ANEXOS

Anexo 1. Implementación modelo de instalación híbrida en bus común.

```
function Pss = ss_model(Ess,Pls,alphas,num_ss,A_matrix)

% Pss ò Ess_(k+1)==> Potencia en los elementos de
almacenamiento
% Ess ==> Energía en los elementos de almacenamiento
% Pls ==> Potencia input al sistema ya sea load y/o source,
matriz 1 x 1
% num_ss ==> número de elementos de almacenamiento
% alphas ==> matrix n x 1, flujos de potencia desde Pls

%%

% Model Parameters

eff_ls_2_ss = ones(num_ss,1); % Suponemos cte.
eff_ss_2_ls = ones(num_ss,1); % Suponemos cte.

if nargin >= 5
 A = A_matrix
else
    A = zeros(num_ss); % No es cte.
end

b =[] ; % matriz 1*m que toma las eff correspondientes

for i=1:1:num_ss
    b_i = (1-signo(Pls))*eff_ls_2_ss(i) +
signo(Pls)*eff_ss_2_ls(i);
    b = [b;b_i];
end

B = alphas .* b; % No es cte

% Space of State
Pss = A*Ess + B*Pls;

end
```

Anexo 2. Implementación modelo de batería.

Anexo 2.1 Modelo de batería

```
function [i_batt,SoC,U_batt,U_ocv,float] =
batt_rc(Pss,r,U_batt,U_ocv,Cnom,SoH,float,T,SoC,SoC_Plomo)

%% Modelo RC eq. para batería.
% Pss ==> potencia de entrada a la batería
% r ==> resistencia de la batería
% c ==> condensador de la batería
% T ==> time sampling de simulacion
% E_batt ==> Energía acumulada
% U_batt ==> tensión en bornes de la batería
% U_ocv ==> tensión en circuito abierto de la batt, tensión
del
% float ==> estado de flotacion si es 1
% E_max ==> Energía máxima de la batería
% aC ==> Decremento de la capacitancia del modelo
%   condensador

%% Últ. Revisión
%   22/06/2019
%

%% Source

num_vasos = 24; %celdas en serie

r_batt = r;

U_float = 2.23*num_vasos;
U_min = 1.85*num_vasos;

i_input = Pss/U_batt; % suponemos que todas las celdas
tiene un comportamiento homogéneo


if (float==0)

%     i_batt = Pss/U_batt;             % corriente de
entrada
%     dU_ocv = i_batt/c_batt*T;        % incremento
%     U_ocv = U_ocv + dU_ocv;          % tensión de ocv
%     U_batt = r_batt*i_batt + U_ocv; % tensión de batt

    i_batt = Pss/U_batt;             % corriente de entrada
```

```
    SoC = SoC + i_batt*T/(Cnom*SoH/100)*100;
    SoC = min(SoC, 100);
    SoC = max(SoC, 0);
    U_ocv =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC)*num_vasos;
    U_batt = U_ocv + i_batt*r_batt;

else
    % Tensión en bornes controlada en función de la
flotación
    if float == 1
        U_batt = U_float;
    elseif float == -1
        U_batt = U_min;
    end

%   i_batt = (U_batt-U_ocv)/r_batt; % Corriente de entrada
%     dU_ocv = i_batt/c_batt*T;       % incremento
%     U_ocv = U_ocv + dU_ocv;         % tensión de ocv

    i_batt = (U_batt-U_ocv)/r_batt; % Corriente de entrada
    SoC = SoC + i_batt*T/(Cnom*SoH/100)*100;
    SoC = min(SoC, 100);
    SoC = max(SoC, 0);
    U_ocv =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC)*num_vasos;

end



% Control de flotación
    % Actualizo tensión de flotación

    % Compruebo si he de entrar en flotación o no
    % float = 1, flotación U max
    % float = -1, flotación U min
    if ((float == 0) && (U_batt >= U_float))
        float = 1;
    elseif ((float == 0) && (U_batt < U_min))
        float = -1;
    elseif (float == 1 && i_input <= i_batt)
        float = 0;
    elseif (float == -1 && i_input >= i_batt)
        float = 0;
    end

% Tomar el valor más próximo
% SoC_Table_aux = abs(SoC_Plomo(:,2)*num_vasos - U_ocv);
```

```
% minim = min(SoC_Table_aux);
% min_idx = find(SoC_Table_aux==minim);
% SoC_value_1 = SoC_Plomo(min_idx,1);
% soc = SoC_value_1;

% Más lento
% soc =
interp1(SoC_Plomo(:,2),SoC_Plomo(:,1),U_ocv/num_vasos);
% E_batt = soc*E_max/100;


end
```

Anexo 2.2 Modelo de envejecimiento

```
function [SoH_new] = envejecimiento(P,Temp,SoH)
%% Modelo de envejecimiento.
% aSoH ==> decremento del SoH del modelo del batería (como
valor negativo)
% P ==> Potencia absorbida o dada por la bateria
% Temp ==> Temperatura ambiente

%% Últ. Revisión
%   17/07/2019
%


%% Dod_Table ==> [DoDs,Num_Ciclos]
% DoD = 111.7043*exp(-6.67*10^-5 * num_ciclo)
% k2 = 111.7043;
% k1 = -6.67e-5;
% nums = linspace(1560,100000,1000);
% DoDs =  k2*exp(nums.*k1);
% DoD_Table = [DoDs', nums'];
%

%% aSoH por Temp
Arrhenius_Table = [-1.5853e-06,-3.9633e-06,-7.9267e-06,-
1.189e-5,-4.75e-5,-14.27e-5,-25.54e-5;2,5,10,15,25,35,45];
%[aSoH[%];Temp]

Temps = Arrhenius_Table(2,:);
aSoHs = Arrhenius_Table(1,:);

% Valores No Grabados Son Negativos
if Temp <= 0
 aSoH_Temp = 0;
else
```

```
 aSoH_Temp =interp1(Temps,aSoHs,Temp);
end


%% aSoH por P
k_P = -0.0000010;
aSoH_P = k_P*abs(P);

%% aSoH total
aSoH = aSoH_P + aSoH_Temp ; % F

SoH_new = SoH + aSoH;

end % function
```

