Atmospheric
Chemistry
and Physics

Supplement of

Rethinking Craig and Gordon's approach to modeling isotopic compositions of marine boundary layer vapor

Xiahong Feng et al.

Correspondence to: Xiahong Feng (xiahong.feng@dartmouth.edu)

Contents of this file

      Text S1 to S4

Additional Supporting Information (Files uploaded separately)

      None.

Introduction

The supporting information contains four parts. The first two parts (S1 and S2) are analytical solutions for the governing equations of the middle (Eq. 14) and the top (Eq. 15) layers of the isotopic marine boundary layer (IMBL) model. S3 contains the expression of the integration constant C_1 (Concentration ratio at the height of h_1). All three equations are presented as Mathematica© output, including variables defined in the manuscript and the notations of Mathematica©. S4 contains the Matlab code for the IMBL model.

Text S1.

Solution of Eq. 14 (the middle layer of the IMBL model)

C[z] = {

  2^(-beta) (-1 + beta)^(-1)

  E^(-(2 h1 Kmax rho - 2 h2 Kmax rho)^(-1) (h1^2 wa + wa z^2))

  h1^(-1) (Km + Kmax)^(-1) Pi^Rational[-1, 2] wa^Rational[-1, 2]

  Gamma[1 + Rational[-1, 2] beta]^(-1)

  Log[Km^(-1) (Km + Kmax)]^(-1) (-2^Rational[1, 2] (C0 - C1)

   E^(h1 (h1 Kmax rho - h2 Kmax rho)^(-1) wa z) (h1 - h2)^Rational[1, 2]

   Kmax^Rational[3, 2] rho^Rational[1, 2]

   Gamma[Rational[1, 2] (3 - beta)] (

   2 Gamma[1 + Rational[-1, 2] beta]

   HermiteH[-1 + beta, 2^Rational[-1, 2] (h1 - h2)^Rational[-1, 2]

   Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2] (-h1 + z)] -

   2^beta Pi^Rational[1, 2] Hypergeometric1F1[Rational[1, 2] (1 - beta),

  Rational[1, 2], Rational[1, 2] (h1 - h2)^(-1) Kmax^(-1) rho^(-1)

   wa (h1 - z)^2]) - 2^beta (-1 + beta) h1 (Km + Kmax) Pi^Rational[1, 2]

   wa^Rational[1, 2]

   Gamma[1 + Rational[-1, 2] beta] ((C1 - CE)

   E^(h1 (h1 Kmax rho - h2 Kmax rho)^(-1) wa z)

   Hypergeometric1F1[Rational[1, 2] (1 - beta),

  Rational[1, 2], Rational[1, 2] (h1 - h2)^(-1) Kmax^(-1) rho^(-1)

   wa (h1 - z)^2] (Log[Km] - Log[Km + Kmax]) - CE

   E^((2 h1 Kmax rho - 2 h2 Kmax rho)^(-1) (h1^2 wa + wa z^2))

   Log[Km^(-1) (Km + Kmax)]))}

Text S2.

Solution of Eq. 15 (the top layer of the IMBL model)

C [Z]={

  2^(-beta) (-1 + beta)^(-1)

  E^(-(2 h1 Kmax rho - 2 h2 Kmax rho)^(-1) (h1^2 wa + h2^2 wa)) h1^(-1)

  Kmax^(-1) (Km + Kmax)^(-1) ((h2 - h3) (Km + Kmax))^((-h2 + h3) Kmax^(-1)

  rho^(-1) wa) Pi^Rational[-1, 2] wa^(-1)

  Gamma[1 + Rational[-1, 2] beta]^(-1)

  Log[Km^(-1) (Km + Kmax)]^(-1) ((C0 - C1)

  E^(h1 h2 (h1 Kmax rho - h2 Kmax rho)^(-1) wa) Kmax^Rational[3, 2]

  rho^Rational[1, 2] Gamma[Rational[1, 2] (3 - beta)] (2 Gamma[1 + Rational[-1, 2] beta] (

