Resolución del problema del Flujo de Cargas

Desarrollo de FLUCAR 3.0

Montevideo, Marzo de 2002.

Estudiantes: Alfredo Costa

Claudio Olmedo

Director de Proyecto: Mario Vignolo

RESUMEN EJECUTIVO DEL PROYECTO

Objetivo: Generar la tercera versión del programa de cálculo de flujo de carga incorporando la regulación automática de los reguladores, mejorando la interfase con el usuario y agregando la posibilidad de ejecutar flujos paramétricos.

Etapas

1. Estudio del método de Newton Raphson aplicado a la resolución de flujos de carga.

2. Estudio del método de Newton Raphson modificado planteado por Peterson Meyer en la publicación número 70 TP 160-PWR de la IEEE.

3. Estudio del programa de flujo de carga existente en el IIE, Flucar 2.0. Estudio de otros programas de flujos de carga comerciales. Verificación y comparación del Flucar con los otros programas.

4. Implementación del método descrito en la etapa 2 y validación mediante comparación con Flucar 20.

5. Implementación de una interfase gráfica amigable.

6. Implementación del flujo de carga paramétrico.

RESUMEN EJECUTIVO DEL PROYECTO 1

1. ESTUDIO DEL FLUJO DE CARGA 4

1.1. PRESENTACIÓN DEL PROBLEMA 4
1.2. FORMULACIÓN DEL PROBLEMA 5
1.2.1. Condiciones iniciales y tipos de barras 5
1.2.2. Numeración de las barras 6
1.2.3. Estudio del problema para un nodo k genérico 6
1.2.4. Formulación de las ecuaciones de comportamiento de la red 7

2. CONSIDERACIONES SOBRE EL CALCULO EN POR UNIDAD. 10

2.1. Cálculo de los valores reales a partir de los valores "en por uno" en base V_B 11
2.2. Transformadores con reguladores 13
2.3. Ejemplo calculo por unidad 14

3. METODO DE NEWTON RAPHSON 16

3.1. APLICACIÓN A LA RESOLUCIÓN DE LAS ECUACIONES DE COMPORTAMIENTO DE LA RED 19

4. ESTUDIO DEL FLUCAR 2.O Y OTROS PROGRAMAS DE CALCULO DE FLUJOS DE CARGA COMERCIALES 25

4.1. ESTUDIO DEL FLUCAR 2.0 26
4.1.1. Datos de entrada 26
4.1.2. Datos de salida 28
4.2. ESTUDIO DEL EPRI 29
4.2.1. Datos de entrada 29
4.2.2. Datos de salida 31
4.3. ESTUDIO DEL SWEDNET 35
4.3.1. Datos de entrada 36
4.3.2. Datos de salida 38
4.3.3. Optimización 38
4.4. PRUEBAS REALIZADAS AL FLUCAR 2.0 39
4.4.1. Corrida 1 39
4.4.1.1. Corrida 1 con FLUCAR 39
4.4.1.2. Corrida 1 con EPRI 40
4.4.1.3. Comparación de los resultados obtenidos entre FLUCAR20 y EPRI 43
4.4.1.4. Corrida 1 con SWEDNET 44
4.4.1.5. Comparación de los resultados obtenidos entre FLUCAR20 y SWEDNET 44
4.4.2. Corrida 2 45
4.4.2.1. Corrida 2 con FLUCAR 2.0 45
4.4.2.2. Corrida 2 con EPRI 47
4.4.2.3. Comparación de los resultados obtenidos entre FLUCAR20 y EPRI 48
4.4.2.4. Corrida 2 en SWEDNET 49
4.4.2.5. Comparación de los resultados obtenidos entre FLUCAR20 y SWEDNET 50
4.4.3. Conclusiones 50

5. MÉTODO DE NEWTON RAPHSON MODIFICADO 51

5.1. INTRODUCCIÓN 51

5.2. ESTUDIO DEL ARTÍCULO "AUTOMATIC ADJUSTMENT OF TRANSFORMER AND PHASE-SHIFTER TAPS IN THE NEWTON POWER FLOW" 52
5.3. DISCREPANCIAS ENCONTRADOS EN EL ARTÍCULO ESTUDIADO....57

6. PRUEBAS REALIZADAS AL NUEVO METODO INCORPORADO EN FLUCAR V3.0....61

6.1. PRUEBA 1 61
6.2. PRUEBA 2 64
6.2.1. RESULTADO DEL FLUCAR V3.0....67
6.2.2. RESULTADO DEL FLUCAR 2.0 68
6.2.3. COMPARATIVO SOLUCIONES....69
6.2.4. PRUEBA 2A....70
6.2.4.1. Conclusiones.... 74

7. CORRIDA DE CASO PARTICULAR: SISTEMA ELÉCTRICO URUGUAYO....75

7.1. ENTRADA DE DATOS....80
7.2. RESULTADOS....84
7.3. MODIFICACIONES....87
7.3.1. Entrada de datos....87
7.3.2. Resultado 91
7.3.3. Comprobación Flucar 20....95
7.3.3.1. Entrada de datos 95
7.3.3.2. Salida de datos.... 99
7.3.3.3. Conclusiones.... 104

8. OTRAS MEJORAS INTRODUCIDAS EN FLUCAR V3.0....107

8.1. INTERFASE GRAFICA DE ENTRADA SALIDA DATOS 107
8.1.1. Introducción....107
8.1.2. Pantalla Principal....109
8.1.2.1. Abrir proyecto existente 109
8.1.2.2. Guardar proyecto 111
8.1.2.3. Impresión de resultados 111
8.1.2.4. Modificación de datos 113
8.1.2.5. Ejecución estándar.... 113
8.1.2.6. Visualización resultado corrida estandar 115
8.1.2.7. Salida del programa 116
8.1.2.8. Acerca.... 117
8.2. EJECUCIÓN PARAMÉTRICA 118
8.2.1. Visualización de resultados corrida paramétrica 119

ANEXO I....121

ANEXO II....130

1. RESOLUCION DEL FLUJO DE CARGA MEDIANTE SOFTWARE....130

1.1. *INTRODUCCIÓN* 130
1.2. *REORDENAMIENTO DE LAS UNIDADES PASCAL* 131
1.3. *IMPLEMENTACIÓN DEL MÉTODO DE NEWTON-RAPHSON* 131

1.3.1. UNIDAD NR2 134
1.3.2. UNIDAD SERVICE2 140
1.3.3. UNIDAD ALGEBRAC 147
1.3.4. UNIDAD XMATDEFS 150
1.3.5. UNIDAD ECUACS 151
1.3.6. UNIDAD USISTEMA 155
1.3.7. UNIDAD MATCPX 160
1.3.8. UNIDAD MATOBJ 164
1.3.9. UNIDAD HORRORES 167
1.3.10. UNIDAD LINKS1 168
1.3.11. UNIDAD TDFS0 172
1.3.12. UNIDAD LEXEMAS 173
1.4. MODELADO DEL PROBLEMA UTILIZANDO OBJETOS 175
1.4.1. UNIDAD BARRS2 175
1.4.2. UNIDAD IMPDS1 179
1.4.3. UNIDAD CUADRI1 182
1.4.4. UNIDAD TRAFOS1 185
1.4.5. UNIDAD REGULADO 189
1.4.6. UNIDAD MATADM1 191
1.5. CÁLCULO DE LOS FLUJOS DE POTENCIA A TRAVÉS DE LAS LÍNEAS 196
1.6. LA FUNCIÓN OBJETIVO DEL FLUJO DE CARGA 197
1.7. EL PROGRAMA PRINCIPAL: FLUCAR 3.0 227
1.7.1. UNIDAD TYVS2 233
1.7.2. UNIDAD CRONOMET 235
1.8. *LA ENTRADA Y SALIDA DE DATOS* 237
1.8.1. ENTRADA DE DATOS 237
1.8.2. SALIDA DE DATOS 243

CAPITULO 1

1. ESTUDIO DEL FLUJO DE CARGA

1.1. Presentación del problema

Hace unos 40 años se utilizaban modelos a escala para poder estudiar el comportamiento de una red eléctrica de potencia. Con el advenimiento de las computadoras PC, la reducción en dimensiones y en costos de las mismas, y las mejoras en los tiempos de ejecución y facilidad de desarrollo, estos analizadores de red se han sustituido por simulaciones en computadoras.

El estudio del flujo de carga, también llamado flujo de potencia, distribución de carga, etc., consiste en la determinación de voltajes, intensidades, potencias activas y reactivas en distintos puntos de una red eléctrica. Se consideran sistemas en régimen, equilibrados, sinusoidales, sin anomalías y se trabaja entre fase y neutro. Los resultados que se obtienen son, generalmente, el módulo y la fase de la tensión en cada barra, así como las potencias activa y reactiva entrantes en cada una de ellas.

Estos resultados pueden ser utilizados para:

- Evaluar el comportamiento del sistema existente en condiciones estacionarias normales o anormales.
- Estudiar alternativas para la planificación de nuevos sistemas o ampliación de los ya existentes.
- Estudiar la estabilidad transitoria y permanente de sistemas de potencia
- Elaborar plan de contingencias ante fallo de un elemento de la red.

A diferencia de los problemas considerados en la Teoría General de Circuitos, cuya solución consistía, utilizando el método de nudos y mallas, en la resolución de un sistema de ecuaciones algebraicas lineales, en una red de potencia, las ecuaciones que ligan las incógnitas son no lineales, por lo cual deberemos valernos de los métodos matemáticos más recientes del cálculo numérico. Estos métodos, en general iterativos, permiten una rápida resolución al problema.