Anexo 3. Implementación de políticas de control

Anexo 3.1 Política de selección de batería con mejor SoH

```
function alphas = max_soh_politics(Pls,SoCs,SoHs,Us,Is,Fs)
%% Gastar antes las baterías en mejor estado
% Pls ==> Potencia de input al sistema
% SoCs ==> Vector de SoC para las n baterias
% SoHs ==> Vector de SoH para las n baterias
% Us ==> Vector de tensión en bornes de las n baterías
% Is ==> Vector de corrientes de las n baterías
% Fs ==> Vector de estado de flotación
% Últ. Revision == 05/08/2019
%% Code

SoHs_aux = SoHs;
Pls_copy = Pls;
alpha_f= 0;
aux = 0;
sum_alphas = 0;

[n_batt, ~] = size(SoHs_aux);
alphas = zeros(n_batt,1);

searching = 1;
iter = 0;
iter_max = n_batt - 1;


SoHs_aux(SoHs_aux<=0) = -500;

while ((searching == 1) &&(iter<=iter_max))
```

```
% [1] Tomar batt con min. SoC
max_SoH = max(SoHs_aux);
max_idx = find(SoHs_aux==max_SoH);

% [3] PLS? y Mirar SoC
if Pls_copy > 0
    I = Is(max_idx);
    F = Fs(max_idx);
    U = Us(max_idx);
    SOC = SoCs(max_idx);

    if F == 1
        P = U*I;
        if round(Pls_copy,3) == 0.000
            alpha_f = 0;
            searching = 0;
        else
            alpha_f = P/Pls_copy;
        end
        alpha_f = min(alpha_f,1-sum_alphas);
        alpha_f = max(alpha_f,0);
        alphas(max_idx,:) = alpha_f;
        sum_alphas = sum_alphas + alpha_f;
        sum_alphas = min(sum_alphas,1);
        Pls_copy = Pls_copy - Pls_copy*alpha_f;
        SoHs_aux(max_idx) = -500; % quito esa batt
        if round(Pls_copy,3) == 0.000
            searching = 0;
        end
    elseif   F == -1
        P = U*I;
        if round(Pls_copy,3) == 0.000
            alpha_f = 0;
            searching = 0;
        else
            alpha_f = P/Pls_copy;
        end
        alpha_f = min(alpha_f,1-sum_alphas);
        alpha_f = max(alpha_f,0);
        alphas(max_idx,:) = alpha_f;
        sum_alphas = sum_alphas + alpha_f;
        sum_alphas = min(sum_alphas,1);
        Pls_copy = Pls_copy - Pls_copy*alpha_f;
        SoHs_aux(max_idx) = -500; % quito esa batt
        if round(Pls_copy,3) == 0.000
            searching = 0;
        end
    else
        alphas(max_idx,:) = 1 - sum_alphas;
```

```
            searching = 0;
        end
    else
        I = Is(max_idx);
        F = Fs(max_idx);
        U = Us(max_idx);
        SOC = SoCs(max_idx);

        if F == -1
            P = U*I;
            if round(Pls_copy,3) == 0.000
                alpha_f = 0;
                searching = 0;
            else
                alpha_f = P/Pls_copy;
            end
            alpha_f = min(alpha_f,1-sum_alphas);
            alpha_f = max(alpha_f,0);
            alphas(max_idx,:) = alpha_f;
            sum_alphas = sum_alphas + alpha_f;
            sum_alphas = min(sum_alphas,1);
            Pls_copy = Pls_copy - Pls_copy*alpha_f;
            SoHs_aux(max_idx) = -500; % quito esa batt
            if round(Pls_copy,3) == 0.000
                searching = 0;
            end

        elseif   F == 1
            P = U*I;
            if round(Pls_copy,3) == 0.000
                alpha_f = 0;
                searching = 0;
            else
                alpha_f = P/Pls_copy;
            end
            alpha_f = min(alpha_f,1-sum_alphas);
            alpha_f = max(alpha_f,0);
            alphas(max_idx,:) = alpha_f;
            sum_alphas = sum_alphas + alpha_f;
            sum_alphas = min(sum_alphas,1);
            Pls_copy = Pls_copy - Pls_copy*alpha_f;
            SoHs_aux(max_idx) = -500; % quito esa batt
            if round(Pls_copy,3) == 0.000
                searching = 0;
            end
        else
            alphas(max_idx,:) = 1 - sum_alphas;
            searching = 0;
        end
```

```
    end

    % Si todas las baterías han sido
    % analizadas,fin del control
    if(sum(SoHs_aux)==-500*5)
        searching = 0;
    end

    iter = iter + 1;

end

end
```

