In [1]:
```
import numpy as np
import math
import matplotlib
import matplotlib.pyplot as plt
import scipy
from scipy import stats, optimize, interpolate
from scipy.integrate import odeint
from scipy.integrate import solve_bvp
plt.rcParams.update({'font.size': 18})
```

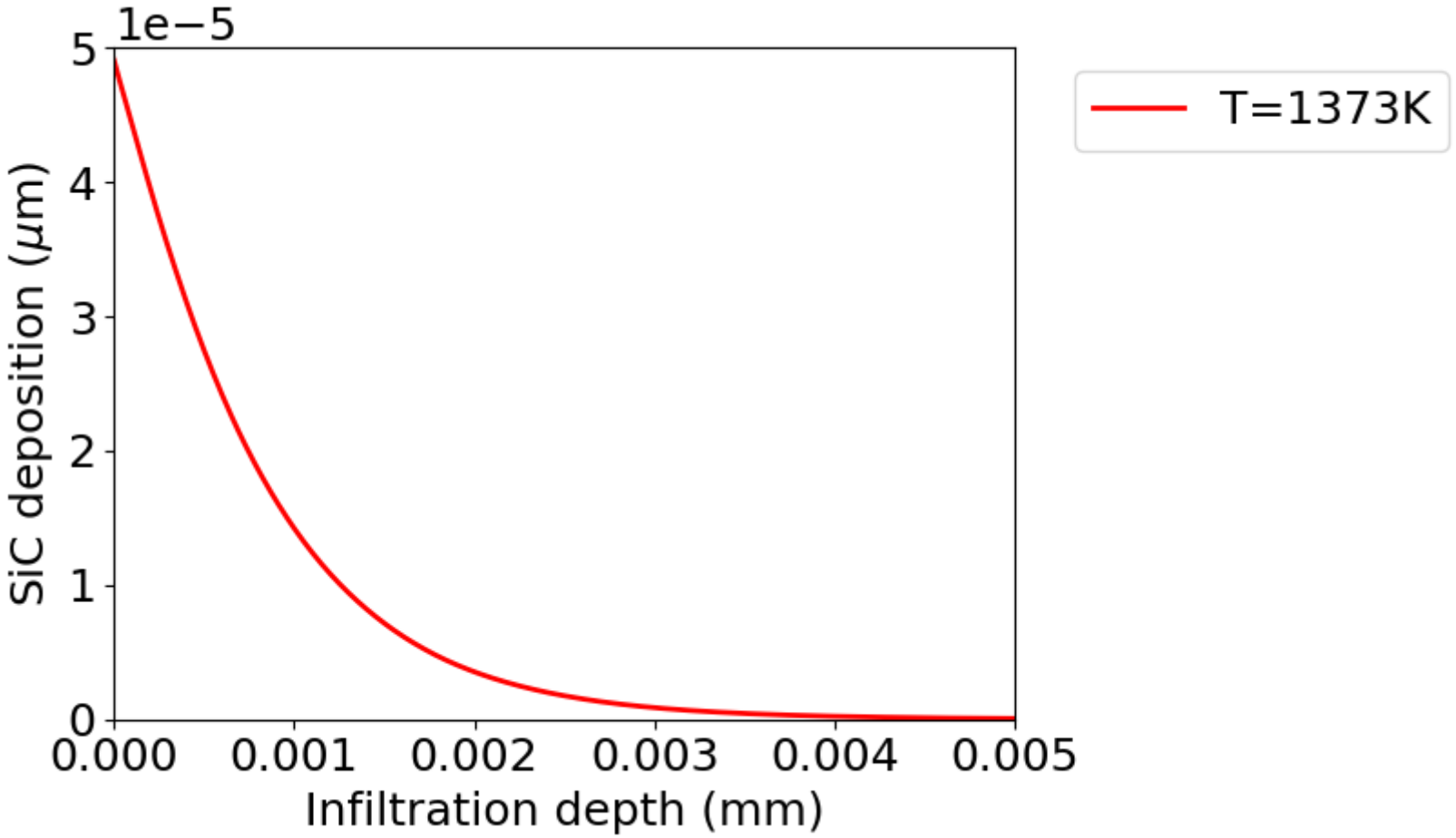
In [2]:
```
def cvi_sim(Tin,Pin, phi0in, rin, ain,clr,pllbl,dt,va):

    plotlabel = str(pllbl)
    clr = str(clr)
    phi0 = phi0in
    L = rin*phi0
    nz = 1000
    h = L/(nz+1)
    rp0 = phi0/2
    dt = dt
    Zrange = np.linspace(0,L,nz)
    alpha = ain
    P = Pin # Pa
    T = Tin # K
    R = 8.314 # J/mol K
    CMTS0 = (1/(1+alpha))*(P / (R*T))
    CH20 = (alpha/(1+alpha))*(P / (R*T))
    dCMTS0dz = dCH20dz = 0
    k0 = 3.89e9
    Ea = 296e3
    k_het = k0*np.exp(-Ea/(R*T))
    dKmts = 3.97 #m/sK^(1/2)
    dKh2 = 34.30 #m/sK^(1/2)
    dKhcl = 8.03 #m/sK^(1/2)
    DKmts=dKmts*phi0*(T**0.5) # m^2/s
    DKh2=dKh2*phi0*(T**0.5) # m^2/s
    DKhcl=dKhcl*phi0*(T**0.5) # m^2/s
    Msic = 0.04011 #kg/mol
    Mmts=0.149 #kg/mol
    Mh2 = 2e-3 #kg/mol
    Mhcl = 3.65e-2 #kg/mol
```

```
rhosic = 3217 #kg/m^3
# rhomts = 1270
# rhoh2 = 0.08375
# rhohcl = 1048
Vsic=Msic/rhosic #m^3/mol
Vmts=1.227e-4 #m^3/mol
Vh2=1.43e-5 #m^3/mol
Vhcl= 25.3e-5 #m^3/mol
DFmtsh2=1.360e-3*((T**(3/2))/(P*(Vh2**(1/3)+Vmts**(1/3))**2))*(1/Mh2 + 1/Mmts)**0.5 #Gilliland formula Fe
DK = DKmts
DF = DFmtsh2
D = (1/DF + 1/DK)**(-1)


print('D=',D,'k_het =',k_het,'T=',T,'P=',P, 'CMTS0=',CMTS0)

rpi = rp0*np.ones(nz)
rp = rpi
drpdzi = np.zeros(nz)
drpdz = drpdzi

# zi = z*nz/L
# print(zi)

Ct = []
rpt = []

n = 0
# for i in range(2):
detA = 1
singular_matrix_flag = 0
percent_closed = 0
while (any(rp<rp0/100) is not True) and (percent_closed < 99):# and (detA != 0):
    A3z = -(rp**2)/h - rp*drpdz
    A2z = 2*((rp**2)/h + rp*(k_het*h)/D)
    A1z = -(rp**2)/h + rp*drpdz

    # print(A3z.shape, A2z.shape, A1z.shape)

    rows, cols = (nz, nz)
