Project 1 - Turbofan Design Parameter Study

AEEM404063 Airbreathing Propulsion

Matt Boller, Pierce Elliott, Jon Frueh, Ben Rupe

Fall Semester 2023

In [ ]:
```
from Project1BackendCalcs import TFComputation
import numpy as np
import matplotlib.pyplot as plt
```

Ideal Cycle comparison

In [ ]:
```
TF = TFComputation()
data, tau, temp, M = TF.fullCycleCalc(isentropic='T')
print('------------- Data Output -------------')
print("mdot = " + str(np.round(data[0],5)) + " (kg/s)")
print("dia = " + str(np.round(data[1],5))+ " (m)")
print("F/mdot = " + str(np.round(data[2],5))+ " (N/(kg/s))")
print("TSFC = " + str(np.round(data[3],5)))
print("f = " + str(np.round(data[4],5)))
print("Thermal efficiency = " + str(np.round(data[5],5)))
print("Propulsive efficiency = " + str(np.round(data[6],5)))
print("Overall efficiency = " + str(np.round(data[7],5)))

print("Tau f = ", np.round(tau[0],5))
print("Tau cH = ", np.round(tau[1],5))
print("Tau tH = ", np.round(tau[2],5))
print("Tau tL = ", np.round(tau[3],5))

print("T_02 = ", np.round(temp[0],5))
print("T_02.5 = ", np.round(temp[1],5))
print("T_03 = ", np.round(temp[2],5))

print("T_04 = ", np.round(temp[3],5))
print("T_04.5 = ", np.round(temp[4],5))
print("T_05 = ", np.round(temp[5],5))
print("P_02 = ", np.round(temp[6],5))
print("P_025 = ", np.round(temp[7],5))
print("P_03 = ", np.round(temp[8],5))
print("P_04 = ", np.round(temp[9],5))
print("P_045 = ", np.round(temp[10],5))
print("P_05 = ", np.round(temp[11],5))

print("M9 = ", np.round(M[0],5))
print("M19 = ", np.round(M[1],5))
print("C9 = ", np.round(M[3],4))
print("C19 = ", np.round(M[4],4))
print("C0 = ", np.round(M[2],4))
```

```
------------- Data Output -------------
mdot = 476.45532 (kg/s)
dia = 2.57505 (m)
F/mdot = 147.36468 (N/(kg/s))
TSFC = 15.09736
f = 0.02447
Thermal efficiency = 0.60844
Propulsive efficiency = 0.63394
Overall efficiency = 0.38572
Tau f =  1.12282
Tau cH =  2.78393
Tau tH =  0.72094
Tau tL =  0.73891
T_02 =  248.26494
T_02.5 =  278.7579
T_03 =  776.04166
T_04 =  1560
T_04.5 =  1124.66012
T_05 =  831.01931
P_02 =  36408.2897
P_025 =  54612.43456
P_03 =  1966047.64398
P_04 =  1966047.64398
P_045 =  530583.92246
P_05 =  158022.43034
M9 =  1.93547
M19 =  1.19389
C9 =  856.5817
C19 =  352.5275
C0 =  250.9859
```

Real Cycle comparison

In [ ]:
```
#B = 10.0; pi_f = 1.5; pi_c = 36
data, tau, temp, M = TF.fullCycleCalc(isentropic='F')
print('------------- Data Output -------------')
print("mdot = " + str(np.round(data[0],5)) + " (kg/s)")
print("dia = " + str(np.round(data[1],5))+ " (m)")
print("F/mdot = " + str(np.round(data[2],5))+ " (N/(kg/s))")
print("TSFC = " + str(np.round(data[3],5)))
print("f = " + str(np.round(data[4],5)))
print("Thermal efficiency = " + str(np.round(data[5],5)))
print("Propulsive efficiency = " + str(np.round(data[6],5)))
print("Overall efficiency = " + str(np.round(data[7],5)))

print("Tau f = ", np.round(tau[0],5))
print("Tau cH = ", np.round(tau[1],5))
print("Tau tH = ", np.round(tau[2],5))
print("Tau tL = ", np.round(tau[3],5))

print("T_02 = ", np.round(temp[0],5))
print("T_02.5 = ", np.round(temp[1],5))
print("T_03 = ", np.round(temp[2],5))

print("T_04 = ", np.round(temp[3],5))
print("T_04.5 = ", np.round(temp[4],5))
print("T_05 = ", np.round(temp[5],5))
print("P_02 = ", np.round(temp[6],5))
print("P_025 = ", np.round(temp[7],5))
print("P_03 = ", np.round(temp[8],5))
print("P_04 = ", np.round(temp[9],5))
print("P_045 = ", np.round(temp[10],5))
print("P_05 = ", np.round(temp[11],5))

print("M9 = ", np.round(M[0],5))
print("M19 = ", np.round(M[1],5))
```

```
------------- Data Output -------------
mdot = 572.54288 (kg/s)
dia = 2.82279 (m)
F/mdot = 122.63306 (N/(kg/s))
TSFC = 16.39353
f = 0.02211
Thermal efficiency = 0.46572
Propulsive efficiency = 0.76274
Overall efficiency = 0.35522
Tau f =  1.13902
Tau cH =  3.11935
Tau tH =  0.66764
Tau tL =  0.68464
T_02 =  248.26494
T_02.5 =  282.77792
T_03 =  882.08366
T_04 =  1560
T_04.5 =  1041.51289
T_05 =  713.06638
P_02 =  36087.5305
P_025 =  54131.29575
P_03 =  1948726.64689
P_04 =  1870777.58102
P_045 =  310180.65077
P_05 =  57517.55814
M9 =  1.24515
M19 =  1.17946
```

Bypass Ratio (BPR) Design Study

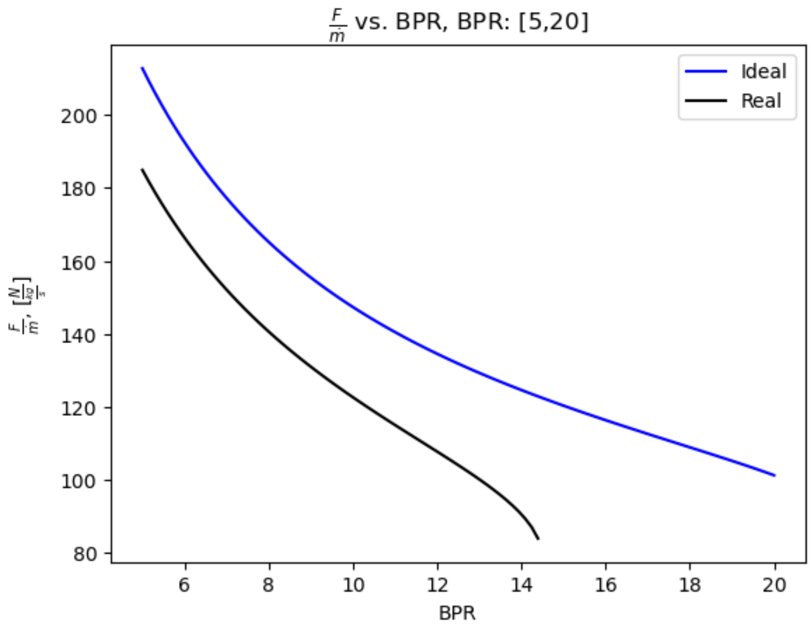
Our team has performed a robust analysis on our turbofan BPR and will demonstrate below the effect of bypass ratios ranging from 5 to 20 on the $\frac{F}{\dot{m}}$, TSFC, f, η_T, η_P, and η_O

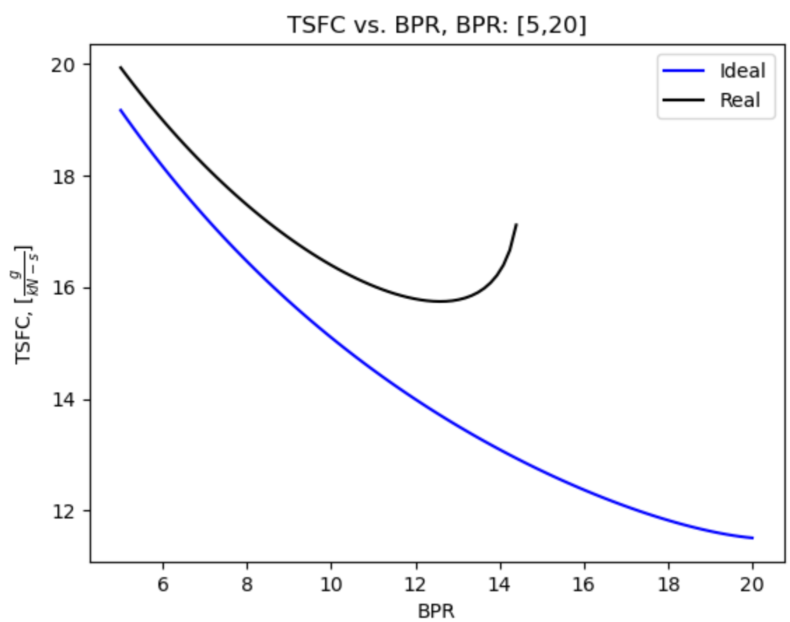
In [ ]:
```
### BPR study ###

# Creates the BPR ranges
BPRsweep = np.linspace(5,20,100)

# Creates the output arrays
FmdotBPR_I = np.zeros(len(BPRsweep))
TSFCBPR_I = np.zeros(len(BPRsweep))
fBPR_I = np.zeros(len(BPRsweep))
nT_BPR_I = np.zeros(len(BPRsweep))
nP_BPR_I = np.zeros(len(BPRsweep))
nO_BPR_I = np.zeros(len(BPRsweep))
```

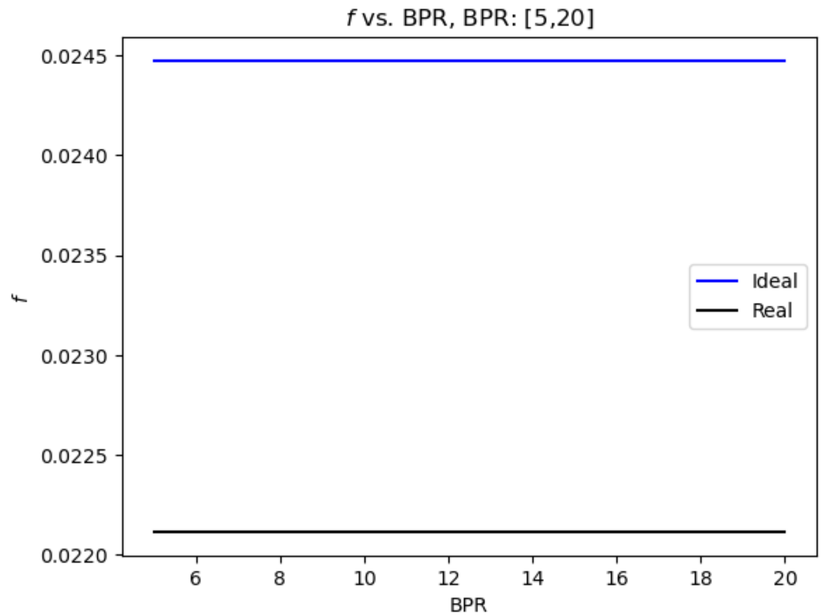
```python
FmdotBPR_R = np.zeros(len(BPRsweep))
TSFCBPR_R = np.zeros(len(BPRsweep))
fBPR_R = np.zeros(len(BPRsweep))
nT_BPR_R = np.zeros(len(BPRsweep))
nP_BPR_R = np.zeros(len(BPRsweep))
nO_BPR_R = np.zeros(len(BPRsweep))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]

# Sweeps through to calculate all the values
# Isentropic
for i in range(len(BPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[i],pi_f=1.5,pi_c=36,isentropic='T')
    FmdotBPR_I[i] = data[2]
    TSFCBPR_I[i] = data[3]
    fBPR_I[i] = data[4]
    nT_BPR_I[i] = data[5]*100
    nP_BPR_I[i] = data[6]*100
    nO_BPR_I[i] = data[7]*100

# Real
for j in range(len(BPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[j],pi_f=1.5,pi_c=36,isentropic='F')
    FmdotBPR_R[j] = data[2]
    TSFCBPR_R[j] = data[3]
    fBPR_R[j] = data[4]
    nT_BPR_R[j] = data[5]*100
    nP_BPR_R[j] = data[6]*100
    nO_BPR_R[j] = data[7]*100



# Creates the plots

# F/mdot Plots
plt.plot(BPRsweep, FmdotBPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, FmdotBPR_R, 'k-',label='Real')
plt.title("$\\frac{F}{\dot{m}}$ vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('$\\frac{F}{\dot{m}}$, $[\\frac{N}{\\frac{kg}{s}}]$')
plt.legend()
plt.show()

# TSFC Plots
plt.plot(BPRsweep, TSFCBPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, TSFCBPR_R, 'k-',label='Real')
plt.title("TSFC vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('TSFC, $[\\frac{g}{kN-s}]$')
plt.legend()
plt.show()

# f Plots
plt.plot(BPRsweep, fBPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, fBPR_R, 'k-',label='Real')
plt.title("$f$ vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('$f$')
plt.legend()
plt.show()


# nT Plots
plt.plot(BPRsweep, nT_BPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, nT_BPR_R, 'k-',label='Real')
plt.title("$\\eta_T$ vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('$\\eta_T$, %')
plt.legend()
plt.show()


# nP Plots
plt.plot(BPRsweep, nP_BPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, nP_BPR_R, 'k-',label='Real')
plt.title("$\\eta_P$ vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('$\\eta_P$, %')
plt.legend()
plt.show()

# nO Plots
plt.plot(BPRsweep, nO_BPR_I, 'b-',label='Ideal')
plt.plot(BPRsweep, nO_BPR_R, 'k-',label='Real')
```

```
plt.title("$\\eta_O$ vs. BPR, BPR: [5,20]")
plt.xlabel('BPR')
plt.ylabel('$\\eta_O$, %')
plt.legend()
plt.show()
```

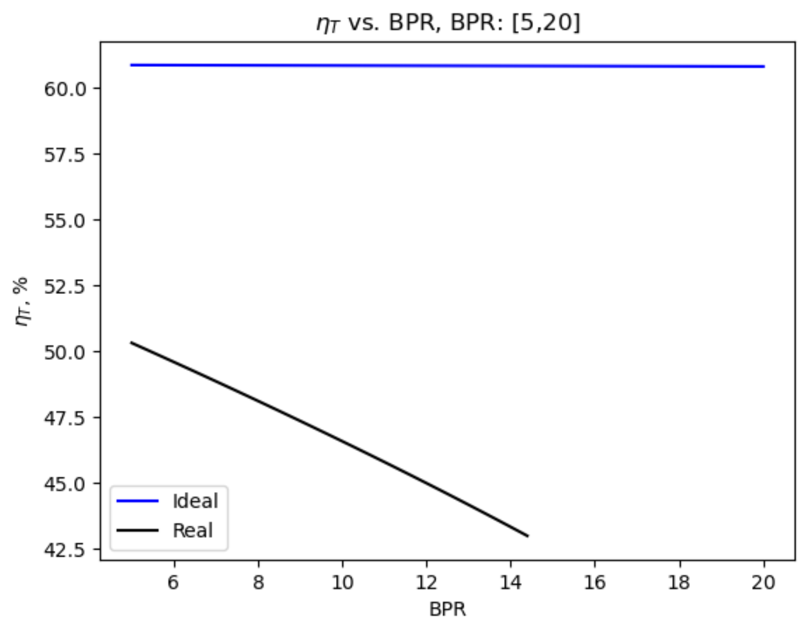
```
c:\Users\Delli\Documents\GitHub\AirbreathingProject1\Project1BackendCalcs.py:200: RuntimeWarning: invalid value encountered in sqr
t
  M9 = np.sqrt((1/(1-nj*(-(pa/self.P_05)**((yh-1)/yh) + 1))-1)*2/(yh-1))
```



