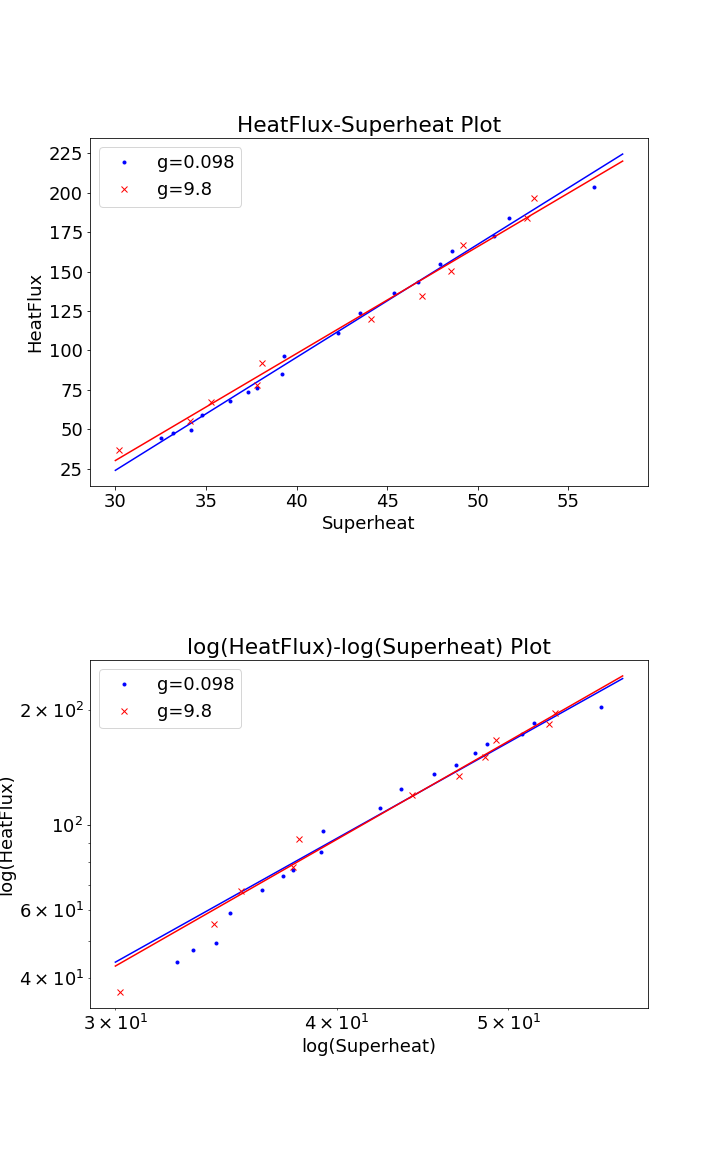
**Report for Project 1**

**Task 1**



The figure above includes a HeatFlux-Superheat plot and a HeatFlux-Superheat log-log plot. Blue dots and lines represent the data and regression under the condition of micro gravity (0.098), while the red ones represent the condition of regular gravity (9.8).

As shown in the figure, heat flux is positively correlated with superheat, and the relationship varied a little in different gravity conditions.

It seems that there could be a linear relationship between heat flux and superheat, so linear regression is applied to the data. (As for the log-log plot, linear relationship should be transferred into exponential relationship according to simple math calculations.) The result is shown in the table below. Mean Square Error (MSE) is used to decide whether the regression is good or not.

