An Automotive Application of the Organic Rankine Cycle for Power Generation Using Recovered Waste Heat

Glenn Michael Clapp

A Project Submitted to the Graduate Faculty of

GRAND VALLEY STATE UNIVERSITY

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science in Engineering

Padnos College of Engineering and Computing

August 2019

Grand Valley State University
School of Engineering

Certificate of Examination

Advisor

Wael
Mokhtar

Digitally signed by Wael Mokhtar
DN: cn=Wael Mokhtar, o=Grand Valley State University, ou=School of Engineering, email=mokhtarw@gvsu.edu, c=US
Date: 2019.08.16 11:51:44 -04'00'

Dr. W. Mokhtar

Examining Board

Digitally signed by Mehmet Sozen
DN: cn=Mehmet Sozen, o=GVSU, ou=SOE, email=sozenm@gvsu.edu, c=US

Dr. M. Sozen

Digitally signed by ridgel@jsjcorp.com
DN: cn=ridgel@jsjcorp.com
Date: 2019.08.16 08:29:15 -04'00'

Larry Ridge

The EGR 693 project by

Glenn Clapp

entitled

An Automotive Application of the Organic Rankine Cycle for Power Generation Using Recovered Waste Heat

is accepted in partial fulfillment of the degree requirements for the degree of Master of Science in Engineering

____

Date

____

Dr. Shabbir Choudhuri
Graduate Program Director

Contents

1. Figures

2. Abstract

The goal of this project was to develop a parametric model of an organic Rankine cycle for the purpose of generating electrical power using waste heat from the coolant system of an automobile. This application required a small package size, and the utilization of low temperature, low quality waste heat.

Some of the barriers presented to the development of low-temperature waste heat recovery are:

1. Long payback periods
2. Material constraints and costs
3. Economies of scale (waste heat recovery does not lend itself well to a general solution)
4. Operation and maintenance costs – Corrosion, scaling, and fouling of heat exchange materials lead to higher maintenance costs and lost productivity.
5. Large heat exchanger size due to low temperature difference

A successful solution to this problem will benefit from mass production and therefore mitigate the capital costs of development, utilize materials common to automotive parts in existing electric motors and pumps as well as refrigerants that are used in climate control applications. This project developed industry knowledge in the area of waste heat recovery and increase the fuel efficiency of all vehicles in which it is implemented. A solution to this problem is desired to help meet increasingly strict fuel economy restrictions and to make responsible use of society's energy resources and to protect the climate.

The model used in this study can be used to develop cycle parameters and design specifications for use in prototype development by indicating whether the design parameters present in a given application are adequate for a power cycle of this type.

The parametric model developed for this project was verified where applicable with physical prototypes and measurements. Previous work in mathematical modeling and literature on the subjects of waste heat recovery and vapor-power cycles were reviewed to inform an efficient starting point for this project.

In this study, It was determined that an organic Rankine cycle is unlikely to be feasible in an automotive application where waste heat is harvested from the cooling system due to the low temperature, and low quality of available heat as well as the excessive heat exchanger sizes that would be required for adequate heat transfer. Finally, there is simply not enough energy available in the waste heat of a typical vehicle to sustain a cycle with the typical efficiencies obtained with today’s technology.

Keywords: Rankine cycle; Working pressure; Working temperature; Boiler; Condenser; Power-vapor cycle; Organic Rankine cycle; Low quality heat; Automotive; Waste heat recovery

3. Introduction and background

The following tables contain the usage of the symbols throughout this document unless specified otherwise.

Table 1: List of symbols and constants

Symbol	Description	Value
σ	Stefan-Boltzmann constant	5.67e-8 W/(m^{2}K^4)
ε	Emissivity	$0 < \varepsilon < 1$
κ	Thermal conductivity (kJ/(kgK)	
h	Heat transfer coefficient (W/(m^{2}K)	
p	Pressure (Pa)	
V	Volume (m^3)	
U	Internal energy (J)	
H	Enthalpy (J)	U + pV
h	Specific enthalpy (J/kg)	
$\dot{Q}$	Heat transfer rate (W)	
A	Area (m^2)	
KE	Kinetic energy (J)	
PE	Potential energy (J)	
Q	Heat transfer (J)	
W	Work (J)	
s	Specific entropy (kJ/kgK)	
T	Temperature	^{o}C
ν	Specific volume (m^3/kg)	
η_{th}	Thermal efficiency	$0 < \eta_{th} < 1$
η	Isentropic efficiency	$0 < \eta < 1$
π	The ratio of the circumference of a circle to its diameter	3.1415 (approximate value)

Table 2: Subscripts

Subscript	Description
b	Boiler
c	Condenser
H	The hot side of a heat exchanger
C	The cool side of a heat exchanger
vapor or g	Denotes a saturated vapor state
liquid or f	Denotes a saturated liquid state
t	Turbine
p	Pump
atm	Property of atmosphere
water	Property of water
cv	Control volume
s	Appended after a state number to denote an isentropic assumption

Internal combustion engines do not generate power very efficiently. Only about 20% of the energy available in gasoline or diesel fuel is converted to mechanical motion. The great majority of the rest of the energy is converted into heat.

According to Singh and Pedersen's work [1], a heat balance for a typical maritime application might look like the following diagram.

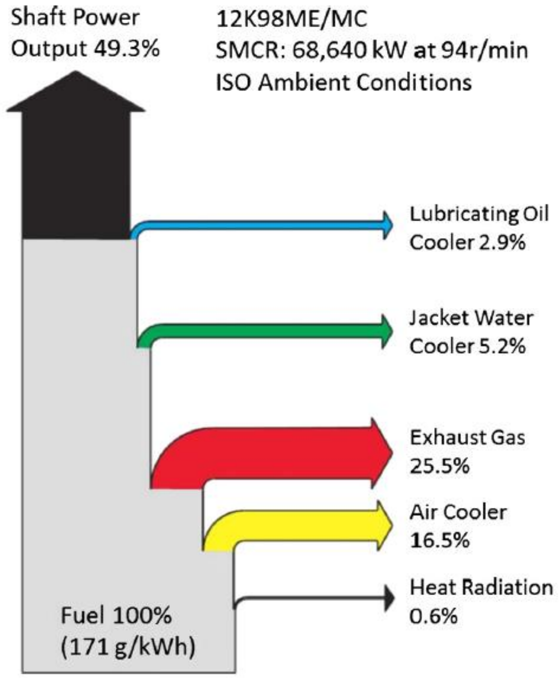


Figure 1: Heat balance diagram for MAN 12K98ME/MC marine diesel engine operating at 100 SMCR under ISO conditions [2]

While this application is not the subject of the proposed study, the breakdown of energy availability is likely to be very similar, though the quantities, temperature, and quality of those heat sources may diverge from the diagram above.

“Engine cooling water temperatures of 80-90 ^{o}C are fairly standard for most engines [2].” While exhaust gas appears to be the most lucrative source of waste thermal energy due to its comparatively high temperature, there are several complications which, though not as important for the Pederson and Singh study, would prevent its being as lucrative in an automotive application. Some of these factors are the increase in exhaust back pressure on the engine, the cooling of exhaust gases below the dew point of steam which could result in liquid water in the exhaust system causing corrosion, and reduced efficacy of reactions in the catalytic converter due to sub-optimal temperatures and high pressures caused by the heat harvesting system.

“Many gaseous waste heat streams are discharged at near-atmospheric pressure (limiting the ability to transport them to and through equipment without additional energy input).” [3] Harvesting heat from the coolant loop relieves several of these concerns, the coolant from which the waste heat will be recovered is already being circulated through the engine and through the radiator for the purpose of cooling the engine. It is plausible that WHR system need not impose an additional requirement for energy to circulate this heat source.

There are several options for low quality waste heat recovery such as thermoelectric and piezoelectric generation which both show promise because of their low technical complexity and long service life due to the lack of moving components. Unfortunately, both of those strategies promise very low efficiency and power yield for a small automotive application and require expensive materials for fabrication.

There is another category of waste heat recovery that is much more common, vapor power cycles. Because of the low temperature and quality of waste heat in automotive applications, the literature on the subject indicates that the optimal cycle for recovering that energy is the Rankine cycle. Specifically, two variations of the Rankine cycle: The Organic Rankine cycle and the Kalina cycle. These variations share the advantage of working fluids with comparatively low phase transition temperatures.

Low waste heat temperatures require large heat exchangers in comparison to traditional steam power generation applications. This is because heat transfer is proportional to the temperature difference between the two fluids between which heat is being transferred. Though the project will not strictly limit the size of the application, the size of the heat exchangers was a point of discussion.

There is further, related upside to using organic compounds as a working fluid. One of the features of organic molecules is their relatively high molecular mass. In comparison with water vapor, the fluids used in ORCs have a higher molecular mass, enabling more compact designs, a higher mass flow rate, and higher turbine efficiencies. However, since the cycle functions at lower working temperatures, the overall efficiency is only around 10%-20%, depending on the precise temperature of the condenser and of the evaporator.

The organic Rankine cycle is so named for the organic molecules that make up typical working fluids – hydrocarbons and refrigerants that are typically used in those cycles. ORCs have the same configuration as traditional steam Rankine cycles but make use of hydrochlorofluorocarbons (HCFCs), fossil fuels such as propane and cyclopentane, benzene, cyclohexane and isobutane, and refrigerants such as R134a, R22, R245fa, R236ea, R152a, R123, R600a, R600, R290 and R227ea as working fluids [2] [4] [5] [4]. These working fluids make the ORC particularly well suited to low heat quality applications due to their high molecular weight and low phase transition temperatures.

A single stage turbine is typically used for an ORC [2] which greatly reduces the technical complexity and size of the turbine, both of which are positive features for an automotive application where physical space is crucial.

The Kalina cycle (KC) is a modified form of the Rankine cycle and has a better operating efficiency in certain applications. The use of non-azeotropic[1] working fluid mixtures in an ORC has often been proposed with the aim of reducing thermal irreversibilities; particularly those occurring between the heat source and the evaporating working fluid. The KC is typically implemented with a water/ammonia mixture as the working fluid. The ratio of that mixture is dynamically varied depending on the temperature of the heat source. [6]

Within the temperature range 200^{o}C - 400^{o}C, the Kalina cycle is 20% - 40% more efficient than a standard Rankine cycle. Some studies indicated that the Kalina cycle had better thermodynamic performance than the organic Rankine cycle as well. Because of the ammonia-water mixture used for a working fluid and the industry's long experience with both substances the Kalina cycle is also considered safe and environmentally friendly. This is not the case with some of the fossil

[1] Azeotrope: n: A liquid mixture that is characterized by a constant minimum or maximum boiling point which is lower or higher than that of any of the components.

fuels and refrigerants used in some organic Rankine cycles. [2] “For low temperature sources, the Kalina is often mentioned as an alternative for the ORC. Although the Kalina cycle is often called to be [sic] superior to the ORC, DiPippo [7] has shown that an existing Kalina cycle has about the same performance as existing ORCs.” [5] “Although the obtained useful powers are actually equal in value, the Kalina cycle requires a very high maximum pressure in order to obtain high thermodynamic performances. So, the adoption of Kalina cycle, at least for low power level and medium-high temperature thermal sources, seems not to be justified because the gain in performance with respect to a properly optimized ORC is very small and must be obtained with a complicated plant scheme, large surface heat exchangers and particular high pressure resistant and no-corrosion materials.” [6] In the quoted study, the recovered power calculated from two identical diesel engines was 1615 kW and 1603 kW for the Kalina and ORC cycles respectively.

The efficiency of a KC can be increased due to a close temperature match with heat transfer fluids in the evaporator and condenser. “For instance, a KC system using an ammonia-water mixture as the working fluid to generate power from waste heat of a gas turbine achieved a thermal efficiency of 32.8%.” [8] “Some studies showed that a KC can achieve a better thermal efficiency than ORC systems.” [8] “In practice, the expansion ratio of the turbine for KCS-34 is relatively high and a multi-stage turbine is required.” [8] This may be the drawback of the KC that makes ORC somewhat more attractive for an automotive application.

The KC also requires a substantial amount of control in order to adjust the mass-fraction of ammonia including density sensors and real-time monitoring of working fluid composition in some implementations making this cycle more complicated to implement. KC is generally used as a method of improving a conventional (steam) RC. In some of these cases, the KC is the only real choice for waste heat recovery. It also appears that the working temperatures are slightly higher than some refrigerant based ORCs. This makes the KC less appropriate for an automotive application where the heat source is less than 100^{o}C.

For the reasons outlined above, the ORC and the KC are the two leading competitors, and from those two, the ORC was selected for further study due to its comparable power output and efficiency that come without the drawbacks of system complexity and corrosion that are common in implementations of the Kalina cycle.

The goal of this project was to design a parameterized mathematical model of an ORC such that an automotive scale generator can be designed and built. This model will use one of several hydrocarbons or refrigerants discussed as a working fluid because the low quality of waste heat in the cooling system requires a working fluid with a low phase transition temperature. The selection of the working fluid will depend on the available waste heat temperature, which much match well with the phase transition temperature of the working fluid, and the slope of the saturated vapor line.

The slope of the saturated vapor line is important. A negative slope defines a “wet” working fluid, a vertical slope defines an isentropic working fluid, and a positive slope a “dry” working fluid. These names are given due the practical implications of the slope which is whether they will begin to condense in the isentropic expansion process; whether the line defining this portion of the cycle travels into, out of, or along the saturated vapor line. Because dry fluids tend not to damage turbine blades they are preferred. An isentropic fluid that does not enter the super-heated region is still more ideal, and R245fa, pentafluoropropane, ($C_3H_3F_5$) fits this description which is why it is a popular choice for organic Rankine cycle implementations. This fluid does have its own drawbacks however, it is an HFC and is considered a greenhouse gas.

There are several viable working fluids for the organic Rankine cycle. The usage of each depends on a number of factors. Foremost among those factors are the working temperatures and pressures of the desired system.

In a traditional Rankine cycle, the working fluid used is water. That is not an appropriate selection for an automotive application due to the comparatively low temperature of the waste heat utilized.

R134a appeared to be the most suitable for small scale solar applications; though R152a, R600a, R600 and R290 were promising though they required handling precautions due to their flammability. Isobutane showed improved system performance when compared to R123 and R245fa. [9] Reading through the literature review in M.A. Khatita et al. did not show consensus on the best working fluid for ORC, however. With most fluids the use of a regenerative ORC instead of the basic cycle reduced the irreversibility of a solar ORC. Additionally, at the two temperature ranges studied fluids with higher molecular complexity resulted in more effective regenerative cycles with the exception of cyclo-hydrocarbons. [9] This was primarily due to higher turbine efficiency and increased mass flow rates.

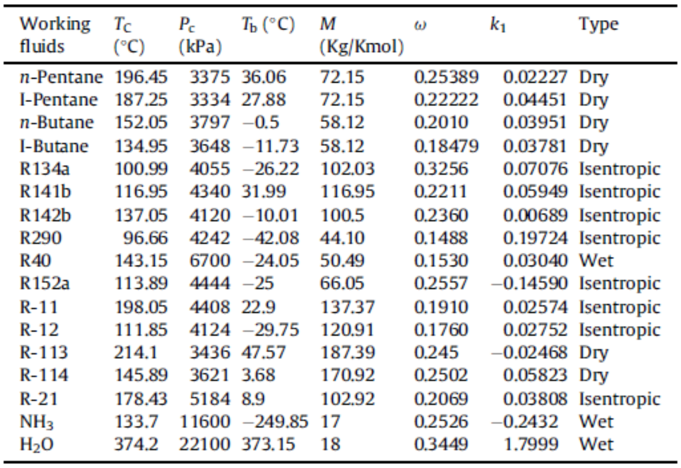
"Unlike water, most organic fluids suffer chemical decomposition and deterioration at high temperatures and pressures." [10] These working fluids eventually foul the heat exchanger passages through which they pass, inhibiting heat transfer from the that source to the working fluid and from the working fluid to the heat sink. This is often referred to as a fouling factor and is an inefficiency that will be considered in the proposed model. ORC systems showed efficiency gains with higher turbine inlet pressures, and efficiency losses for higher condenser outlet temperatures. This suggests that operating conditions, primarily temperature, could have a significant effect on the subsequent fouling of the heat exchangers and the efficiency of a given system. [10] [3]

In addition to the fouling on the working fluid side of the heat exchangers, the heat source side of the heat exchangers can also experience detrimental fouling if the exhaust stream from the vehicle is used as a heat source. This is the reason why, when considering which heat source to model, the cooling loop that exists in the engine compartment was chosen. This is another area in which the proposed project can offer an improvement on current technology. Since the waste heat is harvested from a closed cooling loop, the only fouling potential that exists is from the decomposition of the coolant itself. This is much less than what might be present in an open system where compounds could precipitate from the waste heat stream as it is cooled. The system architecture being considered for this project assumes that there will be a heat exchanger through which the heat source (engine coolant) and the working fluid of the simulated cycle will pass without directly interacting with one another. There is another benefit to this type of heat exchange. Because of the flammable nature of many organic compounds, separating the working fluid from the heat source also reduces fire hazards that might otherwise arise in a system of this kind. Because of the lack of oxygen in a closed loop of working fluid there is no possibility of fire without some sort of leak. There is further advantage to using the cooling loop of the vehicle as a heat source with respect to this design consideration in that there is no oxygen present in either system, and therefore reduced risk even in the case of a leak between the coolant system and the waste heat recovery system.

