Flujo de Cargas

Sistemas Eléctricos de Potencia
3ºC, I.E.M

Ignacio Rafael Pastor Escribano
Guillermo José Mairal Zueras
Pedro Juan Pardo Posadas
Santiago López López
Javier Miñarro Mena
Ignacio Sanz Soriano
Grupo C4

27 de abril de 2016

Índice

Índice de Tablas **1**

1. Enunciado **2**

2. Identificacion de Nudos **3**

3. Matriz de Admitancias Nodales $\mathbf{Y_{BUS}}$ **3**
3.1. Código *Matlab* 4
3.2. Resultados 5

4. Flujo de Cargas: Solución DC **5**
4.1. Código *Matlab* 5
4.2. Resultados 6

5. Flujo de Cargas: Solución Completa **7**
5.1. Código *Matlab* 8
5.2. Resultados 11

6. Resumen de Potencias y Pérdidas en la Red **12**

7. Análisis de Contingencias **14**
7.1. Código *Matlab* 14
7.2. Resultados 15

Referencias **16**

Índice de Tablas

1. Valores de Las Potencias Demandadas 2
2. Identificación de Nudos 3
3. *Mismatches* 11
4. Vector de Actualizaciones 12
5. Evolución de las variables de la Red. 12
6. Solución. 12
7. Tabla de Resumen Potencias 14
8. Tabla de Pérdidas 14
9. Tabla de Análisis de Contingencias 16

1. Enunciado

INI	FIN	Descripción
2	3	**TRANSFORMADOR** 200 MVA, 410/20 kV; $v_{cc} = 13\,\%$
1	4	**LÍNEA**: 400 kV; $z_s = 0.03$+j0.31 pu, $y_p =$ j0.87 pu
1	4	**LÍNEA**: 400 kV; $z_s = 0.03$+j0.31 pu, $y_p =$ j0.87 pu
1	2	**LÍNEA**: 400 kV; $z_s = 0.02$+j0.23 pu, $y_p =$ j0.52 pu
2	4	**LÍNEA**: 400 kV; $z_s = 0.01$+j0.19 pu, $y_p =$ j0.41 pu

NUDO	V[p.u]	θ [º]	PG [MW]	QG [MVAr]	PD [MW]	QD [MVAr]
1	???	???	0.0	0.0	PD_1	QD_1
2	???	???	0.0	0.0	0.0	0.0
3	1.03	???	60.0		0.0	0.0
4	1.01	0.00			0.0	0.0

Grupo	C1	E1	C2	E2	C3	E3	C4	E4	C5	E5	C6
PD_1 [MW]	170	180	190	200	210	220	210	200	190	180	170
QD_1 [MVAr]	20	25	30	35	40	45	20	25	30	35	40

Cuadro 1: Valores de Las Potencias Demandadas

Las tablas muestran los datos de una red de 4 nudos. Las bases trifásicas del sistema son {400, 20} kV y 100 MVA. Los datos del transformador son relativos a sus propias bases. Los datos de las líneas son relativos a las bases del sistema. Se pide:

1. Identificar el tipo de cada nudo, las ecuaciones (*mismatches*) que definen el sistema y las variables que lo conforman.
2. Obtener la matriz de admitancias nodales $\mathbf{Y_{BUS}}$.
3. Obtener la solución aproximada del sistema empleando el flujo de cargas DC.
4. Obtener la solución del problema de flujo de cargas planteado para esta red (considerar tolerancia de 10^{-2} pu para los *mismatches*), empleando para ello el flujo de cargas completo. Tómese como punto de partida para los ángulos de las tensiones la solución aproximada delsistema obtenida en el apartado anterior mediante el flujo de cargas DC, y módulo 1.0 pu.
5. Completar la información del sistema con las siguientes magnitudes: potencia activa producida por el grupo del nudo 4, potencia reactiva producida por los grupos de los nudos 3 y 4, y pérdidas totales de potencia activa en la red.
6. Realizar el análisis de contingencias de cada una de las líneas que conectan los nudos de la red de 400 kV. En el caso del doble circuito que une los nudos 1 y 4, se ha de considerar tanto la contingencia de una de las líneas como la de las dos a la vez.

2. Identificacion de Nudos

A continuación se identifican en la siguiente tabla los nudos que conforman la red con sus características:

NUDO	**1**	**2**	**3**	**4**
tipo	PQ	PQ	PV	SW
mP	*Sí*	*Sí*	*Sí*	*No*
mQ	*Sí*	*Sí*	*No*	*No*
$\Delta\delta$	δ_1	δ_2	δ_3	-
ΔV	ΔV_1	ΔV_2	-	-

Cuadro 2: Identificación de Nudos

Para tener un esquema más claro de la red, a continuación se muestra el esquema unifilar de la misma.

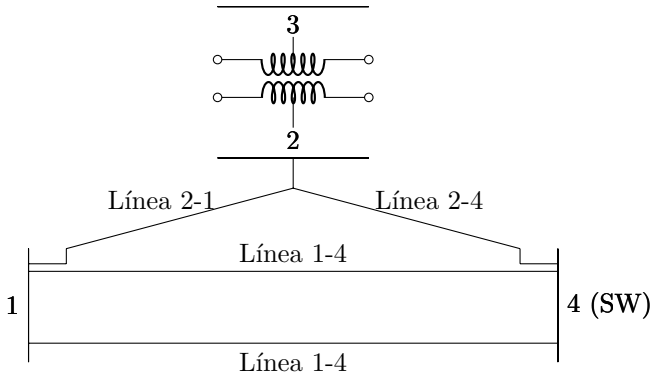

Figura 1: Esquema unifilar de la Red

3. Matriz de Admitancias Nodales $\mathbf{Y_{BUS}}$

Para construir la matriz de Admitancias Nodales de la Red, sabemos que la suma de intensidades que llegan al nudo i se define como:

$$\vec{I}_i = \sum_j [Y_{i,j}\vec{V}_j] \tag{1}$$

y que por lo tanto los elementos de la diagonal de la matriz de admitancias serán la suma de las admitancias que convergen en el nudo i, y que los elemntos de las diagonales secundarias serán las admitancias que van del nudo i al nudo k, con signo negativo.

