NE 155 Final Report

Thermal Heat Transfer Coefficient of Copper Pebbles in Drakesol 260AT

Introduction

In the Pebble-Bed Heat Transfer Experiment (PBHTX), the thermal heat transfer coefficient of copper pebbles in simulant fluid Drakesol is going to be determined analytically. In accordance with other experiments performed in Dr. Per Peterson's lab, copper pebbles ≈1 cm in radius will be used in conjunction with a dimpled test section to simulate a larger reactor core. The results of PBHTX can be used in the further development of Pebble-Bed Fluoride Salt Cooled, High Temperature Reactors (PB-FHR), currently in its Mark I stage. Ensuring an adequate heat transfer from the fluid to the pebbles is vital to the designs of pebble-based fuel reactors. In this paper, the basics of the experiment will be covered, in conjunction with an analysis report using techniques such as numerical differentiation and different forms of interpolation.

Problem Description

To date, no attempts have been taken to confirm the thermal heat transfer coefficients for pebble-based fuels in fluoride salts. Because this experiment would require massive resources, a scaled side effects experiment was undertaken by Doctoral Candidate Lakshana Huddar, in an attempt to further the developmental work for PB-FHRs. With Drakesol being used as a simulant fluid for molten fluoride salts, an adequate simulation for experimentally determining heat transfer coefficients was built by Ali Albaaj, Lahshana Huddar, and Jeff Bickel. Analytically determining the scaled heat transfer coefficients allows for changes to be made in future design of PB-FHR experiments.

Description of Work

In 2015, PBHTX was completely constructed and instrumented. Figure 1 below shows the experimental setup of PBHTX, including instrumentation.

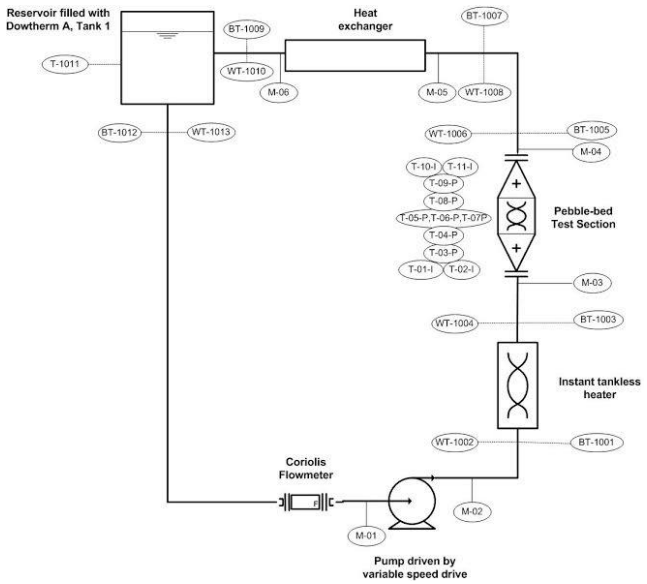

Figure 1: Schematic of experiment

For experimental setup, Labview was used to interpret the data signals, while also altering the power setting of the tankless heater, giving direct control of loop temperature. Sinusoidal wave patterns were utilized to create larger temperature differential patterns, giving much more diverse temperature ranges.

For determining the heat transfer coefficient, a simple energy balance is created, with algebra to isolate the heat transfer coefficient. The convective heat transfer from the pebbles to the fluid must equal the rate of change of internal heat generation in the pebbles, as shown in Equation 1. Equation 2 was used for the calculations of the heat transfer coefficients.

$$ha_v(T_s - T_f) = (1-\epsilon)\rho C_p \frac{dT}{dt}$$

Equation 1: energy balance for the convective heat transfer from pebbles to fluid and the rate of change of internal heat generation inside the pebbles; where: Cp = specific heat capacity of copper (0.39 kJ/kg K(Engineering Toolbox)); ρ = density of copper (8940 kg/m^3 (Engineering Toolbox)); ϵ = porosity of test section (48%); av = specific surface area (567* m^{-1}); dT/dt = time derivative of pebble temperature; ΔT = temperature difference between fluid and pebbles.

$$h = \frac{C_p \rho (1-\epsilon)}{a_v} \left| \frac{\frac{dT}{dt}}{\Delta T} \right|$$

Equation 2: isolation of the heat transfer coefficient.