A final consideration of the working fluid selection is flammability. Many organic fluids (hydrocarbons) are flammable and may not be suitable for exposure to waste heat of certain temperatures or in applications where exposure to open flame is likely. An automotive application presents both of these hazards and so hydrocarbons are not good choices for this type of application.

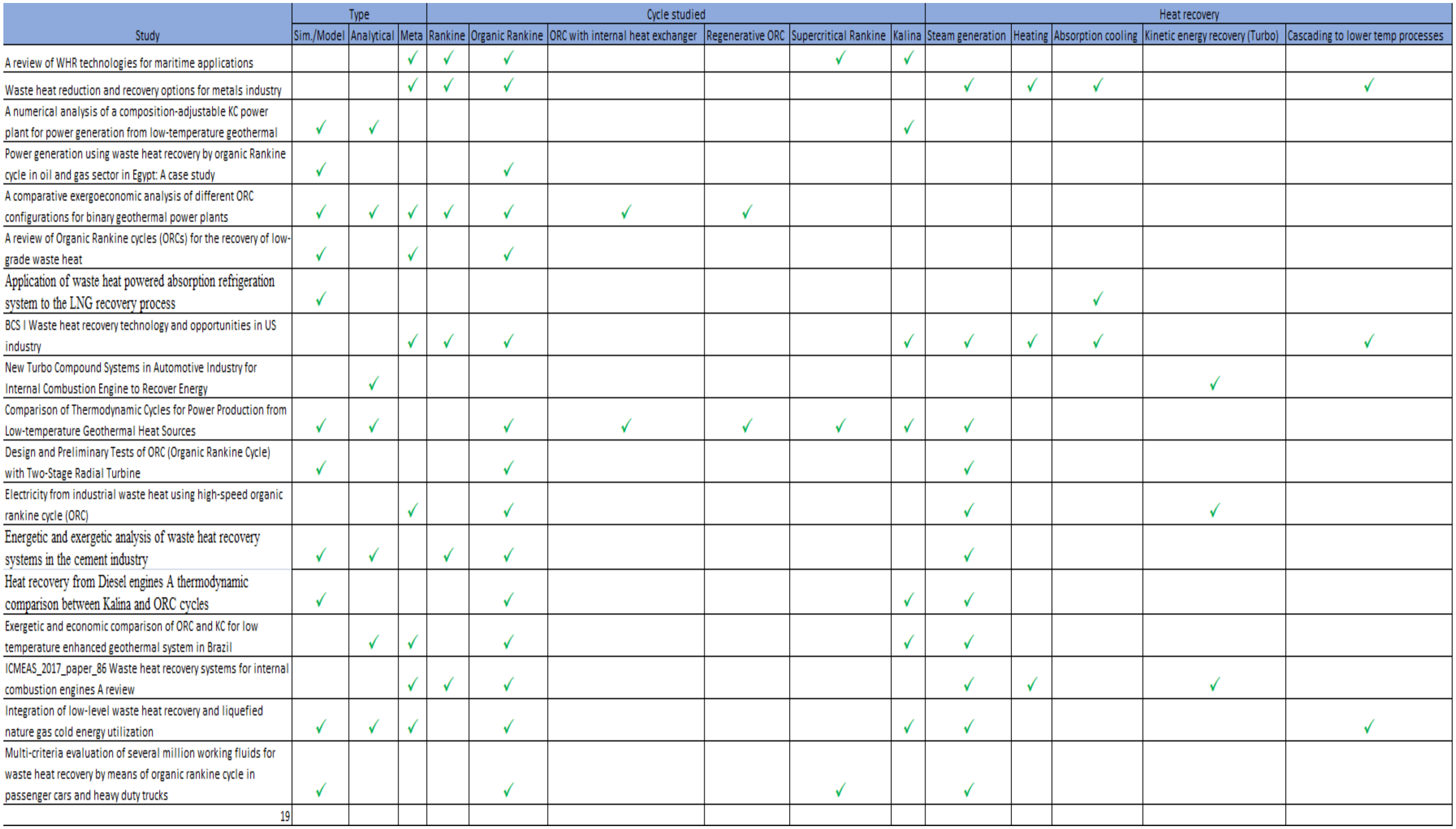
An analysis of a much more exhaustive list of working fluids used in an ORC was contained in the paper Exergetic and economic comparison of ORC and KC for low temperature enhanced geothermal system in Brasil. [11] This analysis included several fluids that are not commonly considered in the literature and may be worth consideration in future studies. Some of those fluids are contained in Table 1.

Table 3: Thermodynamic parameters of the considered pure working fluids [11]



Working fluids	T_C (°C)	P_c (kPa)	T_b (°C)	M (Kg/Kmol)	ω	k_1	Type
n-Pentane	196.45	3375	36.06	72.15	0.25389	0.02227	Dry
I-Pentane	187.25	3334	27.88	72.15	0.22222	0.04451	Dry
n-Butane	152.05	3797	−0.5	58.12	0.2010	0.03951	Dry
I-Butane	134.95	3648	−11.73	58.12	0.18479	0.03781	Dry
R134a	100.99	4055	−26.22	102.03	0.3256	0.07076	Isentropic
R141b	116.95	4340	31.99	116.95	0.2211	0.05949	Isentropic
R142b	137.05	4120	−10.01	100.5	0.2360	0.00689	Isentropic
R290	96.66	4242	−42.08	44.10	0.1488	0.19724	Isentropic
R40	143.15	6700	−24.05	50.49	0.1530	0.03040	Wet
R152a	113.89	4444	−25	66.05	0.2557	−0.14590	Isentropic
R-11	198.05	4408	22.9	137.37	0.1910	0.02574	Isentropic
R-12	111.85	4124	−29.75	120.91	0.1760	0.02752	Isentropic
R-113	214.1	3436	47.57	187.39	0.245	−0.02468	Dry
R-114	145.89	3621	3.68	170.92	0.2502	0.05823	Dry
R-21	178.43	5184	8.9	102.92	0.2069	0.03808	Isentropic
NH_3	133.7	11600	−249.85	17	0.2526	−0.2432	Wet
H_2O	374.2	22100	373.15	18	0.3449	1.7999	Wet

For this study, an organic Rankine cycle was selected using the working fluid R245fa because the phase transition temperatures are appropriate for harvesting waste heat from an automotive cooling loop, and R245fa is an isentropic working fluid with a high molecular mass. Additionally, it is reasonably easy to obtain and doesn’t have many environmental or health impacts to consider.



Study	Type			Cycle studied						Heat recovery				
	Sim./Model	Analytical	Meta	Rankine	Organic Rankine	ORC with internal heat exchanger	Regenerative ORC	Supercritical Rankine	Kalina	Steam generation	Heating	Absorption cooling	Kinetic energy recovery (Turbo)	Cascading to lower temp processes
A review of WHR technologies for maritime applications			✓	✓	✓			✓	✓					
Waste heat reduction and recovery options for metals industry			✓	✓	✓					✓	✓	✓		✓
A numerical analysis of a composition-adjustable KC power plant for power generation from low-temperature geothermal	✓	✓							✓					
Power generation using waste heat recovery by organic Rankine cycle in oil and gas sector in Egypt: A case study	✓				✓									
A comparative exergoeconomic analysis of different ORC configurations for binary geothermal power plants	✓	✓	✓	✓	✓	✓	✓							
A review of Organic Rankine cycles (ORCs) for the recovery of low-grade waste heat	✓		✓		✓									
Application of waste heat powered absorption refrigeration system to the LNG recovery process	✓											✓		
BCS I Waste heat recovery technology and opportunities in US industry			✓	✓	✓				✓	✓	✓	✓		✓
New Turbo Compound Systems in Automotive Industry for Internal Combustion Engine to Recover Energy		✓											✓	
Comparison of Thermodynamic Cycles for Power Production from Low-temperature Geothermal Heat Sources	✓	✓			✓	✓	✓	✓	✓	✓				
Design and Preliminary Tests of ORC (Organic Rankine Cycle) with Two-Stage Radial Turbine	✓				✓					✓				
Electricity from industrial waste heat using high-speed organic rankine cycle (ORC)			✓		✓					✓			✓	
Energetic and exergetic analysis of waste heat recovery systems in the cement industry	✓	✓		✓	✓					✓				
Heat recovery from Diesel engines A thermodynamic comparison between Kalina and ORC cycles	✓				✓				✓	✓				
Exergetic and economic comparison of ORC and KC for low temperature enhanced geothermal system in Brazil		✓	✓		✓				✓	✓				
ICMEAS_2017_paper_86 Waste heat recovery systems for internal combustion engines A review			✓	✓	✓					✓	✓		✓	
Integration of low-level waste heat recovery and liquefied nature gas cold energy utilization	✓	✓	✓		✓				✓	✓				✓
Multi-criteria evaluation of several million working fluids for waste heat recovery by means of organic rankine cycle in passenger cars and heavy duty trucks	✓				✓			✓		✓				
19														

4. Literature review

4.1. Theoretical background

Conductive heat transfer

Heat transfer by conduction is the process by which energy is transferred from energetic molecules to adjacent, less energetic molecules due to interactions between particles. In this form of heat transfer, there is no bulk transfer of material.

$$\dot{Q}_x = -\kappa A \left(\frac{dT}{dx}\right)$$

Equation 1: Conductive heat transfer - Fourier's law

Where $\dot{Q}_x$(W) is the rate of heat transfer across a plane normal to a direction x (m) with area A (m^2). κ is a property of the material through which the energy is transferred referred to as the thermal conductivity of that material (kJ/(kgK)).

Radiative heat transfer

Emitted radiative heat transfer is proportional to the fourth power of the absolute temperature of a given surface, T_b. Note that this equation considers only a single surface.

$$\dot{Q}_e = \epsilon \sigma A T_b^4$$

Equation 2: Radiative heat transfer - Stefan-Botzmann law

Radiative heat transfer between a large surrounding surface of absolute temperature T_s and a smaller surface at absolute temperature T_b (K) can be found by the following equation, where $\dot{Q}_e$ is the rate at which energy is emitted from a surface, ε is the property of a surface that indicates how effectively the surface radiates heat. It is a number between 0 and 1. σ is the Stefan-Boltzmann constant shown in Table 1.

$$\dot{Q}_e = \epsilon\sigma A(T_b^4 - T_s^4)$$

Equation 3: Radiative heat transfer between two bodies

Where T_s is the absolute temperature of the second body (K).

Convective heat transfer

Heat transfer from a solid surface to an adjacent fluid (liquid or gas) is referred to as convection.

$$\dot{Q}_c = hA\left(T_b - T_f\right)$$

Equation 4: Convective heat transfer between a solid and an adjacent fluid

Where $\dot{Q}_c$ (W) is the heat transfer rate from the solid to the fluid, T_b (K) is the temperature of the surface, T_f (K) is the temperature of the fluid, and h is the heat transfer coefficient (W/(m^{2}K)). Unlike conductivity and radiation, bulk movement of mass can take place with convection. Whether or not this bulk motion is a result of external devices such as fans or blowers or is a result of buoyancy forces resulting from the changing density of the heated fluid constitutes the difference between forced and free convection respectively.

The first law of thermodynamics

The first law of thermodynamics states that the energy in a closed system is conserved. Mathematically speaking this means that

$$\Delta KE + \Delta PE + \Delta U = (Q_{in} - Q_{out}) - (W_{out} - W_{in})$$

Equation 5: The first law of thermodynamics

Where KE is the kinetic energy of the system (J), PE is the potential energy of the system (J) and U is the internal energy of the system (J); Q is the heat transfer into the system (J) and W is the work performed by the system (J). The subscript "in" denotes heat transfer into the system, or work done on the system, and the subscript "out" denotes heat transfer out of the system or work done by the system.

In a power cycle such as the one that will be analyzed by this study, the system returns to its initial state after every cycle. Because of this, the change in energy is considered to be 0.

Power cycles

Because the change in energy in a power cycle is considered zero from one iteration of the cycle to the next the following expression can be derived from Equation 5:

$$Q_{in} - Q_{out} = W_{out} - W_{in}$$

Equation 6: Power cycle

And the thermal efficiency of the power cycle can be found by the ratio of the net work produced by the system to the heat transfer into the system.

$$\eta_{th} = \frac{W_{out} - W_{in}}{Q_{in}}$$

Equation 7: Power cycle efficiency

Where η_{th} is the thermal efficiency of the cycle; a number between 0 and 1. In the development of a power cycle a tool called a p-v diagram is commonly utilized. This diagram is unique for any given working fluid and it shows the phase of the working fluid at a given pressure and specific volume. On the diagram, there is a zone in which the vapor and liquid phases simultaneously exist. On the low specific volume boundary of this zone is the saturated liquid line. Departing from this line away from the dome results in a working fluid that is in the compressed (subcooled) liquid state only.

On the high specific volume boundary of this zone is the saturated vapor line. Departing from this line away from the dome results in a working fluid that is exclusively in the superheated vapor state.

Traversing across the vapor dome along the specific volume axis represents a phase change from saturated liquid to saturated vapor or vice-versa at a constant temperature and constant pressure. In other words, as heat is added to a saturated liquid at a constant pressure, the liquid will transition into a vapor without any increase in temperature, but with a rapid increase of specific volume. Between the saturated liquid and saturated vapor states there exists a liquid-vapor mixture. The ratio of vapor mass to the total mass of the mixture is referred to as the mixtures "quality" and is a quantity describing the progress of the phase transition.

$$x = \frac{m_{vapor}}{m_{liquid} + m_{vapor}}$$

Equation 8: Mixture quality

Where x is the mixture quality and m is the mass of the specified phase. x ranges between 0 and 1 where 0 is a saturated liquid and 1 is a saturated vapor. x is only defined for a saturated mixture of liquid and vapor, the area represented underneath the vapor dome.

The specific volume of a saturated liquid is often several orders of magnitude lower than that of a saturated vapor. This fact allows for two useful processes. The first is that the rapidly expanding mass of working fluid can be used to drive a turbine, performing useful work. The second is that the much denser liquid working fluid can be very efficiently pumped back to a boiler due to its much lower volume.

Take the following case for example:

There is a 12mm diameter cylinder filled with 3mm of saturated liquid water with a piston on top. The piston has a mass of 0.56kg and the air on the other side of the piston is standard air at sea level.

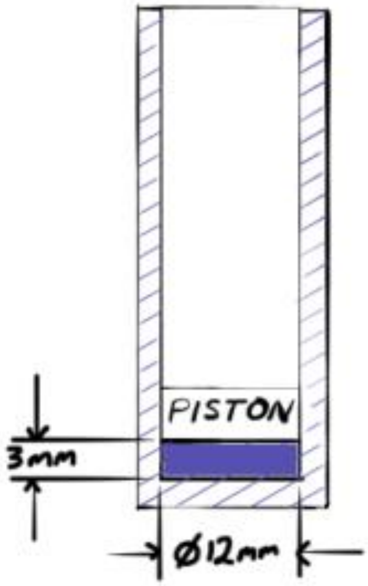


Figure 2: Piston with saturated liquid water

Heat is added to the water until all of the water has evaporated and what remains in the cylinder below the piston is a saturated vapor. The height of the piston can then be calculated by the following means.

