```
%This .m file aims to calculate the binding constants of
given ligands
%and protein based on generalized equation in the text
%The inputs are ligands' molecular mass and concentrations
of protein and
%ligands

clc;
clear;
format long;

xlsFile     = 'xlsfile';
pk_th       = .13;
chg_th      = .04;
re_step     = 0.000001;      % predefined step
h_range     = 70;            % half range for searching
peak

ligand_mass = [222 258 305 330 365 383];
pl_con = [20 5 5 5 5 10 5.5];
temp = 0;
chg_inc = 1;
test = 0;
data = xlsread(xlsFile);
len = length(data(:,1));

figure(1);
plot(data(:,1),data(:,2),'k');

[M,I] = max(data(:,2));
mz_terminate = data(len,1);
in_terminate = chg_th*M;

no_ligand = length(ligand_mass);
pl_mass = ones(1,no_ligand+1);
result = zeros(no_ligand+1,3,8);
f_result = zeros(no_ligand+1,1);
kd_result = zeros(no_ligand,1);
charge_cnt = 1;
h_peak = zeros(40,2);

h_peak = extr(data(:,1), data(:,2), I, re_step, pk_th);
[Mi,Ii] = max(h_peak(:,2));
```

```
charge = charg(h_peak(Ii,1),h_peak(Ii-1,1));
mass = mztomass(h_peak(Ii,1),charge);

% Assume that mass is protein (host)
temp_mass = mass+ligand_mass(no_ligand);
lower_bound = findb(data(:,1),masstomz(temp_mass,charge)-
.6);
upper_bound =
findb(data(:,1),masstomz(temp_mass,charge)+.6);
[Mii,Iii] = max(data(lower_bound:upper_bound,2));

if(Mii > Mi*pk_th)
    residual_mz = nmz(data(lower_bound:upper_bound,1),
data(lower_bound:upper_bound,2), Iii, re_step);
    if(residual_mz ~= 0)
        test =
masschk(data(I,1),residual_mz,0,ligand_mass(no_ligand),char
ge);
    end
end
cnt = 0;

if(test == 0)
    for cnt=1:no_ligand
        temp_mass = mass+ligand_mass(no_ligand)-
ligand_mass(cnt)
        lower_bound =
findb(data(:,1),masstomz(temp_mass,charge)-.6);
        upper_bound =
findb(data(:,1),masstomz(temp_mass,charge)+.6);
        [Mii,Iii] = max(data(lower_bound:upper_bound,2));

        if(Mii > Mi*pk_th)
            residual_mz =
nmz(data(lower_bound:upper_bound,1),
data(lower_bound:upper_bound,2), Iii, re_step);
            if(residual_mz ~= 0)
                test =
masschk(data(I,1),residual_mz,ligand_mass(cnt),ligand_mass(
no_ligand),charge);
            end

            if(test == 1)
                p_mass = mass - ligand_mass(cnt);
```

```
                break;
            end
        end
    end
end

% The actual relative position for the highest peak in the
group of same charge
cnt
if(cnt == 0)
    temp = round(mass);
    pl_mass = [temp temp+ligand_mass];
else
    temp = round(p_mass);
    pl_mass = [temp temp+ligand_mass];
end
hp_l = cnt+1;


% Calculate all values from lowest charge to highest charge
for i=-4:1:4
    lower_bound = findb(data(:,1),masstomz(mass,charge+i)-
.6);
    upper_bound =
findb(data(:,1),masstomz(mass,charge+i)+.6);

    if (lower_bound == 0) || (upper_bound == 0)  ||
(lower_bound > len)
        Mii = 0;
    else
        [Mii,Iii] = max(data(lower_bound:upper_bound,2));

        if(Mii > in_terminate)
            Iii = find(data(:,2) == Mii);
            h_peak = extr(data(:,1), data(:,2), Iii,
re_step, pk_th);
