%TODO EL CODIGO

%Menú

clear all

clc

tic

IniVariables

Tiempo\_y\_Referencia

Datos\_Autos

Patrones\_Consumo

Costos

Trafos

PEVs\_Agregadores

MC=1;

for mc=1:MC;

Demanda

Escenarios\_PEVs

Restricciones

OptimizacionCoordinada

OptimizacionCoordinada1

OptimizacionSoloTrafoPrimario

OptimizacionCoordinaTrafos

VarAdicTarifas

end

LimitesdeTarifas

GraficaETOTAL

GraficasEstrategias

GraficaPDT

TiempoTotal=toc

%Inicializaicón de variables  
Simul=[];

Simul1=[];

PDT\_total=[];

CT=[];

CTTOU=[];

ETOTAL=[];

Emaxevol\_total=[];

Eminevol\_total=[];

Eoptevol\_total=[];

%Tiempo Y REFERENCIA

delta\_t=1;

% inv\_delta\_t =floor(1/delta\_t);

% inv\_delta\_t2 =1/delta\_t;

% tiempo=24\*(1/delta\_t);

% hinicial=12\*(1/delta\_t);

delta\_t\_inv=round(1/delta\_t);

tiempo = round( 24 / delta\_t); % Número de intervalos en 24 horas según delta\_t2

hinicial = round(12 / delta\_t);

%Datos Autos

Cap= [60;60;60;60;60;60;60;60;60;60;60;60;60;60;60];

Pcarga=7;

%Patrones consumo

hllegada=20\*(1/delta\_t);

hpartida=7\*(1/delta\_t);

km\_media=50;

SOCini=0.6;

Cllegada=2\*(1/delta\_t);

Cpartida=2\*(1/delta\_t);

Ckm=15;

CSOC=0.1;

ro=0.92;

%Costos

% Definición de los costos por unidad de tiempo

Pp = 0.138;

Pv = 0.109;

Pr = 0.058;

% Valor de delta\_t (ejemplo: cada 6 minutos que se colgaba)

%delta\_t = 0.16667;

% Número de intervalos en 24 horas según delta\_t

num\_intervals = round(24 / delta\_t);

% Inicialización del vector C

C = zeros(1, num\_intervals);

% Definición de los segmentos de tiempo y sus costos respectivos

time\_segments = [5, 4, 3, 8, 4] \* (1 / delta\_t);

costos = [Pv, Pp, Pv, Pr, Pp];

% Construcción del vector C

startIndex = 1;

for i = 1:length(time\_segments)

segment\_length = round(time\_segments(i));

C(startIndex:startIndex+segment\_length-1) = costos(i);

startIndex = startIndex + segment\_length;

end

%TRAFOS

%Original Ai=[150,150,200,100,100,250,250,100,100,100,100,100,200,100,100,100,150,150,150,150,150,150,630,630,100,100,100,200,250,250,250,100];

Ai=[100,150,100,100,100,150,200,50,100,100,100,100,200,100,100,100,150,100,100,100,150,150,600,500,100,100,100,200,200,200,200,100]; %-50MW

At=sum(Ai)\*0.8757;

%Se puede usar este codigo original

Ctrafo=Ai';

for a=1:tiempo-1

Ctrafo=[Ctrafo,Ai'];

end

%o sino

% Generación de Ctrafo

%Ctrafo = repmat(Ai', 1, tiempo);

%PEV Agregadores

VAR=3;

nag=32;

nPEV=VAR\*150;

nPEVag=VAR\*[4 4 5 2 2 8 8 2 2 2 2 2 5 2 2 2 4 4 4 4 4 4 18 18 2 2 2 5 8 6 9 2];

cPEV=zeros(nag+1,1);

for a=1:nag

cPEV(a+1)=sum(nPEVag(1:a));

end

%DEMANDA

Lbpu1 = 0.5 \* [0.80419, 0.70429, 0.64678, 0.61235, 0.59004, 0.58142, 0.59063, 0.62614, 0.58502, 0.63958, 0.69508, 0.70351, 0.68110, 0.64846, 0.65278, 0.64721, 0.64874, 0.67891, 0.76767, 0.96259, 1, 0.98774, 0.95284, 0.88752];

% Ajuste de índices para Lbpu12\_2

indices1 = mod(hinicial + 1 : tiempo + hinicial, length(Lbpu1));

indices1(indices1 == 0) = length(Lbpu1);

% Cambio de referencia a la hora inicial

Lbpu12 = Lbpu1(indices1);

% Asegurar que Lbpu12\_2 tenga exactamente tiempo elementos

Lbpu12 = Lbpu12(1:tiempo);

% Interpolación para ajustar a cualquier delta\_t

new\_time\_indices = linspace(1, 24, tiempo); % Índices de tiempo ajustados

Lbpu12 = interp1(1:length(Lbpu12), Lbpu12, new\_time\_indices, 'linear');

% Matriz de demandas de cada nag en cada intervalo de tiempo

Lbi12 = Ai' .\* Lbpu12;