1.2. Formulación del problema

1.2.1. Condiciones iniciales y tipos de barras

Se tienen 4 variables reales asociadas a cada una de las barras:

- P Potencia Activa
- Q Potencia Reactiva
- V Módulo de la tensión respecto al neutro N del sistema
- θ Ángulo de fase de la tensión

En cada una de las barras, 2 de estas magnitudes se supondrán inicialmente conocidas y las 2 restantes serán incógnitas del problema. Las distintas combinaciones de incógnitas y datos nos permiten definir los siguientes 4 tipos de barras:

- **Barra de carga** : Se conocen la potencia activa y reactiva entrantes
- **Barra de generación y voltaje controlado** : Se conoce la potencia activa y el módulo de la tensión
- **Barra flotante** : Se conocen el módulo y la fase de la tensión
- **Barra de referencia:** se consideran nulas las 4 variables reales asociadas a ella.

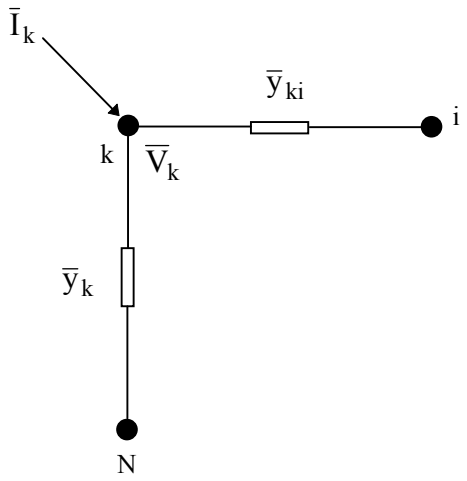
Existe una única barra de referencia por sistema que es el neutro N del sistema.

1.2.2. Numeración de las barras

A los efectos de organizar el desarrollo del cómputo, se asignará la siguiente numeración de las barras :

- 0 Barra de referencia (neutro del sistema)
- 1.. ℓ Barras de Carga
- ℓ +1..n-1 Barras de Generación y Voltaje controlado
- n Barra Flotante

1.2.3. Estudio del problema para un nodo k genérico



k = 1,2,.....,n

i = 1,2,.....,n $i \neq k$

$\bar{I}_k$ = corriente entrante al nodo k

$$\bar{I}_k = \bar{V}_k \bar{y}_k + \sum_{i=1, i\neq k}^{n} (\bar{V}_k - \bar{V}_i)\bar{y}_{ki} \qquad k = 1,2,.....n$$

$$\bar{I}_k = \bar{V}_k \bar{y}_k + \bar{V}_k \sum_{i=1, i\neq k}^{n} \bar{y}_{ki} - \sum_{i=1, i\neq k}^{n} \bar{V}_i \bar{y}_{ki}$$

$$\bar{I}_k = \bar{V}_k (\bar{y}_k + \sum_{i=1, i\neq k}^{n} \bar{y}_{ki}) - \sum_{i=1, i\neq k}^{n} \bar{V}_i \bar{y}_{ki}$$

se define :

$$\bar{y}_k + \sum_{i=1, i\neq k}^{n} \bar{y}_{ki} = \bar{Y}_{kk}$$

$$-\bar{y}_{ki} = \bar{Y}_{ki}$$

y entonces :

$$\bar{I}_k = \bar{V}_k \bar{Y}_{kk} + \sum_{i=1, i\neq k}^{n} \bar{V}_i \bar{Y}_{ki} = \sum_{i=1}^{n} \bar{V}_i \bar{Y}_{ki}$$

$$\boxed{\bar{I}_k = \sum_{i=1}^{n} \bar{V}_i \bar{Y}_{ki} \qquad k = 1,2,....,n}$$

La potencia entrante en el nodo k será : (balance en el nodo k entre lo aportado por un generador y lo absorbido por una carga)

$$\boxed{P_k + jQ_k = \bar{V}_k \hat{I}_k \qquad k = 1,2,....,n}$$

1.2.4. Formulación de las ecuaciones de comportamiento de la red

Como se dedujo en el punto anterior, las ecuaciones para el nodo k son :

$$\bar{I}_k = \sum_{i=1}^{n} \bar{V}_i \bar{Y}_{ki} \qquad k=1,2,....,n \quad (1.1)$$

$$P_k + jQ_k = \bar{V}_k \hat{I}_k \qquad k=1,2,....,n \quad (1.2)$$

donde :

- $\bar{V}_i$ es el fasor voltaje de nodo medido con respecto al nodo de referencia
- $\bar{I}_k$ es el fasor corriente equivalente inyectado al nodo k
- n es el número total de nodos, excluído el de referencia
- $P_k > 0$: es la potencia activa efectivamente inyectada a la red en el nodo k
- $P_k < 0$: $-P_k$ es la potencia activa que efectivamente sale de la red por el nodo k
- $Q_k > 0$: es la potencia reactiva efectivamente inyectada a la red en el nodo k
- $Q_k < 0$: $-Q_k$ es la potencia reactiva que efectivamente sale de la red en el nodo k

La numeración de las barras definida anteriormente puede esquematizarse de la siguiente forma :

NODO	TIPO DE BARRA	DATOS	INCÓGNITAS
0	Barra de Referencia	$\|V\| = 0 \quad \theta = 0$ $P = 0 \quad Q = 0$	
1,2,...., ℓ	Barras de Carga	$P_{ks} \quad Q_{ks}$	$V_k \quad \theta_k$
$\ell+1, \ell+2, ...,$n-1	Barras de Generación y de Voltaje Controlado	$P_{ks} \quad V_{ks}$	$Q_k \quad \theta_k$
n	Barra Flotante	$V_{ns} \quad \theta_{ns}$	$P_n \quad Q_n$

reemplazando (1.1) en (1.2) :

$$P_k + jQ_k = \overline{V}_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \qquad (1.3)$$

separando parte real y parte imaginaria :

$$P_k = \mathrm{Re}\left(\overline{V}_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \right) \qquad (1.4)$$

$$Q_k = \mathrm{Im}\left(\overline{V}_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \right) \qquad (1.5)$$

Como la potencia activa es dato en las ℓ barras de carga y en las n-1-ℓ barras de generación (excluída la flotante), se pueden plantear las siguientes ecuaciones :

$$P_{ks} = \mathrm{Re}\left(V_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \right) \qquad k = 1,2,.....,n-1 \qquad (1.6)$$

La potencia reactiva es dato en las ℓ barras de carga, entonces :

$$Q_{ks} = \mathrm{Im}\left(V_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \right) \qquad k = 1,2,....., \ell \qquad (1.7)$$

El conjunto de las n-1 ecuaciones (1.6) y de las ℓ ecuaciones (1.7), constituyen un sistema de n-1+ℓ ecuaciones no lineales en las incógnitas fasoriales $\overline{V}_k$. De estas incógnitas fasoriales son conocidos los módulos de voltaje de las n-ℓ barras de generación (incluída la flotante) y el ángulo de fase en la barra flotante.

Entonces las incógnitas son los módulos de voltaje en las ℓ barras de carga y los ángulos de fase en todas las barras, excepto la flotante. Estas n-1 incógnitas de ángulo de fase, con las ℓ incógnitas de módulos de voltaje, suman un total de n-1+ℓ incógnitas.
El estudio del flujo de carga consiste, entonces, en resolver el sistema de ecuaciones precedente en las incógnitas señaladas. Una vez conocidos los voltajes en todas las barras, se obtienen en forma directa las potencias reactivas y activas incógnitas, inyectadas en barras, así como pueden calcularse los flujos de potencia activa y reactiva y las corrientes en las líneas.
La naturaleza del sistema de ecuaciones establecido no permite obtener una solución directa, debiéndose recurrir a métodos iterativos.

CAPITULO 2

2. CONSIDERACIONES SOBRE EL CALCULO EN POR UNIDAD.

El cálculo en por unidad es lo más usual cuando se corren Flujos de Carga. Permite una mejor visualización de todo el sistema ya que, por ejemplo, los módulos de todas las tensiones tendrán valores cercanos a 1. El usuario debe introducir, en este caso, todas las variables en por unidad. Los resultados también serán devueltos en por unidad.

El objetivo de este punto es repasar las nociones del cálculo en por unidad para que sirvan de guía en el momento de tener que realizar alguna aplicación particular utilizando el programa de Flujo.

Los cálculos de sistemas eléctricos de potencia se simplifica si todas las magnitudes eléctricas (impedancias, voltajes, corrientes, potencias, etc) se expresan como el cociente de la magnitud eléctrica dividida por una base de referencia de la misma magnitud. Este método permite eliminar los distintos niveles de voltaje, estableciendo un circuito equivalente, en el que no aparecen transformadores.

