# APPENDIX

**APPENDIX A.**

**Python Program to Compute Pressure Drop of a Multiphase Flow in a Vertical Tubing**

```
def mass(WOR,SGo,SGw):     a =SGo*350*(1/(1+WOR))     b= SGw*350*(WOR/(1+WOR))
m = a +b     return m
def den_liq(WOR,SGo,SGw):     a = SGo*62.4*(1/(1+WOR))  b= SGw*62.4*(WOR/(1+WOR))
 m = a + b
 return m
 def avg_p(P1,P2):
 p = (P1 + P2)/2 + 14.7      return p
def avg_t(T1,T2):     T = ((T1) + (T2))/2
return T
def         Z(P1,P2,T1,T2,SGg):
P=avg_p(P1,P2)
T=avg_t(T1,T2)+460     from math import
log,inf,exp
Ppc = 677 + 15.0*SGg + 37.5*SGg**2
Tpc = 168 + 325*SGg + 12.5*SGg**2
Ppr = P/Ppc
Tpr = T/Tpc
A=1.39*(Tpr-0.92)**0.5 - 0.36*Tpr - 0.101
B=(0.62-0.23*Tpr)*Ppr + ((0.066/(Tpr-0.86))-0.037)*Ppr**2 + (0.32/(10**9*(Tpr-1)))*Ppr**6
C=0.132 - 0.32*(log(Tpr))
K=0.3106-0.9*Tpr + 0.1824*Tpr**2
D=10**K
z=A + (1-A)*exp(-B) + C*Ppr**D
P=avg_p(P1,P2)     T=avg_t(T1,T2)+460     from math import exp,inf
Ppc = 677 + 15.0*SGg + 37.5*SGg**2
Tpc = 168 + 325*SGg + 12.5*SGg**2
Ppr = P/Ppc
Tpr = T/Tpc
 t = 1/(Tpr)
 #print(Ppc,Tpc,Tpr,Ppr)
X1 = (-0.06125*Ppr*t)*exp(-1.2*(1-t)**2)
X2 = (14.76*t) - (9.67*(t**2)) - (4.58*(t**3))
X3 = 90.7*t - 242.2*(t**2)- 42.4*(t**3)
X4 = 2.18 + 2.82*t
av=inf     y = Ppr * t * exp(-1.2*(1-t)**2)
while abs(av)>0.00000001:
av = X1 + (y+y**2+y**3+y**4)/(1-y)**3 - X2*y**2 + X3*y**X4
der= ((1 + 4*y + 4*y**2 + 4*y**3 + y**4)/(1-y)**4) - (2*X2*y) + (X3*X4*y**(X4-1))
y = y - av/der
Z = ((0.06125*Ppr*t)/y)*(exp(-1.2*(1-t)**2))
return z
def avg_den_gas(SGg,P1,P2,T1,T2):
d = SGg*0.0764*(avg_p(P1,P2)/14.7)*(520/(avg_t(T1,T2)+460))*(1/Z(P1,P2,T1,T2,SGg))          return d
def avg_den_gas(SGg,P1,P2,T1,T2):
d = SGg*0.0764*(avg_p(P1,P2)/14.7)*(520/(avg_t(T1,T2)+460))*(1/Z(P1,P2,T1,T2,SGg))
return d
def water_vis(T1,T2):   from math import exp
vis = exp(1.003 - 1.479*10**- 2*avg_t(T1,T2) +     1.982*10**-5 * (avg_t(T1,T2)+460)**2)
return vis
def liq_vis(T1,T2,API,WOR):
vis = oil_vis(T1,T2,API)*(1/(1+WOR)) + water_vis(T1,T2)*(WOR/(1+WOR))
return vis
def oil_st():
P = avg_p(P1,P2)
T = avg_t(T1,T2)+460
```

```
C=1-0.024*P**0.45
if P>5000:     st = 0   elif
   P>3997:    st=1  elif
   T<=68: st=(39-0.2571*API)*C  elif
    T<=100:   st = -1.5*C*(T-68)/32 + (39-0.2571*API)*C
    else:
    st = (37.5-0.2571*API)*C              return st
    def water_st():
    T = avg_t(T1,T2)+460
    P = avg_p(P1,P2)
    if T<=74:  st = 75 - 1.108*P**0.349  elif
    T>280:  st = 53 - 0.1048*P**0.037
    else:
    st = ((53 - 0.1048*P**0.037)-(75 - 1.108*P**0.349))*(T-74)/206 + (75 - 1.108*P**0.349)
    return st
    def liq_st(WOR):
     o=oil_st()     w = water_st()
    st = o*(1+(1+WOR)) + w*(WOR+(1+WOR))
    return st
    def RS(SGg,API,T1,T2,P1,P2):
     x = 0.0125*API - 0.0009*(avg_t(T1,T2)+460)
    RS = SGg * ((avg_p(P1,P2)/18.2 + 1.4)*10**x)**1.2048              return RS
     def Bo(SGg,API,T1,T2,P1,P2,SGo):
     Bo   =   0.9759   +0.000120*((RS(SGg,API,T1,T2,P1,P2)*(SGg/SGo)**0.5)+1.25*(avg_t(T1,T2)+460))**1.2
return Bo
     def area(D):
     from math import pi
     area = pi*D**2/4
      return area
     def liq_vis_no(T1,T2,API,WOR,SGo,SGw):
     Nl = 0.15726*liq_vis(T1,T2,API,WOR)*(1/(den_liq(WOR,SGo,SGw)**3))**0.25
     return Nl
      def VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1):   a=5.61*(Qo+Qw)/(86400*area(D))
       b=Bo(SGg,API,T1,T2,P1,P2,SGo)*(1/(1+WOR))     c= Bw*(WOR/(1+WOR))
      vsl = a*(b+c)
 return vsl
def LVN(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,SGw,Bw=1):
a=(VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1))
#print(type(a))      b=den_liq(WOR,SGo,SGw)     c=liq_st(WOR)
#print(type(b))     #print(type(c))
lvn =abs(1.938*a*(b/c)**0.25)
return lvn
def                                                LVN(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,SGw,Bw=1):
a=(VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1))
 #print(type(a))
 b=den_liq(WOR,SGo,SGw)     c=liq_st(WOR)
#print(type(b))
#print(type(c))
lvn = abs(1.938*a*(b/c)**0.25)
return lvn
def NGV(Qo,Qw,GLR,SGo,SGg,API,T1,T2,P1,P2,WOR):
Ngv       =       1.98*VSG(Qo,Qw,GLR,SGg,API,T1,T2,P1,P2,WOR)*abs((den_liq(WOR,SGo,SGw)/liq_vis(T
1,T2,API,WOR)))**0.25
  return Ngv
 def Nd(D,WOR,SGo,SGW,T1,T2,API):
 Nd = 120.872*D*((den_liq(WOR,SGo,SGw)/liq_vis(T1,T2,API,WOR))**0.5)
 return Nd
 def L1L2(D,WOR,SGo,SGw,T1,T2,API):
 if Nd(D,WOR,SGo,SGw,T1,T2,API) < 40:
 L1 = 2
```

```
L2 = 0.1*Nd(D,WOR,SGo,SGw,T1,T2,API) + 0.25 elif
 Nd(D,WOR,SGo,SGw,T1,T2,API) > 40 and Nd(D,WOR,SGo,SGw,T1,T2,API)<70:
L2=1
L1=(19/6)-(1/30*Nd(D,WOR,SGo,SGw,T1,T2,API))
else:
L2=1.1
L1=0.9
return [L1,L2]
def Regime(D,WOR,SGo,SGw,T1,T2,API):
L1= L1L2(D,WOR,SGo,SGw,T1,T2,API)[0]
 L2=L1L2(D,WOR,SGo,SGw,T1,T2,API)[1]          ngv=NGV(Qo,Qw,GLR,SGo,SGg,API,T1,T2,P1,P2,WOR)
nlv=LVN(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,SGw,Bw=1)
try: if ngv<(L1 + L2*nlv):   a = 'Bubble'
elif ngv<(L1 + L2*nlv) and ngv<(50+36*nlv):   a = 'Slug'   else: a='Mist'
except:    a='Mist'
return a
def holdup():
 a = Regime(D,WOR,SGo,SGw,T1,T2,API)          dg=(avg_p(P1,P2)*28.96*SGg)/(10.73*(avg_t(T1,T2)+460))
dl=den_liq(WOR,SGo,SGw)     vsg=VSG(Qo,Qw,GLR,SGg,API,T1,T2,P1,P2,WOR)
Vm = VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1) + vsg
from math import exp if a== Bubble':
vbs=1.41*(oil_st()*(dl-dg))**0.25
Vbf = 1.2*Vm + vbs
Hl = 1 - (vsg/Vbf)
elif a == 'Slug':
ne = 32.4*D**2*(dl-dg)/liq_st()
nv = (D**3 * dl * (dl - dg)/liq_vis(T1,T2,API,WOR))**0.5
if nv>=250:    m=10       elif nv<=18:   m=25   else:     m=69*nv**-0.35
C = 0.345*(1-exp(-0.029*nv))*(1-exp((3.37-ne)/m))
vbs = C*(32.4*D*(dl-dg)/dl)**0.5
Vbf = 1.2*Vm + vbs
Hl = 1 - (vsg/Vbf)
se:  Hl  = 1/ (1+vsg/VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1))
return Hl
def Gm():
Ql=Qo+Qw
dg=(avg_p(P1,P2)*28.96*SGg)/(10.73*avg_t(T1,T2))         gl=den_liq(WOR,SGo,SGw)*Ql/area(D)         gg  =
dg*Qg/area(D)
gm = gl+gg     return gm
def NRE():  nre = Gm()*D/liq_vis(T1,T2,API,WOR)
return nre
def Slip(e=0):  from math import
log,inf,exp Ql=Qo+Qw
L = l/(Ql+Qg)       y = /holdup()**2     if y>1.2:         s=log(2.2*y - 2)
else:
s =log(y)/(-0.0523+3.182*log(y)-0.8725*log(y)**2 + 0.01853*log(y)**4)
fc=inf     fest=0.001     while abs(fcfest)>0.0001:
 fc=(1.74-2*log(2*(e/D) + (18.7/(NRE()*fest**0.5))))**-2  fest =(fc+fest)/2
ftp = exp(s)*fc
return ftp
def press_drop():
dg=(avg_p(P1,P2)*28.96*SGg)/(10.73*(460+avg_t(T1,T2)))                         dl=den_liq(WOR,SGo,SGw)
vsg=VSG(Qo,Qw,GLR,SGg,API,T1,T2,P1,P2,WOR)
vsl=VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1)
hl=holdup()     ftp = Slip(e=0)
Vm = VSL(Qo,Qw,WOR,D,SGg,SGo,T1,T2,P1,P2,Bw=1) + vsg
G=dl*vsl + dg*vsg
dp=(dl*hl + (1- hl)*dg + (ftp*G*Vm)/(2*32.2*D))/144
return dp
```

