REFERENCES

1. Jones, R.L., Keller G.E., and Wells R.C., "Rapid Pressure Swing Adsorption Process with High Enrichment Factor," U.S. Patent 4,194,892 , (1980).

2. Fernandez, G.F., and Kenney C.N., "Modelling of the Pressure Swing Air Separation Process," Chem Eng Sci., 38(6), 827-834, 1983

3. Turnock, P.H., and Kadlec R.H., "Separation of nitrogen and methane via periodic adsorption," AIChE J.,17(2), 335-342, 1971.

4. Yang, R.T., Gas Separation by Adsorption Processes, Butterworths, 1987.

5. Sundaram N., and Wankat, P.C., "Pressure Drop Effects in the Pressurization and Blowdown Steps of Pressure Swing Adsorption," Chem Eng Sci., 43(1), 123-129, 1988.

6. Ruthven, D.M., Principle of Adsorption and Adsorption Processes, John Wiley & Sons , 1984 .

7. Ribeiro, F.R., Rodriques, A.E., and Rollmann L.D., Zeolites : Science and Technology, pp. 657-660, Martinus Nijhoff Publishers, 1984 .

8. Chan, Y.N.I., Hill, F.B., and Wong Y.W., "Equilibrium theory of a Pressure Swing Adsorption Process," Chem Eng Sci., 36, 243-251, 1981.

9. Bird, R.B., Stewart, W.E., and Lightfoot, E.N., Transport Phenomena, John Wiley & Sons, 149-150, 1960 .

10. Perry, J.H., Perry's Chemical Engineering Handbook, pp.3-162, McGraw Hill Book Company, 6th ed., 1984 .

11. Kayser, J.C., and Knaebel, K.S., "Pressure Swing Adsorption : Experimental Study of an Equilibrium Theory," Chem Eng Sci., 41(11), 2931-2938, 1986 .

12. Welty, J.R., Wicks, C.E., and Wilson, R.E., Fundamentals of Momentum Heat and Mass Transfer, pp. 732, John Wiley & Sons , 2nd ed. 1976 .

APPENDIX

APPENDIX A

DERIVATION OF PRESSURE DROP EQUATION

Following Turnknock [3] we used a combination of Darcy's law with the continuity equation to modelize the pressure drop effect in columns with porous medium such as adsorbent pellets.

We start from the continuity equation [8] . The modified equation of continuity may be expressed as

$$\varepsilon \frac{\partial \rho}{\partial t} = -(\nabla \rho v_0) \qquad (A.1)$$

Darcy's law is written as

$$v_0 = \frac{-K}{\mu} (\nabla p - \rho g) \qquad (A.2)$$

where

ε = porosity

K = permeability of the porous medium

v_0 = superficial velocity

substituting v_0 from equation (A.2) in equation (A.1) and defining

$$\nabla \Phi = \nabla p - \rho g \qquad (A.3)$$

We then obtain

$$\varepsilon \frac{\partial \rho}{\partial t} = -(\nabla \rho [\frac{-K}{\mu} (\nabla \Phi)]) \qquad (A.4)$$

$$(\frac{\varepsilon \mu}{K}) \frac{\partial \rho}{\partial t} = (\nabla [\rho \nabla \Phi]) \qquad (A.5)$$

from the equation of state : $\rho = \rho_o p^m \exp(\beta p)$ (A.6)

where ρ_o = fluid density at unit pressure

for gases; and $\beta = 0$ in case of isothermal expansion $m = 1$

We obtain from equation (A.6) when $\beta = o$; and $m = 1$ that

$$\rho = \rho_o \cdot p \qquad (A.7)$$

and equation (A.3) when the gravity term is less than the pressure term gives

$$\nabla \Phi = \nabla p \qquad (A.8)$$

and from equation (A.5) we obtain

$$(\frac{\varepsilon \mu}{K}) \frac{\partial p}{\partial t} = (\nabla [\rho \nabla p]) \qquad (A.9)$$

when $p = \frac{\rho}{\rho_o}$ substituting in equation (A.9) gives

$$\frac{\partial p}{\partial t} = \frac{K \nabla^2 p^2}{2 \varepsilon \mu} \qquad (A.10)$$

we have $C = \frac{K}{2\varepsilon\mu}$ in equation (A.10) and changing in to a differential operator gives

$$\frac{\partial p}{\partial t} = C \frac{\partial^2 p^2}{\partial z^2} \qquad (A.11)$$

APPENDIX B

THE FINITE DIFFERENCE METHOD

The approximation of partial derivatives by a finite difference method is as follows.

