_{data,i} + n_3 \ln(g_{data,i}) \qquad (3)$$

In the program provided, the number of organisms (solutions) N_S is taken to be equal to the number of data points N_D, so for each generation, each solution can be compared to a different data point and all the data is compared in each generation. The order of the solutions in the array that holds the solution constants is constantly changing due to mating and selection, so the pairing is random.

In the program provided, the number of organisms (solutions) N_S is taken to be equal to the number of data points N_D, so for each generation, each solution can be compared to a different data point and all the data is compared in each generation. The order of the solutions in the array that holds the solution constants is constantly changing due to mating and selection, so the pairing is random.

If there are N_D data points in the data set to be analyzed, the total error function F_{err} is the sum of the fractional absolute value of the error for each data point in the population, N_D:

$$F_{err} = \sum_{i=1}^{N_D} \left| f_{err,i} \right| \Big/ \left| \ln q''_{data,i} \right|$$
$$= \sum_{i=1}^{N_D} \Big| -\ln q''_{data,i} + \ln(n_1) + n_2 \ln(T_w - T_{sat})_{data,i} \qquad (4)$$
$$+ n_3 \ln(g_{data,i}) \Big| \Big/ \left| \ln q''_{data,i} \right|$$

Note that RMS error could be used instead of absolute error. The program is set up to use absolute error.

The assembled code will have a different value of $\left[F_{err}\right]_{mean} / N_D$ for each generation. The code seeks the minimum value achieved during the simulation. The constants n_1, n_2, n_3 for that value can be interpreted as providing a best fit. Once the code is fully assembled, run it to see if it successfully achieves a minimum error $\left[\left[F_{err}\right]_{mean} / N_D\right]_{\min}$ less than 0.04. Try running the code for different initial guesses to see how sensitive the output is to initial guesses. Note that the farther the initial guesses are from values that minimize the error, the more generations may be required to achieve low error. In that case, you may want to increase the number of generation `NGEN` as well.

Task 3

Make of copy of your combined **CodeP1.1&CodeP1.2** program that you can modify to train a **five constant model** that includes variation of pressure and the surface tension parameter γ. Specifically, this new program will be designed to find the set of constants n_1 through n_5 in the performance equation

$$q'' = n_1 (T_w - T_{sat})^{n_2} (g + n_4 g_{en} \gamma)^{n_3} P^{n_5} \qquad (5)$$

that best fits an expanded data set that includes variable γ and pressure. Taking the natural log of both sides to convert Eq. (5) to:

$$\ln q'' = \ln(n_1) + n_2 \ln(T_w - T_{sat}) + n_3 \ln(g + n_4 g_{en} \gamma) + n_5 \ln P \qquad (6)$$

Based on this equation, an error function $f_{err,i}$ for a given data point i can be defined as

$$f_{err,i} = -\ln q''_{data,i} + \ln(n_1) + n_2 \ln(T_w - T_{sat})_{data,i} + n_3 \ln(g_{data,i} + n_4 g_{en} \gamma_{data,i}) + n_5 \ln P_{data,i} \qquad (7)$$

If there are N_D data point in the data set to be analyzed, the total error function F_{err} is the sum of the fractional absolute value of the error for each data point in the population, N_D:

$$F_{err} = \sum_{i=1}^{N_D} \left| f_{err,i} \right| \Big/ \left| \ln q''_{data,i} \right|$$
$$= \sum_{i=1}^{N_D} \Big| -\ln q''_{data,i} + \ln(n_1) + n_2 \ln(T_w - T_{sat})_{data,i} \qquad (8)$$
$$+ n_3 \ln(g_{data,i} + n_4 g_{en} \gamma_{data,i}) + n_5 \ln P_{data,i} \Big| \Big/ \left| \ln q''_{data,i} \right|$$

To set up this new analysis, your modifications to the three constant **CodeP1.1&CodeP1.2** program should include the following:

(i) uncomment the red text lines in the ydata array definition to add that additional data