% Suma total de la demanda del sistema

Lbt12 = sum(Lbi12);

AT = ones(1, tiempo) \* At; % Capacidad máximo del PDT en cada hora

Pdisp = max(Ctrafo - Lbi12, 0);

Pdisp\_total = max(AT - Lbt12, 0);

%ESCENARIOS PEVS

Hc=zeros(1,nPEV);

Hp=Hc;

Hni=Hc;

km\_c=Hc;

Cap\_auto=Hc;

SOC\_a=Hc;

SOC\_Da=Hc;

for k=1:nPEV;

Hc(k)=max(round(hllegada+((-1+(1+1)\*rand)\*Cllegada)),(hinicial+1));

Hp(k)=min(round(hpartida+((-1+(1+1)\*rand)\*Cpartida)),(tiempo-hinicial+1));

Hni(k)=(Hp(k)+tiempo)-Hc(k);

km\_c=max(0,km\_media+(Ckm\*randn));

Cap\_auto(k)=(Cap(round(rand\*14+1)));

SOC\_a(k)=Cap\_auto(k)\*(SOCini+(randn\*CSOC));

SOC\_Da(k)=min(SOC\_a(k)+(Hni(k)\*(Pcarga\*(1/delta\_t))),Cap\_auto(k));

end

%Restricciones

% CÁLCULO DE RESTRICCIONES DE ENERGÍA Y POTENCIA INDIVIDUALES

Eni\_min=zeros(nPEV,tiempo);

Eni\_max=zeros(nPEV,tiempo);

SOC\_ARR=zeros(nPEV,tiempo);

Emin\_evol=[];

Emax\_evol=[];

ESOC\_evol=[];

for k=1:nPEV;

Eni\_min(k,Hc(k)-hinicial)=SOC\_a(k); % Condicion inicial

Eni\_max(k,Hc(k)-hinicial)=SOC\_a(k); % Condicion inicial

for n=Hp(k)+hinicial-1:tiempo;

Eni\_min(k,n)=SOC\_Da(k); % Condicion final

Eni\_max(k,n)=SOC\_Da(k); % Condicion final

end

for n=1:Hni(k)-1;

Eni\_min(k,(Hp(k)+hinicial-n))=max((Eni\_min(k,(Hp(k)+hinicial-n+1))-(Pcarga/(1/delta\_t))),SOC\_a(k));

end

Evol\_min=Eni\_min(k,:);

Evol\_min(Eni\_min(k,:)<=0)=SOC\_a(1,k+1-1);

Emin\_evol=[Emin\_evol;Evol\_min];

for n=1:Hni(k)-1

Eni\_max(k,(Hc(k)-hinicial+n))=min((Eni\_max(k,(Hc(k)-hinicial+n-1))+(Pcarga/(1/delta\_t))),SOC\_Da(k));

end

Evol\_max=Eni\_max(k,:);

Evol\_max(Eni\_max(k,:)<=0)=SOC\_a(1,k+1-1);

Emax\_evol=[Emax\_evol;Evol\_max];

SOC\_ARR(k,Hc(k)-hinicial:end)=ones(1,tiempo+hinicial+1-Hc(k)).\*SOC\_a(k);

Evol\_SOC=SOC\_ARR(k,:);

Evol\_SOC(SOC\_ARR(k,:)<=0)=SOC\_a(1,k+1-1);

ESOC\_evol=[ESOC\_evol;Evol\_SOC];

end

Emax\_evol1=[Emax\_evol(1:nPEV,2:tiempo),60\*ones(nPEV,1)];

Emin\_evol1=[Emin\_evol(1:nPEV,2:tiempo),60\*ones(nPEV,1)];

Pmax=zeros(nPEV,tiempo);

for k=1:nPEV;

for n=Hc(k)-hinicial:Hp(k)+hinicial

Pmax(k,n)=Pcarga;

end

end

% CÁLCULO DE RESTRICCIONES DE ENERGÍA y POTENCIA POR AGREGADOR

for a=1:nag

SumEmin([a],[1:tiempo])=sum(Eni\_min([cPEV(a)+1:cPEV(a+1)],[1:tiempo]));

end

for a=1:nag

SumEmax([a],[1:tiempo])=sum(Eni\_max([cPEV(a)+1:cPEV(a+1)],[1:tiempo]));

end

for a=1:nag

SumSOC\_A([a],[1:tiempo])=sum(SOC\_ARR([cPEV(a)+1:cPEV(a+1)],[1:tiempo]));

end

SumSOC\_total=sum(SumSOC\_A);

SumEmin\_total=sum(SumEmin)-SumSOC\_total;

SumEmax\_total=sum(SumEmax)-SumSOC\_total;

SumPmax1=zeros(nag,tiempo);

for a=1:nag

SumPmax1([a],[1:tiempo])=sum(Pmax([cPEV(a)+1:cPEV(a+1)],[1:tiempo]));

end

SumPmax=min(SumPmax1,Pdisp);

%Optcoordinada

C2=C+(0.58/(tiempo)/10)\*(1:(tiempo));

