# Abstract

Keywords: Rankine cycle, Working pressure, Working temperature, Boiler, Condenser, Power-vapor cycle.

# Introduction

# Feasibility

According to Singh and Pedersen’s work, 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 [1]

“Engine cooling water temperatures of 80-90 oC are fairly standard for most engines.” [1] While exhaust gas appears to be the most lucrative source of waste heat energy, there are several complications which, though not as important for the Pederson and Singh study, would prevent it’s 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.

For waste heat streams with a temperature of 95 oC – 260 oC it was recommended that an ORC be used for power generation. It was also noted that the least efficient form of energy recovery from waste heat at this time is electrical generation. [2]

The Kalina Cycle (KC) is another variation on the RC that utilizes low temperature heat sources. The KC uses a variable composition working fluid mixture, typically water and ammonia to track with the heat sink temperature available. 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%.” [3] “Some studies showed that a KC can achieve a better thermal efficiency than ORC systems.” [3] “In practice, the expansion ratio of the turbine for KCS-34 is relatively high and a multi-stage turbine is required.” [3] This may be the drawback of the KC that makes ORC somewhat more attractive for an automotive application due to the extreme space requirements though it might be marginally less efficient. The KC also requires a significant 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 significantly more complicated to implement. KC is also generally used as a method of improving a conventional (steam) RC. Therefore, the system architecture tends to be similar. It also appears that the working temperatures are slightly higher than some refrigerant based ORCs. “When the ammonia mass fraction is 0.8, the bubble and dew temperatures are around 60.3 and 147.3 oC respectively.” [3]

The ORC is a promising cycle for recovery of energy from low heat sources. However, it has been shown that for certain working fluids, benzene or cyclohexane, the most suitable conditions were an expander inlet pressure of 4.1 MPa and temperature of 290 oC – 300oC. [4]

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. Isobutene also showed improved system performance when compared to R123 and R245fa. [4] 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. [4]

# Appendix A – Acronyms

C – Celsius

F – Fahrenheit

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

# Appendix C – Source Code

import csv

import math

from mpl\_toolkits.mplot3d import axes3d

import matplotlib.pyplot as plt

import numpy as np

def interpolate(x1,y1,x2,y2,x):

try:

y = ((y2-y1)/(x2-x1))\*(x-x1) + y1

except TypeError:

y = y1

return(y)

def vlookup(rfile, index, search\_col, result\_col):

# The file is where the data is stored.

# index is the item to search rows for.

# search\_col is the column in which the index should be searched for.

# result\_col should be the column from which the result should be extracted.

index = float(index)

search\_col = int(search\_col)

result\_col = int(result\_col)

RDR = csv.reader(rfile, dialect = 'excel')

pos\_diff = 1000

neg\_diff = -1000

x1 = None

y1 = None

x2 = None

y2 = None

for row in 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.

try:

diff = index - float(row[search\_col])

except ValueError:

if row[search\_col] == "Inf":

diff = math.inf

#print("Header?")

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.

#----------Main----------#

fig = plt.figure()

fig1 = plt.figure()

fig2 = plt.figure()

fig3 = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax1 = fig1.add\_subplot(111, projection='3d')

ax2 = fig2.add\_subplot(111, projection='3d')

ax3 = fig3.add\_subplot(111, projection='3d')

#b\_press = np.arange(30,154.01,10)

#c\_press = np.arange(0.00127,30, 1)

c\_press = np.linspace(0.1225, 0.5, 25)

b\_press = np.linspace(0.5,1,25)

X = []

X2 = []

Y = []

Y2 = []

Z = []

Z2 = []

for xs in c\_press:

for ys in b\_press:

boiler\_pressure = ys

condenser\_pressure = xs

#print("Boiler pressure: ", boiler\_pressure,"\nCondenser pressure: ",condenser\_pressure)

##boiler\_pressure = 1

##condenser\_pressure = 0.25

temp\_col = 0 # Degrees Celsius

press\_col = 1 # MPa

v\_col = 3 # Specific volume of vapor m3/kg

hl\_col = 4 # Enthalpy of saturated liquid kJ/kg

hv\_col = 5 # Enthalpy of saturated vapor kJ/kg

sl\_col = 6 # Entropy of saturated liquid kJ/(kgK)

sv\_col = 7 # Entropy of saturated vapor kJ/(kgK)