First, a free body diagram of the piston yields the following equation which provides the initial pressure of the saturated liquid below the piston.

$$0 = [p_{atm} * (\pi * 0.006^2) + 0.56 * 9.81] - p_{water} * (\pi * 0.006^2)$$

Equation 9: Newtons second law applied to the free body diagram of the piston

The pressure of the atmosphere at sea level is 101325Pa, therefore the pressure of the saturated liquid is 150000Pa. Table A-3 from *Fundamentals of Engineering Thermodynamics* [12] yields the specific volume of saturated liquid and of saturated vapor at that pressure.

$$v_{liquid} = 1.5028e - 3\,\frac{m^3}{kg} \qquad v_{vapor} = 1.159\,\frac{m^3}{kg}$$

Equation 10: Specific volumes of saturated water at liquid and vapor states

Because the mass of the water in the system remains constant, the following relationship holds:

$$m = \frac{V}{v}$$

Equation 11: Relationship between mass, volume and specific volume

$$m_{liquid} = m_{vapor}\,\frac{V_{liquid}}{v_{liquid}} = \frac{V_{vapor}}{v_{vapor}}$$

Equation 12: Volume to specific volume ratio at saturated liquid and saturated vapor states

The volume of the saturated liquid can be found using the dimensions on Figure 2.

$$V_{vapor} = \frac{(0.003 * 0.006^2 * \pi)}{1.5028e - 3} * 1.159$$

Equation 13: Vapor volume

And finally, the height of the volume contained by the piston can be found by dividing that volume by the cross-sectional area of the cylinder yielding 2314mm.

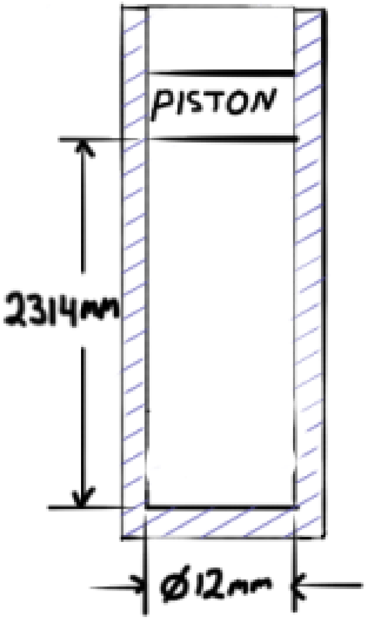


Figure 3: Piston with saturated vapor water

This is a dramatic increase in volume; much more than what is achieved by adding heat to a fluid without its changing phase. This fact is what the Rankine cycle seeks to take advantage of for the generation of power by using this expansion to do useful work on a turbine.

Ideal Rankine cycle

The Rankine cycle is one of the practical ways in which the preceding facts can be utilized to generate power; in the case of this project, for the generation of electric power.

An ideal Rankine cycle contains the following 4 processes:

1. Isentropic expansion of the working fluid through the turbine passing the working fluid to the condenser shown as EF in the figure below. Note that because DE departs the saturated vapor line, E represents a superheated vapor rather than a saturated vapor.

2. Isobaric heat transfer from the working fluid to the surroundings from the condenser resulting in a saturated liquid represented below by the line FA.

3. Isentropic compression in the pump to a compressed liquid passed to the boiler. Represented by the line AB.

4. Isobaric heat transfer to the working fluid from the heat source resulting in a saturated vapor ready to begin the cycle again. This is represented by the line BCDE

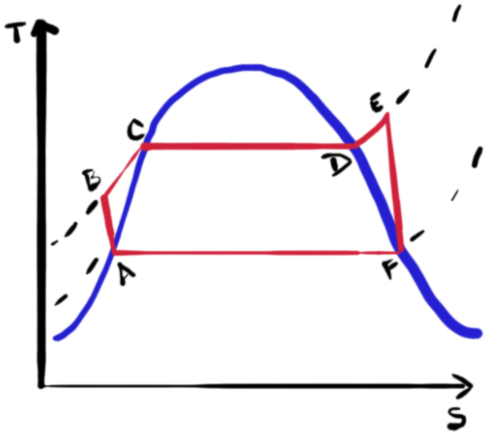


Figure 4: T-s diagram for an ideal Rankine cycle

Boiler

The boiler is a device in which heat from a source at a constant pressure, in this case waste heat from some other process is transferred to a given working fluid. In this device, the pressure of the working fluid is controlled with the pump such that the heat transferred to the fluid causes it to change phase from a liquid to a saturated vapor. The temperature of the fluid in the boiler remains constant as the phase change takes place. The term super-heat refers to any additional heat that is added to the fluid beyond the heat necessary to complete the phase change which will result in rising temperature in the working fluid. This is done to prevent the working fluid from condensing in the turbine, potentially damaging the blades. It should be noted that quality is not defined for a superheated vapor as there is no mixture of phases in this state.

Turbine:

The turbine facilitates isentropic expansion, meaning that the entropy prior to and after expansion through the turbine is equal. This assumption is not valid in practice, and a non-isentropic turbine was modelled by an isentropic efficiency that was input from the user. Reasonable values were determined by experiment in previous, referenced studies. For dry working fluids it is possible that a saturated vapor at the inlet of the turbine will move into the super-heated region during this expansion process.

Condenser

The condenser is a device in which heat from the working fluid is rejected to the environment at a constant pressure such that the working fluid condenses from a vapor to a saturated liquid. The temperature of the fluid in the condenser remains constant as the phase change takes place. In some cases, the liquid may enter the subcooled (compressed liquid) state.

Pump

The pump returns subcooled liquid from the condenser to the boiler. Because of the much higher density of the working fluid in the liquid state, this process takes a comparatively small amount of work with respect to the work produced by the turbine. The pump process was also assumed to be isentropic, and an isentropic efficiency between 0 and 1 was applied to the model such that a non-isentropic process was approximated.

4.2. Cycle evaluation

There are many ways to evaluate the efficacy of an Organic Rankine cycle and across the literature many of these are employed. It is also not uncommon for the authors of a given paper to develop their own fitness criteria and use that to evaluate competing system designs. One common way in which waste heat recovery systems are compared is by comparing efficiency. Maximum theoretical efficiency is given by the following equation (known as the Carnot cycle).

$$\eta_{th} = \left(1 - \frac{T_C}{T_H}\right)$$

Equation 14: Maximum work potential

Where η_{th} is the thermal efficiency of the given power cycle, T_c is the absolute temperature (K) of the cold reservoir and T_H is the absolute temperature (K) of the hot reservoir. This efficiency is not attainable in practice. However, it represents a theoretical maximum given the source and sink temperatures of a given system and is often used as an initial evaluation of the feasibility of a given application.

Other methods of comparison include parameters such as system cost, research and development time and cost, system complexity and maintenance, and other recurring factors that may offset, in some cases entirely, the benefit of adding a waste heat recovery system to the process under consideration. This study considered power output, efficiency and system size.

5. Plan of study

There were professional objectives for this course of study. They were summarized in this section and expounded upon in the following sections. The industry objectives for this study were to design a tool with which to develop design specifications for an automotive waste heat recovery system that can be utilized on a light truck or similar platform. The tool allows for the manipulation of various parameters of a Rankine cycle and estimates the power output of a system conforming to those parameters. The parameters chosen for manipulation are justified in the following sections and supported by the literature review. The parameters which were studied for this application are shown in the table below.

Table 4: Study parameters

Parameter	**Cases**
Working fluid	R245fa
Boiler working pressure	0.5 MPa - 3.5 MPa
Condenser working pressure	0.0 MPa - 0.5 MPa
Pump volumetric flow rate	350 L/min – 570 L/min

The ultimate research objective of this study was to produce a model that can be manipulated to evaluate arbitrary attributes of an Organic Rankine cycle. That model will be used in future work to develop prototypes and eventually commercially viable products for my employer.

5.1. Develop a numerical model for iterative design of an organic Rankine cycle

A model will be developed using Python to simulate a Rankine cycle with the parameters specified by the user. This model will be used to iteratively evaluate cycles with parameters over a specified range. The following flow charts illustrate the program flow. Circular components are user inputs, square components are processes, diamonds are decision points, and parallelograms are data from an external source; in most cases the open source project CoolProp [13] which uses the same data bases as the commercial product REFPROP [14]. Both of these products contain critical and triple points for pure fluids and use equations for the thermodynamic and transport properties to calculate the state points of the fluid or mixture.

5.2. Develop cycle specifications using numerical model

Using the model described under the previous heading, ideal parameters can be developed for an arbitrary automotive application. The purpose of the model is to develop design specifications for an actual application which are to be used in a future project to build a prototype if those specifications prove feasible.

6. Method of numerical analysis

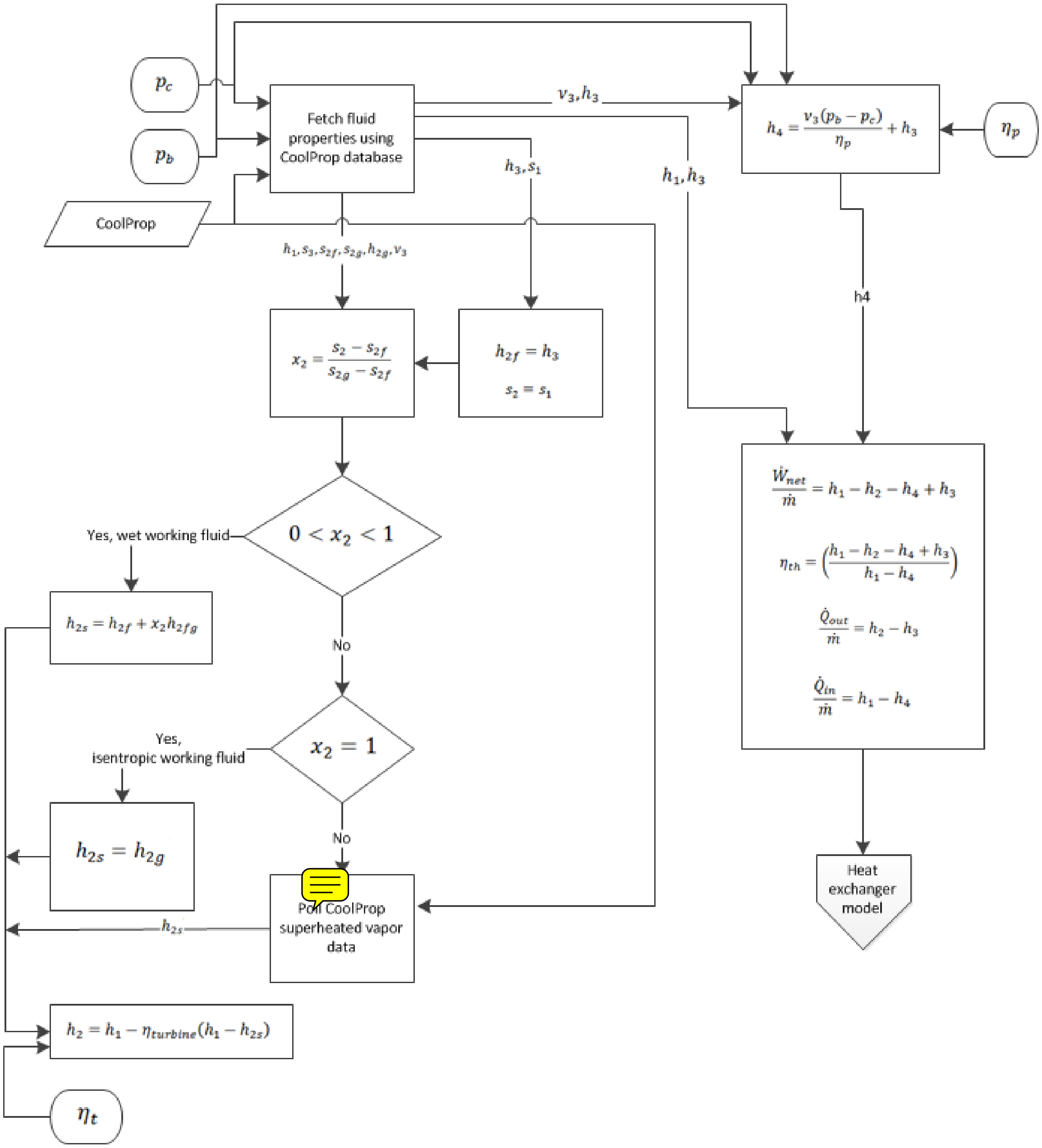

p_c
p_b
CoolProp
Fetch fluid properties using CoolProp database
v_3, h_3
h_3, s_1
h_1, h_3
$h_4 = \frac{v_3(p_b - p_c)}{\eta_p} + h_3$
η_p
h4
$h_1, s_3, s_{2f}, s_{2g}, h_{2g}, v_3$
$x_2 = \frac{s_2 - s_{2f}}{s_{2g} - s_{2f}}$
$h_{2f} = h_3$
$s_2 = s_1$
$\frac{\dot{W}_{net}}{\dot{m}} = h_1 - h_2 - h_4 + h_3$
$\eta_{th} = \left(\frac{h_1 - h_2 - h_4 + h_3}{h_1 - h_4}\right)$
$\frac{\dot{Q}_{out}}{\dot{m}} = h_2 - h_3$
$\frac{\dot{Q}_{in}}{\dot{m}} = h_1 - h_4$
Yes, wet working fluid
$0 < x_2 < 1$
$h_{2s} = h_{2f} + x_2 h_{2fg}$
No
Yes,
isentropic working fluid
$x_2 = 1$
$h_{2s} = h_{2g}$
No
Poll CoolProp superheated vapor data
h_{2s}
Heat exchanger model
$h_2 = h_1 - \eta_{turbine}(h_1 - h_{2s})$
η_t

Figure 5: Proposed study components diagram

6.1. Cycle component models

An ideal Rankine cycle can be modeled by the four principal components of the system. These models are all derived from the mass and energy rate balances for control volumes around the given component.

$$0 = \dot{Q}_{cv} - \dot{W}_t + \dot{m}\left[h_1 - h_2 + \frac{V_1^2 - V_2^2}{2} + g(z_1 - z_2)\right]$$

Equation 15:Energy rate balance for a control volume

For all of the following models *Heat Transfer* [15] was used as a primary reference. For all models, the changes in potential and kinetic energy of the working fluid were ignored as the model is not informed of the physical orientation of the components of a candidate system with respect to each other or with respect to gravity. The diagram below shows the temperature of the working fluid with respect to specific entropy, commonly referred to as a “T-s diagram.” This figure shows a generic Rankine cycle using R245fa as the working fluid.