Las magnitudes de base deben elegirse de tal manera que las leyes de Ohm, Joule y Kirchoff sean válidas también en la red equivalente.

$$P.U. = \frac{Magnitud\_eléctrica}{Magnitud\_base}$$

Por lo tanto las magnitudes eléctricas quedarían de la siguiente manera:

$$V_{PU} = \frac{V}{V_B}$$
$$Z_{PU} = \frac{Z}{Z_B}$$
$$Z_B = \frac{V_B^2}{S_B} \Rightarrow Z_{PU} = \frac{Z \times S_B}{V_B^2}$$

2.1. Cálculo de los valores reales a partir de los valores "en por uno" en base V_B

El cálculo se esquematiza en la siguiente tabla:

Magnitud	**Base**	**VUB**	**VR**
Tensión	V_B	V_{PU}	$V_{PU}V_B$
Potencia	P_B	P_{PU}	$P_{PU}P_B$
Corriente	$I_B = \frac{P_B}{V_B}$	I_{PU}	$I_{PU}\frac{P_B}{V_p}$
Impedancia	$Z_B = \frac{V_B{}^2}{P_B}$	Z_{PU}	$z\frac{V_p{}^2}{P_B}$
Admitancia	$Y_B = \frac{P_B}{V_B{}^2}$	Y_{PU}	$y\frac{P_B}{V_B{}^2}$

En un transformador ideal se verifican las siguientes relaciones:

$$V_S = \frac{n_S}{n_P} V_P$$
$$I_S = \frac{n_P}{n_S} I_P$$
$$S_S = S_P$$

Si se eligen las bases del lado del primario del transformador y del secundario del mismo de manera que verifiquen las siguientes relaciones:

$$V_{SB} = \frac{n_S}{n_P} V_{PB}$$
$$I_{SB} = \frac{n_P}{n_S} I_{PB}$$
$$S_{SB} = S_{PB}$$

Con estas relaciones se cumplirá que el voltaje, la corriente y la potencia en el secundario del transformador ideal son iguales al voltaje, la corriente y la potencia en el primario del transformador.

$$\frac{V_S}{V_{SB}} = \frac{V_P}{V_{PB}} \Rightarrow V_{SPU} = V_{PPU}$$
$$\frac{I_S}{I_{SB}} = \frac{I_P}{I_{PB}} \Rightarrow I_{SPU} = I_{PPU}$$
$$\frac{S_S}{S_{SB}} = \frac{S_P}{S_{PB}} \Rightarrow S_{SPU} = S_{PPU}$$

Para el caso de la impedancia de cortocircuito podemos hacer lo siguiente:

$$Z_{CCP} = \left(\frac{n_P}{n_S}\right)^2 Z_{CCS}$$
$$\frac{Z_{PB}}{Z_{SB}} = \frac{\frac{V_{PB}^2}{S_B}}{\frac{V_{SB}^2}{S_B}} = \frac{V_{PB}^2}{V_{SB}^2} = \left(\frac{n_P}{n_S}\right) \Rightarrow Z_{PB} = \left(\frac{n_P}{n_S}\right)^2 Z_{SB}$$
$$Z_{PUP} = Z_{PUS}$$

O sea que la impedancia de cortocircuito del transformador, expresada en por unidad, es la misma, independientemente del devanado a que esté referida.

Como conclusión podemos decir que en un grupo de redes interconectadas mediante transformadores, si las bases de voltajes tomadas para las redes de un lado y otro de estos están en la misma relación que la relación de vueltas, y además se usa una base de potencia común, el circuito equivalente en por unidad de las redes puede interconectarse eliminando los transformadores.

2.2. Transformadores con reguladores

Si tenemos un transformador con varias derivaciones en algunos de los dos devanados, esto permite cambiar la relación de transformación dentro de un cierto rango.

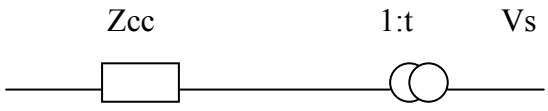
Cuando el número de vueltas del primario se cambia, las bases de voltaje a ambos lados del transformador deberían cambiar de la siguiente manera:

$$V_{PB}^{*} = \frac{n_P^{*}}{n_S} V_{SB}$$

$$Z_{PB}^{*} = \left(\frac{n_P^{*}}{n_S} \right) Z_{PB}$$

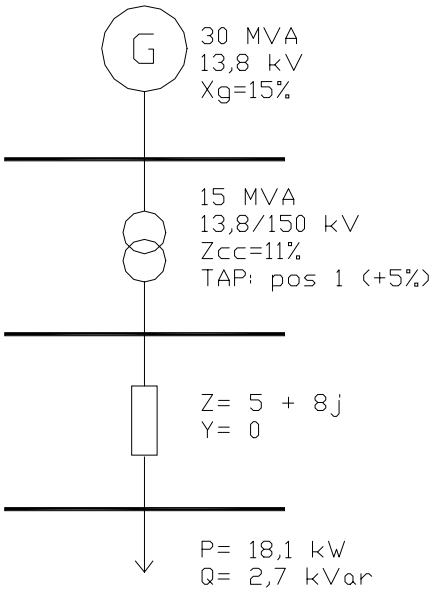
La impedancia de cortocircuito del transformador en por unidad no cambiaría , si se cambiara la bases de voltaje a ambos lados del transformador, adecuándose a la nueva relación de vueltas. Este cambio implicaría cambiar la base de voltaje de todos los elementos de la red. Para evitar esto se hace uso de un auto transformador ideal, el cual cumple lo siguiente:

$$t = \frac{V_{BP} \dfrac{V_S}{V_P}}{V_{BS}}$$



2.3. Ejemplo calculo por unidad

Consideremos el siguiente circuito



Generador:

Ubase= 13,8 kV
Pbase= 30 MVA

$$x_{pu} = 0.15\frac{30}{30} = 0.15j$$

Trafo:

Pbase= 30 MVA
Ubp=13,8 kV
Ubs=150 kV

$$x_{pu} = 0.11\frac{30}{15} = 0.22j$$

Tap: +5% por lo que Us=150*1.05 kV

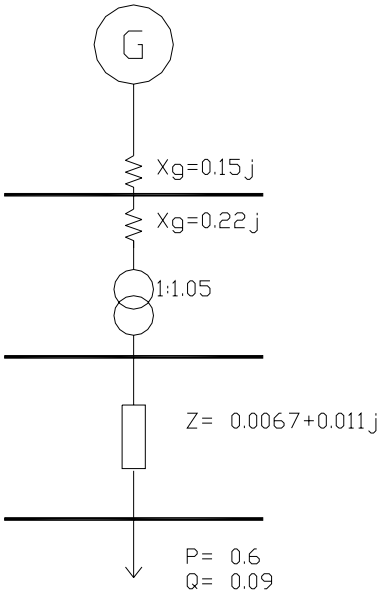
Línea:

$$z_{pu} = (5+8j)\frac{P_{base}}{V_{base}^{\ 2}} = (5+8j)\frac{30MVA}{(150kV)^2} = 0.0067 + 0.011j$$

Cargas:

$$P_{pu} = \frac{18{,}11}{30} = 0.6$$
$$Q_{pu} = \frac{2{,}7}{30} = 0.09$$

El resultado se puede ver en la figura siguiente



CAPITULO 3

3. METODO DE NEWTON RAPHSON

Es un método iterativo que permite resolver sistemas de ecuaciones no lineales. Es muy veloz aunque no siempre converge. Implica un gran número de cálculos en cada iteración ya que debe resolverse un sistema mxm.

Sea el siguiente sistema de m ecuaciones no lineales con m incógnitas $x_1, x_2, \ldots, x_m$:

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_m) &= k_1 \\ f_2(x_1, x_2, \ldots, x_m) &= k_2 \\ &\vdots \\ f_m(x_1, x_2, \ldots, x_m) &= k_m \end{aligned}$$

Vectorialmente puede expresarse :

$$\left[F([X])\right] = [K]$$

Sea $\left[X^0\right]$ una solución inicial aproximada; hacemos el cambio de variable :

$$[X] = \left[X^0\right] + [\Delta X]$$

El sistema de ecuaciones se escribe entonces con incógnitas Δx_1, Δx_2, ..., Δx_m :

$$\begin{aligned} &f_1\left(x^0{}_1 + \Delta x_1, x^0{}_2 + \Delta x_2, \ldots, x^0{}_m + \Delta x_m\right) = k_1 \\ &f_2\left(x^0{}_1 + \Delta x_1, x^0{}_2 + \Delta x_2, \ldots, x^0{}_m + \Delta x_m\right) = k_2 \\ &\vdots \\ &f_m\left(x^0{}_1 + \Delta x_1, x^0{}_2 + \Delta x_2, \ldots, x^0{}_m + \Delta x_m\right) = k_m \end{aligned} \qquad (3.1)$$

Desarrollando en series de Taylor y despreciando los términos de segundo orden y superior el sistema (3.1) se transforma en:

$$\begin{aligned} &f_1\left(\left[X^0\right]\right) + \frac{\partial f_1}{\partial x_1}\left(\left[X^0\right]\right)\Delta x_1 + \frac{\partial f_1}{\partial x_2}\left(\left[X^0\right]\right)\Delta x_2 + \ldots + \frac{\partial f_1}{\partial x_m}\left(\left[X^0\right]\right)\Delta x_m = k_1 \\ &f_2\left(\left[X^0\right]\right) + \frac{\partial f_2}{\partial x_1}\left(\left[X^0\right]\right)\Delta x_1 + \frac{\partial f_2}{\partial x_2}\left(\left[X^0\right]\right)\Delta x_2 + \ldots + \frac{\partial f_2}{\partial x_m}\left(\left[X^0\right]\right)\Delta x_m = k_2 \\ &\vdots \\ &f_m\left(\left[X^0\right]\right) + \frac{\partial f_m}{\partial x_1}\left(\left[X^0\right]\right)\Delta x_1 + \frac{\partial f_m}{\partial x_2}\left(\left[X^0\right]\right)\Delta x_2 + \ldots + \frac{\partial f_m}{\partial x_m}\left(\left[X^0\right]\right)\Delta x_m = k_m \end{aligned}$$

reordenando los términos y en forma matricial se escribe :

