Baffled Boys Heat Exchanger

Lucas Huber, Zachary Karpel, Joshua Cowell, Caden Solle, Caedmon Eng

Liberty University, School of Engineering

November 21, 2025

Abstract

The objective of this study was to design a heat exchanger that meets a client's required deliverables of mass flow rate, temperature, overall heat transfer coefficient, and length while minimizing the annual cost. To achieve this task, hand calculations were done using MATLAB to iterate millions of heat exchangers in minutes, using conditional statements to ensure heat transfer equations implemented only when all the constraints were met. After running the heat transfer calculations, economic equations calculating the annual cost were employed. The algorithm selected the best exchanger by minimizing annual cost. The hand calculations proposed a shell and tube heat exchanger that implemented crossflow using baffles to manipulate flow. However, due to computational constraints in software, simulations verifying the calculations failed to complete. Future studies could validate the calculations performed on this model of heat exchanger. An alternate, annular design was selected to allow for simulation. An ANSYS mesh study was performed to refine the mesh and strengthen the simulation results. ANSYS contour plots were also gathered to find trends and correlations in the fluid flow. Once this was completed, hand calculations and simulation values were checked against each other to verify results. The percent errors for heat transfer, NTU, and effectiveness were determined to be 14.03%, 43.48%, and 14.80%, respectively. These discrepancies can be attributed to hand calculation simplifications and error inherent in any empirical correlation. In contrast, ANSYS runs many complex algorithms to achieve more accurate, realistic flow. Regardless, the errors were minor considering the assumptions made. This validates the chosen annular heat exchanger and confirms it would meet the requirements of the customer while being competitive in cost.

Word Count: 271

Introduction

The team was tasked with designing a heat exchanger that met certain customer specifications. Some key customer requirements can be seen in Table I. Other requirements include one shell pass, one tube pass, an overall heat transfer coefficient value between 850-1700 W/m^2·K, and a length between 1-10m.

TABLE I
CUSTOMER REQUIREMENTS

	Mass Flow Rate	Inlet Temperature	Outlet Temperature
Hot Water	28 kg/s	90°C	Unknown
Cold Water	27 kg/s	34°C	60°C

Beyond these specifications, multiple conservative assumptions were made to drive the design. It was assumed that the system was steady-state and that no heat was lost to the surroundings. Constant surface temperature was chosen as the boundary condition for the inner convection coefficient because it is a more conservative approximation than constant flux. Nusselt number correlations found in equations 8.53[1] and 8.55[1] were used. A lower Nusselt number translates to less heat transfer, which is safer in calculations. The surface roughness of drawn copper tubing was approximated as smooth, since minimal error exists for most Reynolds numbers. Based on McMaster Carr specifications, the copper tubing meets ASTM standard B88, meaning the tube contains 99.9% copper [2]. Pressure drop across any style tube bank was approximated by pressure drop across an equally spaced tube bank. For economic considerations, an interest rate of 8% was chosen with a desired life span of at least twenty years. In calculations, the process was further simplified by stating that axial conduction is negligible, potential and kinetic energies are negligible, specific heat capacity is constant, and the value of U (and UA) are constant. The calculations only considered counter flow, as counter flow is generally more efficient than parallel flow [1]. Due to the creation of turbulence, staggered banks have more efficient than straight banks, so only staggered banks were evaluated. Baffles were needed to ensure the tubes were experiencing crossflow.

After defining all base assumptions, calculations were performed using MATLAB. All equations were surrounded by conditional statements to ensure adherence to each equation's constraints. The desired outlet temperature was quickly determined using Equations 11.6b[1] and 11.7b[1]. Tables were embedded into the MATLAB code, including the saturated water property table, different constants, and copper tube tables. This was done so that the code could independently iterate millions of heat exchanger designs. Using Figure 7.12[1], arrays were created to iterate different ST and SL ratios, seeking to analyze all combinations of rows, columns, tubes, tube spacings, and tube lengths. The calculations, embedded in nested for loops, accounted for all permutations. Two separate MATLAB scripts were written, mostly similar, but distinct in the way they approach the external flow. The first script models an exchanger with baffles to induce crossflow and uses the external convection correlations for flow over a bank of tubes. The second script models an exchanger with no baffles and uses the convection correlation for axial flow in an annulus. The script method uses Equation 7.59[1], constants from Table 7.5[1] and correlation factors from Table 7.6[1]. Next, the Nusselt number for flow over a bank of tubes was determined by using Equation 7.60[1]. The second script uses Table 8.2[1] for fully developed and laminar flow to find the Nusselt numbers. However, for turbulent flow, the Gnielinski correlation was used to solve for the Nusselt number [3]. To find the convection coefficient, the hydraulic diameter was used along with copper properties. It was determined that the crossflow correlations were most accurate to real life as the annulus correlations were not intended for a bundle of tubes and do not account for the extra induced turbulence. For the rest of the heat calculation, the scripts use the same equations. To work with internal flow, a Reynolds number was calculated using Equation 8.6[1]. Using Equations 8.58[1] and 8.62[1] to account for laminar or turbulent flow, the code then solved for the internal Nusselt number and convection coefficient. The only piece left to calculate was wall resistance and conduction, which was found using Equation 3.33[1]. Now that all the variables were determined, the team solved for the overall heat transfer coefficient, the total surface area, and log mean temperature difference using Equations 11.5[1], 11.15[1] and 11.17[1]. The total heat transfer capability could then be found using Equation 8.46a. NTU and effectiveness calculations were completed using Equations 11.18[1], 11.19[1], and 11.24[1]. Material cost was accounted for using basic geometric calculations and the costs embedded in copper tubing tables and preset costs of other materials. Pumping cost was determined by first finding the friction factor and head loss, using Equation 8.21[1] and Hibbeler's Second Edition of Fluid Mechanics with Equations 10-3[4], 10-4[4], and 10-5[4]. The required pumping power was evaluated using Equation 14-15[4] and a pump efficiency of 0.73 was assumed based on prior knowledge and industry experience. This was converted to cost using an average kilowatt hour price of $0.16/hr. Salvage values were then accounted for and economic analysis was conducted to find salvage value, new present value, and equivalent annual costs [5]. To select the best design, the algorithm kept track of the exchanger with the lowest annual cost that also meets all the client's thermal demands. Overall, the shell and tube heat exchanger was comprised of a staggered tube bank with baffles evenly spaced throughout to generate the desired crossflow. Since the bank of tubes could not be simulated due to computational limitations, the annular heat exchanger results provide the basis for the simulation geometry.

Word Count: 897

Crossflow Results (Calculations)

After computing 104,136,192 permutations of heat exchangers, the analysis provided the optimal heat exchanger given the requirements and assumptions. Table A.1 shows the various geometric values of this ideal heat exchanger. The tube length can be found in meters for easy comparison to the requirement, but the rest of the measurements are in English units. The SolidWorks drawing in Figure A.2 serves as a modeled representation of the geometric configuration. When solving for heat transfer values, tabulated properties needed to be accounted for. To work with values at different temperatures, rather than making a base assumption on fluid properties, tables were embedded with linear interpolation into MATLAB to take accurate values at every temperature needed in each iteration. When MATLAB ran through each iteration, different values were pulled to make the data consistently accurate with minimal simplifications. The team chose to analyze costs along with heat transfer, as cost drives corporate decisions. Online data was pulled from McMaster Carr to find prices and performed calculations to optimize for the minimum cost needed to meet the requirements. The heat transfer values, along with pumping power and costs, can be found in Table A.3. All values fall within the requirements and cost considerations have been accounted for. The problem with the crossflow orientation and baffles was the team's inability to simulate this flow due to computational resources. Because of this, another model was considered, the annular method, which was effectively simulated using ANSYS Fluent. The calculations for this model will be discussed in greater detail because its assumptions were readily validated via simulation.

Annular Results (Calculations)

Similar to the crossflow heat exchanger, annular calculations were performed using MATLAB to optimize the design. 286,866,684 permutations of annular heat exchangers were evaluated, with the chosen geometry detailed in Table II. Geometry was selected from many

TABLE II
GEOMETRIC CONFIGURATION: ANNULAR

Geometry	Measurement	Geometry	Measurement
Tube Type	M	**Number of Tubes**	196
Tube Length	6.4 m	**Tube Inner Diameter**	0.569 in
Number of Rows	14	**Tube Outer Diameter**	0.625 in
Number of Columns	14	**Shell Inner Diameter**	13.96 in
Nominal Tube size	0.569 in	**Single Tube Surface Area (Inner)**	450 in^2
ST Ratio	1.1	**Single Tube Surface Area (Outer)**	495 in^2
SL Ratio	0.9	**Annulus Cross Sectional Area**	0.0525 in^2
Tube Cross Sectional Area (Inner)	0.254 in^2	**Shell Cross Sectional Area (Excluding Tubes)**	92.9 in^2
Tube Cross Sectional Area (Outer)	0.307 in^2	**Total Tube Surface Area (Inner)**	8829 in^2

Geometry	**Measurement**	**Geometry**	**Measurement**
Shell Cross Sectional Area	153 in^2	**Total Tube Surface Area (Outer)**	9698 in^2

different loops and conditional statements to minimize the total material, which decreased the cost, while still meeting the heat transfer requirements. With this geometry in mind, two different configurations were technically satisfactory. Inside the shell, the first was a concentric arrangement that can be seen in Figure 1, whereas the second was a staggered arrangement of tubes that can be found in Figure B.1. The difference lies in the tube alignment. Staggered arrangements operate in rows and columns, whereas concentric arrangements have all the tubes circling around the most centric tube. Regardless of configuration, the heat results proved to be the same, since the annular method does not include flow over the bank of tubes. To perform heat

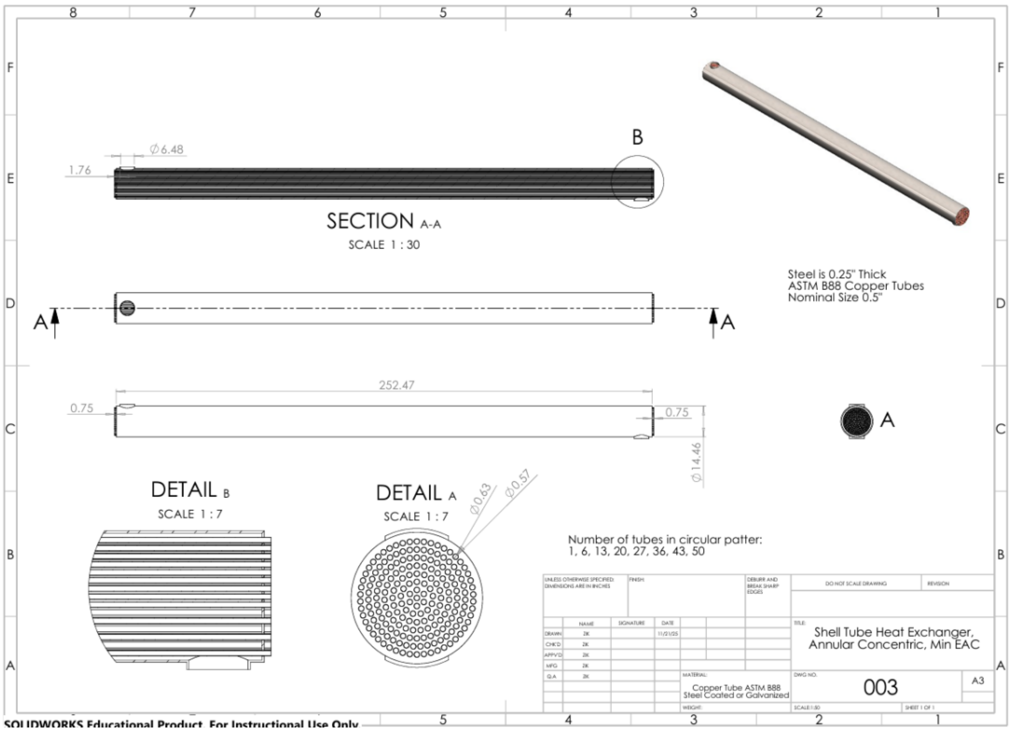


Fig. 1. Annular CAD Drawing: Concentric

calculations, the same method of finding fluid properties was used as that in the crossflow heat exchanger. Similarly, the design was chosen based on cost efficiency, so cost calculations were performed again. The performance and cost data appears in Table III. From the performance calculations, there were a couple of crucial pieces worth highlighting highlight, specifically the hot outlet temperature, heat transfer rate, NTU, effectiveness, pressure loss, and pumping power.

TABLE III
PERFORMANCE AND COST CALCULATIONS: ANNULAR

Performance Calculations	**Value**	**Cost Calculations**	**Value**
Iterations	286866684	**Net Present Value**	$18,068
Total q	$2.95 \cdot 10^6$ W	**Annual Operating Expense**	$2,237

Performance Calculations	**Value**	**Cost Calculations**	**Value**
Ui	1700 W/m^2·K	**Salvage Present Value**	\$655
U	1624 W/K	**Equivalent Annual Cost**	\$1,840
Factor of Safety	1.006	**Life Span**	20 years
Convection Coefficient (Inner, h_i)	6764 W/m^2·K	**Interest Rate**	8%
Tho	65.02°C	**Steel Material Cost**	\$4,909
Effectiveness	0.47	**Copper Material Cost**	\$11,577
NTU	0.86	**Pump Cost Per Year**	\$228
Pressure Loss Per Pipe	3937 Pa	**Exchanger Efficiency**	1.60 kW/USD
Total Pumping Power	164 W		

To begin with, the hot outlet temperature was determined by using a simple energy balance method, assuming all the heat gained by the cold fluid was lost by the hot fluid. Using this knowledge, Equations 11.6b[1] and 11.7[b] were equated and solved. This also gave the heat transfer rate, since the cold fluid properties of mass flow rate, inlet temperature, and outlet temperature remain constant. The effectiveness and NTU could be determined from Equations 11.18[1], 11.19[1], and 11.24[1]. By simply calculating the minimum heat capacity rate, the maximum heat transfer rate could be determined and compared with the actual value, giving effectiveness by Equation 11.19[1]. NTU was calculated using Equation 11.24[1] after the total surface area was determined. NTU and effectiveness relations crucially determine what configuration of heat exchanger to use, so these calculations served to verify the optimal orientation. Pressure loss and pumping power information was determined using Equations 10-3[4] and 14-15[4]. Since pressure and pumping power commonly appear in fluid analysis, a fluid mechanics textbook [4] was used as the primary reference for calculations. Essentially, the team sought to determine the head loss across a single tube using Equation 10-3[4], which could be easily correlated to pressure drop using Bernoulli's equation [4]. With pressure drop across a single tube known, pumping power could be determined using Equation 14-15[4]. Ultimately, all these values were run through a series of loops and iterated to ensure the values obtained were the most effective.

Results (ANSYS Fluent Simulation)

The optimized heat exchanger design was simulated using the representative reduced geometry outlined in the statement of deliverables: a single pipe in an annulus geometry. The cross-sectional areas of the cold fluid and hot fluid were matched in the simplified model, with the overall length of the model equivalent to the calculations as well. Because the large number of tubes was simplified to a singular tube of much greater dimension, the resulting surface area was smaller by a factor of fourteen. To account for this, when comparing the results of the simulation to those of the hand calculations, the overall heat transfer was multiplied by the surface area factor. This adjustment proved to provide a far more accurate value for the overall heat transfer, leading to a margin of error of a mere 14.03%.

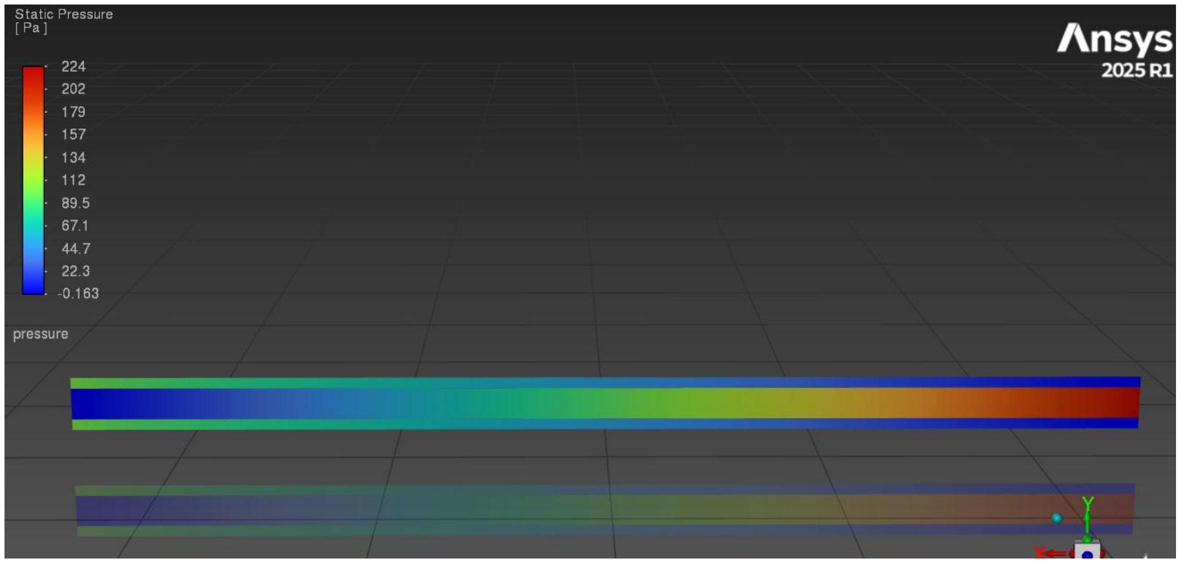


Fig. 2. Vertical Slice, Static Pressure Contour

Figure 2 presents the static pressure distribution, showing the expected decline in pressure as each fluid moves through the exchanger. The gradient reflects viscous losses along both flow paths.

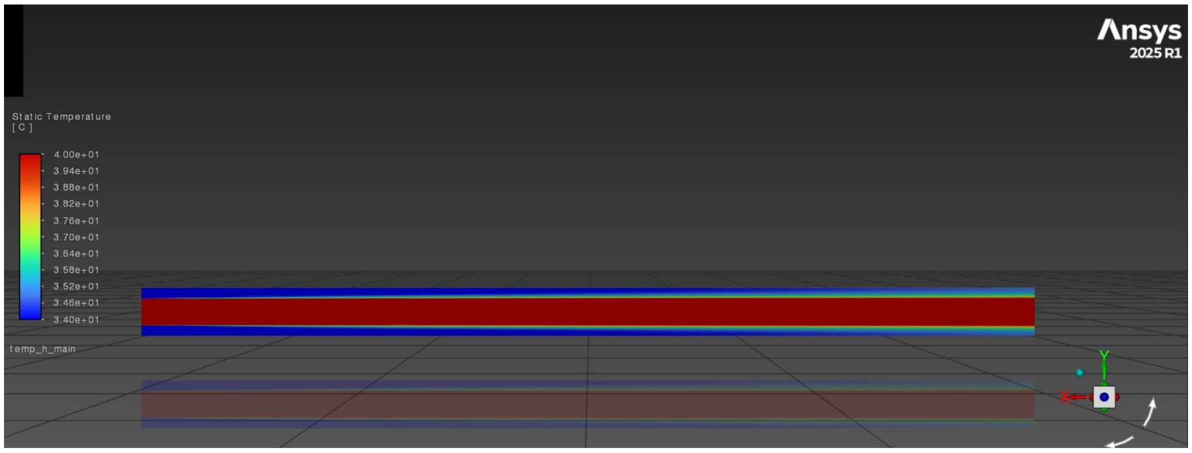


Fig. 3. Vertical Slice, Temperature Contour of Cold Fluid

Figure 3 shows the cold fluid gradually increasing in temperature along its flow path. This rise results from heat transfer across the inner pipe wall from the hot fluid moving in the opposite direction.