       2 (-1 + beta) Kmax^Rational[1, 2] (Km + Kmax) rho^

       Rational[1, 2] (((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) - (

       h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1) wa))

       HermiteH[-2 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

       Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]] +

       2^Rational[1, 2] (h1 - h2)^Rational[1, 2] wa^Rational[1, 2] (

       Km ((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) -

       Km (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1)

       wa) - Kmax (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1)

       wa)) HermiteH[-1 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

       Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]]) +

       2^(Rational[1, 2] + beta) (h1 - h2)^Rational[1, 2] Pi^Rational[1, 2]

       wa^Rational[1, 2] ((-Km ((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) +

       Km (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1)

wa) + Kmax (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1)

   wa)) Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] - (-1 +

   beta) (Km + Kmax) (((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) –

(h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1) wa))

   Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa])) +

  2^beta (-1 + beta) h1 (Km + Kmax) Pi^Rational[1, 2] wa

  Gamma[1 + Rational[-1, 2] beta] (-(C1 - CE) E^(h1 h2 (h1 Kmax rho - h2 Kmax rho)^(-1) wa) (-Km ((h2 -

h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) +

   Km (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1) wa) +

   Kmax (h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1)

   wa)) Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] (Log[Km] -

   Log[Km + Kmax]) + (-1 + beta) (C1 - CE)

   E^(h1 h2 (h1 Kmax rho - h2 Kmax rho)^(-1) wa) (Km +

   Kmax) (((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa) - (

    h2 Km - h3 (Km + Kmax) + Kmax z)^((h2 - h3) Kmax^(-1) rho^(-1) wa))

   Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] (Log[Km] -

   Log[Km + Kmax]) +

   CE E^((2 h1 Kmax rho - 2 h2 Kmax rho)^(-1) (h1^2 wa + h2^2 wa))

   Kmax ((h2 - h3) (Km + Kmax))^((h2 - h3) Kmax^(-1) rho^(-1) wa)

   Log[Km^(-1) (Km + Kmax)]))}

Text S3.

Solution for constant of integration, C_1 (i.e., the value of $C(h_1)$)

C1={(

  2^(-beta) (-1 + beta)^(-1) h1^(-1) Kmax^Rational[3, 2] Pi^Rational[-1, 2]

  rho^Rational[1, 2] Gamma[Rational[1, 2] (3 - beta)]

  Gamma[1 + Rational[-1, 2] beta]^(-1) (2 Gamma[1 + Rational[-1, 2] beta] (

  2 (-1 + beta) Kmax^Rational[1, 2] rho^Rational[1, 2]

  HermiteH[-2 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

  Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]] +

  2^Rational[1, 2] (h1 - h2)^Rational[1, 2] wa^Rational[1, 2]

  HermiteH[-1 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

  Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]]) -

  2^(Rational[1, 2] + beta) (h1 - h2)^Rational[1, 2] Pi^Rational[1, 2]

  wa^Rational[1, 2] (Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] + (-1 + beta)

  Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa])) - (Km +

  Kmax) wa (Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] + (-1 + beta)

  Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa]) (Log[Km] -

  Log[Km + Kmax]))^(-1) (2^(-beta) (-1 + beta)^(-1) C0 h1^(-1) Kmax^Rational[3, 2]

  Pi^Rational[-1, 2] rho^Rational[1, 2] Gamma[Rational[1, 2] (3 - beta)]

  Gamma[1 + Rational[-1, 2] beta]^(-1) (2 Gamma[1 + Rational[-1, 2] beta] (

  2 (-1 + beta) Kmax^Rational[1, 2] rho^Rational[1, 2]

  HermiteH[-2 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]] +

2^Rational[1, 2] (h1 - h2)^Rational[1, 2] wa^Rational[1, 2]

HermiteH[-1 + beta, -2^Rational[-1, 2] (h1 - h2)^Rational[1, 2]

Kmax^Rational[-1, 2] rho^Rational[-1, 2] wa^Rational[1, 2]]) -

2^(Rational[1, 2] + beta) (h1 - h2)^Rational[1, 2] Pi^Rational[1, 2]

wa^Rational[1, 2] (Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] + (-1 + beta)

Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa]) -

CE (Km + Kmax) wa (Hypergeometric1F1[Rational[1, 2] (1 - beta),

Rational[1, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa] + (-1 + beta)

Hypergeometric1F1[Rational[1, 2] (3 - beta),

Rational[3, 2], Rational[1, 2] (h1 - h2) Kmax^(-1) rho^(-1) wa]) (Log[Km] -

Log[Km + Kmax]))}

Text S4.