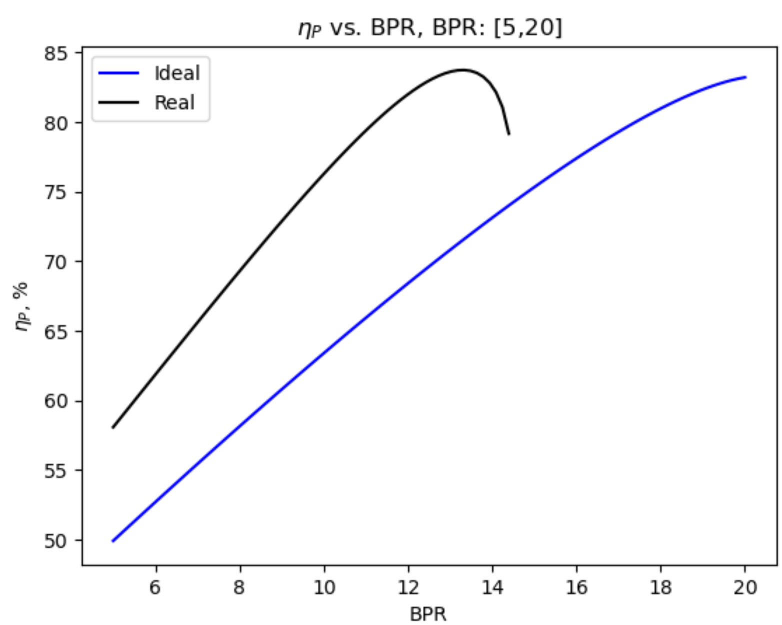




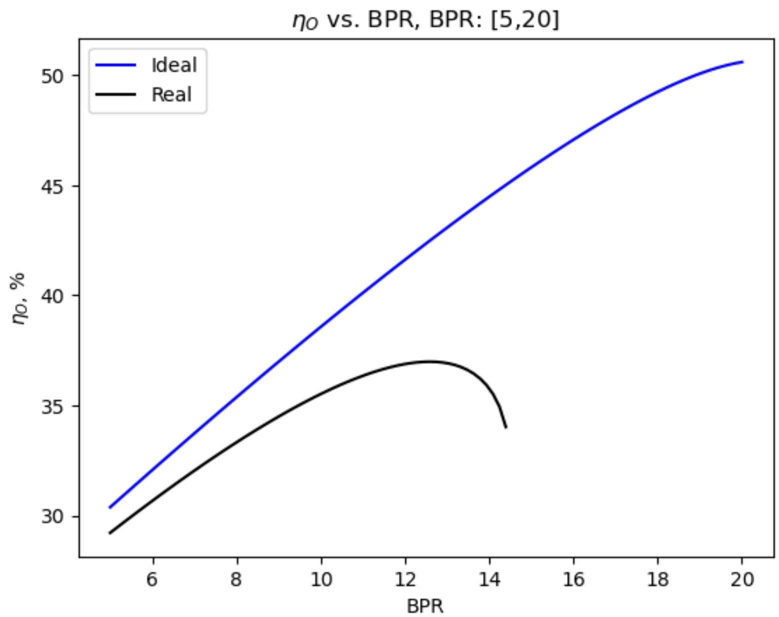
f vs. BPR, BPR: [5,20]
0.0245
0.0240
0.0235
0.0230
0.0225
0.0220
f
6
8
10
12
14
16
18
20
BPR
Ideal
Real



η_T vs. BPR, BPR: [5,20]
60.0
57.5
55.0
52.5
50.0
47.5
45.0
42.5
η_T, %
6
8
10
12
14
16
18
20
BPR
Ideal
Real



η_P vs. BPR, BPR: [5,20]
85
80
75
70
65
60
55
50
η_P, %
6
8
10
12
14
16
18
20
BPR
Ideal
Real

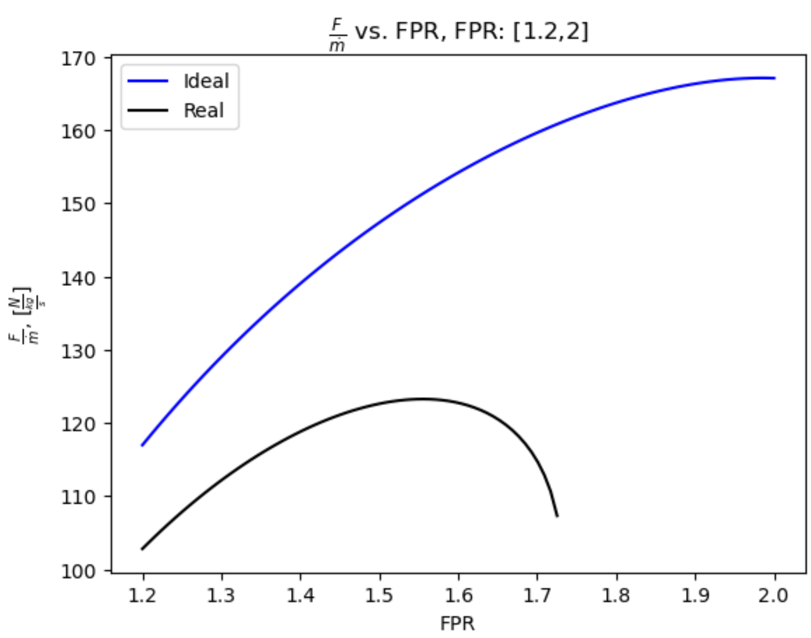


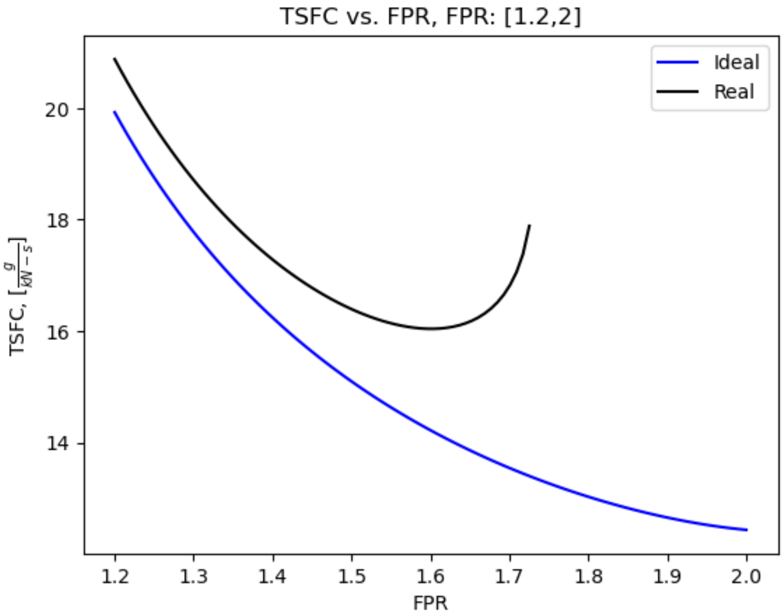
Fan Pressure Ratio (FPR) Design Study

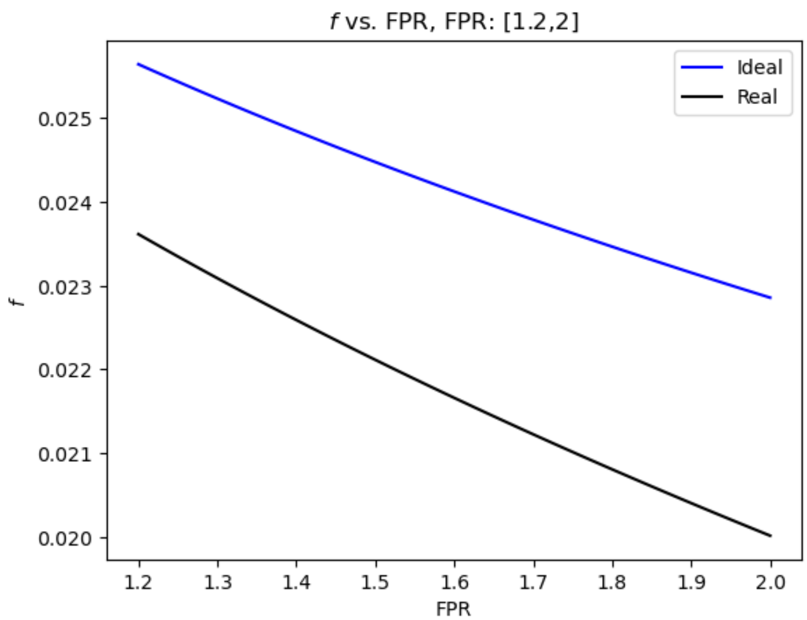
Analysis of the effect of FPR ranging from 1.2 to 2.0 on the $\frac{F}{\dot{m}}$, TSFC, f, η_T, η_P, and η_O will be demonstrated below

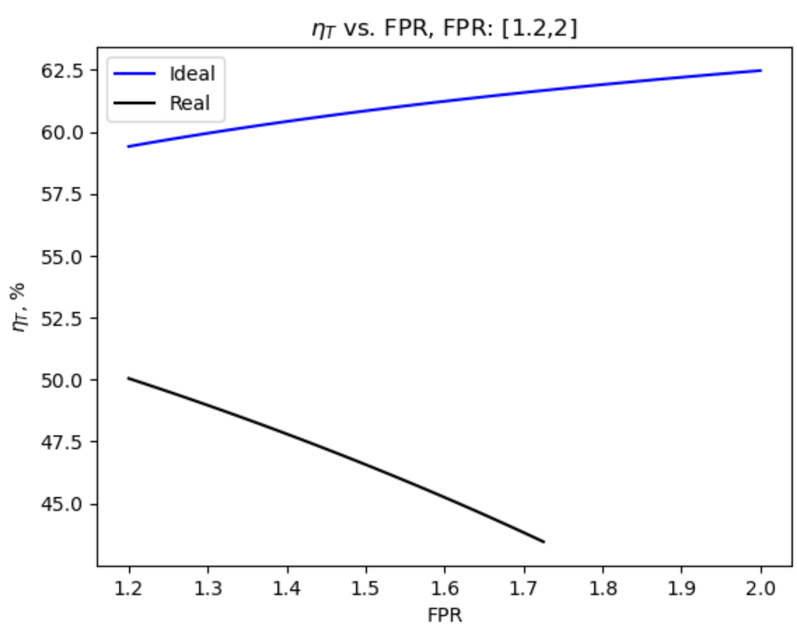
In [ ]:
```
### FPR study ###

# Creates the FPR ranges
FPRsweep = np.linspace(1.2,2,100)

# Creates the output arrays
FmdotFPR_I = np.zeros(len(FPRsweep))
TSFCFPR_I = np.zeros(len(FPRsweep))
fFPR_I = np.zeros(len(FPRsweep))
nT_FPR_I = np.zeros(len(FPRsweep))
nP_FPR_I = np.zeros(len(FPRsweep))
nO_FPR_I = np.zeros(len(FPRsweep))

FmdotFPR_R = np.zeros(len(FPRsweep))
TSFCFPR_R = np.zeros(len(FPRsweep))
fFPR_R = np.zeros(len(FPRsweep))
nT_FPR_R = np.zeros(len(FPRsweep))
nP_FPR_R = np.zeros(len(FPRsweep))
nO_FPR_R = np.zeros(len(FPRsweep))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]

# Sweeps through to calculate all the values
# Isentropic
for i in range(len(FPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=FPRsweep[i],pi_c=36,isentropic='T')
    FmdotFPR_I[i] = data[2]
    TSFCFPR_I[i] = data[3]
    fFPR_I[i] = data[4]
    nT_FPR_I[i] = data[5]*100
    nP_FPR_I[i] = data[6]*100
    nO_FPR_I[i] = data[7]*100

# Real
for j in range(len(FPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=FPRsweep[j],pi_c=36,isentropic='F')
    FmdotFPR_R[j] = data[2]
    TSFCFPR_R[j] = data[3]
    fFPR_R[j] = data[4]
    nT_FPR_R[j] = data[5]*100
    nP_FPR_R[j] = data[6]*100
    nO_FPR_R[j] = data[7]*100



# Creates the plots

# F/mdot Plots
plt.plot(FPRsweep, FmdotFPR_I, 'b-',label='Ideal')
```

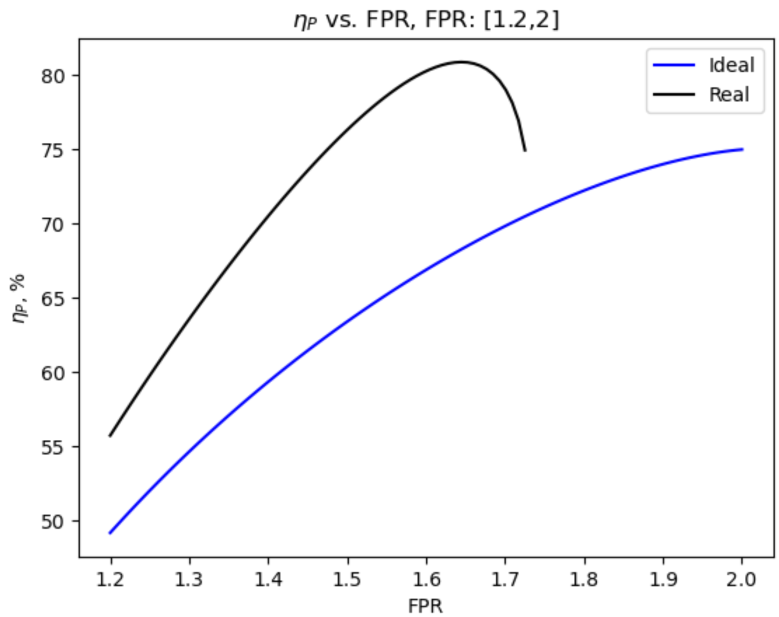
```
plt.plot(FPRsweep, FmdotFPR_R, 'k-',label='Real')
plt.title("$\\frac{F}{\dot{m}}$ vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('$\\frac{F}{\dot{m}}$, $[\\frac{N}{\\frac{kg}{s}}]$')
plt.legend()
plt.show()

# TSFC Plots
plt.plot(FPRsweep, TSFCFPR_I, 'b-',label='Ideal')
plt.plot(FPRsweep, TSFCFPR_R, 'k-',label='Real')
plt.title("TSFC vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('TSFC, $[\\frac{g}{kN-s}]$')
plt.legend()
plt.show()

# f Plots
plt.plot(FPRsweep, fFPR_I, 'b-',label='Ideal')
plt.plot(FPRsweep, fFPR_R, 'k-',label='Real')
plt.title("$f$ vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('$f$')
plt.legend()
plt.show()


# nT Plots
plt.plot(FPRsweep, nT_FPR_I, 'b-',label='Ideal')
plt.plot(FPRsweep, nT_FPR_R, 'k-',label='Real')
plt.title("$\\eta_T$ vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('$\\eta_T$, %')
plt.legend()
plt.show()


# nP Plots
plt.plot(FPRsweep, nP_FPR_I, 'b-',label='Ideal')
plt.plot(FPRsweep, nP_FPR_R, 'k-',label='Real')
plt.title("$\\eta_P$ vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('$\\eta_P$, %')
plt.legend()
plt.show()

# nO Plots
plt.plot(FPRsweep, nO_FPR_I, 'b-',label='Ideal')
plt.plot(FPRsweep, nO_FPR_R, 'k-',label='Real')
plt.title("$\\eta_O$ vs. FPR, FPR: [1.2,2]")
plt.xlabel('FPR')
plt.ylabel('$\\eta_O$, %')
plt.legend()
plt.show()
```