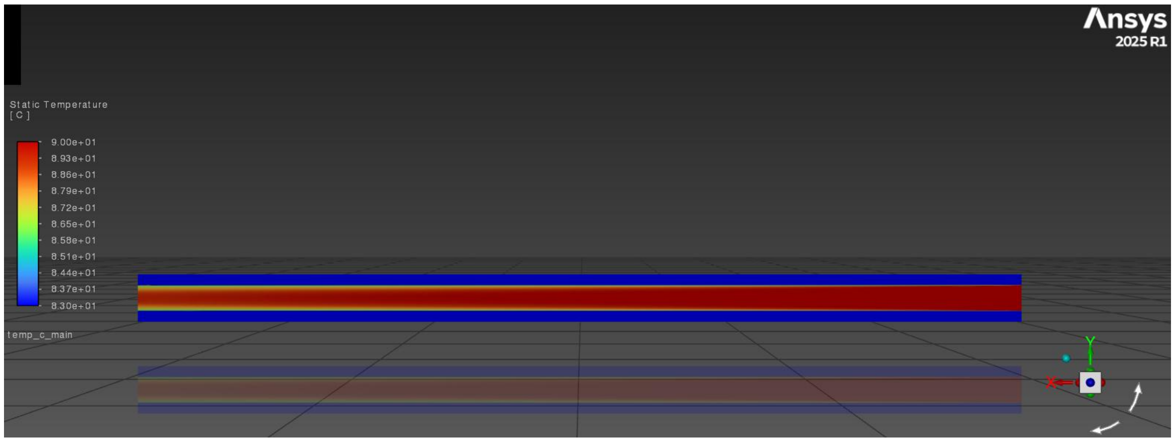


Fig. 4. Vertical Slice, Temperature Contour of Hot Fluid

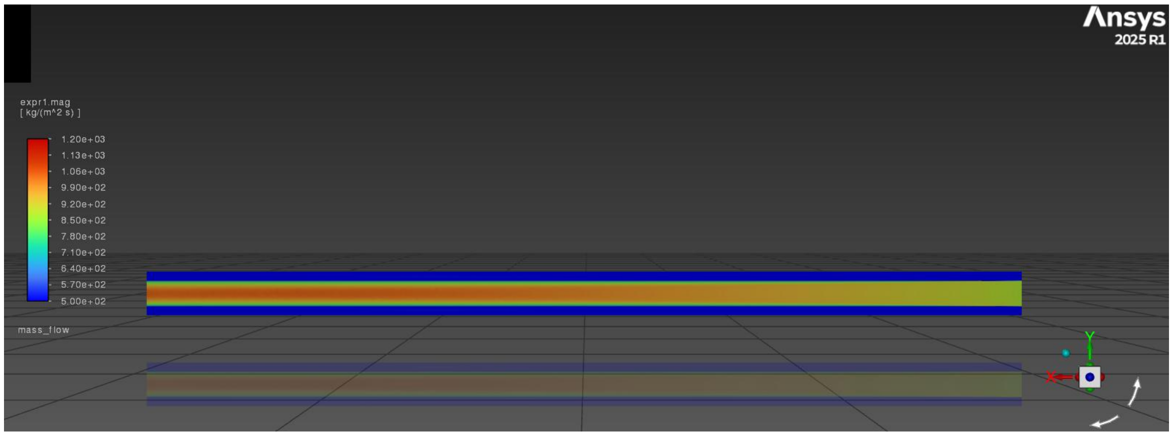


Fig. 5. Vertical Slice, Mass Flow Rate Contour

Figure 4 illustrates the progressive cooling of the hot fluid as it travels from right to left. The temperature gradient along the length demonstrates heat loss to the surrounding cold fluid in the annulus. Figure 5 highlights the development of the velocity profile along the pipe. The slice shows how the boundary layer grows and the flow transitions toward a fully developed state as it progresses downstream. Plots detailing the pressure, temperature, and mass flow rate at the centerlines of the hot and cold fluids can be found in Appendix D.

In the simulation process, the team modeled the annular concentric shell and tube heat exchanger with a singular tube and imported the geometry into ANSYS. The simulation was constrained with the predefined entry mass flow rates and temperatures of the hot and cold fluids. The exit temperature of the cold water, though given in the problem statement, was left undefined in the simulation to avoid over constraining the system in ANSYS and allowing the solver to calculate the exit temperatures itself. With the entry conditions set and the geometry successfully meshed, all that was required was to constrain the outlets as pressure outlets. The simulation was set to iterate 400 times, but it converged after 111 iterations. The result for the NTU was 1.32 and effectiveness was 0.574. The vertical slices of the heat exchanger for static pressure, temperature, and mass flow rate can be found in Fig. 2 through 5. Figures 3 and 4 show the temperature contours for the cold and hot fluids, respectively.

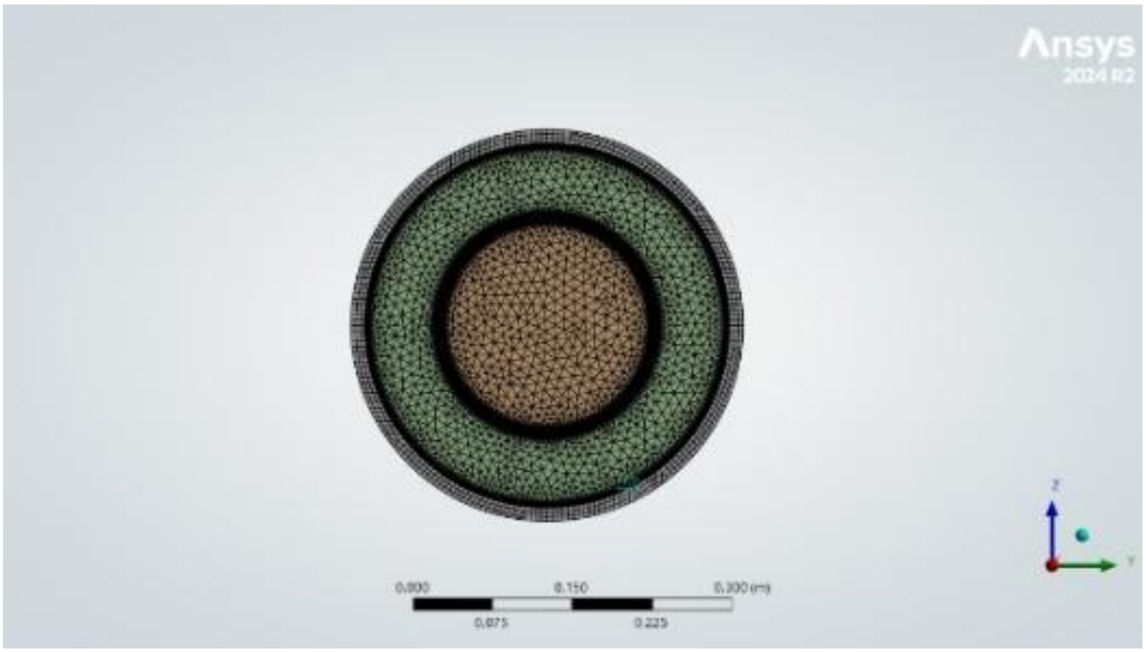


Fig. 6. Final Mesh Iteration

In the interest of further increasing the accuracy and thoroughness of the simulation, a mesh study of the annulus heat exchanger was performed. Starting with a coarse mesh of 31916 cells, the simulation was run and the difference between heat transfer of the cold pipe and that of the hot pipe was recorded. This parameter was chosen for the study as it was noticeably large in the first mesh iteration yet should equal zero as the heat transfer from the hot and cold fluids should be the same. The study was then iterated with finer and finer meshes of 94457, 176276, 569281, 1407382, 10558528, and 21728675 cells until the heat transfer difference converged (see Fig. 7). The heat transfer in the cold fluid was much higher than that of the hot fluid at iteration 1, and by the final iteration the difference between the two was practically zero. This enabled the group to trust both results and not need to pick a "more correct" one. Other mesh statistics were studied as well, including skewness, orthogonal quality, and element quality. However, they will only be discussed regarding the final mesh. The final mesh had a skewness value of 0.13434, and orthogonal quality of 0.87091, an aspect ratio of 8.1604, and an element quality of 0.55265; all numbers that indicate an extremely good quality mesh. The final mesh consisted of a global element size of 0.1 m, 15 inflation layers on the outside of the hot fluid, 15 inflation layers on the inside and outside of the cold fluid, and local tube and shell sizing of 0.005 m. The inflation layers were the biggest factor in the mesh refinement, as by simply adding a couple layers in iteration 4, it was able to improve the difference in heat transfer values by 1947%. This mesh refinement process ensures that the results end up independent of the mesh size, rather than changing by a statistically significant amount every time the mesh was altered. In other words, eventually the results stabilize and asymptote after adding additional nodes.

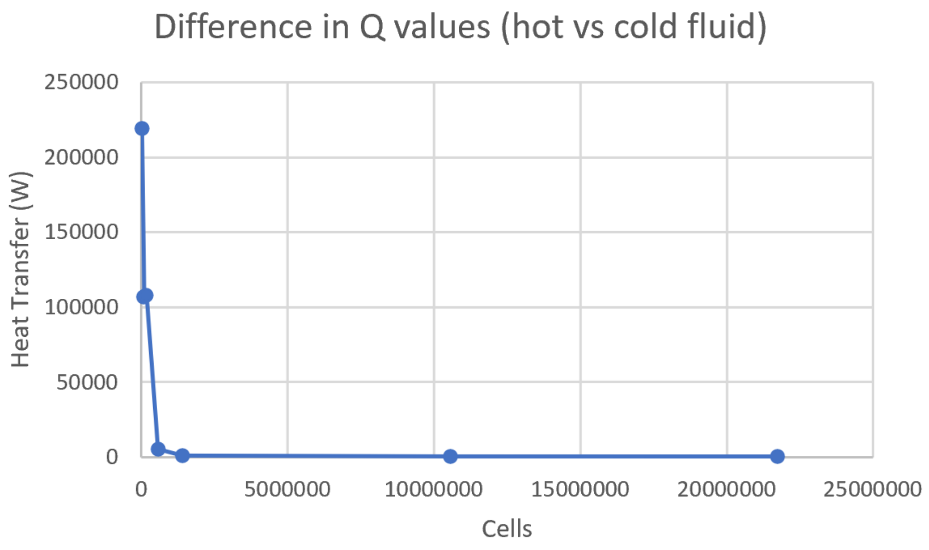


Fig. 7. Plot of Heat Transfer vs Number of Cells

In addition to the simulation of the simplified geometry, the team also attempted to simulate a more complete model of the optimized heat exchanger. Successful face meshes of approximately 24 million and 43 million were produced in ANSYS, though these proved too computationally intensive for the available resources. This full model of the heat exchanger included 10 baffles designed to alter the relative direction of the cold fluid to crossflow, a more efficient method for heat transfer within the exchanger. Baffles also increase the turbulence and mixing of the fluid, preventing thermal stratification which decreases the overall thermal efficiency of the system. To make up for this, additional simulations were run with the singular annulus design but including baffles to get a closer look at the characteristics of this design. Additional figures showing the results of this simulation can be found in Appendix F.

Discussion

The design and analysis performed with the conditions provided by the theoretical customer resulted in an annular heat exchanger that was 6.4 m long and included 194 copper tubes inside the steel shell. The calculated heat transfer coefficient was 1624 W/K and the overall heat transfer was $2.95{\cdot}10^6$ W. Minimizing cost was the primary concern in the analysis; between the initial cost of materials and the annual cost of power necessary to run the system, the equivalent annual cost was $1840 and the per-dollar efficiency was 1.60 kW/USD. The alternate design that incorporated baffles required 434 tubes and a length of only 3.7 m. The 7 baffles forced the fluid into cross flow across the bank of tubes, thus increasing the per-length efficiency of the heat exchanger. Though having a slightly higher equivalent annual cost, the calculations showed that this heat exchanger was more efficient than the annular one, with an exchanger efficiency of 1.36 kW/USD. Because of the simulation limitation for the baffled exchanger, the team stuck with the annular heat exchanger for the remainder of the project. While not perfect, these results should be fairly accurate. Some sources of error include the minor losses associated with the baffles altering the direction of flow that were not accounted for. Additionally, the exclusion of fouling calculations from the annular heat exchanger because ANSYS does not account for fouling would throw off the results slightly in real life. Overtime, fouling would decrease the efficiency of the heat exchanger because of all the built-up debris. The use of proper correlations helps to improve the overall accuracy of the results because the correlations find their basis in actual experimentation. Overall, the ANSYS Fluent simulations validated the hand calculations. For the annular design, the heat transfer was 3284000 W in the calculations and 3630744 W from the simulation, resulting in a 14.03% error. The NTU was 0.92 in the calculations and 1.32 in the simulation, an error of 43.48% which the team found to be acceptable given the circumstances. For effectiveness, the calculations showed 0.5 and the simulation gave 0.574, a difference of only 14.80%. Some of the differences between the two methods were that the hand calculations assumed recirculation and mixing of flow, whereas the ANSYS assumed uniform velocity. The hand calculations also used ideal flow and textbook correlations, while ANSYS simulated more realistic flow and solved energy and Navier-Stokes equations. Finally, the hand calculations assumed constant properties throughout as opposed to ANSYS including appropriate changes in the properties throughout the simulation. Given the scale of the project, the team considered these results to be immensely satisfactory. If the client was provided with a heat exchanger built off the specifications optimized by the calculations, a real-life model should meet the requirements, given that the simulation provided numbers even better than the calculations. Overall, the prescribed heat exchanger met the specifications and requirements laid out by the prospective client: a fixed entry mass flow rate and temperatures for the hot and cold fluids, a U between 850 and 1700 W/m^2·K, a singular shell and tube pass design, and a length between 1 and 10 meters. The final design used the given conditions, had a calculated U of 1624 W/m^2·K, successfully packaged in a single shell and tube design, and had a length of 6.4 m, fully meeting the specifications and requirements, while also being optimized for cost-effectiveness. The primary limitation encountered in pursuit of the optimal heat exchanger was a lack of the computational resources necessary to perform the simulations the team hoped to complete. While successful surface meshes were generated of the full heat exchanger model, the team could not complete the full volume meshes that were needed to run the desired simulations. The calculation process was also delayed by a lack of research regarding the friction factor for cross flow across a bank of tubes. The equations used were for a bank of tubes with the tubes spaced equilaterally apart. The spacing of the optimized heat exchanger was not equilateral, so the equation needed a correction factor to account for this alteration, which the team was unable to procure. This led to the pressure drop and corresponding pump power calculations being slightly less precise than hoped for, though not by much. Proper equations for axial flow through a bundle of tubes would have also proved beneficial for the hand calculations. Some ethical concerns when considering heat exchangers are the environmental impact and footprint. A poorly designed heat exchanger can dramatically increase the necessary power consumption of the system, leading to increased greenhouse gas emissions from the power source. Ethical responsibility would look like maximizing thermal efficiency to conserve resources and reduce the carbon footprint. Another concern is that of death and recycling. Large metal heat exchangers can be difficult to recycle because of the size and energy-intensive nature of the recycling process for durable metals like copper, aluminum, and stainless steel.

In conclusion, the result of this project was a successfully designed annular concentric shell and tube heat exchanger that met the client's specifications and would be considered economically competitive against alternate options and was validated by both hand calculations and ANSYS Fluent simulations.

Word Count: 862

References

[1] F. P. Incropera, *Fundamentals of heat and mass transfer*. John Wiley & Sons, 2007.

[2] Admin, “ASTM B88 Copper Water Pipes and Gas Tubes - MediStreams - Medical Oxygen,” *MedOx Copper Pipes*, Oct. 24, 2025. https://medicalpipelines.com/astm-b88-copper-pipes-tubes/#:~:text=Standard%3A%20Seamless%20Copper%20Water%20Tube,% 25%20(Cu%20%2B%20Ag%20combined)

[3] N. Connor, “What is Gnielinski Equation - Definition,” *Thermal Engineering*, Jun. 04, 2019. https://www.thermal-engineering.org/what-is-gnielinski-equation-definition/

[4] R. C. Hibbeler, *Fluid Mechanics*. Pearson Education, 2017.

[5] L. T. Blank and A. Tarquin, *Engineering economy*. 2024.

APPENDIX A

TABLE A.1
GEOMETRIC CONFIGURATION: CROSSFLOW

Geometry	Measurement	Geometry	Measurement
Tube Type	M	**Number of Tubes**	434
Tube Length	3.7 m	**Tube Inner Diameter**	0.5 in
Number of Rows	14	**Tube Outer Diameter**	0.556 in
Number of Columns	31	**Shell Inner Diameter**	23.0 in
Nominal Tube size	0.5 in	**Number of Baffles**	7
ST Ratio	1.5	**Baffle Spacing**	15.3 in
SL Ratio	1	**Single Tube Surface Area (Inner)**	229 in^2
Tube Cross Sectional Area (Inner)	0.196 in^2	**Single Tube Surface Area (Outer)**	254 in^2
Tube Cross Sectional Area (Outer)	0.243 in^2	**Single Tube Face Material Area**	0.0464 in^2
Shell Cross Sectional Area	415 in^2	**Shell Cross Sectional Area (Excluding Tubes)**	310 in^2
Total Tube Surface Area (Outer)	$1.10 \cdot 10^5$ in^2	**Total Tube Surface Area (Inner)**	$9.31 \cdot 10^4$ in^2

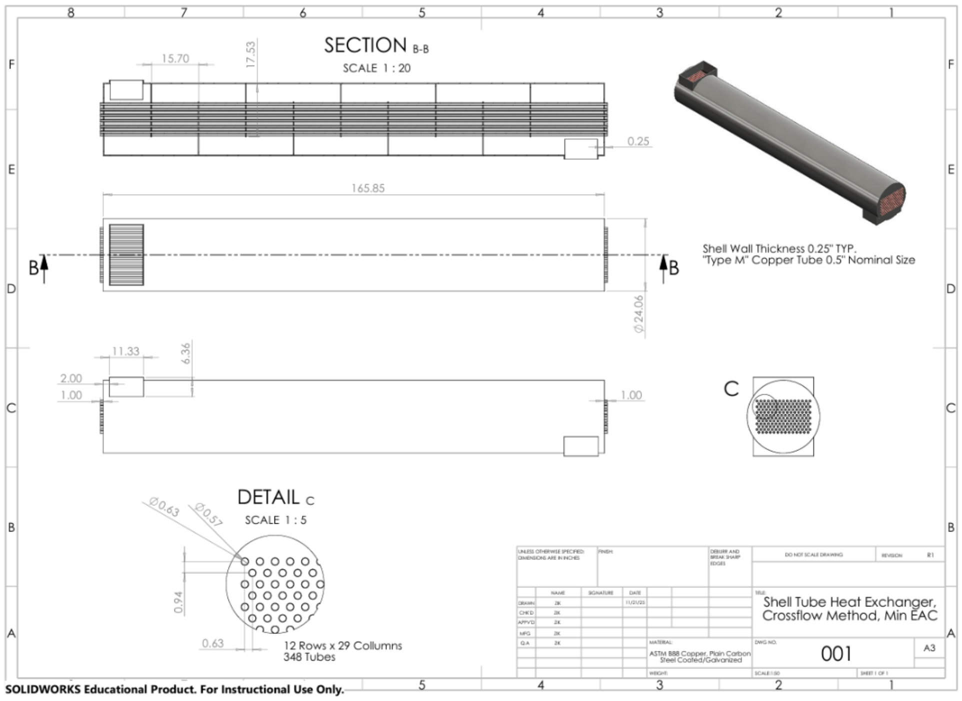

SECTION B-B
SCALE 1 : 20
15.70
17.53
0.25
165.85
B
B
Shell Wall Thickness 0.25" TYP.
"Type M" Copper Tube 0.5" Nominal Size
∅24.06
11.33
6.36
2.00
1.00
1.00
C
DETAIL C
SCALE 1 : 5
∅0.63
∅0.57
0.94
0.63
12 Rows x 29 Collumns
348 Tubes
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN INCHES
DEBURR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION R1
Shell Tube Heat Exchanger, Crossflow Method, Min EAC
MATERIAL: ASTM B88 Copper, Plain Carbon Steel Coated/Galvanized
DWG NO. 001
A3
SOLIDWORKS Educational Product. For Instructional Use Only.
Fig. A.2.CAD Drawing: Crossflow with Baffles

TABLE A.3
PERFORMANCE AND COST CALCULATIONS: CROSSFLOW

Performance Calculations	**Value**	**Cost Calculations**	**Value**
Iterations	104136192	**Net Present Value**	$21,197
Total q	$2.94 \cdot 10^6$ W	**Annual Operating Expense**	$1,842
Ui	1450 $W/m^2 \cdot K$	**Salvage Present Value**	$769
U	1402 W/K	**Equivalent Annual Cost**	$2,159
Factor of Safety	1.002	**Life Span**	20 years
Convection Coefficient (Inner, h_i)	4188 $W/m^2 \cdot K$	**Interest Rate**	8%
Convection Coefficient (Outer, h_o)	5786 $W/m^2 \cdot K$	**Steel Material Cost**	$6,636
Hot Outlet Temperature (T_{ho})	65.02℃	**Copper Material Cost**	$13,489
Effectiveness	0.47	**Pump Cost Per Year**	$188
NTU	0.85	**Exchanger Efficiency**	1.39 kW/USD
Pressure Loss Per Pipe	945 Pa		
Total Pumping Power	135 W		