$$\frac{\partial p(z,t)}{\partial t} = \frac{p[m,n+1] - p[m,n]}{\Delta t} \qquad (B.1)$$

$$\frac{\partial p(z,t)}{\partial z} = \frac{p[m+1,n] - p[m,n]}{\Delta z} \quad \text{or} \qquad (B.2)$$

$$\frac{p[m,n]-p[m-1,n]}{\Delta z} \qquad (B.3)$$

$$\frac{\partial^2 p(z,t)}{\partial z^2} = \frac{p[m+1,n] - 2p[m,n] + p[m-1,n]}{(\Delta z)^2} \qquad (B.4)$$

where m refers to distance (z) which goes up to m=max with increments Δ z, and n refers to time (t) at increments Δ t

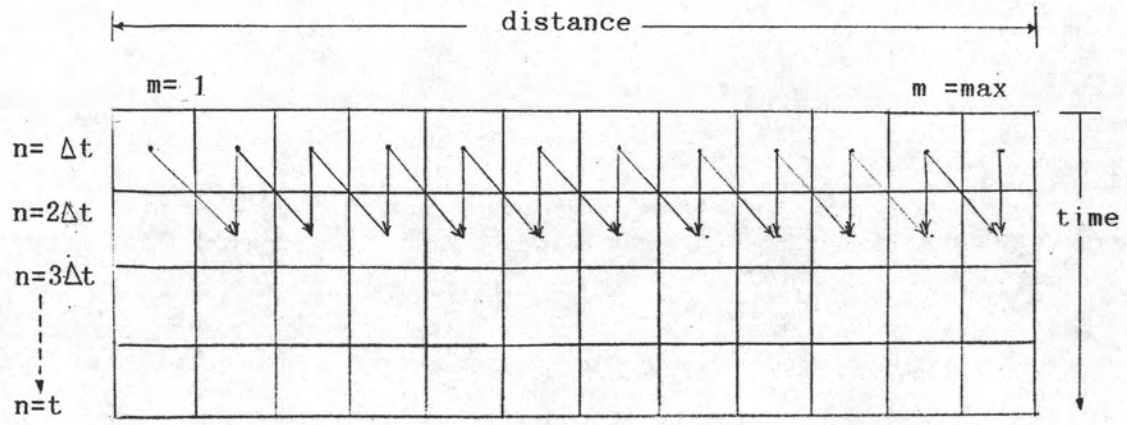

Figure B.1 Diagram illustrating how to fill the vectors using the finite difference method.

If we know the solution at time zero, then we can find the solution for $n = \Delta t$ at each increment (Δz) and so on, and we can determine solution, at time = $2\Delta t$, $3\Delta t$, $4\Delta t$, ... starting with the boundary conditions we set. The calculation step in this simulation involves first finding the pressure at each increment Δz from equation (3.1), with the boundary condition for the feed gas introduction period in equation (3.32).This is shown in the form of a finite difference as follows.

From equation (3.1) :

$$\frac{p[m,n+1]-p[m,n]}{\Delta t} = 2c \{ p[m,n] \left(\frac{p[m+1,n]-2p[m,n]+p[m-1,n]}{(\Delta z)^2}\right) + \left(\frac{p[m+1,n]-p[m,n]}{\Delta z}\right)^2 \} \qquad (B.5)$$

From equation (3.32) :

$$v[max,n]p[max,n]A\varepsilon = V_g\frac{[p[max,n+1]-p[max,n]]}{\Delta t} + ERT \qquad (B.6)$$

Secondly we use equation (3.28) for finding the mole fraction (y). Equation (3.28) becomes the following in finite difference form

$$p[m,n][K_1-(K_1-K_2)y[m,n]][y[m,n](1-y[m,n])(K_1-K_2)]\frac{[p[m,n+1]-p[m,n]]}{\Delta t}$$

$$+\ p[m,n][K_1-y[m,n](K_1-K_2)]^2\frac{[y[m,n+1]-y[m,n]]}{\Delta t}$$

$$+\ K_1K_2\{\ -2cp[m,n]\frac{[p[m,n]-p[m-1,n]]}{\Delta z}\ +\ 2cp[max,n]\frac{[p[max,n]-p[max-1,n]]}{\Delta z}$$

$$+[\frac{V_s}{\varepsilon A}[\frac{p[max,n]-p[max,n+1]]}{\Delta t}+\frac{ERT}{A}]][\frac{1}{K_2}+\frac{(K_2-K_1)}{K_1K_2}y[max,n]]\}\frac{[y[m,n]-y[m-1,n]]}{\Delta z}$$