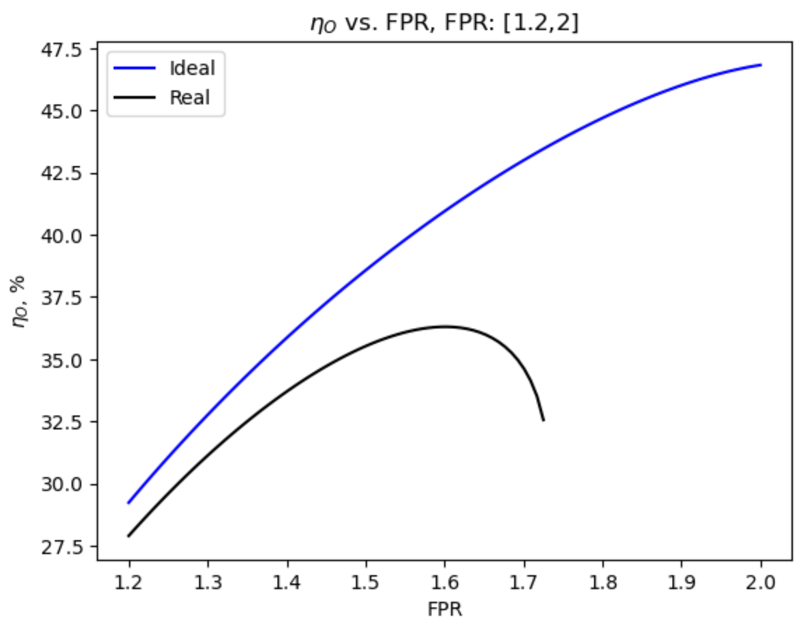



TSFC vs. FPR, FPR: [1.2,2]
Ideal
Real
TSFC, [$\frac{g}{kN-s}$]
20
18
16
14
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
FPR


f vs. FPR, FPR: [1.2,2]
Ideal
Real
f
0.025
0.024
0.023
0.022
0.021
0.020
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
FPR


η_T vs. FPR, FPR: [1.2,2]
Ideal
Real
η_T, %
62.5
60.0
57.5
55.0
52.5
50.0
47.5
45.0
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
FPR



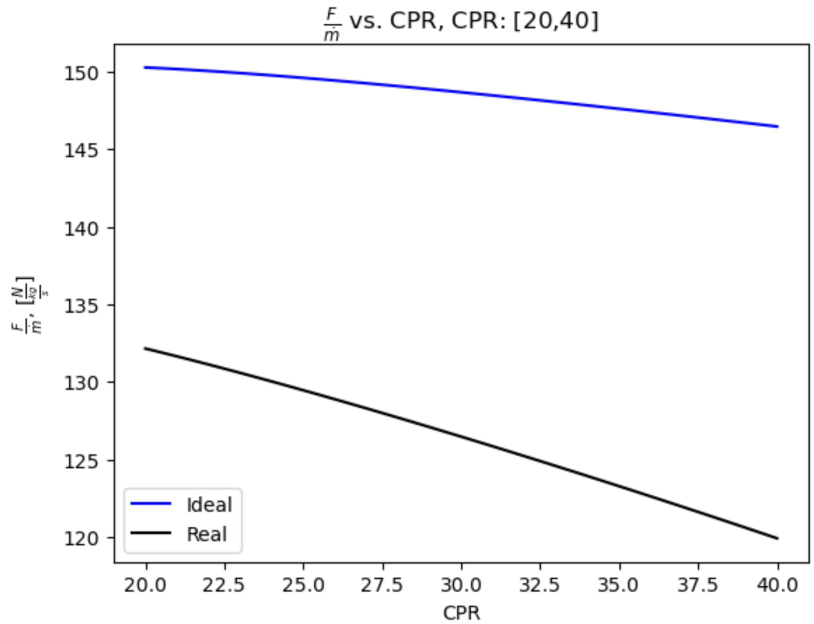


Compressor Pressure Ratio Design Study

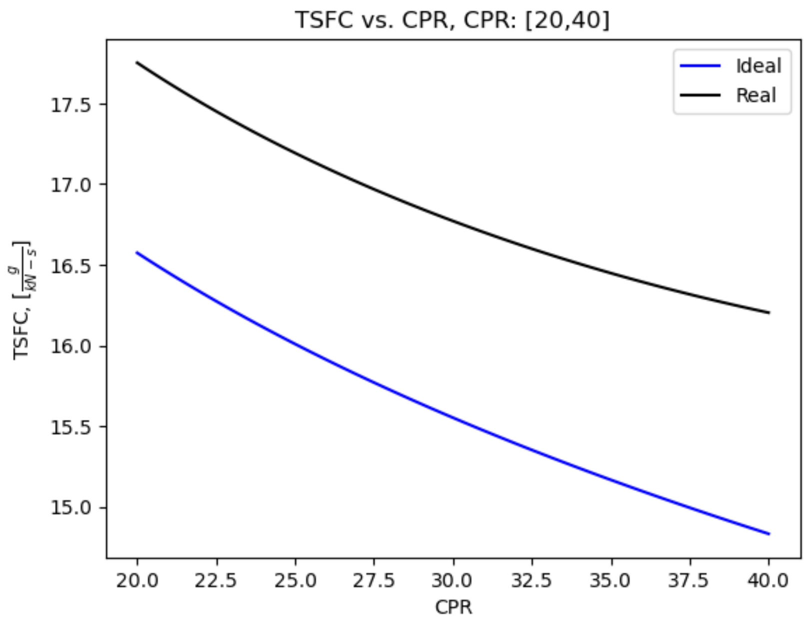
Analysis of the effect of compressor pressure ratio values ranging from 20 to 40 on the $\frac{F}{\dot{m}}$, TSFC, f, η_T, η_P, and η_O will be demonstrated below

In [ ]:
```
### CPR study ###

# Creates the CPR ranges
CPRsweep = np.linspace(20,40,100)

# Creates the output arrays
FmdotCPR_I = np.zeros(len(CPRsweep))
TSFCCPR_I = np.zeros(len(CPRsweep))
fCPR_I = np.zeros(len(CPRsweep))
nT_CPR_I = np.zeros(len(CPRsweep))
nP_CPR_I = np.zeros(len(CPRsweep))
nO_CPR_I = np.zeros(len(CPRsweep))

FmdotCPR_R = np.zeros(len(CPRsweep))
TSFCCPR_R = np.zeros(len(CPRsweep))
fCPR_R = np.zeros(len(CPRsweep))
nT_CPR_R = np.zeros(len(CPRsweep))
nP_CPR_R = np.zeros(len(CPRsweep))
nO_CPR_R = np.zeros(len(CPRsweep))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]
```

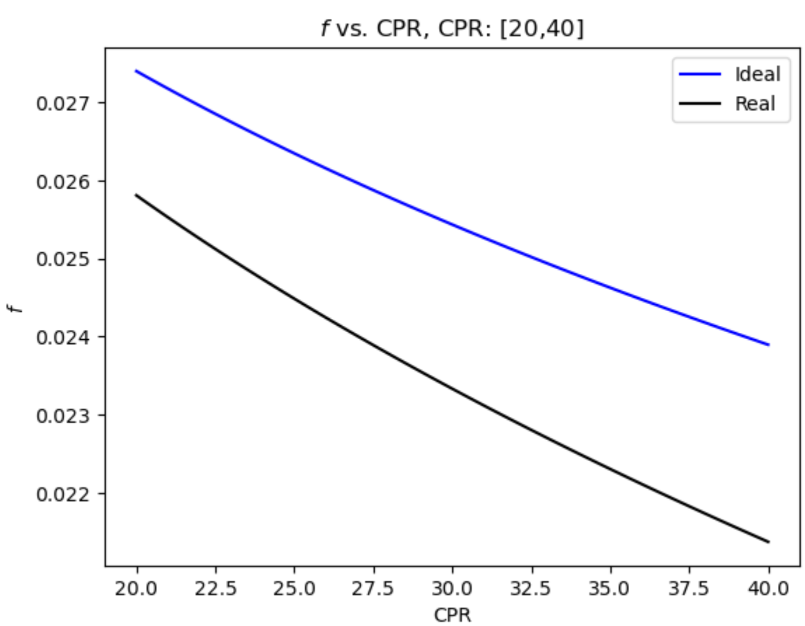
```python
# Sweeps through to calculate all the values
# Isentropic
for i in range(len(CPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=1.5,pi_c=CPRsweep[i],isentropic='T')
    FmdotCPR_I[i] = data[2]
    TSFCCPR_I[i] = data[3]
    fCPR_I[i] = data[4]
    nT_CPR_I[i] = data[5]*100
    nP_CPR_I[i] = data[6]*100
    nO_CPR_I[i] = data[7]*100

# Real
for j in range(len(CPRsweep)):
    data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=1.5,pi_c=CPRsweep[j],isentropic='F')
    FmdotCPR_R[j] = data[2]
    TSFCCPR_R[j] = data[3]
    fCPR_R[j] = data[4]
    nT_CPR_R[j] = data[5]*100
    nP_CPR_R[j] = data[6]*100
    nO_CPR_R[j] = data[7]*100



# Creates the plots

# F/mdot Plots
plt.plot(CPRsweep, FmdotCPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, FmdotCPR_R, 'k-',label='Real')
plt.title("$\\frac{F}{\dot{m}}$ vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('$\\frac{F}{\dot{m}}$, $[\\frac{N}{\\frac{kg}{s}}]$')
plt.legend()
plt.show()

# TSFC Plots
plt.plot(CPRsweep, TSFCCPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, TSFCCPR_R, 'k-',label='Real')
plt.title("TSFC vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('TSFC, $[\\frac{g}{kN-s}]$')
plt.legend()
plt.show()

# f Plots
plt.plot(CPRsweep, fCPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, fCPR_R, 'k-',label='Real')
plt.title("$f$ vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('$f$')
plt.legend()
plt.show()


# nT Plots
plt.plot(CPRsweep, nT_CPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, nT_CPR_R, 'k-',label='Real')
plt.title("$\\eta_T$ vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('$\\eta_T$, %')
plt.legend()
plt.show()


# nP Plots
plt.plot(CPRsweep, nP_CPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, nP_CPR_R, 'k-',label='Real')
plt.title("$\\eta_P$ vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('$\\eta_P$, %')
plt.legend()
plt.show()

# nO Plots
plt.plot(CPRsweep, nO_CPR_I, 'b-',label='Ideal')
plt.plot(CPRsweep, nO_CPR_R, 'k-',label='Real')
plt.title("$\\eta_O$ vs. CPR, CPR: [20,40]")
plt.xlabel('CPR')
plt.ylabel('$\\eta_O$, %')
plt.legend()
plt.show()
```



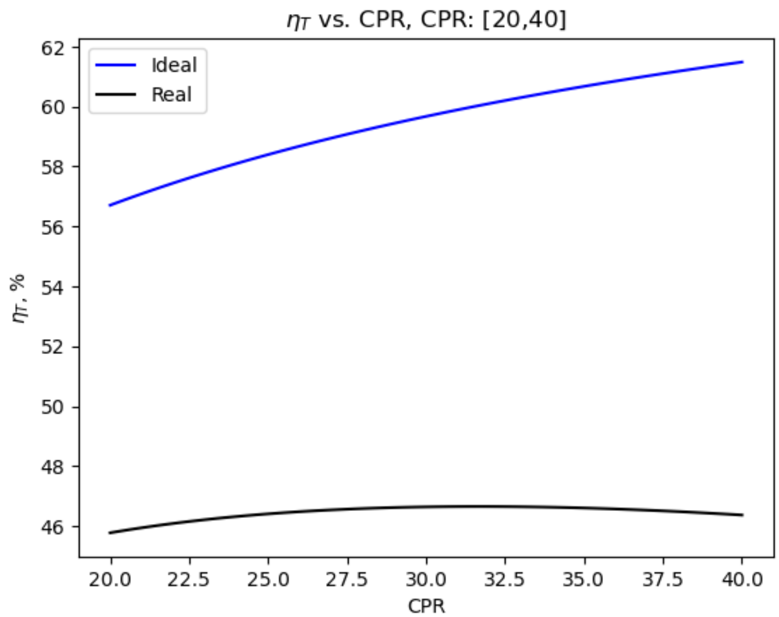
$\frac{F}{\dot{m}}$ vs. CPR, CPR: [20,40]
$\frac{F}{\dot{m}}$, $[\frac{N}{\frac{kg}{s}}]$
150
145
140
135
130
125
120
Ideal
Real
20.0
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
CPR