A3=[];

A4=[];

A=[];

Atrafo=[];

A3i=-ones(1,tiempo);

A4i=ones(1,tiempo);

for a=1:nag

A3=blkdiag(A3,A3i);

A4=blkdiag(A4,A4i);

Atrafo=[Atrafo,eye(tiempo)];

end

A3=[A3,zeros(nag,tiempo)+1];

A4=[A4,zeros(nag,tiempo)];

Atrafo=[Atrafo,-eye(tiempo)];

ub=[];

for a=1:nag

ub=[ub,SumPmax(a,1:tiempo)];

end

ub=ub';

ub(tiempo\*(nag+1),1)=0;

lb=zeros(nag\*tiempo+tiempo,1);

for a=1:nag

SOC\_aAG(a)=sum(SOC\_a(cPEV(a)+1:cPEV(a+1)));

end

for a=1:nag

SOC\_DaAG(a)=sum(SOC\_Da(cPEV(a)+1:cPEV(a+1)));

end

B3=[];

for a=1:nag

B3(a)=(sum(SOC\_DaAG(a))-sum(SOC\_aAG(a)));

end

B3=B3';

B4=-B3;

Btrafo=max(At\*ones(tiempo,1)-Lbt12',0);

A=[A3;A4;Atrafo];

B=[B3;B4;Btrafo];

x0=zeros(1,tiempo\*(nag+1));

CTotal=[];

for a=1:nag

CTotal=[CTotal,C2];

end

CTotal=[CTotal,1000\*ones(1,tiempo)];

fun = @(x)x\*CTotal';

options = optimoptions('fmincon', 'Display','iter','Algorithm','sqp', 'MaxFunEvals', 1000000, 'MaxIter', 5000, 'Display', 'off');

sol=fmincon(fun,x0,A,B,[],[],lb,ub,[],options);

Ppref=[];

Prueba=[];

PruebaLb=[];

resultadoscoord=[];

for a=1:nag

Ppref=[Ppref;sol(1+(a-1)\*tiempo:tiempo+(a-1)\*tiempo)];

end

for a=1:nag

Prueba1=Ppref(a,:);

Prueba2=Lbi12(a,:);

Prueba=[Prueba;Prueba1([hinicial/(delta\_t\_inv)+1:tiempo/(delta\_t\_inv)]),Prueba1([1:hinicial/(delta\_t\_inv)])];

PruebaLb=[PruebaLb;[Prueba2([hinicial/(delta\_t\_inv)+1:tiempo/(delta\_t\_inv)]),Prueba2([1:hinicial/(delta\_t\_inv)])]];

end

resultadosCoord=Prueba+PruebaLb;

%Optcoordinada 1

C2=C+(0.58/(tiempo)/10)\*(1:(tiempo)); % Para carga temprana

A1=[];

A2=[];

A=[];

%(1/delta\_t)

Atrafo=[];

% En comparación con Optcoordinada solo se cambia A3 por A1 y A4 por A2

A1=[tril(ones(tiempo,tiempo),0),zeros(tiempo,tiempo)]; %crea una matriz inferior diagonal de 1

A2=[tril(-ones(tiempo,tiempo),0),zeros(tiempo,tiempo)]; %crea una matriz inferior diagonal de -1

Atrafo=[eye(tiempo),-eye(tiempo)]; %De 24x792 a 24x48

Ppref1=[];

ub=[];

%SumPmax\_total=min(SumPmax1(a,:),(ones(1,tiempo)\*Ai(a)-Lbi12(a,:)));

%%Cuando está FV

SumPmax\_total=(sum(SumPmax1)); %suma por columnas lo obtenido en SumPmax1

ub=SumPmax\_total'; %Cambia de 792x1 a 24x1

ub=[ub;At\*ones(tiempo,1)]; %Le agrega la restricción de máximo del transformador para que tenga la misma dimensión que x0 (48x1)

lb=zeros(tiempo+tiempo,1); % Le sumo 1, para que tenga la misma dimensión que x0 (Cambia de 792x1 a

%%

B1=(SumEmax\_total).\*(1/delta\_t);%%%si incluye delta t, multiplicar por ro

B2=(-SumEmin\_total).\*(1/delta\_t);%%%si incluye delta t, multiplicar por ro

Btrafo=max(At\*ones(tiempo,1)-Lbt12',0);

A=[A1;A2;Atrafo];

B=[B1';B2';Btrafo];

CTotal=[];

CTotal=[CTotal,C2];

CTotal=[CTotal,1000\*ones(1,tiempo)];

x0=zeros(1,tiempo+tiempo);

%Crea un identificador de una función: es un tipo de dato de MATLAB® que almacena una asociación con una función

%Funtion\_handle Un identificador de función es un tipo de dato de MATLAB® que representa una función.

fun = @(x)x\*CTotal';%función anonima @(x) que es x\*CTotal'

options = optimoptions('fmincon', 'Display','iter','Algorithm','sqp', 'MaxFunEvals', 100000, 'MaxIter', 5000, 'Display', 'off');

%optimoptions=Crea opciones de optimización --- Display = Nivel de visualización

%'iter' muestra la salida en cada iteración y emite el mensaje de salida predeterminado.