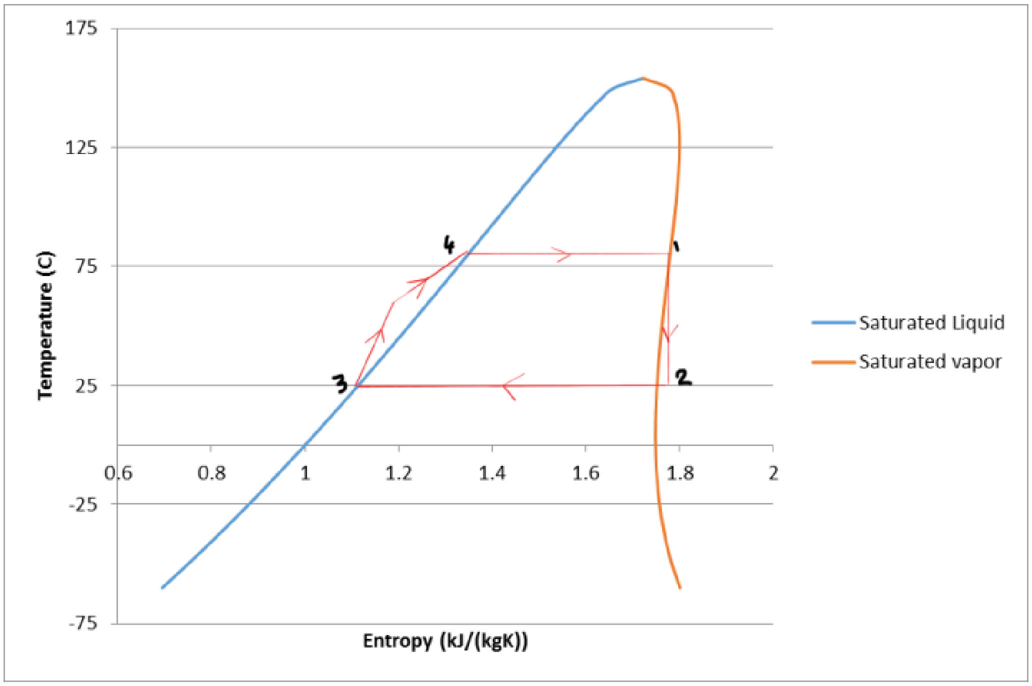


Figure 6: R245fa Rankine cycle

For each segment of the cycle shown in Figure 6, Equation 15 can be simplified to provide the model for the component responsible for that segment. It should also be noted that the area contained by the cycle on this chart has units of kJ/kg which means that energy per unit mass flow rate can be assessed visually using the area contained by a given cycle on the T-s diagram. The model for the turbine is:

$$\frac{\dot{W}_t}{\dot{m}} = h_1 - h_2$$

Equation 16: Turbine model

Where $\dot{m}$ is the mass flow rate of the working fluid, $\dot{W}_t$ is the rate of work produced by the turbine, and h is the enthalpy at the respective states of the working fluid as denoted in Figure 2. The specific enthalpy, specific entropy, and temperature at state 1 were retrieved from CoolProp using the boiler pressure (p_b) and retrieving saturated vapor properties. Using the assumption of an isentropic expansion process in the turbine, the flowing relationship holds:

$$s_{2s} = s_1$$

Equation 17: Specific entropy of the turbine outlet assuming isentropic expansion

If state 2 is found under the vapor dome, the enthalpy can be found by the following process:

$$h_{2s} = h_{2f} + x_{2s} h_{2fg}$$

Equation 18: Enthalpy of a vapor-liquid mixture

In which h_{2f} is the enthalpy of a saturated liquid. Note that this is also the specific enthalpy at state 3, h_3, and can be retrieved from CoolProp using the condenser pressure (p_c) and saturated liquid data. h_{2g} is the specific enthalpy of a saturated vapor at p_c, also retrieved from CoolProp. h_{fg} is the specific enthalpy required for evaporation and is the difference between h_{2g} and $h_{2f.}$ Finally, x_2 is the quality of the vapor-liquid mixture at state 2 which is given by the following:

$$x_{2s} = \frac{s_{2s} - s_{2f}}{s_{2g} - s_{2f}}$$

Equation 19: Vapor-liquid mixture quality[2]

[2] This equation is only valid for adiabatic, internally reversible processes. A different method of determining the enthalpy of a fluid at state 2 will be required for dry working fluids – Those working fluids with a positively sloped saturated vapor line.

Where s_2 is the entropy of state 2, s_{2g} is the entropy of a saturated vapor, and s_{2f} is the entropy of a saturated liquid. both were acquired using p_c and saturated vapor and liquid data respectively. The specific enthalpy at state 2 for a non-isentropic expansion is then found by the following:

$$h_2 = h_1 - \eta_{turbine}(h_1 - h_{2s})$$

Equation 20: State 2 enthalpy adjusted for turbine irreversibilities

If state 2 is found on the saturated vapor line, h_{2s} is simply equal to h_{2g}. If state 2 is found in the super-heated region as was usually the case in this study, h_{2s} is retrieved from the CoolProp data base using s_2 and p_c respectively. While x_2 is not defined for super-heated or sub-cooled fluids, it was used as a heuristic in the model to determine in which region state 2 would fall.

$$\begin{cases} 0 < x_{2s} < 1: liquid\ vapor\ mixture \\ x_{2s} = 1: saturated\ vapor \\ x_{2s} > 1: super-heated\ vapor \end{cases}$$

Equation 21: Heuristic for state 2 region

The model for the condenser is:

$$\frac{\dot{Q}_{out}}{\dot{m}} = h_2 - h_3$$

Equation 22: Condenser model

Where $\dot{Q}_{out}$ is the heat exchange rate from the condenser to the surroundings. Finally, the specific enthalpy at state 4 was found using the following:

$$h_4 = \frac{\nu_3(p_b - p_c)}{\eta_p} + h_3$$

Equation 23: Specific enthalpy at state 4 given a non-isentropic pump process

Where ν_3 and p are the specific volume and the pressure respectively at the states indicated.

The pump model is then:

$$\frac{\dot{W}_p}{\dot{m}} = h_4 - h_3$$

Equation 24: Pump model

Where $\dot{m}$ is the mass flow rate of the working fluid, and $\dot{W}_p$ is the rate of work consumed by the pump. The following is the model for the boiler:

$$\frac{\dot{Q}_{in}}{\dot{m}} = h_1 - h_4$$

Equation 25: Boiler model

Where $\dot{Q}_{in}$ is the heat exchange rate from the waste heat source to the boiler.

There are several assumptions that are made for an engineering model of an ideal Rankine cycle.

1. Each component of the cycle is analyzed as a control volume at steady state.
2. The turbine and the pump operate adiabatically.
3. Kinetic and potential energy effects are negligible
4. Saturated vapor enters the turbine. Condensate exits the condenser as a saturated liquid.

Now that the enthalpy at all four states is known, the net work per unit mass flow and thermal efficiency of the cycle can be determined as follows:

$$\frac{\dot{W}_{net}}{\dot{m}} = h_1 - h_2 - h_4 + h_3$$

Equation 26: Rankine cycle net power production

$$\eta_{th} = \left(\frac{h_1 - h_2 - h_4 + h_3}{h_1 - h_4}\right)$$

Equation 27: Rankine cycle thermal efficiency

Note that these formulas are valid both for the ideal case, and the case with irreversibilities because the irreversibilities were considered when calculating the enthalpy at each of the states. The same is true for the overall thermal efficiency.

Python code was developed for the purpose of performing these calculations so that a range of working pressures for the boiler and condenser could be iterated over and optimized numerically. The following section will validate that code using an Organic Rankine cycle that was evaluated experimentally by L. Li, Y.T. Ge, & S.A. Tassou. [16]

The specified working pressures of the boiler and condenser will also fix the temperatures at those states which can be used in conjunction with the mass flow rate and specific heat of the working fluid to determine the necessary area of the heat exchangers for that particular case. Because the heat is being exchanged between two fluid flows, a crossflow or counter-flow heat exchanger model can be used to make these estimates. The following procedure is found in *Heat Transfer* [15].

$$\dot{Q} = \mathrm{UAF}\Delta T_m$$

Equation 28: Heat exchanger heat transfer equation

Where $\dot{Q}$ = heat transfer rate, U = Overall heat-transfer coefficient, A = surface area for heat transfer consistent with the definition of U, ΔT_m = suitable mean temperature difference across the heat exchanger, F = A correction factor for multiple tube and/or shell passes in the heat exchanger.

$$\dot{Q} = \frac{\dot{Q}}{\dot{m}}\dot{m}$$

Equation 29: Heat exchange given heat exchange per unit mass flow rate and flow rate

Equation 28 can be rearranged as follows to yield heat exchange area:

$$A = \frac{\dot{Q}}{UF\Delta T_m}$$

Equation 30: Heat exchange area

The mean temperature difference in the case of a heat exchanger of this type is found with the following equation known as the log-mean temperature difference.

$$\Delta T_m = \frac{(T_{h2} - T_{c2}) - (T_{h1} - T_{c1})}{\ln\left(\frac{T_{h2} - T_{c2}}{T_{h1} - T_{c1}}\right)}$$

Equation 31: Log mean temperature difference

Where T_{h1} is the temperature of the hot fluid at its inlet and T_{h2} is the temperature of the hot fluid at its outlet, and T_{c1} is the temperature of the cool fluid at its inlet and T_{c2} is the temperature of the cool fluid at its outlet. Because there is a phase change occurring in the boiler, the following relationship holds:

$$T_{c1} = T_{c2} = T_b$$

Equation 32: Equal inlet and outlet temperatures due to the phase change in the boiler

Where T_b is the working temperature of the boiler.

A correction factor was applied to Equation 28 which considers the configuration of the counter-flow heat exchanger which depends on the number of tube passes as well as the ratio of temperatures at the inlets and outlets as described by the following:

$$R = \frac{T_{h1} - T_{h2}}{T_{c2} - T_{c1}}$$

Equation 33: Temperature difference ratio

$$P = \frac{T_{h2} - T_{c1}}{T_{h1} - T_{c1}}$$

Equation 34: Tube pass factor

Using the two factors R and P, figure 10-8 from *Heat Transfer* [15] was used to determine the exchanger correction factor. However, a phase change is taking place in both of the heat exchangers involved with this project. And depending on whether the hot or the cool fluid is undergoing a phase change, either R or P will equal 0. In both cases, the correction factor is equal to 1 and therefore, Equation 28 remained unmodified.

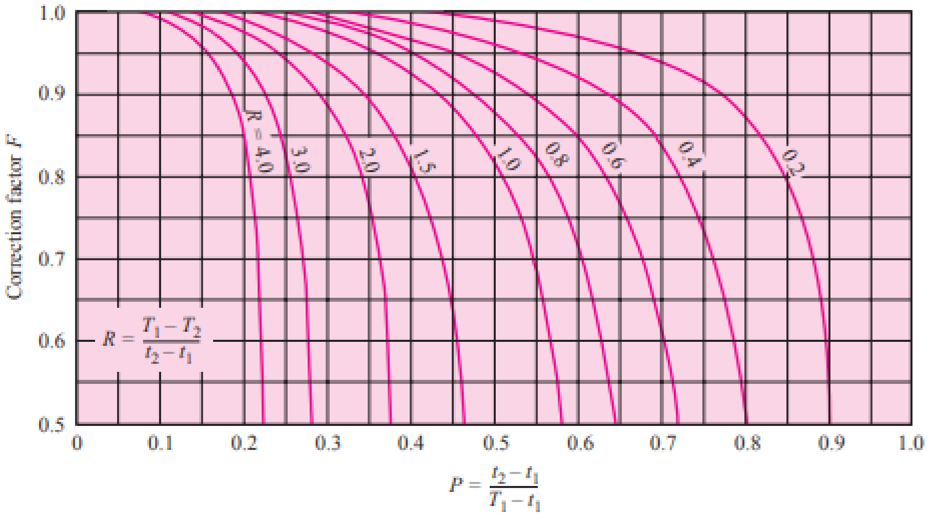


Figure 7: Figure 10-8 from Heat Transfer *[15]*

The heat transfer from the heat source working fluid to the working fluid of the Rankine cycle is equal assuming no external heat loss. Therefore, after iterating over possible boiler and condenser working pressures, the heat transfer into the cycle can be used to determine the temperature drop of the heat source fluid using the following equation:

$$\dot{Q} = \dot{m}_{ref}(h_1 - h_4) = (T_{h1} - T_{h2})c_h\dot{m}_h$$

Equation 35: Heat transfer in a heat exchanger

Where $\dot{Q}$ = the heat exchange from one fluid to the other, c_h = specific heat of the heat source fluid (isochoric liquid water/glycol mixture (50/50) = 3.7682 kJ/kgK), $\dot{m}_{ref}$ is the mass flow rate of the refrigerant in the Rankine cycle (kg/s), and $\dot{m}_h$ is the mass flow rate of the water/glycol. This equation can be rearranged in order to give the outlet temperature of the heat source as shown:

$$T_{h2} = T_{h1} - \frac{\dot{m}_{ref}(h_1 - h_4)}{\dot{m}_h c_h}$$

Equation 36: Boiler outlet temperature given refrigerant mass flow rate, heat source mass flow rate, heat source specific heat and heat source initial temperature

Iteration can be performed using Equation 35 to ensure that T_{h2} is still greater than the working temperature of the boiler; indicating that heat transfer from one fluid to the other is still occurring and that the design of the cycle doesn’t demand more heat than is available from the source. The figure for that model is below:

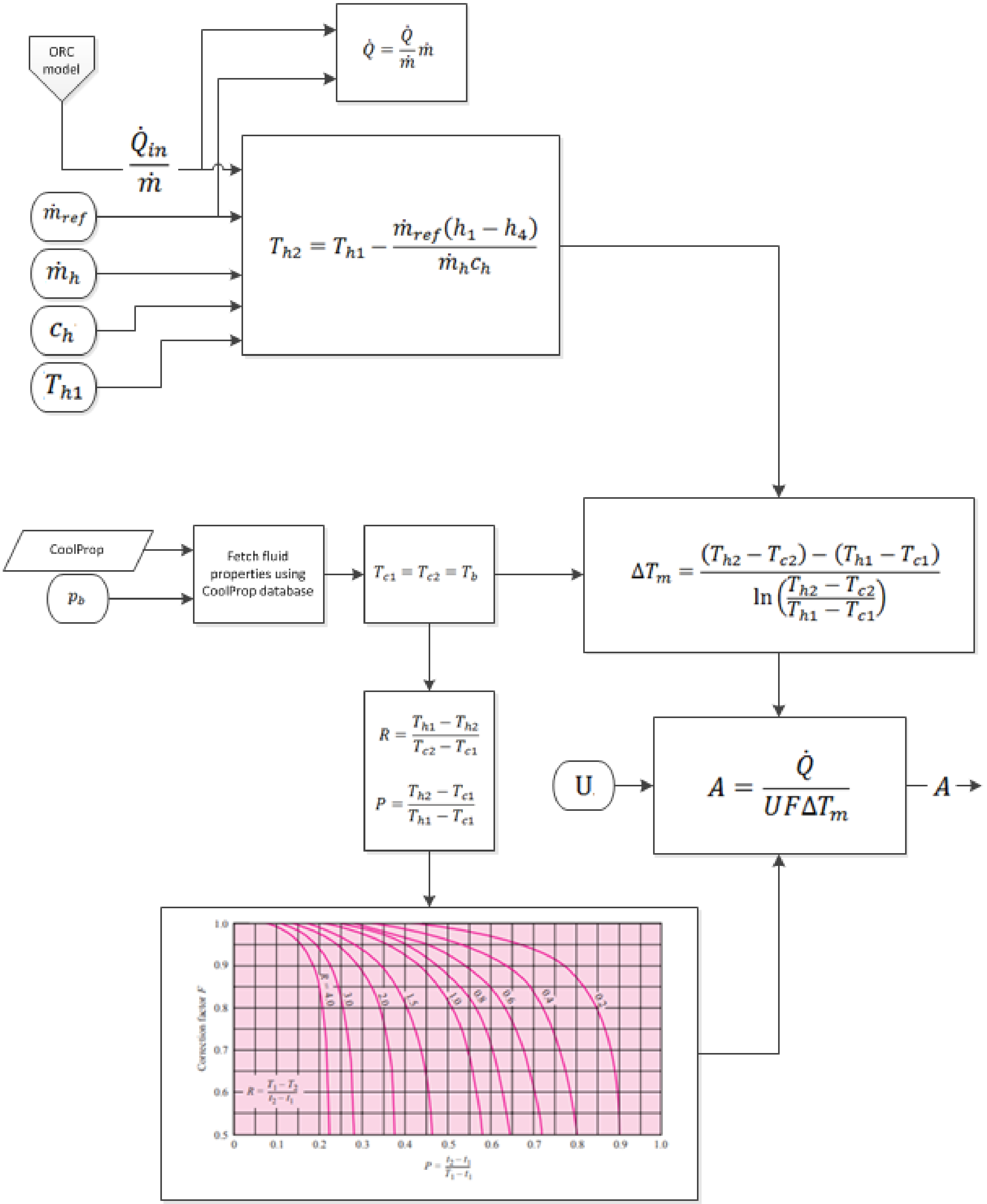

ORC model
$\frac{\dot{Q}_{in}}{\dot{m}}$
$\dot{Q} = \frac{\dot{Q}}{\dot{m}}\dot{m}$
$\dot{m}_{ref}$
$\dot{m}_h$
c_h
T_{h1}
$T_{h2} = T_{h1} - \frac{\dot{m}_{ref}(h_1 - h_4)}{\dot{m}_h c_h}$
CoolProp
p_b
Fetch fluid properties using CoolProp database
$T_{c1} = T_{c2} = T_b$
$\Delta T_m = \frac{(T_{h2} - T_{c2}) - (T_{h1} - T_{c1})}{\ln\left(\frac{T_{h2} - T_{c2}}{T_{h1} - T_{c1}}\right)}$
$R = \frac{T_{h1} - T_{h2}}{T_{c2} - T_{c1}}$
$P = \frac{T_{h2} - T_{c1}}{T_{h1} - T_{c1}}$
U
$A = \frac{\dot{Q}}{UF\Delta T_m}$
A
Correction factor F
$R = \frac{T_1 - T_2}{t_2 - t_1}$
$P = \frac{t_2 - t_1}{T_1 - t_1}$
R = 4.0
3.0
2.0
1.5
1.0
0.8
0.6
0.4
0.2
1.0
0.9
0.8
0.7
0.6
0.5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

Figure 8: Heat exchanger model flow diagram

In order to determine the heat exchange area, the overall heat-transfer coefficient must be determined, table 10-1 from *Heat Transfer* gives an approximate value for the overall heat transfer coefficient for various situations including a water-to-oil exchanger which approximates a coolant to R245fa situation like the one in this paper to be 110-350 W/m^2 ^{o}C. In order to obtain a conservative estimate for heat exchanger size, the lowest number in this range is used to obtain a maximum heat exchanger size; 110 W/m^2 ^{o}C. [15]

For a tube-in-shell heat exchanger, the heat exchange area is the outside surface of the tubes which can be found using the following:

$$A = n * L * \pi d$$

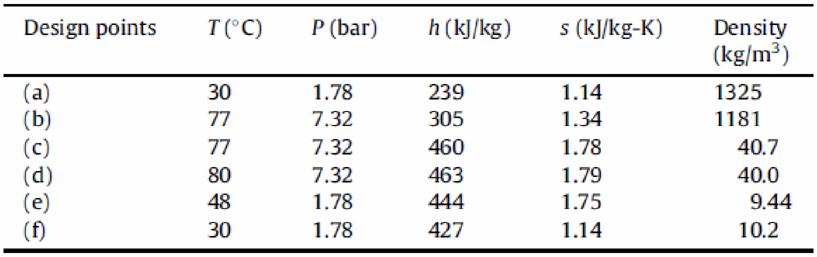
Equation 37: Heat exchange surface area for a shell in tube heat exchanger

Where n is the number of tubes, L is the length of the tubes, and is the diameter of the tubes.