$$=\quad 0 \qquad\qquad (B.7)$$

Thirdly we can find the velocity from equation (3.27) using boundary condition (3.32), but for equation (3.32) we replace the term $\frac{\partial p(L,t)}{\partial t}$ by $C\,\frac{\partial^2 p^2(L,t)}{\partial z^2}$ and equation (3.32) becomes

$$v(L,t) = \left[\frac{V_s}{A\ \varepsilon\ p(L,t)}\right]\left[C\,\frac{\partial^2 p^2(L,t)}{\partial z^2}\right] + \frac{ERT}{A\,\varepsilon\,p(L,t)} \qquad (B.8)$$

substituting in equation (3.27) the equation in finite difference form is as follows :

$$v[m,n+1] = \{-2cp[m,n+1][p[m+1,n+1]-p[m,n+1]]$$

$$\frac{}{\Delta z}$$

$$+ 2cp[max,n+1][p[max-1,n+1]-p[max,n+1]]$$

$$\frac{}{\Delta z}$$

$$+\left(\frac{V_s}{A\cdot\varepsilon}\right.\left[2cp[max,n+1]\left[\frac{p[max,n+1]-2p[max-1,n+1]+p[max-2,n+1]}{(\Delta z)^2}\right]\right.$$

$$+ 2c\left[\frac{p[max,n+1]-p[max-1,n+1]}{\Delta z}\right]^2\left.\left.+\frac{ERT}{\varepsilon A}\right)\left[\frac{1}{K_2}+\frac{(K_2-K_1)}{K_1K_2}y[max,n+1]\right]\right\}$$

$$/\left[p[m,n+1]\left[\frac{1}{K_2}+\frac{(K_2-K_1)}{K_1K_2}y[m,n+1]\right]\right] \quad (C.9) \qquad (B.9)$$

For the reverse outward flow period we use the same equation but with different boundary conditions as mentioned before .