%Algorithm=Permite escoger el algoritmo de optimización --- SQP = Algoritmo

%de optimización

%MaxFunEvals = Número máximo de evaluaciones de función permitidas, un entero positivo.

%MaxIter = Número máximo de iteraciones permitidas, un entero positivo.

%'off' o 'none' no muestran salida alguna.

sol1=fmincon(fun,x0,A,B,[],[],lb,ub,[],options);

%minimiza con las opciones de optimización especificadas en options.

%Utilice optimoptions para configurar estas opciones.

%Si no hay restricciones de igualdad o desigualdad no lineales, establezca nonlcon = [].

Ppref1=[Ppref1;sol1(1:tiempo)];

%optsolotrafoprimario

A1=[];

A2=[];

A=[];

Atrafo=[];

A1=[tril(ones(tiempo,tiempo),0),zeros(tiempo,tiempo)];

A2=[tril(-ones(tiempo,tiempo),0),zeros(tiempo,tiempo)];

Atrafo=[eye(tiempo),-ones(tiempo)];

ub=[];

%SumPmax\_total=min(sum(SumPmax1),(ones(1,tiempo)\*At-Lbt12));

SumPmax\_total=(sum(SumPmax1));

ub=SumPmax\_total';

ub=[ub;At\*ones(tiempo,1)]; %Le agrego la restricción de máximo del transformador para que tenga la misma dimensión que x0

lb=zeros(tiempo+tiempo,1); % Le sumo 1, para que tenga la misma dimensión que x0

B1=SumEmax\_total.\*(1/delta\_t);%%%si incluye delta t, multiplicar por ro

B2=-SumEmin\_total.\*(1/delta\_t);%%%si incluye delta t, multiplicar por ro

A=[A1;A2];

B=[B1';B2'];

Aeq=[-Atrafo];

Beq=Lbt12';

x0=zeros(1,tiempo+tiempo);

%fun = @(x)(sum(x((tiempo+1):end)\*tiempo)-sum(x(1:tiempo)));

fun = @(x)(sum(x(1:tiempo)));

options = optimoptions('fmincon', 'Display','iter','Algorithm','sqp', 'MaxFunEvals', 1000000, 'MaxIter', 5000, 'Display', 'off');

solTrafo=fmincon(fun,x0,A,B,Aeq,Beq,lb,ub,[],options);

LimTrafo=solTrafo(1,(tiempo+1:end));

LimTrafoPorcentaje=((solTrafo(1,tiempo+1:end))./sum(Ai))\*100;

PprefTrafo=[];

PprefTrafo=[PprefTrafo;solTrafo(1:tiempo)];

%opt coodrina trafos

A1=[];

A2=[];

A=[];

Atrafo=[];

A1=[tril(ones(tiempo,tiempo),0),zeros(tiempo,1)]; %crea una matriz inferior diagonal de 1

A2=[tril(-ones(tiempo,tiempo),0),zeros(tiempo,1)]; %crea una matriz inferior diagonal de -1

%como opticoordinada 1

Atrafo=[eye(tiempo),-ones(tiempo,1)];

PprefLimTrafo=[];

%Prueba=[];

%PruebaLb=[];

%resultados=[];

% con el siguiente bucle, realiza la coordinación por cada trafo 32

for a=1:nag;

ub=[];

SumPmax\_total=min(SumPmax1(a,:),ones(1,tiempo)\*Ai(a)-Lbi12(a,:));

%SumPmax\_total=(SumPmax1(a,:));

ub=SumPmax\_total';

ub=[ub;Ai(a)]; %Le agrego la restricción de máximo del transformador para que tenga la misma dimensión que x0

lb=zeros(tiempo+1,1); % Le sumo 1, para que tenga la misma dimensión que x0

B1=(SumEmax(a,:)-SumSOC\_A(a,:)).\*(1/delta\_t);

B2=-(SumEmin(a,:)-SumSOC\_A(a,:)).\*(1/delta\_t);

A=[A1;A2];

B=[B1';B2'];

Aeq=[-Atrafo];

Beq=Lbi12(a,:)';

x0=zeros(1,tiempo+1);

fun = @(x)((x(tiempo+1)\*tiempo)-sum(x(1:tiempo)));

options = optimoptions('fmincon', 'Display','iter','Algorithm','sqp', 'MaxFunEvals', 1000000, 'MaxIter', 5000, 'Display', 'off');

solLimTrafo=fmincon(fun,x0,A,B,Aeq,Beq,lb,ub,[],options);

PprefLimTrafo(a,:)=solLimTrafo(1:tiempo);

lim\_opt(a)=solLimTrafo(tiempo+1);

end

%for a=1:nag

%Prueba1=PprefLimTrafo(a,:);

%Prueba2=Lbi12(a,:);