$$[Y_{BUS}] = \begin{pmatrix} \sum_k y_{1,k} & \cdots & -y_{1,k} \\ \vdots & \ddots & \vdots \\ -y_{i,1} & \cdots & \sum_i y_{i,k} \end{pmatrix} \tag{2}$$

teniendo en cuenta que de entre los distintos elemntos de la red se encontrarán las impedancias de las líneas, los *shunts*, los transformadores ... Estos últimos precisarán de una matriz Y_{BUS} específica que tiene en cuenta la relación de espiras del propio transformador, que será de la forma:

$$[Y_{BUS}^{trafo}] = \begin{pmatrix} \frac{y_{cc}}{\vec{t}^2} & \frac{-y_{cc}}{\vec{t}^*} \\ \frac{-y_{cc}}{\vec{t}} & y_{cc} \end{pmatrix} \tag{3}$$

3.1. Código *Matlab*

```
clear all
format short
disp('  ')
disp('  ')
echo on

% Los datos de la red de 4 nudos son los siguientes:
%
% INI   FIN Descripcion
% 2 3   TRANSFORMADOR: 200 MVA, 410/20kV; ucc = 13 %
% 1 4   LINEA: 400kV; zs = 0.03+j0.31 p.u., yp = j0.87 p.u.
% 1 4   LINEA: 400kV; zs = 0.03+j0.31 p.u., yp = j0.87 p.u.
% 1 2   LINEA: 400kV; zs = 0.02+j0.23 p.u., yp = j0.52 p.u.
% 2 4   LINEA: 400kV; zs = 0.01+j0.19 p.u., yp = j0.41 p.u.
%
% Los datos de los transformadores son relativos a sus propias bases.
% Los datos de las lineas son relativos a las bases del sistema.
% Obtener la matriz de admitancias nodales Ybus, considerando bases trifasicas del
    sistema de 400/220kV y 100 MVA.

%%%%%%%%%%%%%%%%%%%%CONSTANTES
J = sqrt(-1) ;

% Bases TRIFASICAS
Sbase = 100/3 ;
Ubase = [ 400 20 ]/sqrt(3) ;
Ibase = Sbase./Ubase
Zbase = Ubase./Ibase

% YBUS
YBUS = zeros(4,4)

% Lineas
% linea 1 - 4
zs_14 = 0.03 + J*0.31
yp_14 = J*0.87
YBUS(1,1) = YBUS(1,1) + 1/zs_14 + yp_14/2 ;
YBUS(1,4) = YBUS(1,4) - 1/zs_14 ;
YBUS(4,1) = YBUS(4,1) - 1/zs_14 ;
YBUS(4,4) = YBUS(4,4) + 1/zs_14 + yp_14/2 ;
YBUS
% linea 1 - 4 BIS
zs_14bis = 0.03 + J*0.31;
yp_14bis = J*0.87;
YBUS(1,1) = YBUS(1,1) + 1/zs_14 + yp_14/2 ;
YBUS(1,4) = YBUS(1,4) - 1/zs_14bis ;
YBUS(4,1) = YBUS(4,1) - 1/zs_14bis ;
YBUS(4,4) = YBUS(4,4) + 1/zs_14bis + yp_14bis/2 ;
YBUS
% linea 1 - 2
zs_12 = 0.02 + J*0.23;
yp_12 = J*0.52;
YBUS(1,1) = YBUS(1,1) + 1/zs_12 + yp_12/2 ;
YBUS(1,2) = YBUS(1,2) - 1/zs_12 ;
YBUS(2,1) = YBUS(2,1) - 1/zs_12 ;
YBUS(2,2) = YBUS(2,2) + 1/zs_12 + yp_12/2 ;
YBUS
% linea 2 - 4
zs_24 = 0.01 + J*0.19;
yp_24 = J*0.41;
YBUS(2,2) = YBUS(2,2) + 1/zs_24 + yp_24/2 ;
YBUS(2,4) = YBUS(2,4) - 1/zs_24 ;
YBUS(4,2) = YBUS(4,2) - 1/zs_24 ;
YBUS(4,4) = YBUS(4,4) + 1/zs_24 + yp_24/2 ;
YBUS

% Trafos
```

```
% trafo 2 - 3
trafos_Snom23 = 200/3 ;
trafos_VnomA23 = 410/sqrt(3) ;
trafos_VnomB23 = 20/sqrt(3) ;
trafos_ucc23 = 13/100 ;
trafos_zcc23_basesTRAFO = J*trafos_ucc23 ;
trafos_zcc23 = trafos_zcc23_basesTRAFO*((trafos_VnomB23^2)/trafos_Snom23)/Zbase(2)  % La
    ponemos en el lado de BAJA
trafos_t23 = (trafos_VnomA23/trafos_VnomB23)/(Ubase(1)/Ubase(2))   % en el lado de ALTA
YBUS(2,2) = YBUS(2,2) + 1/((trafos_t23*trafos_t23)*trafos_zcc23)  ;
YBUS(2,3) = YBUS(2,3) - 1/((trafos_t23)*trafos_zcc23)  ;
YBUS(3,2) = YBUS(3,2) - 1/((trafos_t23)*trafos_zcc23)  ;
YBUS(3,3) = YBUS(3,3) + 1/trafos_zcc23  ;
YBUS

%%%%%%%%%%%%%%%%RESULTADOS
echo off
disp('Matriz de admitancias nodales YBUS');
disp(num2str(YBUS,'%8.3f'))
```

3.2. Resultados

Matriz de Admitancias Nodales (p.u.):

$$Y_{BUS} = \begin{pmatrix} 0{,}994 - 9{,}577i & -0{,}375 + 4{,}315i & 0{,}000 + 0{,}000i & -0{,}619 + 6{,}392i \\ -0{,}375 + 4{,}315i & 0{,}651 - 23{,}742i & 0{,}000 + 15{,}009i & -0{,}276 + 5{,}249i \\ 0{,}000 + 0{,}000i & 0{,}000 + 15{,}009i & 0{,}000 - 15{,}385i & 0{,}000 + 0{,}000i \\ -0{,}619 + 6{,}392i & -0{,}276 + 5{,}249i & 0{,}000 + 0{,}000i & 0{,}895 - 10{,}565i \end{pmatrix} \tag{4}$$

4. Flujo de Cargas: Solución DC

Para una primera resolución del flujo de cargas empleando el flujo de cargas en DC, se han de asumir ciertas aproximaciones previas.