APPENDIX B

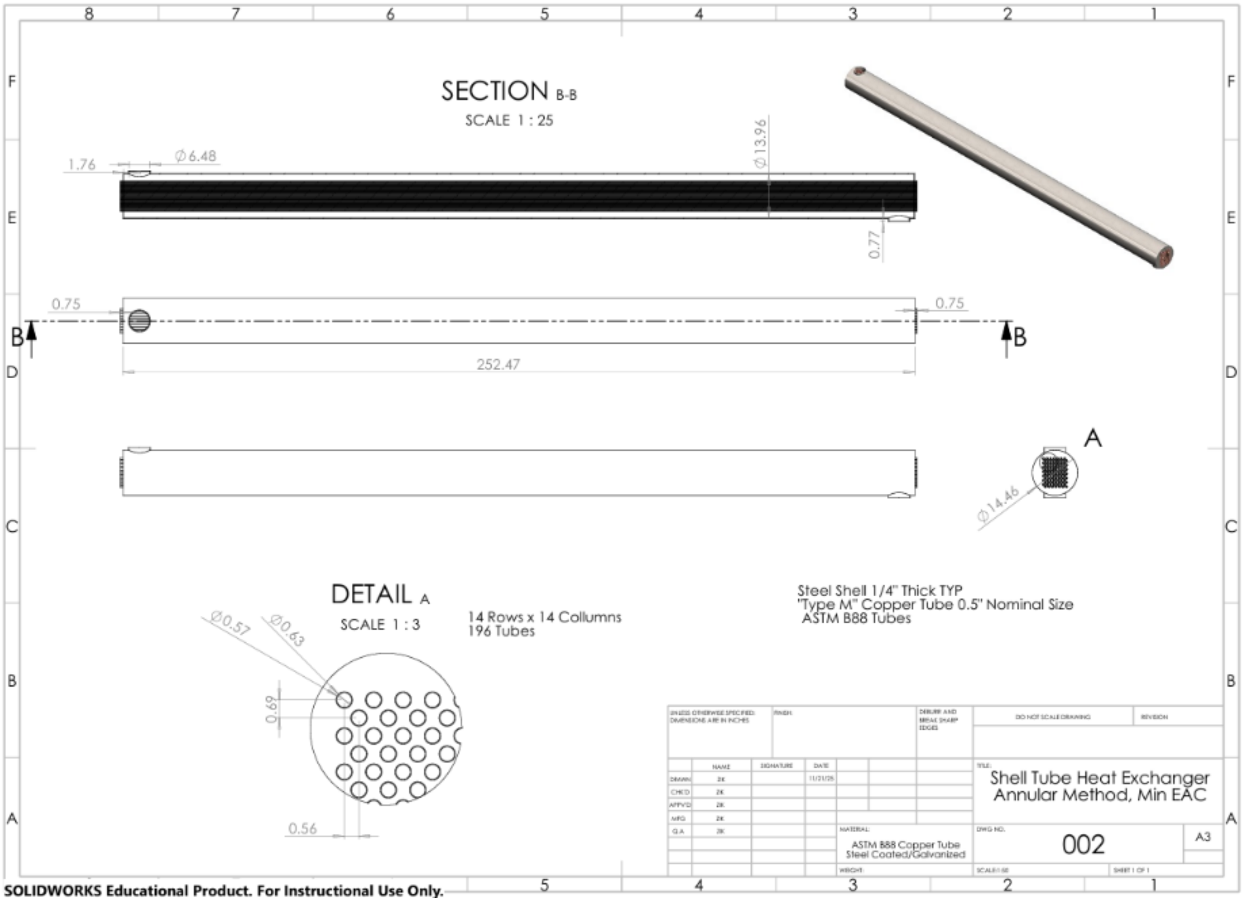


Fig. B.1. Annular CAD Drawing: Staggered

APPENDIX C

TABLE C.1

HEAT TRANSFER EQUATIONS [1]

3.33	$R_{t,cond} = \frac{\ln\left(\frac{r_2}{r_1}\right)}{2\pi L k}$
7.59	$\overline{Nu}_D\rvert_{(N_L<20)} = C_2\overline{Nu}_D\rvert_{(N_L\geq 20)}$
7.60	$Nu_D = 1.13 C_1 Re_D^m Pr^{1/3}$
8.6	$Re_D = \frac{4\dot{m}}{\pi D \mu}$
8.21	$f = (0.790\ln(Re_D) - 1.64)^{-2}$ $3000 \leq Re_D \leq 5 \times 10^6$
8.46a	$q = \overline{U} A_s \Delta T_{lm}$

8.53	$Nu_D \equiv \frac{hD}{k} = 4.36 \quad q''_s = constant$
8.55	$Nu_D = 3.66 \quad T_s = constant$
8.56	$Gz_D \equiv \left(\frac{D}{x}\right) Re_D Pr$
8.58	$\overline{Nu}_D = \dfrac{\dfrac{3.66}{\tanh\left[2.264 Gz_D^{-\frac{1}{3}}\right]} + 0.0499 Gz_D \tanh\left(Gz_D^{-1}\right)}{\tanh\left(2.432 Pr^{\frac{1}{6}} Gz_D^{-\frac{1}{6}}\right)}$ $[T_s = constant, combined\ entry\ length, Pr \geq 1]$
8.62	$Nu_D = \dfrac{\left(\frac{f}{8}\right)(Re_D - 1000)Pr}{1 + 12.7(\frac{f}{8})^{\frac{1}{2}}(Pr^{\frac{2}{3}} - 1)}$
11.5	$\frac{1}{UA} = \frac{1}{h_i A_i} + \frac{R''_{f,i}}{A_i} + \frac{\ln(\frac{D_o}{D_i})}{2\pi k L} + \frac{R''_{f,o}}{A_o} + \frac{1}{h_o A_o}$
11.6b	$q = \dot{m}_h c_{p,h}(T_{h,i} - T_{h,o})$
11.7b	$q = \dot{m}_c c_{p,c}(T_{c,o} - T_{c,i})$
11.15	$\Delta T_{lm} = \dfrac{\Delta T_2 - \Delta T_1}{\ln(\frac{\Delta T_2}{\Delta T_1})}$
11.17	$\Delta T_1 = T_{h,i} - T_{c,o}$ $\Delta T_2 = T_{h,o} - T_{c,i}$
11.18	$q_{max} = C_{min}(T_{h,i} - T_{c,i})$
11.19	$\varepsilon \equiv \frac{q}{q_{max}}$
11.22	$q = \varepsilon C_{min}(T_{h,i} - T_{c,i})$
11.24	$NTU \equiv \frac{UA}{C_{min}}$

TABLE C.2

GNIELINSKI EQUATION [3]

Gnielinski Equation	$Nu_{Dh} = \dfrac{\left(\frac{f}{8}\right)(Re_{Dh} - 1000)Pr}{1 + 12.7(\frac{f}{8})^{\frac{1}{2}}(Pr^{\frac{2}{3}} - 1)}$

TABLE C.3

FLUID MECHANICS EQUATIONS [4]

10-3	$h_L = f\frac{L}{D}\frac{V^2}{2g}$
10-4	$f = \frac{64}{Re}$
10-5	$h_L = f\frac{L}{D}\frac{V^2}{2g}$
14-15	$\dot{W}_{pump} = Q\gamma h_{pump}$

APPENDIX D

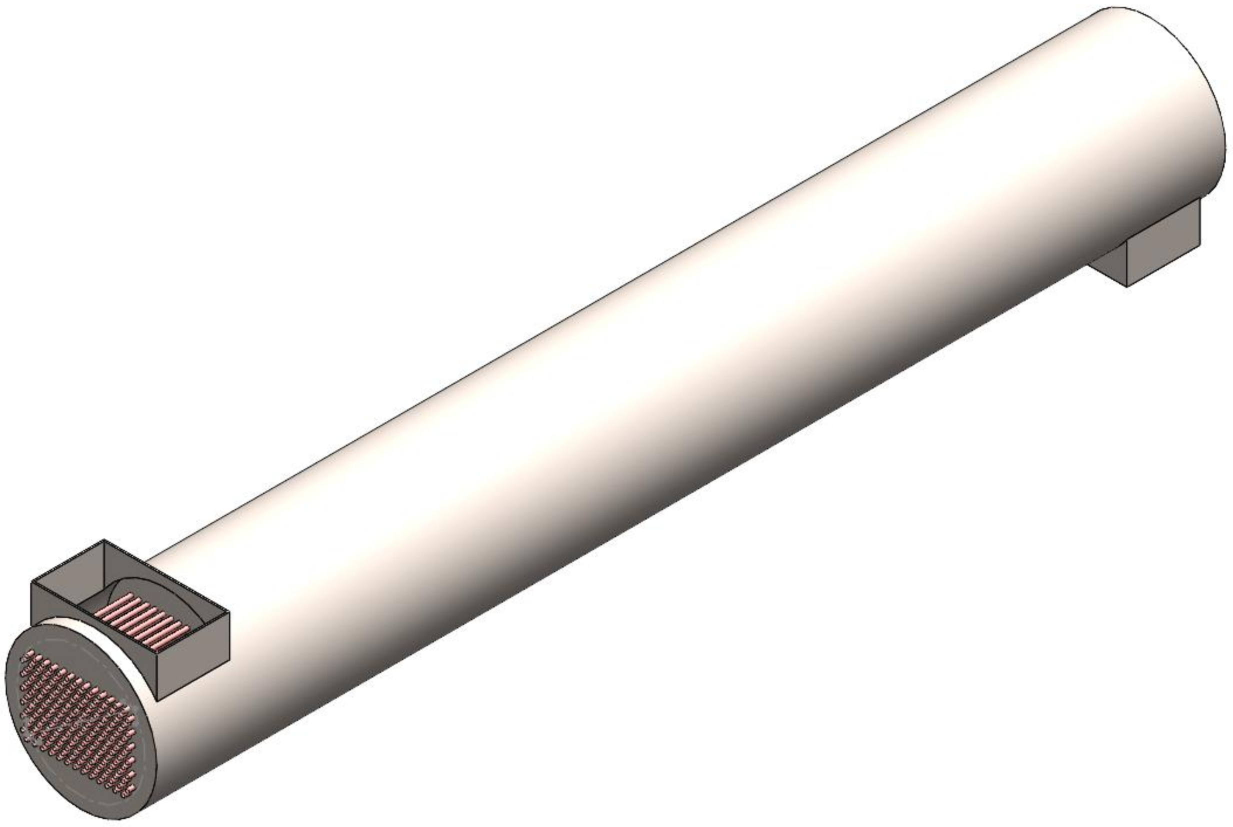


Fig. D.1. 3D Model of Shell and Tube Heat Exchanger

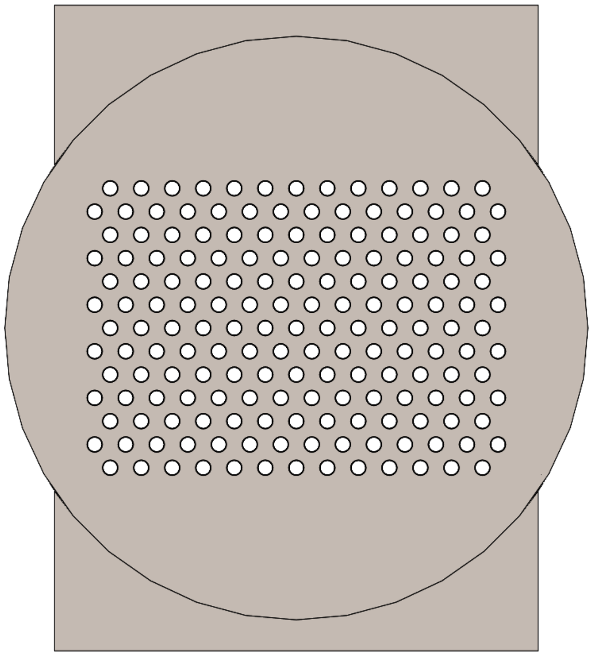


Fig. D.2. 3D Model of Shell and Tube Heat Exchanger (Side View)