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_m} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_m \end{bmatrix} = \begin{bmatrix} \Delta f^0{}_1 \\ \Delta f^0{}_2 \\ \vdots \\ \Delta f^0{}_m \end{bmatrix}$$

o también :

$$\boxed{J_f\left(\left[X^0\right]\right)}\left[\Delta X\right] = \left[\Delta F^0\right] \qquad (3.2)$$

donde $\boxed{J_f\left(\left[X^0\right]\right)}$ es la matriz jacobiana del sistema de ecuaciones, calculada en $\left[X^0\right]$ y $\left[\Delta F^o\right]$ es el vector cuyos elementos son :

$$\Delta f_j^0 = k_j - f_j\left(\left[X^0\right]\right), \text{ para } j = 1,2,....,m$$

El sistema (1) puede resolverse obteniendo :

$$\left[\Delta X^o\right] = \boxed{J_f\left(\left[X^0\right]\right)}^{-1}\left[\Delta F^0\right]$$

con lo cual puede corregirse la solución inicial aproximada para obtener :

$$\left[X^1\right] = \left[X^0\right] + \left[\Delta X^0\right] = \left[X^0\right] + \boxed{J_f\left(\left[X^0\right]\right)}^{-1}\left[\Delta F^0\right]$$

Si $\left[X^0\right]$ es un punto ubicado en un entorno suficientemente pequeño de la solución $\tilde{X}$, $\left[X^1\right]$ está entonces más próximo a $\tilde{X}$.

El uso de $\left[X^1\right]$ en lugar de $\left[X^0\right]$, como una mejor solución aproximada, conduce a un procedimiento iterativo.

La convergencia dependerá de la forma de las funciones $f_j\left(\left[X\right]\right)$ y de la aproximación inicial que se adopte.

Si el proceso resulta convergente entonces, en sucesivas iteraciones, irán disminuyendo las diferencias entre los términos independientes k y los valores computados de las funciones f. Para la v-ésima iteración, en forma simultánea :

$$\left[X^v\right] = \left[X^{v-1}\right] + \boxed{J_f\left(\left[X^{v-1}\right]\right)}^{-1}\left[\Delta F^{v-1}\right]$$

$$\left[\Delta F^v\right] = \left[k\right] - \left[F\left(\left[X^v\right]\right)\right]$$

El proceso puede darse por concluido si cada uno de los elementos de $\left[\Delta F^v\right]$,

en valor absoluto, resulta menor que un índice de precisión prefijado ε:

$$\left|\Delta f_j^{\nu}\right| \le \varepsilon \qquad j = 1,2,....,m$$

Entonces el valor de $\left[X^{\nu}\right]$ resulta ser la solución buscada dentro de la precisión impuesta.

3.1. Aplicación a la resolución de las ecuaciones de comportamiento de la red

Sean :

$$\overline{V}_i = V_i e^{j\theta_i} = V_i\left(\cos\theta_i + j\,\mathrm{sen}\theta_i\right)$$
$$\overline{Y}_{ki} = a_{ki} + jb_{ki}$$

De las ecuaciones (3.1) y (3.2) del punto 1.2.4 y utilizando estas expresiones, se obtiene :

$$P_k + jQ_k = V_k \sum_{i=1}^{n} V_i\left(a_{ki} - jb_{ki}\right)e^{j\left(\theta_k - \theta_i\right)}$$
$$P_k + jQ_k = V_k \sum_{i=1}^{n} V_i\left(a_{ki} - jb_{ki}\right)\left[\cos\left(\theta_k - \theta_i\right) + j\,\mathrm{sen}\left(\theta_k - \theta_i\right)\right]$$

Separando en parte real y parte imaginaria se obtienen :

$$P_k = V_k \sum_{i=1}^{n} V_i\left(a_{ki}\cos\left(\theta_k - \theta_i\right) + b_{ki}\,\mathrm{sen}\left(\theta_k - \theta_i\right)\right) \qquad (3.3)$$

$$Q_k = V_k \sum_{i=1}^{n} V_i\left(a_{ki}\,\mathrm{sen}\left(\theta_k - \theta_i\right) - b_{ki}\cos\left(\theta_k - \theta_i\right)\right) \qquad (3.4)$$

Sea el vector de las 2n variables reales, componentes polares de las tensiones en las n barras activas :

$$\left[X_{2n}\right] = \begin{array}{|c|c|c|c|c|c|c|c|}\hline \theta_1 & \theta_2 & \cdots & \theta_n & V_1 & V_2 & \cdots & V_n \\ \hline \end{array}^{t}$$

Las incógnitas serán los argumentos en todas las barras, excepto la flotante, y los módulos en las barras de carga :

$\theta_1, \theta_2, \ldots, \theta_{n-1}$ n - 1 incognitas

$V_1, V_2, \ldots, V_\ell$ ℓ incognitas

Sea el vector de las desviaciones de fase :

$$[\Delta\theta] = \begin{array}{|c|c|c|c|}\hline \Delta\theta_1 & \Delta\theta_2 & \cdots & \Delta\theta_{n-1} \\ \hline \end{array}^t$$

Sea el vector de las desviaciones relativas de los módulos de la tensión :

$$\left[\frac{\Delta V}{V}\right] = \begin{array}{|c|c|c|c|}\hline \frac{\Delta V_1}{V_1} & \frac{\Delta V_2}{V_2} & \cdots & \frac{\Delta V_\ell}{V_\ell} \\ \hline \end{array}^t$$

Las datos son la potencia activa en n-1 barras y la potencia reactiva en las ℓ barras de carga :

P_{ks} $k = 1,2,\ldots,n-1$

Q_{ks} $k = 1,2,\ldots,\ell$

Las ecuaciones (3.3) y (3.4) pueden escribirse poniendo en evidencia la dependencia funcional en los primeros miembros :

$$P_k\left(\left[X_{2n}\right]\right) = V_k \sum_{i=1}^{n} V_i \left(a_{ki}\cos(\theta_k - \theta_i) + b_{ki}\,\mathrm{sen}(\theta_k - \theta_i)\right)$$

$$Q_k\left(\left[X_{2n}\right]\right) = V_k \sum_{i=1}^{n} V_i \left(a_{ki}\,\mathrm{sen}(\theta_k - \theta_i) - b_{ki}\cos(\theta_k - \theta_i)\right)$$

Para todas las barras excepto la flotante puede escribirse :

$$P_k\left(\left[X_{2n}\right]\right) = P_{ks} \qquad k = 1,2,\ldots,n-1$$

Para las barras de carga :

$$Q_k\left(\left[X_{2n}\right]\right) = Q_{ks} \qquad k = 1,2,\ldots,\ell$$

Siguiendo el procedimiento general antes descripto, se toma una solución inicial estimada para las incógnitas :

$$\theta_1^0, \theta_2^0, \ldots, \theta_{n-1}^0, V_1^0, V_2^0, \ldots, V_\ell^0$$

Luego, se debe tener en cuenta que para las variables datos, los valores incrementales son nulos, así :

$$\Delta\theta_{ns} = 0$$
$$\Delta V_{\ell+1,s} = 0, \ldots, \Delta V_{ns} = 0$$

Desarrollando en series de Taylor hasta el primer orden, se obtiene :

$$P_{ks} - P_k\left(\left[X_{2n}^0\right]\right) = \frac{\partial P_k}{\partial \theta_1}\left(\left[X_{2n}^0\right]\right)\Delta\theta_1 + \ldots + \frac{\partial P_k}{\partial \theta_{n-1}}\left(\left[X_{2n}^0\right]\right)\Delta\theta_{n-1}$$
$$+ \frac{\partial P_k}{\partial V_1}\left(\left[X_{2n}^0\right]\right)\Delta V_1 + \ldots + \frac{\partial P_k}{\partial V_\ell}\left(\left[X_{2n}^0\right]\right)\Delta V_\ell \quad k = 1,2,\ldots, n-1$$

$$Q_{ks} - Q_k\left(\left[X_{2n}^0\right]\right) = \frac{\partial Q_k}{\partial \theta_1}\left(\left[X_{2n}^0\right]\right)\Delta\theta_1 + \ldots + \frac{\partial Q_k}{\partial \theta_{n-1}}\left(\left[X_{2n}^0\right]\right)\Delta\theta_{n-1}$$
$$+ \frac{\partial Q_k}{\partial V_1}\left(\left[X_{2n}^0\right]\right)\Delta V_1 + \ldots + \frac{\partial Q_k}{\partial V_\ell}\left(\left[X_{2n}^0\right]\right)\Delta V_\ell \quad k = 1,2,\ldots, \ell$$

Podemos escribir :

$$\frac{\partial P_k}{\partial V_h}\Delta V_h = \left(\frac{\partial P_k}{\partial V_h} V_h\right)\frac{\Delta V_h}{V_h}$$
$$\frac{\partial Q_k}{\partial V_h}\Delta V_h = \left(\frac{\partial Q_k}{\partial V_h} V_h\right)\frac{\Delta V_h}{V_h} \qquad h=1,2,\ldots,\ell$$