(ii) change ND and NS to 77

(iii) modify the following lines of code to compute the appropriate error and heat flux quantities for the five constant model:

```
Ferr[i] = n[i][0]*lydata[i][0] + math.log(n[i][1]) + n[i][2]*lydata[i][1]
Ferr[i] = Ferr[i] + n[i][3]*math.log( ydata[i][2] )

for i in range(ND):
      Ferravgn[i] = -1.*lydata[i][0] + math.log(n1avg[k]) + n2avg[k]*lydata[i][1]
      Ferravgn[i] = Ferravgn[i] + n3avg[k]*math.log( ydata[i][2] )

for i in range(ND):
      qpppred[i] = n1min*(ydata[i][1]**n2min) * ((ydata[i][2])**n3min)
```

(iv) To include constants n_4 and n_5 in the mating, uncomment the following lines of code:

```
if (numpy.random.rand() < 0.5):
    ntemp[nkeep+j+1][4] = n[nmate1][4]*(1.+0.09*2.*(0.5-numpy.random.rand())) # property 4,mutation added
else:
    ntemp[nkeep+j+1][4] = n[nmate2][4]*(1.+0.09*2.*(0.5-numpy.random.rand()))

if (numpy.random.rand() < 0.5):
    ntemp[nkeep+j+1][5] = n[nmate1][5]*(1.+0.09*2.*(0.5-numpy.random.rand())) # property 5, mutation added
else:
    ntemp[nkeep+j+1][5] = n[nmate2][5]*(1.+0.09*2.*(0.5-numpy.random.rand()))
```

(v) For the final output of constants, modify the following lines of code to output all five constants with the corresponding error parameter:

```
print('ENDING: pop.avg n1-n3,aFerrmean:', n1avg[k], n2avg[k], n3avg[k], aFerrmeanavgn[k])
print('MINUMUM:  avg n1-n3,aFerrmeanMin:', n1min, n2min, n3min, aFerrmeanavgnMin)
print('TIME AVG:    avg n1-n3,aFerrmean:', n1ta, n2ta, n3ta, aFerrta)
```

(vi) To alter the first plot for the five constant model, delete or comment-out the first line of code below and uncomment the three lines following it:

```
plt.legend(['aFerrmeanavgn', 'n1 avg', 'n2 avg', 'n3 avg'], loc='lower left')
#plt.plot(gen, n4avg)
#plt.plot(gen, n5avg)
#plt.legend([' aFerrmeanavgn','n1 avg','n2 avg','n3 avg','n4 avg','n5 avg'],loc='upper right')
```

(vii) Change the initial guesses to:

```
n0i =  -1.0
n1i = 0.000476
n2i = 3.028
n3i = 0.2249
n4i = 1.054
n5i = 0.217
```

Run this new model to determine n1 through n5 to best fit the data. Run for a range of initial guesses to be sure you have a best fit (`aFerrmeanavgnMin` less than 0.03). Summarize the constants n_1 through n_5 for your best fit to the data in a table. Use the resulting curve-fit equation to create a log-log surface plot of $q''/(T_w - T_{sat})^{n_2}$ versus g and γ for $1.0 < g < 20$ m/s^2 and $0.001 < \gamma < 2$ at a pressure of 10 kPa.

Task 4.
Take the data used in Task 3 and convert it to the non dimensional parameters suggested by the well-known Rohsenow correlation for nucleate boiling:

$$Q_s = 10\frac{q''}{\mu_l h_{lv}}\sqrt{\frac{\sigma}{g_{en}(\rho_l - \rho_v)}} \qquad (9)$$

$$\mathrm{Ja}_s = 100\frac{c_p(T_w - T_{sat})}{h_{lv}} \qquad (10)$$

$$\mathrm{Pr}_l \qquad (11)$$

$$g / g_{en} \qquad (12)$$

$$\gamma = \frac{\sigma_w - \sigma_{mix}}{\sigma_{mix}} \qquad (13)$$

In the ydata array, use these relations to compute the new dimensionless variable for each data point, and convert it from $[q'', T_w - T_{sat}, g, \gamma, P]$ to $[Q_s, \mathrm{Ja}_s, g/g_{en}, \gamma, \mathrm{Pr}_l]$. To compute the new dimensionless variables from the raw data, use the properties in the table below for pressures of 5.5, 7 and 9.5 kPa.