Por un lado, en el flujo de cargas en DC, solo se consideran las potencias activas de los nudos (P_i) y los respectivos ángulos (δ) de las tensiones en cada nudo. Esto simplifica mucho la resolución con una linealización del problema bajo ciertas aproximaciones. Por otro lado, se despreciarán la componente resistiva de las impedancias del sistema, y además, la función trigonométrica del seno se aproximará por su ángulo ya que este es bastante pequeño ($\delta < 10^o$). De esta forma, consideramos lo siguiente:

$$z_{i,j} = r_{i,j} + j\chi_{i,j} \approx \chi_{i,j} \rightarrow g_{i,j} \approx 0 \tag{5}$$

Además, con la siguiente aproximación del seno calcularemos los flujos de potencia entre nudos aproximadamente con la ecuación 7:

$$\lim_{\delta \to 0} \sin(\delta) \approx \delta \tag{6}$$

$$P_{i \to j} = \frac{V_i V_j}{\chi_{i,j}} \sin(\delta_{i,j}) \approx \frac{\delta_i - \delta_j}{\chi_{i,j}} \tag{7}$$

Una vez asumidas estas simplificaciones, la resolución del flujo de cargas será:

$$Pe_i = \sum_j [Y_{i,j}^{DC} \delta_{i,j}] \tag{8}$$

4.1. Código *Matlab*

```
disp('  ')
echo on
%%%%%%%%%%%%%%%%%%%%%CONSTANTES
```

```
J = sqrt(-1) ;
Sbase = 100 ;
numeroNUDOS = 4 ;

%%%%%%%%%%%%%%%%%%%%%%DATOS
x12 = 0.23 ;
x14bis = 0.31;
x14 = 0.31 ;
x23 = 0.065 ;
x24 = 0.19;
% Ybus DC
YbusDC = [
    [ (1/x12+1/x14+1/x14bis)       -1/x12               0        -(1/x14+1/x14bis)] ;
    [ -1/x12                 (1/x12+1/x23+1/x24)      -1/x23          -1/x24      ] ;
    [   0                    -1/x23          (1/x23)         0        ] ;
    [ -(1/x14+1/x14bis)            -1/x24                0      (1/x24+1/x14+1/x14bis)];
]

% El slack es el 4 -> recortar fila y columna
YbusDC_mismP = YbusDC(1:3,1:3)
invYbusDC_mismP = YbusDC_mismP\eye(size(YbusDC_mismP)) % A\B = inv(A)*B

% Potencias especificadas
Pe1 =  -210/Sbase %   Activa nudo 2, en pu
Pe2 =     0/Sbase
Pe3 =    60/Sbase %   Activa nudo 3, en pu

% Vector potencias inyectadas
Piny = [
    Pe1 ;
    Pe2 ;
    Pe3 ;
    ]

% Angulos de las tensiones
vecTH = invYbusDC_mismP*Piny
vecTH = [  vecTH ; 0.0 ] % Anadir el del slack
TH1 = vecTH(1)
TH2 = vecTH(2)
TH3 = vecTH(3)
TH4 = vecTH(4)  % referencia

disp('Vector de angulos(grados)')
vecTH = [  vecTH ]*180/pi
```

4.2. Resultados

Vector de ángulos ($\vec{\delta}$) en cada uno de los nudos (Nudo 4, *Slack*):

$$\vec{\delta_i} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \end{pmatrix} = \begin{pmatrix} -11{,}8617^o \\ -1{,}7891^o \\ 0{,}4454^o \\ 0^o \end{pmatrix} \tag{9}$$

5. Flujo de Cargas: Solución Completa

La solución del flujo de cargas completa se lleva a cabo mediante métodos numéricos iterativos, en este caso *Newton-Raphson*, que precisan del cálculo de unos *mismatches* de potencia activa y reactiva (*MP* y *MQ*) que se irán actualizando hasta converger con la solución final. Una vez alcanzados los valores de tolerancia para los *mismatches* (10^{-2}), obtendremos los valores de la tensión (V) y su correspondiente ángulo (δ) en cada nudo. Para ello, primero calcularmos los *mismatches* iniciales como en la ecuación 11:

$$\vec{I_i} = \sum_j [Y_{i,j}\vec{V_j}] \tag{10}$$

$$\vec{S_i} = [\vec{V_i}] \cdot [\vec{I_i}]^* = [P_i + jQ_i] \tag{11}$$

A continuación mediante *Newton-Raphson*, aproximaremos el valor de las tensiones y los ángulos (V y δ) en el punto n+1, como la derivada de P y Q en n por un cierto incremento de potencias (ΔP y ΔQ), que serán los *mismatches*. Se puede resumir como:

$$x^{t)} = x^{t-1)} + \Delta x \rightarrow \Delta x = -\frac{g(x^{t-1)})}{\frac{\partial g}{\partial x}(x^{t-1)})} \tag{12}$$

Particularizando para el caso de las funciones P(δ,V) y Q(δ,V), y para un caso de k nudos, habremos de sustituir la derivada parcial en un punto por el correspondiente jacobiano:

$$\jmath = \begin{pmatrix} \frac{\partial P}{\partial \delta} & \frac{\partial P}{\partial V} \\ \frac{\partial Q}{\partial \delta} & \frac{\partial Q}{\partial V} \end{pmatrix} \tag{13}$$

$$\begin{pmatrix} \Delta\delta \\ \Delta V \end{pmatrix}_{t)} = \begin{pmatrix} J_{P,\delta} & J_{P,V} \\ J_{Q,\delta} & J_{Q,V} \end{pmatrix}^{-1}_{t-1)} \begin{pmatrix} MP \\ MQ \end{pmatrix}_{t-1)} \tag{14}$$