            [Mi,Ii] = max(h_peak(:,2));
            mz_n    = h_peak(Ii,1);
            area_n  = sum(h_peak(:,2));

            for j=1:no_ligand+1
                if(j ~= hp_l)
```

```
                    result(j,:,charge_cnt) =
proces(pl_mass(j),data(:,1),data(:,2),charge+i,re_step,pk_t
h);
                else
                    result(j,:,charge_cnt) = [mz_n area_n
charge+i];
                end
            end
            charge_cnt = charge_cnt + 1;
        end
    end
end

for i=1:charge_cnt-1
    f_result = f_result + result(:,2,i);
end

% Calculate all sum
sum_all = sum(f_result);
p_con = f_result(1)*pl_con(1)/sum_all;

% Calculate Kd
for i=1:no_ligand
    kd_result(i) = (f_result(1)*pl_con(i+1))/f_result(i+1)
- p_con;
end


% Extract data for peak
function output = proces(mass, mz, in, charge, re_step,
threshold)

Mii = 0;
Iii = 0;
h_peak = zeros(40,2);

lower_bound = findb(mz,masstomz(mass,charge)-.6);
upper_bound = findb(mz,masstomz(mass,charge)+.6);
[Mii,Iii] = max(in(lower_bound:upper_bound));
Iii = find(in == Mii);
h_peak = extr(mz, in, Iii, re_step, threshold);
[Mi,Ii] = max(h_peak(:,2));
output = [h_peak(Ii,1) sum(h_peak(:,2)) charge];
end
```

```
% Estimate the next location
function output = npeakc(mz, mass, charge, step)
output = round(abs(mz-((mass+charge)/charge)));
end

% Check mass
function output = masschk(mz1, mz2, m1, m2, charge)

a = charge*(mz2 - mz1);
b = m2 - m1;

if abs(a-b) < 3
    output = 1;
else
    output = 0;
end
end


% Calculate nominal mz for center peak
function output = nmz(mz, in, pos, re_step)
mz_b = ones(1,10);
in_b = ones(1,10);
temp_length = length(mz);

if(pos>temp_length-10 || pos<10)
    output = 0;
else
    for i=1:10
        if (in(pos-i) < in(pos-i+1))
            mz_b(i) = mz(pos-i);
            in_b(i) = in(pos-i);
        else
            break;
        end
    end
    mz_f = [fliplr(mz_b(1:i-1)) mz(pos)];
    in_f = [fliplr(in_b(1:i-1)) in(pos)];

    for i=1:10
        if (in(pos+i) < in(pos+i-1))
            mz_b(i) = mz(pos+i);
            in_b(i) = in(pos+i);
```

```
        else
            break;
        end
    end
    mz_f = [mz_f mz_b(1:i-1)];
    in_f = [in_f in_b(1:i-1)];

    output = tmean(mz_f, in_f, re_step);
    output = output(1);
end
end


% Estimate the next location
function output = npeak(mz_data, charge, m_real, m_npeak,
step)

mn = m_real + m_npeak;
output = round((masstomz(mn,charge)-mz_data)/step);
end


% Calculate mass
function output = masstomz(mass,charge)

output = mass/charge+1;
end

% Calculate mass
function output = mztomass(mz,charge)

output = charge*(mz-1);
end


% Calculate charge
function output = charg(peak1,peak2)

charge = 0;
if peak1 > peak2
    charge = round(1/(peak1 - peak2));
else
    charge = round(1/(peak2 - peak1));
end
```

```
output = charge;
end


% Detect the highest peak, form then extract upto the first
lowest from left and right
% in = intensity
% mz = m/z
% pos = position of the highest peak
function output = extr(mz, in, pos, re_step, threshold)

half_way = 12;
cnt = 1;