Matlab© code to solve for and display isotopologue concentrations in the marine boundary layer.

```
function  PBL_analy_numer
% Analytic/Numerical solution of the piecewise BVP for
% isotopic transport in the planetary boundary layer (PBL)
% using Matlab's boundary value problem solver bvp5c
%
%  Eric Posmentier and Leslie Sonder       Spring 2014
%  November 2014 - split between analytic (lo) numerical (mid-hi)

%clear all
close all
disp(' ');
disp(' ');
warning('off','MATLAB:deval:NonuniqueSolution')

%    Exact solutions for Low layer: Mathematica output
% Low layer, Clo[z]
%    {(-(C1*Log[h1*Km]) + C0*Log[h1*(Km + Kmax)] + ...
%    (-C0 + C1)*Log[h1*Km + Kmax*z])/Log[(Km + Kmax)/Km]}
```

```
% Low layer gradient, Clo'[z]
%    {{((-C0 + C1)*Kmax)/((h1*Km + Kmax*z)*Log[(Km + Kmax)/Km])}}
% Low layer, Clo[z] at z=h1
%    {{C1}}
% Low layer gradient, Clo'[z], at z=h1
%    {{((-C0 + C1)*Kmax)/(h1*(Km + Kmax)*Log[(Km + Kmax)/Km])}}
% In addition,
%    Clo(0)=(C0)
%    Clo'(0)=(-C0 + C1)*Kmax/(h1*Log[Km + Kmax])
%
% define adjustable parameters

% heights
h1 = 100;         % height of top of bottom layer (m)
h2 = 650;         % height of top of middle layer (m)
h3 = 1000;         % height of top of top layer (m)
% external air
CE0 = 0.0005;     % mixing ratio of external air (kg vapor / kg da)
CE18O = -33;      % delta18O of external air
CED = -239;       % deltaD of external air
% atmospheric and sea surface parameters
Tssc = 5;         % sea surface temp, ∞C
SSP = 1.013e5;    % sea surface pressure (Pa)
SSd18O = 0.0;     % sea surface delta-O18 (moles 18O liquid/kmole 16O liquid)
SSdD = 0.0;       % sea surface delta-D (moles D liquid/kmole 1H, per mil)
% mixing and vertical advection
wa = 0.15;        % max upwards velocity (m/s) h1:h2 and above
alpha = 0.05;     % proportion of external air entrained; equal to
beta in the manuscript
Kmax=1.000;          % maximum of total diffusion coefficient for O16, z1:z2
Ktop=100;         % factor by which turbulent diffusion at h3 exceeds km0

% define constants & other preliminary setup

% isotope constants
vsmowO18 = 2005.20e-6; % molecular ratios to vsmow16
vsmowD = 2*155.75/(1e6-155.76);
% molecular masses of isotopologues (kg/mole)
mDair = 0.02897;    %  dry air
mO16 =  0.0180106;  %  H216O
mO18 =  0.0200148;  %  H218O
mD =     0.0190163;  %  HD16O
% concentrations of 3 isotopologues in converging external air
CEarray = zeros(1,3);
CEarray(1,1) = CE0/(1+vsmowO18*mO18/mO16 + vsmowD*mD/mO16);
CEarray(1,2) = CEarray(1)*(CE18O/1000 + 1)*mO18*vsmowO18/mO16;
CEarray(1,3) = CEarray(1)*(CED/1000 + 1)*mD*vsmowD/mO16;
% convert from velocity (m/s) to da mass flux (kg/m2/s)
SST = 273.14+Tssc;  % Sea Surface Temperature (K)
rho = (SSP-9.8*1.225*h3)/(287.053*(SST-9.8*h3/1000)); % da density at h3
% Stull, 2000, also assuming dP/dz = standard SS value below h3
% molecular diffusion constants for 3 isotopologues (m2/s)