Se ha estructurado el Jacobiano como sigue:

$$\jmath = \left(\begin{array}{c|cc} M_{i,k} & N_{i,i} + 2|V_i|^2 g_{i,i} & -N_{i,k} \\ \vdots & -N_{i,k} & N_{i,i} + 2|V_i|^2 g_{i,i} \\ \hline N_{i,k} & -M_{i,i} - 2|V_i|^2 b_{i,i} & -M_{i,k} \\ \vdots & -M_{i,k} & -M_{i,i} - 2|V_i|^2 b_{i,i} \end{array}\right) \tag{15}$$

donde definimos los elementos del jacobiano como sigue:

$$M_{i,k} = \frac{\partial P_i}{\partial \delta_k} = -V_i V_k Y_{i,k} \sin(\theta_{i,k} + \delta_k - \delta_i) \tag{16}$$

$$M_{i,i} = \frac{\partial P_i}{\partial \delta_i} = -Q_i - |V_i|^2 b_{i,i} \tag{17}$$

$$N_{i,k} = \frac{\partial Q_i}{\partial \delta_k} = -V_i V_k Y_{i,k} \cos(\theta_{i,k} + \delta_k - \delta_i) \tag{18}$$

$$N_{i,i} = \frac{\partial Q_i}{\partial \delta_i} = P_i - |V_i|^2 g_{i,i} \tag{19}$$

siendo P_i y Q_i las potencias activa y reactiva inyectadas en el nudo i, que tienen la forma:

$$P_i = |V_i|^2 g_{i,i} + \sum_{i \neq j} V_i V_k Y_{i,k} \cos(\theta_{i,k} + \delta_k - \delta_i) \tag{20}$$

$$Q_i = -|V_i|^2 b_{i,i} - \sum_{i \neq j} V_i V_k Y_{i,k} \sin(\theta_{i,k} + \delta_k - \delta_i) \tag{21}$$

A continuación, procedemos a resolver el flujo de cargas completo.