center = 0;

%% proceed from center to left
for i=1:15
    dl(i).mzl = 0;
    dl(i).inl = 0;
end

means = ones(half_way,2);
mz_b = ones(1,10);
in_b = ones(1,10);

for i=1:10
    if (in(pos-i) < in(pos-i+1))
        mz_b(i) = mz(pos-i);
        in_b(i) = in(pos-i);
    else
        break;
    end
end
mz_c = [fliplr(mz_b(1:i-1)) mz(pos)];
in_c = [fliplr(in_b(1:i-1)) in(pos)];
i_l = i;

for i=1:10
    if (in(pos+i) < in(pos+i-1))
        mz_b(i) = mz(pos+i);
        in_b(i) = in(pos+i);
    else
        break;
    end
```

```
end
mz_c = [mz_c mz_b(1:i-1)];
in_c = [in_c in_b(1:i-1)];
i_r = i;

center = tmean(mz_c, in_c, re_step);

pos_l  = pos-i_l+2;

% selecting 15 subpeaks
for i=1:half_way
    for j=1:10
        if (in(pos_l-j) < in(pos_l-j-1))
            dl(i).mzl(j) = mz(pos_l-j);
            dl(i).inl(j) = in(pos_l-j);
        else
            break;
        end
    end
    cnt = j-1;
    pos_l = pos_l-cnt;

    for j=1:10
        if (in(pos_l-j) > in(pos_l-j-1))
            dl(i).mzl(j+cnt) = mz(pos_l-j);
            dl(i).inl(j+cnt) = in(pos_l-j);
        else
            dl(i).mzl(j+cnt) = mz(pos_l-j);
            dl(i).inl(j+cnt) = in(pos_l-j);
            break;
        end
    end
    pos_l = pos_l-j+1;
end


% calculate means
cnt = -1;
for i=1:half_way
    means(i,:) = tmean(fliplr(dl(i).mzl),
fliplr(dl(i).inl), re_step);
    cnt = cnt + 1;

    if(i == half_way)
```

```
        break;
    end

    % discard low values
    if(i>1) && (means(i,2) < center(2)*threshold)
        break;
    end
end
output = [flipud(means(1:cnt,1)), flipud(means(1:cnt,2))];

%% proceed from center to right
for i=1:half_way
    dl(i).mzr = 0;
    dl(i).inr = 0;
end


output = [output;center];

pos_r  = pos+i_r-2;

% selecting 15 subpeaks
for i=1:half_way
    for j=1:10
        if (in(pos_r+j) < in(pos_r+j+1))
            dl(i).mzr(j) = mz(pos_r+j);
            dl(i).inr(j) = in(pos_r+j);
        else
            break;
        end
    end
    cnt = j-1;
    pos_r = pos_r+cnt;

    for j=1:10
        if (in(pos_r+j) > in(pos_r+j+1))
            dl(i).mzr(j+cnt) = mz(pos_r+j);
            dl(i).inr(j+cnt) = in(pos_r+j);
        else
            dl(i).mzr(j+cnt) = mz(pos_r+j);
            dl(i).inr(j+cnt) = in(pos_r+j);
            break;
        end
    end
```

```
    pos_r = pos_r+j-1;
end

% calculate means
cnt = -1;
for i=1:half_way
    means(i,:) = tmean(dl(i).mzr, dl(i).inr, re_step);
    cnt = cnt + 1;

    if(i==half_way)
        break;
    end

    % discard low values
    if(i>1) && (means(i,2) < center(2)*threshold)
        break;
    end
end

output = [output;means(1:cnt,1), means(1:cnt,2)];
end


% Finding the mean of input distribution function
function output = tmean(mz, in, re_step)

inlen = length(mz);
n_mz = mz(1)-re_step:re_step:mz(inlen)+re_step;

n_in = spline(mz,in,n_mz);

[m_in,index] = max(n_in);

output = [n_mz(index) m_in];
end


function output = findb(array, num)

output = 0;
len = length(array);
for i=1:len
    if (num < array(i))
        output = i;
        break;
```

```
    end
end
end
```