Anexo 3.2 Política de selección de batería con peor SoH

```
function alphas = min_soh_politics(Pls,SoCs,SoHs,Us,Is,Fs)
%% Gastar antes las baterías en peor estado
% Pls ==> Potencia de input al sistema
% SoCs ==> Vector de SoC para las n baterias
% SoHs ==> Vector de SoH para las n baterias
% Us ==> Vector de tensión en bornes de las n baterías
% Is ==> Vector de corrientes de las n baterías
% Fs ==> Vector de estado de flotación
% Últ. Revision == 05/08/2019
%% Code

SoHs_aux = SoHs;
Pls_copy = Pls;
alpha_f= 0;
aux = 0;
sum_alphas = 0;

[n_batt, ~] = size(SoHs_aux);
alphas = zeros(n_batt,1);

searching = 1;
iter = 0;
iter_max = n_batt - 1;


SoHs_aux(SoHs_aux<=0) = 500;

while ((searching == 1) &&(iter<=iter_max))
    % [1] Tomar batt con min. SoC
    min_SoH = min(SoHs_aux);
    min_idx = find(SoHs_aux==min_SoH);
```

```
% [3] PLS? y Mirar SoC
if Pls_copy >= 0
    I = Is(min_idx);
    F = Fs(min_idx);
    U = Us(min_idx);
    SOC = SoCs(min_idx);

    if F == 1
        P = U*I;
        if round(Pls_copy,3) == 0.000
            alpha_f = 0;
            searching = 0;
        else
            alpha_f = P/Pls_copy;
        end
        alpha_f = min(alpha_f,1-sum_alphas);
        alpha_f = max(alpha_f,0);
        alphas(min_idx,:) = alpha_f;
        sum_alphas = sum_alphas + alpha_f;
        sum_alphas = min(sum_alphas,1);
        Pls_copy = Pls_copy - Pls_copy*alpha_f;
        SoHs_aux(min_idx) = 500; % quito esa batt
        if round(Pls_copy,3) == 0.000
            searching = 0;
        end
    elseif   F == -1
        P = U*I;
        if round(Pls_copy,3) == 0.000
            alpha_f = 0;
            searching = 0;
        else
            alpha_f = P/Pls_copy;
        end
        alpha_f = min(alpha_f,1-sum_alphas);
        alpha_f = max(alpha_f,0);
        alphas(min_idx,:) = alpha_f;
        sum_alphas = sum_alphas + alpha_f;
        sum_alphas = min(sum_alphas,1);
        Pls_copy = Pls_copy - Pls_copy*alpha_f;
        SoHs_aux(min_idx) = 500; % quito esa batt
        if round(Pls_copy,3) == 0.000
            searching = 0;
        end
    else
        alphas(min_idx,:) = 1 - sum_alphas;
        searching = 0;
    end
else
    I = Is(min_idx);
```

```
        F = Fs(min_idx);
        U = Us(min_idx);
        SOC = SoCs(min_idx);

        if F == -1
            P = U*I;

            if round(Pls_copy,3) == 0.000
                alpha_f = 0;
                searching = 0;
            else
                alpha_f = P/Pls_copy;
            end
            alpha_f = min(alpha_f,1-sum_alphas);
            alpha_f = max(alpha_f,0);
            alphas(min_idx,:) = alpha_f;
            sum_alphas = sum_alphas + alpha_f;
            sum_alphas = min(sum_alphas,1);
            Pls_copy = Pls_copy - Pls_copy*alpha_f;
            SoHs_aux(min_idx) = 500; % quito esa batt
            if round(Pls_copy,3) == 0.000
                searching = 0;
            end

        elseif   F == 1
            P = U*I;

            if round(Pls_copy,3) == 0.000
                alpha_f = 0;
                searching = 0;
            else
                alpha_f = P/Pls_copy;
            end
            alpha_f = min(alpha_f,1-sum_alphas);
            alpha_f = max(alpha_f,0);
            alphas(min_idx,:) = alpha_f;
            sum_alphas = sum_alphas + alpha_f;
            sum_alphas = min(sum_alphas,1);
            Pls_copy = Pls_copy - Pls_copy*alpha_f;
            SoHs_aux(min_idx) = 500; % quito esa batt
            if round(Pls_copy,3) == 0.000
                searching = 0;
            end
        else
            alphas(min_idx,:) = 1 - sum_alphas;
            searching = 0;
        end
    end
```

```
    % Si todas las baterías han sido
    % analizadas,fin del control
    if(sum(SoHs_aux)==500*5)
        searching = 0;
    end

    iter = iter + 1;

end

end
```

Anexo 4. Implementación de políticas de reemplazo

Anexo 4.1 Política de reemplazo de batería cuando su estado de salud es del 0%

Esta es la política implementada para la política de carga basada en selección de la batería con peor estado de salud.

```
    min_soh_value = min(SoHs);
    if round(min_soh_value,2) == 0.00
        soh_0_idx = find(SoHs == min_soh_value);
        switch soh_0_idx
           case 1
               SoHs(soh_0_idx) = 100;
               SoH_1 = 100;
               Cnom_1 = 920*3600;
               C1 = c_1;
           case 2
               SoHs(soh_0_idx) = 100;
               SoH_2 = 100;
               Cnom_2 = 920*3600;
           case 3
               SoHs(soh_0_idx) = 100;
               SoH_3 = 100;
               Cnom_3 = 920*3600;
           case 4
               SoHs(soh_0_idx) = 100;
               SoH_4 = 100;
               Cnom_4 = 920*3600;
           case 5
               SoHs(soh_0_idx) = 100;
               Cnom_1 = 920*3600;
               SoH_5 = 100;
```

```
        otherwise
             %pass
             disp('')
    end
end
```

Anexo 4.2 Política de reemplazo del pack de baterías cuando el estado de salud total es del 0%

Esta es la política implementada para la política de carga basada en selección de la batería con mejor estado de salud.

```
sum_SoHs = sum(SoHs);
if round(sum_SoHs,2) == 0.00
            SoHs(1) = 100;
            SoH1 = 100;
            Cnom_1 = 920*3600;
            SoHs(2) = 100;
            SoH2 = 100;
            Cnom_2 = 920*3600;

            SoHs(3) = 100;
            SoH3 = 100;
            Cnom_3 = 920*3600;

            SoHs(4) = 100;
            SoH4 = 100;
            Cnom_4 = 920*3600;

            SoHs(5) = 100;
            Cnom_5 = 920*3600;
            SoH5 = 100;
end
```