For evaluation of PBHTX, numerical differentiation codes for Newton-Cotes were created for backward, forward, and center difference method. In addition, a lower and higher order of accuracy code was created for each Newton-Cotes method. These codes were applied to the time derivative of temperature required in the heat transfer coefficient equation.

Differing modes of interpolation code were developed to determine the best mode of interpolation. Polynomial interpolation function were developed and compared to sample data, then compared to the built-in cubic spline function in numpy, ensuring accuracy of developed code.

Code Use

A function "readCSVColumn" was created to filter out unwanted data while retrieving all data desired. "readCSVColumn" takes a list of column values from a CSV file, filters out strings, and removes the unwanted numerical values, such as channels, XO, and others that would alter the behavior of the data; it returns a numpy array of float numbers ready for use in the rest of the python script provided. Appendix B explains the filters within the code.

The numerical differentiation code developed used Newton-Cote's method of differentiation, creating a function for forward difference, backwards difference, and central difference. Inputs of differentiation needed are the x-values, y-values, and number of points evaluated at. The output returns the value of the error, the value of the grid size, an array containing derivative at each point, and an array displaying the evaluated points.

The numerical differentiation function, called "TemperatureDiff," takes 4 inputs, a temperature vector, a time vector, a string of either 'center,' 'backward,' or 'forward,' and an integer for the order of accuracy, either 1, 2, or 4. Once one of the methods has been selected, the function evaluates the derivate at the various points allowed by the order of accuracy. An example of this is demonstrated by Figure 2 below.

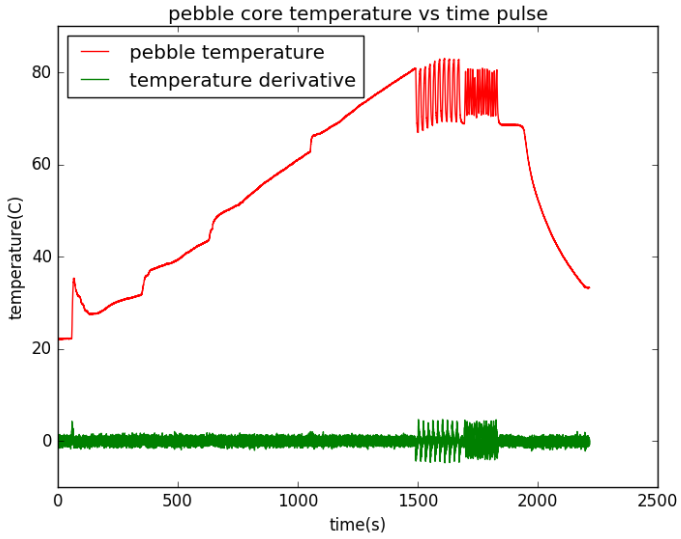

Figure 2: Core temperature of pebble5 with Newton-Cotes center differential method, order of accuracy=2

The lack of clarity in the patterns of the derivative is explained by the noise of the temperature readings. With a reading frequency of 20 Hz, small disparities can have large sway on the behavior of the temperature. The most important point of Figure 2 is the increase in magnitude of derivate change during the pulsing of the experiment, validating the correct behavior of the code.

Polynomial interpolation was developed for use in this paper. Polynomial interpolation for the code required the use of two functions, "FindPoly," which creates a matrix of x-values raised to their corresponding polynomial power and finds the least squares approximation of the actual coefficients with the y-values inputted into the function. To interpolate, an array with arbitrary grid size could be used in the function "PolyVal," based upon the same named function in MATLAB. "PolyVal" takes each coefficient outputted from the "FindPoly" function, multiplies it by the corresponding x-values inputted into the function, and returns an array of same length to the x-values with its corresponding evaluation from the polynomial interpolation. Due to the lack of a method for piecewise interpolation for polynomial interpolation, the corresponding values fluctuate heavily and do not match expected values. Figure 3 displays the differences between the developed polynomial interpolation and the built-in cubic spline interpolation.