Low pressure water properties

	P = 5.5 kPa	P = 7.0 kPa	P = 9.5 kPa
T_{sat} (°C)	34.9	38.0	45.0
c_{pl} (kJ/kg°C)	4.18	4.18	4.18
h_{lv} (kJ/kg°C)	2418	2406	2394
μ_l (Ns/m^2)	7.19x10^{-6}	6.53x10^{-6}	5.96x10^{-6}
Pr_l	4.83	4.54	3.91
ρ_l (kg/m^3)	994	993	990
ρ_v (kg/m^3)	0.0397	0.0476	0.182
σ (N/m)	0.0706	0.0692	0.0688

Interpretation of the dimensionless group definitions, and trends in the data, suggest the following postulated relation for the interrelationship among these parameters:

$$Q_s = n_1 \mathrm{Ja}_s^{n_2} \left(\frac{g}{g_{en}} + n_3 \gamma \right)^{n_4} \mathrm{Pr}_l^{-n_5} \qquad (14)$$

Take the natural log of both sides and define an error function similar to that used in the raw data analysis. Note this should be different in form because the postulated relation is different from the raw data analysis. Show all the steps in deriving the relation for $\ln Q_s$ and its use to derive appropriate relations for $f_{err,i}$ and F_{err}. Documents these in your summary report. Create a printout table of the dimensionless data array similar to that provided for the raw data in the Appendix of this writeup. Include that table in an appendix of your report.

Task 5.
Make a copy of the five-constant raw data code program developed in Task 3, and make appropriate changes to it so it will use the genetic algorithm to determine the set of constants n_1 through n_5 in the performance equation (14) that provide a best fit to the dimensionless data set developed in Task 4. Summarize the constants n_1 through n_5 for your best fit to the data in a table.

Rearrange Eq. (14) and solve it for $Q_s \, \mathrm{Pr}_l^{n_5} / \mathrm{Ja}_s^{n_2}$ as a function of g / g_{en} and γ. Use the resulting relation to construct as surface plot of $Q_s \, \mathrm{Pr}_l^{n_5} / \mathrm{Ja}_s^{n_2}$ for $0.01 \le g / g_{en} \le 2$ and $0.001 \le \gamma \le 2$ with both the g / g_{en} axis and the γ axis logarithmic.

Task 6.
Assemble a report that summarizes the results of your work on Tasks 1-5. Include in your report:

- A statement of how the work in the Tasks was divided between the members of the team.
- In Tasks where you are required to create a new program by modifying an earlier one, summarize the modifications you made to the previous version of the code.
- Be sure to document and changes you made to the basic code structure in the example code provided.
- Be sure to include plots and tables of results specified in the Task descriptions.
- In the report, present a summary of your conclusions regarding how initial guesses of model constants affect the stability of the algorithm and the speed of convergence to a satisfactory fit. Summarize results in tables or plots as appropriate.

- In the report, also provide a discussion of the advantages and disadvantages of raw data analysis versus dimensionless data analysis for this heat transport process.

- Include a copy of each of your programs in an Appendix at the end of the report.

Deliverables:

A pdf of your summary report is due by **2/16/21 @ 5:00 PM.**

Grade will be based on:

(1) thoroughness of documentation of your analysis and program development
(2) accuracy of best fit constants
(3) thoroughness of the assessments of the program performance
(4) documentation of results, including plots, tables, etc. should be professional quality