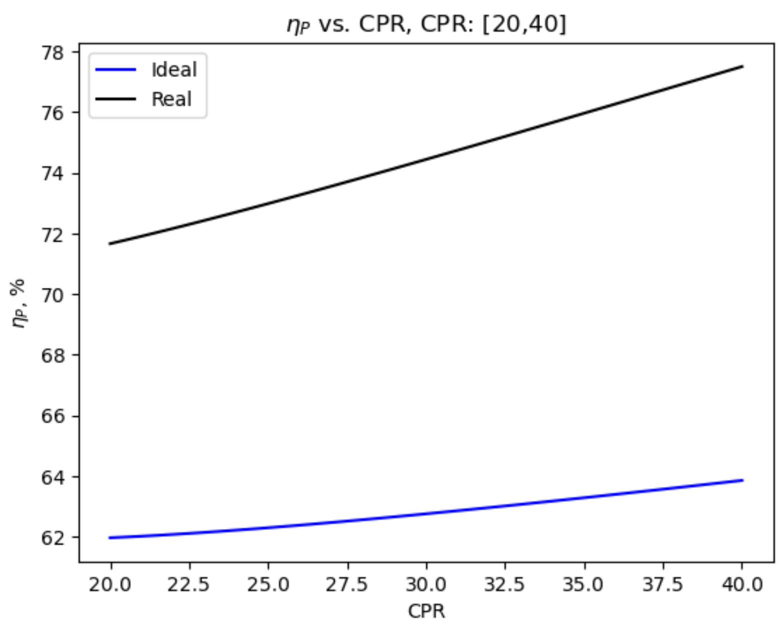


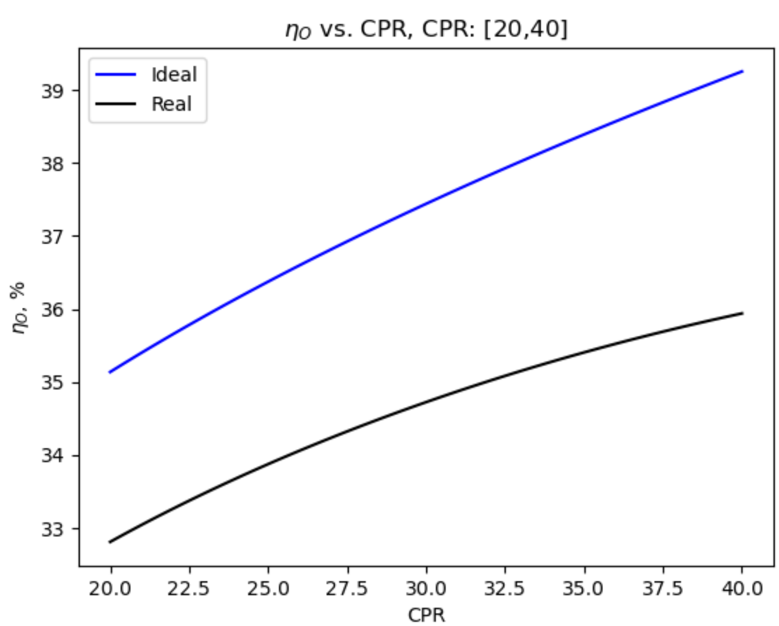
TSFC vs. CPR, CPR: [20,40]
TSFC, $[\frac{g}{kN-s}]$
17.5
17.0
16.5
16.0
15.5
15.0
Ideal
Real
20.0
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
CPR



f vs. CPR, CPR: [20,40]
f
0.027
0.026
0.025
0.024
0.023
0.022
Ideal
Real
20.0
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
CPR


η_T vs. CPR, CPR: [20,40]
Ideal
Real
η_T, %
CPR


η_P vs. CPR, CPR: [20,40]
Ideal
Real
η_P, %
CPR


η_O vs. CPR, CPR: [20,40]
Ideal
Real
η_O, %
CPR

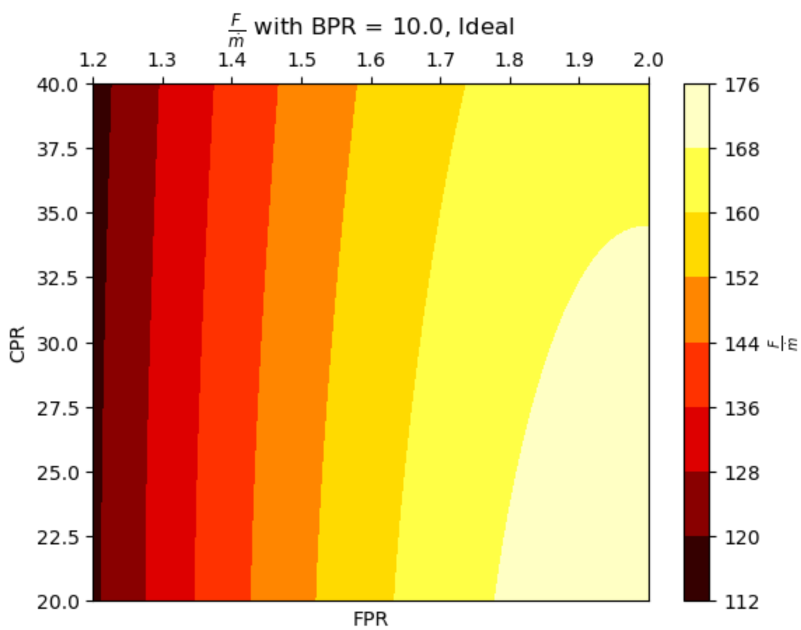
Combining Parameters

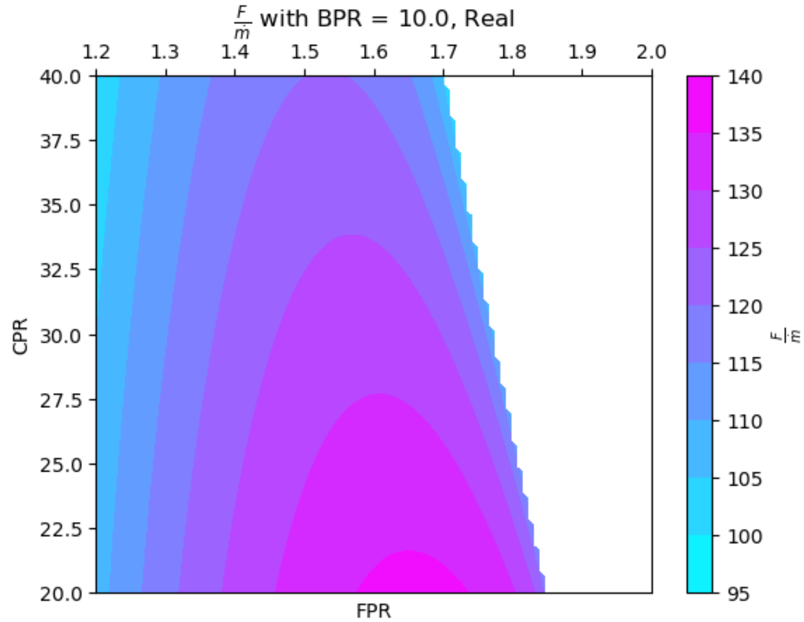
Now, varying multiple variables can create a mesh to find the optimal parameters

In [ ]:
```
### BPR Combined Study ###

# Holding BPR = 10

FPRsweep = np.linspace(1.2,2.0,100)
CPRsweep = np.linspace(20,40,100)


# Creates the output arrays
FmdotBPRConst_I = np.zeros((100,100))
TSFCBPRConst_I = np.zeros((100,100))
fBPRConst_I = np.zeros((100,100))
nT_BPRConst_I = np.zeros((100,100))
nP_BPRConst_I = np.zeros((100,100))
nO_BPRConst_I = np.zeros((100,100))

FmdotBPRConst_R = np.zeros((100,100))
TSFCBPRConst_R = np.zeros((100,100))
fBPRConst_R = np.zeros((100,100))
nT_BPRConst_R = np.zeros((100,100))
nP_BPRConst_R = np.zeros((100,100))
nO_BPRConst_R = np.zeros((100,100))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]

# Sweeps through to calculate all the values
# Isentropic
row = 0
col = 0
for i in range(len(CPRsweep)):
    for j in range(len(FPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=FPRsweep[j],pi_c=CPRsweep[i],isentropic='T')
        FmdotBPRConst_I[i][j] = data[2]
        TSFCBPRConst_I[i][j] = data[3]
        fBPRConst_I[i][j] = data[4]
        nT_BPRConst_I[i][j]= data[5]*100
        nP_BPRConst_I[i][j] = data[6]*100
        nO_BPRConst_I[i][j] = data[7]*100
        col += 1
    row += 1

# Real
row = 0
col = 0
for i in range(len(CPRsweep)):
    for j in range(len(FPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=10,pi_f=FPRsweep[j],pi_c=CPRsweep[i],isentropic='F')
        FmdotBPRConst_R[i][j] = data[2]
        TSFCBPRConst_R[i][j] = data[3]
        fBPRConst_R[i][j] = data[4]
        nT_BPRConst_R[i][j] = data[5]*100
        nP_BPRConst_R[i][j] = data[6]*100
        nO_BPRConst_R[i][j] = data[7]*100
        col += 1
    row+= 1


# Creates the plots

##### F/mdot #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,FmdotBPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,FmdotBPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
```

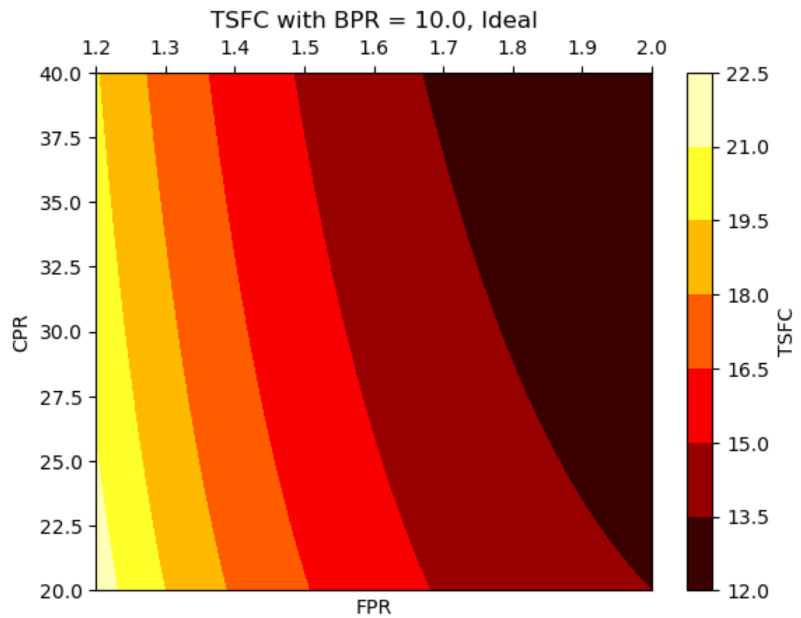
```python
# plt.gca().invert_yaxis()





###### TSFC ######
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,TSFCBPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,TSFCBPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### f  #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,fBPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,fBPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### nT #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nT_BPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nT_BPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### nP #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nP_BPRConst_I,cmap='hot')
cbar = plt.colorbar()
```

```
cbar.set_label("$\\eta_P$")
ax1.set_title('$\\eta_P$ with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nP_BPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_P$")
ax1.set_title('$\\eta_P$ with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()




##### nO #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nO_BPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
ax1.set_title('$\\eta_O$ with BPR = 10.0, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,CPRsweep,nO_BPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
ax1.set_title('$\\eta_O$ with BPR = 10.0, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()
```

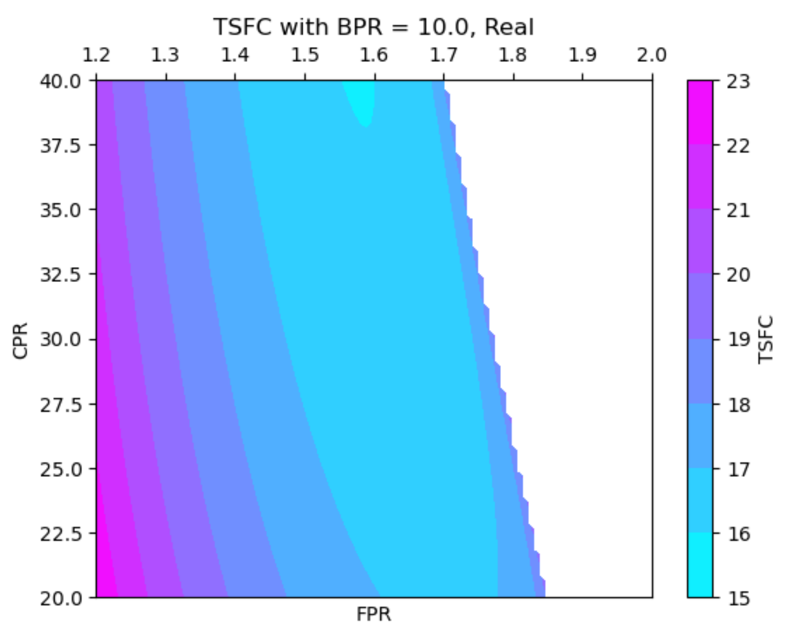




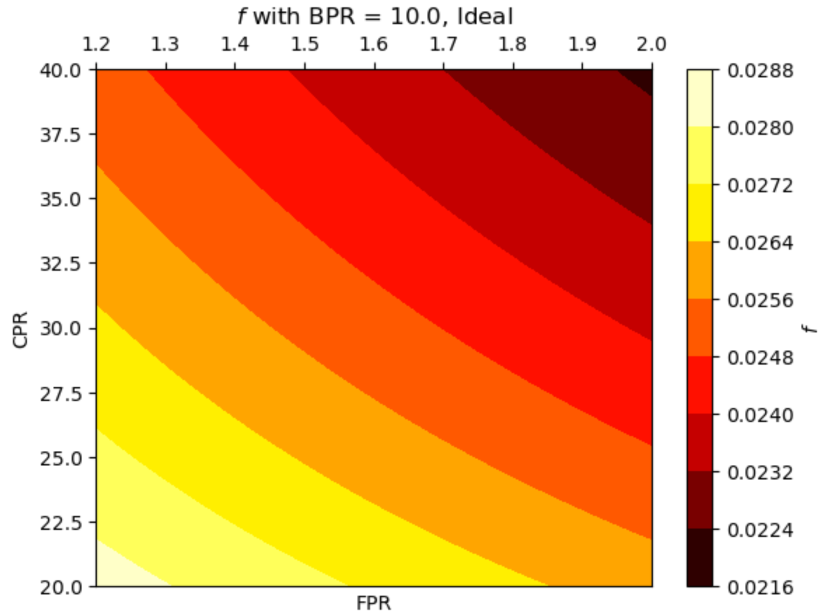
$\frac{F}{m}$ with BPR = 10.0, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
140 135 130 125 120 115 110 105 100 95
$\frac{F}{m}$



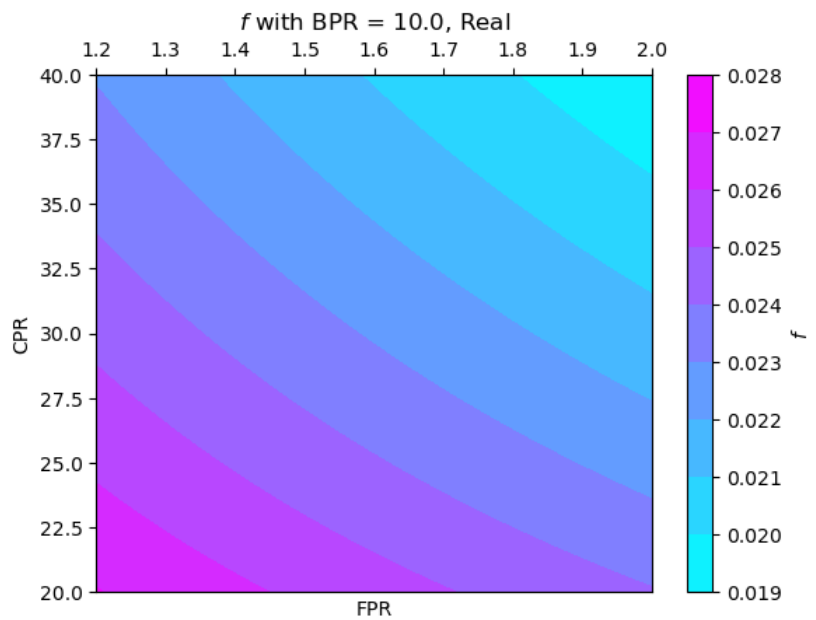
TSFC with BPR = 10.0, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
22.5 21.0 19.5 18.0 16.5 15.0 13.5 12.0
TSFC