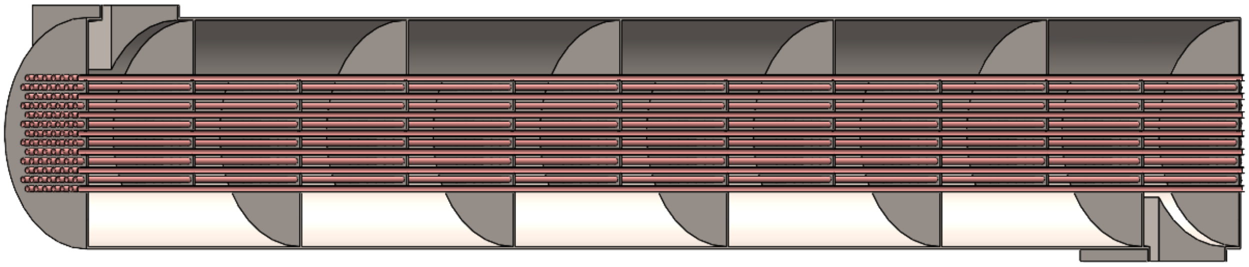


Fig. D.3. Section View of Crossflow Heat Exchanger with Baffles

APPENDIX E

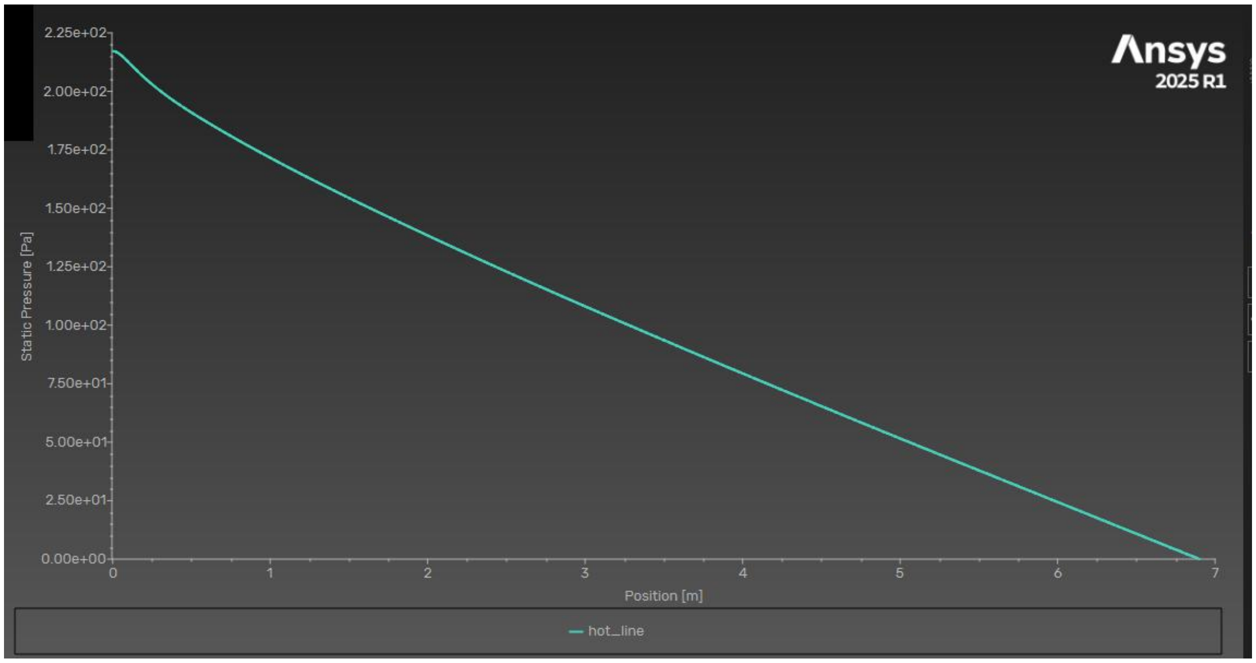


Fig. D.1. Plot of Position vs Static Pressure, Centerline of Hot Fluid

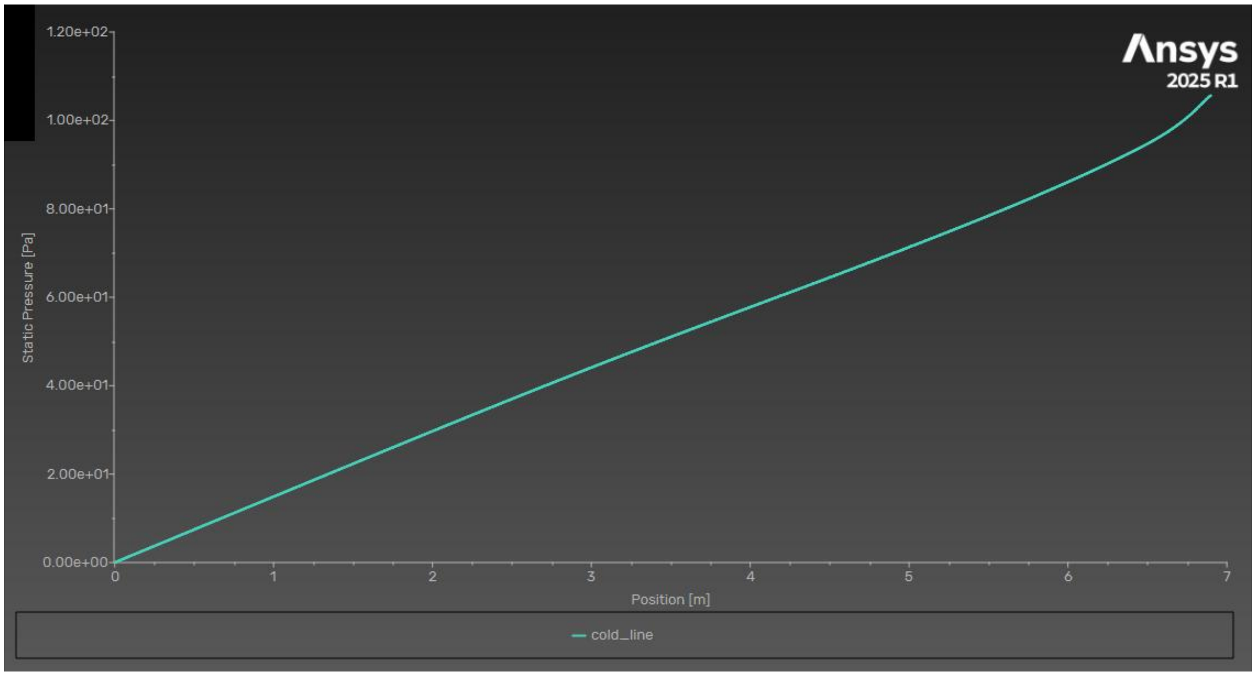


Fig. D.2. Plot of Position vs Static Pressure, Centerline the Gap of Cold Fluid

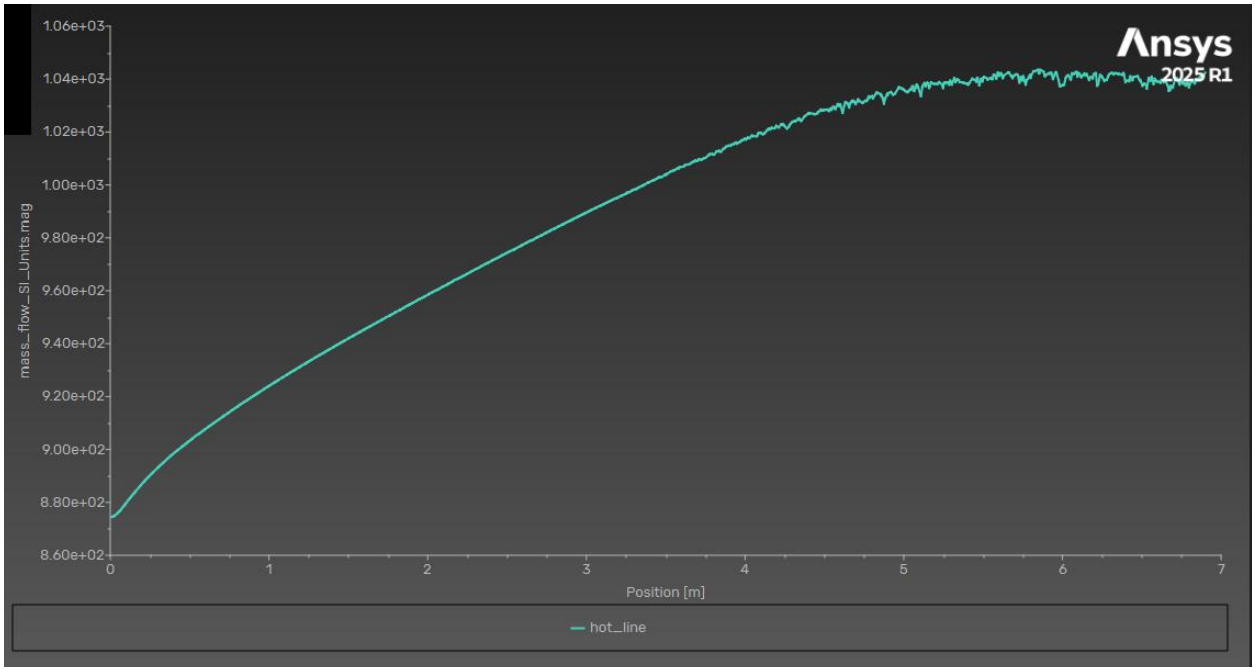


Fig. D.3. Plot of Position vs Mass Flow Rate, Centerline of Hot Fluid

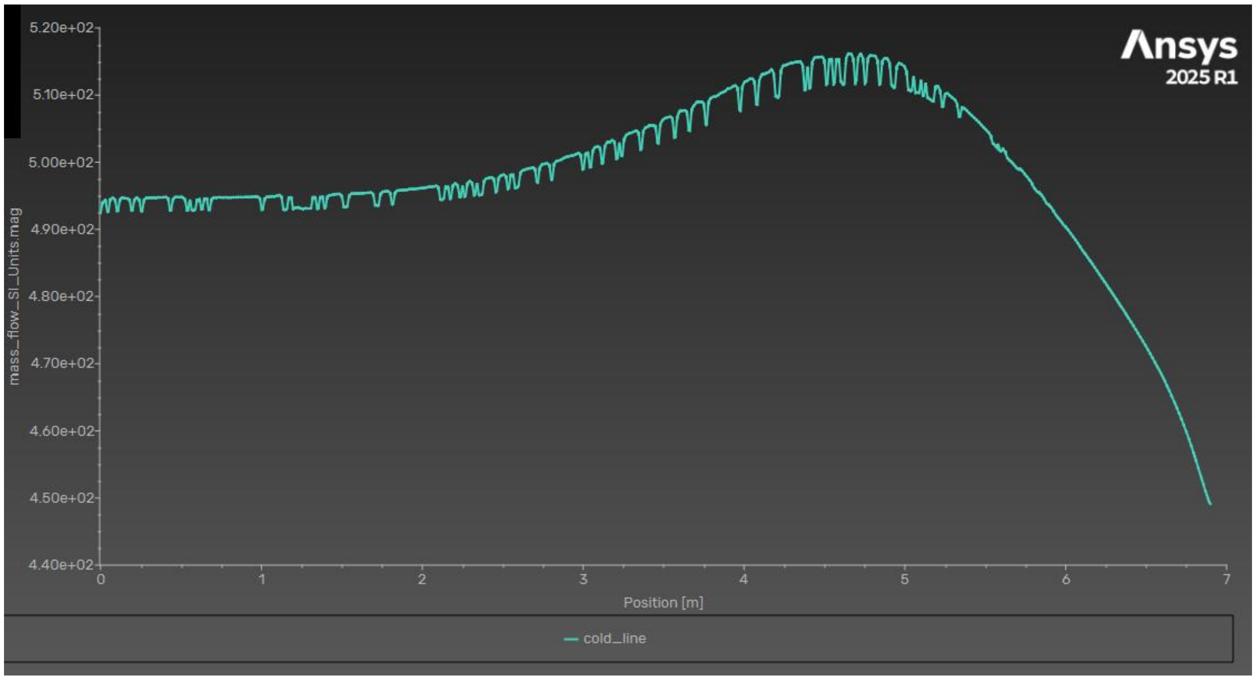


Fig. D.4. Plot of Position vs Mass Flow Rate, Centerline of Gap of Cold Fluid

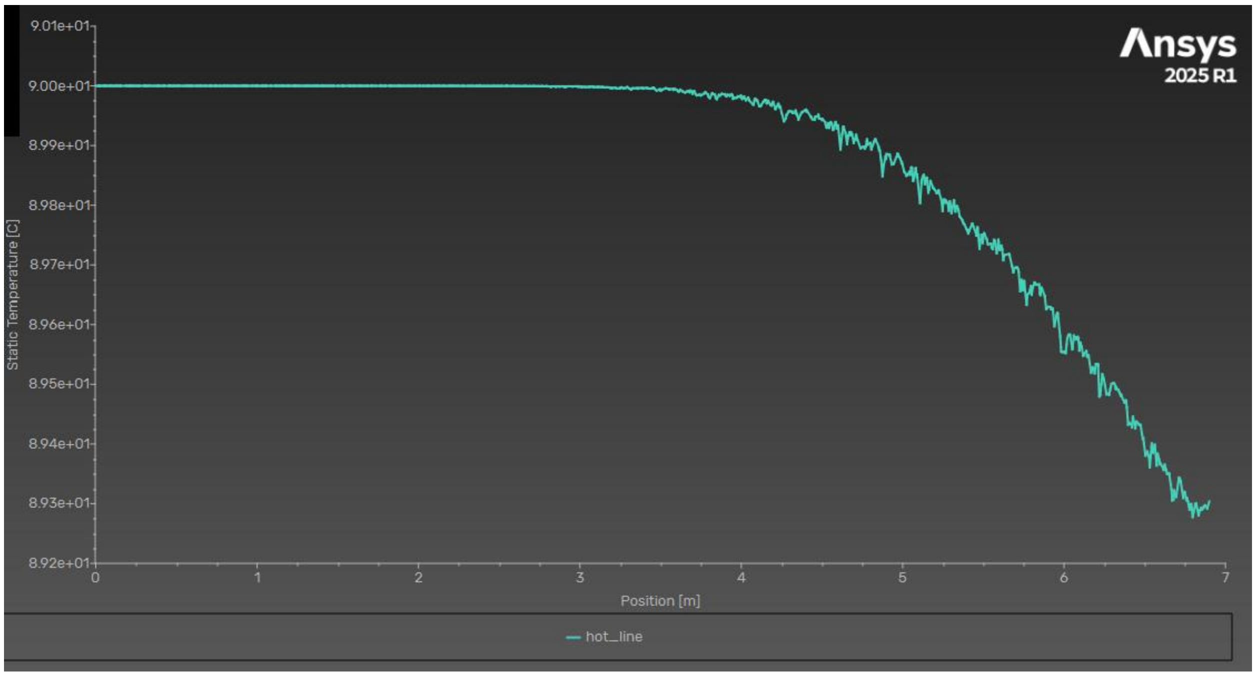


Fig. D.5. Plot of Position vs Static Temperature, Centerline of Hot Fluid

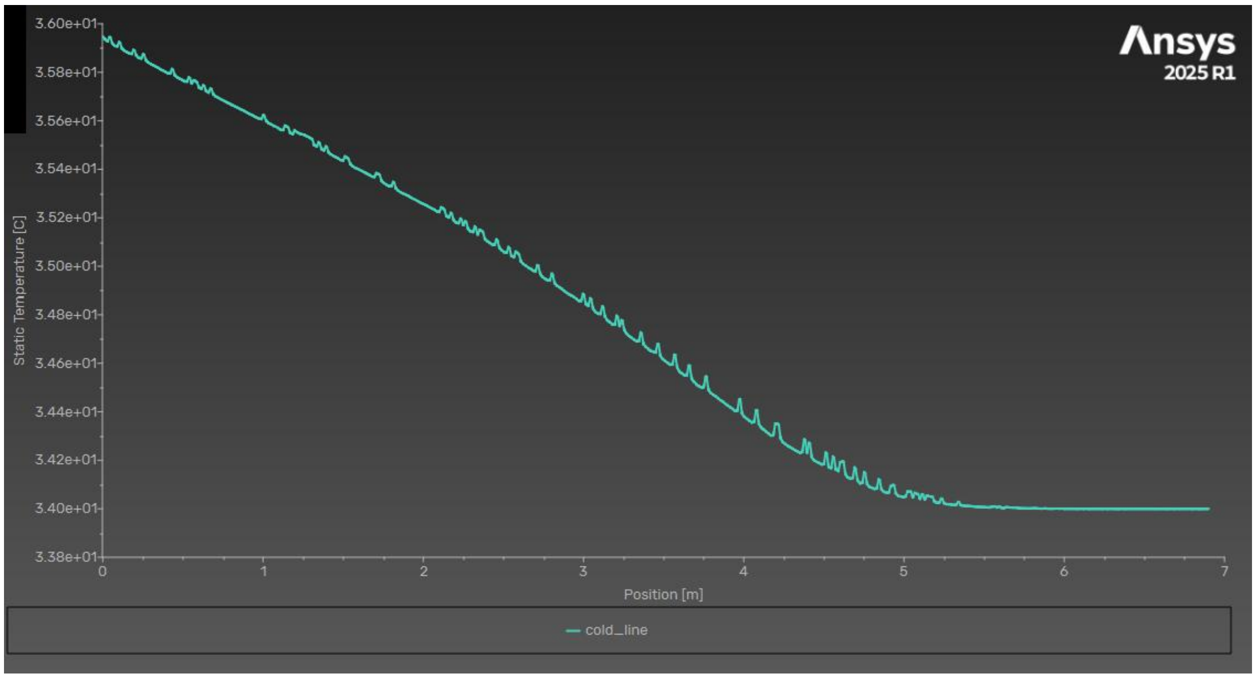


Fig. D.6. Plot of Position vs Static Temperature, Centerline of Gap of Cold Fluid

APPENDIX F

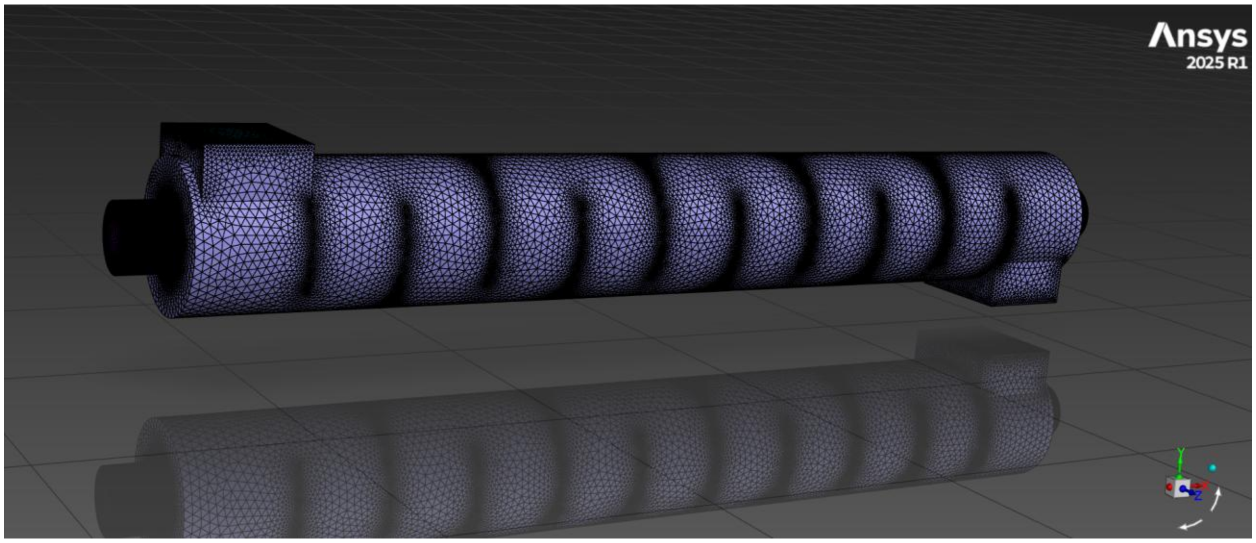


Fig. F.1. ANSYS Mesh of Heat Exchanger with Baffles

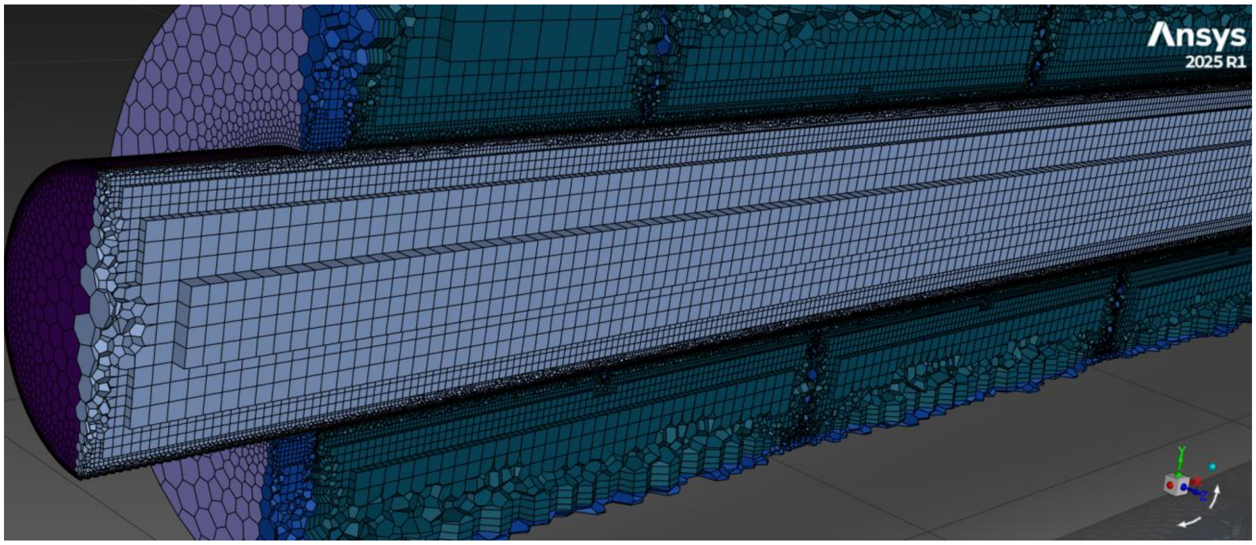
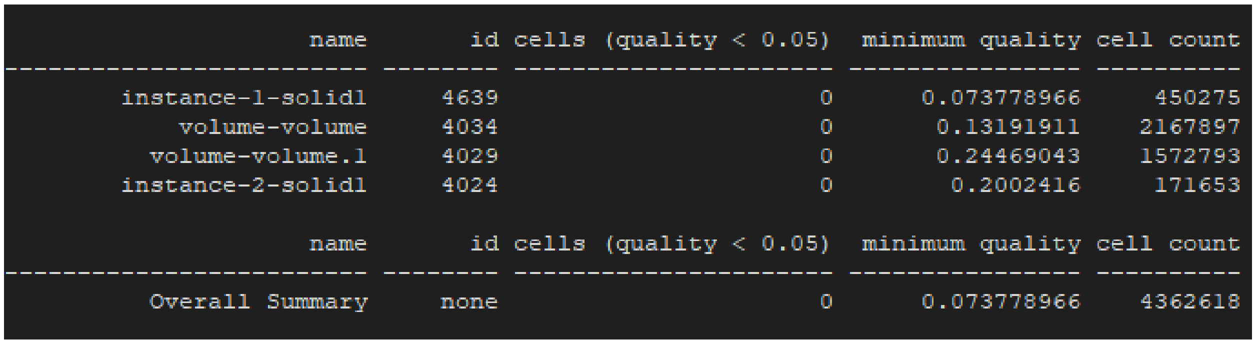


Fig. F.2. ANSYS Cutaway View of Volume Mesh, Annulus with Baffles



```
                    name       id cells (quality < 0.05)  minimum quality cell count
------------------------- -------- ----------------------  ---------------- ----------
        instance-1-solid1     4639                      0       0.073778966     450275
            volume-volume     4034                      0        0.13191911    2167897
          volume-volume.1     4029                      0        0.24469043    1572793
        instance-2-solid1     4024                      0         0.2002416     171653

                    name       id cells (quality < 0.05)  minimum quality cell count
------------------------- -------- ----------------------  ---------------- ----------
          Overall Summary     none                      0       0.073778966    4362618
```