km0 = -2.777e-6 + 4.479e-8*SST + 1.656e-10*SST*SST;
```

```matlab
kmarray = [km0 km0/1.0285 km0/1.0251]; % Contains values of alpha
klo=(Kmax-km0)/h1; % K gradient in Low layer, same for all isotopologues
Ktop=km0*Ktop;       % convert K(h3) from ratio to value, m2/s
wa = wa*rho;         % vertical "velocity" in mass/m^2/s units

junkdz=h1*km0/Kmax; % zstar for 16
loplot = horzcat( linspace(0,2*junkdz,25),2.5*junkdz,...
                  linspace(3*junkdz,40*junkdz,37),40.5*junkdz,...
                  linspace(41*junkdz,h1,h1) );  % plot Lo (0 to h1)
mhplot = linspace(h1,h3,h3+1);  % plot every meter Mid & Hi (h1 to h3)

% "concentrations" (mass mixing ratios) of 3 isotopologues at SS (x=0)
delta18f = -2.0667 - 0.4156e3/SST + 1.137e6/SST/SST; % L-V fractionation
deltaDf = 52.612 - 76.248e3/SST +24.844e6/SST/SST;
alpha18 = exp(delta18f/1000);
alphaD=exp(deltaDf/1000);
d18O=(SSd18O+1000)/alpha18-1000;    % SS air delta value
dD=(SSdD+1000)/alphaD-1000;
ssdex=dD-8*d18O;
Sats = (((((0.638780966e-8*Tssc + 0.203886313e-5)*Tssc +.302246994e-3)...
    *Tssc + 0.0265027242)*Tssc +1.43053301)*Tssc + 44.3986062)...
    *Tssc + 611.176750;  % saturation vapor pressure (Pa) Flateau et al
Sats = Sats*0.622/(SSP - Sats);     % convert to kg vapor/kg da
c0array = zeros(1,3); % SS Boundary values of "concentrations" of 3 iso's
c0array(1) = Sats/(1+vsmowO18*mO18/mO16 + vsmowD*mD/mO16);
c0array(2) = c0array(1)*(d18O/1000 + 1)*mO18*vsmowO18/mO16;
c0array(3) = c0array(1)*(dD/1000 + 1)*mD*vsmowD/mO16;
c1array=zeros(1,3); % analytic c(h1) for i'th isotopologue
    % end of defining constants & other preliminary setup
    disp(' ');
    disp(' ');

% print adjustable parameters
disp(' ');
disp(' ');
disp('parameters');
disp('h1, h2 ,h3,     CE0, CE18O, CED');
fprintf('%d %d %d    %f %f %f \n', h1, h2 ,h3, CE0, CE18O, CED);
disp('Tssc, SSP, SSd18O, SSdD, wa, alpha, Kmax, Ktop');
fprintf('%d %d ', Tssc, SSP);
fprintf('%f %f %f %f %f %f \n\n', SSd18O, SSdD, wa, alpha, Kmax, Ktop);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Loop once for each isotopologue
for i = 1:3
    disp(' ');
    disp(['in main loop, for i = ' num2str(i) ])
    km = kmarray(i);
    c0 = c0array(i);
    CE = CEarray(i);
    Kmax=h1*klo+km; % changes with i
```

```matlab
        % initial grid in height -- defines mid and hi layers by
        % repeating height values at boundary h2.
        xinit = [linspace(h1,h2,(h2-h1) + 1) ...
                 linspace(h2,h3,(h3-h2) + 1)];
        c1init=0.5*c0; % initial guess for c1 (constant in analytic solution)
        solinit=bvpinit(xinit,@yinitfcn,c1init); % initial guess for solution

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % solve the differential equations
        options = bvpset('Stats','off','RelTol',1e-9);
        sol = bvp5c(@PBL_ODE,@PBL_ODE_BC,solinit, options);
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        c1array(i)=sol.parameters;