6.2. Code validation

In an experimental study performed by Seok Hun Kang [17], the following cycle parameters were used to construct an organic Rankine cycle:

Table 5: Thermodynamic properties of the working fluid at the cycle design points [17]



Design points	T (°C)	P (bar)	h (kJ/kg)	s (kJ/kg-K)	Density (kg/m^3)
(a)	30	1.78	239	1.14	1325
(b)	77	7.32	305	1.34	1181
(c)	77	7.32	460	1.78	40.7
(d)	80	7.32	463	1.79	40.0
(e)	48	1.78	444	1.75	9.44
(f)	30	1.78	427	1.14	10.2

Where the state labels are derived from the following diagram:

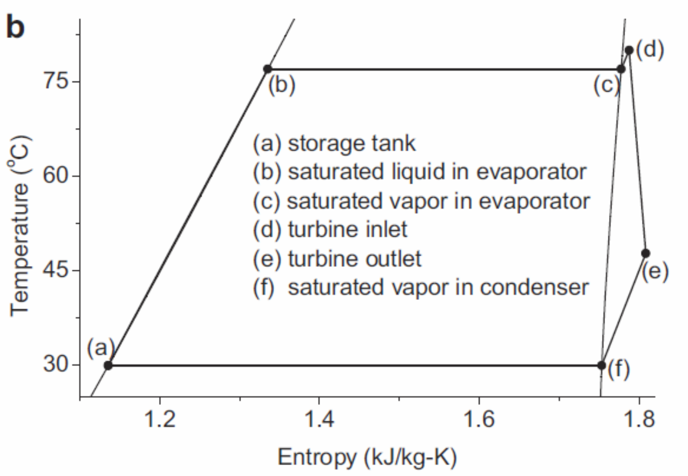
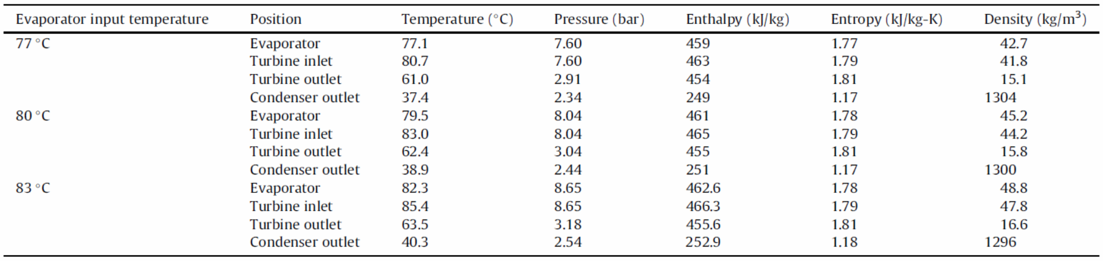


Figure 9: T-s diagram of the experimental cycle [17]

In the actual study, Kang observed different operating pressures however, and recorded them in the table below:

Table 6: Observed operating conditions of an ORC [17]



Evaporator input temperature	Position	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kg-K)	Density (kg/m^3)
77 °C	Evaporator	77.1	7.60	459	1.77	42.7
	Turbine inlet	80.7	7.60	463	1.79	41.8
	Turbine outlet	61.0	2.91	454	1.81	15.1
	Condenser outlet	37.4	2.34	249	1.17	1304
80 °C	Evaporator	79.5	8.04	461	1.78	45.2
	Turbine inlet	83.0	8.04	465	1.79	44.2
	Turbine outlet	62.4	3.04	455	1.81	15.8
	Condenser outlet	38.9	2.44	251	1.17	1300
83 °C	Evaporator	82.3	8.65	462.6	1.78	48.8
	Turbine inlet	85.4	8.65	466.3	1.79	47.8
	Turbine outlet	63.5	3.18	455.6	1.81	16.6
	Condenser outlet	40.3	2.54	252.9	1.18	1296

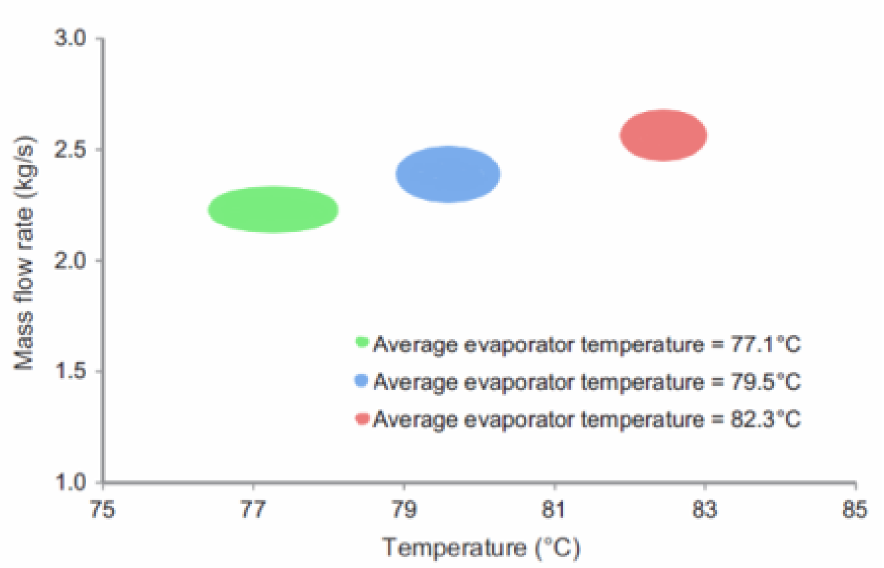


Figure 10: Mass flow rate observations for each evaporator input temperature test case [17]

Using the boiler pressure 8.65 bar (0.865 MPa) and the condenser pressure 2.54 bar (0.254 MPa) From the 83°C evaporator input temperature test case – which is closest to the likely heat source temperature for the automotive applications we desire to study – a mass flow rate of 2.5 kg/s which is approximately what was used for that test case, and a turbine efficiency of 0.787 which was published by Kang as the maximum achieved turbine efficiency; the following cycle parameters are found using the model developed in this study assuming a pump efficiency of 0.9:

Table 7: Model validation output

Input		Output	
Condenser pressure	0.254 MPa	Power per unit mass flow rate	44.5 kW
Boiler pressure	0.865 MPa	Efficiency	0.084
Turbine efficiency	0.787	Condenser temperature	40.55 °C
Pump efficiency	0.900	Boiler temperature	83.86 °C
Mass flow rate	2.5 kg/s		

Using an assumed efficiency of 90% for both the pump and the turbine, the power output of the cycle was predicted to be 44.5kW and the thermal efficiency of the cycle was predicted to be 8.4%.

The following figures show the published results obtained by Kang:

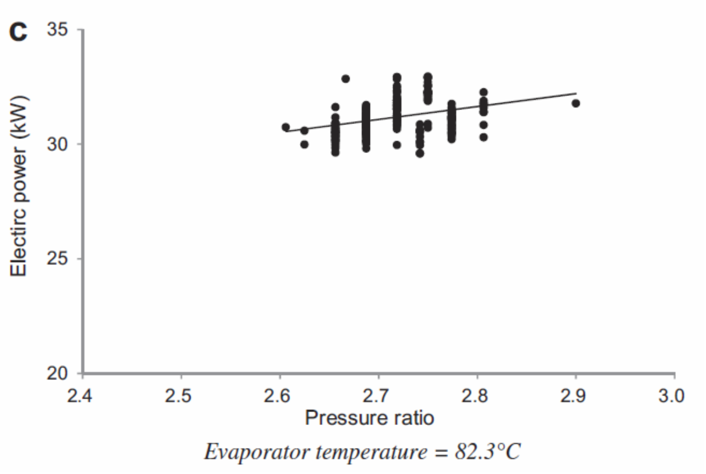


Figure 11: Electric power output for the experimental ORC [17]

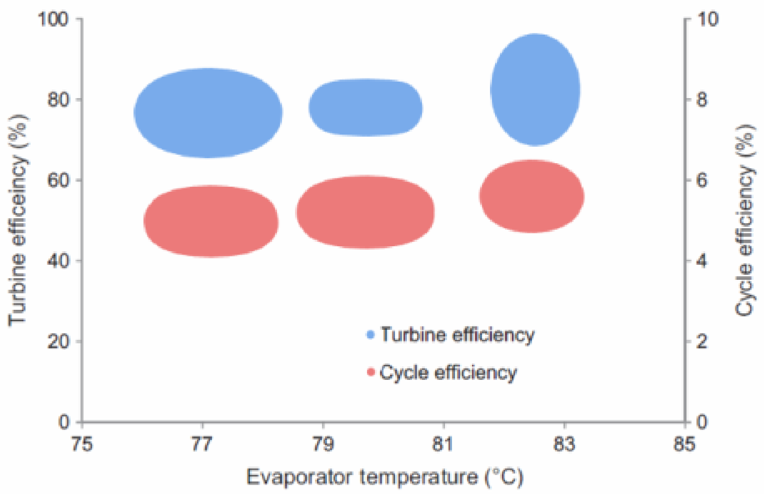


Figure 12: Turbine and cycle efficiencies [17]

So, the model developed in this study slightly over-estimated the power output of the ORC and correctly predicted the cycle efficiency within the margin of the published study. This could be a result of a bad assumption of pump efficiency or efficiency losses elsewhere in the system that were not modeled such as fouling in the heat exchangers which is common in systems that use refrigerants as a working fluid. There are also inefficiencies associated with the way the working fluid condenses in the condenser where the liquid can insulate the gas from further heat transfer.

For the purposes of this project, this is a good result. If a “best case” solution reveals that there is little promise in a design of this kind, a more sophisticated model is unlikely to yield more promising results.

7. Experimental design

Some of the parameters of the system cannot be independent for each of the subsystems. The parameter that is necessarily shared by all subsystems is the working fluid and the mass flow rate of the working fluid.

There are efficiency improving features of a Rankine cycle which could be explored. Boiler temperatures super, trans, and sub critical are all discussed in the literature; recuperators, secondary turbine stages, turbine bleeds, heated feedwater and many others are discussed. Most of these are beyond the scope of this study but will likely be included among the recommendations for further study as the literature seems to indicate that marginal gains in efficiency and power output can be achieved by the integration of some or all of these features. The following sections detail the parameters that are manipulated by the numerical model built for this study, and that model was used to generate all of the results in the following chapter using the equations specified in the previous chapter.

7.1. Boiler factors

In addition to the working pressure and temperature of the boiler, mass flow rates of the heat source, the working fluid from the automobile’s cooling system, and of the vapor power system can be manipulated to increase the rate of heat transfer. The interaction surface area of the heat exchanger can also be manipulated. Some of these factors are much easier to manipulate than others. So, while the model was developed such that any relevant parameter can be manipulated, this study will focus on the results of changing those that make the most practical and economic sense.

For the boiler, because this application requires a very small size with respect to traditional power generation layouts, the surface area of the heat exchanger is not a parameter that can be manipulated easily in practice. The temperature is also difficult to manipulate as the application is going to be integrated into an existing vehicle system which will have waste heat at a given temperature over which designers have very little control. The factor that does make sense to manipulate is the operating pressure.

7.2. Turbine factors

The number of stages, and turbine blade size, shape, and angle can all be manipulated to affect the isentropic efficiency of the turbine. The type of cycle in this case, the organic Rankine cycle, also has a large effect on the efficiency of this component of the system.

The literature review indicates that the turbine design selected for an organic Rankine cycle is usually a single stage turbine. This greatly simplifies the design space for the turbine. Blade size, shape and angle can be optimized for a given application and therefore are not parameters that are transparent to the user of the mathematical model as these parameters will be determined by the selection of other parameters.

These reasons make the turbine design an area in which there was not a lot of interesting work to be done in this project. Because of this, it was not a subject that was given much specific attention in this study.

7.3. Condenser factors

In addition to the working pressure and temperature of the condenser, the mass flow rate of the heat sink can be manipulated; though the mass flow rate of the heat sink, in this case the ambient air, is harder to manipulate. The interaction surface area of the heat exchanger can also be manipulated.

The condenser also represents a potential design challenge in an automotive application because the environment in which the system will be placed, the engine compartment, can be expected to have a highly variable temperature. It will also not be practically feasible to manipulate that temperature in any meaningful way.

The condenser factors face many of the same limitations as those of the boiler. Primarily that space in this application is a premium and therefore will likely be fixed. The condenser temperature is fixed to the ambient temperature which, in addition to being largely out of the control of this device, is somewhat variable depending on other heat sources and sinks in the engine compartment as well as operating conditions of the vehicle and the weather in which the vehicle is operating.

As a result, the parameter that is of interest to this study is primarily that of working pressure of the condenser.

7.4. Pump factors

The pump working pressure can be manipulated, and it may be possible to drive the pump directly with mechanical energy from the turbine shaft rather than electrically which could represent some efficiency gains. This area is also of particular interest to GHSP as they design and manufacture pumps currently and can be considered experts in this area.

Mass flow rate of the working fluid and the pressure of the boiler, which is the output pressure of the pump, are parameters of interest to this study. The pressure is explored in the preliminary results, but the mass flow rate is represented in the results as a variable quantity and will be one of the primary foci of the proposed study.

8. Results

8.1. Working fluid

The most common working fluids used in ORC are R134a, R245fa, R22, isobutene, pentane, propane and PFCs. In the past, CFCs and HCFC were commonly used but are being phased out of current applications and avoided for new applications due to environmental and safety concerns. Any leaks that may occur in the system additionally pose a fire safety hazard if the working fluid selected is flammable. If a hydro-carbon were selected, heat exchangers in which the working fluid does not come into direct contact with the heat source become necessary and the flash point, the temperature at which a particular organic compound gives off sufficient vapor to ignite in the air, of the working fluid must be considered as a boiler maximum temperature.

The working fluid used for this study was R245fa, a popular choice for similar applications with medium to low grade waste heat. The phase transition diagram is shown below. A table from the same source was used in the Python model to produce the results shown in the following section.

