```
function [reduced_section] = FOCuS_Section(model_in,biom,rxn_max_in,bound_glu, ↙
bound_O2,bound_ATPM,bound_biom)

%%%%%%%%%%%%% inputs to indexing program %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% model_in          COBRA stoichiometric model in standard structure format
% biom              Biomass reaction
% rxn_max_in        Target reaction
% bound_glu         input flux value for glucose
% bound_O2          input flux value for oxygen
% bound_ATPM        input flux value for maintenance ATP

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

clc
tic

global model
global rxn_max

model = model_in;
rxn_max = rxn_max_in;

%%%%%%%%%%%%%%%%%%%%%%%%% FOCuS Parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
para=[100 0.8];
n=para(1);
p=para(2);
m=14;
pop=n;

gen=0;
max_gen=1;

global d
d=5; %Number of genes to be deleted

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

rxns_size = size(model.rxns);
rxns_count = rxns_size(1);

for i=1:1:rxns_count

    rxn_ind = strcmp(model.rxns(i),rxn_max);
    if rxn_ind > 0
        rxn_max_no = i;
    end

end
clear i
disp('Target reaction number =')
disp(rxn_max_no);


for i=1:1:rxns_count
```

```
    biomass_ind = strcmp(model.rxns(i),biom);
    if biomass_ind > 0
        biomass_max_no = i;
    end

end
disp('Biomass reaction number =')
disp(biomass_max_no);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% READ INDEX ↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ind=xlsread('C:\Users\....\MATLAB\cobra\test_codes\FOCUS\acetate_index.xlsx' )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ↙
%%%

global ind_cont
global bound_cont
global pind

ind_cont = 1:length(ind);


section=0;
sectionmax= 30 %10 %20; %15
ro=round((length(ind)/sectionmax));

half=[];

while (section<sectionmax)


    if section==0
        ind_cont1= 1:(ro*1);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end


    if section==1
        ind_cont1= (ro+1):(ro*2);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);

    end

    if section==2
        ind_cont1= ((ro*2)+1):(ro*3);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==3
        ind_cont1= ((ro*3)+1):(ro*4);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==4
        ind_cont1= ((ro*4)+1):(ro*5);  %length(ind);%
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==5
        ind_cont1= ((ro*5)+1):(ro*6);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==6
        ind_cont1= ((ro*6)+1):(ro*7);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==7
        ind_cont1= ((ro*7)+1):(ro*8);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==8
        ind_cont1= ((ro*8)+1):(ro*9);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==9
        ind_cont1= ((ro*9)+1):(ro*10);%length(ind);%
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==10
        ind_cont1= ((ro*10)+1):(ro*11);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==11
        ind_cont1= ((ro*11)+1):(ro*12);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==12
        ind_cont1= ((ro*12)+1):(ro*13);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==13
        ind_cont1= ((ro*13)+1):(ro*14);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==14
        ind_cont1= ((ro*14)+1):(ro*15);%length(ind);%
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

     if section==15
        ind_cont1= ((ro*15)+1):(ro*16);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==16
        ind_cont1= ((ro*16)+1):(ro*17);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==17
        ind_cont1= ((ro*17)+1):(ro*18);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==18
        ind_cont1= ((ro*18)+1):(ro*19);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==19
        ind_cont1= ((ro*19)+1):(ro*20);%length(ind);%
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

     if section==20
        ind_cont1= ((ro*20)+1):(ro*21);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==21
        ind_cont1= ((ro*21)+1):(ro*22);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==22
        ind_cont1= ((ro*22)+1):(ro*23);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==13
        ind_cont1= ((ro*13)+1):(ro*14);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==24