Escribiendo las ecuaciones en forma matricial : (3.5)

$$\begin{bmatrix} P_{1s} \\ \vdots \\ P_{n-1,s} \\ Q_{1s} \\ \vdots \\ Q_{\ell s} \end{bmatrix} - \begin{bmatrix} P_1\left(\left[X_{2n}^0\right]\right) \\ \vdots \\ P_{n-1}\left(\left[X_{2n}^0\right]\right) \\ Q_1\left(\left[X_{2n}^0\right]\right) \\ \vdots \\ Q_\ell\left(\left[X_{2n}^0\right]\right) \end{bmatrix} = \begin{bmatrix} \frac{\partial P_1}{\partial \theta_1} & \cdots & \frac{\partial P_1}{\partial \theta_{n-1}} & \frac{\partial P_1}{\partial V_1}V_1 & \cdots & \frac{\partial P_1}{\partial V_\ell}V_\ell \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial P_{n-1}}{\partial \theta_1} & \cdots & \frac{\partial P_{n-1}}{\partial \theta_{n-1}} & \frac{\partial P_{n-1}}{\partial V_1}V_1 & \cdots & \frac{\partial P_{n-1}}{\partial V_\ell}V_\ell \\ \frac{\partial Q_1}{\partial \theta_1} & \cdots & \frac{\partial Q_1}{\partial \theta_{n-1}} & \frac{\partial Q_1}{\partial V_1}V_1 & \cdots & \frac{\partial Q_1}{\partial V_\ell}V_\ell \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Q_\ell}{\partial \theta_1} & \cdots & \frac{\partial Q_\ell}{\partial \theta_{n-1}} & \frac{\partial Q_\ell}{\partial V_1}V_1 & \cdots & \frac{\partial Q_\ell}{\partial V_\ell}V_\ell \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_{n-1} \\ \frac{\Delta V_1}{V_1} \\ \vdots \\ \frac{\Delta V_\ell}{V_\ell} \end{bmatrix}$$

donde los elementos de la matriz jacobiana (n-1+ ℓ)x(n-1+ ℓ) están calculados en los valores estimados $\left[\mathrm{X}_{2\mathrm{n}}^0\right]$

El primer término contiene las diferencias entre los valores de potencias activas y reactivas reales (dato) y los correspondientes valores calculados a partir de la solución inicial estimada :

$$\begin{bmatrix} [P_s] \\ [Q]_s \end{bmatrix} - \begin{bmatrix} \left[P\left(\left[X_{2n}^0\right]\right)\right] \\ \left[Q\left(\left[X_{2n}^0\right]\right)\right] \end{bmatrix} = \begin{bmatrix} \left[\Delta P\left(\left[X_{2n}^0\right]\right)\right] \\ \left[\Delta Q\left(\left[X_{2n}^0\right]\right)\right] \end{bmatrix}$$

Escribiendo en forma compacta la ecuación 3.5:

$$\begin{bmatrix} \left[\Delta P\left(\left[X_{2n}^0\right]\right)\right] \\ \left[\Delta Q\left(\left[X_{2n}^0\right]\right)\right] \end{bmatrix} = \begin{bmatrix} \mathrm{H}\left(\left[X_{2n}^0\right]\right) & \mathrm{N}\left(\left[X_{2n}^0\right]\right) \\ \mathrm{M}\left(\left[X_{2n}^0\right]\right) & \mathrm{T}\left(\left[X_{2n}^0\right]\right) \end{bmatrix} \begin{bmatrix} [\Delta\theta] \\ \left[\frac{\Delta V}{V}\right] \end{bmatrix}$$

donde los elementos de las submatrices son:

$$H_{kk} = \frac{\partial P_k}{\partial \theta_k} = -b_{kk} V_k^{\,2} - Q_k$$

$$H_{ki} = \frac{\partial P_k}{\partial \theta_i} = V_k V_i \left[a_{ki} \operatorname{sen}(\theta_k - \theta_i) - b_{ki} \cos(\theta_k - \theta_i) \right] \quad k \neq i$$

$$M_{kk} = \frac{\partial Q_k}{\partial \theta_k} = -b_{kk} V_k^{\,2} + P_k$$

$$M_{ki} = \frac{\partial Q_k}{\partial \theta_i} = -V_k V_i \left[a_{ki} \cos(\theta_k - \theta_i) + b_{ki} \operatorname{sen}(\theta_k - \theta_i) \right] \quad k \neq i$$

$$N_{kk} = \frac{\partial P_k}{\partial V_k} V_k = a_{kk} V_k^{\,2} + P_k$$

$$N_{ki} = \frac{\partial P_k}{\partial V_i} V_i = V_k V_i \left[a_{ki} \cos(\theta_k - \theta_i) + b_{ki} \operatorname{sen}(\theta_k - \theta_i) \right] \quad k \neq i$$

$$T_{kk} = \frac{\partial Q_k}{\partial V_k} V_k = Q_k - b_{kk} V_k^{\,2}$$

$$T_{ki} = \frac{\partial Q_k}{\partial V_i} V_i = V_k V_i \left[a_{ki} \operatorname{sen}(\theta_k - \theta_i) - b_{ki} \cos(\theta_k - \theta_i) \right] \quad k \neq i$$

Si resultara, para un ε prefijado :

$$\begin{aligned} &\left| \Delta P_k \left(\left[X_{2n}^0 \right] \right) \right| \leq \varepsilon \qquad k = 1,2,\ldots,n-1 \\ &\left| \Delta Q_h \left(\left[X_{2n}^0 \right] \right) \right| \leq \varepsilon \qquad h = 1,2,\ldots,\ell \end{aligned} \tag{3.6}$$

entonces la solución inicial sería aceptable y terminaría el método.

Si eso no ocurriese, las desviaciones de las potencias activa y reactiva se usarán para calcular las correcciones de las variables estimadas, por medio de la ecuación (3.5). De esta manera, sumando a los valores estimados las correcciones obtenidas se hallan nuevos valores estimados:

$$\begin{aligned} \theta_k^1 &= \theta_k^0 + \Delta\theta_k^0 \qquad k = 1,2,\ldots,n-1 \\ V_h^1 &= V_h^0 + \Delta V_h^0 \qquad h = 1,2,\ldots,\ell \end{aligned}$$

El conjunto de las variables dato y las variables estimadas corregidas, constituyen un nuevo vector de variables: $\left[X_{2n}^1 \right]$

De esta forma el proceso iterativo continúa hasta que se cumplan las condiciones (3.4) o se alcance un numero máximo de iteraciones preestablecido.

CAPITULO 4

4. ESTUDIO DEL FLUCAR 2.O Y OTROS PROGRAMAS DE CALCULO DE FLUJOS DE CARGA COMERCIALES

En esta etapa se estudiará el FLUCAR 2.0 a nivel de usuario, además de otros 2 flujos de carga comerciales, EPRI, utilizado en Despacho Nacional de Carga y SWEDNET usado en distribución Montevideo con el cual se hiciera el estudio de factibilidad de cambio y unificación de tensión en la red de MT de UTE.

4.1. ESTUDIO DEL FLUCAR 2.0

4.1.1. Datos de entrada

La interacción del programa FLUCAR con el usuario se da a través de dos archivo de texto. Uno dedicado a la entrada de datos (extensión .dat) y otro para la salida de datos (extensión .res).
Los datos de entrada se organizan de la siguiente manera:

```
{           EJEMPLO 1 DEL FLUCAR
            RED FICTICIA


      A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo     P        Q         V   delta   Vmin  Vmax  Qmin  Qmax }

            J30.        1     0         0         1    0       N     N
            321A        2    -0.5      -0.3       1    0       N     N
            022ª        2    -0.7      -0.4       1    0       N     N
            021A        2     0         0         1    0       N     N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                                 Tipo 2:  Barra de carga (Datos: P,Q)
                                 Tipo 3:  Barra de generaci¢n (Datos: P,V)
                                 Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor inicial
                                 del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites £nicamente
             (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
             flujo sin l¡mites en una barra se debe escribir la letra N (no
             hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con la
             £nica diferencia que se hace un ajuste del regulador para ajustar la
             tensi¢n.  Si una barra con regulador se corre como tipo 2 no se har
             ajuste de la tensi¢n tom ndose el regulador como un trafo de relaci¢n
             constante e igual al valor de n inicial
}


{+IMPEDANCIAS}
{
                                          Z
                        b1---------/////////---------b2

}


                        { Nombre   bs     bll    Z      Imax }


                          {  Z1     S.J.    N    0+j1      0  }
```

```
+CUADRIPOLOSPI
 {                                        Z12
                                  b1 -------////////------- b2
                                     -              -
                                     /              /
                                 Y13 /              / Y23
                                     /              /
                                     -              -
                                     N              N
 }


          { b1    b2    b3              Y13                        Z12                            Y23                          Imax}
 cua001  J30.   321A    N    0+j0.014030414  0.003587713783+j0.002735222979   0+j0.01403041414                              0
 cua002  J30.   021A    N    0+j0.023377409  0.0017938322+j0.002519502143     0+j0.023377409                                0


+TRAFOS
{
                                                 1:n
                                                 - -  - -      Zcc
                                      Nodo1  ----- -     -     - ----/////---- Nodo2
                                                 - -  - -
}

                { Nombre     b1    b2      n     Zcc        Imax }
                  traf001    021A  022A    1   0+j0.11       0



{Nota:      La corriente m xima admisible por una impedancia, cuadripolo,
            o transformador debe anotarse debajo de Imax.  En el caso de
            no querer establecer restricci¢n sobre la corriente escribir
            0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de partida.
            nmin es el valor m¡nimo de n admisible.
            nmax es el valor m ximo de n admisible.
            deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre  b1    b2            n   nmin     nmax    deltan    Zcc       Imax
traf002   MVD5  MVD1   1.12  0.95       1.2     0.005     0+j0.03       0
traf006   MVD5  MVD2   1.12  0.95       1.2     0.005     0+j0.03       0
traf007   MVD5  MVD3   1.12  0.95       1.2     0.005     0+j0.03       0}

+TOLERANCIA     0.001

+NITS        50

+FIN.
```

Figura 4.1

Los datos se ingresan de la siguiente manera:

- Barras, con los datos que corresponda según el tipo de barra. Además existe la posibilidad de ingresar valores máximos y mínimos de Q y de V

- Impedancias, nodos de origen y destino, impedancia y corriente máxima admisible , si es necesario.
- Lineas. Estas deben ingresarse en forma de cuadripolo PI. Tambien permiten ingresar la corriente máxima admisible.
- Transformadores. Se ingresan los datos necesarios, relación de vueltas, Zcc y corriente máxima admisible.
- Reguladores. Se ingresan la relacion mínima y máxima de vueltas del tap, salto del tap y Zcc.
- Para finalizar se deben ingresar la tolerancia exigida , y la máxima cantidad de iteraciones permitida.