    arr = [[0 for i in range(cols)] for j in range(rows)]
    A = np.array(arr)
    precision = 7
```

```
        for i in range(rows):
            for j in range(cols):
                if j==i:
                    A1zi = A1z[i]*(10**precision)
                    A2zi = A2z[i]*(10**precision)
                    A3zi = A3z[i]*(10**precision)
                    A[i][j] = A2zi
                    if not i==0:
                        # print(A1zi)
                        A[i][j-1] = A1zi
                    if not j==cols-1:
                        A[i][j+1] = A3zi
                    else:
                        A[i][j] = A2zi + A3zi #boundary condition: C(n+1)=C(n) ==> A3(z_n)C(n+1) + A2(z_n)C_n
        A = A*(10**(-1*precision))
        # print(A.shape)
        f = np.zeros(nz)
        f[0] = -A1z[0]*CMTS0 #boundary condition: A3(z_1)C(2) + A2(z_1)C(1) + A1(z_1)C(0) = 0 --> A3(z_1)C(2)
        detA = np.linalg.det(A)
        C = np.linalg.solve(A, f)

        nu = k_het*C
        rp = rp*(1-dt*nu*Vsic)
        drpdz[:-1] = (rp[1:] - rp[:-1])/nz
        drpdz[-1] = drpdz[-2]

        n+=1
        if n % 10 == 0:
            Ct.append(C)
            rpt.append(rp)
            percent_closed = 100*(1 -(rp[0]-rp0/100)/rp0)
            print('n=',n,': detA = ',round(detA,10),', pore is', round(percent_closed,2),'% closed')
            #print('n=',n,': at t =',round(n*dt*0.000277778/100,2),'hours, pore is', round(percent_closed,2),

    rpl = len(rpt)
    print(n)
    if va == "rp":
        if plotlabel == "none" or plotlabel == "":
            plt.plot(Zrange,(rp0-rp),linewidth=2,color=clr)
        else:
            plt.plot(Zrange,(rp0-rp),label=plotlabel,linewidth=2,color=clr)
        lines = str(rp0-rp)
        with open('rp_'+'T'+str(T)+'_P'+str(P)+'_phi0'+str(round(phi0*1e6,2))+'_ar'+str(round(L/phi0,2))+'.tx
```

```
            f.writelines(lines)
        plt.xlabel(r'Infiltration depth (mm)')