Anexo 5. Simulador de instalación de almacenamiento híbrida

```
%% SIMULACIÓN PASO VARIABLE
% Ultimate  ==> 20/07/19
% Comentarios:
%   - alphas ctes

clear all
close all


%% LOAD DATA
workspace = pwd;
sigena_name = 'Sigena_definitivo.mat';
sigena_name2 = 'INTEG_Sigena_recons.mat';
path_sigena = strcat(workspace,'\',sigena_name);
path_sigena2 = strcat(workspace,'\',sigena_name2);
load(path_sigena);
load(path_sigena2);

date_aux = data.Sigena.RADIA.time';
date_aux2 = integ_reconstruida.fecha';

fecha_inicio = datenum('01/01/2013 00:00','dd/mm/yyyy
HH:MM');
fecha_fin = datenum('31/12/2016 23:45','dd/mm/yyyy HH:MM');
% fecha_fin = datenum('01/02/2013 23:45','dd/mm/yyyy
HH:MM');

idx_inicio = find(date_aux == fecha_inicio);
idx_fin = find(date_aux == fecha_fin);
idx_inicio2 = find(date_aux2 == fecha_inicio);
idx_fin2 = find(date_aux2 == fecha_fin);

i_ls = integ_reconstruida.valor(idx_inicio2:idx_fin2)' ;
% i_ls = zeros(1,length(i_ls));

% doblo perfil
i_ls = [i_ls;i_ls];
i_ls(isnan(i_ls)) = 0;


t_ls = data.Sigena.INTEG.time(idx_inicio:idx_fin)' ;
diff_years = fecha_fin-fecha_inicio;
t_ls = [t_ls,t_ls+diff_years];

temps = data.Sigena.TEMPI.signals(idx_inicio:idx_fin)' ;
```

```
temps = [temps,temps];
[~, cols] = size(t_ls);

% La cargamos solo una vez.
load('SoC_Plomo.mat')
load('C_fitted_v.mat')


%% SIMULATION
Cnom = 920*3600;
num_vasos = 24;
C_ocv = @(ocv) 1/(feval(C_fitted,ocv/num_vasos)*num_vasos);
% Función Cbulk - OCV

% SIMULATION PARAMS
% N°1 Battery
% U_ocv_1 = 53; % V OCV
% U_batt_1 = 53; % V Bornes
SoC_1 = 95;
r_1 = 100e-3; % ohm
Cnom_1 = 920*3600; % Capacidad máxima inicial[A*s ==
Culombios]
SoH_1 = 100; % SoH inicial
c_1 = Cnom_1*SoH_1/100; % F
U_batt_1 =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC_1)*num_vasos;
U_ocv_1 = U_batt_1;

% N°2 Battery
% U_ocv_2 = 53; % V OCV
% U_batt_2 = 53; % V Bornes
SoC_2 = 95;
r_2 = 100e-3; % ohm
Cnom_2 = 920*3600; % Capacidad máxima inicial
SoH_2 = 99; % SoH inicial
c_2 = Cnom_2*SoH_2/100; % F
U_batt_2 =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC_2)*num_vasos;
U_ocv_2 = U_batt_2;


% N°3 Battery
% U_ocv_3 = 53; % V OCV
% U_batt_3 = 53; % V Bornes
SoC_3 = 95;
r_3 = 100e-3; % ohm
Cnom_3 = 920*3600; % Capacidad máxima inicial
SoH_3 = 98; % SoH inicial
c_3 = Cnom_3*SoH_3/100; % F
```

```
U_batt_3 =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC_3)*num_vasos;
U_ocv_3 = U_batt_3;


% N°4 Battery
% U_ocv_4 = 53; % V OCV
% U_batt_4 = 53; % V Bornes
SoC_4 = 95;
r_4 = 100e-3; % ohm
Cnom_4 = 920*3600; % Capacidad máxima inicial
SoH_4 = 97; % SoH inicial
c_4 =  Cnom_4*SoH_4/100; % F
U_batt_4 =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC_4)*num_vasos;
U_ocv_4 = U_batt_4;


% N°5 Battery
% U_ocv_5 = 53; % V OCV
% U_batt_5 = 53; % V Bornes
SoC_5 = 95;
r_5 = 100e-3; % ohm
Cnom_5 = 920*3600; % Capacidad máxima inicial
SoH_5 = 96; % SoH inicial
c_5 = Cnom_5*SoH_5/100; % F
U_batt_5 =
interp1(SoC_Plomo(:,1),SoC_Plomo(:,2),SoC_5)*num_vasos;
U_ocv_5 = U_batt_5;


%flotacion flag
f1 = 0;
f2 = 0;
f3 = 0;
f4 = 0;
f5 = 0;

[i1,SoC_1,U_batt_1,U_ocv_1,f1] =
batt_rc(0,r_1,U_batt_1,U_ocv_1,Cnom_1,SoH_1,0,0,SoC_1,SoC_P
lomo);
[i2,SoC_2,U_batt_2,U_ocv_2,f2] =
batt_rc(0,r_2,U_batt_2,U_ocv_2,Cnom_2,SoH_2,0,0,SoC_2,SoC_P
lomo);
[i3,SoC_3,U_batt_3,U_ocv_3,f3] =
batt_rc(0,r_3,U_batt_3,U_ocv_3,Cnom_3,SoH_3,0,0,SoC_3,SoC_P
lomo);
```

```
[i4,SoC_4,U_batt_4,U_ocv_4,f4] =
batt_rc(0,r_4,U_batt_4,U_ocv_4,Cnom_4,SoH_4,0,0,SoC_4,SoC_P
lomo);
[i5,SoC_5,U_batt_5,U_ocv_5,f5] =
batt_rc(0,r_5,U_batt_5,U_ocv_5,Cnom_5,SoH_5,0,0,SoC_5,SoC_P
lomo);

SoCs = [SoC_1;SoC_2;SoC_3;SoC_4;SoC_5];
SoHs = [SoH_1;SoH_2;SoH_3;SoH_4;SoH_5];
U_ss = [U_batt_1;U_batt_2;U_batt_3;U_batt_4;U_batt_5];
I_ss = [i1;i2;i3;i4;i5];
F_ss = [f1;f2;f3;f4;f5];

num_ss = 5;
alphas = [0;0;0;0;0];
% T = 1;
t_final = cols;


flag_end = 0;

% PLOT MATRIX
resta = [];
ks = [];
tss_s = [];

%% LOOP

% Paso variable
% tss = 1 ; % T_Sample
k = 1; % instante actual
% tol_max = 5e-3; % tolerancia máx.
% tol_min = 0; % tolerancia mín.
% tss_max = 20;
% tss_min = 1;

% Paso fijo
tss = 900;


fin_sim = 0;

plot_pss        = zeros(5,1000000);
plot_ess_rc     = zeros(5,1000000);
plot_u_rc       = zeros(5,1000000);
plot_uocv_rc    = zeros(5,1000000);
plot_pls        = zeros(5,1000000);
plot_t          = zeros(5,1000000);
plot_soc        = zeros(5,1000000);
```

```
plot_C          = zeros(5,1000000);
plot_soh        = zeros(5,1000000);
plot_is         = zeros(5,1000000);
time_s          = zeros(1,1000000);

plot_a = [];

Caidas_State = [0;0;0;0;0];
vector_de_caidas = [0;0;0;0;0];


while ((k < t_final) && (fin_sim==0))

    % get inputs, Pls

    temp = temps(k); % Temperatura
    % Potencia de consumo de la carga
    carga = 10; %A
    v_bus = 48; %V
    % Balance de potencia en el bus de continua
    Pls = (i_ls(k)- carga)*v_bus; %W


    % politics
    alphas =
max_soh_politics(Pls,SoCs,SoHs,U_ss,I_ss,F_ss);
%     alphas =
min_soh_politics(Pls,SoCs,SoHs,U_ss,I_ss,F_ss);

    % System model
%     Pss = ss_model(E_ss,Pls,alphas,num_ss);
%     alphas = [0 0 0 0 1];
    Pss = Pls * alphas;

    % RC_BATT Model
    [SoH_1] = envejecimiento(Pss(1),temp,SoH_1);
    [i1,SoC_1,U_batt_1,U_ocv_1,f1] =