        disp(['km, Sats = ' num2str(km) ' ' num2str(Sats)]);
        disp(['C0 C1 CE = ' num2str(c0) '  ' num2str(c1array(i))...
                                        '  ' num2str(CE)]);
        dcdz0=(-c0 + c1array(i))*Kmax/(h1*log(km + Kmax)); % grad c at z=0
        % print mixing ratios and gradients at 0, h1, h2,h3
        disp('solutions for mixing ratios and gradients at [0 h1 h2 h3] are ');
        fprintf(' r       %.3e   %.3e   %.3e   %.3e\n', ...
            c0, deval(sol, [h1 h2 h3],1) );
        fprintf('dr/dz    %.3e   %.3e   %.3e   %.3e\n', ...
            dcdz0,deval(sol,[h1 h2 h3],2) );
        disp(' ');

        % create variables for use in plotting and printing
        rho0 = SSP/(287.053*SST); % da density at z=0
        switch i
            case 1  % H2O16
                concH2O16 = deval(sol,mhplot,1);
                flux16=-km*dcdz0*rho;
            case 2  % H2O18
                concH2O18 = deval(sol,mhplot,1);
                flux18=-km*dcdz0*rho;
            case 3 % D2O16
                concHDO16 = deval(sol,mhplot,1);
                flux2D=-km*dcdz0*rho;
        end

end % of looping through 3 isotopologues

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Post-processing, Plotting, and Printing
disp(' ');

% print surface fluxes
fprintf('Surface fluxes of 16 18 and 2D (kg/m^2/s):  %.3e  %.3e
%.3e\n',...
    flux16, flux18, flux2D);
evap=flux16+flux18+flux2D;
```

```
fluxd18=1000*((flux18/flux16)/(vsmowO18*mO18/mO16) - 1); % convert to delta
fluxd2D=1000*((flux2D/flux16)/(  vsmowD*mD  /mO16) - 1);
fluxdex = fluxd2D - 8*fluxd18;
fprintf('  flux delta 18 and D, and flux dex, (mmol/mol/m^2/s):  ');
fprintf('  %.3e  %.3e  %.3e\n', fluxd18, fluxd2D, fluxdex);
fprintf('evaporation rate, cm/y  %.1e\n',evap*100*3600*24*365.25/1000);
disp('  ');

% find variables at special height, z=0
c16_0=PBL_LO_ANALYTIC(1,0); % c16 concentration at z=0 (analytic)
c18_0=PBL_LO_ANALYTIC(2,0);
c2D_0=PBL_LO_ANALYTIC(3,0);
del18_0=((c18_0/c16_0)/(mO18*vsmowO18/mO16)-1)*1000; % delta-18 at z=0
delD_0=((c2D_0/c16_0)/(mD*vsmowD/mO16)-1)*1000;
dex_0 = delD_0 - 8*del18_0;

% find & print variables at special height, z=2
c16_2=PBL_LO_ANALYTIC(1,2); % c16 concentration at z=0 (analytic)
c18_2=PBL_LO_ANALYTIC(2,2);
c2D_2=PBL_LO_ANALYTIC(3,2);
fprintf('RH at z=2m is   %.3e\n',...
    (c16_2+c18_2+c2D_2)/Sats);
disp('  ');

% find & print variables at special height, hs
hs=15; % special height in lower layer, for printed output
c16hs=PBL_LO_ANALYTIC(1,hs); % c16 concentration at height hs (analytic)
c18hs=PBL_LO_ANALYTIC(2,hs);
c2Dhs=PBL_LO_ANALYTIC(3,hs);
del18hs=((c18hs/c16hs)/(mO18*vsmowO18/mO16)-1)*1000; % delta-18 at hs
delDhs=((c2Dhs/c16hs)/(mD*vsmowD/mO16)-1)*1000;
dexhs=delDhs-8*del18hs;
fprintf('at height %d m, dO18, dD, and dex are    %.2f   %.2f
%.2f\n',...
        hs, del18hs, delDhs, dexhs);