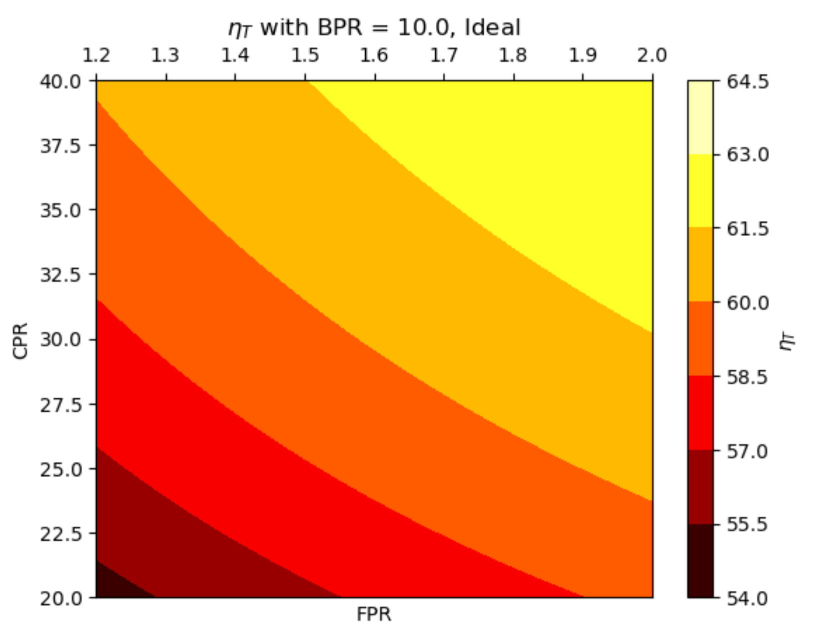
TSFC with BPR = 10.0, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
23 22 21 20 19 18 17 16 15
TSFC



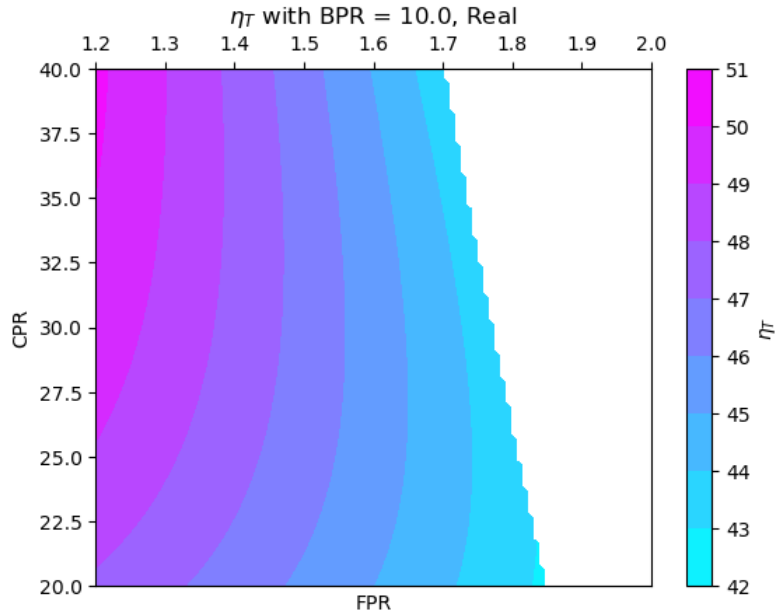
f with BPR = 10.0, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
0.0288 0.0280 0.0272 0.0264 0.0256 0.0248 0.0240 0.0232 0.0224 0.0216
f



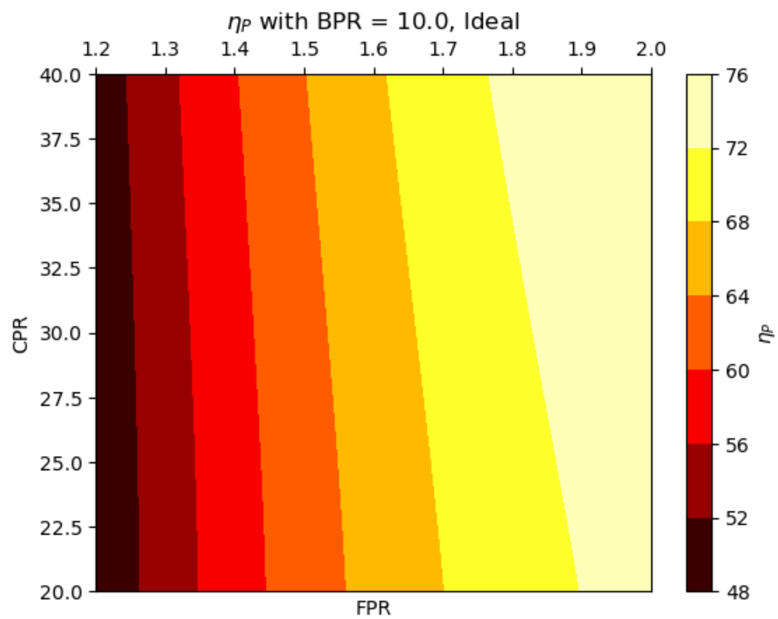
f with BPR = 10.0, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
0.028 0.027 0.026 0.025 0.024 0.023 0.022 0.021 0.020 0.019
f



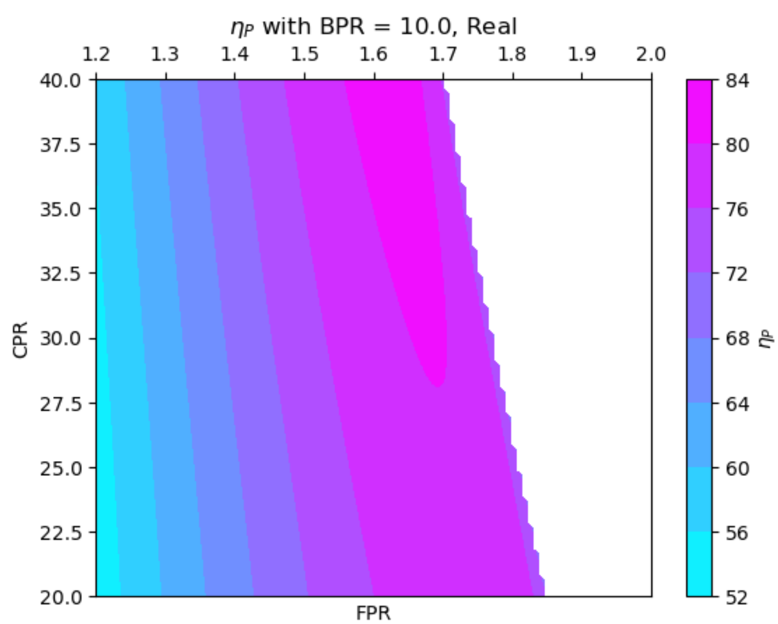
η_T with BPR = 10.0, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
64.5 63.0 61.5 60.0 58.5 57.0 55.5 54.0
η_T



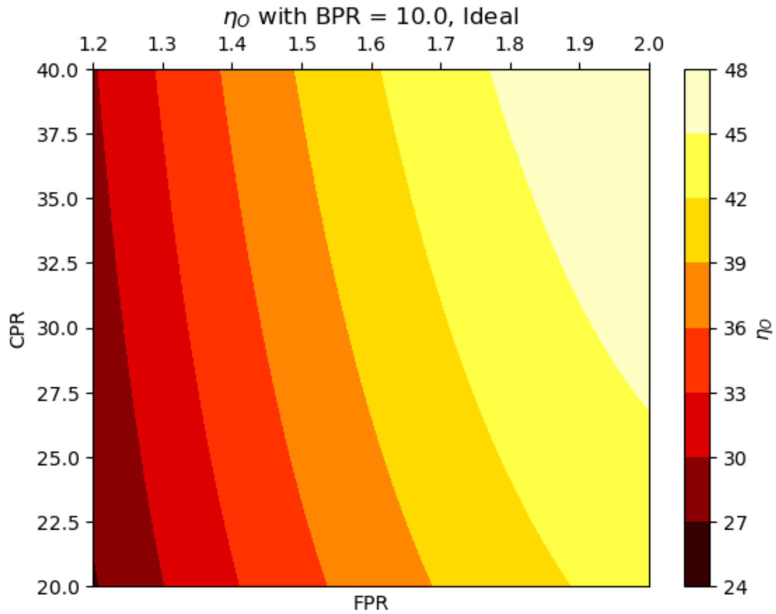
η_T with BPR = 10.0, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
51 50 49 48 47 46 45 44 43 42
η_T

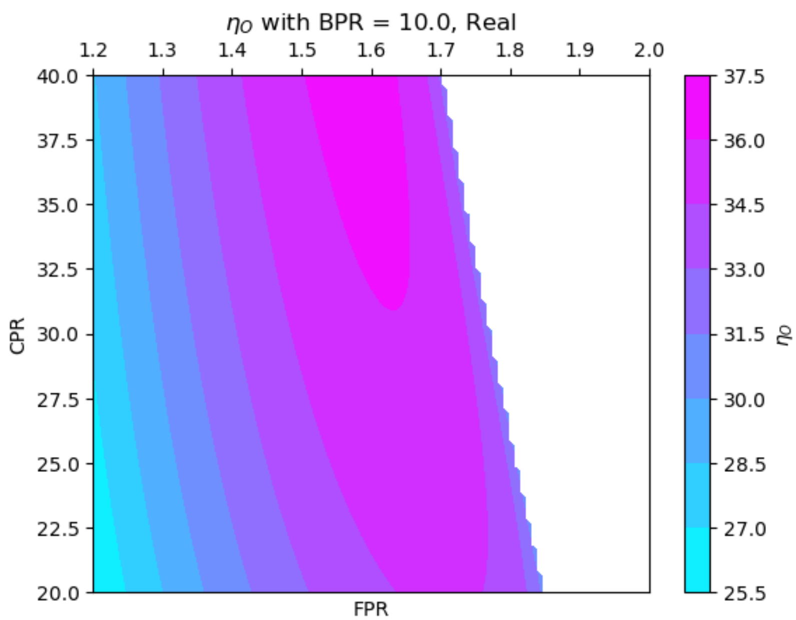


η_P with BPR = 10.0, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
76 72 68 64 60 56 52 48
η_P



η_P with BPR = 10.0, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
FPR
84 80 76 72 68 64 60 56 52
η_P





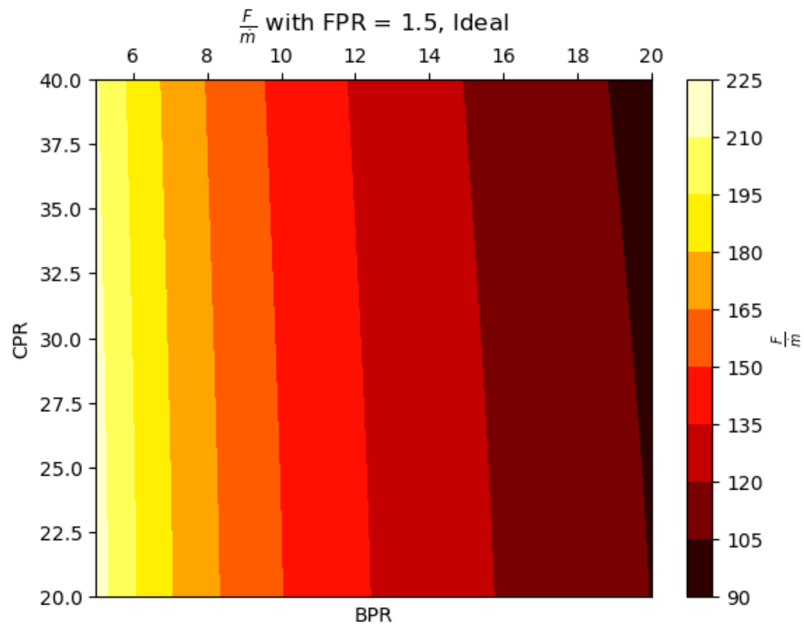
In [ ]:

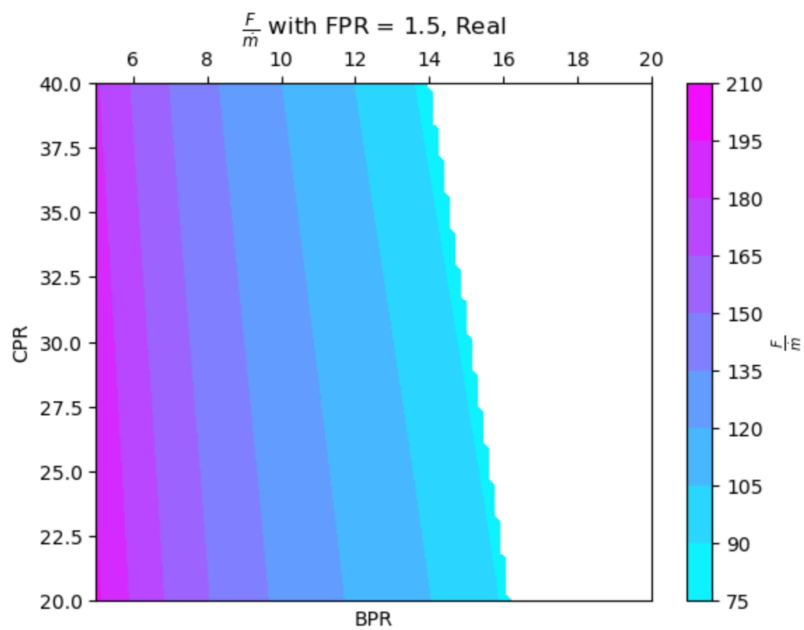
```
### FPR Combined Study ###

# Holding FPR = 1.5

BPRsweep = np.linspace(5,20,100)
CPRsweep = np.linspace(20,40,100)


# Creates the output arrays
FmdotFPRConst_I = np.zeros((100,100))
TSFCFPRConst_I = np.zeros((100,100))
fFPRConst_I = np.zeros((100,100))
nT_FPRConst_I = np.zeros((100,100))
nP_FPRConst_I = np.zeros((100,100))
nO_FPRConst_I = np.zeros((100,100))

FmdotFPRConst_R = np.zeros((100,100))
TSFCFPRConst_R = np.zeros((100,100))
fFPRConst_R = np.zeros((100,100))
nT_FPRConst_R = np.zeros((100,100))
nP_FPRConst_R = np.zeros((100,100))
nO_FPRConst_R = np.zeros((100,100))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]

# Sweeps through to calculate all the values
```

