6 Appendix

6.1 Detailed derivation of governing equation and corresponding solution

The gas penetration into a falling film is focus of present modeling work. Fig. 1 illustrates the coordinates and the mass transfer properties. Assuming two dimensional, steady state mass transfer, no chemical reaction between gas (A) and falling liquid (B), no diffusion in flow direction (y), no convective mass transfer in diffusion direction (z), constant diffusivity (D)

and liquid density, then the continuity of mass results in Eq. S. 1, where c_A is the concentration profile of the penetrating gas.

$$u_y \frac{\partial c_A}{\partial y} = D \frac{\partial^2 c_A}{\partial z^2} \tag{S. 1}$$

The velocity profile in y-direction is given as in Eq. S. 2, where ρ is density, g is gravity acceleration, μ is the viscosity of falling liquid film and δ is the liquid film thickness.

$$u_y = \frac{\rho g \delta^2}{2\mu}\left[1-\left[\frac{z}{\delta}\right]^2\right] \tag{S. 2}$$

Short contact time assumption reads $z \ll \delta$, therefore, Eq. S. 2 read Eq. S. 3, where u_m is the maximum velocity of falling liquid film, which is related to the average velocity ($\bar{u}_y$) as $u_m = (3/2)\bar{u}_y$.

$$u_y \cong \frac{\rho g \delta^2}{2\mu} = u_m \tag{S. 3}$$

Therefore, the final form of governing equation reads Eq. S. 4 where $b = 2D/3\bar{u}_y$

$$\frac{\partial c_A}{\partial y} = b \frac{\partial^2 c_A}{\partial z^2} \tag{S. 4}$$

For the infinite penetration depth, the solution of partial differential equation (PDE) can be found in a previous paper [3]. For the finite penetration depth, using the separation of variables method by setting $c_A(y,z) = Y(y) \times Z(z)$ and $\varphi_A = c_A - c_{A0}$, the PDE in Eq. S. 4 can be transformed into Eq. S. 5 where λ^2 is the separation factor.

$$\frac{Y'}{bY} = \frac{Z''}{Z} = -\lambda^2 \tag{S. 5}$$

The general solutions to each parts in Eq. S. 5 can be obtained as given in Eqs. S. 6 and S. 7.

$$Y = C_1 e^{-b\lambda^2 y} \tag{S. 6}$$

$$Z = C_2 \sin(\lambda z) + C_3 \cos(\lambda z) \tag{S. 7}$$

By applying boundary conditions, one would obtain

$$z = 0 \quad \rightarrow \quad \begin{matrix} c_A = c_{A0} \\ \varphi_A = 0 \end{matrix} \quad \rightarrow \quad C_3 = 0 \tag{S. 8}$$

$$z = \xi \quad \rightarrow \quad \begin{matrix} \dfrac{\partial c_A}{\partial z} = 0 \\ \dfrac{\partial \varphi_A}{\partial z} = 0 \end{matrix} \quad \rightarrow \quad C_2 \lambda \cos(\lambda \delta) = 0 \quad \rightarrow \quad \lambda_n = \frac{2n+1}{\delta} \frac{\pi}{2} \tag{S. 9}$$

$$y = 0 \quad \rightarrow \quad \begin{matrix} c_A = c_{Ai} \\ \varphi_{A0} = c_{A0} - c_{Ai} \end{matrix} \quad \rightarrow \quad \sum_{n=1}^{\infty} C_n \sin(\lambda_n z) = c_{Ai} \quad \rightarrow \quad C_n = \frac{2\varphi_{A0}}{\lambda_n \delta} \tag{S. 10}$$

Finally, the concentration profile is given as in Eq. S. 11 or similarly as in Eq. S. 12.

$$\frac{c_A - c_{Ai}}{c_{A0} - c_{Ai}} = \frac{4}{\pi} \sum_{n=0}^{\infty} \left[\frac{e^{-\frac{2D}{3\bar{u}_y}\left[\frac{1}{\xi}\frac{2n+1}{2}\pi\right]^2 y} \sin\left(\frac{z}{\xi}\frac{2n+1}{2}\pi\right)}{2n+1} \right] \tag{S. 11}$$

$$\frac{c_A - c_{A0}}{c_{Ai} - c_{A0}} = 1 - \frac{4}{\pi} \sum_{n=0}^{\infty} \left[\frac{e^{-\frac{2D}{3\bar{u}_y}\left[\frac{1}{\xi}\frac{2n+1}{2}\pi\right]^2 y} \sin\left(\frac{z}{\xi}\frac{2n+1}{2}\pi\right)}{2n+1} \right] \tag{S. 12}$$

To find ξ, the introduced finite penetration depth, one could use the boundary condition $z = \xi \leftrightarrow c_A = c_{A0}$, which reads Eq. S. 13.

$$\frac{\pi}{4}=\sum_{n=0}^{\infty}\left[\frac{(-1)^n}{2n+1}e^{-\frac{2D}{3\bar{u}_y}\left[\frac{2n+1}{2\xi}\pi\right]^2 y}\right] \qquad \text{S. 13}$$