batt_rc(Pss(1),r_1,U_batt_1,U_ocv_1,Cnom_1,SoH_1,f1,tss,SoC
_1,SoC_Plomo);

    [SoH_2] = envejecimiento(Pss(2),temp,SoH_2);
    [i2,SoC_2,U_batt_2,U_ocv_2,f2] =
batt_rc(Pss(2),r_2,U_batt_2,U_ocv_2,Cnom_2,SoH_2,f2,tss,SoC
_2,SoC_Plomo);

    [SoH_3] = envejecimiento(Pss(3),temp,SoH_3);
    [i3,SoC_3,U_batt_3,U_ocv_3,f3] =
batt_rc(Pss(3),r_3,U_batt_3,U_ocv_3,Cnom_3,SoH_3,f3,tss,SoC
_3,SoC_Plomo);
```

```
    [SoH_4] = envejecimiento(Pss(4),temp,SoH_4);
    [i4,SoC_4,U_batt_4,U_ocv_4,f4] =
batt_rc(Pss(4),r_4,U_batt_4,U_ocv_4,Cnom_4,SoH_4,f4,tss,SoC
_4,SoC_Plomo);

    [SoH_5] = envejecimiento(Pss(5),temp,SoH_5);
    [i5,SoC_5,U_batt_5,U_ocv_5,f5] =
batt_rc(Pss(5),r_5,U_batt_5,U_ocv_5,Cnom_5,SoH_5,f5,tss,SoC
_5,SoC_Plomo);



    U_ss = [U_batt_1;U_batt_2;U_batt_3;U_batt_4;U_batt_5];
    U_ocv = [U_ocv_1;U_ocv_2;U_ocv_3;U_ocv_4;U_ocv_5];
    SoCs = [SoC_1;SoC_2;SoC_3;SoC_4;SoC_5];
    SoHs = [SoH_1;SoH_2;SoH_3;SoH_4;SoH_5];
    I_ss = [i1;i2;i3;i4;i5];
    F_ss = [f1;f2;f3;f4;f5];
    Cs =
([0.2*Cnom_1;0.2*Cnom_2;0.2*Cnom_3;0.2*Cnom_4;0.2*Cnom_5].*
SoHs/100) +
[0.8*Cnom_1;0.8*Cnom_2;0.8*Cnom_3;0.8*Cnom_4;0.8*Cnom_5];

    [vector_de_caidas,state] =
num_caidas(U_ss,Caidas_State, vector_de_caidas);



    % get data to plot
    plot_pss(:,k) = Pss;
    plot_u_rc(:,k) = U_ss;
    plot_uocv_rc(:,k) = U_ocv;
    plot_pls(:,k) = Pls;
    plot_t(:,k) = k;
    plot_soc(:,k) = SoCs;
    plot_soh(:,k) = SoHs;
    plot_is(:,k) = I_ss;
    plot_C(:,k) = Cs;
    time_s(k) = t_ls(k);
    plot_a = [plot_a,alphas];

    %End simulación si SoH de todas sea 0
%     soh_sum = sum(SoHs);
%     if soh_sum <= 0
%         fin_sim = 1;
%     end

%     recambio de batería al SoH == 0% por unas nuevas
```

```
    min_soh_value = min(SoHs);
    if round(min_soh_value,2) == 0.00
        soh_0_idx = find(SoHs == min_soh_value);
        switch soh_0_idx
           case 1
               SoHs(soh_0_idx) = 100;
               SoH_1 = 100;
               Cnom_1 = 920*3600;
               C1 = c_1;
           case 2
               SoHs(soh_0_idx) = 100;
               SoH_2 = 100;
               Cnom_2 = 920*3600;

           case 3
               SoHs(soh_0_idx) = 100;
               SoH_3 = 100;
               Cnom_3 = 920*3600;

           case 4
               SoHs(soh_0_idx) = 100;
               SoH_4 = 100;
               Cnom_4 = 920*3600;

           case 5
               SoHs(soh_0_idx) = 100;
               Cnom_1 = 920*3600;
               SoH_5 = 100;

           otherwise
                %pass
                disp('')
        end
    end

%     recambio de todo el pack, cuando hay 1 batería al SoH
== 0%
%     sum_SoHs = sum(SoHs);
%     if round(sum_SoHs,2) == 0.00
%                 SoHs(1) = 100;
%                 SoH1 = 100;
%                 Cnom_1 = 920*3600;
%
%                 SoHs(2) = 100;
%                 SoH2 = 100;
%                 Cnom_2 = 920*3600;
%
%
%                 SoHs(3) = 100;
```

```
%                 SoH3 = 100;
%                 Cnom_3 = 920*3600;
%
%
%                 SoHs(4) = 100;
%                 SoH4 = 100;
%                 Cnom_4 = 920*3600;
%
%
%                 SoHs(5) = 100;
%                 Cnom_5 = 920*3600;
%                 SoH5 = 100;
%
%     end
%

%     k = k + tss;

k = k+1;

    %% cálculo de tss
%     if k <= t_final
%     % 1. Evaluo modelo en el instante siguiente con tss
antiguo
%     temp = temps(k);
%     carga = 5; %A
%     v_bus = 48; %V
%     Pls = (i_ls(k)- carga)*v_bus; %W
%
%     % politics regulator
%     %[alphas,state] = priority_politics(Pls,U_ss,state);
%     alphas = [1];
%
%     % System model
%     Pss = ss_model(E_ss,Pls,alphas,num_ss);
%
%     % RC_BATT Model
%     [a_C1,a_E1] = envejecimiento(DoD1,temp);
%     aC1 = aC1 + a_C1;
%     e_max1 = e_max1 + a_E1;
%     [SoC1,E_batt_1,U_batt_1,U_ocv_1,dU_ocv_1_k1,f1,C1] =
batt_rc(Pss(1),r,c_1,U_ocv_1,U_batt_1,f1,e_max1,tss,aC1);
%     DoD1 =  calc_DoD(SoC1);
%     SoH1 = calc_SoH(c_1,C1);
%
%     U_ss = [U_batt_1];
%     U_ocv = [U_ocv_1];
%     E_ss = [E_batt_1];
%     SoC_ss = [SoC1];
```

```
%     C = [C1];
%
%
%     % 2. Cálculo abs(restas(derivadas))
%     resta_1 = abs(dU_ocv_1_k1/tss - dU_ocv_1_k/tss);
%     resta = [resta_1];
%
%     % 3. Compruebo tolerancias para dinámica crítica
(Modelo de Litio o Ultra-Capacidad ==> elemento 1 por
ejemplo)
%     if resta_1 > tol_max
%         new_tss = tss/2; % disminuyo tss
%     elseif resta_1 < tol_min
%         new_tss = tss*2;
%     else
%         new_tss = tss;
%     end
%
%     % 4. Compruebo que new_tss este entre [tss_min,
tss_max]
%     if new_tss > tss_max
%         new_tss = tss_max;
%     elseif new_tss < tss_min
%         new_tss = tss_min;
%     end
%
%     % 5. Actualizo k quitandoles tss y añadiendo el
new_tss
%     k = k - tss + round(new_tss,0);
%     tss = round(new_tss,0);
%     ks = [ks k];
%     tss_s = [tss_s tss];
%     end


end

% Resize data to plot
plot_pss = plot_pss(:,1:k-1);
plot_ess_rc = plot_ess_rc(:,1:k-1);
plot_u_rc = plot_u_rc(:,1:k-1);
plot_uocv_rc = plot_uocv_rc(:,1:k-1);
plot_pls = plot_pls(:,1:k-1);
plot_t = plot_t(:,1:k-1);
plot_soc = plot_soc(:,1:k-1);
plot_C = plot_C(:,1:k-1);