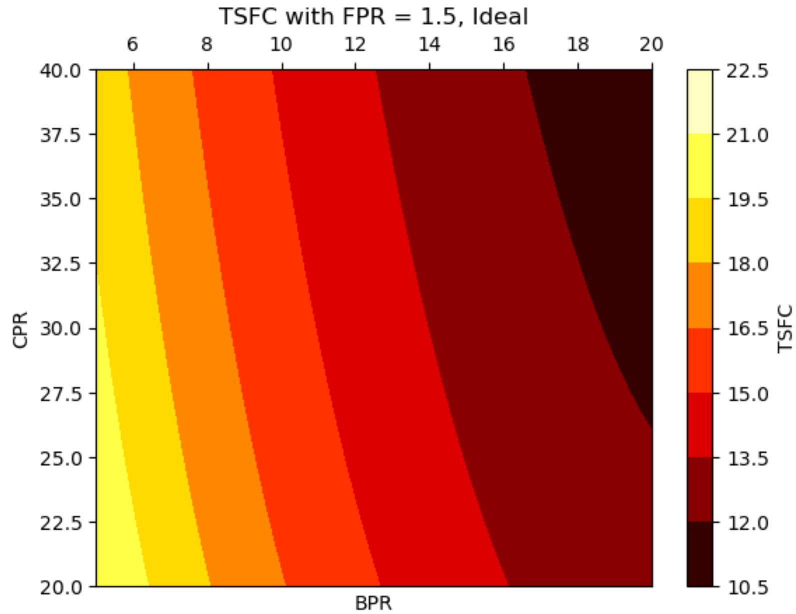
```python
# Isentropic
row = 0
col = 0
for i in range(len(CPRsweep)):
    for j in range(len(BPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[j],pi_f=1.5,pi_c=CPRsweep[i],isentropic='T')
        FmdotFPRConst_I[i][j] = data[2]
        TSFCFPRConst_I[i][j] = data[3]
        fFPRConst_I[i][j] = data[4]
        nT_FPRConst_I[i][j]= data[5]*100
        nP_FPRConst_I[i][j] = data[6]*100
        nO_FPRConst_I[i][j] = data[7]*100
        col += 1
    row += 1

# Real
row = 0
col = 0
for i in range(len(CPRsweep)):
    for j in range(len(BPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[j],pi_f=1.5,pi_c=CPRsweep[i],isentropic='F')
        FmdotFPRConst_R[i][j] = data[2]
        TSFCFPRConst_R[i][j] = data[3]
        fFPRConst_R[i][j] = data[4]
        nT_FPRConst_R[i][j] = data[5]*100
        nP_FPRConst_R[i][j] = data[6]*100
        nO_FPRConst_R[i][j] = data[7]*100
        col += 1
    row+= 1


# Creates the plots

##### F/mdot #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,FmdotFPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,FmdotFPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()






###### TSFC ######
fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,TSFCFPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,TSFCFPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()






##### f  #####
fig1, ax1 = plt.subplots(1,1)
```

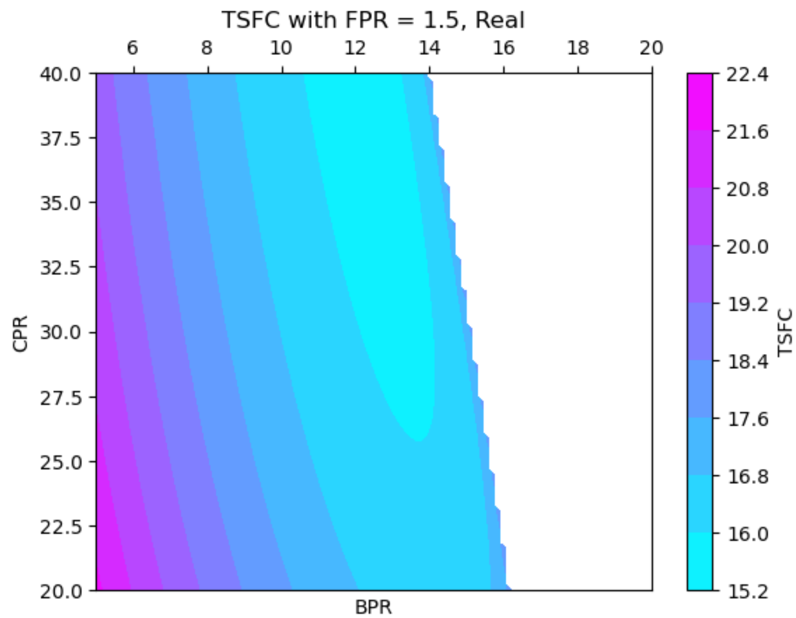
```
plt.contourf(BPRsweep,CPRsweep,fFPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,fFPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()




##### nT #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nT_FPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nT_FPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### nP #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nP_FPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_P$")
ax1.set_title('$\\eta_P$ with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nP_FPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_P}$")
ax1.set_title('$\\eta_P$ with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()




##### nO #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nO_FPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
ax1.set_title('$\\eta_O$ with FPR = 1.5, Ideal')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(BPRsweep,CPRsweep,nO_FPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
```

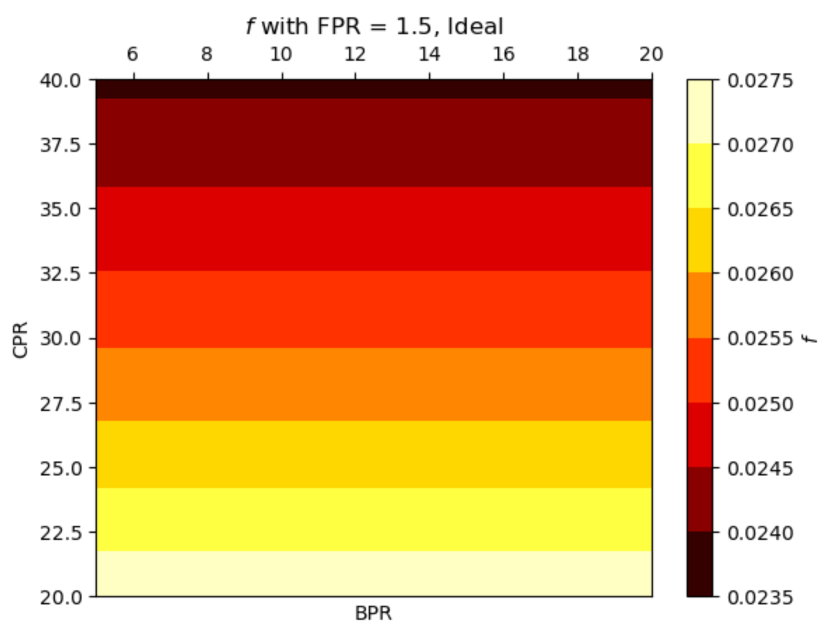
```python
ax1.set_title('$\\eta_O$ with FPR = 1.5, Real')
ax1.set_xlabel('BPR')
ax1.set_ylabel('CPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()
```

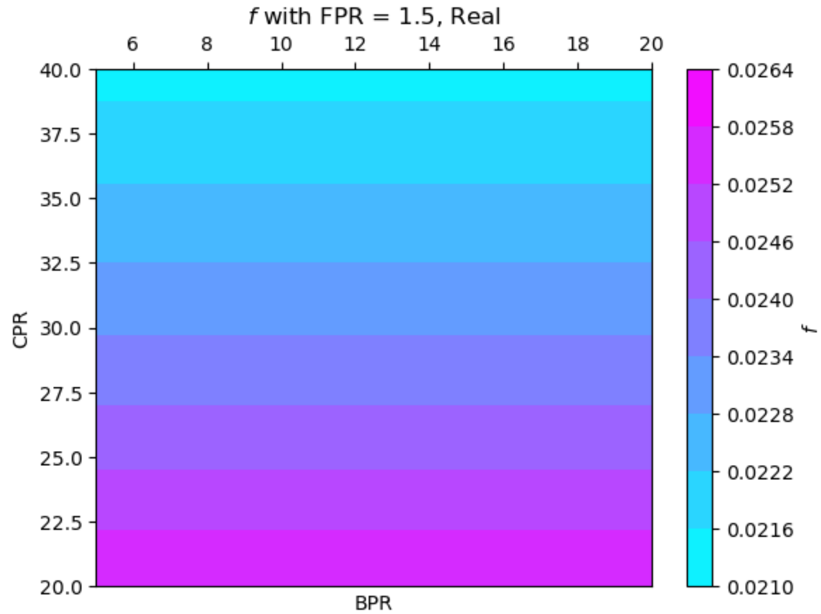


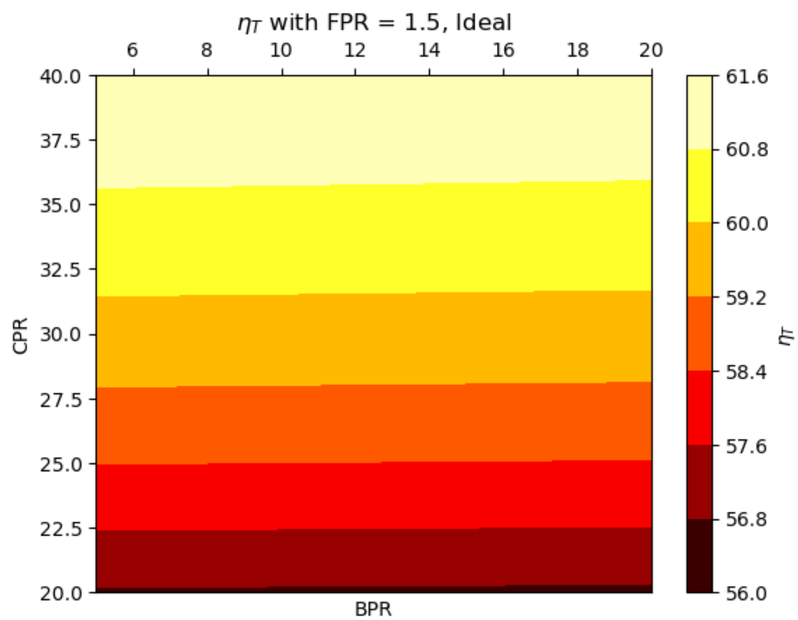


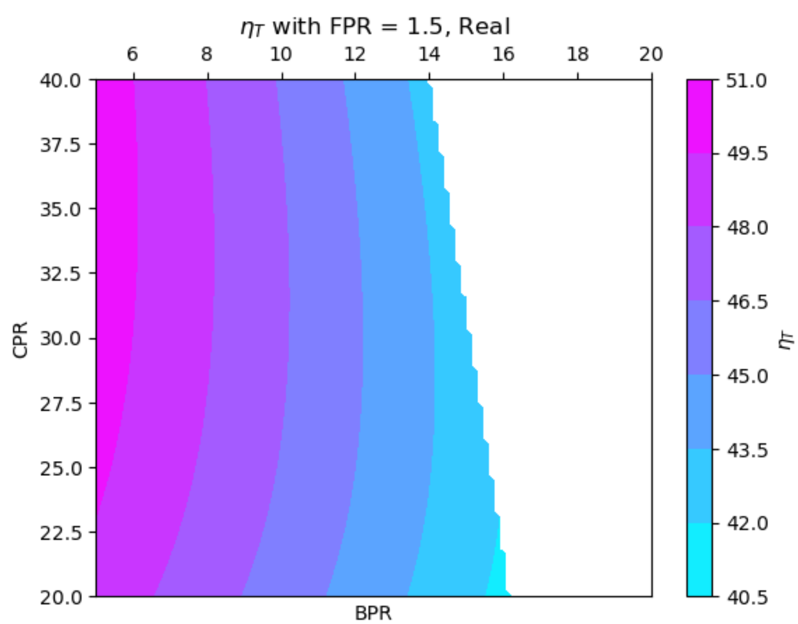

TSFC with FPR = 1.5, Ideal
6
8
10
12
14
16
18
20
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
20.0
CPR
BPR
22.5
21.0
19.5
18.0
16.5
15.0
13.5
12.0
10.5
TSFC

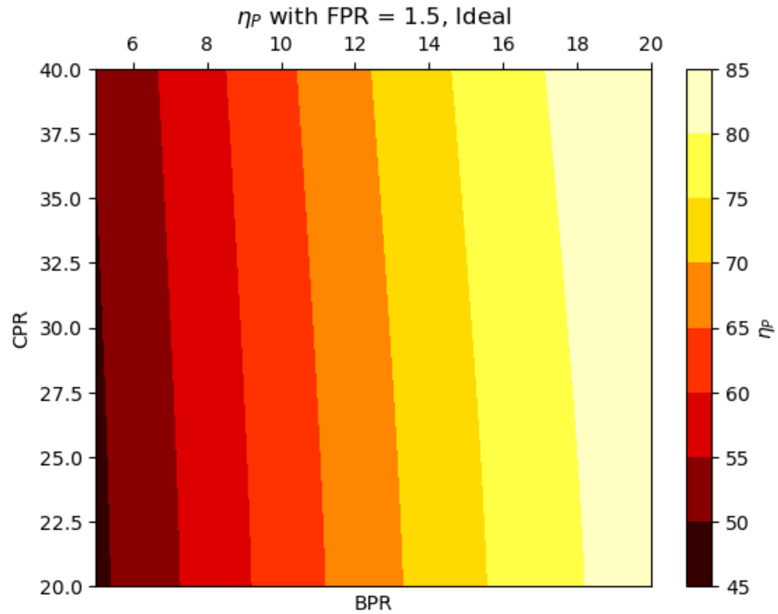

TSFC with FPR = 1.5, Real
6
8
10
12
14
16
18
20
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
20.0
CPR
BPR
22.4
21.6
20.8
20.0
19.2
18.4
17.6
16.8
16.0
15.2
TSFC


f with FPR = 1.5, Ideal
6
8
10
12
14
16
18
20
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
20.0
CPR
BPR
0.0275
0.0270
0.0265
0.0260
0.0255
0.0250
0.0245
0.0240
0.0235
f