%Prueba=[Prueba;Prueba1([hinicial/(1/delta\_t)+1:tiempo/(1/delta\_t)]),Prueba1([1:hinicial/(1/delta\_t)])];

%PruebaLb=[PruebaLb;[Prueba2([hinicial/(1/delta\_t)+1:tiempo/(1/delta\_t)]),Prueba2([1:hinicial/(1/delta\_t)])]]

%end

%resultados=Prueba+PruebaLb;

%VarAdicTrafi

% Pago del agregador para los 1000 casos bajo la tarifa TOU

CTOU=[Ppref(4,:)\*C'];

CTTOU=[CTTOU,CTOU];

% Para el calculo de la tarifa POU

Simul1=[Simul1;PprefLimTrafo(1,:)];

M=mean(Simul1);

for a=1:nag;

Simul=[Simul;PprefLimTrafo(a,:)];

end

% Para calcular la gráfica de Emin y Emax

Emaxevol\_total=[Emaxevol\_total;sum(Emax\_evol1)/1000];

Eminevol\_total=[Eminevol\_total;sum(Emin\_evol1)/1000];

Eoptevol\_total=[Eoptevol\_total;((A1(1:tiempo,1:tiempo)\*(sum(PprefLimTrafo)')/(delta\_t\_inv))+(sum(ESOC\_evol)'))/1000];

% Guarda y grafica los MC perfiles del transformador de distribución primaria

PDT\_total=[PDT\_total;sum(PprefLimTrafo)];

%Limite tarifas

DifDesviTotal=[];

for a=1:nag;

MCA=[];

for mc=1:MC;

MCA=[MCA;Simul((mc\*nag)-(nag-1)+a-1,:)];

medMCA=mean(MCA);

Name=genvarname(['medMCA',num2str(a)]);

DifDesviMCAmax=(max(MCA)-medMCA);

DifDesviMCAmin=(medMCA-min(MCA));

%Name=genvarname(['DifDesviMCA',num2str(a)]);

%eval([Name,'=DifDesviMCA']);

end

%DifDesviTotal=[DifDesviTotal;DifDesviMCA];

%Grafica de las N simumlaciones junto con la grafica de la banda tarifaria mediante diagrama de caja y bigote

%axes1 = axes('Parent',figure,'Position',[0.13 0.22 0.85 0.75]);

axes1 = axes('Parent',figure,'Position',[0.13 0.22 0.85 0.75]);

xlim(axes1,[1 tiempo]);

box(axes1,'on');

hold(axes1,'on');

ylabel('Potencia [kW]','FontWeight','bold')

xlabel('Tiempo [h]','FontWeight','bold')

set(gcf,'units','normalized');

set(gcf,'WindowStyle','normal');

set(gcf,'position',[0 0 0.25 0.25]);

%prueba=max(max(MCA));%Para definir el límite máximo de la gráfica

for b=1:MC;

%plot(MCA(b,:));%,'Color',[0.5 0.5 0.5]

hold on

end

plot(medMCA+DifDesviMCAmax,'MarkerFaceColor',[0 0 0],'Marker','none','LineWidth',1,...

'LineStyle','-','Color',[0 0.44 0.73]);%Grafica el límite superior de la banda de tarifa [0 0.44 0.73]

plot(abs(medMCA-DifDesviMCAmin),'MarkerFaceColor',[0 0 0],'Marker','none','LineWidth',1,...

'LineStyle','-', 'Color',[0 0.44 0.73]);%Grafica el límite inferior de la banda de tarifa

%boxplot((MCA),'color',[0 0 0],'boxstyle','filled','symbol','w'); % Diagrama de caja y bigotes

%axis([1 tiempo 0 300])

%xlim(axes1,[1 tiempo]);

%ylim(axes1,[0 max(max(MCA))+50]);%max(max(MCA)) es para definir el límite máximo de la gráfica

set(axes1,'FontName','Times New Roman','FontSize',14,...

'XColor',[0 0 0],'XTick',[1 3 6 9 12 15 18 21 24],'XTickLabel',...

{'12','15','18','21','0','3','6','9','12'},'XTickLabelRotation',0,...

'YColor',[0 0 0],'ZColor',[0 0 0]);

end

clc

%Grafica Etotal

% Definir el eje y la figura

axes1 = axes('Parent', figure, 'Position', [0.1 0.2 0.87 0.75]);

% Definir límites del eje X

xlim(axes1, [1, 24]);

box(axes1, 'on');

hold(axes1, 'on');

ylabel('Power (kW)', 'FontName', 'Times New Roman', 'FontSize', 12, 'FontWeight', 'bold')

xlabel('Time (h)', 'FontName', 'Times New Roman', 'FontSize', 12, 'FontWeight', 'bold')

% Modifica la ventana de la figura

set(gcf, 'units', 'normalized');

set(gcf, 'WindowStyle', 'normal');

set(gcf, 'position', [0 0 0.25 0.215]);

% Configurar etiquetas y posiciones del eje X

xTicks = [1, 3, 6, 9, 12, 15, 18, 21, 24];

xLabels = {'12', '15', '18', '21', '0', '3', '6', '9', '12'};

% Configurar propiedades del eje X

set(axes1, 'FontName', 'Times New Roman', 'FontSize', 12, ...