plot_soh = plot_soh(:,1:k-1);
plot_is = plot_is(:,1:k-1);
time_s = time_s(:,1:k-1);
```

```
%% PLOTS
colors = ['b','g','r','c','m'];

% fold = 'Figures_MaxSoH';
% fold =  'Figures_MinSoH';
% fold = 'Figures_RemplazoBatt_SoH_0\Figures_MaxSoH'
% fold = 'Figures_RemplazoBatt_SoH_0\Figures_MinSoH'
fold = 'Figures_RemplazoBatt_SoH_0
_Perfil_Doble\Figures_MaxSoH'
% fold = 'Figures_RemplazoBatt_SoH_0
_Perfil_Doble\Figures_MinSoH'
path_figures = strcat(workspace,'\',fold);
txt_file_path = strcat(path_figures,'\','results.txt');

% fileID = fopen(txt_file_path,'a');


%% Esto es muy ineficiente. Mejor de la siguiente manera

[rows,cols] = size(time_s);
% time_str = [];
%
% for idx=1:1:cols
%     str = datestr(time_s(idx),'dd-mmm-yyyy HH:MM:SS') ;
%     date = datetime(str);
%     time_str = [time_str, date];
% end

str = datestr(time_s,'dd-mmm-yyyy HH:MM:SS');
time_str = datetime(str);

%% PLOTS


figure()
grid on
for i =1:1:5
    hold on
    plot(time_str,plot_pss(i,:),colors(i),'LineWidth', 1);
end

legend('Pss_1 batt','Pss_2 batt','Pss_3 batt','Pss_4
batt','Pss_5 batt','Location', 'NorthWest')
tit = strcat('Pss Batt');
title(tit)
xlabel('time')
ylabel('power units')
savefig(strcat(path_figures,'\',tit,'.fig'))
```

```
% figure()
% grid on
% hold on
% plot(time_str,plot_ess_rc(1,:),'b','LineWidth', 1)
% plot(time_str,plot_ess_rc(2,:),'g','LineWidth', 1)
% plot(time_str,plot_ess_rc(3,:),'r','LineWidth', 1)
% plot(time_str,plot_ess_rc(4,:),'c','LineWidth', 1)
% plot(time_str,plot_ess_rc(5,:),'m','LineWidth', 1)
% legend('Ess_1 batt','Ess_2 batt','Ess_3 batt','Ess_4
batt','Ess_5 batt','Location', 'NorthWest')
% tit = 'Ess';
% title(tit)
% xlabel('time')
% savefig(strcat(path_figures,'\',tit,'.fig'))


figure()
grid on

for i =1:1:5
     hold  on
     plot(time_str,plot_u_rc(i,:),colors(i),'LineWidth',
1);
end
plot(time_str,44.44*ones(1,cols),'--r');
hold off
legend('U Batt1','U Batt2','U Batt3','U Batt4','U
Batt5','Location', 'NorthEast')
tit = strcat('U Batt');
title(tit)
xlabel('time')
ylabel('voltage units')
savefig(strcat(path_figures,'\',tit,'.fig'))

figure()
grid on
for i = 1:1:5
    hold on
    plot(time_str,plot_is(i,:),colors(i));
end
tit = strcat('I Batt');
legend('Iss_1 batt','Iss_2 batt','Iss_3 batt','Iss_4
batt','Iss_5 batt','Location', 'NorthWest')
title(tit)
xlabel('time')
ylabel('voltage units')
savefig(strcat(path_figures,'\',tit,'.fig'))
```

```
figure()
grid on
for i=1:1:5
    hold on
    plot(time_str,plot_uocv_rc(i,:),colors(i),'LineWidth',
1);
end
tit = strcat('U_o_c_v Batt');
title(tit)
legend('U_o_c_v Batt1','U_o_c_v Batt2','U_o_c_v
Batt3','U_o_c_v Batt4','U_o_c_v Batt5','Location',
'NorthEast')
xlabel('time')
ylabel('voltage units')
savefig(strcat(path_figures,'\',tit,'.fig'))

figure()
grid on
plot(time_str,plot_pls,'b','LineWidth', 1);
tit = 'Pls';
title(tit)
ylabel('power units')
xlabel('time')
savefig(strcat(path_figures,'\',tit,'.fig'))

figure()
grid on
grid on
hold on
plot(time_str,plot_soc(1,:),'b','LineWidth', 1);
plot(time_str,plot_soc(2,:),'g','LineWidth', 1);
plot(time_str,plot_soc(3,:),'r','LineWidth', 1);
plot(time_str,plot_soc(4,:),'c','LineWidth', 1);
plot(time_str,plot_soc(5,:),'m','LineWidth', 1);
legend('SoC1','SoC2','SoC3','SoC4','SoC5','Location',
'NorthEast')
tit = strcat('SoC Batt');
title(tit)
xlabel('time')
ylabel('%')
savefig(strcat(path_figures,'\',tit,'.fig'))


figure()
grid on
hold on
plot(time_str,plot_C(1,:)/3600,'b','LineWidth', 1);
plot(time_str,plot_C(2,:)/3600,'g','LineWidth', 1);
plot(time_str,plot_C(3,:)/3600,'r','LineWidth', 1);
```

```
plot(time_str,plot_C(4,:)/3600,'c','LineWidth', 1);
plot(time_str,plot_C(5,:)/3600,'m','LineWidth', 1);
hold off
legend('C_1','C_2','C_3','C_4','C_5','Location',
'NorthWest')
tit = 'Cap_B_a_t_t';
title(tit)
xlabel('time')
ylabel('Ah')
savefig(strcat(path_figures,'\',tit,'.fig'))

figure()
grid on
hold on
plot(time_str,plot_soh(1,:),'b','LineWidth', 1);
plot(time_str,plot_soh(2,:),'g','LineWidth', 1);
plot(time_str,plot_soh(3,:),'r','LineWidth', 1);
plot(time_str,plot_soh(4,:),'c','LineWidth', 1);
plot(time_str,plot_soh(5,:),'m','LineWidth', 1);
hold off
legend('SoH1','SoH2','SoH3','SoH4','SoH5','Location',
'NorthWest')
tit = 'SoH_B_a_t_t';
title(tit)
xlabel('time')
ylabel('%')
savefig(strcat(path_figures,'\',tit,'.fig'))

% figure()
% grid on
% hold on
% euros_init_1 = 0.2*plot_C(1,1)/3600; %€
% euros_vector_1 = 0.2*plot_C(1,:)/3600 - euros_init_1;
% euros_init_2 = 0.2*plot_C(2,1)/3600; %€
% euros_vector_2 =0.2*plot_C(2,:)/3600 - euros_init_2;
% euros_init_3 = 0.2*plot_C(3,1)/3600; %€
% euros_vector_3 = 0.2*plot_C(3,:)/3600 - euros_init_3;
% euros_init_4 = 0.2*plot_C(4,1)/3600; %€
% euros_vector_4 = 0.2*plot_C(4,:)/3600 - euros_init_4;
% euros_init_5 = 0.2*plot_C(5,1)/3600; %€
% euros_vector_5 =0.2*plot_C(5,:)/3600 - euros_init_5;
% plot(time_str,euros_vector_1,'b','LineWidth', 1);
% plot(time_str,euros_vector_2,'g','LineWidth', 1);
% plot(time_str,euros_vector_3,'r','LineWidth', 1);
% plot(time_str,euros_vector_4,'c','LineWidth', 1);
% plot(time_str,euros_vector_5,'m','LineWidth', 1);
% hold off
% legend('€_1','€_2','€_3','€_4','€_5','Location',
'NorthWest')
```