4.1.2. Datos de salida

Los datos de salida se organizan de la siguiente manera:

```
P R O Y E C T O  F L U C A R

          I I E  -  1 9 9 7


DATOS: EJEMPLO1.dat

RESULTADOS: EJEMPLO1.res


 RESULTADOS:

b: J30.   S:   1.2024552 + j  0.7076328  V:  1.0000000(  0.0000000ø)
b: 321A  S:  -0.5000000 + j  -0.3000000  V:  0.9974170( -0.0196100ø)
b: 022A  S:  -0.7000000 + j  -0.4000000  V:  0.9478546( -4.7251897ø)
b: 021A  S:   0.0000000 + j  0.0000000  V:  0.9975884( -0.0542807ø)
 TSGeneracion:  1.2024552 + j  0.7076328
 TSConsumo:     1.2000000 + j  0.7000000

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001   0.5011966 + j  0.2729239
321A->cua001  -0.5000000 + j  -0.3000000

J30.->cua002   0.7012586 + j  0.4347089
021A->cua002  -0.7000000 + j  -0.4795834

Perdidas Joule en los cuadripolos:  0.0024552

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
021A->traf001   0.7000000 + j  0.4795834
022A->traf001  -0.7000000 + j  -0.4000000

Perdidas Joule en los transformadores:  0.0000000

 REGULADORES
Perdidas Joule en los reguladores:  0.0000000

Perdidas Joule totales en las l¡neas:  0.0024552
```

```
VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 3 converge: TRUE Tiempo:   0.0000s
```

Figura 4.2.

Como se ve en la Figura 4.2, en los resultados se presentan las potencias entrantes a las barras, sus tensiones correspondientes, potencias entregadas y pérdidas Joule en las impedancias, cuadripolos y transformadores, violaciones de cualquiera de los límites y cantidad de iteraciones realizadas.

4.2. Estudio del EPRI

El segundo flujo de carga que se analizó fue el EPRI, creado por Electric Power Research Institute, USA. Este es un flujo de carga que fue utilizado por el Despacho Nacional de cargas. La forma que se ingresa los datos, y los resultados de las corridas tienen cierta similitud con el FLUCAR.

4.2.1. Datos de entrada

La interacción del programa EPRI con el usuario se da a través de dos archivo de texto. Uno dedicado a la entrada de datos (extensión .flu) y otro para la salida de datos (extensión .sal).

Los datos de entrada se organizan de la siguiente manera:

```
HDG
 Caso Base: EJEMPLO 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
 a
 a
BAS
C Definicion de areas:
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
C  NNNNNNNNNnnnnnnnnvvvv SSSSSSSS Z1 Z2 Z3 Z4 Z5 Z6 Z7 Z8 Z9 Z0       VVVVvvvv
C                MJ      E2     31.5             0 MJ                                  31.531.5
C 345678901234567890123456789012345678901234567890123456789012345678901234567890
C booonnnnnnnnvvvvzzpppppqqqqqllllccccPPPPpppppQQQQQqqqqqVVVVvvvvNNNNNNNNtttt%%%
```

```
B          MJ     31.5                        15                          1000
B          E2_6    6.3                                -7  -4
B          E2_30  31.5
B          E32    31.5                                -5  -3
C ********************************************************************************
C 3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
C looonnnnnnnnvvvvmNNNNNNNNVVVVisIIIInrrrrrrrxxxxxxxgggggbbbbbbbbmmmmCCCCCCCCmaamaa
L           MJ    31.5  E2_30         31.5          0.01780.025          0.000047
L           MJ    31.5  E32           31.5          0.03560.02714        0.000028

T           E2_6   6.3  E2_30         31.5     999           1.1
ZZ
TOL
                      0.100000.05000
REM   2
A Caso FLUJO BASE
 Incr. al 2000: 1.0000001
AZC
ZZ
SOL00001              50  MJ     31.5    0.
OUT   0   2   0          YES
SAV   8 9999 9999 9999 9999  MC3FXH
REM   1
MAP   0
IND
END
```

Figura 4.3.

Como se ve en la Figura 4.3, los datos se ingresan de la siguiente manera:

- Areas, permite diferenciar entre diferentes áreas y zonas de trabajo, esto es útil a nivel informativo para cuando se maneja gran cantidad de datos
- Barras, con los datos que corresponda según el tipo de barra. Además existe la posibilidad de ingresar valores máximos y mínimos de P, Q y de V
- Lineas. Estas deben ingresarse en forma de cuadripolo PI. También permiten ingresar la corriente máxima admisible.
- Transformadores. Se ingresan los datos necesarios, relación de vueltas, Zcc .
- Para finalizar se debe ingresar la tolerancia exigida .

A diferencia del FLUCAR, el ingreso de los datos se permite hacer de forma desordenada. Esto da cierta flexibilidad para ingresar los datos por niveles de tensión, zonas geográficas o algún otro criterio.