'XColor', [0 0 0], 'XTick', xTicks, 'XTickLabel', xLabels, ...

'XTickLabelRotation', 0, 'YColor', [0 0 0], 'ZColor', [0 0 0]);

Eopt\_total = [];

Eoptevol1\_total = Eoptevol\_total';

for a = 1:MC

Eopt\_total = [Eopt\_total; Eoptevol1\_total(1 + (a-1) \* tiempo:tiempo + (a-1) \* tiempo)];

end

% Se ajustan los datos para que se grafiquen dentro del rango deseado

% Suponiendo que el tiempo es un número entero y que los datos tienen la misma cantidad de puntos

time\_indices = linspace(1, 24, size(Emaxevol\_total, 2));

for b = 1:MC

plot(time\_indices, Emaxevol\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

'LineStyle', '-.', 'Color', [0 0 1]);

hold on

plot(time\_indices, Eopt\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

'LineStyle', '-', 'Color', [0 0 0.8]);

plot(time\_indices, Eminevol\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

'LineStyle', '--', 'Color', [0 0 0.6]);

ylabel('Energía [MWh]')

xlabel('Tiempo [h]')

end

% Crear las cajas de texto

annotation('textbox', [0.23 0.61 0.1 0.12], ...

'String', {'E\_M\_A\_X'}, 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

annotation('textbox', [0.70 0.32 0.09 0.12], ...

'String', 'E\_M\_I\_N', 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

annotation('textbox', [0.46 0.57 0.09 0.12], ...

'String', {'E\_P\_O\_U'}, 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

%% Sirve pero no es el mejor

% % Definir el eje y la figura

% axes1 = axes('Parent', figure, 'Position', [0.1 0.2 0.87 0.75]);

%

% % Definir límites del eje X

% xlim(axes1, [1 tiempo]);

% box(axes1, 'on');

% hold(axes1, 'on');

% ylabel('Power (kW)', 'FontName', 'Times New Roman', 'FontSize', 12, 'FontWeight', 'bold')

% xlabel('Time (h)', 'FontName', 'Times New Roman', 'FontSize', 12, 'FontWeight', 'bold')

%

% % Modifica la ventana de la figura

% set(gcf, 'units', 'normalized');

% set(gcf, 'WindowStyle', 'normal');

% set(gcf, 'position', [0 0 0.25 0.215]);

%

% % Calcular las etiquetas del eje X basadas en 'tiempo'

% xTicks = linspace(1, tiempo, min(tiempo, 9));

% xLabels = mod(xTicks + 11, 24); % Ajustar las etiquetas para el formato deseado

%

% % Configurar propiedades del eje X

% set(axes1, 'FontName', 'Times New Roman', 'FontSize', 12, ...

% 'XColor', [0 0 0], 'XTick', xTicks, 'XTickLabel', cellstr(num2str(xLabels')), ...

% 'XTickLabelRotation', 0, 'YColor', [0 0 0], 'ZColor', [0 0 0]);

%

% Eopt\_total = [];

% Eoptevol1\_total = Eoptevol\_total';

% for a = 1:MC

% Eopt\_total = [Eopt\_total; Eoptevol1\_total(1 + (a-1) \* tiempo:tiempo + (a-1) \* tiempo)];

% end

%

% for b = 1:MC

% plot(Emaxevol\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

% 'LineStyle', '-.', 'Color', [0 0 1]);

% hold on

% plot(Eopt\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

% 'LineStyle', '-', 'Color', [0 0 0.8]);

% plot(Eminevol\_total(b, :), 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, ...

% 'LineStyle', '--', 'Color', [0 0 0.6]);

% ylabel('Energía [MWh]')

% xlabel('Tiempo [h]')

% end

%

% % Crear las cajas de texto

% annotation('textbox', [0.23 0.61 0.1 0.12], ...

% 'String', {'E\_M\_A\_X'}, 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

% annotation('textbox', [0.70 0.32 0.09 0.12], ...

% 'String', 'E\_M\_I\_N', 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

% annotation('textbox', [0.46 0.57 0.09 0.12], ...

% 'String', {'E\_P\_O\_U'}, 'LineWidth', 1, 'FontWeight', 'bold', 'FontName', 'Times New Roman', 'FitBoxToText', 'off');

%

%

%% CODIGO ORIGINAL

% %

% axes1=axes('Parent',figure,...

% 'Position',[0.1 0.2 0.87 0.75]);

%

% xlim(axes1,[1 tiempo]);

% box(axes1,'on');

% hold(axes1,'on');

% ylabel('Power (kW)','FontName','Times New Roman','FontSize',12,'FontWeight','bold')

% xlabel('Time (h)','FontName','Times New Roman','FontSize',12,'FontWeight','bold')

%

% % Modifica la ventana de la figura

% set(gcf,'units','normalized');

% set(gcf,'WindowStyle','normal');

% set(gcf,'position',[0 0 0.25 0.215]);

% set(axes1,'FontName','Times New Roman','FontSize',12,...