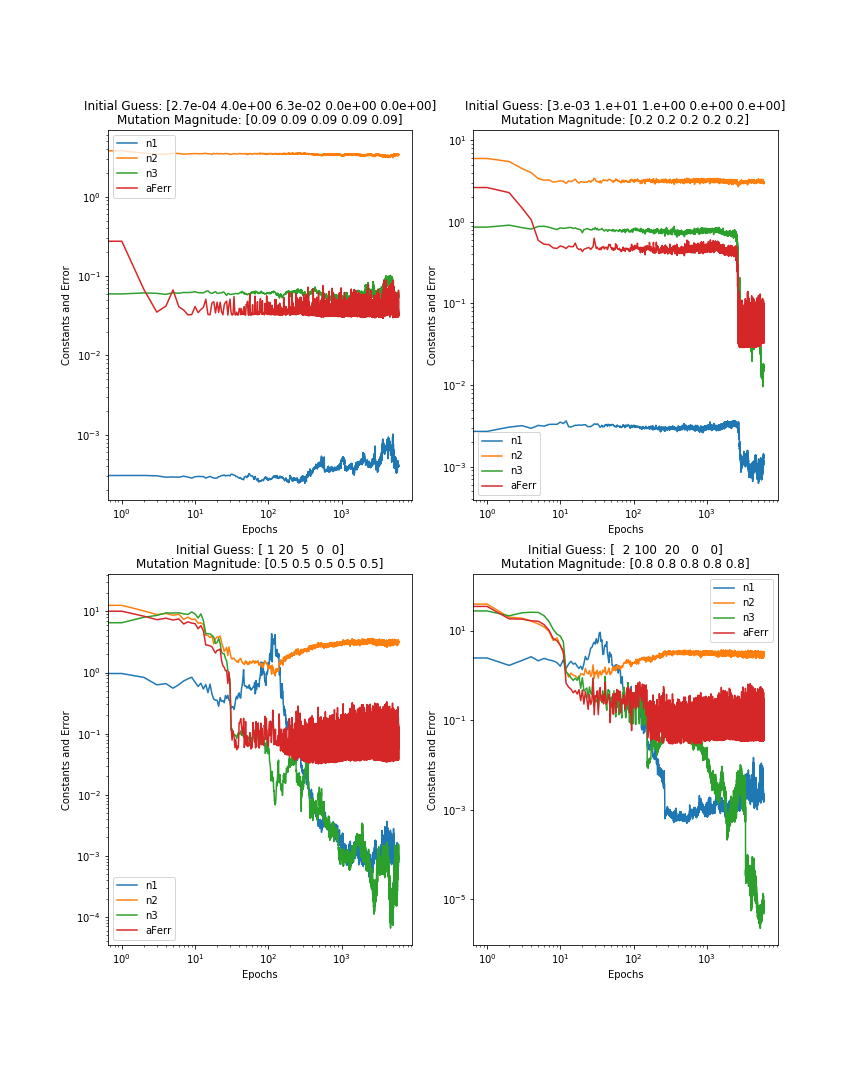
|  |  |  |
| --- | --- | --- |
| g | Lin. Regression | Exp. Regression |
| 0.098 | y = 7.167503 \* x + -191.125966  MSE = 15.158677 | y = 0.006760 \* x ^(2.581364)  MSE = 72.352377 |
| 9.8 | y = 6.785368 \* x + -173.429351  MSE = 38.131234 | y = 0.005384 \* x ^(2.641377)  MSE = 32.514733 |

Therefore, linear regression has a better performance when gravity is low, while exponential regression works better under a regular-gravity condition. However, the conclusion isn’t accurate enough because of inadequacy of data.

Code Modified:

* Save the data as a csv file, so that the reading process will look concise.
* The data type is changed to pandas.DataFrame, also to make the code concise.
* Linear and exponential regression added.
* Plot the scatter data and the regression graph and save it.

**Task 2**



The figure above indicates how constants and errors change (within totally 6000 generations/epochs) under different initial guesses and mutation magnitude. Mutation magnitude are adjusted to make the genetic algorithm come to ideal results. With a more dramatic initial guess (farther from the optimal parameters), the mutation magnitude has to be adjusted larger for a broader range of trial. Parameters and some of the results are listed below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Parameters | | | |
| n1 | n2 | n3 | Mutation  Magnitude |
| 1 (Original) | 2.7e-4 | 4 | 6.3e-2 | 9% |
| 1 (Optimal) | 7.640e-4 | 3.194 | 6.064e-2 |
| 2 (Original) | 3.0e-3 | 10 | 1 | 20% |
| 2 (Optimal) | 1.290e-3 | 3.055 | 5.000e-2 |
| 3 (Original) | 1 | 20 | 5 | 50% |
| 3 (Optimal) | 3.682e-3 | 2.755 | 1.098e-2 |
| 4 (Original) | 2 | 100 | 20 | 80% |
| 4 (Optimal) | 9.000e-4 | 3.150 | 4.700e-2 |

|  |  |  |
| --- | --- | --- |
| Experiment | Results | |
| Relative Absolute Error | Best Epoch |
| 1 | 2.958% | 3679 |
| 2 | 2.872% | 3233 |
| 3 | 3.186% | 485 |
| 4 | 2.935% | 682 |