4.2.2. Datos de salida

Los datos de salida se organizan de la siguiente manera:

```
         ********************************************************************************
         ********************************************************************************
         **                                    **
         **    E L E C T R I C  P O W E R  R E S E A R C H  I N S T I T U T E     **
         **                                    **
         **         3412 HILLVIEW AVENUE - PALO ALTO - CALIFORNIA 94304          **
         **                                    **
         ********************************************************************************
         **                                    **
         **                                    **
         **                  R P  7 4 5                **
         **                                    **
         **   T R A N S I E N T / M I D T E R M  S T A B I L I T Y  P A C K A G E   **
         **                                    **
         **                A U G U S T  1 9 7 8             **
         **                                    **
         **                                    **
         **          1000 B U S  P O W E R  F L O W  P R O G R A M         **
         **                                    **
         **       RELEASE  1        VERSION  1        PTF  0       **
         **                                    **
         ********************************************************************************
         **                                    **
         **        C O R R E G I D O  Y  A M P L I A D O  E N  E L         **
         **                                    **
         **        D E S P A C H O  N A C I O N A L  D E  C A R G A S         **
         **                                    **
         **          A G U A  Y  E N E R G I A  E L E C T R I C A           **
         **                                    **
         **        C.C. 19 - 2121 PEREZ - PCIA. SANTA FE - REP. ARGENTINA         **
         **                                    **
         ********************************************************************************
         ********************************************************************************

         **                                    **
         **             VERSION ADAPTADA A COMPUTADOR VAX-11750              **
         **                                    **
         **             - DESPACHO NACIONAL DE CARGAS - UTE -                **
         **                                    **
         **          Camino Peixoto s/n - Melilla - Montevideo - Uruguay           **
         **                                    **
         ********************************************************************************
         ********************************************************************************

1
0CONTROL CARD HDG   0   0   0   0   0                0.0000   0.0000
1Caso Base: EJEMPLO 1  asdfasfd asdfadf dfaci¿n montevideo j rscxh00.flu
 a
 a
0CONTROL CARD BAS   0   0   0   0   0                0.0000   0.0000



 ENTERING DATA INPUT SUBPROGRAM  --  PROCESS BUS DATA CARDS
0COMMENT CARD

     Definicion de areas:
```

```
0COMMENT CARD

     3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
0COMMENT CARD

      NNNNNNNNNNnnnnnnnnvvvv SSSSSSSS Z1 Z2 Z3 Z4 Z5 Z6 Z7 Z8 Z9 Z0       VVVVvvvv
0COMMENT CARD

      MJ      E2     31.5       0 MJ                          31.531.5
0COMMENT CARD

     3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
0COMMENT CARD

     booonnnnnnnnvvvvzzpppppqqqqqllllccccPPPPpppppQQQQQqqqqqVVVVvvvvNNNNNNNNtttt%%%
 *CARD IMAGE*B    MJ     31.5               15             1000
 *CARD IMAGE*B    E2     31.5                 -7   -4
 *CARD IMAGE*B    E32    31.5                 -5   -3
0COMMENT CARD

     ********************************************************************************
0COMMENT CARD

     3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
0COMMENT CARD

     looonnnnnnnnvvvvmNNNNNNNNVVVVisIIIInrrrrrrrxxxxxxxgggggbbbbbbbbmmmmCCCCCCCCmaa




 ALL BUS CARDS PROCESSED  --  PROCESS LINE AND TRANSFORMER CARDS
 *CARD IMAGE*L    MJ     31.5 E2     31.5      0.01780.025       0.000047
 *CARD IMAGE*L    MJ     31.5 E32    31.5      0.03560.02714     0.000028




 ALL SYSTEM DATA PROCESSED.

 THERE WERE    0 BUS ERRORS,    0 LINE ERRORS,    0 UNRECOVERABLE ERRORS, AND    0 PROGRAM
ERRORS.




 *MESSAGE   THIS SYSTEM CONTAINS    3 BUSES AND    2 LINES.
0CONTROL CARD TOL    0    0    0    0    0               0.0000    0.0000
0CONTROL CARD REM    2    0    0    0    0               0.0000    0.0000

  Caso FLUJO BASE

 Incr. al 2000: 1.0000001
0CONTROL CARD AZC    0    0    0    0    0               0.0000    0.0000
1ENTERING BUS DELETE,LOAD CHANGE AND GENERATION CHANGE ROUTINE
0CARD  DATA--ZZ                 0.00    0   0.00
0CONTROL CARD SOL    1    0    0    0   50  MJ     31.5      0.0000    0.0000
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
 a
 a
0THE MAXIMUM NO. OF BRANCHES IS     2

         BEGIN SOLUTION WITH    3 BUSES AND    2 LINES.                          1/ 1/1970 -3: 0: 0

 ITERATION  SUM ABS(DELP)  SUM ABS(DELQ)  BUSES UNSOLVED  MATRIX SIZE       TCUL REG. TIES  DC
     1          0.120          0.070          2              7            0    0    0    0
     2          0.000          0.000          0              7            0    0    0    0
```

```
     3         0.000         0.000         0             7             0   0   0   0   0

          CASE SOLVED.
+                                                                        1/ 1/1970 -6: 0: 0
0ENTERING TAPEA
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu                    SUMMARY REPORT PAGE NO.
1
 a                                                                        GENERAL SUMMARY
 a                                                                      1/ 1/1970 -6: 0: 0

      TYPE OF BUS              ACTUAL  MAXIMUM                TYPE OF LINE (SECTION)      ACTUAL
MAXIMUM
    P AND E - NO Q LIMITS     (BE)    0     1000              BUS TIES              (Z=0)     0     200
    P AND Q - NO V LIMITS     (B)     3     1000              FIXED TAP TRANSFORMERS (T)      0     2000
    DC TERMINAL BUSES         (BD)    0     20                FIXED PHASE SHIFT     (TP)      0     100
                                                              VARIABLE TAP OR PHASE (R)       0     100
 TOTAL NON-REGULATING BUSES             3     1000            DC LINES - TWO TERMINAL (LD)    0     10
                                                              LINE EQUIVALENTS      (E)       0     2000
    P AND E - SELF VAR LIMITS (BQ)    0     666               NORMAL LINES                    2     2000
    P AND E - USE REMOTE VARS (BC)    0     50
    P AND Q - V LIMITS        (BV)    0     666               TOTAL LINE SECTIONS             2     2000
    P - SUPPLY REMOTE VARS    (BG)    0     250
                                                              TOTAL BRANCHES                  2     2000
 TOTAL REGULATING BUSES                 0     666
                                                                MISCELLANEOUS INFORMATION
 TOTAL BUSES IN THIS STUDY              3     1000            NO. OF ITERATIONS               3     50
                                                              NO. OF INTERCHANGE AREAS        0     60
 ASSOCIATED WITH NON-REG. BUSES ARE                           NO. OF ZONES                    1     90
    P AND Q - V BY REG. XFMR  (BT)    0     100             NO. OF BUSES OUT OF TOLERANCE   0     0
    SWITCHED REACTANCE BUSES  (BX)    0     100               TOLERANCE AT EACH BUS - MW OR MVAR
0.010

    P. U. VOLTAGE ARE 0.70 TO 1.50                          TIE LINE CONTROL NOT EXERCISED.



                       SYSTEM DATA     MEGAWATTS   MEGAVARS

                    TOTAL LOAD              0.00       0.00
                    LOSSES-I*I(R+X)         0.02       0.03
                    CHARGING-E*E(Y-PI)      0.01       0.00
                    NET  ADM                0.00       0.00

                    SYSTEM MISMATCH         0.00       0.00

                    TOTAL GENERATION        0.03       0.03
                    MAXIMUM CAPACITY       15.00
                    RESERVE CAPACITY       14.97




      NOTE-- CHARGING INCLUDES REAL AND REACTIVE COMPONENTS OF POWER (P+JQ) DERIVED FROM
EQUIVALENT Y-PI LINES.
          SOME ARE DEVELOPED INTERNALLY FROM COMPOSITE LINES COMBINED USING ABCD
CONSTANTS.
          P(CHARGING) IS TREATED AS MW LOAD.
          Q(CHARGING) IS TREATED AS MVAR LOAD.
          BOTH MAY BE EITHER POSITIVE OR NEGATIVE.



                          SYSTEM SLACK BUS IS MJ     31.5
0CONTROL CARD OUT   0   2   0   0   0 YES              0.0000   0.0000
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu                    SUMMARY REPORT PAGE NO.
1
```

```
 a                                                          AREA SUMMARY (         )
 a                                                              1/ 1/1970 -6: 0: 0

                        SUMMARY OF GENERATOR DATA

                                                ACTUAL    DESIRED
  BUS NAME          MW       MW      MVAR     MVAR LIMITS    VOLTAGE   VOLTAGE
                   PMAX              LOW      HIGH            NOT HELD
 E2     31.5       -7.00     0.00    -4.00    (P+Q NO LIMITS)
 MJ     31.5       12.03    15.00     7.03    (P+Q NO LIMITS)
 E32    31.5       -5.00     0.00    -3.00    (P+Q NO LIMITS)

                    SUMMARY OF    0 BUSES WITH VOLTAGE OVER  1.05 PER UNIT

                    SUMMARY OF    0 BUSES WITH VOLTAGE UNDER 0.95 PER UNIT

                              THERE WERE NO OVERLOADED  TRANSMISSION LINES

                              THERE WERE NO OVERLOADED TRANSFORMERS

                              THERE WERE NO REGULATING TRANSFORMERS
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu                    SUMMARY REPORT PAGE NO.
2
 a                                                          OWNER LOSS SUMMARY
 a                                                              1/ 1/1970 -6: 0: 0

                  SUMMARY OF LOSSES BY OWNER IN MW

     OWNER  LOSSES      OWNER  LOSSES      OWNER  LOSSES      OWNER  LOSSES

          0.03
       ********** TOTAL SYSTEM LOSSES ARE    0.03 MW **********
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu                  POWER FLOW REPORT    PAGE NO.
1
 a                                                      ZONE (  ) OF AREA (         )
 a                                                              1/ 1/1970 -6: 0: 0
0X------------------------------BUS DATA------------------------------X----------------------LINE DATA----------------------X
  NAME-BASE    P.U.      GENERATION        LOAD          SHUNT     ID     TO        LINE FLOWS      LINE LOSSES
PCT
  ACTUAL KV    ANGLE     MW     MVAR     MW     MVAR     MW     MVAR      BUS NAME      MW     MVAR     MW
MVAR   OVL

 E2      31.5  0.9977     -7.0    -4.0     0.0     0.0     0.0     0.0 ------------------------------------------------------
     31.4 KV    -0.1                                                    MJ      31.5    -7.0    -4.0    0.01    0.02
 MJ      31.5  1.0000     12.0     7.0     0.0     0.0     0.0     0.0 ------------------------------------------------------
     31.5 KV     0.0                                                    E2      31.5     7.0     4.0    0.01    0.02
                                                                   E32     31.5     5.0     3.0    0.02    0.01
 E32     31.5  0.9974     -5.0    -3.0     0.0     0.0     0.0     0.0 ------------------------------------------------------
     31.4 KV     0.0                                                    MJ      31.5    -5.0    -3.0    0.02    0.01

                                  END OF REPORT FOR THIS CASE
1
0CONTROL CARD SAV    8 9999 9999 9999 9999  MC3FXH            0.0000    0.0000
1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
 a
 a
0 SE INICIALIZA UN NUEVO ARCHIVO HISTORICO CON EL NOMBRE IDENTIFICATORIO MC3FXH

0********** THIS WILL BE FIRST RECORD ON MC3FXH
 DATA FROM THE PREVIOUS CASE HAS BEEN SAVED ON TAPE HANDLER- 8 WITH CASE IDENTITY MC3FXH
AS RECORD NO- 1.


1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
 a
 a
0CONTROL CARD REM    1    0    0    0    0                0.0000    0.0000
```

```
 AP   0
0CONTROL CARD IND   0   0   0   0   0              0.0000   0.0000
0CONTROL CARD END   0   0   0   0   0              0.0000   0.0000
```

Figura 4.4

Tal como se muestra en la figura 4.4, en los resultados se presentan las potencias entrantes a las barras, sus tensiones correspondientes, potencias entregadas y pérdidas Joule. También se presentan violaciones de cualquiera de los límites prefijados, cantidad de iteraciones realizadas y tamaño de la matriz a resolver. Incluye un reporte de errores encontrados durante el procesamiento de los datos ingresados.