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

db\_path = 'H:\\WIP\\12343 - Research & Development\\Issue #251 - Rankine cycle research\\Additional references'

# Fix states with specified pressures

p1 = boiler\_pressure

p4 = boiler\_pressure

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p1, press\_col, temp\_col)

boiler\_temp = interpolate(x1, y1, x2, y2, p1)

file.close()

p2 = condenser\_pressure

p3 = condenser\_pressure

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, temp\_col)

condenser\_temp = interpolate(x1, y1, x2, y2, p2)

file.close()

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p1, press\_col, hv\_col)

h1 = interpolate(x1, y1, x2, y2, p1)

#print("h1 = ", h1)

file.close()

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p1, press\_col, sv\_col)

s1 = interpolate(x1, y1, x2, y2, p1)

s2 = s1

#print("s1 = ", s1,"\ns2 = ", s2)

file.close()

# Calculate the quality of state 2

# First find the liquid and vapor entropy at the condenser pressure

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, sl\_col)

s2L = interpolate(x1, y1, x2, y2, p2)

file.close()

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, sv\_col)

s2v = interpolate(x1, y1, x2, y2, p2)

file.close()

#print("sL = ",s2L,"\nsv = ", s2v)

try:

qual\_2 = (s2 - s2L)/(s2v - s2L)

except ZeroDivisionError:

qual\_2 = 0

except RuntimeWarning:

qual\_2 = 0

#print("x2 = ", x2)

# Note that evaporating enthalpy is equal to the difference between the enthalpy

# of a saturated vapor and the enthalpy of a saturated liquid at a given

# temperature or pressure.

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, hl\_col)

h2L = interpolate(x1, y1, x2, y2, p2)

#print("h2L = ", h2L)

file.close()

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, hv\_col)

h2v = interpolate(x1, y1, x2, y2, p2)

#print("h2v = ", h2v)

hLv = h2v - h2L

#print("hLv = ", hLv)

file.close()

h2 = h2L + (qual\_2\*hLv)

#print("h2 = ", h2)

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, hl\_col)

h3 = interpolate(x1, y1, x2, y2, p2)

file.close()

#print("h3 = ", h3)

file = open("%s/%s" %(db\_path, R245fa\_db), mode = 'r', newline='')

x1, y1, x2, y2 = vlookup(file, p2, press\_col, v\_col)

v3 = interpolate(x1, y1, x2, y2, p2)

file.close()

#print("v3 = ", v3)

h4 = h3 + v3\*(p4-p3)

#print("h4 = ", h4)

W\_m = h1-h2-h4+h3 # Watts of power per kg/s of mass flow rate

#print("Watts per kg/s of mass flow rate = ", W\_m)

efficiency = ((h1-h2) - (h4-h3))/(h1 - h4)

X.append(boiler\_pressure)

X2.append(boiler\_temp)

Y.append(condenser\_pressure)

Y2.append(condenser\_temp)

Z.append(W\_m)

Z2.append(efficiency)

ax.set\_xlabel("Boiler Pressure (MPa)")

ax.set\_ylabel("Condenser Pressure (MPa)")

ax.set\_zlabel("Power output per unit mass flow rate (Watts)")

ax.scatter(X, Y, Z)

ax1.set\_xlabel("Boiler Pressure (MPa)")

ax1.set\_ylabel("Condenser Pressure (MPa)")

ax1.set\_zlabel("Efficiency")

ax1.scatter(X, Y, Z2)

ax2.set\_xlabel("Boiler Temperature (C)")

ax2.set\_ylabel("Condenser Temperature (C)")

ax2.set\_zlabel("Power output per unit mass flow rate (Watts)")

ax2.scatter(X2, Y2, Z, color='r')

ax3.set\_xlabel("Boiler Temperature (C)")

ax3.set\_ylabel("Condenser Temperature (C)")

ax3.set\_zlabel("Efficiency")

ax3.scatter(X2, Y2, Z2, color='r')

plt.show()

# References

|  |  |
| --- | --- |
| [1] | 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. |
| [2] | C. B. Arvind Thekdi, "Waste Heat Reduction and Recovery Options for Metals Industry," *Energy Technology. John Wiley & Sons, Inc.,* pp. 17-24, 2011. |
| [3] | 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. |
| [4] | 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. |