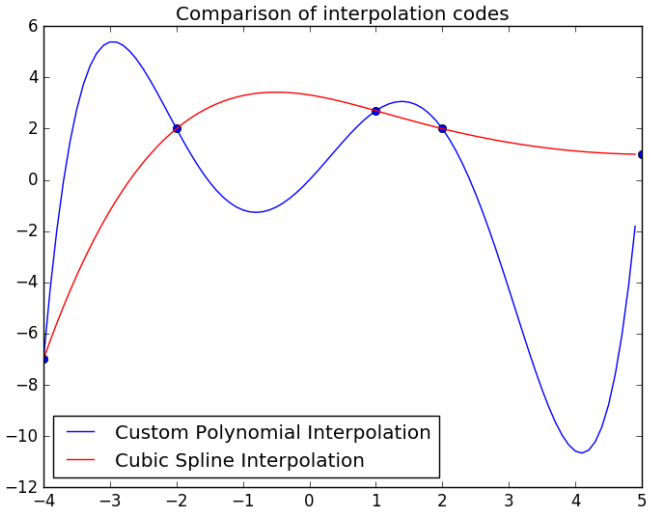

Figure 3: Interpolation code comparison

Due to a lack of accuracy regarding the developed code, and a lack of adequate computing power, only the built in cubic spline interpolation was used in analysis of the information given.

Results

In analyzing the data from PBHTX, only the thermocouple temperatures from inside the test section were used to compute the heat transfer coefficient. While the rest of the information is necessary for reproducibility and maintaining a level or certainty, parameters and data from outside the test section is not necessary for the scope of this paper. In temperature analysis, three pebble core temperatures were measured throughout the experiment along with two inline bulk temperatures. Figure 4 shows the raw data retrieved from the earliest pulse tests conducted.

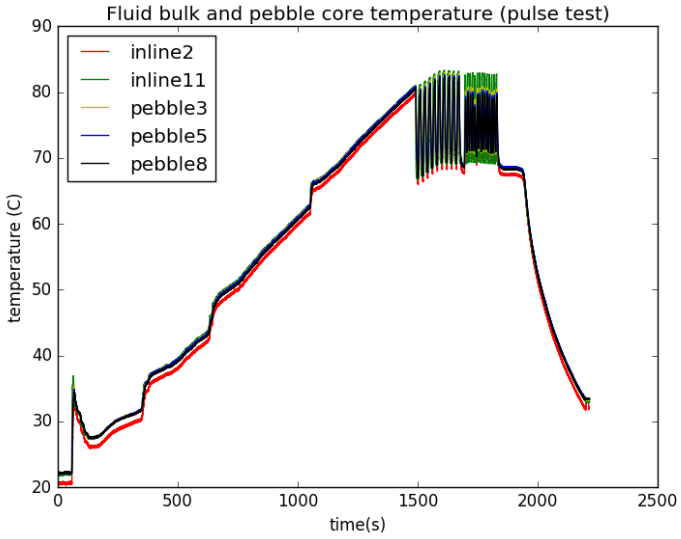

Figure 4: Raw data from PBHTX pulse experiments

Demonstrated above, much of the raw temperatures are very near each other, allowing for only the extremes to be used. The most notable temperature curve is "inline2," which is a thermocouple located near the top of the test section, farthest away from the heater. The location of the thermocouple explains the lower temperature of "inline2" relative to the rest of the data. Figure 5 below has a more detailed look at the curvatures existing during the pulse periods for the pebble cores.

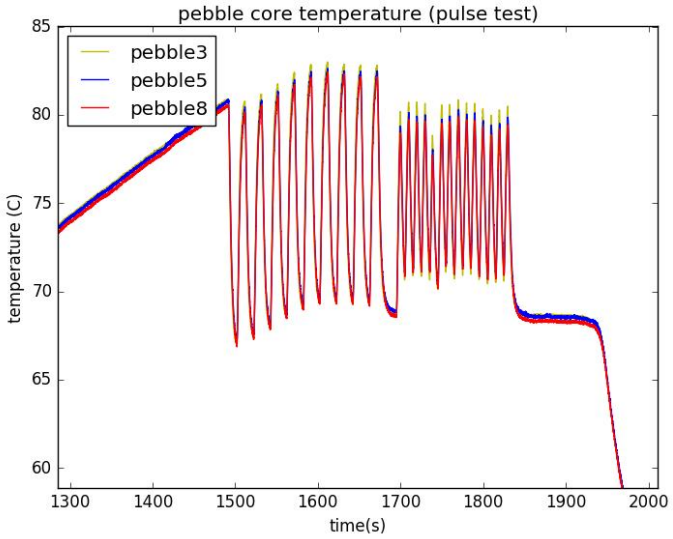

Figure 5: Pebble core behavior during pulses

Looking at Figure 5, the temperature disparities from pebble to pebble are slim, allowing simpler calculations of the heat transfer coefficient, since an arbitrary decision would not affect the ending result of the equation by anything noticeable. Thus, the data from “pebble3” and “inline2” were chosen to maximize the temperature difference between the pebbles and the fluid, minimizing the heat transfer coefficient. Figure 6 graphs the heat transfer coefficient (minus a few extreme outliers) with the pebble and bulk temperatures used using Newton-Cotes differentiation method with an order of accuracy of 2.