Some more details can be found out from the tables. Though different solutions are reached when initial guesses vary, all of the solutions have the same order of magnitudes, which means these solutions are closed to the exact optimal solution. All of the relative absolute errors are near 3%, and it also proves genetic algorithm works well in this case.

As this problem can also be considered as a linear regression problem, therefore we could create a baseline with linear regression algorithm (See Task2\_LR.py). Listed below is the result of linear regression.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameters | | | Results |
| n1 | n2 | n3 | Relative Absolute Error |
| 3.933e-3 | 2.746 | 4.943e-2 | 2.995% |

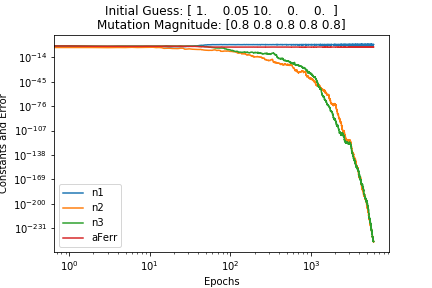
It is amazing that the solutions from Experiment 1, 2, 4 perform better than linear regression when the loss function is Relative Absolute Error (no doubt linear regression has the minimum mean square loss). All of the results above prove genetic algorithm to be potential when the model is non-linear or the loss function is partly non-differentiable.

The result of task 2 also aligns with that of task 1. One of the conclusions from task 1 is that the relationship between heat flow and superheat varied a little under different conditions of gravity. According to task 2, there are two orders of magnitude difference between n2 and n3, correspondingly represents the weight of superheat and gravity. It is a quantitative proof for the conclusion of task 1.

However, adjusting the mutation magnitude doesn’t always promise good solutions. Furthermore, if initial values are extremely badly chosen, the algorithm will hardly approach the optimal solution, no matter how mutation magnitude is chosen. Here’s an example.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Parameters | | | |
| n1 | n2 | n3 | Mutation  Magnitude |
| 5 (Original) | 1 | 0.05 | 10 | 80% |
| 5 (Optimal) | 96.70 | 1.7e-5 | 2.072e-2 |
| LR sol. (Comparison) | 3.933e-3 | 2.746 | 4.943e-2 | / |

|  |  |  |
| --- | --- | --- |
| Experiment | Results | |
| Relative Absolute Error | Best Epoch |
| 5 | 9.576% | 61 |



As shown in the graph, n2 and n3 rapidly slopes to zero. However, it’s not the single case. Similar trends could also be observed in experiment 2, 3, 4 (the n3/green line).

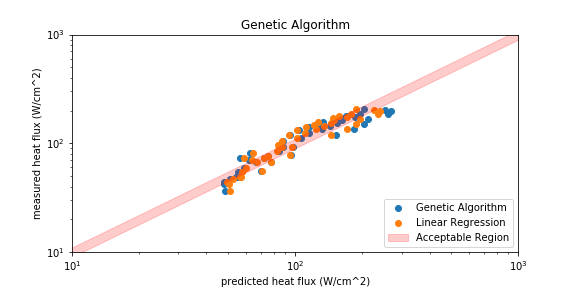
Here is my explanation to this phenomenon: genes naturally have a higher probability to become less. Each gene has a uniform chance to mutate by [-m, m], where m is the magnitude of mutation. However, according to the property of uniform distribution, after several generations, the new genes are more possibly to be less than the original ones (even though the expectation doesn’t change). This phenomenon is especially observable when m is large.