APPENDIX C

COMPUTER PROGRAM

Following this is computer program in c language which used in this simulation.

```
/* program for simulation for  C  language  */
/* PROGRAM  AIR SEPRATION  (ADSORPTION & BLOWDOWN ) */

#include <stdio.h>

float  ph,max,dp,por,vis,l,d,c,g,a,yf,dels,delt,tip,tib,endcy,k[3],ki[3],
       p0[52],p1[52],v0[52],v1[52],y0[52],y1[52],Q1,Q2,Q3,A1,A2,vc,d2,den,
       mout,mw,b;


int    cyb,cyp, p1,v2,z ;


main()
{
  int   i,m;

      printf("  enter no of cycle (endcy ) =  ");
     scanf(" %f ",&endcy);
     printf( " endcycle = %f \n ",endcy );
     printf("  enter time of pressurization =  ");
     scanf(" %f ",&tip);
     printf( " tip = %f \n ",tip );
     printf("  enter time of blowdown = " );
     scanf(" %f ",&tib);
     printf( " tib = %f \n ",tib );
     scanf(" %f ",&mout);
     printf( " moleout = %f \n ",mout );
     scanf(" %f ",&dp);
     printf( " dp = %f \n ",dp );

      ph=4.4 ; p1 =1;
      por =.43 ;vis= 1.8257e-10;
      l = 150; d = 7.62; v2=2250 ;
      g = por*por*por*dp*dp/(1-por)/(1-por)/150 ;
      c = 2*g /(2*por*vis);
      a = 3.1416/4*d*d ;
      ki[2] = 9.94 ; ki[1] = 5.4;
      yf = .21;
      dels =7.5 ; delt=.001;
     den =1.1769e-3   ; mw =29;
```

```
        for (i=1; i<=2; i++ )
        {
          k[i] =(por+(1-por)*ki[i])/por ;
          printf("k[i]=%f \n ",k[i] );
        }

        /* constant variable  */
        vc =  v2/(por*a) ;
        d2 =  dels*dels ;
        A1 =  a*por*delt/v2;
        Q1 =  1/k[2] ;
        Q2 =  (k[2]-k[1])/(k[1]*k[2]) ;
        Q3 =  (k[1]-k[2]);

      max = 19 ;
      printf( " max = %f \n ",max );
      cyp = (int)(tip/delt) ;
      printf( " cyp = %d \n ",cyp );
      cyb = (int)(tib/delt) ;
      printf( " cyb = %d \n ",cyb );

      printf( " program final3.c with v2= %d \n ",v2 );
      for (m = 0 ; m <= max ; m++)
        {
          p0[m] = pi ;
          y0[m] = yf ;
          v0[m] = 0 ;
        }

/* PROGRAM HAVE 2 STEP (ADSORPTION & BLOWDOWN ) */

/*  MAIN LOOP AT Z=Z+1 */

    z =0;
 while ( z++ <= endcy )
      {
        prez();
        blow();
      }

   printf("   end of cycle at cy =  %f \n",endcy );
   printf("  cal at Delt =  %f and Dels =  %f \n",delt,dels );
}   /*  for main  */

/*   ********************** function pressurization  ******************  *

prez()
{
   int    j,m ;
register   float  b1,b2,pc0,pc1,pc2,yc0,pmax0,ymax0,pmax1,ymax1,pdel,co
                  fin,fo,x,;

  /* PRESSURIZATION STEP */

    A2 = mout*82.05*298*delt/v2;
     con  =  mout*82.05*298/a/por;
       x =50;
```

```
for (j= 1 ; j <= cyp ; j++)
  {
     p1[0] = ph;
     y1[0] = yf;
     pmax0 = p0[max];
     ymax0 = y0[max];

 /*  ********************* CALCULATE PRESSURE ************************ */


     for (m = 1; m <= (max-1) ; m++)
       {
       pc0 = p0[m];
       pc1  = (p0[m+1]-pc0)/dels ;
       pc2  = pc1*pc1 ;

        p1[m] =pc0 +delt*c*(pc0*(p0[m-1]-2*pc0 + p0[m+1])/d2+pc2) ;

       }
       p1[max] =(v0[max]*pmax0*A1-A2)+pmax0 ;


 /*  ************************ calculate mole fraction  ***************  */

     for (m=1 ; m <= max ; m++)
       {
       yc0 = y0[m];
       pc0 = p0[m];

        b1 = (c*pc0*(pc0-p0[m-1])/dels-c*pmax0*(pmax0-p0[max-1])/dels
             -(vc*(p1[max]-pmax0)/delt+con)*(Q1+Q2*ymax0))/
              (pc0*(Q1+Q2*yc0))*(yc0-y0[m-1])/dels;

        b2 = -yc0*(1-yc0)*Q3*(p1[m]-pc0)/delt/pc0;

        y1[m]= delt/(k[1]-yc0*Q3)*(b1+b2)+yc0;

        }

 /*  ************************  velocity ***************************** */
         pmax1 = p1[max] ;
         ymax1 = y1[max] ;
     for (m=0 ; m <= (max-1) ;m++ )
       {
        pdel  = (pmax1-p1[max-1])/dels ;
        v1[m] = (-c*p1[m]*(p1[m+1]-p1[m])/dels+c*pmax1*pdel
               +(vc*(c*pmax1*(pmax1-2*p1[max-1]+p1[max-2])/d2
               +c*pdel*pdel)+con)*(Q1+Q2*ymax1))/(p1[m]*(Q1+Q2*y1[m])) ;

        }

        v1[max] = (4*v1[max-1]-v1[max-2])/3;
```

```
/*  %%%%%%%%%%%%%%% print data on file %%%%%%%%%%%%%%%%%% */
   if (z<10)
   {
    if (  j == cyp )
    {
    printf(" at cycle( z )=  %d \n",z );
    printf("%d %d %d   \n",j,j,j  );
    for ( m=0 ; m<= max ; m++ )
      {