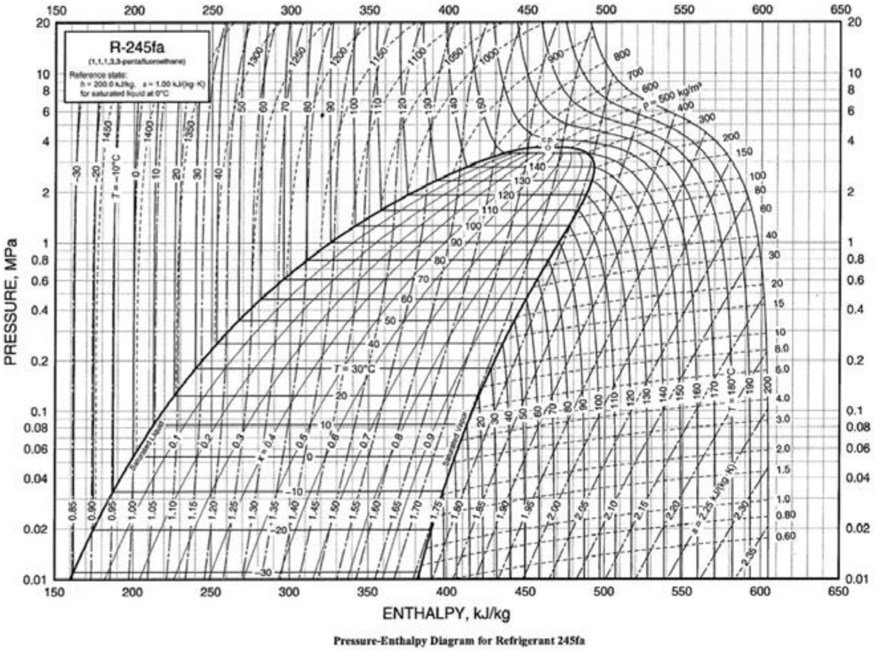


Figure 13: R245fa Pressure/Enthalpy diagram [18]

8.2. Boiler working pressure

First, the planned range for boiler working pressure was tested. Because the temperature of the heat source is limited, the working pressure will also be limited. The model that was developed was used to iteratively increase the working pressure of the boiler until the corresponding temperature was equal to the maximum source temperature. Those results follow.

Table 8: Max boiler working pressure and temperature

Boiler working pressure (MPa)	Boiler working temperature (^{o}C)
1.26	100

This working pressure was then used as an upper bound in the numerical analysis which used the model to determine the maximum power and maximum thermal efficiency of the cycle and report back the working pressures that achieved each. A mass flow rate in the ORC was set to 1kg/s to return results per unit mass flow rate. Those results are shown below.

Table 9: Working pressures in a cycle optimized for power and efficiency respectively

Maximum power output (kW/(kg/s))	33.97
Boiler pressure at max power (MPa)	1.26
Condenser pressure max power (MPa)	0.1225
Maximum thermal efficiency	13.6%

The working temperatures for these cases were 100^{o}C and 20^{o}C for the boiler and condenser respectively. These temperatures are very close to the temperature of the heat source (engine cooling system) and the heat sink (ambient air) Therefore, the heat transfer between this cycle and the source and sink will be very low as it is linearly dependent on the temperature difference. So, it was necessary to optimize the cycle for heat exchanger size as well.

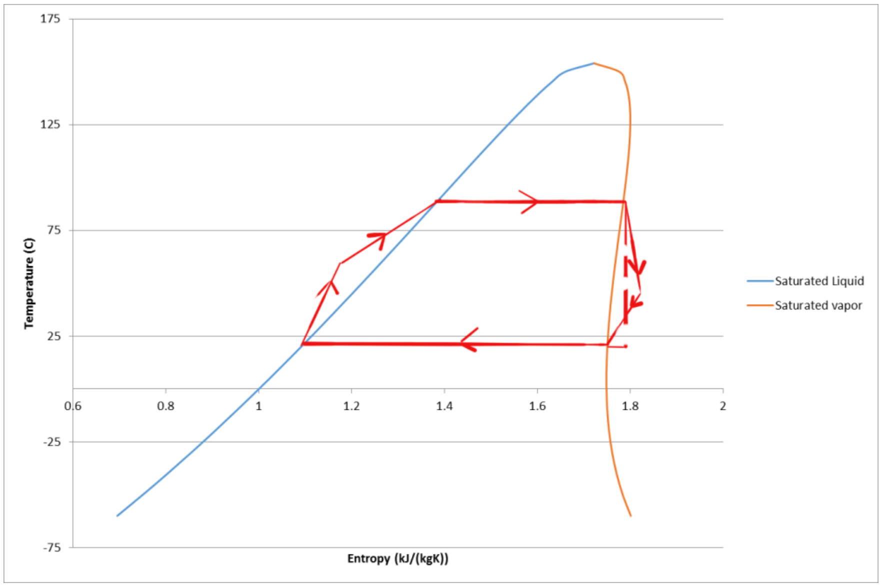


Figure 14: Maximum output cycle given heat source and sink temperatures

Finally, the output from the model developed can be used with some further assumptions in Equation 35. The experimental case did not perform the following extension, but it useful for the purpose of this paper, as it gives another design criterion by which working pressures might be chosen, heat exchanger surface area. The model is a counter-flow pipe-in-shell heat exchanger with a single pass.

The assumptions that were used in the test case were a mass flow rate of 1kg/s for both the working fluid and the flow rate of the vehicle cooling system working fluid. This was chosen such that the results could be easily adapted to a particular use case where the flow rates are known. For similar reasons, a heat exchanger tube length of 1m was chosen. The waste heat source working fluid was assumed to be a 50/50 water glycol mixture with a specific heat of 3.7682 kJ/(kgK).

U, the over-all heat transfer coefficient, was selected from Table 10-1 [15] for a water-to-oil heat exchanger. The table gives a range of 110-350 W/m^{2o}C. The most conservative value, 110 W/m^{2o}C was chosen.

Equation 31 was used to determine the exit temperature of the waste heat source fluid given a heat exchange demand based on the working temperature of the boiler. If this exit temperature is ever lower than the working temperature of the boiler, heat exchange between the two fluids will cease. Therefore, the model was used iteratively to determine the heat transfer rate at which the source temperature remains marginally higher than the working temperature of the boiler.

After iterating over boiler and condenser working pressures, the following cycle characteristics were determined to define a viable cycle given the heat available from a heat source of 100^{o}C where the mass flow rate of the heat source fluid and the mass flow rate of the ORC are equal:

Table 10: Cycle parameters given maximum heat source and equal mass flow rates

Maximum power output (kW/(kg/s))	11.56
Boiler pressure at max power (MPa)	0.27
Condenser pressure max power (MPa)	0.1225
Maximum thermal efficiency	5%
Boiler working temperature	43.56^{o}C

Using these parameters, the exit temperature of the source can be found as follows:

$$T_{h2} = T_{h1} - \frac{\dot{m}_{ref}}{\dot{m}_h} \frac{(h_1 - h_4)}{c_h}$$

Equation 38: Heat source fluid exit temperature

$$43.85 = 100 - 1\frac{(437.86 - 226.28)}{3.7682}$$

Note that the exit temperature of the heat source fluid is marginally higher than the working temperature of the boiler.

With that temperature, enough information is known to determine the log mean temperature difference across the boiler to determine the required heat exchange surface area using Equation 31.

$$10.65^{\text{o}}C = \frac{(43.85^{o}C - 43.56^{o}C) - (100^{o}C - 43.56^{o}C)}{\ln[\frac{43.85^{o}C - 43.56^{o}C}{100^{o}C - 43.56^{o}C}]}$$

Equation 30 can be used along with the assumption of 1kg/s mass flow rate and 110W/(m^{2o}C) overall heat transfer coefficient to determine the heat exchange area:

$$180.61m^2 = \frac{211.59\frac{kW}{\frac{kg}{s}} * 1.0\left(\frac{kg}{s}\right)}{0.110\left(\frac{kW}{m^{2o}C}\right) * 1.0 * 10.65^{\text{o}}C}$$

The final results of this study were:

- A max boiler pressure of 0.27MPa, limited by heat availability from the waste heat source
- A max power output per unit mass flow rate of 11.56kW/(kg/s)
- A heat requirement of 211.59 kW/(kg/s) to achieve max power output
- A max thermal efficiency of approximately 5%
- A boiler heat exchange area of 180.61m^2 per unit mass flow rate.

This working temperature resulted in the following cycle:

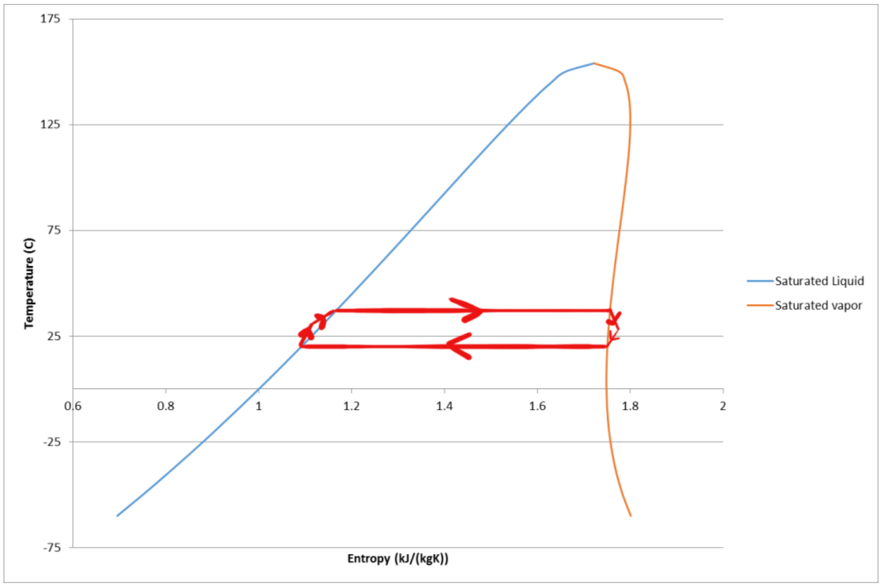


Figure 15: Maximum output cycle considering heat source limitations

8.3. Condenser working pressure

Next, the heat rejection situation was examined for the condenser. For any heat to be rejected, the condenser temperature must be higher than the ambient air temperature. For the cycle to be viable, the condenser temperature must be lower than the boiler temperature. Therefore, the maximum ambient air temperature for a viable ORC where the mass flow rates of the boiler and the heat source are equal is 43.56^{o}C. This temperature can easily be reached simply by parking a vehicle in direct sunlight during the summer. This makes the operating conditions for an ORC dependent on a low range of weather-related operating conditions and could mean that the design space for this cycle disappears entirely in warm weather.

8.4. Heat exchanger size

Finally, the size of a boiler heat exchanger can be estimated using Equation 37. Considering a heat exchange area of 181m^{2} a heat exchanger length of 1m and a tube diameter of 12.7mm, which is considered practically small for a tube diameter, the number of tubes required for the boiler application in this study was 4537 tubes. With a diameter of 12.7mm this would require a very large diameter shell. Lengthening the heat exchanger can reduce the number of tubes required, but as can be seen below, the heat exchanger length then becomes prohibitive, no longer fitting in the engine compartment.

Table 11: Number of heat exchanger tubes required for a given heat exchanger length

Tube length (m)	Number of tubes
1	4537
2	2269
3	1513
4	1135
5	908

9. Discussion and conclusion

In this study, a numerical model of a Rankine cycle was built that pulled state data from a data base, took from the user boiler and condenser working pressures, and output power per unit mass flow rate, thermal efficiency, heat input and heat output per unit mass flow rate, and those data were used to estimate heat exchanger sizes and temperature requirements.

The results from this study show that the expected limitations on an automotive waste heat recovery system are indeed the low quantity and quality of waste heat, and the size requirements of the heat exchanger, and the high ambient air temperature in the engine compartment. Further work could be done to increase the overall heat exchange coefficient, isentropic turbine efficiency and temperature of the waste heat and make some improvements on these results. Similarly, heat from the condenser could be rejected outside the engine compartment, or to some other cooler area isolated from the heat from the engine or summer conditions. Pursuing several of these improvements is recommended for further study before building a prototype to ensure that adequate energy can be harvested to produce a viable cycle and that operating conditions can be guaranteed to be favorable for heat rejection.

Furthermore, without significant technical innovation, the necessary size of the heat exchangers for a cycle of this type makes them unsuitable for an automotive waste heat recovery application.

Finally, it was requested that I look into driving the pump mechanically using the turbine. However, there are a few practical problems with this system architecture. This setup would tie the speed of the pump to the speed of the turbine and would make the cycle difficult to start without a variable transmission or clutch of some kind making it a project sufficiently complicated that it deserves its own exploration outside of the scope of this project.

10. Acknowledgements

There are academic and personal objectives for this study as well. The academic objectives for this course of study are to expand my knowledge of vapor power cycles to the end of developing a useful automotive application. Further to complete my degree at GVSU in order to open up avenues for further study on this and related subjects. It is also an objective of this work to contribute to the body of knowledge in this field for the benefit of the engineering profession at large and for the environmental and commercial benefits of improving energy utilization.

The personal objectives for this project include satisfying a personal interest I have in the field of alternative energy, and novel technical projects.

I would like to take the next few lines to thank the people who helped me to achieve these objectives.

I’d like to thank my committee members for their guidance and generosity with their time. Dr. Mokhtar, Dr. Sozen and Larry ridge.

I’d like to thank Lee Karner, Steve Degryse, Dan Kushner, and GHSP for allowing me the flexibility to take classes after work, and often give me flexibility in working hours to meet school requirements and for the financial support to pursue this degree program.

I’d like to thank my wife Karrie and my daughter Dexter who was born during this project for their understanding, patience and support throughout this program.

11. References

[1] E. P. Dig Vijay Singh, "Energy Conversion and Management," *Elsevier,* no. 111, pp. 315-328, 2016.

[2] E. P. Dig Vijay Singh, "A review of waste heat recovery technologies for maritime applications," *Energy Conversion and Management,* no. 111, pp. 315-328, 2016.

[3] BCS, Incorporated, "Waste Heat Recovery: Technology and Opportunities in U.S. Industry," *U.S. Department of Energy Industrial Technologies Program,* p. 112, 2008.

[4] S. H. Kang, "Design and Preliminary Tests of ORC (Organic Rankine Cycle) with Two-Stage Radial Turbine," *Energy,* no. 96, pp. 142-154, 2016.

[5] B. L. W. D. Daniel Walraven, "Comparison of Thermodynamic cycles for Power Production from Low-temperature Geothermal Heat Sources," *Energy Conversion and Management,* no. 66, p. 14, 2012.

[6] C. M. I. C. P. Paola Bombarda, "Heat Recovery from Diesel Engines: A Thermodynamic Comparison Between Kalina and ORC cycles," *Applied Thermal Engineering,* no. 30, pp. 212-219, 2010.

[7] R. DiPippo, "Second Law Assessment of Binary Plants Generating Power from Low-Temperature Geothermal Fluids," *Geothermics,* no. 33, pp. 565-586, 2003.

[8] Z. Y. Enhua Wang, "A numerical analysis of a composition-adjustable Kalina cycle power plant for power generation from low-temperature geothermal sources," *Applied Energy,* vol.

180, pp. 834-848, 2016.

[9] T. S. A. F. H. A. I. M. I. Mohammed A. Khatita, "Power generation using waste heat recovery by organic Rankine cycle in oil and gas sector in Egypt: A case study," *Energy,* vol. 64, pp. 462-472, 2013.

[10] T. C. Hung, T. Shai and S. K. Wang, "A Review of Organic Rankine Cycles (ORCs) for the Recovery of Low-Grade Waste Heat," *Energy,* vol. 22, no. 7, pp. 661-667, 1997.

[11] J. C. E. P. O. J. V. E. E. S. L. V. M. C. D. M. d. S. F. R. L. D. V. G. Carlos Eymel Campos Rodriguez, "Exergetic and economic comparison of ORC and KC for low temperature enhanced geothermal system in Brazil," *Applied Thermal Engineering,* no. 52, pp. 109-119, 2013.

[12] H. N. S. Michael J. Moran, Fundamentals of Engineering Thermodynamics, John Wiley & Sons, 2008.

[13] I. H. a. W. J. a. Q. S. a. L. V. Bell, "Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and," 2014. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/ie4033999. [Accessed 10 August 2019].

[14] I. B. M. H. M. M. E.W. Lemmon, "NIST Standard Reference Database 23: Reference Fluid Thermodynamic and Transport Properties-REFPROP, Version 10.0, National Institute of Standards and Technology," 2018. [Online]. Available: https://www.nist.gov/srd/refprop. [Accessed 11 August 2019].