        ind_cont1= ((ro*24)+1):(ro*25);%length(ind);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

     if section==25
        ind_cont1= ((ro*25)+1):(ro*26);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==26
```

```
        ind_cont1= ((ro*26)+1):(ro*27);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==27
        ind_cont1= ((ro*27)+1):(ro*28);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==28
        ind_cont1= ((ro*28)+1):(ro*29);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end

    if section==29
        ind_cont1= ((ro*29)+1):length(ind);
        bound_cont=ind_cont1;
        Lb = min(ind_cont1)*ones(1,d);
        Ub = max(ind_cont1)*ones(1,d);
    end



    pind = ind(ind_cont);

    bound_cont;

↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ↙
%%

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%   select gene knocks ↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    model = changeRxnBounds(model, 'EX_glc(e)',bound_glu,'l');           %glucose
    model = changeRxnBounds(model, 'EX_o2(e)',bound_O2,'l');        %oxygen uptake
    model = changeRxnBounds(model,biom,bound_biom, 'l');
    model = changeRxnBounds(model, 'ATPM',bound_ATPM,'l');

    Sol = zeros(pop,d);


    for i=1:1:pop
        Sol(i,:)=Lb+(Ub-Lb).*rand(1,d);
        Sol(i,:)=round(Sol(i,:));
        s_1 = unique(Sol(i,:));
        len = length(s_1);

        if len == (d-1)
```

```
            s_2 = setdiff(bound_cont, s_1);
            s_3 = randsample(s_2,1);
            Sol(i,:) = [s_1 s_3];
        end

        if len == (d-2)
            s_2 = setdiff(bound_cont, s_1);
            s_3 = randsample(s_2,2);
            Sol(i,:) = [s_1 s_3];
        end

        if len == (d-3)
            s_2 = setdiff(bound_cont, s_1);
            s_3 = randsample(s_2,3);
            Sol(i,:) = [s_1 s_3];
        end

        clear len s_1 s_2 s_3
    end


    for i=1:1:pop
        Sol(i,:);
        Fitness(i)=Fun(Sol(i,:));
    end


↙
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ↙
%%
    % %Initialize the population/solutions

    [Fitness, indf] = sort(Fitness,'descend');
    best_pop = Sol(indf(1:end),:);
    firstsol = [best_pop Fitness'];

    fmax=Fitness(1);
    best=Sol(indf(1),:);

    clear indf

    S1=zeros(n,d);
    S2=zeros(n,d);

    bestu=[];


    while(gen<max_gen)

        for i=1:1:n
            S1(i,:) = randsample(ind_cont, d);
            S1(i,:)=simplebounds(S1(i,:),Lb,Ub);
        end

        Fitness1 = zeros(1,n);
        for i=1:1:n
            S2(i,:) = randsample(ind_cont, d);
            S2(i,:)=simplebounds(S2(i,:),Lb,Ub);
        end

        Fitness2 = zeros(1,n);
        Fitness3 = zeros(1,n);


        if rand>p


            for i=1:1:n
                L=Levy(d);
                dS=L.*(Sol(i,:)-best);
                S1(i,:)=Sol(i,:)+dS;
                S1(i,:)=simplebounds(S1(i,:),Lb,Ub);
                Fitness1(i)=Fun(S1(i,:));
            end

        else
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

            for i=1:1:pop
                Sol(i,:);
                Fitness(i)=Fun(Sol(i,:));
            end

            [Fitness, indf] = sort(Fitness,'descend');

            clone = Sol(indf(1:m),:);
            clear indf

            clone_pop = zeros(n,d);
            for i=1:1:18
                clone_pop(i,:)=clone(1,:);
            end
            for i=19:1:34
                clone_pop(i,:)=clone(2,:);
            end
            for i=35:1:48
                clone_pop(i,:)=clone(3,:);
            end
            for i=49:1:60
                clone_pop(i,:)=clone(4,:);
            end
            for i=61:1:70
                clone_pop(i,:)=clone(5,:);
            end
            for i=71:1:78
                clone_pop(i,:)=clone(6,:);
            end
            for i=79:1:84
                clone_pop(i,:)=clone(7,:);
            end
            for i=85:1:88
                clone_pop(i,:)=clone(8,:);
            end
            for i=89:1:90
                clone_pop(i,:)=clone(9,:);
            end
            for i=91:1:92
                clone_pop(i,:)=clone(10,:);
            end
            for i=93:1:94
                clone_pop(i,:)=clone(11,:);
            end