Appendix

ydata: $[q", T_w - T_{sat}, g, \gamma, P]$

heat flux $q"$ ~ W/m^2, superheat $T_w - T_{sat}$ ~°C, gravitational acceleration g~ m/s^2, surface tension parameter γ ~ dimensionless), pressure P ~ kPa

```
[[44.1, 32.5, 0.098, 1.79, 5.5],
[47.4, 33.2, 0.098, 1.79, 5.5],
[49.4, 34.2, 0.098, 1.79, 5.5],
[59.2, 34.8, 0.098, 1.79, 5.5],
[67.8, 36.3, 0.098, 1.79, 5.5],
[73.6, 37.3, 0.098, 1.79, 5.5],
[76.3, 37.8, 0.098, 1.79, 5.5],
[85.3, 39.2, 0.098, 1.79, 5.5],
[96.5, 39.3, 0.098, 1.79, 5.5],
[111.0, 42.3, 0.098, 1.79, 5.5],
[124.0, 43.5, 0.098, 1.79, 5.5],
[136.2, 45.4, 0.098, 1.79, 5.5],
[143.5, 46.7, 0.098, 1.79, 5.5],
[154.6, 47.9, 0.098, 1.79, 5.5],
[163.1, 48.6, 0.098, 1.79, 5.5],
[172.8, 50.9, 0.098, 1.79, 5.5],
[184.2, 51.7, 0.098, 1.79, 5.5],
[203.7, 56.4, 0.098, 1.79, 5.5],
[36.7, 30.2, 9.8, 1.79, 5.5],
[55.1, 34.1, 9.8, 1.79, 5.5],
[67.5, 35.3, 9.8, 1.79, 5.5],
[78.0, 37.8, 9.8, 1.79, 5.5],
[92.0, 38.1, 9.8, 1.79, 5.5],
[120.0, 44.1, 9.8, 1.79, 5.5],
[134.3, 46.9, 9.8, 1.79, 5.5],
[150.3, 48.5, 9.8, 1.79, 5.5],
[167.0, 49.2, 9.8, 1.79, 5.5],
[184.0, 52.7, 9.8, 1.79, 5.5],
[196.5, 53.1, 9.8, 1.79, 5.5],
[42.4, 28.0, 19.6, 1.79, 9.5],
[48.7, 29.3, 19.6, 1.79, 9.5],
[54.5, 29.6, 19.6, 1.79, 9.5],
[62.1, 28.5, 19.6, 1.79, 9.5],
[70.8, 30.5, 19.6, 1.79, 9.5],
[73.7, 30.3, 19.6, 1.79, 9.5],
[81.8, 30.6, 19.6, 1.79, 9.5],
[91.9, 34.5, 19.6, 1.79, 9.5],
[103.9, 34.5, 19.6, 1.79, 9.5],
[119.1, 35.4, 19.6, 1.79, 9.5],
[133.7, 36.8, 19.6, 1.79, 9.5],
[139.9, 38.1, 19.6, 1.79, 9.5],
[148.3, 39.1, 19.6, 1.79, 9.5],
[157.0, 40.0, 19.6, 1.79, 9.5],
[169.1, 42.2, 19.6, 1.79, 9.5],
[179.2, 43.2, 19.6, 1.79, 9.5],
[205.0, 46.0, 19.6, 1.79, 9.5],
[42.4, 29.7, 19.6, 1.79, 5.5],
[48.7, 31.0, 19.6, 1.79, 5.5],
[54.5, 31.2, 19.6, 1.79, 5.5],
[70.8, 32.4, 19.6, 1.79, 5.5],
[73.7, 31.4, 19.6, 1.79, 5.5],
[81.8, 32.5, 19.6, 1.79, 5.5],
[91.9, 36.3, 19.6, 1.79, 5.5],
[103.9, 36.3, 19.6, 1.79, 5.5],
[119.1, 37.2, 19.6, 1.79, 5.5],
[133.7, 38.4, 19.6, 1.79, 5.5],
[139.9, 39.7, 19.6, 1.79, 5.5],
[148.3, 40.9, 19.6, 1.79, 5.5],
[157.0, 41.6, 19.6, 1.79, 5.5],
[169.1, 43.9, 19.6, 1.79, 5.5],
[179.2, 45.0, 19.6, 1.79, 5.5],
[205.0, 47.9, 19.6, 1.79, 5.5],
[77.0, 41.5, 9.8, 0.0, 7.0],
[71.0, 40.5, 9.8, 0.0, 7.0],
[66.0, 39.5, 9.8, 0.0, 7.0],
[62.0, 38.5, 9.8, 0.0, 7.0],
[42.0, 34.0, 9.8, 0.0, 7.0],
[60.0, 37.5, 9.8, 0.0, 7.0],
[53.0, 37.0, 9.8, 0.0, 7.0],
[71.7, 36.4, 0.098, 1.71, 5.5],
[81.5, 38.5, 0.098, 1.71, 5.5],
[90.7, 39.5, 0.098, 1.71, 5.5],
[103.3, 41.6, 0.098, 1.71, 5.5],
[117.0, 43.1, 0.098, 1.71, 5.5],
[138.6, 45.4, 0.098, 1.71, 5.5],
[161.7, 47.9, 0.098, 1.71, 5.5],
[207.5, 50.9, 0.098, 1.71, 5.5]]
```