For engineering applications, the mass transfer characteristics are of the interest including the gas flux at the interface of falling film and gas stream ($N_A|_{z=0}$), the mass transfer coefficient (k_c), and the average mass transfer coefficient alongside the vertical coordinate i.e. falling film wall from *y=0* to *y=L* ($\bar{k}_c$) as given in Eq. S. 14-S. 16.

$$N_A|_{z=0}=-D\frac{\partial c_A}{\partial z}\bigg|_{z=0} \quad\rightarrow\quad \frac{2D(c_{Ai}-c_{A0})}{\xi}\sum_{n=0}^{\infty}e^{-\frac{2D}{3\bar{u}_y}\left[\frac{1}{\xi}\frac{2n+1}{2}\pi\right]^2 y} \qquad \text{S. 14}$$

$$k_c=\frac{N_A|_{z=0}}{(c_{Ai}-c_{A0})} \quad\rightarrow\quad \frac{2D}{\xi}\sum_{n=0}^{\infty}e^{-\frac{2D}{3\bar{u}_y}\left[\frac{1}{\xi}\frac{2n+1}{2}\pi\right]^2 y} \qquad \text{S. 15}$$

$$\bar{k}_c=\frac{\int_0^L k_c dy}{\int_0^L dy} \quad\rightarrow\quad 12\frac{\bar{u}_y\xi}{L\pi^2}\left[\frac{\pi^2}{8}-\sum_{n=0}^{\infty}\frac{e^{-\frac{2D}{3\bar{u}_y}\left[\frac{1}{\xi}\frac{2n+1}{2}\pi\right]^2 L}}{[2n+1]^2}\right] \qquad \text{S. 16}$$

6.2 Python scripts for computational experiments

Code for case A:

```
##############################
# MAK - python3 code
# file name: varD.py
# (in)finite penetration depth : varying D
##############################
import numpy as np
import math
##############################
# system data
L=1 # wall lenght
gamma=0.05 # film flow rate kg/m.s
cA0=0 # initial concentration
cAi=0.0366 # interfacial concentration
var=[1.0, 2.0, 2.5, 5.0, 7.0, 8.0, 9.0]
varD=[vard/1000000000 for vard in var]
##############################
Rho=998 # kg/m3
Mue=0.000894 # kg/m.s at STP
g=9.807 # m/s2
##############################
# y-coordinate
YY=[0.00000000001, 0.0000000001, 0.000000001, 0.00000001, 0.0000001,
0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1]
```

```
Y=[L*y for y in YY]
# film thickness
delta=((3*Mue*gamma)/((Rho**2)*g))**(1/3)
u=gamma/(Rho*delta)
# z-coordinate
Z=np.linspace(0, delta, num=50)
# mass transfer properties
############################
for D in varD:
      ############################
      b=(2*D)/(3*u)
      #
      infinity=100
      for y in Y:
            # finding kisi from BC2
            guesslist=np.logspace(0, 10, num=10000)
            guesslist=delta/guesslist
            guesslist=np.fliplr([guesslist])[0]
            KICI=[delta]
            for kisi in guesslist:
                  sigma=0
                  for n in range(infinity):
                        Sin=((-1)**n)/(2*n+1)
                        lmbda=((2*n+1)*math.pi)/(2*kisi)
                        lmbda2=lmbda**2
                        Expot=(-1)*lmbda2*b
                        Expt=y*Expot
                        Exp=np.exp(Expt)
                        An=Sin*Exp
                        sigma += An
                  Err=sigma - (math.pi/4)
                  if abs(Err)==0:
                        KICI.append(kisi)
            kisi=min(KICI)
            # calculating kc and NA
            Sum=0
            for n in range(infinity):
                  lmbda=((2*n+1)*math.pi)/(2*kisi)
                  lmbda2=lmbda**2
                  Expot=(-1)*lmbda2*b
                  Expt=y*Expot
                  Exp=np.exp(Expt)
                  An=Exp
                  Sum += An
            kc_infY=(2*D/kisi)*Sum
            NA_infY=kc_infY*(cAi - cA0)
            kc_infN=((3*u*D)/(2*math.pi*y))**0.5
            NA_infN=(cAi-cA0)*kc_infN
            # kcbar calculation
            Sum=0
            for n in range(infinity):
                  lmbda=((2*n+1)*math.pi)/(2*kisi)
                  lmbda2=lmbda**2
                  Expot=(-1)*lmbda2*b
                  Expt=L*Expot
                  Exp=np.exp(Expt)
                  An=Exp/((2*n+1)**2)
                  Sum += An
            kcbar_infY=(12*u*kisi/(L*(math.pi**2)))*((math.pi**2)/8 - Sum)
            kcbar_infN=((6*u*D)/(math.pi*L))**0.5
            # calculating cA
```

```
            for z in Z:
                  Sum=0
                  for n in range(infinity):
                        lmbda=((2*n+1)*math.pi)/(2*kisi)
                        lmbda2=lmbda**2
                        Expot=(-1)*lmbda2*b
                        Expt=y*Expot
                        Exp=np.exp(Expt)
                        Sin=math.sin(z*lmbda)
                        An=Sin*Exp/(2*n+1)
                        Sum += An
                  cA_infY=cA0+(cAi-cA0)*(1-(4/math.pi)*Sum)
                  Exp_infN=0.5*(((3*u*(z**2))/(2*D*y))**0.5)
                  cA_infN=cA0+(cAi-cA0)*(1-math.erf(Exp_infN))
                  print (D, y, z, delta, kisi, cA_infY, cA_infN, NA_infY, 
NA_infN, kc_infY, kc_infN, kcbar_infY, kcbar_infN, gamma, u)
```