        plt.ylabel(r'SiC deposition ($\mu$m)')
        plt.xlim([0,L/2])
        plt.ylim([0,rp0])
        plt.legend(bbox_to_anchor=(1.04, 1), loc="upper left")
        plt.savefig('rp_'+'T'+str(T)+'_P'+str(P)+'_phi0'+str(round(phi0*1e6,2))+'_ar'+str(round(L/phi0,2))+'.

    if va == "C":
        if plotlabel == "none" or plotlabel == "":
            plt.plot(Zrange,C,linewidth=2,color=clr)
        else:
            plt.plot(Zrange,C,label=plotlabel,linewidth=2,color=clr)
        lines = str(C)
        with open('C_'+'T'+str(T)+'_P'+str(P)+'_phi0'+str(round(phi0*1e6,2))+'_ar'+str(round(L/phi0,2))+'.txt
            f.writelines(lines)
        plt.xlabel(r'Infiltration depth (mm)')
        plt.ylabel(r'Gas concentration (mol/m$^3$)')
        plt.xlim([0,L/2])
        plt.legend(bbox_to_anchor=(1.04, 1), loc="upper left")
        plt.savefig('C_'+'T'+str(T)+'_P'+str(P)+'_phi0'+str(round(phi0*1e6,2))+'_ar'+str(round(L/phi0,2))+'.p


    # print(rp0-rp)

cvi_sim(1373,20000,100e-6, 100, 10, "red","T=1373K",1000000,"rp")

#cvi_sim(1223,10000,34e-6, (10e-3)/(34e-6), 5, "red","T=1223K",10000000,"rp")

# cvi_sim(1223,20000,100e-6, 100, 10, "red","T=1223",10000000,"C")
# cvi_sim(1073,20000,100e-6, 100, 10, "blue","T=1073",500000000,"C")
```

```
D= 0.014695257706332171 k_het = 0.021304420671514133 T= 1373 P= 20000 CMTS0= 0.15927835183731004
n= 10 : detA =  0.0 , pore is 35.64 % closed
n= 20 : detA =  0.0 , pore is 58.18 % closed
n= 30 : detA =  0.0 , pore is 72.86 % closed
n= 40 : detA =  0.0 , pore is 82.4 % closed
n= 50 : detA =  0.0 , pore is 88.59 % closed
n= 60 : detA =  0.0 , pore is 92.66 % closed
n= 70 : detA =  0.0 , pore is 95.3 % closed
n= 80 : detA =  0.0 , pore is 97.04 % closed
n= 90 : detA =  0.0 , pore is 98.24 % closed
n= 100 : detA =  0.0 , pore is 99.1 % closed
100
```


1e−5
5
4
3
2
1
0
SiC deposition (μm)
0.000
0.001
0.002
0.003
0.004
0.005
Infiltration depth (mm)
T=1373K