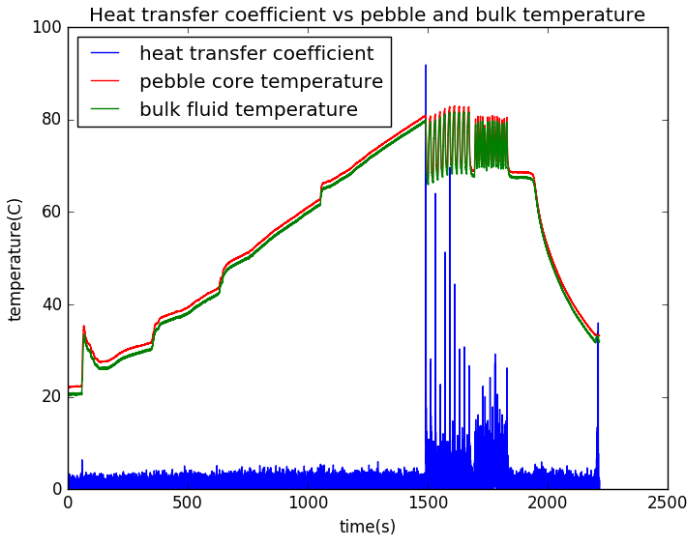

Figure 6: Heat transfer coefficient results

As expected from Equation 1, areas with the largest magnitude time derivative have the most significant heat transfer coefficient. From the large disparities of the pulse regions, the median of the heat transfer coefficient was calculated, minimizing the overall effect these massive numbers may have over other statistically useful numbers such as the average. The median was determined to be 0.96. The method of numerical differentiation in this case was center Newton-Cotes with an order of accuracy of 2.

An interesting phenomena occurred when the order of accuracy for derivatives was increased. The derivative at almost every point dramatically increased in magnitude. Figure 7 displays pebble3 core temperature vs the time derivative of temperature via Newton-Cotes center difference method with an order of accuracy of 4.

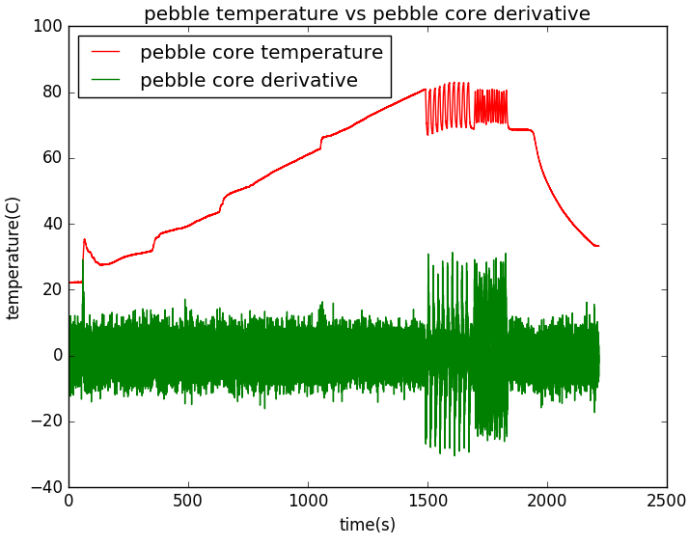

Figure 7: Pebble core temperature vs time derivative: Newton-Cotes center difference, order of accuracy=4

When the higher accuracy derivative was used, the frequency of extreme outliers increased tenfold, such that it was unwise to remove the outliers from Figure 8. In addition, the median value of the heat transfer coefficient increased tremendously, to 7.70, an unrealistic value. Thus, Newton-Cotes center difference with an order of accuracy of 2 is the better choice for determining and reporting the heat transfer coefficient.