% 'XColor',[0 0 0],'XTick',[1 3 6 9 12 15 18 21 24],'XTickLabel',...

% {'12','15','18','21','0','3','6','9','12'},'XTickLabelRotation',0,...

% 'YColor',[0 0 0],'ZColor',[0 0 0]);

%

% Eopt\_total=[];

% Eoptevol1\_total=Eoptevol\_total';

% for a=1:MC

% Eopt\_total=[Eopt\_total;Eoptevol1\_total(1+(a-1)\*tiempo:tiempo+(a-1)\*tiempo)];

% end

%

%

% for b=1:MC;

% plot(Emaxevol\_total(b,:),'MarkerFaceColor',[0 0 0],'LineWidth',1,...

% 'LineStyle','-.','Color',[0 0 1]);

% hold on

% plot(Eopt\_total(b,:),'MarkerFaceColor',[0 0 0],'LineWidth',1,...

% 'LineStyle','-','Color',[0 0 0.8]);

% plot(Eminevol\_total(b,:),'MarkerFaceColor',[0 0 0],'LineWidth',1,...

% 'LineStyle','--','Color',[0 0 0.6]);

% ylabel('Energía [MWh]')

% xlabel('Tiempo [h]')

% end

% %Crea la caja de texto

% annotation('textbox',[0.23 0.61 0.1 0.12],...

% 'String',{'E\_M\_A\_X'},'LineWidth',1,'FontWeight','bold','FontName','Times New Roman','FitBoxToText','off');

% annotation('textbox',[0.70 0.32 0.09 0.12],...

% 'String','E\_M\_I\_N','LineWidth',1,'FontWeight','bold','FontName','Times New Roman','FitBoxToText','off');

% annotation('textbox',[0.46 0.57 0.09 0.12],...

% 'String',{'E\_P\_O\_U'},'LineWidth',1,'FontWeight','bold','FontName','Times New Roman','FitBoxToText','off');

% %

%Grafica estrat

%% Grafica Estrategias Modificada 1

% Crea ejes

figure;

axes1 = axes('Position', [0.13 0.22 0.85 0.75]);

hold(axes1, 'on');

% Establece los límites del eje X

xlim(axes1, [1 24]);

% Configura las propiedades del eje X

xTicks = [1 4 7 10 13 16 19 22 25];

xLabels = {'12', '15', '18', '21', '0', '3', '6', '9', '12'};

set(axes1, 'FontName', 'Times New Roman', 'FontSize', 12, ...

'XColor', [0 0 0], 'XTick', xTicks, 'XTickLabel', xLabels, ...

'XTickLabelRotation', 0, 'YColor', [0 0 0], 'ZColor', [0 0 0]);

% Comprueba y ajusta los datos para el ciclo horario

n = length(Lbt12);

factor = 24 / n; % Ajuste del factor de escala si es necesario

% Ajustar la longitud de los datos

xData = linspace(1, 24, n);

% Crea Gráfica

plot(xData, sum(PprefLimTrafo) + Lbt12, 'MarkerFaceColor', [0 0 0], ...

'LineWidth', 2, 'LineStyle', '-', 'Color', [0 0.44 0.73], ...

'DisplayName', 'S\_F\_T\_-\_P\_O\_U');

hold on

% Grafico el perfil proporcionado

plot([3575.62 3404.27 3426.95 3397.70 3405.74 3564.12 4030.09 5053.38 5249.77 5185.41 5002.19 4659.28 ...

4999.99 4999.99 4999.99 4999.41 4996.55 4993.72 4987.55 4394.53 3071.22 3357.65 3649.01 3693.27], ...

'Parent', axes1, 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, 'LineStyle', '-.', 'Color', [0 0 0], ...

'DisplayName', 'S\_T\_O\_U');

hold on

plot(xData, Lbt12, 'MarkerFaceColor', [0 0 0], 'LineWidth', 2, ...

'LineStyle', '--', 'Color', [0.92 0.68 0.13], 'DisplayName', 'BL');

ylabel('Potencia [kW]', 'FontWeight', 'bold')

xlabel('Tiempo [h]', 'FontWeight', 'bold')

plot(xData, PprefTrafo + Lbt12, 'MarkerFaceColor', [0 0 0], 'LineWidth', 2, ...