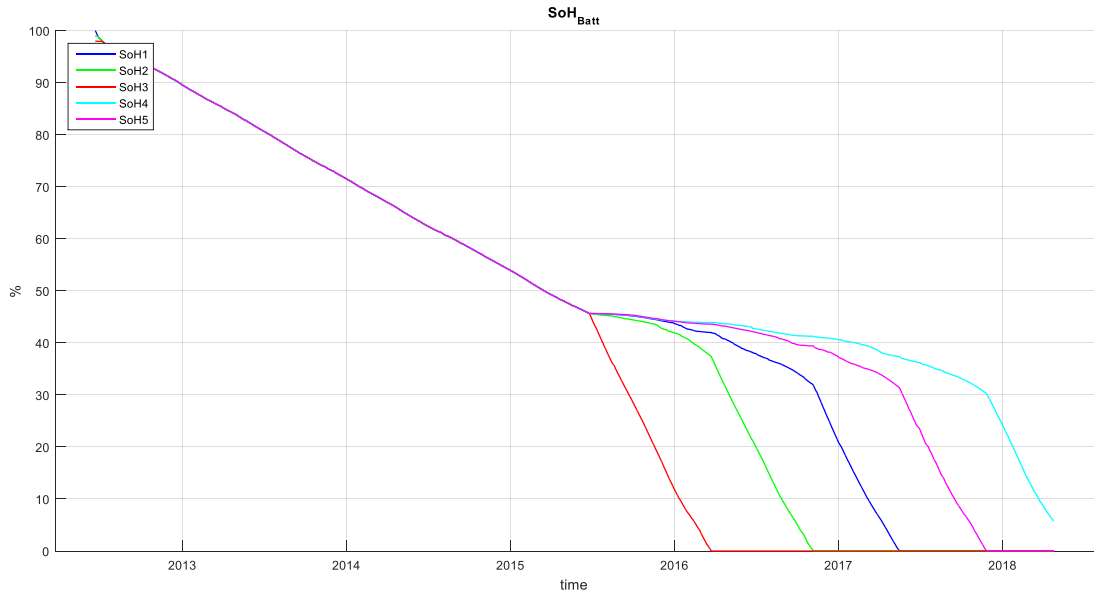
```
% tit = 'Amort_B_a_t_t';
% title(tit)
% xlabel('time')
% ylabel('€')
% savefig(strcat(path_figures,'\',tit,'.fig'))


% % caidas vs. batt
batts =  1:1:num_ss;
figure()
bar(batts,vector_de_caidas,0.2,'r')
for i = 1:1:num_ss
    if vector_de_caidas(i) ~ 0
     text(i-
0.05,vector_de_caidas(i)/2,int2str(vector_de_caidas(i)));
    end
end
grid on
tit = strcat('Num. Caídas Per Batt');
title(tit)
xlabel('Batería')
ylabel('Caídas')
savefig(strcat(path_figures,'\',tit,'.fig'))

% cap. Eq
c_eq = sum(plot_C/3600);
figure()
plot(time_str,c_eq,'r');
grid on
tit = strcat('Cap_E_q');
title(tit)
xlabel('time')
ylabel('Ah')
savefig(strcat(path_figures,'\',tit,'.fig'))

% fclose(fileID);
```