It can explain why parameter n3 also slopes to zero in experiment 3 and 4, since the original parameter is chosen large enough, genetic algorithm passes by (but not converges into) the optimal solution. Consequently, many of generations are wasted, as the best generations for these two experiments are 485 and 682 (far less than 6000). As for experiment 5, the original n2 plays a trivial role in the equation, and therefore it will follow its natural trend to decline into 0.

So, here’s some conclusions from these experiments.

1. If the right values are unknown, large original parameters and mutation magnitude can be chosen.
2. Use small mutation magnitude to fine-tune the solution.
3. Uniform distribution may not be good enough for mutation rate. Normal distribution can be tried.

Also we can use the optimal parameters generated from genetic algorithm and linear regression to see if the predicted heat flux aligns with true heat flux.



The blue prints are results from genetic algorithm, while the orange points are those from linear regression, and the red region is the acceptable region where absolute error is within 10%. An unacceptable rate is defined to describe the proportion of points outside the acceptable region. The statistical results are listed below.

|  |  |  |  |
| --- | --- | --- | --- |
| Methodology | RMS Error | RMS Deviation | Unacceptable Rate |
| Genetic Algorithm | 0.02679 | 0.1752 | 53.33% |
| Linear Regression | 0.02433 | 0.1627 | 62.22% |

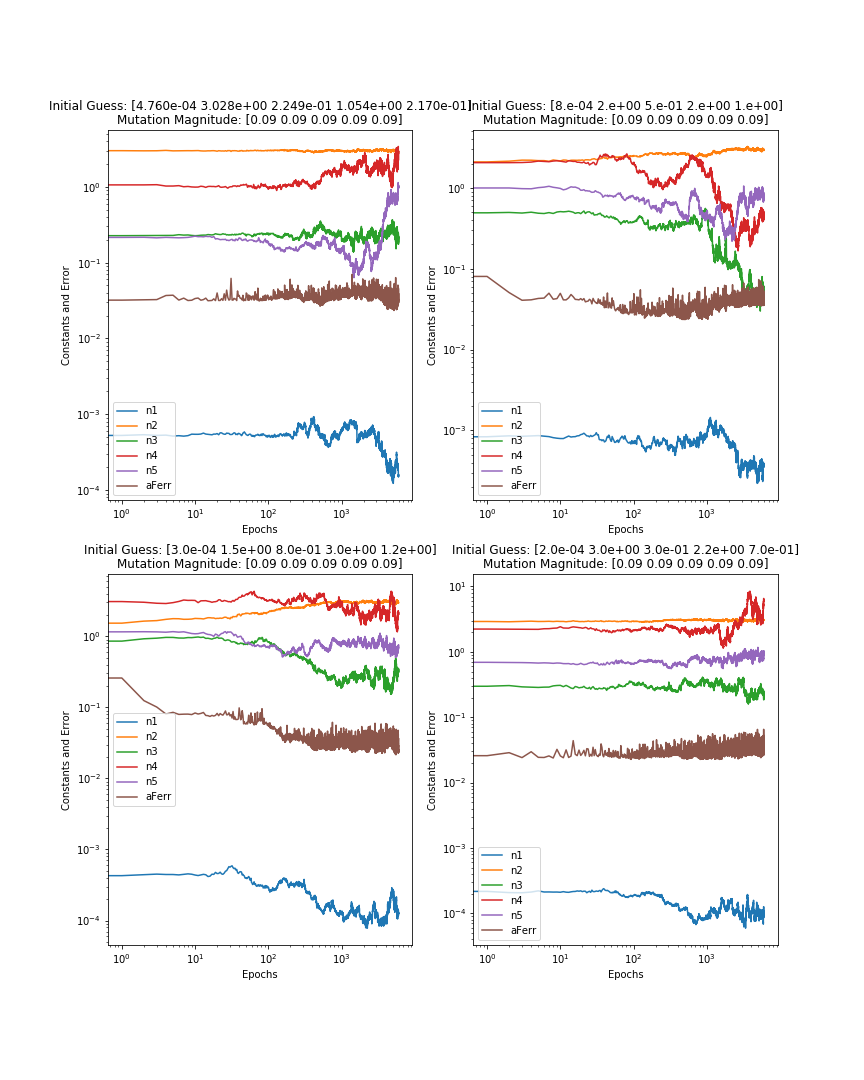
For both of the 2 methodologies, deviation is larger than the uncertainty. It may not be a good fit for this case because of the high RMS deviation and unacceptable rate as well. The RMD deviation is above the noise in the data. Therefore, this model is not accurate enough to predict the trend for this problem. To further improve the accuracy of the prediction, we need to find some different models and maybe more parameters.