5.1. Código *Matlab*

```
% Flujo de Cargas: Solucion Completa.
% Se incluye tambien a su derecha la matriz de admitancias nodales Ybus de la misma,
% considerando como potencia base 100 MVA
% Tomando como punto de partida en perfil plano de tensiones,
% obtener la solucion del problema de flujo de cargas planteado para esta red (
    considerar tolerancia en p.u. = 10-2),
% empleando para ello el flujo de cargas completo.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%CONSTANTES
J = sqrt(-1) ;
toleranciaMISMATCHES = 1.0e-2 ;
numITERACIONES = 0 ;
Sbase = 100 ;
numeroNUDOS = 4 ;

%Ybus
Gbus = real(Ybus)
Bbus = imag(Ybus)

% TIPO de NUDOS:
%   Nudo   Pe    Qe      tipo    mismP    mismQ    TH esp.   V esp.
%      1   SI    SI  -->   PQ       SI       SI       NO       NO
%      2   SI    SI  -->   PQ       SI       SI       NO       NO
%      3   SI    SI  -->   PV       SI       NO       NO       SI
%      4   NO    NO  -->   SW       NO       NO       NO       NO

% Potencias especificadas
Pe1 =  -210/Sbase %   Activa nudo 1, en pu
Qe1 =   -20/Sbase %   Reactiva nudo 1, en pu
Pe2 =     0/Sbase %   Activa nudo 2, en pu
Qe2 =     0/Sbase %   Reactiva nudo 2, en pu
Pe3 =    60/Sbase %   Reactiva nudo 3, en pu

% Tensiones en modulo (V) argumento (TH) y forma fasorial (fasV)
% ??? -> Perfil plano de tensiones
V1 = 1.00 ;  fasV1 = V1*exp(J*TH1) ;
V2 = 1.00 ;  fasV2 = V2*exp(J*TH2) ;
V3 = 1.03 ;  fasV3 = V3*exp(J*TH3) ;
V4 = 1.01 ;  fasV4 = V3*exp(J*TH4) ;
% Vector de tensiones en forma fasorial (OJO: vector columna)
vecV = [ fasV1 ; fasV2 ; fasV3 ;fasV4]
% Guardar para RESULTADOS
X_inicial = [ TH1 ; TH2 ; TH3 ; V1 ; V2 ]  ;

% COMPROBACION MISMATCHES
% Vector de corrientes netas inyectadas
vecI = Ybus*vecV
% Potencias netas inyectadas, esto es, potencias calculadas (Sc=Pc+J*Qc)
Sc1 = vecV(1)*conj(vecI(1))
Sc2 = vecV(2)*conj(vecI(2))
Sc3 = vecV(3)*conj(vecI(3))
Sc4 = vecV(4)*conj(vecI(4))
% En forma de vector
Sc = [ Sc1 ; Sc2 ; Sc3 ; Sc4] ;
% Mismatches, segun tabla:
MP1 = Pe1 - real(Sc1)
MQ1 = Qe1 - imag(Sc1)
MP2 = Pe2 - real(Sc2)
MQ2 = Qe2 - imag(Sc2)
MP3 = Pe3 - real(Sc3)
% Vector de Mismatches (OJO: vector columna)
vecMISMATCHES = [ MP1 ; MP2 ; MP3 ; MQ1 ; MQ2 ]
% Guardar para RESULTADOS
vecMISMATCHES_inicial = vecMISMATCHES ;
% Error MAXIMO
errmaxMISMATCHES = max(abs(vecMISMATCHES))
```

```
% Proceso iteratico: se entrara y no se saldra hasta que se alcanze la tolerancia
while (errmaxMISMATCHES>toleranciaMISMATCHES)

    % Si estamos aqui, es necesario hacer una iteracion mas
    numITERACIONES =numITERACIONES + 1

    % Matriz Jacobiana
    % IMPORTANTE: no confundir el NUMERO de nudo con POSICION en el subjacobiano
        correspondiente
    % subjacobiano Jpth (3x3)
    Jpth (1,1) = - imag(Sc1) - V1^2*Bbus(1,1) ;  % dPc1_dTH1
    Jpth (1,2) = V1*V2*( Gbus(1,2)*sin(TH1-TH2) - Bbus(1,2)*cos(TH1-TH2) ) ; % dPc1_dTH2
    Jpth (1,3) = V1*V3*( Gbus(1,3)*sin(TH1-TH3) - Bbus(1,3)*cos(TH1-TH3) ) ; % dPc1_dTH3

    Jpth (2,1) = V1*V2*( Gbus(1,2)*sin(TH1-TH2) - Bbus(1,2)*cos(TH1-TH2) ) ; % dPc1_dTH2
    Jpth (2,2) = - imag(Sc2) - V2^2*Bbus(2,2) ; % dPc2_dTH2
    Jpth (2,3) = V2*V3*( Gbus(2,3)*sin(TH2-TH3) - Bbus(2,3)*cos(TH2-TH3) ) ; % dPc2_dTH3

    Jpth (3,1) = V3*V1*( Gbus(3,1)*sin(TH3-TH1) - Bbus(3,1)*cos(TH3-TH1) ) ; % dPc3_dTH1
    Jpth (3,3) = - imag(Sc3) - V3^2*Bbus(3,3) ; % dPc3_dTH3
    Jpth (3,2) = V3*V2*( Gbus(3,2)*sin(TH3-TH2) - Bbus(3,2)*cos(TH3-TH2) ) ; % dPc3_dTH2
    Jpth
    % subjacobiano Jpv (3x2)
    Jpv (1,1) = real(Sc1) + V1^2*Gbus(1,1)  ; % V1*dPc1_dV1
    Jpv (1,2) = V1*V2*( Gbus(1,2)*cos(TH1-TH2) + Bbus(1,2)*sin(TH1-TH2) ) ;  % V2*
        dPc1_dV2

    Jpv (2,1) = V2*V1*( Gbus(2,1)*cos(TH2-TH1) + Bbus(2,1)*sin(TH2-TH1) ) ;  % V1*
        dPc2_dV1
    Jpv (2,2) = real(Sc2) + V2^2*Gbus(2,2)  ; % V2*dPc2_dV2

    Jpv (3,1) = V3*V1*( Gbus(3,1)*cos(TH3-TH1) + Bbus(3,1)*sin(TH3-TH1) ) ;  % V1*
        dPc3_dV1
    Jpv (3,2) = real(Sc3) + V3^2*Gbus(3,3)  ; % V2*dPc3_dV2
    Jpv
    % subjacobiano Jqth (2x3)
    Jqth (1,1) = real(Sc1) - V1^2*Gbus(1,1) ;  % dQc1_dTH1
    Jqth (1,2) = - V1*V2*( Gbus(1,2)*cos(TH1-TH2) + Bbus(1,2)*sin(TH1-TH2) ) ;  %
        dQc1_dTH2
    Jqth (1,3) = - V1*V3*( Gbus(1,3)*cos(TH1-TH3) + Bbus(1,3)*sin(TH1-TH3) ) ;  %
        dQc1_dTH3

    Jqth (2,2) = real(Sc2) - V2^2*Gbus(2,2) ;  % dQc2_dTH2
    Jqth (2,1) = - V2*V1*( Gbus(2,1)*cos(TH2-TH1) + Bbus(2,1)*sin(TH2-TH1) ) ;  %
        dQc2_dTH1
    Jqth (2,3) = - V2*V3*( Gbus(2,3)*cos(TH2-TH3) + Bbus(2,3)*sin(TH2-TH3) ) ;  %
        dQc2_dTH3
    Jqth
    % subjacobiano Jqv (2x2)
    Jqv (1,1) = imag(Sc1) - V1^2*Bbus(1,1) ; % V1*dQc1_dV1
    Jqv (1,2) = V1*V2*( Gbus(1,2)*sin(TH1-TH2) - Bbus(1,2)*cos(TH1-TH2) ) ; % V2*
        dQc1_dV2

    Jqv (2,1) = V1*V2*( Gbus(1,2)*sin(TH1-TH2) - Bbus(1,2)*cos(TH1-TH2) ) ; % V1*
        dQc2_dV1
    Jqv (2,2) = imag(Sc2) - V2^2*Bbus(2,2) ; % V2*dQc2_dV2
    % jacobiano
    JACOBIANO = [
        [  Jpth   Jpv  ] ;
        [  Jqth   Jqv  ] ;
        ]
    % Guardar para RESULTADOS
    iteracionesJACOBIANO(:,:,numITERACIONES) = JACOBIANO ;
    % Vector de actualizaciones de las variables
    incX = inv(JACOBIANO)*vecMISMATCHES
    % actualizaciones de las variables
    incTH1   = incX(1)
    incTH2   = incX(2)
    incTH3   = incX(3)
    incV1_V1 = incX(4)
```

```
    incV2_V2 = incX(5)
    % Guardar para RESULTADOS
    iteracionesINCX(:,numITERACIONES) = incX ;

    % Actualizar variables
    TH1 = TH1 + incTH1
    TH2 = TH2 + incTH2
    TH3 = TH3 + incTH3
    V1 = V1*( 1 + incV1_V1)
    V2 = V2*( 1 + incV2_V2)
    % fasores y vector de tensiones en forma fasorial (OJO: vector columna)
    fasV1 = V1*exp(J*TH1) ;
    fasV2 = V2*exp(J*TH2) ;
    fasV3 = V3*exp(J*TH3) ;
    fasV4 = V4*exp(J*TH4) ;
    vecV = [ fasV1 ; fasV2 ; fasV3 ; fasV4 ]
    % Guardar para RESULTADOS
    iteracionesX(:,numITERACIONES) = [ TH1 ; TH2 ; TH3 ; V1 ; V2 ] ;

    % COMPROBACION MISMATCHES
    % Vector de corrientes netas inyectadas
    vecI = Ybus*vecV
    % Potencias netas inyectadas, esto es, potencias calculadas (Sc=Pc+J*Qc)
    Sc1 = vecV(1)*conj(vecI(1))
    Sc2 = vecV(2)*conj(vecI(2))
    Sc3 = vecV(3)*conj(vecI(3))
    Sc4 = vecV(4)*conj(vecI(4))
    % En forma de vector
    Sc = [ Sc1 ; Sc2 ; Sc3 ; Sc4] ;
    % Mismatches, segun tabla:
    MP1 = Pe1 - real(Sc1)
    MP2 = Pe2 - real(Sc2)
    MP3 = Pe3 - real(Sc3)
    MQ1 = Qe1 - imag(Sc1)
    MQ2 = Qe2 - imag(Sc2)
    % Vector de Mismatches (OJO: vector columna)
    vecMISMATCHES = [ MP1 ; MP2 ; MP3 ; MQ1 ; MQ2 ]
    % Guardar para RESULTADOS
    iteracionesMISMATCHES(:,numITERACIONES) = vecMISMATCHES ;
    % Error MAXIMO
    errmaxMISMATCHES = max(abs(vecMISMATCHES))

end

%%%%%%%%%%%%%%%%%%%%RESULTADOS
echo off
disp('   ');
disp('Evolucion de los MISMATCHES');
disp([ 'iter       ' num2str(0:numITERACIONES,'%8d') ])
disp([ ['MP1  ';'MP2  ';'MP3  ';'MQ1  ';'MQ2  '] num2str([ vecMISMATCHES_inicial
    iteracionesMISMATCHES ],'%8.4f')])
disp('   ');
disp('Evolucion del jacobiano');
for ii=1:numITERACIONES
    disp([ '*** iteracion  ' num2str(ii,'%2d') ])
    disp('        incTH1    incTH2    incTH3  incV1/V1  incV2/V2  ')
    disp([ ['MP1  ';'MP2  ';'MP3  ';'MQ1  ';'MQ2  '] num2str(iteracionesJACOBIANO(:,:,ii
        ),'%10.3f')])
end
disp('   ');
disp('Evolucion del vector de actualizaciones');
disp([ 'iter            ' num2str(1:numITERACIONES,'%8d') ])
disp([ ['incTH1    ';'incTH2    ';'incTH3    ';'incV1/V1  ';'incV2/V2  '] num2str(
    iteracionesINCX,'%8.4f')])
disp('   ');
disp('Evolucion de las variables');
disp([ 'iter       ' num2str(0:numITERACIONES,'%8d') ])
disp([ ['TH1    ';'TH2    ';'TH3    ';'V1     ';'V2     '] num2str([ X_inicial
    iteracionesX ],'%8.4f')])
disp('   ');
```

```
disp('Resumen Solucion');
disp('n    V (pu)    TH (o)    Pc (MW)   Qc(Mvar) ')
for ii=1:numeroNUDOS
    disp(sprintf('%1d %9.4f %9.3f %10.2f %10.2f',ii,abs(vecV(ii)),angle(vecV(ii))*180/pi
        ,Sbase*real(Sc(ii)),Sbase*imag(Sc(ii))))
end
```