'LineStyle', ':', 'Color', [0.5 0.18 0.55], 'DisplayName', 'S\_P\_D\_T');

hold on

% Agregar leyenda

legend1 = legend(axes1, 'show');

set(legend1, 'Position', [0.155 0.79 0.135 0.093], 'FontSize', 9.5);

% %% Grafica Estrategias ORIGINAL

% % Crea ejes

% axes1 = axes('Parent', figure, 'Position', [0.13 0.22 0.85 0.75]);

% hold(axes1, 'on');

%

% % Establece los límites del eje X

% xlim(axes1, [1 24]); % Ajustar a 24 si tus datos llegan hasta allí

% box(axes1, 'on');

%

% % Modifica la ventana de la figura

% set(gcf, 'units', 'normalized');

% set(gcf, 'WindowStyle', 'normal');

% set(gcf, 'position', [0 0 0.25 0.215]);

%

% % Configura las propiedades del eje X

% xTicks = [1 3 6 9 12 15 18 21 24];

% xLabels = {'12', '15', '18', '21', '0', '3', '6', '9', '12'};

%

% set(axes1, 'FontName', 'Times New Roman', 'FontSize', 12, ...

% 'XColor', [0 0 0], 'XTick', xTicks, 'XTickLabel', xLabels, ...

% 'XTickLabelRotation', 0, 'YColor', [0 0 0], 'ZColor', [0 0 0]);

%

% % Crea Gráfica

% plot(sum(PprefLimTrafo) + Lbt12, 'MarkerFaceColor', [0 0 0], 'LineWidth', 2, ...

% 'LineStyle', '-', 'Color', [0 0.44 0.73], 'DisplayName', 'S\_F\_T\_-\_P\_O\_U');

% hold on

%

% % % Grafico el perfil proporcionado

% % plot([3575.62 3404.27 3426.95 3397.70 3405.74 3564.12 4030.09 5053.38 5249.77 5185.41 5002.19 4659.28 ...

% % 4999.99 4999.99 4999.99 4999.41 4996.55 4993.72 4987.55 4394.53 3071.22 3357.65 3649.01 3693.27], ...

% % 'Parent', axes1, 'MarkerFaceColor', [0 0 0], 'LineWidth', 1, 'LineStyle', '-.', 'Color', [0 0 0], ...

% % 'DisplayName', 'S\_T\_O\_U');

% % hold on

%

% plot(Lbt12, 'MarkerFaceColor', [0 0 0], 'LineWidth', 2, ...

% 'LineStyle', '--', 'Color', [0.92 0.68 0.13], 'DisplayName', 'BL');

% ylabel('Potencia [kW]', 'FontWeight', 'bold')

% xlabel('Tiempo [h]', 'FontWeight', 'bold')

%

% plot(PprefTrafo + Lbt12, 'Parent', axes1, 'MarkerFaceColor', [0 0 0], 'LineWidth', 2, ...

% 'LineStyle', ':', 'Color', [0.5 0.18 0.55], 'DisplayName', 'S\_P\_D\_T');

% hold on

%

% legend1 = legend(axes1, 'show');

% set(legend1, 'Position', [0.155 0.79 0.135 0.093], 'FontSize', 9.5);

%GraficaPDT

% Crear ejes

axes1 = axes('Parent', figure, 'Position', [0.13 0.22 0.85 0.75]);

xlim(axes1, [1 24]); % Ajustar el límite del eje X

ylim(axes1, [0.95 1.05]); % Ajustar el límite del eje Y

box(axes1, 'on');

hold(axes1, 'on');

ylabel('Power (kW)', 'FontWeight', 'bold')

xlabel('Time (h)', 'FontWeight', 'bold')

% Modificar la ventana de la figura

set(gcf, 'units', 'normalized');

set(gcf, 'WindowStyle', 'normal');

set(gcf, 'position', [0 0 0.25 0.215]);

% Configurar etiquetas y posiciones del eje X

xTicks = [1 3 6 9 12 15 18 21 24];

xLabels = {'12', '15', '18', '21', '0', '3', '6', '9', '12'};

set(axes1, 'FontName', 'Times New Roman', 'FontSize', 12, 'FontWeight', 'bold', ...

'XColor', [0 0 0], 'XTick', xTicks, 'XTickLabel', xLabels, ...

'XTickLabelRotation', 0, 'YColor', [0 0 0], 'ZColor', [0 0 0]);

% Ajustar el rango de datos para el gráfico

% Crear un vector de tiempo que mapea los valores de tus datos al rango del eje X

% Aquí se asume que tus datos están en el rango 1 a 24 y deben ajustarse a las etiquetas

% Suponiendo que tus datos están en PDT\_total con dimensiones (MC, N)

% Asegúrate de que cada fila de PDT\_total se ajuste a los valores del eje X

numPoints = size(PDT\_total, 2); % Número de puntos de datos en cada serie

% Graficar los datos

for b = 1:MC

plot(linspace(1, 24, numPoints), PDT\_total(b, :), 'Color', [0.5 0.5 0.5]); % Ajusta x a 1:24

hold on

end

% Crear el diagrama de caja y bigotes

boxplot(PDT\_total', 'Color', [0 0 0], 'BoxStyle', 'filled', 'Symbol', 'w'); % Transponer PDT\_total para boxplot

set(gca, 'XTick', xTicks, 'XTickLabel', xLabels); % Asegurarse de que las etiquetas se muestren correctamente

% Ajustar límites del eje Y (si es necesario)

ylim(axes1, [0 max(max(PDT\_total)) + 400]);