**APPENDIX B**
**Additional Correlations Used**

a) Gas solubility, $R_s$ (scf/stb) ………………………[Standing(1981)]

$$R_s = \gamma_g [\left(\frac{P}{18.2} + 1.4\right) 10^X]^{1.2048}$$

$$x = 0.0125API - 0.0009T$$

Where:
$\gamma_g$= Gas specific gravity
$API = gravity\ of\ stock\ tank\ oil$ ($^o$API)

b) Oil Formation Volume Factor (FVF), $B_o$ (bbl/stb) ……..[Standing (1981)]

$$B_o = 0.9759 + 0.00012[R_s \left(\frac{\gamma_g}{\gamma_o}\right)^{0.5} + 1.25T]^{1.2}$$

Where:
$\gamma_g$= Oil specific gravity

c) Water Formation Volume Factor (FVF), $B_W$ (bbl/stb)

$B_W = A_1 + A_1 P + A_3 P^2$
Wheere the coefficients $A_1$ to $A_3$ are given by the following expression
$A_1$=a1 +a2T + a3T$^2$

**Table 2:** For Gas- Free Water.

| A | a1 | a2 | a2 |
|---|---|---|---|
| A1 | $9.947*10^{-1}$ | $5.8*10^{-6}$ | $1.02*10^{-6}$ |
| A2 | -48.2 | $1.8376*10^{-8}$ | -78.7 |
| A3 | $1.3*10^{-10}$ | -25.855 | $4.285*10^{-15}$ |

d) Dead oil viscosity, $\mu_o$ (cp) ………………………[Beggs-Robinson(1975)]

$\mu_o = 10^x - 1$
x = Y * T-1.163
Y = $10^Z$
Z = 3.0324 – 0.02023 *API

e) Water viscosity, $\mu_W$ (cp) ……………………[Beggs and Brills (1978)]

$\mu W = e(1.003-1.479*10^{-2}T+1.982*10^{-5}T^2$

f) Gas viscosity, $\mu_g$ (cp) ……………….[The Lee-Gonzalez-Eakin Method]

$\rho g/62.4)Y]$ $[x($
$\mu g = 10-4Ke$

$$K = \frac{(9.4 + 0.02M_a)(T + 460)^{1.5}}{209 + 19M_a + (T + 460)}$$

$$x = 3.5 + \frac{986}{T + 460} + 0.01M_a$$

$$Y = 2.4 - 0.2x$$

$$\rho_g = \frac{2.703\gamma_g P}{z_g(T + 460)}$$

$Where$:

$M_a$ = Apparent Molecular weight of the gas mixture
$z_g$= Gas compressibility factor

g) Gas compressibility factor, z …………………. [The Hall-Yarborough Method]

$$z = \frac{0.06125P_{Pr} * t}{y} e^{[-1.2(1-t)^2]}$$

$P_{Pr}$ = P/$P_{Pc}$ $T_{Pr}$= T/$T_{PC}$

$$P_{Pc} = 677 + 15.0\gamma_g + 37.5{\gamma_g}^2$$
$$T_{Pc} = 168 + 325\gamma_g + 12.5{\gamma_g}^2$$

Where:
$P_{Pr}$=Pseudo-reduced pressure
$P_{PC}$ = Pseudo-critical Pressure
$T_{Pr}$ = Pseudo-reduced Temperature
$T_{Pc}$ = Pseudo-critical temperature
$t$ = Reciprocal of the pseudo-reduced temperature
$y$= The reduced density obtained from the solution of the following equation

$$X1 + \frac{y+y^2+y^3+y^4}{(1-y)^3} - X2y^2 + X3y^{(X4)} = 0 \quad \ldots \quad \ldots\ldots\ldots\ldots\ldots (1)$$

$X1 = -0.06125P_{Pr}te[-1.2(1-t)2]$
$X2 = 14.76t - 9.76t^2 - 4.58t^3$
$X3 = 90.7t - 242.2t^2 - 42.4t^3$;
$X4 = 2.18 + 2.82t$

This equation is usually solved using Newton-Raphson iteration Technique. This technique involves the following steps;

Step 1: Make an initial guess of the unknown parameters $y^k$ where k is an iteration counter. An appropriate guess of y is given by

$y_k = 0.0125P_{Pr}t\ e[-1.2((1-t)2]$

Step 2: Substitute the initial values= in equation (1), [f(y)]

Step 3 : If f(y)=0, a new improved estimate of y($y^{k+1}$) is calculated from

$y^{k+1}$ = $y^k$- f(y)/f'($y^k$)

f'($y^k$) $= \frac{1+4y+4y^2+4y^3+y^4}{(1-y)^4} - 2(x2)y + (X3)(X4)\ y^{(X4-1)}$)

Step 4: Continue iteration until y converges.