Anexo 6. Entrelazado de políticas de control

Se presenta los resultados de una implementación del entrelazado de políticas de carga, el instante de cambio de política de carga se obtiene en función del valor de capacidad equivalente de la instalación, cuando la capacidad de la instalación es menor a 4100 Ah (punto de intersección y cambio de las pendientes de degradación, visible en *Figura 16*) se pasa de una política de selección en base a mejor estado de salud a la de selección en base a peor estado de salud.

- Sin reemplazo de baterías

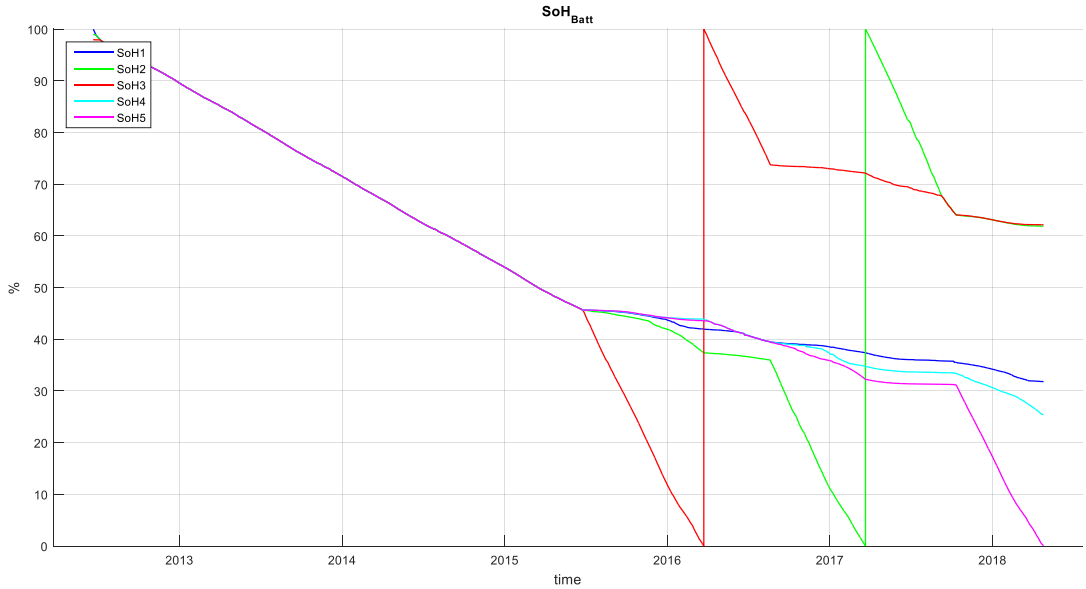
En la siguiente imagen exponemos la evolución del estado de salud de los elementos de almacenamiento de la instalación. Hasta mitad del año 2015, se observa un trabajo entrelazado de las baterías, sin embargo, a partir de este instante se observa como se empieza a individualizar el trabajo de las baterías. Exponemos también que al finalizar los ocho años de simulación aún queda un batería que no ha finalizado su ciclo de vida.



1. Estado de Salud para entrelazado de políticas de carga, sin reemplazo

- Con reemplazo de baterías

Al igual que en el apartado anterior, exponemos la evolución del estado de salud para las diferentes baterías de la instalación, añadiendo política de reemplazo de las baterías que llegan al fin de su ciclo de vida. Observamos reemplazo de únicamente dos baterías, por lo que ahorramos el coste de reemplazar tres baterías con respecto la política de selección en base a mejor estado de salud, y dos baterías en comparación con la otra política de carga planteada en este trabajo.



2. Estado de Salud para entrelazado de políticas de carga, con reemplazo

Este anexo refleja pequeñas mejoras en cuanto autonomía y coste de reemplazo, pero como se ha reflejado en *Conclusiones y Líneas de Trabajo Futuro*, podría ser una línea de trabajo futuro, donde habría que tener en cuenta diferentes aspectos para el cambio de política de carga, incluso tendría cabida la inclusión de nuevas políticas de carga, en adicción a las planteadas.