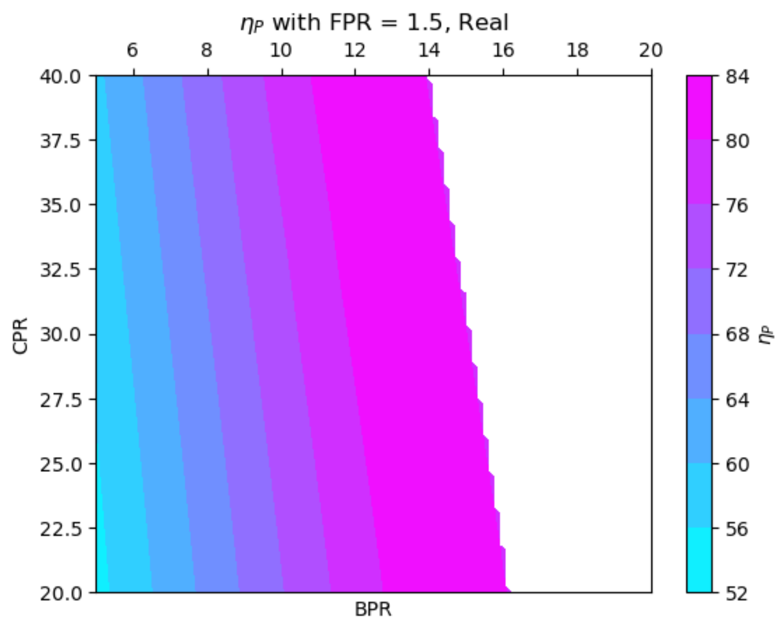

f with FPR = 1.5, Real
6 8 10 12 14 16 18 20
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
BPR
0.0264 0.0258 0.0252 0.0246 0.0240 0.0234 0.0228 0.0222 0.0216 0.0210
f


η_T with FPR = 1.5, Ideal
6 8 10 12 14 16 18 20
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
BPR
61.6 60.8 60.0 59.2 58.4 57.6 56.8 56.0
η_T

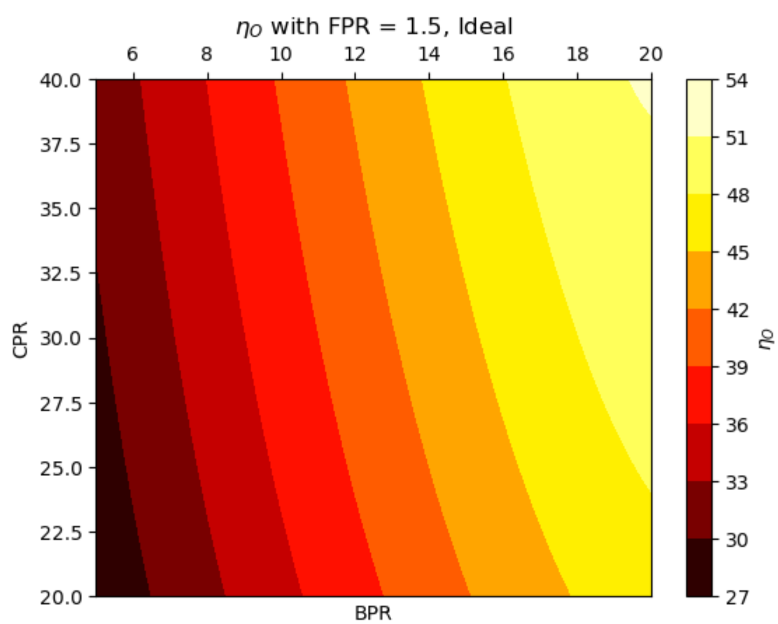

η_T with FPR = 1.5, Real
6 8 10 12 14 16 18 20
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
CPR
BPR
51.0 49.5 48.0 46.5 45.0 43.5 42.0 40.5
η_T



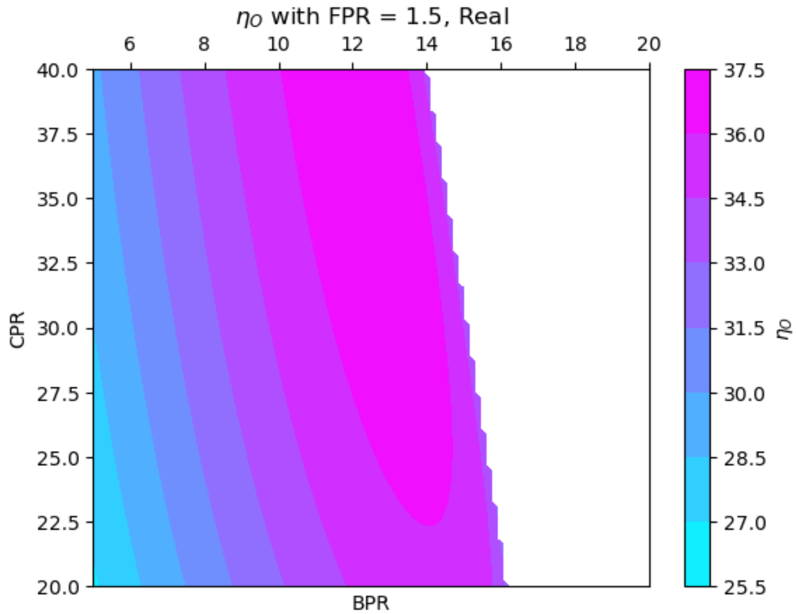
η_P with FPR = 1.5, Ideal
6 8 10 12 14 16 18 20
CPR
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
BPR
85 80 75 70 65 60 55 50 45
η_P



η_P with FPR = 1.5, Real
6 8 10 12 14 16 18 20
CPR
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
BPR
84 80 76 72 68 64 60 56 52
η_P



η_O with FPR = 1.5, Ideal
6 8 10 12 14 16 18 20
CPR
40.0 37.5 35.0 32.5 30.0 27.5 25.0 22.5 20.0
BPR
54 51 48 45 42 39 36 33 30 27
η_O



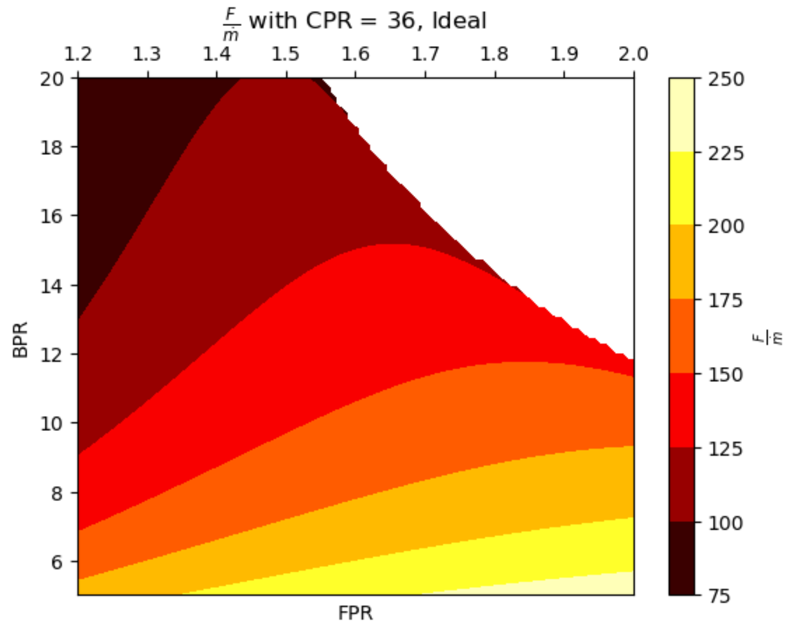
In [ ]:

```
### CPR Combined Study ###

# Holding CPR = 36

BPRsweep = np.linspace(5,20,100)
FPRsweep = np.linspace(1.2,2,100)


# Creates the output arrays
FmdotCPRConst_I = np.zeros((100,100))
TSFCCPRConst_I = np.zeros((100,100))
fCPRConst_I = np.zeros((100,100))
nT_CPRConst_I = np.zeros((100,100))
nP_CPRConst_I = np.zeros((100,100))
nO_CPRConst_I = np.zeros((100,100))

FmdotCPRConst_R = np.zeros((100,100))
TSFCCPRConst_R = np.zeros((100,100))
fCPRConst_R = np.zeros((100,100))
nT_CPRConst_R = np.zeros((100,100))
nP_CPRConst_R = np.zeros((100,100))
nO_CPRConst_R = np.zeros((100,100))



# [mdot, dia, (F/mdot), TSFC, f, thermoEff, propEff, overEff]

# Sweeps through to calculate all the values
# Isentropic
row = 0
col = 0
for i in range(len(BPRsweep)):
    for j in range(len(FPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[i],pi_f=FPRsweep[j],pi_c=36,isentropic='T')
        FmdotCPRConst_I[i][j] = data[2]
        TSFCCPRConst_I[i][j] = data[3]
        fCPRConst_I[i][j] = data[4]
        nT_CPRConst_I[i][j]= data[5]*100
        nP_CPRConst_I[i][j] = data[6]*100
        nO_CPRConst_I[i][j] = data[7]*100
        col += 1
    row += 1

# Real
row = 0
col = 0
for i in range(len(BPRsweep)):
    for j in range(len(FPRsweep)):
        data, tau, temp, M = TF.fullCycleCalc(B=BPRsweep[i],pi_f=FPRsweep[j],pi_c=36,isentropic='F')
        FmdotCPRConst_R[i][j] = data[2]
        TSFCCPRConst_R[i][j] = data[3]
        fCPRConst_R[i][j] = data[4]
        nT_CPRConst_R[i][j] = data[5]*100
        nP_CPRConst_R[i][j] = data[6]*100
        nO_CPRConst_R[i][j] = data[7]*100
        col += 1
```

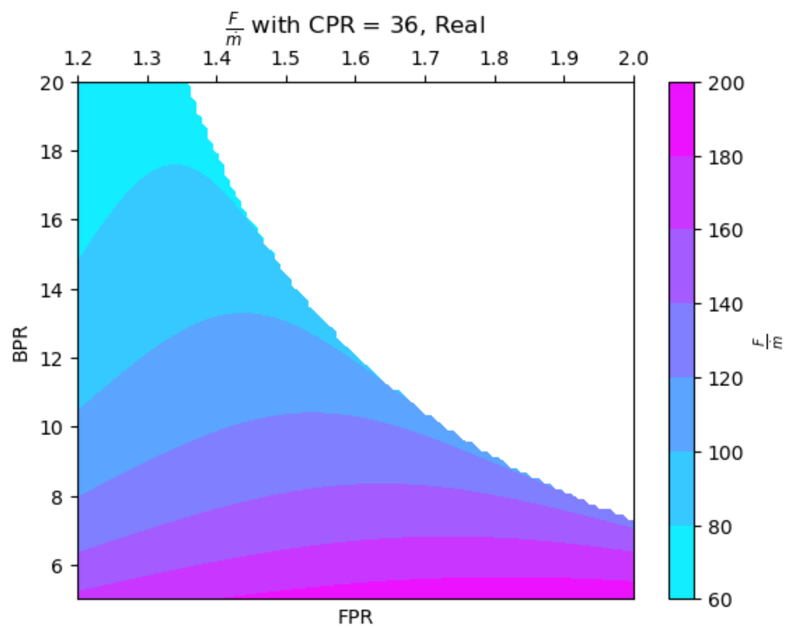
```python
    row+= 1


# Creates the plots

##### F/mdot #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,FmdotCPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,FmdotCPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\frac{F}{\dot{m}}$")
ax1.set_title('$\\frac{F}{\dot{m}}$ with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





###### TSFC ######
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,TSFCCPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,TSFCCPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("TSFC")
ax1.set_title('TSFC with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### f  #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,fCPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,fCPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$f$")
ax1.set_title('$f$ with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()





##### nT #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nT_CPRConst_I,cmap='hot')
cbar = plt.colorbar()
```

```python
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nT_CPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_T$")
ax1.set_title('$\\eta_T$ with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()






##### nP #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nP_CPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_P$")
ax1.set_title('$\\eta_P$ with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nP_CPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_P$")
ax1.set_title('$\\eta_P$ with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()




##### nO #####
fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nO_CPRConst_I,cmap='hot')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
ax1.set_title('$\\eta_O$ with CPR = 36, Ideal')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()

fig1, ax1 = plt.subplots(1,1)
plt.contourf(FPRsweep,BPRsweep,nO_CPRConst_R,cmap='cool')
cbar = plt.colorbar()
cbar.set_label("$\\eta_O$")
ax1.set_title('$\\eta_O$ with CPR = 36, Real')
ax1.set_xlabel('FPR')
ax1.set_ylabel('BPR')
ax1.xaxis.tick_top()
# plt.gca().invert_yaxis()
```

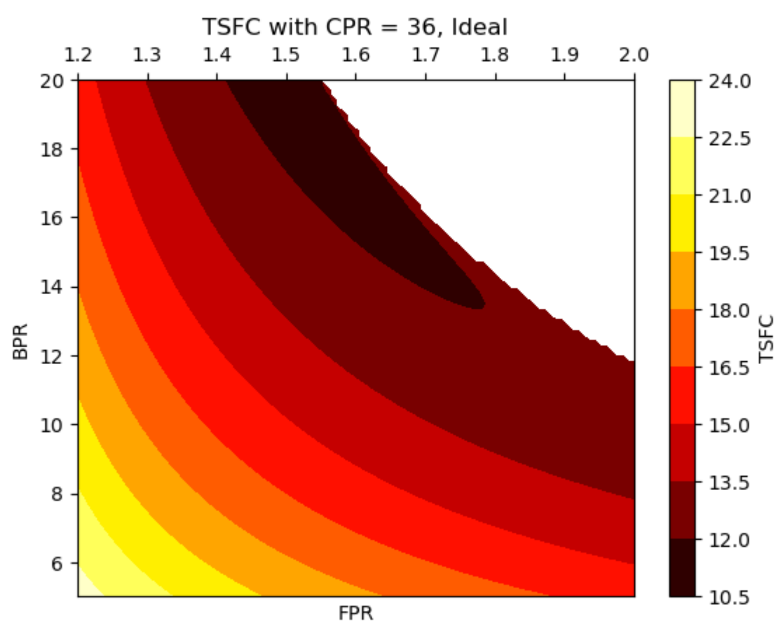
```
c:\Users\Delli\Documents\GitHub\AirbreathingProject1\Project1BackendCalcs.py:182: RuntimeWarning: invalid value encountered in sca
lar power
  self.P_05 = self.P_04_5/(self.T_04_5/self.T_05)**(y/(nt*(y-1)))
```



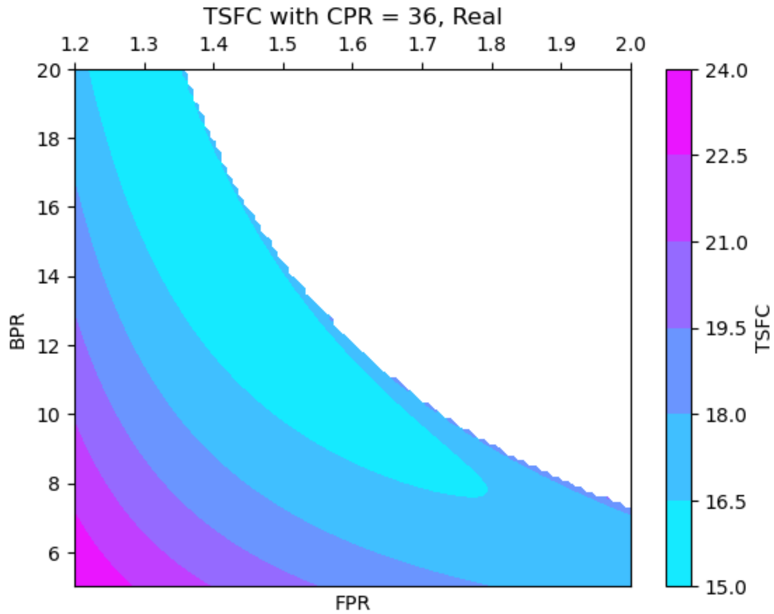
$\frac{F}{\dot{m}}$ with CPR = 36, Ideal
BPR
FPR
$\frac{F}{\dot{m}}$