5.2. Resultados

Se mostrarán los resultados por el siguiente orden:

1. Evolución de los mismatches.
2. Evolución del Jacobiano.
3. Evolucion del vector de actualizaciones.
4. Evolucion de las variables.
5. Resumen Solucion.

1. Evolución de los *mismatches*.

iter	0	1	2	3	4
MP1	3.7514	-4.7510	-0.7223	-0.0740	-0.0010
MP2	0.0946	1.9021	0.3495	0.0294	-0.0005
MP3	-1.2047	0.2568	0.0148	0.0007	0.0000
MQ1	-1.2537	-2.8452	-0.7863	-0.0733	-0.0010
MQ2	0.8853	-1.2546	-0.2873	-0.0230	-0.0014

Cuadro 3: *Mismatches*

2. Evolución del Jacobiano.

$$\mathcal{J}_{1a\ iter} = \begin{pmatrix} 8{,}523 & -3{,}540 & 0{,}000 & -4{,}858 & -2{,}496 \\ -3{,}540 & 24{,}627 & -15{,}354 & 1{,}848 & 0{,}557 \\ 0{,}000 & -15{,}354 & 15{,}354 & 0{,}000 & 1{,}805 \\ -6{,}845 & 2{,}496 & 0{,}000 & 10{,}631 & -3{,}540 \\ -1{,}848 & -0{,}746 & 1{,}805 & -3{,}540 & 22{,}857 \end{pmatrix}$$

$$\mathcal{J}_{2a\ iter} = \begin{pmatrix} 16{,}127 & -7{,}047 & 0{,}000 & 4{,}599 & 0{,}568 \\ -7{,}047 & 31{,}017 & -18{,}021 & -1{,}775 & -1{,}017 \\ 0{,}000 & -18{,}021 & 18{,}021 & 0{,}000 & 0{,}343 \\ 0{,}703 & -0{,}568 & 0{,}000 & 21{,}417 & -7{,}047 \\ 1{,}775 & -2{,}788 & 0{,}343 & -7{,}047 & 33{,}526 \end{pmatrix}$$

$$\mathcal{J}_{3a\ iter} = \begin{pmatrix} 13{,}230 & -5{,}650 & 0{,}000 & 0{,}056 & -1{,}011 \\ -5{,}650 & 28{,}616 & -17{,}047 & 0{,}021 & 0{,}444 \\ 0{,}000 & -17{,}047 & 17{,}047 & 0{,}000 & 0{,}585 \\ -2{,}811 & 1{,}011 & 0{,}000 & 14{,}402 & -5{,}650 \\ -0{,}021 & -1{,}143 & 0{,}585 & -5{,}650 & 29{,}191 \end{pmatrix}$$

$$\mathcal{J}_{4a\ iter} = \begin{pmatrix} 11{,}949 & -5{,}035 & 0{,}000 & -0{,}799 & -1{,}221 \\ -5{,}035 & 27{,}494 & -16{,}633 & 0{,}333 & 0{,}726 \\ 0{,}000 & -16{,}633 & 16{,}633 & 0{,}000 & 0{,}599 \\ -3{,}253 & 1{,}221 & 0{,}000 & 11{,}695 & -5{,}035 \\ -0{,}333 & -0{,}784 & 0{,}599 & -5{,}035 & 27{,}540 \end{pmatrix}$$

3. Evolución del vector de actualizaciones.

iter	1	2	3	4
incTH1	0.7325	-0.2481	-0.0527	-0.0070
incTH1	0.0380	0.0094	0.0080	0.0001
incTH1	-0.0600	0.0247	0.0097	0.0003
incV1/V1	0.4000	-0.1421	-0.0750	-0.0094
incV2/V2	0.1659	-0.0536	-0.0243	-0.0026

Cuadro 4: Vector de Actualizaciones

4. Evolución de las variables.

iter	0	1	2	3	4
TH1	-0.6211	0.1114	-0.1367	-0.1894	-0.1964
TH1	-0.0937	-0.0557	-0.0463	-0.0383	-0.0382
TH1	0.0233	-0.0367	-0.0120	-0.0023	-0.0020
V1	1.0000	1.4000	1.2011	1.1111	1.1006
V2	1.0000	1.1659	1.1034	1.0766	1.0737

Cuadro 5: Evolución de las variables de la Red.

5. Resumen Solución.

n	V (pu)	TH (º)	Pc (MW)	Qc(Mvar)
1	1.1006	-11.255	-209.90	-19.90
2	1.0737	-2.186	0.05	0.14
3	1.0300	-0.115	60.00	-26.72
4	1.0100	0.000	154.29	-202.44

Cuadro 6: Solución.