fprintf('          and    r, rsat, and RH are    %.3e   %.3e
%.3e\n',...
    c16hs+c18hs+c2Dhs, Sats, (c16hs+c18hs+c2Dhs)/Sats);

% Low layer concentrations (analytic)
LoconcH2O16=PBL_LO_ANALYTIC(1,loplot);
LoconcH2O18=PBL_LO_ANALYTIC(2,loplot);
LoconcHDO16=PBL_LO_ANALYTIC(3,loplot);

% delta values for z above h1
delta18 = (concH2O18./concH2O16/mO18*mO16/vsmowO18 -1)*1000;
deltaD = (concHDO16./concH2O16/mD*mO16/vsmowD -1)*1000;
dxs = deltaD-8*delta18;
Lodelta18 = (LoconcH2O18./LoconcH2O16/mO18*mO16/vsmowO18 -1)*1000;
LodeltaD = (LoconcHDO16./LoconcH2O16/mD*mO16/vsmowD -1)*1000;
Lodxs = LodeltaD-8*Lodelta18;

% print isotope ratio anomalies at heights hi
```

```matlab
disp(' ');
fprintf(' z      delta_O18    delta_D and dex :\n');
fprintf('%3d    %.4f    %.4f    %.4f\n',...
    0, del18_0, delD_0, dex_0);
hout=h1;
fprintf('%d    %.4f    %.4f    %.4f\n',...
    hout, delta18(hout+1), deltaD(hout+1), dxs(hout+1));
hout=h2;
fprintf('%d    %.4f    %.4f    %.4f\n',...
    hout, delta18(hout+1), deltaD(hout+1), dxs(hout+1));
hout=h3;
fprintf('%d    %.4f    %.4f    %.4f\n',...
    hout, delta18(hout+1), deltaD(hout+1), dxs(hout+1));
disp(' ');

% plot concentrations
figure(1)
subplot(1,3,1)
hold 'on'
plot(concH2O16,mhplot)
plot(LoconcH2O16,loplot)
title('H2O16')
ylabel('height, m')
xlabel('concentration')
%draw horizontal lines at layer boundaries
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')

subplot(1,3,2)
hold 'on'
plot(concH2O18,mhplot)
plot(LoconcH2O18,loplot)
title('H2O18')
ylabel('height, m')
xlabel('concentration')
%draw horizontal lines at layer boundaries
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')

subplot(1,3,3)
hold 'on'
plot(concHDO16,mhplot)
plot(LoconcHDO16,loplot)
title('HDO16')
ylabel('height, m')
xlabel('concentration')
%draw horizontal lines at layer boundaries
```

```matlab
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')
hold 'off'

% compute, print & plot delta values
figure(2)
delta18 = (concH2O18./concH2O16/mO18*mO16/vsmowO18 -1)*1000;
deltaD = (concHDO16./concH2O16/mD*mO16/vsmowD -1)*1000;
dxs = deltaD-8*delta18;
Lodelta18 = (LoconcH2O18./LoconcH2O16/mO18*mO16/vsmowO18 -1)*1000;
LodeltaD = (LoconcHDO16./LoconcH2O16/mD*mO16/vsmowD -1)*1000;
Lodxs = LodeltaD-8*Lodelta18;

LLo18=length(Lodelta18);
for iprt = 1:LLo18;
  fprintf('%.4e   %.4e\n', ...
      Lodelta18(1,iprt), LodeltaD(1,iprt));
end

subplot(1,3,1)
hold 'on'
plot(delta18,mhplot)
plot(Lodelta18,loplot)
xlabel('delta-O18')
ylabel('height, m')
%draw horizontal lines at layer boundaries
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')
hold 'off'

subplot(1,3,2)
hold 'on'
plot(deltaD,mhplot)
plot(LodeltaD,loplot)
xlabel('delta-D')
ylabel('height, m')
%axis([-91 -80 0 800])
%draw horizontal lines at layer boundaries
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')
hold 'off'

subplot(1,3,3)
hold 'on'
```

```matlab
plot(dxs,mhplot)