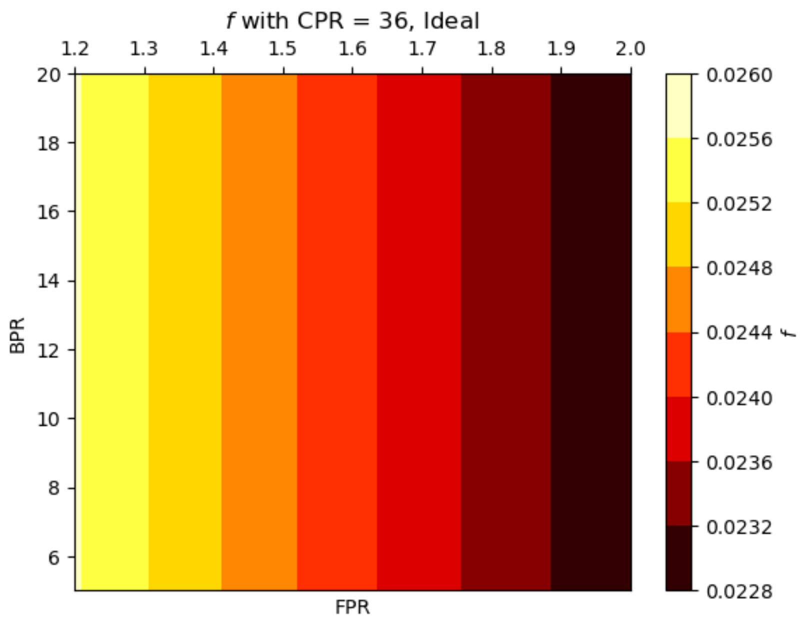


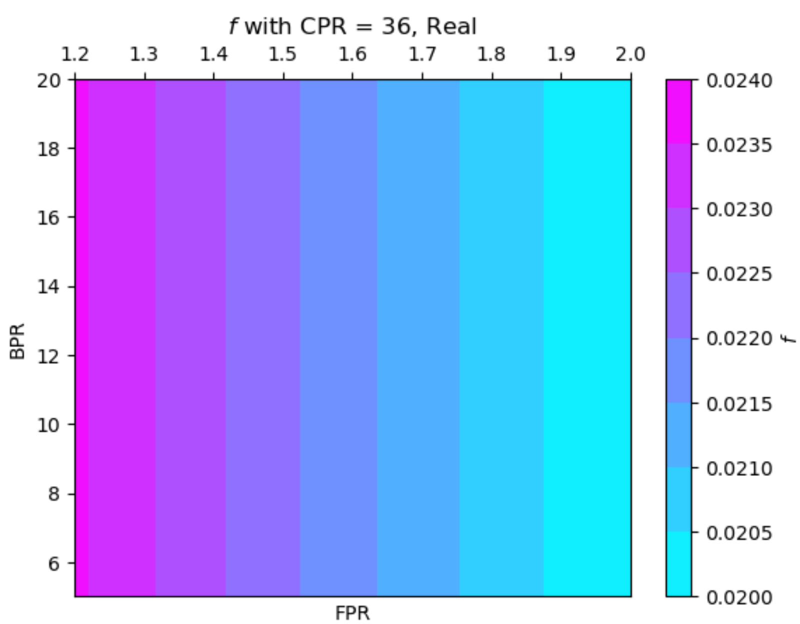
$\frac{F}{\dot{m}}$ with CPR = 36, Real
BPR
FPR
$\frac{F}{\dot{m}}$

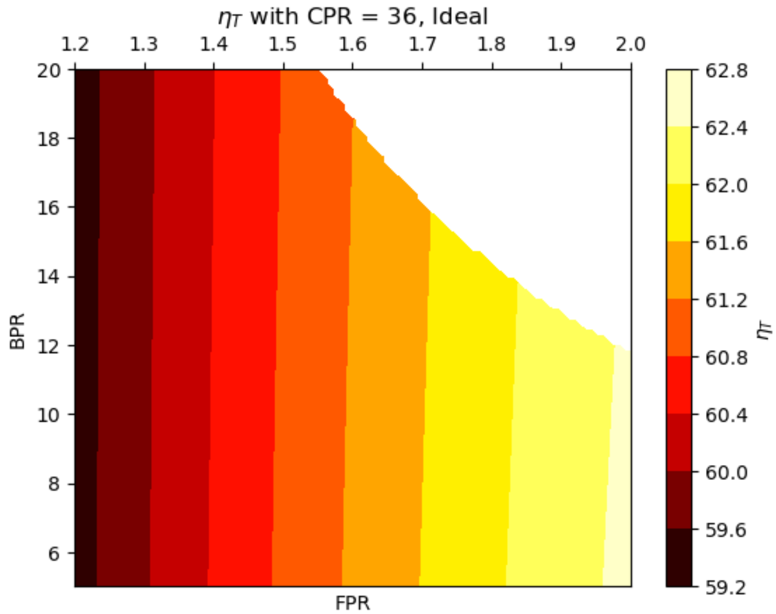


TSFC with CPR = 36, Ideal
BPR
FPR
TSFC

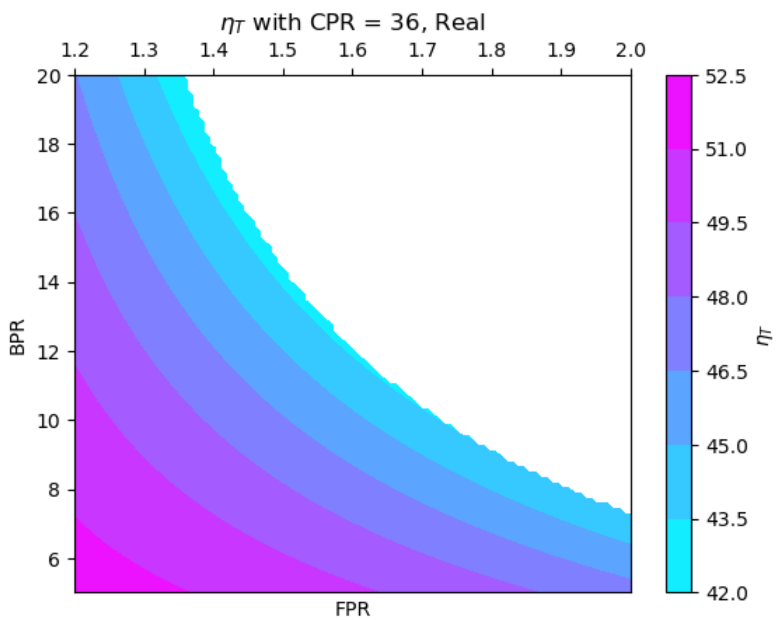

TSFC with CPR = 36, Real
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
20
18
16
14
12
10
8
6
BPR
FPR
24.0
22.5
21.0
19.5
18.0
16.5
15.0
TSFC


f with CPR = 36, Ideal
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
20
18
16
14
12
10
8
6
BPR
FPR
0.0260
0.0256
0.0252
0.0248
0.0244
0.0240
0.0236
0.0232
0.0228
f

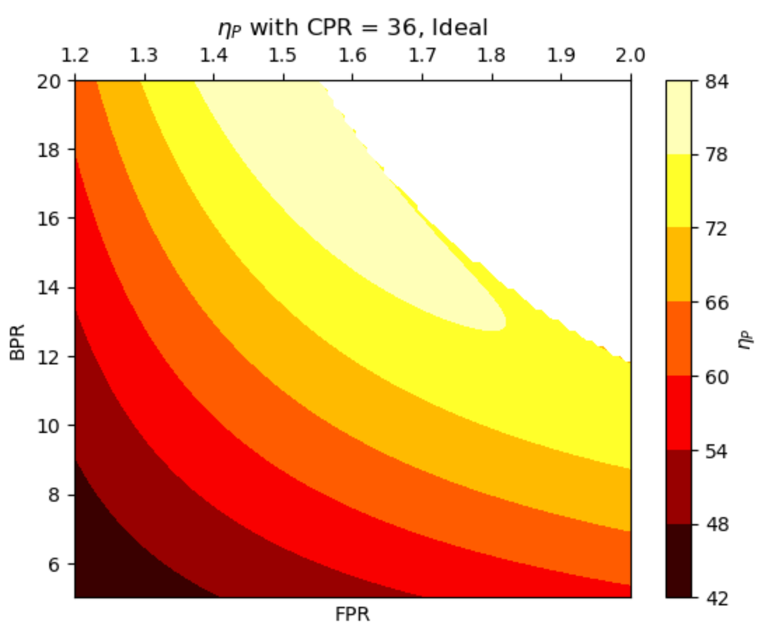

f with CPR = 36, Real
1.2
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
20
18
16
14
12
10
8
6
BPR
FPR
0.0240
0.0235
0.0230
0.0225
0.0220
0.0215
0.0210
0.0205
0.0200
f



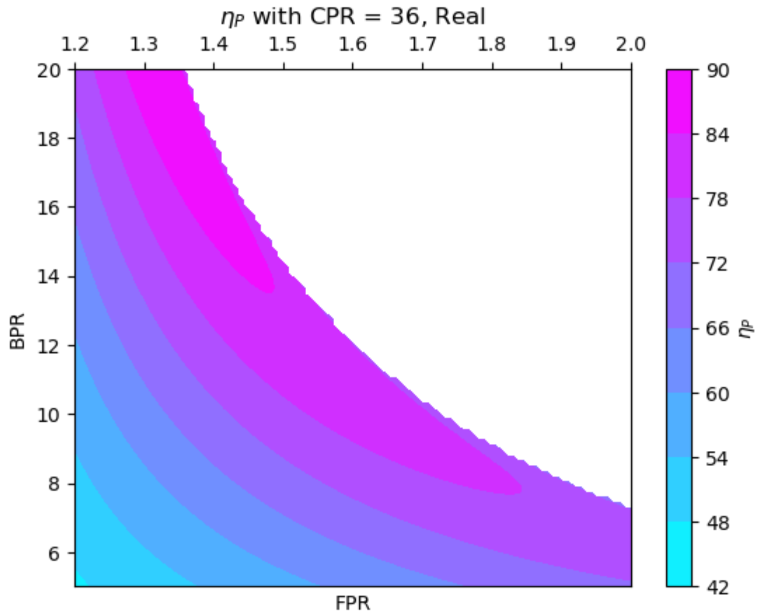
η_T with CPR = 36, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
BPR
20 18 16 14 12 10 8 6
FPR
62.8 62.4 62.0 61.6 61.2 60.8 60.4 60.0 59.6 59.2
η_T

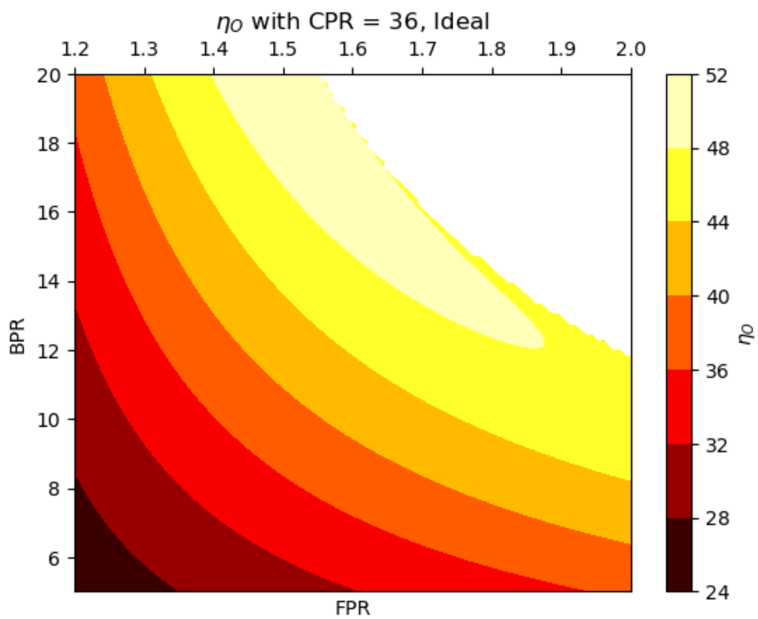


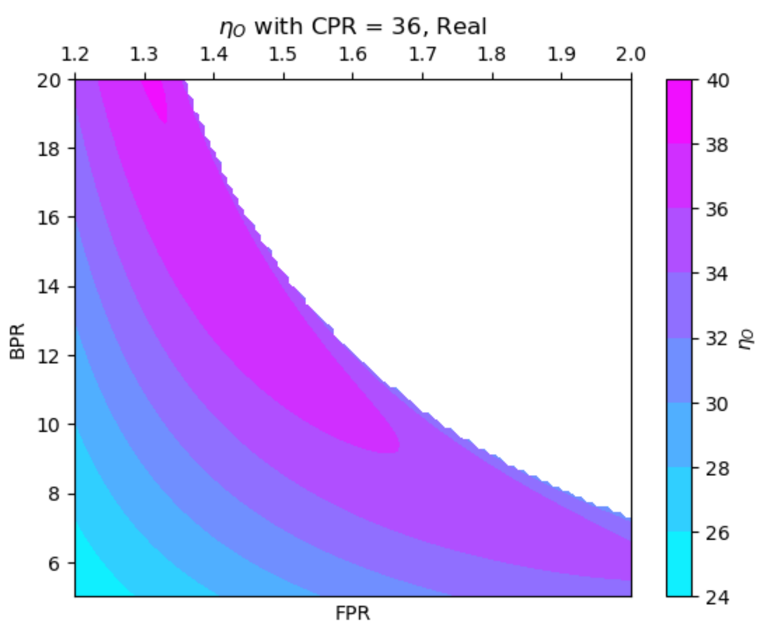
η_T with CPR = 36, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
BPR
20 18 16 14 12 10 8 6
FPR
52.5 51.0 49.5 48.0 46.5 45.0 43.5 42.0
η_T



η_P with CPR = 36, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
BPR
20 18 16 14 12 10 8 6
FPR
84 78 72 66 60 54 48 42
η_P


η_P with CPR = 36, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
20 18 16 14 12 10 8 6
BPR
FPR
90 84 78 72 66 60 54 48 42
η_P


η_O with CPR = 36, Ideal
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
20 18 16 14 12 10 8 6
BPR
FPR
52 48 44 40 36 32 28 24
η_O


η_O with CPR = 36, Real
1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
20 18 16 14 12 10 8 6
BPR
FPR
40 38 36 34 32 30 28 26 24
η_O