Fig. F.3. ANSYS Volume Mesh Data, Annulus with Baffles

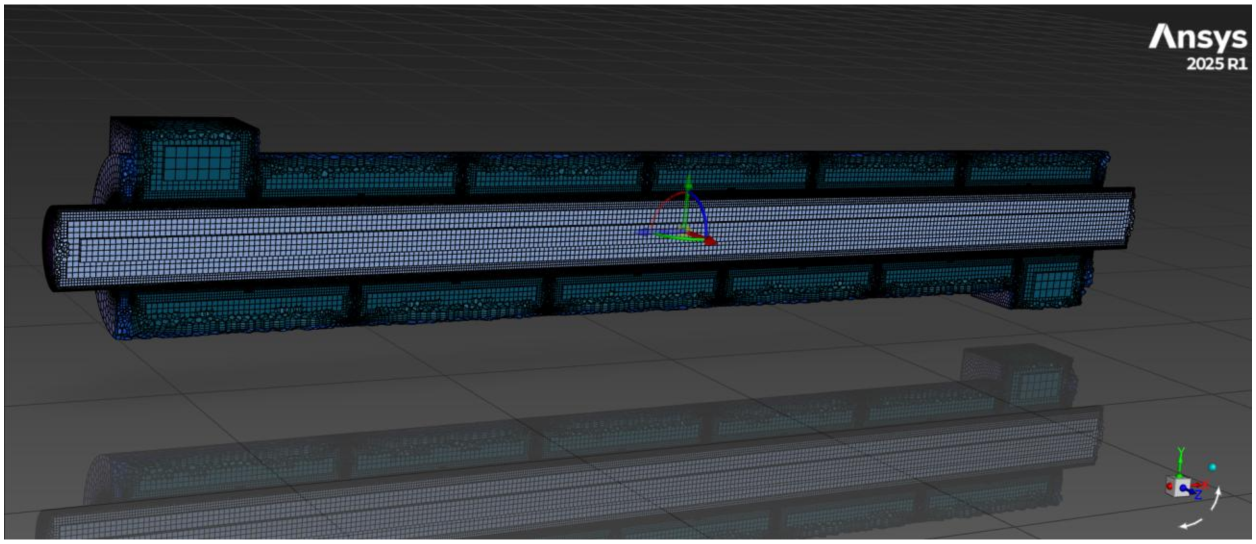


Fig. F.4. Cutaway View of Full Volume Mesh, Annulus with Baffles

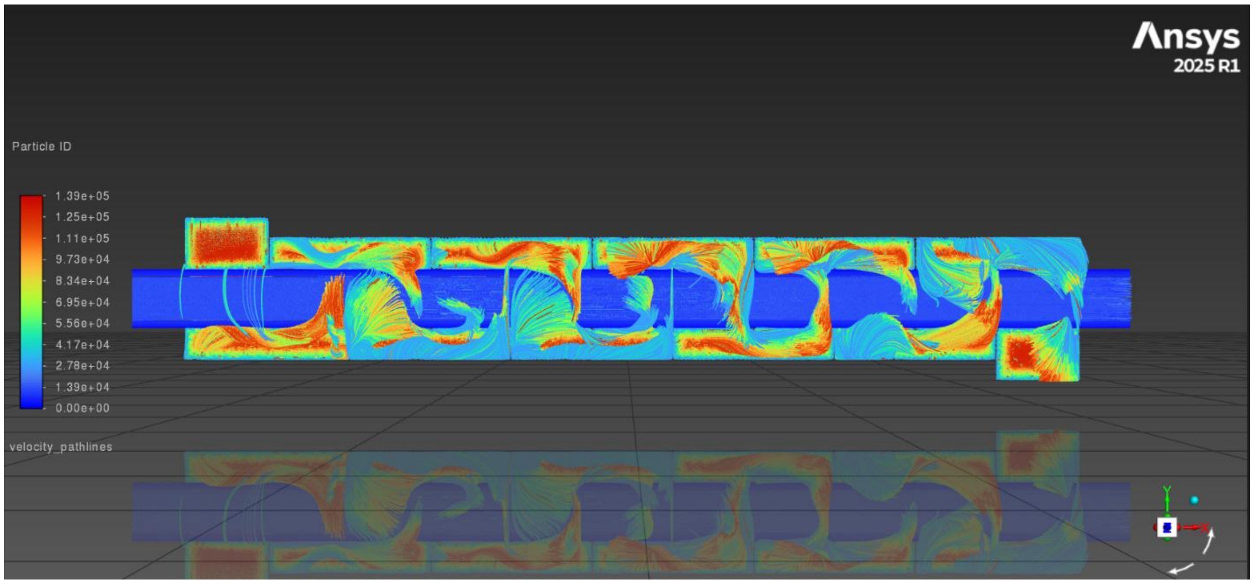


Fig. F.5. Velocity Path Lines, Annulus with Baffles

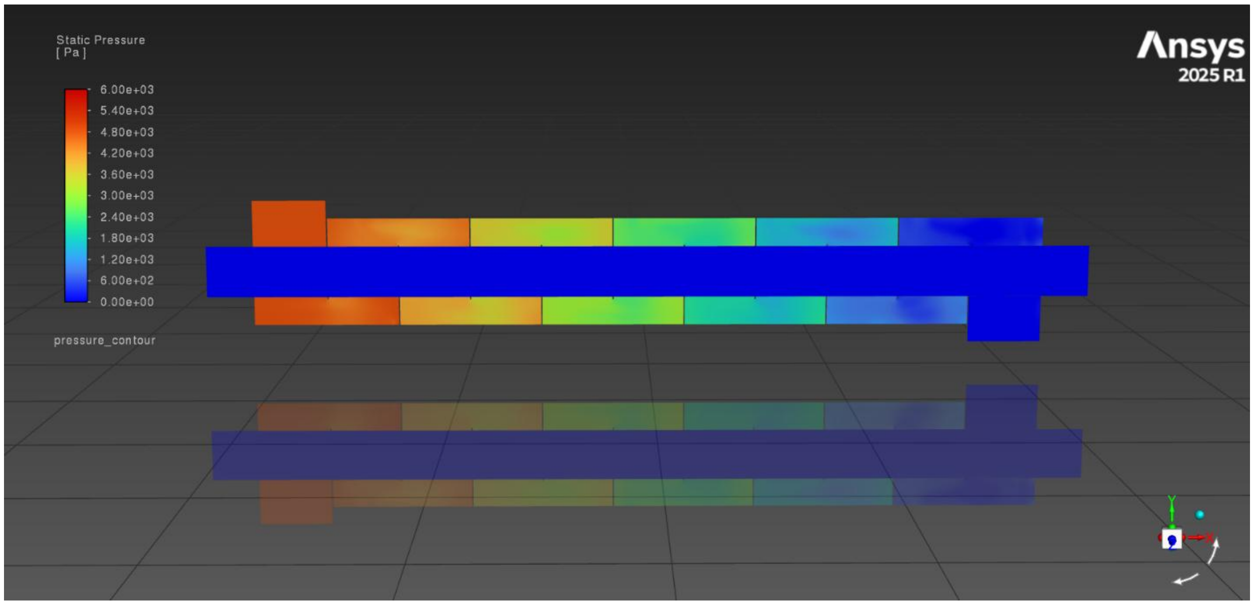


Fig. F.6. Pressure Contour, Annulus with Baffles

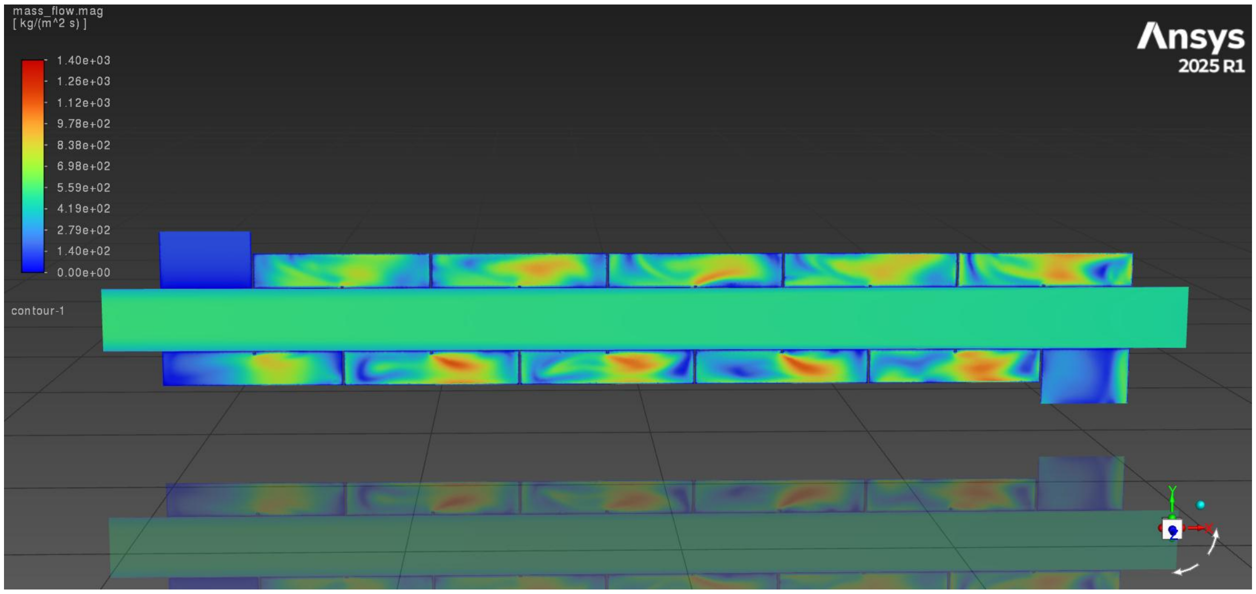


Fig. F.7. Mass Flow within the Baffles

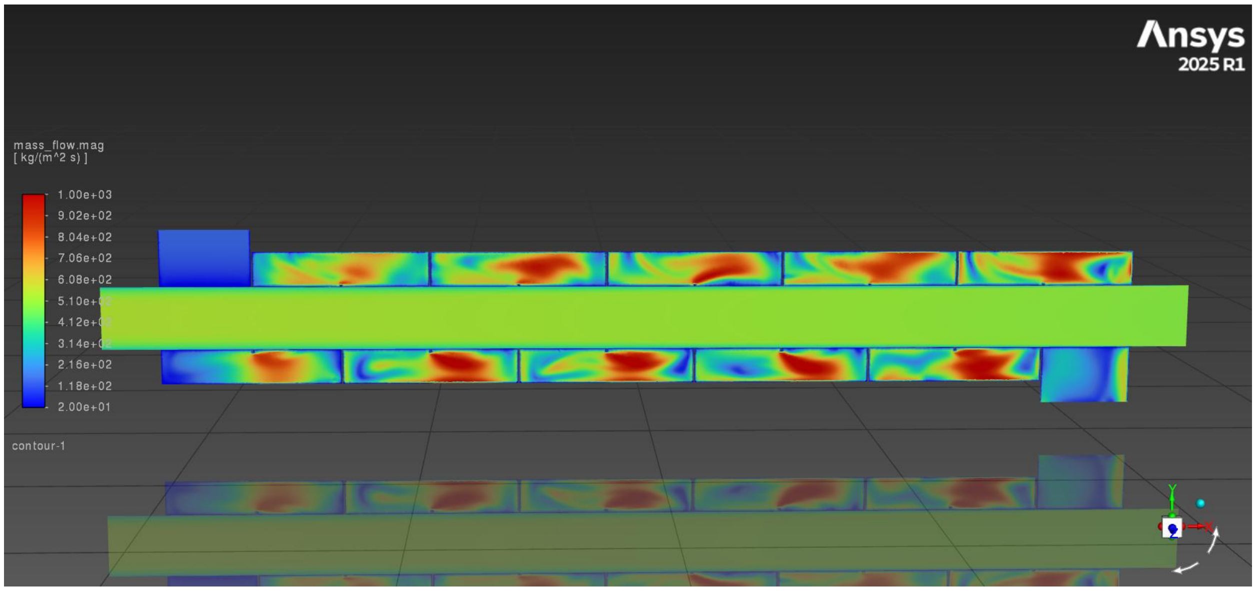


Fig. F.8. Mass Flow within the Baffles, Alternate Scale

APPENDIX G

CODE APPENDIX: MATLAB CALCULATIONS FOR CROSSFLOW WITH BAFFLES

```
clear, clc
%%%%%%%%%%%%%%%%%%%%%
%%   ASSUMPTIONS   %%
%%%%%%%%%%%%%%%%%%%%%

    % 1. Steady state
    % 2. No heat is lost to surroundsings, i.e. shell is “well insulatied”
    % 3. When instructions say “working fluid is water” they mean both
    % fluids must be water
    % 4. Hot water is inside the tubes while cold water is in the shell
    % (see figure on page 3 of instructions)
    % 5. If flow inside pipes is laminar, constant surface temperature is
    % the chosen boundary condition because it is more conservative
    % 6. Surface roughness of the drawn copper tubing can be approximated
    % as smooth
    % 7. Pressure drop across tube bank is approximated by pressure drop
    % across equally spaced tube bank
    % 8. Interest rate of 8%, exchagner life span of at least 20 years

% from page 677
    % 1. Well insulated
    % 2. Axial conduction along tube is negligable
    % 3. PE and KE change is negligible
    % 4. Cp is constant
    % 5. U (and UA) is constant

% simplifications
```

```
    % 1. Baffles allow for crossflow over tubes
    % 2. Counter flow is generally more efficient than parallel flow, so we
    % will only consider counter flow
    % 3. Staggered banks are generally more efficient than straight banks,
    % so we will only consider staggered banks (due to turbulence)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   GIVENS AND PROJECT CONSTRAINTS   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mdot_h = 28;        %[kg/s]
mdot_c = 27;        %[kg/s]
Thi = 90+273;       %[C]
Tci = 34+273;       %[C]
Tco = 60+273;       %[C]
U_max = 1700;       %[W/m^2K]
U_min = 850;        %[W/m^2K]
L_max = 10;         %[m]
L_min = 1;          %[m]

% ### Equation numbers are from physical book (6th edition) unless other
% wise noted ###

%%%%%%%%%%%%%%%%
%%   COPPER   %%
%%%%%%%%%%%%%%%%

% copper tubing size and pricing (type M)
% see: “McMaster Carr, Low-Pressure Copper Tubing for Drinking Water”

% tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
% can us the k value for pure copper

%    index:  1        2         4         5        6         7         8
9       10
%  ID [in]:  3/8      ½       ¾        1         1 ¼      1 ½     2          2 ½
3
cop_tubeM = [0.01143  0.0144526 0.0205994 0.026797 0.0327914 0.0387858
0.0510286 0.063373 0.0757174;   %[m] ID
             0.000635 0.0007112 0.0008128 0.000889 0.0010668 0.0012446
0.0014732 0.001651 0.0018288;   %[m] wall thickness
             30.30    28.13     45.29     69.35    128.88    177.12    272.57
402.14   533.86];     %[$] Cost of 10’

% copper tubing size and pricing (type L)
% see: “McMaster Carr, Medium-Pressure Copper Tubing for Drinking Water”

% tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
% can us the k value for pure copper

%    index:  1        2        3        4         5        6        7        8
9        10       11
```

```
 %  ID [in]:  ¼      3/8      ½      5/8       ¾      1        1 ¼    1 ½    2
2 ½    3
 cop_tubeL = [0.008001 0.010922 0.013843 0.0169164 0.019939 0.026035 0.032131
0.038227 0.050419 0.062611 0.074803;   %[m] ID
              0.000762 0.000889 0.001016 0.0010668 0.001143 0.00127  0.001397
0.001524 0.001778 0.002032 0.002286;   %[m] wall thickness
              26.27    38.38    37.81    74.21     70.19    103.30   163.59
207.85   294.53   450.57   629.00];    %[$] Cost of 10'

 % copper tubing size and pricing (type K)
 % see: "McMaster Carr, High-Pressure Copper Tubing for Drinking Water"

 % tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
 % can us the k value for pure copper

 %    index:  1        2         3         4         5        6        7
8         9         10       11
 %  ID [in]:  ¼      3/8       ½       5/8       ¾      1        1 ¼    1 ½
2         2 ½    3
 cop_tubeK = [0.007747 0.0102108 0.0133858 0.0165608 0.018923 0.025273 0.031623
0.0376174 0.0497586 0.061849 0.0738378;   %[m] ID
              0.000889 0.0012446 0.0012446 0.0012446 0.001651 0.001651 0.001651
0.0018288 0.0021082 0.002413 0.0027686;   %[m] wall thickness
              30.18    53.62     62.95     77.55     116.54   151.31   191.83
252.57   371.14    554.86   765.29];     %[$] Cost of 10'

 cop_tube = cat(2, cop_tubeM, cop_tubeL, cop_tubeK)
```

```
cop_tube = 3×31

    0.0114    0.0145    0.0206    0.0268    0.0328    0.0388    0.0510 ⋯
    0.0006    0.0007    0.0008    0.0009    0.0011    0.0012    0.0015
   30.3000   28.1300   45.2900   69.3500  128.8800  177.1200  272.5700
```

```
 % tables values for pure copper, estimated at 300K
 kc = 401;  %[W/Mk]

 %%%%%%%%%%%%%%
 %%   STEEL   %%
 %%%%%%%%%%%%%%
 % assume rolling ¼" sheet and weld
 % 1388K185
 steel_L = 1.8288;                   %[m]
 steel_W = 0.6096;                   %[m]
 steel_cost = 746.78;                %[USD]
 steel_area = steel_L * steel_W;     %[m^2]
 t_steel = 0.00635;                  %[m]

 %%%%%%%%%%%%%%%%%
 %%   FOULING   %%
 %%%%%%%%%%%%%%%%%
 % fouling, table 11.1
```

```
Rfi = 0.0002;   %[m^2 K/W] because Tmi > 50C
Rfo = 0.0001;   %[m^2 K/W] because Tmi < 50C

%%%%%%%%%%%%%%%%
%%   THERMO   %%
%%%%%%%%%%%%%%%%
% find mean temperatures and table values
% GUESS Tho (iterated to balance qh and qc) 65.0206 C
Tho_g = 65.0206+273;        %[K]
Tmc = (Tco+Tci)/2;          %[K]
Tmh = (Tho_g+Thi)/2;        %[K]

[~, cp_c, ~, ~, ~] = satH2Otable(Tmc);
[~, cp_h, ~, ~, ~] = satH2Otable(Tmh);

% energy must be conserved, so, q_cold = q_hot
q = mdot_c * cp_c * (Tci-Tco);  %[W]
Tho = q/(mdot_h * cp_h) + Thi;  %[K]
% check this against Tho_g
Tho - 273    %[C]
```

```
ans = 65.0206
```

```

%get mean temperatures
Tmc = (Tco+Tci)/2;
Tmh = (Tho+Thi)/2;
[rho_h, cp_h, mu_h, kf_h, Pr_h] = satH2Otable(Tmh);
[rho_c, cp_c, mu_c, kf_c, Pr_c] = satH2Otable(Tmc);

%%%%%%%%%%%%%%%%%%%
%%   ITERATION   %%
%%%%%%%%%%%%%%%%%%%

% Storage
best_efficiency = 10e6;
best_design = [];

% VARIABLES TO ITERATE
% ST scalar max is ~2.99 min is ~1.05
% SL scalar max is ~2.99 min is ~1
ST_ratios = [1.25, 1.5, 2, 3];
%ST_ratios = 1.05:0.05:2.99;
SL_ratios = [0.6, 0.9, 1, 1.125, 1.25, 1.5, 2, 3];
%SL_ratios = 1.05:0.05:2.99;
Rows = 5:1:40;
Cols = 5:1:40;
Lengths = 2:0.1:10;
Tube_Idxs = 1:length(cop_tube);
```

```
 permutations = 
length(ST_ratios)*length(SL_ratios)*length(Rows)*length(Cols)*length(Lengths)*le
ngth(Tube_Idxs);
 fprintf("Analyzing %d permutations... ", permutations);
```

```
Analyzing 104136192 permutations...
```

```
 counter = 0;

 for t_idx = Tube_Idxs
     Di = cop_tube(1, t_idx);
     t_wall = cop_tube(2, t_idx);
     Do = Di + 2*t_wall;
     cost_ten_ft = cop_tube(3, t_idx);
     for L = Lengths
         for n_r = Rows
             for n_c = Cols
                 n_tubes = n_r * n_c;
                 % code optimization, if area is less than 20 m^2 don't
                 % bother calculating (would force U to be greater than
                 % 1700, not possible)
                 area_approx = pi*Do*L*n_tubes;
                 if area_approx < 20; continue; end

                 for st_r = ST_ratios
                     for sl_r = SL_ratios
                         counter = counter + 1;

                         ST = st_r*Do;
                         SL = sl_r*Do;
                         if ST < 1.05*Do; continue; end
                         diag_pitch = sqrt((ST/2)^2 + SL^2);
                         if (diag_pitch < 1.05*Do); continue; end
```

Flow over the bank of tubes

```
                             % calculate tube bank height and width
                             bank_h = (1+n_r)*(Do) + (n_r-1)*(ST-Do) + (ST/2 –
Do);  %[m]
                             bank_w = n_c*Do + (n_c-1)*(SL-
Do);                      %[m]

                             % calculate shell diameter based on tube bank
dimensions,
                             % bank 'diameter' +10%
                             min_shell_D = sqrt(bank_h^2 +
bank_w^2);                    %[m]
                             Di_shell =
min_shell_D*1.1;                                %[m]
```

```
                              A_Shell =
pi*(Di_shell/2)^2;                                    %[m^2]
                              baffle_spacing =
Di_shell*(2/3);                                %[m]
                              n_baffles = (L/baffle_spacing)-
2;                            %[]
                              if n_baffles < 0
                                  n_baffles = 0;
                              end


                              % Ratio of “open gap” to “total width” is (ST – Do)
/ ST
                              A_inlet_c = baffle_spacing * Di_shell * ((ST – Do)
/ ST);  %[m^2]
                              u_o_max = mdot_c ./ (rho_c *
A_inlet_c);                    %[m/s]

                              % find reynolds number for external flow across
bank eqn 7.59
                              Re_oDmax = (rho_c*u_o_max*Do)/mu_c;          %[]

                              % get constants from table 7.5
                              [C1, m] = table7_5(ST, SL, Do);         %[]
                              % get out of loops if there is no corelation
                              % for geometry
                              if C1 == 0; continue; end

                              %get correction factor from table 7.6
                              C2 = table7_6(n_r);                     %[]

                              % eqn 7.60, find nussel and ho
                              if (Re_oDmax >= 2000 && Re_oDmax <= 40000 && Pr_c
>= 0.7)
                                  NuDo = (1.13*C1*(Re_oDmax^m)*(Pr_c^(1/3))) *
C2;   %[]
                              else
                                  NuDo = 0;
                              end

                              if NuDo == 0; continue; end
                              % this is the convection coefficent for the *entire
tube bank*
                              ho = NuDo*(kf_c/Do);    %[W/m^2 K]

                              % find friction factor
                              [c0, c1, c2, c3, c4] = eulerPolynomial(ST/Do,
Re_oDmax);

                              % Euler number calculates pressure drop per row
                              Eu = c0/Re_oDmax^0 + c1/Re_oDmax^1 + c2/Re_oDmax^2
+ c3/Re_oDmax^3 + c4/Re_oDmax^4;
```

```
                             dPo = (n_baffles+1) * n_r *
Eu*0.5*rho_c*u_o_max^2;   %[Pa]
```

Flow inside of a single tube

```
                             % find reynolds number for flow inside a single
tube
                             mdot_h_single = mdot_h/n_tubes;          %[kg/s]
                             Di = cop_tube(1, t_idx);                 %[m]
                             Re_Id = (4*mdot_h_single)/(pi*Di*mu_h); %[]

                             % friction factor
                             % fluids textbook: “drawn tubing = 0.0015mm”
                             % engineers toolbox: “drawn copper... = 0.0.001-
0.002mm”
                             % heat transfer textbook: “drawn tubing = 0.0015mm”
                             % for the smallest pipe ¼, e/D = 0.0002 which is a
good approximation for
                             % reynolds numbers below 10e6
                             % for normal pipe diameters of 1”, e/D = 0.00001
which is a very good
                             % approximation!
                             % thus, we can be confident that our error from
assuming a smooth pipe is
                             % less than the ~15% error difference in using
equation 8.62 compared to
                             % 8.60

                             ReDc = 2300;        %[] onset of turbulence

                             if (Re_Id >= ReDc && Re_Id <= 5e6)
                                 f = (0.79*log(Re_Id) – 1.64)^-2;
                             elseif (Re_Id <= ReDc)
                                 f = 64/Re_Id;
                             else
                                 f = 0;
                             end

                             % set up if structure to filter corelations
                             if (Re_Id <= 2300)
                                 Gz = (Di/L)*Re_Id*Pr_h;
                                 NuDi = (3.66/(tanh(2.264*Gz^(-1/3) + 1.7*Gz^(-
2/3))) + 0.0499*Gz*tanh(Gz^-1))  /  (tanh(2.432*Pr_h^(1/6)*Gz^(-1/6))); %[]
                                 % assumes constant Ts, more conservative
assumption, eqn 8.58 in online book
                                 % Pr will always be > 0.1 for water, combined
entry length problem
                             elseif (Re_Id > 2300)
                                 NuDi = ((f/8)*(Re_Id-1000)*Pr_h)/(1 +
12.7*(f/8)^(1/2)*(Pr_h^(2/3)-1)); %[] eqn 8.62
                             end
                             hi = NuDi*(kf_h/Di);    %[W/m^2 K] for all tubes

                             % claculate pressure drop, from fluids text
                             % book, eqn 10-3 / 10-5
                             u_i_max =
mdot_h_single/(rho_h*(pi*(Di/2)^2));  %[m/s]
                             Hl = f*(L/Di)*(u_i_max^2/(2*9.81)); %[m]
                             Dp_single = Hl*(rho_h*9.81);        %[Pa]
                             dPi = Dp_single; %[Pa] because tubes are in
parallel!
```

Wall Resistance, Fouling, Overall Heat Transfer Coefficient

```
                             % wall resistance and conduction
                             Rw_single = log((Do/2) / (Di/2)) /
(2*pi*kc*L);  %Rw = ln(ro/ri)/(2*pi*k*L)
                             Rw = Rw_single/n_tubes;                        %
resistance is in parallel

                             % surface area
                             Sai = pi*Di*L*n_tubes;   %[m^2]
                             Sao = pi*Do*L*n_tubes;   %[m^2]

                             % U calculations, eqn 11.5 and similar
                             Ui = (1/(hi*Sai) + Rfi/(Sai) + Rw + Rfo/(Sao) +
1/(ho*Sao))^-1 / (Sai);      %[W/m^2K]
                             Uo = (1/(hi*Sai) + Rfi/(Sai) + Rw + Rfo/(Sao) +
1/(ho*Sao))^-1 / (Sao);      %[W/m^2K]
                             U = (1/hi + Rfi + log(Do/Di)*(Di/(2*kc)) + Rfo +
1/ho)^-1;                   %[W/m^2K]
                             Um =
mean([Ui,Uo]);                                                  %[W/m^2K
]

                             % find log mean temperature difference 11.17 and
11.15

                             DT1 = Thi-Tco;                 %[K]
                             DT2 = Tho-Tci;                 %[K]
                             LMTD = (DT2-DT1)/log(DT2/DT1);  %[K]

                             % the calculated heat the exchanger can move
                             q_ht = Ui*Sai*LMTD;            %[W]

                             % check U
                             if U <= U_min; continue; end
                             if U >= U_max; continue; end
```