[15] J. P. Holman, "Heat Transfer," in *Heat Transfer,* 10th ed., New York, McGraw-Hill, 2010, p. 713.

[16] Y. G. S. T. L. Li, "Experimental study on a small-scale R245fa organic Rankine cycle

system for low-grade thermal energy recovery," *Energy Procedia*, no. 105, pp. 1827-1832, 2017.

[17] S. H. Kang, "Design and Experimental Study of ORC (Organic Rankine Cycle) and a Radial Turbine Using R245fa Working Fluid," *Energy,* no. 41, pp. 514-524, 2011.

[18] American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc., "Chapter 30 - Thermophysical Properties of Refrigerants," in *2009 ASHRAE Handbook,* Atlanta, https://app.knovel.com/hotlink/toc/id:kpASHRAE37/ashrae-handbook-fundamentals/ashrae-handbook-fundamentals, 2009, p. 75.

[19] C. B. Arvind Thekdi, "Waste Heat Reduction and Recovery Options for Metals Industry," *Energy Technology. John Wiley & Sons, Inc.,* pp. 17-24, 2011.

[20] BCS, Incorporated, "Waste Heat Recovery: - Technology and Opportunities in U.S. Industry -," 2008.

[21] 15 December 2017. [Online]. Available: https://en.wikipedia.org/wiki/Carnot_cycle.

[22] 28 December 2017. [Online]. Available: https://hellafunctional.com/?p=629.

12. Appendix A – Acronyms

C – Celsius

CHP – Combined cooling heating and power

F – Fahrenheit

LNG – Liquefied natural gas

ORC – Organic Rankine cycle

RC – Rankine cycle

SCRC – Super-critical Rankine cycle

SRC – Steam Rankine cycle

WHR – Waste heat recovery

WHRS – Waste heat recovery system

13. Glossary

Adiabatic: Without heat transfer

Isobaric: Constant pressure

Isochoric: Constant volume

Isothermal: Constant temperature

Enthalpy: The sum of the internal energy of a system and the product of its pressure and volume.

$H = U +pV$

Entropy: A quantity representing the unavailability of a system's thermal energy for conversion into mechanical work, often interpreted as the degree of disorder or randomness in the system.

14. Appendix C – Source Code

In addition to the custom libraries written for this project, several open source libraries were also used. The source code for those libraries is not included.

14.1. Shell.py

```
import ORC_Model as orc
import heat_exchanger_model as heatex
import os
import numpy as np
import pandas as pd
import time
import sanitize_inputs as si

np.set_printoptions(precision=3)
# Sets the decimal precision when printing numpy arrays to 3. Note that further
# significant figures are preserved, just not printed.

def find_max_boiler(condenser_pressure, boiler_pressure, turbine_efficiency, pump_efficiency,
max_source_temp,c_h,m_ORC,m_h, db_path):

    test_temp = 0
    b_temp = max_source_temp-0.1
    while test_temp < b_temp:
        (Wm,eff,b_temp,c_temp,Qin_m,Qout_m) = orc.ORC_model(condenser_pressure,
                                                          boiler_pressure,
                                                          turbine_efficiency,
                                                          pump_efficiency,
                                                          db_path)

        test_temp = heatex.feedwater_exit_temp(max_source_temp,c_h,Qin_m,m_ORC,m_h)

        print("T: {:4.2f} deg, P: {:4.2f} MPa".format(test_temp, boiler_pressure))
        boiler_pressure -= 0.01
    print("Power: {:4.2f}kW/(kg/s)\nEfficiency: {:4.2f}" .format(Wm, eff))
    print("Heat in: {:4.2f}kW/(kg/s)\nHeat out: {:4.2f} kW/(kg/s)".format(Qin_m,Qout_m))
    print("Condenser temperature: {:4.2f} deg Celsius\nBoiler temperature: {:4.2f} deg
Celsius".format(c_temp,b_temp))
    print("Boiler pressure: {:4.2f} MPa".format(boiler_pressure))

R245fa_db = '\\R245fa Saturated properties temperature table.csv'

db_upper_pressure_limit = 3.651
db_lower_pressure_limit = 0.00127

os.chdir("..")# Navigate up a directory
db_path=os.path.abspath(os.curdir)+r"\GHSP study\Additional references"+R245fa_db
# Navigate to the directory containing the working fluid database.

p_boiler_max = si.get_real_number("Enter the boiler working pressure upper limit
(MPa):\n",upper=db_upper_pressure_limit,lower=db_lower_pressure_limit)
p_boiler_min = si.get_real_number("Enter the boiler working pressure lower limit
(MPa):\n",upper=p_boiler_max,lower=db_lower_pressure_limit)

p_condenser_max = si.get_real_number("Enter the condenser working pressure upper limit
(MPa):\n",upper=p_boiler_min,lower=db_lower_pressure_limit)
p_condenser_min = si.get_real_number("Enter the condenser working pressure lower limit
(MPa)(0.123 corresponds to a 20C working
temp):\n",upper=p_condenser_max,lower=db_lower_pressure_limit)
turbine_efficiency = si.get_real_number("Enter isentropic turbine efficiency (0-
1):\n",upper=1,lower=0)# Study value = 0.787
```

```
pump_efficiency = si.get_real_number("Enter isentropic pump efficiency (0-
1):\n",upper=1,lower=0)# Study value = 0.9

max_source_temp = si.get_real_number("Enter the maximum heat source temperature:\n")
source_specific_heat = si.get_real_number("Enter the specific heat of the heat source fluid
(kJ/kgK)",lower=0)
# 3.7682 kJ/kgK for 5050 water/glycol mixture
m_245 = si.get_real_number("Enter the mass flow rate of the ORC working fluid (kg/s)",lower=0)
m_source = si.get_real_number("Enter the mass flow rate of the source working fluid
(kg/s)",lower=0)

# Creates a numpy array with 25 data points between p_condenser_min and p_condenser_max.
##condenser_pressure_range = np.linspace(p_condenser_min, p_condenser_max, 25)
##boiler_pressure_range = np.linspace(p_boiler_min, p_boiler_max, 25)

##condenser_pressure = 0.13
##boiler_pressure = 1.26
##turbine_efficiency = 0.787
##pump_efficiency = 0.9
##max_source_temp = 100

find_max_boiler(p_condenser_min,
                p_boiler_max,
                turbine_efficiency,
                pump_efficiency,
                max_source_temp,
                source_specific_heat,
                m_245,
                m_source,
                db_path)

user = None
while(1):
    user = input()
    if user == 'q':
        break
    else:
        pass
```

14.2. ORC_Model.py

```
import csv
import math
import sanitize_inputs as si
import time
import os
import CoolProp.CoolProp as CP

def ORC_model(cond_pres, boil_pres, eff_t, eff_p,working_fluid_db):
    '''This function takes a working pressure for a condenser and boiler, an
    efficiency for a pump and turbine, and outputs the work per unit mass flow
    and thermal efficiency for a generic ORC, as well as the working temperature
    and enthalpy at each of the four fixed states which can then be used in the
    models for the individual major components.'''

    fluid = 'R245fa'

    h1 = CP.PropsSI('H', 'P', boil_pres*1000000, 'Q', 1, 'R245fa')/1000
    s1 = CP.PropsSI('S','P',boil_pres*1000000, 'Q', 1, fluid)/1000


    h3 = CP.PropsSI('H', 'P', cond_pres*1000000, 'Q', 0, fluid)/1000
    s3 = CP.PropsSI('S','P',cond_pres*1000000, 'Q', 0, fluid)/1000

    s2 = s1
    s2f = CP.PropsSI('S','P', cond_pres*1000000,'Q',0,fluid)/1000
    s2g = CP.PropsSI('S','P', cond_pres*1000000,'Q',1,fluid)/1000


    try:
        quality = (s2 - s2f)/(s2g - s2f)
    except ZeroDivisionError:
        quality = 0
    if quality > 1:
        print("Turbine outlet is superheated vapor! x = {:4.4f}".format(quality))
        h2s = CP.PropsSI('H','S',s2*1000,'P', cond_pres*1000000, fluid)/1000

    elif 0 <= x <= 1:
        h2f = h3

        h2g = CP.PropsSI('H','P', cond_pres*1000000, 'Q',1,fluid)/1000


        h2fg = h2g-h2f
        h2s = h2f + quality* h2fg

    h2 = h1 - eff_t*(h1 - h2s)

    specific_vol_3 = 1/CP.PropsSI('D','P',boil_pres*1000000,'Q',1,fluid)

    h4 = h3 + (specific_vol_3*(boil_pres-cond_pres))/eff_p

    print("h1 = {:4.2f}, h2 = {:4.2f}, h3 = {:4.2f}, h4 = {:4.2f}".format(h1,h2,h3,h4))
    W_m = h1-h2-h4+h3 # kilowatts of power per kg/s of mass flow rate
    Qin_m = h1-h4 # kilowatts of heat trasfer in per kg/s of mass flow rate
    Qout_m = h2-h3 # kilowatts of heat transfer out per kg/s of mass flow rate
    efficiency = ((h1-h2) - (h4-h3))/(h1-h4)

    cond_temp = CP.PropsSI('T','P',cond_pres*1000000,'Q',1,fluid)-273
    boil_temp = CP.PropsSI('T','P',boil_pres*1000000,'Q',1,fluid)-273

    return(W_m,efficiency,boil_temp,cond_temp,Qin_m,Qout_m)

#------Main------#
if __name__ == '__main__':
    R245fa_db = '\\R245fa Saturated properties temperature table.csv'
    os.chdir("..")# Navigate up a directory
    db_path=os.path.abspath(os.curdir)+r"\GHSP study\Additional references"+R245fa_db
    # Navigate to the directory containing the working fluid database.
    '''Manual entry begins here.'''
    condenser_pressure = si.get_real_number("Enter condenser pressure (MPa).\n>>>",lower = 0)
    boiler_pressure = si.get_real_number("Enter boiler pressure (MPa).\n>>>", lower =
condenser_pressure)
    turbine_efficiency = si.get_real_number("Enter the turbine efficiency (0-1).\n>>>",
                                            upper=1.0,lower=0)
    pump_efficiency = si.get_real_number("Enter the pump efficiency (0-1).\n>>>",
                                         upper=1.0, lower=0)
    max_heat = si.get_real_number("Enter maximum heat source temperature (C).\n>>>", lower = -
273)
    m = si.get_real_number("Enter the mass flow rate (kg/s).\n>>>",lower = 0)

    (Wm,efficiency,boil_temp,cond_temp,Qin_m,Qout_m) =
ORC_model(condenser_pressure,boiler_pressure,turbine_efficiency,pump_efficiency,db_path)
    print("Power: {}\nEfficiency: {}\nCondenser temperature: {}\nBoiler temperature: {}"\
          .format(Wm*m,efficiency,cond_temp,boil_temp))


    time.sleep(30)
else:
    pass
```

14.3. Sanitize_inputs.py

```
'''This package allows the user to request input from the user and handles
most error checking and input rules.'''

__version__ = "0.3.0"

import numpy as np
import readchar
from colorama import init

init()

# select function built by Kamik423 in cutie library
def select(
        options,
        deselected_prefix: str = '\033[1m[ ]\033[0m ',
        selected_prefix: str = '\033[1m[\033[32;1mx\033[0;1m]\033[0m ',
        selected_index: int = 0) -> int:
    """Select an option from a list.
    Args:
        options (List[str]): The options to select from.
        deselected_prefix (str, optional): Prefix for deselected option ([ ]).
        selected_prefix (str, optional): Prefix for selected option ([x]).
        selected_index (int, optional): The index to be selected at first.
    Returns:
        int: The index that has been selected.
    """
    print('\n' * (len(options) - 1))
    while 1:
        print(f'\033[{len(options) + 1}A')
        for i, option in enumerate(options):
            print('\033[K{}{}'.format(
                selected_prefix if i == selected_index else deselected_prefix,
                option))
        keypress = readchar.readkey()
        if keypress == readchar.key.UP:
            if selected_index == 0:
                selected_index = len(options) - 1
            else:
                selected_index -= 1
        elif keypress == readchar.key.DOWN:
            if selected_index == len(options) - 1:
                selected_index = 0
            else:
                selected_index += 1
        else:
            break
    return selected_index

class col_vec():
    '''Retrieves a list of real number for x, y, and z from the user,
    and constructs a numpy column vector.'''
    def __init__(self, coords):
        self.x = coords[0]
        self.y = coords[1]
        self.z = coords[2]
        self.vec = np.array([[self.x],[self.y],[self.z]])

def get_real_number(prompt=None, upper=float('Inf'), lower=float('-Inf')):
    '''Gets a real number from the user with an optional prompt. Positive and
    negative limits can be set. If not set, the default values are 'Inf' and
    '-Inf' respectively.'''

    num_flag = False
    while(not num_flag):
        try:
            number = float(input(prompt))
            if lower < number < upper:
                num_flag = True
            else:
                print("value must be between {} and {} exclusive.".format(lower, upper))
                print("\033[2A\033[K\033[1A\033[K\r", end='')

        except ValueError:
            print("\033[1A\033[K\033[1A\033[K\r", end='')
            num_flag = False
    print("\033[K", end='')
    return(number)

def get_integer(prompt=None, upper=float('Inf'), lower=float('-Inf')):
    '''Gets an integer from the user with an optional prompt. Positive and
    negative limits can be set. If not set, the default values are 'Inf' and
    '-Inf' respectively.'''
    num_flag = False
    while(not num_flag):
        try:
            number = int(input(prompt))
            number += 0
            # This will throw an exception if number is not an integer.

            if lower < number < upper: # excludes endpoints
                num_flag = True
            else:
                print("value must be between {} and {} exclusive.".format(lower, upper))
                print("\033[2A\033[K\033[1A\033[K\r", end='')


        except ValueError:
            print("\033[1A\033[K\033[1A\033[K\r", end='')
            # \033[K = Erase to the end of line
            # \033[1A = moves the cursor up 1 line.
            # \r = return
            num_flag = False
    print("\033[K", end='')
    return(number)

def get_letter(prompt=None, accept=None):
    '''Gets a single alpha character that is included in the list 'accept'
    Optionally include a prompt to the user
    omitting the accept list allows all alpha characters.'''

    flag = False
    while(not flag):
        letter = str(input(prompt))
        if(letter.isalpha() and len(letter) == 1):
            if accept != None:
                for i in accept:
                    if letter == i or accept==None:
                        flag = True
                        break
                    else:
                        pass
            else:
                flag = True

        else:
            pass

    return(letter)

def get_coords(rows=3):
    '''This function gets the coordinates for a point in 3D space from the user.
    It includes the error checking logic required to ensure the point's
    useability in subsequent functions.'''

    P_x = get_real_number("X >>> ")
    P_y = get_real_number("Y >>> ")
    P_z = get_real_number("Z >>> ")

    point = col_vec([P_x,P_y,P_z])
```

```
if rows == 3:
    return(point)
elif rows == 4:
    point.vec = np.row_stack([point.vec,[1]])
    return(point)
else:
    print("Invalid argument.")
    return(None)
```