plot(Lodxs,loplot)
xlabel('D-excess')
ylabel('height, m')
%draw horizontal lines at layer boundaries
aaa = axis;
xmin = aaa(1);
xmax = aaa(2);
line([xmin xmax],[h1 h1],'Color','k')
line([xmin xmax],[h2 h2],'Color','k')
hold 'off'

% delta O18 vs delta D
figure(3)
hold 'on'
plot(delta18,deltaD,'k-');              % plot results, black
plot(Lodelta18,LodeltaD,'k-');
xmin = min(horzcat(Lodelta18, delta18));
xmax = max(horzcat(Lodelta18, delta18));
ymin = min(horzcat(LodeltaD, deltaD));
ymax = max(horzcat(LodeltaD, deltaD));
plot([xmin xmax],[8*xmin 8*xmax],'r-'); % plot MWL, red
plot(Lodelta18(1),LodeltaD(1),'ro');    % plot surface point, red circle
plot(delta18(h1),deltaD(h1),'bo');      % plot h1 values, blue circle
plot(delta18(h2),deltaD(h2),'bo');      % plot h2 values, blue circle
xlabel('delta-O18')
ylabel('delta-D')
legend('calculated','MWL','surface value','Location','SouthEast')
range18 = abs(xmin-xmax);
rangeD = abs(ymin-ymax);
axis([xmin-0.1*range18
      xmax+0.1*range18
      ymin - 0.1*rangeD
      ymax + 0.1*rangeD]);

% End of output section
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% internal functions to initialize and define BVP

function y = yinitfcn(~,region) % unused first argument was 'x'
    % initial guess: [concentration, gradient] set to [c0, -c0/h2]
    switch region
        case 1  % middle layer
            y = [c0;-c0/h2];
        case 2   % top layer
            y = [c0;-c0/h2];
    end % of cases
end % of function yinitfcn

function dydx = PBL_ODE(x,y,region,~)
% defines differential equations for the mid & hi layers
```

```matlab
    dydx=zeros(2,1);
    switch region
        case 1  % middle layer
            dydx = [y(2);
                    (wa*(x-h1)/(h2-h1)*y(2)-alpha*wa/(h2-h1)*(CE-y(1)))...
                    / (rho*(km+klo*h1))];
        case 2  % top layer
            dydx = [y(2);
                    ((km+klo*h1-Ktop)/(h3-h2)+wa/rho)*y(2) / ...
                    (km+klo*h1-(x-h2)*(klo*h1-Ktop)/(h3-h2))];
    end % of cases
end % of function dydx

function res = PBL_ODE_BC(ybot,ytop, c1)
    % defines BC at h1 and h3, and at internal interface at h2
    cph1=(c1-c0)*Kmax/(h1*(km+Kmax)*log((km+Kmax)/km)); % dc/dz(h1)
    res = [ybot(1,1)-c1          % c = c1 at z = h1
           ybot(2,1)-cph1        % dc/dz is continuous at h1
           ytop(1,1)-ybot(1,2)   % c is continuous at h2
           ytop(2,1)-ybot(2,2)   % dc/dz is continuous at h2
           ytop(2,2)];           % dc/dz = 0 at h3
end % of function PBL_ODE_B

function ylo = PBL_LO_ANALYTIC(i,x)
    % evaluate the analytic solution in the low layer, height x<=h1
    c0 = c0array(i);
    c1=c1array(i);
%   fprintf(' in ylo c1 = %.3e\n',c1)
    Km=kmarray(i);
    ylo=(c0*log(h1*(Km+Kmax))+ ...
        (c1-c0)*log(h1*Km+Kmax*x)-c1*log(h1*Km))...
        /log((Km+Kmax)/Km);
end % of function PBL_LO_ANALYTIC
end % of "main" function
```