Code for case B:

```
##############################
# MAK - python3
# varG.py
# (in)finite penetration depth : varying Gamma
##############################
import numpy as np
import math
##############################
# system data
L=1 # wall length
varG=[0.01, 0.02, 0.05, 0.08, 0.10, 0.20]
cA0=0 # initial concentration
cAi=0.0366 # interfacial concentration
D=1.96/1000000000 # diffusivity
##############################
Rho=998 # kg/m3
Mue=0.000894 # kg/m.s at STP
g=9.807 # m/s2
##############################
# y-coordinate
YY=[0.00000000001,   0.0000000001,   0.000000001,   0.00000001,   0.0000001, 
0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1]
Y=[L*y for y in YY]
############################
for gamma in varG:
      ############################
      # mass transfer properties
      delta=((3*Mue*gamma)/((Rho**2)*g))**(1/3)
      u=gamma/(Rho*delta)
      # z-coordinate
      Z=np.linspace(0, delta, num=50)
      b=(2*D)/(3*u)
      #
      infinity=100
      for y in Y:
            # finding kisi from BC2
            guesslist=np.logspace(0, 10, num=10000)
            guesslist=delta/guesslist
            guesslist=np.fliplr([guesslist])[0]
            KICI=[delta]
            for kisi in guesslist:
                  sigma=0
```

```
                  for n in range(infinity):
                        Sin=((-1)**n)/(2*n+1)
                        lmbda=((2*n+1)*math.pi)/(2*kisi)
                        lmbda2=lmbda**2
                        Expot=(-1)*lmbda2*b
                        Expt=y*Expot
                        Exp=np.exp(Expt)
                        An=Sin*Exp
                        sigma += An
                  Err=sigma - (math.pi/4)
                  if abs(Err)==0:
                        KICI.append(kisi)
            kisi=min(KICI)
            # calculating kc and NA
            Sum=0
            for n in range(infinity):
                  lmbda=((2*n+1)*math.pi)/(2*kisi)
                  lmbda2=lmbda**2
                  Expot=(-1)*lmbda2*b
                  Expt=y*Expot
                  Exp=np.exp(Expt)
                  An=Exp
                  Sum += An
            kc_infY=(2*D/kisi)*Sum
            NA_infY=kc_infY*(cAi - cA0)
            kc_infN=((3*u*D)/(2*math.pi*y))**0.5
            NA_infN=(cAi-cA0)*kc_infN
            # kcbar calculation
            Sum=0
            for n in range(infinity):
                  lmbda=((2*n+1)*math.pi)/(2*kisi)
                  lmbda2=lmbda**2
                  Expot=(-1)*lmbda2*b
                  Expt=L*Expot
                  Exp=np.exp(Expt)
                  An=Exp/((2*n+1)**2)
                  Sum += An
            kcbar_infY=(12*u*kisi/(L*(math.pi**2)))*((math.pi**2)/8  -  Sum)
            kcbar_infN=((6*u*D)/(math.pi*L))**0.5
            # calculating cA
            for z in Z:
                  Sum=0
                  for n in range(infinity):
                        lmbda=((2*n+1)*math.pi)/(2*kisi)
                        lmbda2=lmbda**2
                        Expot=(-1)*lmbda2*b
                        Expt=y*Expot
                        Exp=np.exp(Expt)
                        Sin=math.sin(z*lmbda)
                        An=Sin*Exp/(2*n+1)
                        Sum += An
                  cA_infY=cA0+(cAi-cA0)*(1-(4/math.pi)*Sum)
                  Exp_infN=0.5*(((3*u*(z**2))/(2*D*y))**0.5)
                  cA_infN=cA0+(cAi-cA0)*(1-math.erf(Exp_infN))
                  print  (gamma,  u,  y,  z,  delta,  kisi,  cA_infY,  cA_infN,
NA_infY, NA_infN, kc_infY, kc_infN, kcbar_infY, kcbar_infN, D)
```

References

1. McCabe, W.L. and J.C. Smith, *Unit Operations of Chemical Engineering*. 2 ed. 1967, New York: McGraw-Hill.
2. Treybal, R.E., *Mass Transfer Operations*. 3rd ed. McGraw-Hill Chemical Engineering Series. 1980: McGraw-Hill.
3. Ali Aroon, M. and M.A. Khansary, *Generalized similarity transformation method applied to partial differential equations (PDEs) in falling film mass transfer*. Computers & Chemical Engineering, 2017. **101**: p. 73-80.
4. Bird, R.B., W.E. Stewart, and E.N. Lightfoot, *Transport Phenomena*. 2006: John Wiley & Sons, Inc.