14.4. Misc_functions.py

```
'''It is recommended to use this package with the sanitize_inputs package.\n
The functions contained herein do not check for erroneous inputs.'''

__version__ = "0.2.6"

import math
import csv
import pandas as pd
import pdb
import os
from difflib import SequenceMatcher
import datetime as dt
import time

def dxdy(df, y_col, x_col):
    '''Given a dataframe and labels of an x and y column, This function
    returns the data frame with an additional column dx/dy that is the first
    derivative of x with respect to y.'''

    df_shifted = df.shift(1)
    df["dx/dy"] = (df[y_col]-df_shifted[y_col])/(df[x_col]-df_shifted[x_col])
    return(df)

def activate(x, k, epsilon=1):
    '''This function takes a value x, a trigger value k, and a smoothing factor
    epsilon and returns a number between 0 and 1 with an epsilon smoothed
    transition at x=k.'''

    y=math.tanh((x-k)/epsilon)/2+0.5
    return(y)

def deactivate(x, k, epsilon=1):
    '''This function takes a value x, a trigger value k, and a smoothing factor
    epsilon and returns a number between 0 and 1 with an epsilon smoothed
    transition at x=k.'''

    y=math.tanh((x-k)/epsilon)/-2+0.5
    return(y)

def timestamp():
    now = dt.datetime.fromtimestamp(time.time())
    sep=":"
    stamp = sep.join([str(now.year),
                      str(now.month),
                      str(now.day),
                      str(now.hour),
                      str(now.minute),
                      str(now.second)])
    return("["+stamp+"]")

def similar(a,b):
    return(SequenceMatcher(None,a,b).ratio())

def similar_dir(directory, desired, threshold=0.75):
    '''Takes the name of a desired subdirectory, and returns a directory that
    best matches the desired one with certainty above given threshold.'''
    candidates = []
    for f in os.listdir(directory):
        candidates.append((f, similar(f,desired)))

    candidates.sort(key=lambda x: x[1],reverse=True)
    best_match = candidates[0]
    if best_match[1] >= threshold:
        return(best_match[0])
    else:
        return(None)

def interpolate(x1,y1,x2,y2,x):
    '''This function returns a value, y, linearly interpolated using two x,y
    pairs of data and a given x between those pairs.'''

    try:
        y = ((y2-y1)/(x2-x1))*(x-x1) + y1
    except TypeError:
        y = y1

    return(y)

def interpolate_y(x1,y1,x2,y2,y):
    '''This function returns a value x, linearly interpolated using two x,y
    pairs of data and a given y between those pairs.'''

    try:
        m = (y2-y1)/(x2-x1)
        b = y1
        x = (y-b)/m
    except TypeError:
        x = x1

    return(x)

def tab_dict(rfile):
    '''This is a function that opens an excel file and returns a dictionary
    where the keys of the dictionary are the sheet names and the values are
    dataframes containing the data from the sheet. rfile must include the path
    if the file is not in the current working directory.'''

    try:
        xlsx = pd.ExcelFile(rfile)
        Sheet_frames = {sh:xlsx.parse(sh) for sh in xlsx.sheet_names}
        # This line creates a dictionary where the keys are the tab names,
        # and the values are the data from that tab.
        return(Sheet_frames)

    except FileNotFoundError:
        print(rfile,"Does not exist.")
        return(None)

def list_headers(rfile, r_c='r'):
    '''rfile is the csv file in which the data are stored. pass 'r' or 'c' for
    the second argument to indicate whether the headers are in the first row or
    the first column.'''

    headers = []
    RDR = csv.reader(open(rfile))
    if r_c.lower() == 'c':
        for row in RDR:
            print(row[0])
            headers.append(row[0])

    elif r_c.lower() == 'r':
        headers = next(RDR)

    return(headers)

def vlookup(rfile, index, search_col, result_col,skip_headers=False):
    '''rfile is the name of file in which data are stored. index is the value
    to search database rows for. search_col is the column in which the
    index can be found. result_col should be the column from which the result
    should be extracted. This function is made to work smoothly with
    interpolate() Skip headers allows the user to skip searching the first row
    which will not happen automatically if the column labels are numbers.'''

    index = float(index)
    search_col = int(search_col)
    result_col = int(result_col)

    RDR = csv.reader(open(rfile,'r'), dialect = 'excel')
    pos_diff = math.inf
    neg_diff = math.inf*-1

    x1 = None
    y1 = None
    x2 = None
    y2 = None

    for i, row in enumerate(RDR):
        # Search for the rows just smaller and just larger than the search
        # term. Calculate the difference between the x value in a given row
        # and the search term. Keep the rows that result in the smallest
        # positive difference and the smallest negative difference.
        if i == 0 and skip_headers:
            #next(RDR)
            print("Advanced a row")
            continue
        try:
            diff = index - float(row[search_col])

        except ValueError:
            if row[search_col] == "Inf":
                diff = math.inf
            continue

        if diff < pos_diff and diff > 0:
            x1 = float(row[search_col])
            y1 = float(row[result_col])
            pos_diff = diff

        elif diff > neg_diff and diff < 0:
            x2 = float(row[search_col])
            y2 = float(row[result_col])
            neg_diff = diff

        elif diff == 0:
            x1 = float(row[search_col])
            y1 = float(row[result_col])
            x2 = None
            y2 = None

    return (x1, y1, x2, y2)
    # Return the x,y pairs of the search column and result column just
    # above and below the desired x value.

def bernoulli_trial(n, k, p):
    '''Returns the probability between 0 and 1 of exactly k successes given
    n trials where the probability of success is p. k and n must be integers
    and p is a float between 0 and 1.'''

    q = 1-p
    binomial_coeff = math.factorial(n)/(math.factorial(k)*math.factorial(n-k))
    P = binomial_coeff*(p**k)*(q**(n-k))
    return(P)

def bernoulli_trial_n(k, p, P=0.95):
    '''Returns the sample size required to observe at least k successes if
    the probability of success in each trial is p to a confidence level of P.'''

    n = k # Sample size must be at least k in order to observe k successes.
    test_P = 0
    while test_P < P:
        test_P = 0
        for test_k in range(k, n-1):
            test_P += bernoulli_trial(n, test_k, p)
        n+= 1
        print("One can be {0:4.2f} confident that at least {1} successes will occur if {2} parts
are tested.".format(test_P, k, n))
    return(n)

def favstats(rfile, column):
    '''This function calculates common statistical values for a given column
    of data found in the specified file.'''
    df = pd.read_csv(rfile)
    xbar = df[column].mean()
    sd = df[column].std()
    minimum = df[column].min()
    first = df[column].quantile(0.25) # first quartile
    median = df[column].median()
    third = df[column].quantile(0.75) # third quartile
    maximum = df[column].max()
    IQR = third - first

    print("Minimum: ", minimum,
          "\nFirst quartile: ", first,
          "\nMedian: ", median,
          "\nMean: ", xbar,
          "\nThird quartile: ",third,
          "\nMaximum: ", maximum,
          "\nStandard deviation: ",sd,
          "\nInter-quartile range: ",IQR,sep='')

def t_test_file(rfile, col, xbar=0, alpha=0.05, twotail=True, lower=True):
    '''One sample t-test. Arguments are the csv file in which the data are
    located and the column in which the data are found along with an alpha
    value. var is the column name in which the category of interest is stored.
    col is the column in which the response variable is stored. xbar is the
    variable to which the mean will be compared. twotail tells the function
    whether it should do a two tail test as opposed to a one tail test. lower
    is ignored for two tail, but determines which tail is considered in the one
    tail variant.'''

    df = pd.read_csv(rfile)

    # This line pulls data out of the data frame creating two new data frames
    # one for each label.

    # The resulting data structure is a tuple where element 0 is the group name
    # and element 1 is the actual sub-dataframe.
    xbar_test = df[col].mean()
    sd = df[col].std()
    n = len(df[col])
    DOF = n-1

    # Look up the appropriate t statistic - a 2 parameter interpolation function
    # would be nice here for an arbitrary value of alpha.
    if twotail:
        lookupfile = "twotail tstat.csv"

    else:
        lookupfile = "onetail tstat.csv"
    headers = list_headers(lookupfile,'r')
    for i, h in enumerate(headers):
        try:
            if float(h) == float(alpha):
                print("Alpha level is: ",float(alpha))
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfile, DOF, 0, i,skip_headers=False)
    tsalpha = interpolate(x1,y1,x2,y2,DOF)

    std_err = sd/n**0.5

    # calculate the confidence interval
    diff = (xbar_test - xbar)
    upper = (diff) + tsalpha*std_err
    lower = (diff) - tsalpha*std_err
    print((1-float(alpha))*100,"% Confidence interval: ",lower," - ",upper,sep='')

    # calculate p-value
    ts = abs(diff/std_err)
    if twotail:
        #find p for given ts in twotail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")

    else:
        #find p for given ts in onetail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")
        if lower:
            pass
        else:
            pass
    headersT = list_headers(lookupfileT,'r')
    for i, h in enumerate(headersT):
        try:
            if float(h) == float(DOF):
                break
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfileT, ts, i, 0,skip_headers=True)
    print("({0},{1}) - ({2},{3})".format(x1,y1,x2,y2))
    print("avg: {0}\nsd: {1}\nn: {2}\ndiff: {3}\nstd_err: {4}"\
          .format(xbar_test,sd,n,diff,std_err))
    print("ts = {0}".format(ts))
    p = interpolate(x1,y1,x2,y2,ts)
    print("p = ",p)
    # formulate conclusion

def t_test2_file(rfile, var, c1, c2, treat, alpha=0.05, twotail=True, lower=True):
    '''Two sample t-test. Arguments are the csv file in which the data are
    located and the two columns to be compared along with an alpha value.
    var is the column name in which the categories are stored, c1 and c2 are
    the two labels in that column to be compared. treat is the treatment
    varible. ie the variable that will be used to compare the groups. twotail
    tells the function whether it should do a two tail test as opposed to a one
    tail test. lower is ignored for two tail, but determines which tail is
    considered in the one tail variant.'''

    df = pd.read_csv(rfile)
    groups = dict((x,y) for x,y in df.groupby(var))

    # This line pulls data out of the data frame creating two new data frames
    # one for each label.

    # The resulting data structure is a tuple where element 0 is the group name
    # and element 1 is the actual sub-dataframe.
    xbar1 = groups[c1][treat].mean()
    xbar2 = groups[c2][treat].mean()

    # Pandas standard deviation function uses Bessel's correction by default.
    s1 = groups[c1][treat].std()
    s2 = groups[c2][treat].std()

    n1 = len(groups[c1][treat])
    n2 = len(groups[c2][treat])
    n = min(n1,n2)
    DOF = n-1
    # n will be used to calculate the standard error. Choosing the smaller of
    # the two sample sizes yields a conservative estimate.

    # Calculate the pooled standard deviation
    sp = (((n1-1)*s1**2+(n2-1)*s2**2)/(n1+n2-2))**0.5

    # Look up the appropriate t statistic - a 2 parameter interpolation function
    # would be nice here for an arbitrary value of alpha.
    if twotail:
        lookupfile = "twotail tstat.csv"

    else:
        lookupfile = "onetail tstat.csv"
    headers = list_headers(lookupfile,'r')
    for i, h in enumerate(headers):
        try:
            if float(h) == float(alpha):
                break
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfile, DOF, 0, i,skip_headers=True)
    print("DOF: {}\ni: {}".format(DOF,i))
    tsalpha = interpolate(x1,y1,x2,y2,DOF)
    # There is an issue here where if a sample size n=2, DOF=1 and the lookup
    # table returns a nonetype

    std_err = sp/n**0.5

    # calculate the confidence interval
    diff = (xbar1 - xbar2)
    upper = (diff) + tsalpha*std_err
    lower = (diff) - tsalpha*std_err

    # calculate p-value
    ts = abs(diff/std_err)
    if twotail:
        #find p for given ts in twotail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")

    else:
        #find p for given ts in onetail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")
        if lower:
            pass
        else:
            pass
    headersT = list_headers(lookupfileT,'r')
    for i, h in enumerate(headersT):
        try:
            if float(h) == float(DOF):
                break
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfileT, ts, i, 0,skip_headers=True)
    print("x1,y1,x2,y2",x1,y1,x2,y2)
    p = interpolate(x1,y1,x2,y2,ts)
    # formulate conclusion

    print("xbar1 = ",xbar1,
          "\nxbar2 = ",xbar2,
          "\ns1 = ",s1,
          "\ns2 = ",s2,
          "\nn1 = ",n1,
          "\nn2 = ",n2,
          "\nn = ",n,
          "\nsp = ",sp,
          "\ntsalpha = ",tsalpha,
          "\nts = ",ts,
          "\ndiff = ",diff,
          "\nupper = ",upper,
          "\nlower = ",lower,
          "\nstd_err = ",std_err,
          "\np = ",p,sep='')

    return(df)

def paired_t_test(DF1, DF2, alpha=0.05, mu=0, twotail=True, lower=True):
    '''This function takes two pandas dataframes which must have the same index
    and find the change in each item and perform a t-test to compare the
    change to a given number.
    Index entries which don't appear in both dataframes will be removed
    automatically.'''

    diffDF = pd.DataFrame()
    for col in DF1:
        try:
            diffDF[col] = DF1[col].sub(DF2[col])
        except:
            print("Paired data can only be calculated for numeric data.")
            continue

    diffDF = diffDF.dropna()
    print(diffDF)
    p = t_test(diffDF, alpha, mu, twotail, lower)
    return(p)

def t_test(series, alpha=0.05, mu=0, twotail=True, lower=True):
    '''This function performs a 1 sample t-test on a pandas data series rather
    than a file.'''

    xbar_test = series.mean()
    sd = series.std()
    n = len(series)
    DOF = n-1

    # Look up the appropriate t statistic - a 2 parameter interpolation function
    # would be nice here for an arbitrary value of alpha.
    if twotail:
        lookupfile = "twotail tstat.csv"

    else:
        lookupfile = "onetail tstat.csv"
    headers = list_headers(lookupfile,'r')
    for i, h in enumerate(headers):
        try:
            if float(h) == float(alpha):
                print("Alpha level is: ",float(alpha))
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfile, DOF, 0, i,skip_headers=False)
    tsalpha = interpolate(x1,y1,x2,y2,DOF)

    std_err = sd/n**0.5

    # calculate the confidence interval
    diff = (xbar_test - mu)
    upper = (diff) + tsalpha*std_err
    lower = (diff) - tsalpha*std_err
    print((1-float(alpha))*100,"% Confidence interval: ",lower," - ",upper,sep='')

    # calculate p-value
    ts = abs(diff/std_err)
    if twotail:
        #find p for given ts in twotail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")

    else:
        #find p for given ts in onetail tstat.csv
        lookupfileT = ("twotail tstat Transpose.csv")
        if lower:
            pass
        else:
            pass
    headersT = list_headers(lookupfileT,'r')
    for i, h in enumerate(headersT):
        try:
            if float(h) == float(DOF):
                break
            else:
                pass
        except ValueError:
            continue
    x1,y1,x2,y2 = vlookup(lookupfileT, ts, i, 0,skip_headers=True)
    print("({0},{1}) - ({2},{3})".format(x1,y1,x2,y2))
    print("avg: {0}\nsd: {1}\nn: {2}\ndiff: {3}\nstd_err: {4}"\
          .format(xbar_test,sd,n,diff,std_err))
    print("ts = {0}".format(ts))
    p = interpolate(x1,y1,x2,y2,ts)
    print("p = ",p)

    return(p)

def r_ch_arc(Arc, Chord, dr):
    '''Find the radius of a circle given a chord length and an arc length. This
    is a numerical solution. The argument dr is the desired level of
    precision.'''

    a = float(Arc)
    c = float(Chord)
    dr = float(dr)

    radius = 0
    error = 100
    # numberical solution for radius. Iterates until error is less than
    # specified dr.
    while (error > dr):
        radius += dr

        tempA = math.sin(a/(2*radius))
        tempB = c/(2*radius)
        error = abs(tempA-tempB)

    return(radius)
```

14.5. Heat_exchanger_model.py

```
import csv
import math
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
import numpy as np
import misc_functions as mf

def LMTD(Th1, Th2, Tc1, Tc2):
    '''Returns the log mean temperature difference given the entry and exit
    temperatures of the hot fluid, Th1 and Th2, and the entry and exit
    temperatures of the cool fluid, Tc1 and Tc2.'''

    log_mean_temp_diff = ((Th2-Tc2)-(Th1-Tc1))/math.log(((Th2-Tc2)/(Th1-Tc1)))
    return(log_mean_temp_diff)

def feedwater_exit_temp(Th1,c_h,Q_m,m_ORC,m_h):
    '''Returns the feedwater exit temp given the entry temp, the specific heat
    of the source fluid, the heat transfer per unit mass flow rate required,
    and the mass flow rates for both fluids.'''

    Th2 = Th1 - (m_ORC*Q_m)/(m_h*c_h)
    return(Th2)

def tube_in_shell(dT, Q, U = 850):
    '''Returns the required heat exchange area given the heat transfer required,
    The temperature difference between the two working fluids, and the overall
    heat transfer coefficient.'''

    A = q_in*1000/(U*dTm) # square meters
    print("Boiler heat exchange area: {:4.2f}m^2".format(A))

    # Calculate the number of tubes in shell based on tube diameter and tube and
    # and shell exchanger length.

    L = 0.25 # meters
    d = 0.003 # diameter of tubes in meters
    circ = A/L # circumference of all tubes
    n = math.ceil(circ/(math.pi*d))

    print("Number of tubes: ", n)

if __name__ == "__main__":
    pass
```