Finnancial Considerations

```
                             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                             %%   CALCULATE MATERIAL COST   %%
```

```
                              %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                              cop_mat_cost =
n_tubes*cop_tube(3,t_idx)*((L*3.28084)/10);  %[USD]
                              % from CAD, assume baffle takes up 80% of shell
                              % face area. Baffle spacing is 2/3 Di_shell. Area
where
                              % copper tubes pass through is not considered
                              % (don't pay for it)
                              baffles = (n_baffles)*((pi*(Di_shell/2)^2) -
n_tubes*(pi*(Do/2)^2))*0.8 + 2*(pi*(Di_shell/2)^2);      %[m^2]
                              steel_mat_cost = (((Di_shell*pi*L)+baffles) /
(steel_area)) * steel_cost; %[USD]
                              mat_cost = cop_mat_cost +
steel_mat_cost;                                  %[USD]

                              %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                              %%   CALCULATE PUMPING COST   %%
                              %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                              % from fluids book, derived from equation 14-15
(general
                              % hydraulic power equation) knowing that
gamma*h_pump = Dp
                              % assume pump efficiency of 0.73 (from JCI
                              % experience)
                              pump_power = ((mdot_h/rho_h)*dPi +
(mdot_c/rho_c)*dPo)/(1000*0.73);    %[Kw]
                              hours_in_a_year =
24*365;                                            %[h]
                              yearly_power =
pump_power*hours_in_a_year;                           %[kWh]
                              avg_kwh_cost =
0.1583;                                               %[USD/kWh]
                              pump_cost_yr =
yearly_power*avg_kwh_cost;                            %[USD]

                              %%%%%%%%%%%%%%%%%%%%%%%
                              %%   SALVAGE VALUE   %%
                              %%%%%%%%%%%%%%%%%%%%%%%

                              cop_scrap = 7.70;       %[USD/kg] $3-4 per lb
                              steel_scrap = 0.358274; %[USD/kg] $0.16 per lb
                              rho_cop = 8933;         %[kg/m^3]
                              rho_steel = 7854;       %[kg/m^3]

                              cop_salvg = (n_tubes*L*((pi*(Do/2)^2)-
(pi*(Di/2)^2)))*rho_cop*cop_scrap; %[USD]
                              steel_salvg = (L*((pi*((Di_shell+2*t_steel)/2)^2)-
(pi*(Di_shell/2)^2)) + (n_baffles)*t_steel*((pi*(Di_shell/2)^2) -
n_tubes*(pi*(Do/2)^2))*0.8 + 2*0.00635*(pi*(Di_shell/2)^2)) * rho_steel *
steel_scrap; %[USD]
```

```
                              salvg = cop_salvg + steel_salvg;    %[USD]

                              %%%%%%%%%%%%%%%%%%%%%%%
                              %%   ECON ANALYSIS   %%
                              %%%%%%%%%%%%%%%%%%%%%%%
                              length = 0.08;        %[] discount rate
                              N = 20;          %[yrs] analysis horizon (lifespan
of heat exchanger)

                              % finding present value of operating cost and
                              % salvage value
                              PV_Op = pump_cost_yr*((1-(1+i)^-N)/i);  %[USD]
                              PV_sal = salvg/((1+i)^N);               %[USD]

                              % new present value
                              NPV = mat_cost + PV_Op – PV_sal;        %[USD]

                              % equivalent annual cost
                              CRF = (i*(1+i)^N)/((1+i)^N-1);          %[USD]
                              EAC = NPV*CRF;                          %[USD]

                              %%%%%%%%%%%%%%%%%%%%
                              %%   EFFICIENCY   %%
                              %%%%%%%%%%%%%%%%%%%%

                              q_thermo = -q;
                              if (q_ht >= q_thermo && (q_ht/q_thermo) < 1.1)
                                  Nusd = q_ht/(1000*EAC);        %[Kw/USD]
                                  if EAC < best_efficiency
                                      best_efficiency = EAC;
                                      if (t_idx <= 10)
                                          str_t_idx = “Type M”;
                                      elseif (t_idx >= 11 && t_idx <= 21)
                                          str_t_idx = “Type L”;
                                      elseif (t_idx >= 22 && t_idx <= 32)
                                          str_t_idx = “Tpye K”;
                                      else
                                          fprintf(“Error: tube index out of
bounds”);
                                      end

                                      best_Nusd = Nusd;        %[Kw/USD]
                                      design = [L, n_r, n_c, cop_tube(1,
t_idx)*39.3700787, st_r, sl_r, n_tubes];
                                      results = [q_ht, Ui, Sai, U, q_ht/q_thermo,
hi, ho, Sai, Sao, Rw];
                                      geometry = [Di*39.3700787, Do*39.3700787,
Di_shell*39.3700787, n_baffles, baffle_spacing*39.3700787];
                                      economic = [NPV, PV_Op, PV_sal, EAC, N,
length, steel_mat_cost, cop_mat_cost, pump_cost_yr];
                                      report = [Dp_single, pump_power];
```

```
                                end
                            end
                        end
                    end
                end
            end
        end
        if (mod(t_idx, 4) == 0 || t_idx == 1)
            fprintf("Finished tube index %d\n", t_idx);
        end
    end
```

```
Finished tube index 1

Finished tube index 4

Finished tube index 8

Finished tube index 12

Finished tube index 16

Finished tube index 20

Finished tube index 24

Finished tube index 28
```

```
fprintf("Analyzed %d permutations", permutations)
```

```
Analyzed 104136192 permutations
```

```
%%%%%%%%%%%%%%%%%
%%   RESULTS   %%
%%%%%%%%%%%%%%%%%
str_design = {"Length [m]"; "Number of Rows"; "Number of Columns"; "Inner Tube Size [in]"; "ST Ratio"; "SL Ratio"; "Number of Tubes"};
str_geom = {"Tube Inner Diameter [in]"; "Tube Outer Diameter [in]"; "Shell Inner Diameter [in]"; "Number of Baffles"; "Baffle Spacing [in]"};
str_results = {"Heat Transfer Capacity [W]"; "Ui [W/m^2K]"; "Surface Area (inner) [m^2]"; "U [W/K]"; "FOS"; "Inner Convection Coefficient (hi) [W/m^2K]"; "Outer Convection Coefficient (ho) [W/m^2K]"; "Surface Area (inner) [m^2]"; "Surface Area (outer) [m^2]"; "Wall Resistance [W/K]"};
str_econ = {"Net Present Value [$]"; "OpEx PV [$]"; "Salvage PV [$]"; "Equivalent Annual Cost [$]"; "Exchanger Lifespan [yrs]"; "Discount Rate []"; "Steel Material Cost [$]"; "Copper Material Cost [$]"; "Pump Cost per Year [$]"};
str_report = {"Pressure Drop Inside A Single Tube [Pa]"; "Total System Pump Power [Kw]"};
```

```
str_report = 2×1 cell
```

	1
1	"Pressure Drop Inside A Single Tube [Pa]"

	1
2	“Total System Pump Power [Kw]”

```
fprintf(“Tube Index: %s\nExchanger Efficiency %.4f [W/USD]”, str_t_idx,
best_Nusd)
```

```
Tube Index: Type M

Exchanger Efficiency 1.3613 [kW/USD]
```

```
tab_design = table(str_design, design’)
```

```
tab_design = 7×2 table
```

	str_design	Var2
1	“Length [m]”	4.2000
2	“Number of Rows”	12
3	“Number of Columns”	29
4	“Inner Tube Size [in]”	0.5690
5	“ST Ratio”	1.5000
6	“SL Ratio”	1
7	“Number of Tubes”	348

```
tab_geom = table(str_geom, geometry’)
```

```
tab_geom = 5×2 table
```

	str_geom	Var2
1	“Tube Inner Diameter [in]”	0.5690
2	“Tube Outer Diameter [in]”	0.6250
3	“Shell Inner Diameter [in]”	23.5569
4	“Number of Baffles”	8.5290
5	“Baffle Spacing [in]”	15.7046

```
tab_results = table(str_results, results’)
```

```
tab_results = 10×2 table
```

	str_results	Var2
1	“Heat Transfer Capacity [W]”	2.9390e+06
2	“Ui [W/m^2K]”	1.4517e+03
3	“Surface Area (inner) [m^2]”	66.3628
4	“U [W/K]”	1.4019e+03
5	“FOS”	1.0016
6	“Inner Convection Coefficient (hi) [W/m^2K]”	4.1877e+03
7	“Outer Convection Coefficient (ho) [W/m^2K]”	5.7862e+03
8	“Surface Area (inner) [m^2]”	66.3628
9	“Surface Area (outer) [m^2]”	72.8941
10	“Wall Resistance [W/K]”	2.5491e-08

```
tab_econ = table(str_econ, economic’)
```

```
tab_econ = 9×2 table
```

	str_econ	Var2
1	“Net Present Value [$]”	2.1197e+04
2	“OpEx PV [$]”	1.8415e+03
3	“Salvage PV [$]”	769.0841
4	“Equivalent Annual Cost [$]”	2.1590e+03
5	“Exchanger Lifespan [yrs]”	20
6	“Discount Rate []”	0.0800
7	“Steel Material Cost [$]”	6.6356e+03
8	“Copper Material Cost [$]”	1.3489e+04

	str_econ	Var2
9	“Pump Cost per Year [$]”	187.5558

```
tab_report = table(str_report, report’)
```

```
tab_report = 2×2 table
```

	str_report	Var2
1	“Pressure Drop Inside A Single Tube [Pa]”	944.5972
2	“Total System Pump Power [Kw]”	0.1353

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   EFFECTIVENESS NTU   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%
C_h = cp_h*mdot_h;     %[W/K]
C_c = cp_c*mdot_c;     %[W/K]

if C_h < C_c
    Cmin = C_h;
else
    Cmin = C_c;
end

if (C_c < C_h)
    q_max = C_c*(Thi-Tci);   %[W]
elseif (C_h < C_c)
    q_max = C_h*(Thi-Tci);   %[W]
else
    fprintf(“Null Solution”)
    q_max = 0;               %[W]
end

eff = results(1)/q_max;                          %[] eqn 11.19
NTU = (results(2)*results(3)) / Cmin;            %[] eqn 11.24
fprintf(“Heat Exchanger Effectiveness: %.2f\nHeat Exchanger NTU: %.2f”, eff,
NTU)
```

```
Heat Exchanger Effectiveness: 0.47

Heat Exchanger NTU: 0.85
```

```
%%%%%%%%%%%%%%%%%%%%%%%%
%%   ANNULUS METHOD   %%
%%%%%%%%%%%%%%%%%%%%%%%%
% section 8.6 in book, table 8.2
```

```
 % assumes fully laminar, fully developed
 % trash assumptions
 gap = sqrt((design(5)*geometry(2)/39.3700787)^2 +
(design(6)*geometry(2)/39.3700787)^2) - geometry(2)/39.3700787;
 Di_ann = geometry(2)/39.3700787;
 Do_ann = geometry(2)/39.3700787+2*gap
```

```
Do_ann = 0.0414
```

```
 DiDo = Di_ann/Do_ann;

 if (DiDo < 0.075)
     NuDi_ann = 17.46;
 elseif (DiDo >= 0.075 && DiDo < 0.175)
     NuDi_ann = 11.56;
 elseif (DiDo >= 0.175 && DiDo < 0.375)
     NuDi_ann = 7.37;
 elseif (DiDo >= 0.375 && DiDo < 0.75)
     NuDi_ann = 5.74;
 elseif (DiDo >= 0.75 && DiDo < 1)
     NuDi_ann = 4.86;
 end

 Dh = Do_ann - Di_ann;

 ho = (NuDi_ann*kf_c)/Dh;

 % calculate a new Ui
 % fouling
 Ui = (1/(results(6)*results(8)) + Rfi/(results(8)) + results(10) +
Rfo/(results(9)) + 1/(ho*results(9)))^-1 / (results(8));      %[W/m^2K]
 % no fouling
 %Ui = (1/(results(6)*results(3)) + results(8) + 1/(ho*results(7)))^-1 /
(results(3));      %[W/m^2K]

 % find log mean temperature difference 11.17 and 11.15
 DT1 = Thi-Tco;                    %[K]
 DT2 = Tho-Tci;                    %[K]
 LMTD = (DT2-DT1)/log(DT2/DT1);   %[K]

 % the calculated heat the exchanger can move
 q_ht_ann = Ui*results(8)*LMTD             %[W]
```

```
q_ht_ann = 2.9564e+05
```

```
 UA_ann = Ui*results(8)              %[W/K]
```

```
UA_ann = 9.6908e+03
```

```
 % quantify difference
 per_diff_q = (results(1) - q_ht_ann)/(mean([results(1), q_ht_ann])) *
100;      %[%]
```

```
fprintf("The percent difference in the annular method and crossflow bank method
is: %.2f%%", per_diff_q)
```

```
The percent difference in the annular method and crossflow bank method is: 163.44%
```

```
%%%%%%%%%%%%%%%%%%%%
%%   NO FOULING   %%
%%%%%%%%%%%%%%%%%%%%
Ui = (1/(results(6)*results(8)) + results(10) + 1/(results(7)*results(9)))^-1 /
(results(8));      %[W/m^2K]
q_no_fouling = Ui*results(8)*LMTD       %[W]
```

```
q_no_fouling = 5.0891e+06
```

```
%%%%%%%%%%%%%%%%%%%
%%   TABLE 7.6   %%
%%%%%%%%%%%%%%%%%%%
function [C2] = table7_6(n_tubes)
% table 7.6
    if (n_tubes == 1)
        C2 = 0.68;
    elseif (n_tubes == 2)
        C2 = 0.75;
    elseif (n_tubes == 3)
        C2 = 0.83;
    elseif (n_tubes == 4)
        C2 = 0.89;
    elseif (n_tubes == 5)
        C2 = 0.92;
    elseif (n_tubes == 6)
        C2 = 0.95;
    elseif (n_tubes == 7)
        C2 = 0.97;
    elseif (n_tubes == 8)
        C2 = 0.98;
    elseif (n_tubes == 9)
        C2 = 0.99;
    else
        C2 = 1;
    end
end
%%%%%%%%%%%%%%%%%%%%
%%   TABLE 7.5   %%
%%%%%%%%%%%%%%%%%%%

function [C1, m] = table7_5(ST,SL,Do)
% table 7.5 (only the 'staggered' section)
% SL/D and ST/D are put into buckets based on table
if (SL/Do <= 0.75 && SL/Do >= 0.6)                          %0.6
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.213;
        m = 0.636;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 0.95 && SL/Do > 0.75)                      %0.9
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.446;
        m = 0.571;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.401;
        m = 0.581;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 1.0625 && SL/Do > 0.95)                    %1
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0.497;
        m = 0.558;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0;
        m = 0;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 1.1875 && SL/Do > 1.0625)                  %1.125
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
```

```
        C1 = 0;
        m = 0;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.478;
        m = 0.565;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.518;
        m = 0.560;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 1.375 && SL/Do > 1.1875)                  %1.25
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0.518;
        m = 0.556;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0.505;
        m = 0.554;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.519;
        m = 0.556;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.522;
        m = 0.562;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 1.75 && SL/Do > 1.375)                    %1.5
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0.451;
        m = 0.568;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0.460;
        m = 0.562;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.452;
        m = 0.568;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.488;
        m = 0.568;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 2.5 && SL/Do > 1.75)                      %2
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0.404;
        m = 0.572;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0.416;
        m = 0.568;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.482;
        m = 0.556;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.449;
        m = 0.570;
    else
        C1 = 0;
        m = 0;
    end
elseif (SL/Do <= 3 && SL/Do > 2.5)                          %3
    if (ST/Do <= 1.375 && ST/Do >= 1.25) %1.25
        C1 = 0.310;
        m = 0.592;
    elseif (ST/Do <= 1.75 && ST/Do > 1.375) %1.5
        C1 = 0.356;
        m = 0.580;
    elseif (ST/Do <= 2.5 && ST/Do > 1.75) %2
        C1 = 0.440;
        m = 0.562;
    elseif (ST/Do <= 3 && ST/Do > 2.5) %3
        C1 = 0.428;
        m = 0.574;
    else
        C1 = 0;
        m = 0;
    end
else
    C1 = 0;
    m = 0;
end
end
%%%%%%%%%%%%%%%%%%%%%%
%%   EULER NUMBER   %%
%%%%%%%%%%%%%%%%%%%%%%

%this function returns the constants for a taylor series polynomial that
%calculates the Euler number for a range of tube spacing ratios and
%reynolds numbers

%https://thermopedia.com/content/1211/

function [co, c1, c2, c3, c4] = eulerPolynomial(SL,Re)
    if (SL <= 1.375 && Re >= 3 && Re < 1e3)
        co = 0.795;
        c1 = 0.247e3;
        c2 = 0.335e3;
        c3 = -0.155e4;
        c4 = 0.241e4;
    elseif (SL <= 1.375 && Re >= 1e3 && Re <= 2e6)
        co = 0.245;
        c1 = 0.339e4;
        c2 = -0.984e7;
        c3 = 0.132e11;
        c4 = -0.599e13;
    elseif (SL <= 1.75 && SL > 1.375 && Re >= 3 && Re < 1e3)
        co = 0.683;
        c1 = 0.111e3;
        c2 = -0.973e2;
        c3 = -0.426e3;
        c4 = 0.574e3;
    elseif (SL <= 1.75 && SL > 1.375 && Re >= 1e3 && Re <= 2e6)
        co = 0.203;
        c1 = 0.248e4;
        c2 = -0.758e7;
        c3 = 0.104e11;
        c4 = -0.482e13;
    elseif (SL <= 2.25 && SL > 1.75 && Re >= 7 && Re < 1e2)
        co = 0.713;
        c1 = 0.448e2;
        c2 = -0.126e3;
        c3 = -0.582e3;
        c4 = 0;
    elseif (SL <= 2.25 && SL > 1.75 && Re >= 1e2 && Re < 1e4)
        co = 0.343;
        c1 = 0.303e3;
        c2 = -0.717e5;
        c3 = 0.88e7;
        c4 = -0.38e9;
    elseif (SL <= 2.25 && SL > 1.75 && Re >= 1e4 && Re <= 1e6)
        co = 0.162;
        c1 = 0.181e4;
        c2 = 0.792e8;
        c3 = -0.165e13;
        c4 = 0.872e16;
    elseif (SL > 2.25 && Re >= 1e2 && Re < 5e3)
        co = 0.33;
        c1 = 0.989e2;
        c2 = -0.148e5;
        c3 = 0.192e7;
        c4 = -0.862e8;
    elseif (SL > 2.25 && Re >= 5e3 && Re < 2e6)
        co = 0.119;
        c1 = 0.498e4;
        c2 = -0.507e8;
        c3 = 0.251e12;
        c4 = -0.463e15;
    else
        co = 0;
        c1 = 0;
        c2 = 0;
        c3 = 0;
        c4 = 0;
    end
end
%%%%%%%%%%%%%%%%%%%%%%
%%   TABLE VALUES   %%
%%%%%%%%%%%%%%%%%%%%%%

% this fucntion takes in any temperature from 300 K (27 C) to 370 K (97 C)
% ***in Kelvin***
% and returns the table values rho, cp, mu, kf, Pr
function [rho, cp, mu, kf, Pr] = satH2Otable(T)