       printf(" %f %f %f  \n",p1[m],y1[m],v1[m] );
       }
    printf("   flow in =  %f \n",fin );
    printf("   flow out =  %f \n",fo );
     }
   }
   if (z >=10)
   {
    if( j == 1)
    {
    printf(" ***v1[0] =  %f \n",v1[0]);
    }
    if (  j == x )
    {
    printf(" at cycle( z )=  %d \n",z );
    printf("%d %d %d   \n",j,j,j  );
    for ( m=0 ; m<= max ; m++ )
      {
       printf(" %f %f %f  \n",p1[m],y1[m],v1[m] );
       }
    printf("   flow in =  %f \n",fin );
    printf("   flow out =  %f \n",fo );
     x = x+50;
     }
   }
/*  &&&&&&&&&&&&&&&& change p y v  to restart &&&&&&&&&&&&&&&&&& */
  for (m=0 ;m<= max; m++ )
  {
    p0[m] = p1[m] ;
    y0[m] = y1[m] ;
    v0[m] = v1[m] ;
   }
 }      /*  for j=1 to cyp */
}       /*  for adsorption function */
```

```
/*  function blowdown */

 blow()
 {
              int  j,m;
  register  float  b1,b2,pc0,pc1,pc2,yc0,pmax0,ymax0,pmax1,ymax1,pdel,con,
                   fin,fo,x;.

   /* BLOWDOWN STEP */
      b=y0[max];
     A2 = mout*82.05*298*delt/v2;
      con  =  mout*82.05*298/a/por;
       x   =  1000;

   for ( j=1 ; j <= cyb ; j++ )
    {

   /*  %%%%%%%%%%%%%%%%%%%% CALCULATE PRESSURE %%%%%%%%%%%%%%%%%%%% */

        p1[0] = 1;
        pmax0 = p0[max];
        ymax0 = y0[max];


        for (m=1 ;m<= max-1 ;m++ )
        {
        pc0 = p0[m];
        pc1  = (p0[m+1]-pc0)/dels ;
        pc2  = pc1*pc1 ;

        p1[m] =pc0 +delt*c*(pc0*(p0[m-1]-2*pc0 + p0[m+1])/d2+pc2) ;

         }
         p1[max] =-(v0[max]*pmax0*A1+A2)+pmax0 ;

     /*  %%%%%%%%%%%%%%%%%%%%%% clculate mole fraction  %%%%%%%%%%%%%%%  */

    for (m=0; m<=max-1 ;m++)
     {
        yc0 = y0[m];
        pc0 = p0[m];

         b1 = (c*pc0*(p0[m+1]-pc0)/dels-c*pmax0*(pmax0-p0[max-1])/dels
              -(-vc*(p1[max]-pmax0)/delt-con)*(Q1+Q2*ymax0))/
               (pc0*(Q1+Q2*yc0))*(y0[m+1]-y0[m])/dels;

         b2 = -yc0*(1-yc0)*Q3*(p1[m]-pc0)/delt/pc0;

       y1[m]= delt/(k[1]-yc0*Q3)*(b1+b2)+yc0;
     }
       y1[max] =b ;
```

```
/*  %%%%%%%%%%%%%%%%%%%%%%%%   velocity  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  */

    pmaxl = pl[max] ;
    ymaxl = yl[max] ;
    pdel  = (pmaxl-pl[max-1])/dels ;
for (m=0 ; m<=max-1 ;m++ )
   {
    vl[m] = (-c*pl[m]*(pl[m+1]-pl[m])/dels+c*pmaxl*pdel
             +(-vc*(c*pmaxl*(pmaxl-2*pl[max-1]+pl[max-2])/d2
             +c*pdel*pdel)-con)*(Q1+Q2*ymaxl))/(pl[m]*(Q1+Q2*yl[m])) ;

   if (vl[m] >0 )
    {
     vl[m] = vl[m-1] ;
     }

   }

   vl[max] =(4*vl[max-1]-vl[max-2])/3 ;
   if (vl[max] >0 )
    {
     vl[max] = vl[max-1] ;
     }
     min = vl[0]*a*por*den/mw;
     fin = vl[0]*a*por;
     fo = mout*mw/den;
     mout1 = mout*delt*j;
/*  %%%%%%%%%%%%%%% print data on file %%%%%%%%%%%%%%%%%%  */
  if (z<10)
  {
  if (  j == cyb )
   {
   printf(" at cycle( z )=  %d \n",z );
    printf("%d %d %d   \n",j,j,j  );
    for ( m=0 ; m<=max ; m++ )
      {
       printf( " %d %f %f %f \n",m,pl[m],yl[m],vl[m]);
       }
     printf("   flow in =  %f \n",fin );
     printf("   flow out =  %f \n",fo );
    }
```

```
      if (z >=10)
       {
        if( j == 1)
        {
        printf(" ***v1[0] = \n",v1[0]);
        }
        if (  j == x )
        {
        printf(" at cycle( z )=  %d \n",z );
        printf("%d %d %d   \n",j,j,j  );
        for ( m=0 ; m<= max ; m++ )
          {
           printf(" %f %f %f  \n",p1[m],y1[m],v1[m] );
           }
        printf("   flow in =  %f \n",fin );
        printf("   flow out =  %f \n",fo );
         x = x+1000;
         }
       }
    /*  &&&&&&&&&&&&&&&& change p y v  to restart &&&&&&&&&&&&&&&&&& */
       for (m=0 ;m<= max; m++ )
          {
           p0[m] = p1[m] ;
           y0[m] = y1[m] ;
           v0[m] = v1[m] ;
           }

     }          /* for j=1 to cyb */
}             /* for blowdown function */
```

BIOGRAPHY

Wannakul Kuttiajewa was born on January 18,1962 in Bangkok, Thailand. She graduated Mathayom 5 from Satrividhaya 2 school in 1980 .She recieved her Bachelors Degree of Science in chemical engineering program from Chulalongkorn University in 1984. She continued her Master's degree at Chulalongkorn University in 1985. She was granted the degree in May 1990.