Code Modified:

* Reconstruct the code to align with my coding habits.
* Use functions and packages (numpy, pandas) to make the codes concise.
* Include a linear regression algorithm as a baseline.
* Make the graphs easier to read.

Task 3

Now, a 5-parameter model is applied to this problem. Since the model can no longer be transformed into a linear regression problem, only genetic algorithm is feasible. With some simple modification to the codes, we can obtain a constants-and-error plot for this new model.



Experiments are conducted 4 times. However, according to the conclusion from Task 2, all of the initial parameters are chosen closed to the optimal solution to promise and magnitude of mutation is chosen as 0.09, so as to promise the convergence and accuracy of the algorithm.

Some of the detailed parameters and results are listed below.

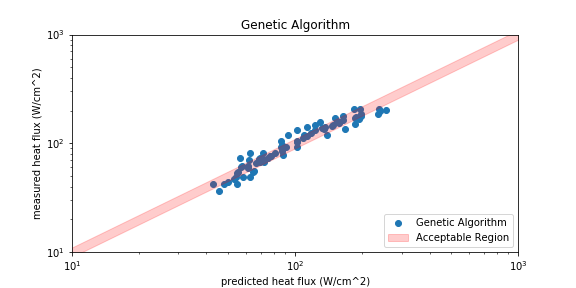
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Experiment | Parameters | | | | | |
| n1 | n2 | n3 | n4 | n5 | Mutation  Magnitude |
| 1 (Original) | 4.76e-4 | 3.028 | 0.2249 | 1.054 | 0.217 | 9% |
| 1 (Optimal) | 1.546e-4 | 2.976 | 0.3454 | 1.798 | 0.6690 |
| 2 (Original) | 8e-4 | 2 | 0.5 | 2 | 1 | 9% |
| 2 (Optimal) | 6.619e-4 | 2.675 | 0.3712 | 1.716 | 0.4266 |
| 3 (Original) | 3e-4 | 1.5 | 0.8 | 3 | 1.2 | 9% |
| 3 (Optimal) | 1.669e-4 | 2.946 | 0.4143 | 1.572 | 0.5742 |
| 4 (Original) | 2e-4 | 3 | 0.3 | 2.2 | 0.7 | 9% |
| 4 (Optimal) | 1.065e-4 | 3.004 | 0.3637 | 2.018 | 0.7475 |

|  |  |  |
| --- | --- | --- |
| Experiment | Results | |
| Relative Absolute Error | Best Epoch |
| 1 | 2.336% | 5196 |
| 2 | 2.347% | 505 |
| 3 | 2.151% | 5524 |
| 4 | 2.255% | 1496 |

Despite different initial guesses, all of the experiments have similar optimal solution to this problem. All of the relative absolute errors are around 2.2%, which is more likely to be a precise prediction model compared with the one in Task 2. As experiment 3 has a minimum error among all the experiments, solution from experiment 3 is chosen for further use in the following tasks.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | n1 | n2 | n3 | n4 | n5 |
| Chosen Parameters | 1.669e-4 | 2.946 | 0.4143 | 1.572 | 0.5742 |

Deviation and other statistical features can also be calculated in the same way as Task 2. Results are shown below.



|  |  |  |  |
| --- | --- | --- | --- |
| Methodology | RMS Error | RMS Deviation | Unacceptable Rate |
| Genetic Algorithm | 0.01484 | 0.1289 | 42.86% |

All of the criterions are having a better result compared with task 2, which means the 5-parameter model is more precise than the 3-parameter model. However, the RMS deviation of 12.89% is still higher than measurement uncertainty, while the unacceptable rate is still as high as 42.86%. There’s still way to improve the model.

Code Modified:

* Loss Function