    %manual input of table values
    %        [kg/m^3]   [J/kgK] [Ns/m^2] [W/Mk]  []
    %             rho       cp    mu      kf      Pr
    tab300 = [1/(1.003e-3) 4179 855e-6 613e-3 5.83];
    tab305 = [1/(1.005e-3) 4178 769e-6 620e-3 5.20];
    tab310 = [1/(1.007e-3) 4178 695e-6 628e-3 4.62];
    tab315 = [1/(1.009e-3) 4179 631e-6 634e-3 4.16];
    tab320 = [1/(1.011e-3) 4180 577e-6 640e-3 3.77];
    tab325 = [1/(1.013e-3) 4182 528e-6 645e-3 3.42];
    tab330 = [1/(1.016e-3) 4184 489e-6 650e-3 3.15];
    tab335 = [1/(1.018e-3) 4186 453e-6 656e-3 2.88];
    tab340 = [1/(1.021e-3) 4188 420e-6 660e-3 2.66];
    tab345 = [1/(1.024e-3) 4191 389e-6 668e-3 2.45];
    tab350 = [1/(1.027e-3) 4195 365e-6 668e-3 2.29];
    tab355 = [1/(1.030e-3) 4199 343e-6 671e-3 2.14];
    tab360 = [1/(1.034e-3) 4203 324e-6 674e-3 2.02];
    tab365 = [1/(1.038e-3) 4209 306e-6 677e-3 1.91];
    tab370 = [1/(1.041e-3) 4214 289e-6 679e-3 1.80];

    if (T == 300)
        rho = tab300(1);
        cp = tab300(2);
        mu = tab300(3);
        kf = tab300(4);
        Pr = tab300(5);
    elseif (T > 300 && T < 305)
        rho = linI(300, tab300(1), 305, tab305(1), T);
        cp = linI(300, tab300(2), 305, tab305(2), T);
        mu = linI(300, tab300(3), 305, tab305(3), T);
        kf = linI(300, tab300(4), 305, tab305(4), T);
        Pr = linI(300, tab300(5), 305, tab305(5), T);
    elseif (T == 305)
        rho = tab305(1);
        cp = tab305(2);
        mu = tab305(3);
        kf = tab305(4);
        Pr = tab305(5);
    elseif (T > 305 && T < 310)
        rho = linI(305, tab305(1), 310, tab310(1), T);
        cp = linI(305, tab305(2), 310, tab310(2), T);
        mu = linI(305, tab305(3), 310, tab310(3), T);
        kf = linI(305, tab305(4), 310, tab310(4), T);
        Pr = linI(305, tab305(5), 310, tab310(5), T);
    elseif (T == 310)
        rho = tab310(1);
        cp = tab310(2);
        mu = tab310(3);
        kf = tab310(4);
        Pr = tab310(5);
    elseif (T > 310 && T < 315)
        rho = linI(310, tab310(1), 315, tab315(1), T);
        cp = linI(310, tab310(2), 315, tab315(2), T);
        mu = linI(310, tab310(3), 315, tab315(3), T);
        kf = linI(310, tab310(4), 315, tab315(4), T);
        Pr = linI(310, tab310(5), 315, tab315(5), T);
    elseif (T == 315)
        rho = tab315(1);
        cp = tab315(2);
        mu = tab315(3);
        kf = tab315(4);
        Pr = tab315(5);
    elseif (T > 315 && T < 320)
        rho = linI(315, tab315(1), 320, tab320(1), T);
        cp = linI(315, tab315(2), 320, tab320(2), T);
        mu = linI(315, tab315(3), 320, tab320(3), T);
        kf = linI(315, tab315(4), 320, tab320(4), T);
        Pr = linI(315, tab315(5), 320, tab320(5), T);
    elseif (T == 320)
        rho = tab320(1);
        cp = tab320(2);
        mu = tab320(3);
        kf = tab320(4);
        Pr = tab320(5);
    elseif (T > 320 && T < 325)
        rho = linI(320, tab320(1), 325, tab325(1), T);
        cp = linI(320, tab320(2), 325, tab325(2), T);
        mu = linI(320, tab320(3), 325, tab325(3), T);
        kf = linI(320, tab320(4), 325, tab325(4), T);
        Pr = linI(320, tab320(5), 325, tab325(5), T);
    elseif (T == 325)
        rho = tab325(1);
        cp = tab325(2);
        mu = tab325(3);
        kf = tab325(4);
        Pr = tab325(5);
    elseif (T > 325 && T < 330)
        rho = linI(325, tab325(1), 330, tab330(1), T);
        cp = linI(325, tab325(2), 330, tab330(2), T);
        mu = linI(325, tab325(3), 330, tab330(3), T);
        kf = linI(325, tab325(4), 330, tab330(4), T);
        Pr = linI(325, tab325(5), 330, tab330(5), T);
    elseif (T == 330)
        rho = tab330(1);
        cp = tab330(2);
        mu = tab330(3);
        kf = tab330(4);
        Pr = tab330(5);
    elseif (T > 330 && T < 335)
        rho = linI(330, tab330(1), 335, tab335(1), T);
        cp = linI(330, tab330(2), 335, tab335(2), T);
        mu = linI(330, tab330(3), 335, tab335(3), T);
        kf = linI(330, tab330(4), 335, tab335(4), T);
        Pr = linI(330, tab330(5), 335, tab335(5), T);
    elseif (T == 335)
        rho = tab335(1);
        cp = tab335(2);
        mu = tab335(3);
        kf = tab335(4);
        Pr = tab335(5);
    elseif (T > 335 && T < 340)
        rho = linI(335, tab335(1), 340, tab340(1), T);
        cp = linI(335, tab335(2), 340, tab340(2), T);
        mu = linI(335, tab335(3), 340, tab340(3), T);
        kf = linI(335, tab335(4), 340, tab340(4), T);
        Pr = linI(335, tab335(5), 340, tab340(5), T);
    elseif (T == 340)
        rho = tab340(1);
        cp = tab340(2);
        mu = tab340(3);
        kf = tab340(4);
        Pr = tab340(5);
    elseif (T > 340 && T < 345)
        rho = linI(340, tab340(1), 345, tab345(1), T);
        cp = linI(340, tab340(2), 345, tab345(2), T);
        mu = linI(340, tab340(3), 345, tab345(3), T);
        kf = linI(340, tab340(4), 345, tab345(4), T);
        Pr = linI(340, tab340(5), 345, tab345(5), T);
    elseif (T == 345)
        rho = tab345(1);
        cp = tab345(2);
        mu = tab345(3);
        kf = tab345(4);
        Pr = tab345(5);
    elseif (T > 345 && T < 350)
        rho = linI(345, tab345(1), 350, tab350(1), T);
        cp = linI(345, tab345(2), 350, tab350(2), T);
        mu = linI(345, tab345(3), 350, tab350(3), T);
        kf = linI(345, tab345(4), 350, tab350(4), T);
        Pr = linI(345, tab345(5), 350, tab350(5), T);
    elseif (T == 350)
        rho = tab350(1);
        cp = tab350(2);
        mu = tab350(3);
        kf = tab350(4);
        Pr = tab350(5);
    elseif (T > 350 && T < 355)
        rho = linI(350, tab350(1), 355, tab355(1), T);
        cp = linI(350, tab350(2), 355, tab355(2), T);
        mu = linI(350, tab350(3), 355, tab355(3), T);
        kf = linI(350, tab350(4), 355, tab355(4), T);
        Pr = linI(350, tab350(5), 355, tab355(5), T);
    elseif (T == 355)
        rho = tab355(1);
        cp = tab355(2);
        mu = tab355(3);
        kf = tab355(4);
        Pr = tab355(5);
    elseif (T > 355 && T < 360)
        rho = linI(355, tab355(1), 360, tab360(1), T);
        cp = linI(355, tab355(2), 360, tab360(2), T);
        mu = linI(355, tab355(3), 360, tab360(3), T);
        kf = linI(355, tab355(4), 360, tab360(4), T);
        Pr = linI(355, tab355(5), 360, tab360(5), T);
    elseif (T == 360)
        rho = tab360(1);
        cp = tab360(2);
        mu = tab360(3);
        kf = tab360(4);
        Pr = tab360(5);
    elseif (T > 360 && T < 365)
        rho = linI(360, tab360(1), 365, tab365(1), T);
        cp = linI(360, tab360(2), 365, tab365(2), T);
        mu = linI(360, tab360(3), 365, tab365(3), T);
        kf = linI(360, tab360(4), 365, tab365(4), T);
        Pr = linI(360, tab360(5), 365, tab365(5), T);
    elseif (T == 365)
        rho = tab365(1);
        cp = tab365(2);
        mu = tab365(3);
        kf = tab365(4);
    elseif (T > 365 && T < 370)
        rho = linI(365, tab365(1), 370, tab370(1), T);
        cp = linI(365, tab365(2), 370, tab370(2), T);
        mu = linI(365, tab365(3), 370, tab370(3), T);
        kf = linI(365, tab365(4), 370, tab370(4), T);
        Pr = linI(365, tab365(5), 370, tab370(5), T);
    elseif (T == 370)
        rho = tab370(1);
        cp = tab370(2);
        mu = tab370(3);
        kf = tab370(4);
        Pr = tab370(5);
    end

     %error check range
    if (T > 370 || T < 300)
```

```
        fprintf(“Error: temperature falls outside of the valid range (300K-
370K”)
        rho = 0;
        cp = 0;
        mu = 0;
        kf = 0;
        Pr = 0;
    end
end
function y = linI(x0, y0, x1, y1, x)
    y = y0 + ( (y1 - y0) / (x1 - x0) ) * (x - x0);
end
```

APPENDIX Z

CODE APPENDIX: MATLAB CALCULATIONS FOR ANNULAR

```
clear, clc
%%%%%%%%%%%%%%%%%%%%%
%%   ASSUMPTIONS   %%
%%%%%%%%%%%%%%%%%%%%%

    % 1. steady state
    % 2. no heat is lost to surroundsings, i.e. shell is "well insulatied"
    % 3. when instructions say "working fluid is water" they mean both
    % fluids must be water
    % 4. hot water is inside the tubes while cold water is in the shell
    % (see figure on page 3 of instructions)
    % 5. if flow inside pipes is laminar, constant surface temperature is
    % the chosen boundary condition because it is more conservative
    % 6. surface roughness of the drawn copper tubing can be approximated
    % as smooth
    % 7. pressure drop across tube bank is approximated by pressure drop
    % across equally spaced tube bank
    % 8. interest rate of 8%, exchagner life span of at least 20 years

% from page 677
    % 1. well insulated
    % 2. axial conduction along tube is negligable
    % 3. PE and KE change is negligible
    % 4. cp is constant
    % 5. U (and UA) is constant

% simplifications
    % 1. baffles allow for crossflow over tubes
    % 2. counter flow is generally more efficient than parallel flow, so we
    % will only consider counter flow
    % 3. staggered banks are generally more efficient than straight banks,
    % so we will only consider staggered banks (due to turbulence)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   GIVENS AND PROJECT CONSTRAINTS   %%
```

```
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 mdot_h = 28;        %[kg/s]
 mdot_c = 27;        %[kg/s]
 Thi = 90+273;       %[C]
 Tci = 34+273;       %[C]
 Tco = 60+273;       %[C]
 U_max = 1700;       %[W/m^2K]
 U_min = 850;        %[W/m^2K]
 L_max = 10;         %[m]
 L_min = 1;          %[m]

 % ### Equation numbers are from physical book (6th edition) unless other
 % wise noted ###

 %%%%%%%%%%%%%%%%
 %%   COPPER   %%
 %%%%%%%%%%%%%%%%

 % copper tubing size and pricing (type M)
 % see: "McMaster Carr, Low-Pressure Copper Tubing for Drinking Water"

 % tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
 % can us the k value for pure copper

 %    index:  1        2         4         5        6         7         8
9       10
 %  ID [in]:  3/8      1/2       3/4       1        1 1/4     1 1/2     2
2 1/2    3
 cop_tubeM = [0.01143  0.0144526 0.0205994 0.026797 0.0327914 0.0387858
0.0510286 0.063373 0.0757174;   %[m] ID
              0.000635 0.0007112 0.0008128 0.000889 0.0010668 0.0012446
0.0014732 0.001651 0.0018288;   %[m] wall thickness
              30.30    28.13     45.29     69.35    128.88    177.12    272.57
402.14   533.86];     %[$] Cost of 10'

 % copper tubing size and pricing (type L)
 % see: "McMaster Carr, Medium-Pressure Copper Tubing for Drinking Water"

 % tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
 % can us the k value for pure copper

 %    index:  1        2        3        4         5        6        7        8
9         10        11
 %  ID [in]:  1/4      3/8      1/2      5/8       3/4      1        1 1/4    1
1/2    2         2 1/2    3
 cop_tubeL = [0.008001 0.010922 0.013843 0.0169164 0.019939 0.026035 0.032131
0.038227 0.050419 0.062611 0.074803;   %[m] ID
              0.000762 0.000889 0.001016 0.0010668 0.001143 0.00127  0.001397
0.001524 0.001778 0.002032 0.002286;   %[m] wall thickness
              26.27    38.38    37.81    74.21     70.19    103.30   163.59
207.85   294.53   450.57   629.00];    %[$] Cost of 10'
```

```
 % copper tubing size and pricing (type K)
 % see: "McMaster Carr, High-Pressure Copper Tubing for Drinking Water"

 % tubing meets standard ASTM B88 which ensrues alloy is 99.9% copper, so we
 % can us the k value for pure copper

 %    index:  1        2         3         4         5        6        7
8         9         10       11
 %  ID [in]:  1/4      3/8       1/2       5/8       3/4      1        1 1/4
1 1/2     2         2 1/2    3
 cop_tubeK = [0.007747 0.0102108 0.0133858 0.0165608 0.018923 0.025273 0.031623
0.0376174 0.0497586 0.061849 0.0738378;   %[m] ID
              0.000889 0.0012446 0.0012446 0.0012446 0.001651 0.001651 0.001651
0.0018288 0.0021082 0.002413 0.0027686;   %[m] wall thickness
              30.18    53.62     62.95     77.55     116.54   151.31   191.83
252.57    371.14    554.86   765.29];     %[$] Cost of 10'

 cop_tube = cat(2, cop_tubeM, cop_tubeL, cop_tubeK)
```

```
cop_tube = 3×31

    0.0114    0.0145    0.0206    0.0268    0.0328    0.0388    0.0510 ⋯
    0.0006    0.0007    0.0008    0.0009    0.0011    0.0012    0.0015
   30.3000   28.1300   45.2900   69.3500  128.8800  177.1200  272.5700
```

```
 % tables values for pure copper, estimated at 300K
 kc = 401;  %[W/mK]

 %%%%%%%%%%%%%%%
 %%   STEEL   %%
 %%%%%%%%%%%%%%%
 % assume rolling 1/4" sheet and weld
 % 1388K185
 steel_L = 1.8288;                   %[m]
 steel_W = 0.6096;                   %[m]
 steel_cost = 746.78;                %[USD]
 steel_area = steel_L * steel_W;     %[m^2]
 t_steel = 0.00635;                  %[m]

 %%%%%%%%%%%%%%%%%
 %%   FOULING   %%
 %%%%%%%%%%%%%%%%%
 % fouling, table 11.1
 Rfi = 0.0002;   %[m^2 K/W] because Tmi > 50C
 Rfo = 0.0001;   %[m^2 K/W] because Tmi < 50C

 %%%%%%%%%%%%%%%%
 %%   THERMO   %%
 %%%%%%%%%%%%%%%%
 % find mean temperatures and table values
 % GUESS Tho (iterated to balance qh and qc) 65.0206 C
 Tho_g = 65.0206+273;        %[K]
 Tmc = (Tco+Tci)/2;          %[K]
 Tmh = (Tho_g+Thi)/2;        %[K]

 [~, cp_c, ~, ~, ~] = satH2Otable(Tmc);
 [~, cp_h, ~, ~, ~] = satH2Otable(Tmh);

 % energy must be conserved, so, q_cold = q_hot
 q = mdot_c * cp_c * (Tci-Tco);  %[W]
 q_thermo = -q;                  %[W]
 Tho = q/(mdot_h * cp_h) + Thi;  %[K]
 % check this against Tho_g
 Tho - 273    %[C]
```

```
ans = 65.0206
```

```

 %get mean temperatures
 Tmc = (Tco+Tci)/2;
 Tmh = (Tho+Thi)/2;
 [rho_h, cp_h, mu_h, kf_h, Pr_h] = satH2Otable(Tmh);
 [rho_c, cp_c, mu_c, kf_c, Pr_c] = satH2Otable(Tmc);

 %%%%%%%%%%%%%%%%%%%
 %%   ITERATION   %%
 %%%%%%%%%%%%%%%%%%%

 % Storage
 best_eff = 10e6;
 best_design = [];

 % VARIABLES TO ITERATE
 % ST scalar max is ~2.99 min is ~1.05
 % SL scalar max is ~2.99 min is ~1
 ST_ratios = 0.8:0.1:2;
 %ST_ratios = 1.05:0.05:2.99;
 SL_ratios = 0.8:0.1:2;
 %SL_ratios = 1.05:0.05:2.99;
 Rows = 5:1:30;
 Cols = 5:1:30;
 Lengths = 2:0.1:10;
 Tube_Idxs = 1:length(cop_tube);

 permutations =
length(ST_ratios)*length(SL_ratios)*length(Rows)*length(Cols)*length(Lengths)*le
ngth(Tube_Idxs);
 fprintf("Analyzing %d permutations... ", permutations);
```

```
Analyzing 286866684 permutations...
```

```

 counter = 0;
 for t_idx = Tube_Idxs
     Di = cop_tube(1, t_idx);
     t_wall = cop_tube(2, t_idx);
     Do = Di + 2*t_wall;
     cost_ten_ft = cop_tube(3, t_idx);
     for L = Lengths
         for n_r = Rows
             for n_c = Cols
                 n_tubes = n_r * n_c;
                 % code optimization, if area is less than 20 m^2 don't
                 % bother calculating (would force U to be greater than
                 % 1700, not possible)
                 area_approx = pi*Do*L*n_tubes;
                 if area_approx < 20; continue; end

                 for st_r = ST_ratios
                     for sl_r = SL_ratios
                         counter = counter + 1;

                         ST = st_r*Do;
                         SL = sl_r*Do;
                         if ST < 1.05*Do; continue; end
                         diag_pitch = sqrt((ST/2)^2 + SL^2);
                         if (diag_pitch < 1.05*Do); continue; end
```

Flow over the bank of tubes, using annulus method

```
                             % calculate tube bank height and width
                             bank_h = (1+n_r)*(Do) + (n_r-1)*(ST-Do) + (ST/2 -
Do);  %[m]
                             bank_w = n_c*Do + (n_c-1)*(SL-
Do);                      %[m]

                             % calculate shell diameter based on tube bank
dimensions,
                             % bank 'diameter' +10%
                             min_shell_D = sqrt(bank_h^2 +
bank_w^2);                    %[m]
                             Di_shell =
min_shell_D*1.1;                                  %[m]
                             A_Shell =
pi*(Di_shell/2)^2;                                 %[m^2]

                             %%%%%%%%%%%%%%%%%%%%%%%%
                             %%   ANNULUS METHOD   %%
                             %%%%%%%%%%%%%%%%%%%%%%%%
                             % section 8.6 in book, table 8.2
                             % assumes fully laminar, fully developed
                             % trash assumptions
                             gap = sqrt((st_r*Do)^2 + (sl_r*Do)^2) - Do;
                             Di_ann = Do;
                             Do_ann = Do+2*gap;
                             DiDo = Di_ann/Do_ann;

                             % reynolds
                             Dh = Do_ann - Di_ann;
                             u_o_max = mdot_c./(rho_c*n_tubes*(pi*(Do_ann/2)^2 -
pi*(Di_ann/2)^2));      %[m/s]
                             Re_o_ann =
(rho_c*u_o_max*Dh)/mu_c;                                        %[]
                             ReDc =
2300;                                                               %[] onset
of turbulence

                             if (Re_o_ann < ReDc)
                                 if (DiDo < 0.075)
                                     NuDi_ann = 17.46;
                                 elseif (DiDo >= 0.075 && DiDo < 0.175)
                                     NuDi_ann = 11.56;
                                 elseif (DiDo >= 0.175 && DiDo < 0.375)
                                     NuDi_ann = 7.37;
                                 elseif (DiDo >= 0.375 && DiDo < 0.75)
                                     NuDi_ann = 5.74;
                                 elseif (DiDo >= 0.75 && DiDo < 1)
                                     NuDi_ann = 4.86;
                                 end
                             elseif (3000 <= Re_o_ann && 5e6 >= Re_o_ann)
                                 % use gnielinski if turbulent
                                 % https://www.thermal-engineering.org/what-is-
gnielinski-equation-definition/
                                 f_turb = (0.79*log(Re_o_ann) - 1.64)^-2;  %
8.21
                                 NuDi_ann = ((f_turb/8)*(Re_o_ann-1000)*Pr_c)/(1
+ 12.7*(f_turb/8)^(1/2)*(Pr_c^(2/3)-1));
                             else
                                 NuDi_ann = 0;
                             end

                             if NuDi_ann == 0; continue; end
                             ho = (NuDi_ann*kf_c)/Dh;

                             % pressure drop, table 8.1
                             f_ann = 96/Re_o_ann;
                             hL_ann =
f_ann*(L/Di_ann)*(u_o_max^2/(2*9.81)); %[m]
                             dP_single_ann = hL_ann*(rho_c*9.81);        %[Pa]
                             dPo = dP_single_ann;                        %[Pa]
because annuli are in parallel?
```

Flow inside of a single tube

```
                             % find reynolds number for flow inside a single
tube
```

```
                             mdot_h_single = mdot_h/n_tubes;         %[kg/s]
                             Di = cop_tube(1, t_idx);                %[m]
                             Re_iD = (4*mdot_h_single)/(pi*Di*mu_h); %[]

                             % friction factor
                             % fluids textbook: "drawn tubing = 0.0015mm"
                             % engineers toolbox: "drawn copper... = 0.0.001-
0.002mm"
                             % heat transfer textbook: "drawn tubing = 0.0015mm"
                             % for the smallest pipe 1/4, e/D = 0.0002 which is
a good approximation for
                             % reynolds numbers below 10e6
                             % for normal pipe diameters of 1", e/D = 0.00001
which is a very good
                             % approximation!
                             % thus, we can be confident that our error from
assuming a smooth pipe is
                             % less than the ~15% error difference in using
equation 8.62 compared to
                             % 8.60

                             if (Re_iD >= ReDc && Re_iD <= 5e6)
                                 f = (0.79*log(Re_iD) - 1.64)^-2;
                             elseif (Re_iD <= ReDc)
                                 f = 64/Re_iD;
                             else
                                 f = 0;
                             end

                             if f == 0; continue; end

                             % set up if structure to filter corelations
                             if (Re_iD <= 2300)
                                 Gz = (Di/L)*Re_iD*Pr_h;
                                 NuDi = (3.66/(tanh(2.264*Gz^(-1/3) + 1.7*Gz^(-
2/3))) + 0.0499*Gz*tanh(Gz^-1))  /  (tanh(2.432*Pr_h^(1/6)*Gz^(-1/6))); %[]
                                 % assumes constant Ts, more conservative
assumption, eqn 8.58 in online book
                                 % Pr will always be > 0.1 for water, combined
entry length problem
                             elseif (Re_iD > 2300)
                                 NuDi = ((f/8)*(Re_iD-1000)*Pr_h)/(1 +
12.7*(f/8)^(1/2)*(Pr_h^(2/3)-1)); %[] eqn 8.62
                             end

                             hi = NuDi*(kf_h/Di);    %[W/m^2 K] for all tubes

                             % claculate pressure drop, from fluids text
                             % book, eqn 10-3 / 10-5
                             u_i_max =
mdot_h_single/(rho_h*(pi*(Di/2)^2));  %[m/s]
                             hL = f*(L/Di)*(u_i_max^2/(2*9.81)); %[m]
                             dP_single = hL*(rho_h*9.81);        %[Pa]
                             dPi = dP_single; %[Pa] because tubes are in
parallel!
```

Wall Resistance, Fouling, Overall Heat Transfer Coefficient