6. Resumen de Potencias y Pérdidas en la Red

Por balance de potencias, calculamos la potencia que se pierde en cada una de las líneas. En la siguiente tabla se recogen los datos de potencias activa (P) y reactiva (Q) de cada nudo, y se adjuntan además las pérdidas en las líneas del sistema.

```
% De la resolucion del apartado anterior conocemos los vectores de tension
V1=1.1005*exp(J*(-11.264)*pi/180);
V2=1.0737*exp(J*(-2.193)*pi/180);
```

```
V3=1.0300*exp(J*(-0.121)*pi/180);
V4=1.0100*exp(J*0.00*pi/180);
% Vector de tensiones (vector columna)
vecV= [V1; V2; V3; V4]
% Vector de corrientes netas inyectadas
vecI=YBUS*vecV
% Potencias netas inyectadas, esto es, potencias calculadas
% (Sc=Pc+J*Qc)
Sc1=vecV(1)*conj(vecI(1))
Sc2=vecV(2)*conj(vecI(2))
Sc3=vecV(3)*conj(vecI(3))
Sc4=vecV(4)*conj(vecI(4))

% Dado que para las potencias que nos piden no tenemos mismatches:
% La potencia activa generada por el nudo 4 (SW) es:
Pc4=real(Sc4)
% La potencia reactiva generada por el nudo 4 (SW) es:
Qc4=imag(Sc4)
% La potencia reactiva generada por el nudo 3 es:
Qc3=imag(Sc3)

% Balance de potencias en lineas y trafos
% linea 1 - 4
Ilinea_14 = V1*(yp_14/2) + (V1-V4)/(zs_14)
Ilinea_41 = V4*(yp_14/2) + (V4-V1)/(zs_14)
Plinea_14 = real(V1*conj(Ilinea_14))
Plinea_41 = real(V4*conj(Ilinea_41))
% linea 1 - 4 bis
Ilinea_14bis = V1*(yp_14bis/2) + (V1-V4)/(zs_14bis)
Ilinea_41bis = V4*(yp_14bis/2) + (V4-V1)/(zs_14bis)
Plinea_14bis = real(V1*conj(Ilinea_14bis))
Plinea_41bis = real(V4*conj(Ilinea_41bis))
% linea 1 - 2
Ilinea_12 = V1*(yp_12/2) + (V1-V2)/(zs_12)
Ilinea_21 = V2*(yp_12/2) + (V2-V1)/(zs_12)
Plinea_12 = real(V1*conj(Ilinea_12))
Plinea_21 = real(V2*conj(Ilinea_21))
% linea 2 - 4
Ilinea_24 = V2*(yp_24/2) + (V2-V4)/(zs_24)
Ilinea_42 = V4*(yp_24/2) + (V4-V2)/(zs_24)
Plinea_24 = real(V2*conj(Ilinea_24))
Plinea_42 = real(V4*conj(Ilinea_42))
% trafo 2 - 3
V2prima = V2/trafos_t23
Itrafo_2prima3 = (V2prima-V3)/(trafos_zcc23)
Itrafo_23 = Itrafo_2prima3/trafos_t23
Itrafo_32 = (V3-V2prima)/(trafos_zcc23)
Ptrafo_23 = real(V2*conj(Itrafo_23))
Ptrafo_32 = real(V3*conj(Itrafo_32))

%%%%%%%%%%%%%%%%RESULTADOS
echo off
disp('   ');
disp('Potencia activa generada por el nudo 4')
disp([ '    ' num2str(Pc4) ] )
disp('   ');
disp('Potencia reactiva generada por el nudo 4')
disp([ '    ' num2str(Qc4) ] )
disp('   ');
disp('Potencia reactiva generada por el nudo 3')
disp([ '    ' num2str(Qc3) ] )
disp('   ');
disp('Balance de potencia activa en lineas')
disp([ '   1 a 4 : ' num2str(Plinea_14,'%7.3f') ' ; 4 a 1 : ' num2str(Plinea_41,'%7.3f')
     ' ; perdidas : ' num2str(Plinea_14+Plinea_41,'%7.3f') ] )
disp([ '   1 a 4(bis) : ' num2str(Plinea_14bis,'%7.3f') ' ; 4 a 1(bis) : ' num2str(
    Plinea_41bis,'%7.3f') ' ; perdidas : ' num2str(Plinea_14bis+Plinea_41bis,'%7.3f') ]
    )
disp([ '   1 a 2 : ' num2str(Plinea_12,'%7.3f') ' ; 2 a 1 : ' num2str(Plinea_21,'%7.3f')
     ' ; perdidas : ' num2str(Plinea_12+Plinea_21,'%7.3f') ] )
```

```
disp([ '   2 a 4 : ' num2str(Plinea_24,'%7.3f') ' ; 4 a 2 : ' num2str(Plinea_42,'%7.3f')
     ' ; perdidas : ' num2str(Plinea_24+Plinea_42,'%7.3f') ] )
disp('Balance de potencia activa en trafos')
disp([ '   2 a 3 : ' num2str(Ptrafo_23,'%7.3f') ' ; 3 a 2 : ' num2str(Ptrafo_32,'%7.3f')
     ' ; perdidas : ' num2str(Ptrafo_23+Ptrafo_32,'%7.3f') ] )
```

NUDO	V[p.u]	$\delta_i[^o]$	PG [MW]	QG [MVAr]	PD[MW]	QD[MVAr]
1	1.1006	-11.254	0	0	209.89	19.90
2	1.0737	-2.186	0.05	0.14	0.0	0.0
3	1.0300	-0.114	60.00	0.0	0.0	26.72
4	1.0100	0.000	154.29	0.0	0.0	-202.44

Cuadro 7: Tabla de Resumen Potencias

	Línea 1-4	Línea 1-4 bis	Línea 1-2	Línea 2-4	Línea 2-3
$P_{i\to j}[MW]$	0.672	0.672	0.799	0.200	0.600
$P_{j\to i}$[MW]	-0.656	-0.656	-0.787	-0.199	-0.600
Pérdidas [MW]	0.016	0.016	0.011	0.002	-0.000

Cuadro 8: Tabla de Pérdidas

7. Análisis de Contingencias

Para analizar de forma aproximada como evolucionan los ángulos de las tensiones de los nudos cuando se pierden una o varias líneas, usaremos el método DC, y los principios de sustitución y superposición en la Red y la línea, que nos permite el comportamiento lineal de este método. Sabiendo que:

$$P_e = Y_{DC}\,\theta \to \theta = Y_{DC}^{-1}\,P_e \to \theta = Z_{DC}\,P_e \tag{22}$$