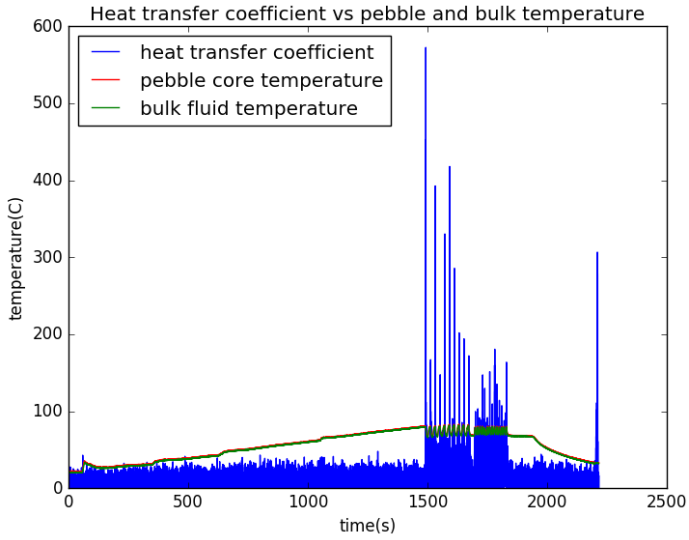

Figure 8: Heat transfer coefficient determined with Newton-Cotes center difference method, order of accuracy=4

Limitations of Analysis

Storing large data sets in a numpy array is memory intensive and extremely large data sets with high reading frequency become difficult to process. More experimental work has been conducted, with smaller frequency temperature pulses and higher temperatures, however reading this data with the created “readCSVColumn” function is too memory intensive for the current mode of analysis. When attempting to limit the data processed, available computing power was still not enough to complete the task desired.

Conclusion

A median heat transfer coefficient of 0.96 is good news for the pebble-bed reactor community. The heat transfer losses for pebble-based fuel is minimal, at 4%, allowing for further investigation into the macroscopic aspect of an actual reactor core without worries of overheating the fuel. For future experiments, the measuring frequency and heat generation oscillation frequency will be drastically decreased, as the fit of cubic spline to a limited amount of existing data is adequate for longer tests with longer sinusoidal periods (see Figure 9 in the Appendix A). Future work still needs to be conducted regarding differences in the heat transfer coefficient radially from the center and axially from the heat source.

Works Cited

"Metals and Alloys - Densities." *Metals - Specific Heats*. The Engineering Toolbox, n.d. Web. 10 May http://www.engineeringtoolbox.com/metal-alloys-densities-d_50.html

"Metals - Specific Heats." *Metals - Specific Heats*. The Engineering Toolbox, n.d. Web. 10 May 2016. http://www.engineeringtoolbox.com/specific-heat-metals-d_152.html

Appendix A

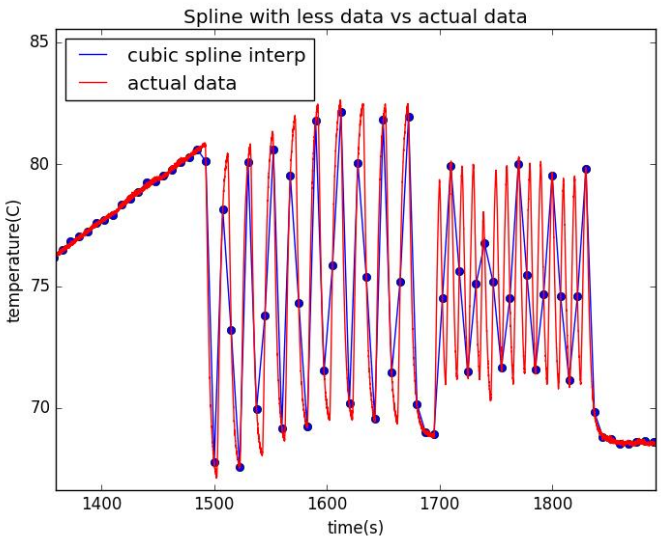

Figure 9: Cubic spline interpolation for limited data points in the pulse region

Appendix B

"readCSVColumn" was a function developed in order to read and filter the labview data provided in a CSV file created from a LVM file (the default data export for labview data). "readCSVColumn" takes a list of strings varying in values from strings to floating numbers within the string and converts the list to a numpy array by use of a for loop with a try except function within. If the value within the index selected is able to be converted to a float, then the if statements within the try function will be executed. When an exception is brought up, the for loop continues, ignoring the value that does not fit in a number array. The values removed along with reasoning for removal are in Table 1.

Indexes of (31*n) + 19	These values indicated the initial starting second for each second, creating large numbers in the thousands
24	Number of channels measuring temperature
20	Samples per second
2	Writer and reader version of labview included in the spreadsheet
0.05	The change in x values was included in the spreadsheet

Table 1