```
                             % wall resistance and conduction
                             Rw_single = log((Do/2) / (Di/2)) /
(2*pi*kc*L);  %Rw = ln(ro/ri)/(2*pi*k*L)
                             Rw = Rw_single/n_tubes;                  %
resistance is in parallel

                             % surface area
                             SAi = pi*Di*L*n_tubes;   %[m^2]
                             SAo = pi*Do*L*n_tubes;   %[m^2]

                             % U calculations, eqn 11.5 and similar
                             Ui = (1/(hi*SAi) + Rw + 1/(ho*SAo))^-1 /
(SAi);      %[W/m^2K]
                             Uo = (1/(hi*SAi) + Rw + 1/(ho*SAo))^-1 /
(SAo);      %[W/m^2K]
                             %U = (1/hi + log(Do/Di)*(Di/(2*kc)) + 1/ho)^-
1;      %[W/m^2K]
                             Um =
mean([Ui,Uo]);                                 %[W/m^2K]

                             % find log mean temperature difference 11.17 and
11.15
                             DT1 = Thi-Tco;                  %[K]
                             DT2 = Tho-Tci;                  %[K]
                             LMTD = (DT2-DT1)/log(DT2/DT1);  %[K]

                             % the calculated heat the exchanger can move
                             q_ht = Ui*SAi*LMTD;             %[W]

                             % check U
                             if Ui < U_min; continue; end
                             if Ui > U_max; continue; end
```

Financial Considerations

```
                             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                             %%   CALCULATE MATERIAL COST   %%
                             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                             cop_mat_cost =
n_tubes*cop_tube(3,t_idx)*((L*3.28084)/10);  %[USD]
                             % from CAD, assume baffle takes up 80% of shell
                             % face area, where copper tubes pass through is not
considered
                             % (don't pay for it)
                             steel_mat_cost = (((Di_shell*pi*L) +
2*pi*(Di_shell/2)^2)/ (steel_area)) * steel_cost; %[USD]
                             mat_cost = cop_mat_cost +
steel_mat_cost;                                   %[USD]

                             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                             %%   CALCULATE PUMPING COST   %%
                             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                             % from fluids book, derived from equation 14-15
(general
                             % hydraulic power equation) knowing that
gamma*h_pump = dP
                             % assume pump efficiency of 0.73 (from JCI
                             % experience)
                             pump_power = ((mdot_h/rho_h)*dPi +
(mdot_c/rho_c)*dPo)/(1000*0.73);    %[kW]
                             hours_in_a_year =
24*365;                                                %[h]
                             yearly_power =
pump_power*hours_in_a_year;                               %[kWh]
                             avg_kwh_cost =
0.1583;                                                   %[USD/kWh]
                             pump_cost_yr =
yearly_power*avg_kwh_cost;                                %[USD]

                             %%%%%%%%%%%%%%%%%%%%%%%
                             %%   SALVAGE VALUE   %%
                             %%%%%%%%%%%%%%%%%%%%%%%

                             cop_scrap = 7.70;       %[USD/kg] $3-4 per lb
                             steel_scrap = 0.358274; %[USD/kg] $0.16 per lb
                             rho_cop = 8933;         %[kg/m^3]
                             rho_steel = 7854;       %[kg/m^3]

                             cop_salvg = (n_tubes*L*((pi*(Do/2)^2)-
(pi*(Di/2)^2)))*rho_cop*cop_scrap; %[USD]
                             steel_salvg = (((Di_shell*pi*L*t_steel) +
2*t_steel*pi*(Di_shell/2)^2)) * rho_steel * steel_scrap; %[USD]
                             salvg = cop_salvg + steel_salvg;    %[USD]

                             %%%%%%%%%%%%%%%%%%%%%%%
                             %%   ECON ANALYSIS   %%
                             %%%%%%%%%%%%%%%%%%%%%%%
                             i = 0.08;       %[] discount rate
                             N = 20;         %[yrs] analysis horizon (lifespan
of heat exchanger)

                             % finding present value of operating cost and
                             % salvage value
                             PV_Op = pump_cost_yr*((1-(1+i)^-N)/i);  %[USD]
                             PV_sal = salvg/((1+i)^N);               %[USD]
```

```
                              % new present value
                              NPV = mat_cost + PV_Op - PV_sal;        %[USD]

                              % equivalent annual cost
                              CRF = (i*(1+i)^N)/((1+i)^N-1);          %[USD]
                              EAC = NPV*CRF;                          %[USD]

                              %%%%%%%%%%%%%%%%%%%%
                              %%   EFFICIENCY   %%
                              %%%%%%%%%%%%%%%%%%%%

                              if (q_ht >= q_thermo && (q_ht/q_thermo) < 1.1)
                                  nUSD = q_ht/(1000*EAC);         %[kW/USD]
                                  if EAC < best_eff
                                      best_eff = EAC;
                                      best_nUSD = nUSD;
                                      if (t_idx <= 10)
                                          str_t_idx = "Type M";
                                      elseif (t_idx >= 11 && t_idx <= 21)
                                          str_t_idx = "Type L"
                                      elseif (t_idx >= 22 && t_idx <= 32)
                                          str_t_idx = "Tpye K"
                                      else
                                          fprintf("Error: tube index out of
bounds");
                                      end

                                      design = [L, n_r, n_c, cop_tube(1,
t_idx)*39.3700787, st_r, sl_r, n_tubes];
                                      results = [q_ht, Ui, SAi, Um,
q_ht/q_thermo, hi, SAo, Rw];
                                      geometry = [Di*39.3700787, Do*39.3700787,
Di_shell*39.3700787, t_idx];
                                      economic = [NPV, PV_Op, PV_sal, EAC, N, i,
steel_mat_cost, cop_mat_cost, pump_cost_yr];
                                      hempe = [hi, ho, Ui, Uo, Um, gap, SAi, SAo,
q_ht];
                                      report = [dP_single, pump_power];
                                  end
                              end
                      end
                  end
              end
          end
      end
      fprintf("Finished tube index %d\n", t_idx);
 end
```

```
Finished tube index 1

Finished tube index 2
Finished tube index 3
Finished tube index 4
Finished tube index 5
Finished tube index 6
Finished tube index 7
Finished tube index 8
Finished tube index 9
Finished tube index 10
Finished tube index 11
Finished tube index 12
Finished tube index 13
Finished tube index 14
Finished tube index 15
Finished tube index 16
Finished tube index 17
Finished tube index 18
Finished tube index 19
Finished tube index 20
Finished tube index 21
Finished tube index 22
Finished tube index 23
Finished tube index 24
Finished tube index 25
Finished tube index 26
Finished tube index 27
Finished tube index 28
Finished tube index 29
Finished tube index 30
Finished tube index 31
```

```
fprintf("Analyzed %d permutations", permutations)
```

```
Analyzed 286866684 permutations
```

```
%%%%%%%%%%%%%%%%%
%%   RESULTS   %%
%%%%%%%%%%%%%%%%%
```

```
str_design = {"Length [m]"; "Number of Rows"; "Number of Columns"; "Nominal Tube Size [in]"; "ST Ratio"; "SL Ratio"; "Number of Tubes"};
str_geom = {"Tube Inner Diameter [in]"; "Tube Outer Diameter [in]"; "Shell Inner Diameter [in]"; "Tube Index"};
str_results = {"Heat Transfer Capacity [W]"; "Ui [W/m^2K]"; "Surface Area (inner) [m^2]"; "U [W/K]"; "FOS"; "Inner Convection Coefficient (hi) [W/m^2K]"; "Surface Area (outer) [m^2]"; "Wall Resistance [W/K]"};
str_econ = {"Net Present Value [$]"; "OpEx PV [$]"; "Salvage PV [$]"; "Equivalent Annual Cost [$]"; "Exchanger Lifespan [yrs]"; "Discount Rate []"; "Steel Material Cost [$]"; "Copper Material Cost [$]"; "Pump Cost per Year [$]"};
str_hempe = {"hi"; "ho"; "Ui"; "Uo"; "Um"; "gap"; "SAi"; "SAo"; "heat transfer"}; %hi, ho, Ui, Uo, Um, SAi, SAo, q_ht
str_report = {"Pressure Drop Inside A Single Tube [Pa]"; "Total System Pump Power [kW]"};
fprintf("Tube Index: %s\nExchanger Efficiency %.4f [W/USD]", str_t_idx, best_nUSD)
```

```
Tube Index: Type M

Exchanger Efficiency 1.6048 [kW/USD]
```

```
tab_design = table(str_design, design')
```

```
tab_design = 7×2 table
```

	str_design	Var2
1	"Length [m]"	6.4000
2	"Number of Rows"	14
3	"Number of Columns"	14
4	"Nominal Tube Size [in]"	0.5690
5	"ST Ratio"	1.1000
6	"SL Ratio"	0.9000
7	"Number of Tubes"	196

```
tab_geom = table(str_geom, geometry')
```

```
tab_geom = 4×2 table
```

	str_geom	Var2
1	"Tube Inner Diameter [in]"	0.5690

	str_geom	Var2
2	"Tube Outer Diameter [in]"	0.6250
3	"Shell Inner Diameter [in]"	13.9634
4	"Tube Index"	2

```
tab_results = table(str_results, results')
```

```
tab_results = 8×2 table
```

	str_results	Var2
1	"Heat Transfer Capacity [W]"	2.9533e+06
2	"Ui [W/m^2K]"	1.6997e+03
3	"Surface Area (inner) [m^2]"	56.9550
4	"U [W/K]"	1.6235e+03
5	"FOS"	1.0064
6	"Inner Convection Coefficient (hi) [W/m^2K]"	6.7641e+03
7	"Surface Area (outer) [m^2]"	62.5604
8	"Wall Resistance [W/K]"	2.9701e-08

```
tab_econ = table(str_econ, economic')
```

```
tab_econ = 9×2 table
```

	str_econ	Var2
1	"Net Present Value [$]"	1.8068e+04
2	"OpEx PV [$]"	2.2370e+03
3	"Salvage PV [$]"	655.2844
4	"Equivalent Annual Cost [$]"	1.8402e+03
5	"Exchanger Lifespan [yrs]"	20
6	"Discount Rate []"	0.0800

	str_econ	Var2
7	"Steel Material Cost [$]"	4.9092e+03
8	"Copper Material Cost [$]"	1.1577e+04
9	"Pump Cost per Year [$]"	227.8424

```
%tab_hempe = table(str_hempe, hempe')
tab_report = table(str_report, report')
```

```
tab_report = 2×2 table
```

	str_report	Var2
1	"Pressure Drop Inside A Single Tube [Pa]"	3.9370e+03
2	"Total System Pump Power [kW]"	0.1643

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   EFFECTIVENESS NTU   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%
C_h = cp_h*mdot_h;     %[W/K]
C_c = cp_c*mdot_c;     %[W/K]

if C_h < C_c
    Cmin = C_h;
else
    Cmin = C_c;
end

if (C_c < C_h)
    q_max = C_c*(Thi-Tci);   %[W]
elseif (C_h < C_c)
    q_max = C_h*(Thi-Tci);   %[W]
else
    fprintf("Null Solution")
    q_max = 0;               %[W]
end

eff = results(1)/q_max;                     %[] eqn 11.19
NTU = (results(2)*results(3))/Cmin;           %[] eqn 11.24
fprintf("Heat Exchanger Effectiveness: %.2f\nHeat Exchanger NTU: %.2f", eff,
NTU)
```

```
Heat Exchanger Effectiveness: 0.47

Heat Exchanger NTU: 0.86
```

```
%%%%%%%%%%%%%%%%%%%%%%
%%   TABLE VALUES   %%
%%%%%%%%%%%%%%%%%%%%%%

% this fucntion takes in any temperature from 300 K (27 C) to 370 K (97 C)
% ***in Kelvin***
% and returns the table values rho, cp, mu, kf, Pr
function [rho, cp, mu, kf, Pr] = satH2Otable(T)

    %manual input of table values
    %        [kg/m^3]   [J/kgK] [Ns/m^2] [W/mK]  []
    %             rho       cp    mu      kf      Pr
    tab300 = [1/(1.003e-3) 4179 855e-6 613e-3 5.83];
    tab305 = [1/(1.005e-3) 4178 769e-6 620e-3 5.20];
    tab310 = [1/(1.007e-3) 4178 695e-6 628e-3 4.62];
    tab315 = [1/(1.009e-3) 4179 631e-6 634e-3 4.16];
    tab320 = [1/(1.011e-3) 4180 577e-6 640e-3 3.77];
    tab325 = [1/(1.013e-3) 4182 528e-6 645e-3 3.42];
    tab330 = [1/(1.016e-3) 4184 489e-6 650e-3 3.15];
    tab335 = [1/(1.018e-3) 4186 453e-6 656e-3 2.88];
    tab340 = [1/(1.021e-3) 4188 420e-6 660e-3 2.66];
    tab345 = [1/(1.024e-3) 4191 389e-6 668e-3 2.45];
    tab350 = [1/(1.027e-3) 4195 365e-6 668e-3 2.29];
    tab355 = [1/(1.030e-3) 4199 343e-6 671e-3 2.14];
    tab360 = [1/(1.034e-3) 4203 324e-6 674e-3 2.02];
    tab365 = [1/(1.038e-3) 4209 306e-6 677e-3 1.91];
    tab370 = [1/(1.041e-3) 4214 289e-6 679e-3 1.80];

    if (T == 300)
        rho = tab300(1);
        cp = tab300(2);
        mu = tab300(3);
        kf = tab300(4);
        Pr = tab300(5);
    elseif (T > 300 && T < 305)
        rho = linI(300, tab300(1), 305, tab305(1), T);
        cp = linI(300, tab300(2), 305, tab305(2), T);
        mu = linI(300, tab300(3), 305, tab305(3), T);
        kf = linI(300, tab300(4), 305, tab305(4), T);
        Pr = linI(300, tab300(5), 305, tab305(5), T);
    elseif (T == 305)
        rho = tab305(1);
        cp = tab305(2);
        mu = tab305(3);
        kf = tab305(4);
        Pr = tab305(5);
    elseif (T > 305 && T < 310)
        rho = linI(305, tab305(1), 310, tab310(1), T);
        cp = linI(305, tab305(2), 310, tab310(2), T);
        mu = linI(305, tab305(3), 310, tab310(3), T);
        kf = linI(305, tab305(4), 310, tab310(4), T);
        Pr = linI(305, tab305(5), 310, tab310(5), T);
    elseif (T == 310)
        rho = tab310(1);
        cp = tab310(2);
        mu = tab310(3);
        kf = tab310(4);
        Pr = tab310(5);
    elseif (T > 310 && T < 315)
        rho = linI(310, tab310(1), 315, tab315(1), T);
        cp = linI(310, tab310(2), 315, tab315(2), T);
        mu = linI(310, tab310(3), 315, tab315(3), T);
        kf = linI(310, tab310(4), 315, tab315(4), T);
        Pr = linI(310, tab310(5), 315, tab315(5), T);
    elseif (T == 315)
        rho = tab315(1);
        cp = tab315(2);
        mu = tab315(3);
        kf = tab315(4);
        Pr = tab315(5);
    elseif (T > 315 && T < 320)
        rho = linI(315, tab315(1), 320, tab320(1), T);
        cp = linI(315, tab315(2), 320, tab320(2), T);
        mu = linI(315, tab315(3), 320, tab320(3), T);
        kf = linI(315, tab315(4), 320, tab320(4), T);
        Pr = linI(315, tab315(5), 320, tab320(5), T);
    elseif (T == 320)
        rho = tab320(1);
        cp = tab320(2);
        mu = tab320(3);
        kf = tab320(4);
        Pr = tab320(5);
    elseif (T > 320 && T < 325)
        rho = linI(320, tab320(1), 325, tab325(1), T);
        cp = linI(320, tab320(2), 325, tab325(2), T);
        mu = linI(320, tab320(3), 325, tab325(3), T);
        kf = linI(320, tab320(4), 325, tab325(4), T);
        Pr = linI(320, tab320(5), 325, tab325(5), T);
    elseif (T == 325)
        rho = tab325(1);
        cp = tab325(2);
        mu = tab325(3);
        kf = tab325(4);
        Pr = tab325(5);
    elseif (T > 325 && T < 330)
        rho = linI(325, tab325(1), 330, tab330(1), T);
        cp = linI(325, tab325(2), 330, tab330(2), T);
        mu = linI(325, tab325(3), 330, tab330(3), T);
        kf = linI(325, tab325(4), 330, tab330(4), T);
        Pr = linI(325, tab325(5), 330, tab330(5), T);
    elseif (T == 330)
        rho = tab330(1);
        cp = tab330(2);
        mu = tab330(3);
        kf = tab330(4);
        Pr = tab330(5);
    elseif (T > 330 && T < 335)
        rho = linI(330, tab330(1), 335, tab335(1), T);
        cp = linI(330, tab330(2), 335, tab335(2), T);
        mu = linI(330, tab330(3), 335, tab335(3), T);
        kf = linI(330, tab330(4), 335, tab335(4), T);
        Pr = linI(330, tab330(5), 335, tab335(5), T);
    elseif (T == 335)
        rho = tab335(1);
        cp = tab335(2);
        mu = tab335(3);
        kf = tab335(4);
        Pr = tab335(5);
    elseif (T > 335 && T < 340)
        rho = linI(335, tab335(1), 340, tab340(1), T);
        cp = linI(335, tab335(2), 340, tab340(2), T);
        mu = linI(335, tab335(3), 340, tab340(3), T);
        kf = linI(335, tab335(4), 340, tab340(4), T);
        Pr = linI(335, tab335(5), 340, tab340(5), T);
    elseif (T == 340)
        rho = tab340(1);
        cp = tab340(2);
        mu = tab340(3);
        kf = tab340(4);
        Pr = tab340(5);
    elseif (T > 340 && T < 345)
        rho = linI(340, tab340(1), 345, tab345(1), T);
        cp = linI(340, tab340(2), 345, tab345(2), T);
        mu = linI(340, tab340(3), 345, tab345(3), T);
        kf = linI(340, tab340(4), 345, tab345(4), T);
        Pr = linI(340, tab340(5), 345, tab345(5), T);
    elseif (T == 345)
        rho = tab345(1);
        cp = tab345(2);
        mu = tab345(3);
        kf = tab345(4);
        Pr = tab345(5);
    elseif (T > 345 && T < 350)
        rho = linI(345, tab345(1), 350, tab350(1), T);
        cp = linI(345, tab345(2), 350, tab350(2), T);
        mu = linI(345, tab345(3), 350, tab350(3), T);
        kf = linI(345, tab345(4), 350, tab350(4), T);
        Pr = linI(345, tab345(5), 350, tab350(5), T);
    elseif (T == 350)
        rho = tab350(1);
        cp = tab350(2);
        mu = tab350(3);
        kf = tab350(4);
        Pr = tab350(5);
    elseif (T > 350 && T < 355)
        rho = linI(350, tab350(1), 355, tab355(1), T);
        cp = linI(350, tab350(2), 355, tab355(2), T);
        mu = linI(350, tab350(3), 355, tab355(3), T);
        kf = linI(350, tab350(4), 355, tab355(4), T);
        Pr = linI(350, tab350(5), 355, tab355(5), T);
    elseif (T == 355)
        rho = tab355(1);
        cp = tab355(2);
        mu = tab355(3);
        kf = tab355(4);
        Pr = tab355(5);
    elseif (T > 355 && T < 360)
        rho = linI(355, tab355(1), 360, tab360(1), T);
        cp = linI(355, tab355(2), 360, tab360(2), T);
        mu = linI(355, tab355(3), 360, tab360(3), T);
        kf = linI(355, tab355(4), 360, tab360(4), T);
        Pr = linI(355, tab355(5), 360, tab360(5), T);
    elseif (T == 360)
        rho = tab360(1);
        cp = tab360(2);
        mu = tab360(3);
        kf = tab360(4);
        Pr = tab360(5);
    elseif (T > 360 && T < 365)
        rho = linI(360, tab360(1), 365, tab365(1), T);
        cp = linI(360, tab360(2), 365, tab365(2), T);
        mu = linI(360, tab360(3), 365, tab365(3), T);
        kf = linI(360, tab360(4), 365, tab365(4), T);
        Pr = linI(360, tab360(5), 365, tab365(5), T);
    elseif (T == 365)
        rho = tab365(1);
        cp = tab365(2);
        mu = tab365(3);
        kf = tab365(4);
    elseif (T > 365 && T < 370)
        rho = linI(365, tab365(1), 370, tab370(1), T);
        cp = linI(365, tab365(2), 370, tab370(2), T);
        mu = linI(365, tab365(3), 370, tab370(3), T);
        kf = linI(365, tab365(4), 370, tab370(4), T);
        Pr = linI(365, tab365(5), 370, tab370(5), T);
    elseif (T == 370)
        rho = tab370(1);
        cp = tab370(2);
        mu = tab370(3);
        kf = tab370(4);
        Pr = tab370(5);
    end

     %error check range
    if (T > 370 || T < 300)
        fprintf("Error: temperature falls outside of the valid range (300K-
370K")
        rho = 0;
        cp = 0;
        mu = 0;
        kf = 0;
        Pr = 0;
    end
end
function y = linI(x0, y0, x1, y1, x)
    y = y0 + ( (y1 - y0) / (x1 - x0) ) * (x - x0);
end
```