El incremento en los ángulos de las tensiones de los nudos de la Red, se deberá unicamente al incremento de potencia debido a la fuente de potencia que simula la variación de potencia que supone la pérdida de una línea. El valor de flujo de potencia que se inyecta en el nudo i y sale por el nudo j ha de ser nulo. Por lo tanto la variación de ángulos debida a dicha fuente de potencia será:

$$\Delta\theta_i = \sum_k (Z_{DC_{i,k}} \cdot \Delta P_{e_k}) = Z_{DC_{i,i}} \cdot \Delta P^{ctg} - Z_{DC_{i,j}} \cdot \Delta P^{ctg} \tag{23}$$

$$\Delta\theta_i = \sum_k (Z_{DC_{j,k}} \cdot \Delta P_{e_k}) = Z_{DC_{j,i}} \cdot \Delta P^{ctg} - Z_{DC_{j,j}} \cdot \Delta P^{ctg} \tag{24}$$

Finalemente se obtiene el valor de la fuente de potencia adicional:

$$\Delta P^{ctg} = \frac{P^0_{i\to j}}{1 - \dfrac{Z_{DC_{i,i}} + Z_{DC_{j,j}} - Z_{DC_{i,j}} - Z_{DC_{j,i}}}{\chi_{i,j}}} \tag{25}$$

7.1. Código *Matlab*

```
%%% Analisis de contingencias
% OJO con invYbusDC_mismP, puesto que posicion no corresponde con nudo
```

```
% linea 1-2
% Flujo inicial
f12_0 = (TH1-TH2)/x12
% potencia ficticia (+ en 1, - en 2)
incP_ctg12 = f12_0/( 1 - (invYbusDC_mismP(1,1) + invYbusDC_mismP(2,2) - invYbusDC_mismP
    (1,2) - invYbusDC_mismP(2,1) )/x12) % los elementos correspondientes al nudo 1 son
    nulos
incTH_ctg12 = invYbusDC_mismP*[ incP_ctg12; -incP_ctg12 ; 0.0 ]
vecTH_ctg12 = vecTH + [  incTH_ctg12 ; 0.0 ] % Anadir el del slack

% linea 2-4
% Flujo inicial
f24_0 = (TH2-TH4)/x24
% potencia ficticia (+ en 2, - en 4)
incP_ctg24 = f24_0/( 1 - ( invYbusDC_mismP(2,2) + 0.0 - 0.0 - 0.0 )/x24)
incTH_ctg24 = invYbusDC_mismP*[ 0.0 ; incP_ctg24 ; 0.0 ]
vecTH_ctg24 = vecTH + [  incTH_ctg24 ; 0.0] % Anadir el del slack

%linea 2-3
% Flujo inicial
f23_0 = 2*(TH2-TH3)/x23
% potencia ficticia (+ en 1, - en 4)
incP_ctg23 = f23_0/( 1 - (invYbusDC_mismP(2,2) + invYbusDC_mismP(3,3) - invYbusDC_mismP
    (2,3) - invYbusDC_mismP(2,3) )/x14) % los elementos correspondientes al nudo 1 son
    nulos
incTH_ctg23 = invYbusDC_mismP*[ 0.0 ; incP_ctg23; -incP_ctg23 ]
vecTH_ctg23 = vecTH + [  incTH_ctg23 ; 0.0 ] % Anadir el del slack

% las dos lineas 1-4 son iguales, por tanto si se pierde solo 1 de las dos
% es indiferente cual se haya perdido

%linea 1-4 (si se pierde solo 1 de las dos)
% Flujo inicial
f14_0 = (TH1-TH4)/x14
% potencia ficticia (+ en 1, - en 4)
incP_ctg14 = f14_0/( 1 - (invYbusDC_mismP(1,1) + 0.0 - 0.0 - 0.0 )/x14) % los elementos
    correspondientes al nudo 1 son nulos
incTH_ctg14 = invYbusDC_mismP*[ incP_ctg14; 0.0 ; 0.0 ]
vecTH_ctg14 = vecTH + [  incTH_ctg14 ; 0.0 ] % Anadir el del slack

%linea 1-4 (si se pierden las dos lineas)
%Hacemos el paralelo de las 2 impedancias y volvemos a calcular
% Flujo inicial
f14_0_bis = 2*(TH1-TH4)/(x14*0.5)
% potencia ficticia (+ en 1, - en 4)
incP_ctg14_bis = f14_0/( 1 - (invYbusDC_mismP(1,1) + 0.0 - 0.0 - 0.0 )/(0.5*x14)) % los
    elementos correspondientes al nudo 1 son nulos
incTH_ctg14_bis = invYbusDC_mismP*[ +incP_ctg14_bis; 0.0 ; 0.0 ]
vecTH_ctg14_bis = vecTH + [  incTH_ctg14_bis ; 0.0 ] % Anadir el del slack


%%%%%%%%%%%%%%%%RESULTADOS
echo off
disp('   ');
disp('Resumen Solucion');
disp('   TH [o] ')
disp('n    base    linea 12  linea 23  linea 24  linea 14  linea 14x2 ')
for ii=1:4
    disp(sprintf('%1d  %9.4f  %9.4f  %9.4f  %9.4f  %9.4f  %9.4f ',ii,vecTH(ii)*180/pi,
        vecTH_ctg12(ii)*180/pi,vecTH_ctg23(ii)*180/pi,vecTH_ctg24(ii)*180/pi,vecTH_ctg14
        (ii)*180/pi,vecTH_ctg14_bis(ii)*180/pi))
end
```

7.2. Resultados

El análisis de las contingencias considerando la pérdida de cada una de las líneas de la red se puede resumir como sigue:

$\delta_i^{DC}[^o]$	base	linea 12	linea 23	linea 24	linea 14	linea 14x2
$\delta_1[^o]$	-11.8617	-18.6498	-11.8617	-13.3213	-18.6863	-27.9324
$\delta_2[^o]$	-1.7891	6.5317	-1.7891	-5.4145	-4.8764	-9.0592
$\delta_3[^o]$	0.4454	8.7663	6.1002	-3.1799	-2.6419	-6.8247
$\delta_4[^o]$	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Cuadro 9: Tabla de Análisis de Contingencias

Referencias

[1] Francisco M. Echavarren Cerezo y Andrés D. Díaz Casado: *Apuntes sobre Flujo de Cargas*, Universidad Pontificia de Comillas (2016).