            for i=95:1:96
                clone_pop(i,:)=clone(12,:);
            end
            for i=97:1:98
                clone_pop(i,:)=clone(13,:);
            end
            for i=99:1:100
                clone_pop(i,:)=clone(14,:);
            end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            clone_pop;



            for j=1:1:n
                epsilon=rand;
                JK=randperm(n);
                S2(j,:)=clone_pop(j,:)+ (3*epsilon*(Sol(JK(1),:)-Sol(JK(2),:)));
                S2(j,:)=simplebounds(S2(j,:),Lb,Ub);
                Fitness2(j)=Fun(S2(j,:));
            end

        end

        S3 = [Sol; S1; S2];
        size(S3);


        for k=1:1:length(S3)
            Fitness3(k)=Fun(S3(k,:));
        end

        [Fitness3, ind3] = sort(Fitness3,'descend');
        best_S3 = S3(ind3(1:end),:);
        clear ind3

        size(best_S3);
        Sol_lev_clon = [best_S3 Fitness3'];
        size(Sol_lev_clon);
        Sol=best_S3(1:n,:);

        best_1=Sol(1,:);
        if Sol_lev_clon(1,end)>fmax
            best=best_1;
        end

        bestSol=pind(best);
        model.rxns(bestSol)

        fmax_gen(gen+1)=Sol_lev_clon(1,end);

        %%%%%%%%%%%% META-DYNAMIC STEP  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        if round(gen)==gen
            temp=best;
            bestu=[bestu; temp fmax_gen(end)];
            clear temp
            if gen>0
                sz_bestu=size(bestu);
                k_1=bestu((sz_bestu(1)-1),end);
                k_2=bestu(sz_bestu(1),end);
                Toler = k_1-k_2
                if Toler >= 0
                    for i=1:1:pop
                        Sol(i,:)=randsample(ind_cont,d);
                        Sol(i,:)=simplebounds(Sol(i,:),Lb,Ub);
                    end

                end
            end

        end
        %%%%%%%%%%%% META-DYNAMICS STEP ENDS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


        gen= gen+1;
    end

    [~,idx]=unique(Sol_lev_clon,'rows');
    half_tmp=Sol_lev_clon(sort(idx),:);
    half_tmp=half_tmp(1:20,:);

    half =[half; half_tmp];
    clear half_tmp

    fmax_gen=fmax_gen';
    fmax = fmax_gen(end,1);
    bestu;
    bestSol=pind(best);
    model.rxns(bestSol);
    section = section+1
    clear Fitness Sol bestu best

    if (0 < section)&& (section<sectionmax)
        gen=0;
    end
```

```
end

final_pop=half;

net_out=final_pop(:,1:(end-1));
net_out=unique(net_out);
net_out=pind(net_out);
net_out=net_out';
reduced_section = net_out;

xlswrite('C:\Users\...\MATLAB\cobra\FOCuS\acetate_iter1.xlsx' , reduced_section)

toc

end


function s=simplebounds(s,Lb,Ub)

global bound_cont
global d

ns_tmp=s;
I=ns_tmp<Lb;
ns_tmp(I)=Lb(I);
J=ns_tmp>Ub;
ns_tmp(J)=Ub(J);

s=ns_tmp;
s=round(s);

s_1=unique(s);
len = length(s_1);

if len == (d-1)
    s_2 = setdiff(bound_cont, s_1);
    s_3 = randsample(s_2,1);
    s = [s_1 s_3];
end

if len == (d-2)
    s_2 = setdiff(bound_cont, s_1);
    s_3 = randsample(s_2,2);
    s = [s_1 s_3];
end

if len == (d-3)
    s_2 = setdiff(bound_cont, s_1);
    s_3 = randsample(s_2,3);
    s = [s_1 s_3];
end

end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function L =Levy(d)
beta=3/2;
tmpdiv=(gamma((1+beta)/2)*beta*2^((beta-1)/2))^(1/beta);
sigma = (gamma(1+beta)*sin(pi*beta/2))/tmpdiv;
u=randn(1,d)*sigma;
v=randn(1,d);
step=u./abs(v).^(1/beta);
L=0.01*step;
end


function fun_eval = Fun(p1)

global model
global pind
global rxn_max

p2=pind(p1);
deletions = model.rxns(p2);

nDel = length(deletions);
model_KO = model;
targetRxn=rxn_max;
toler = 1e-7;

for i = 1:nDel
    model_KO = changeRxnBounds(model_KO,deletions{i},0, 'b');
end

AfterKO = optimizeCbModel(model_KO);
growthRate = AfterKO.f;

if (AfterKO.stat == 1)
    round_off = floor(AfterKO.f/toler)*toler;
    model_KO = changeRxnBounds(model_KO,model_KO.rxns(model_KO.c==1),round_off, 'l');
    model_KO = changeObjective(model_KO,targetRxn);
    Max_Sol = optimizeCbModel(model_KO, 'max');
    Min_Sol = optimizeCbModel(model_KO, 'min');
    Prod_Max = Max_Sol.f;
    Prod_Min = Min_Sol.f;

else
    Prod_Max = 0;
    Prod_Min = 0;
end

fun_eval=Prod_Max;

clear model_KO Max_Sol Min_Sol AfterKO p1 p2 growthRate targxnind
end
```