4.3. Estudio del SWEDNET

El tercer flujo de carga que estudiamos es el SWEDNET, creado por la empresa Swed Power, Suecia. Este es un programa ampliamente usado en Distribución, SWEDNET es un programa DOS que permite ser manejado en entorno Windows. Este programa maneja redes radiales, anilladas desde 0.2 a 70 kV. Ademas de realizar flujos de carga, incorpora analisis de fallas y optimización. A diferencia de los dos programas estudiados previamente, este incorpora una base de datos en el que se van a encontrar todos los datos técnicos de la red. Esto hace que este programa tenga mayor atractivo para su uso en las empresas. A su vez tiene dos funcionamientos posibles, con ingreso de datos gráficamente o directo a sus bases de datos. La interfase gráfica es Digsilent, programa de diseño CAD.

En la Figura 4.5 se ve a modo de ejemplo la pantalla inicial del programa:

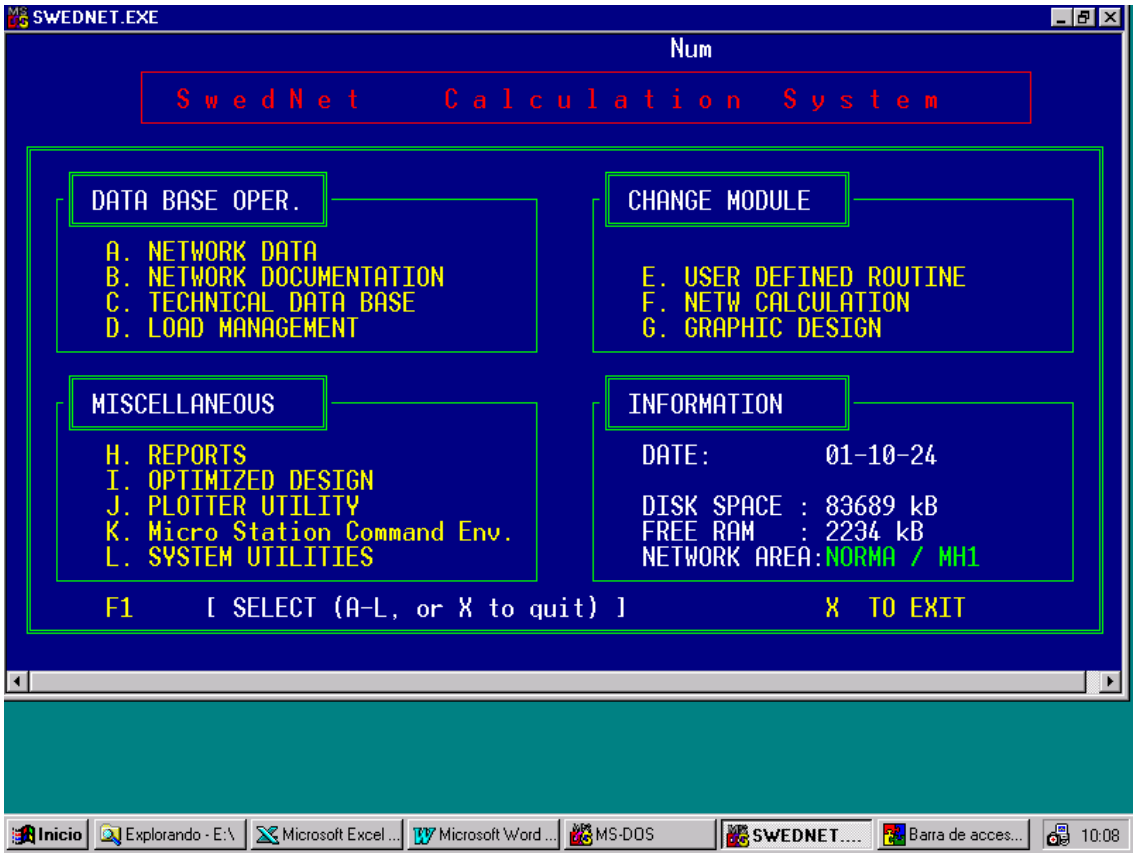


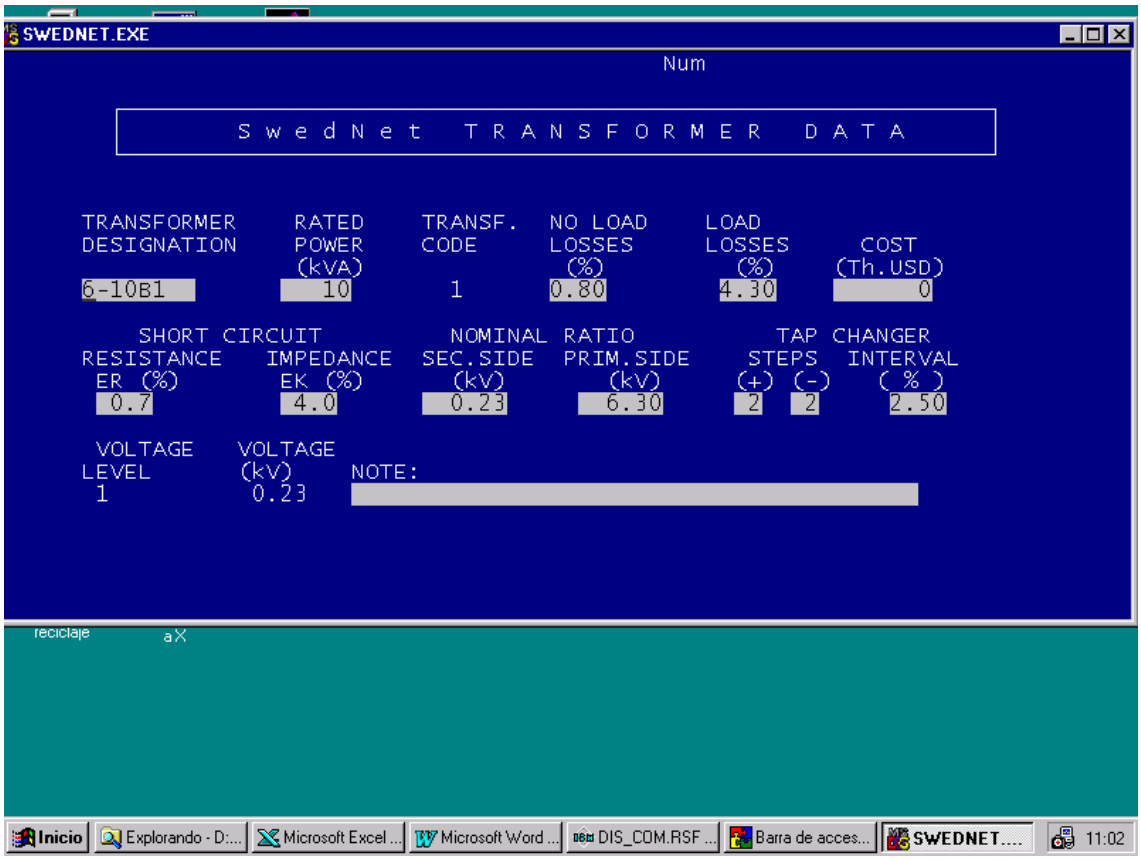
Figura 4.5

4.3.1. Datos de entrada

Los datos de una red de distribución eléctrica son guardados en tres bases de datos diferentes:

- Base de datos de nodos: aquí se encuentran los datos de los nodos, ya sea nombres, cargas, tipos, etc.
- Base de datos técnicos: donde se encuentran los datos de cables y lineas que existen en la red. Además de sección, largo, nodos que están conectados, nivel de voltaje, datos de generadores, capacitores, etc.
- Base de datos de secciones. contiene los datos de secciones, largos, nodos. Se entiende por sección una línea, un transformador, etc.
- Base de datos de rutas. no usado en esta etapa pues se utiliza en el programa de optimización y en él se ingresa el tipo de suelo, costo de metro de zanjado, etc.

A modo de ejemplo se observa en la figura 4.6 los dátos técnicos utilizados en un transformador:


SWEDNET.EXE
Num
SwedNet TRANSFORMER DATA
TRANSFORMER DESIGNATION
RATED POWER (kVA)
TRANSF. CODE
NO LOAD LOSSES (%)
LOAD LOSSES (%)
COST (Th.USD)
6-10B1
10
1
0.80
4.30
0
SHORT CIRCUIT RESISTANCE ER (%)
SHORT CIRCUIT IMPEDANCE EK (%)
NOMINAL RATIO SEC.SIDE (kV)
NOMINAL RATIO PRIM.SIDE (kV)
TAP CHANGER STEPS (+)
TAP CHANGER STEPS (-)
TAP CHANGER INTERVAL (%)
0.7
4.0
0.23
6.30
2
2
2.50
VOLTAGE LEVEL
VOLTAGE (kV)
NOTE:
1
0.23
reciclaje
Inicio
Explorando - D:...
Microsoft Excel ...
Microsoft Word ...
DIS_COM.RSF ...
Barra de acces...
SWEDNET....
11:02
Figura 4.6

4.3.2. *Datos de salida*

Como datos de salida el Swednet brinda lo siguiente:

Nodos: tensiones, caídas de tensión, corriente, potencia de cortocircuito, corriente de cortocircuito.
Cuadripolos: corriente, potencia activa, potencia reactiva, perdidas, costo de pérdidas, porcentaje de utilización del cable o línea.

También brinda datos de utilización y pérdidas en los transformadores y calidad de tensión.
En la Figura 4.7 se presenta un ejemplo de cómo se presentan los datos a la salida:

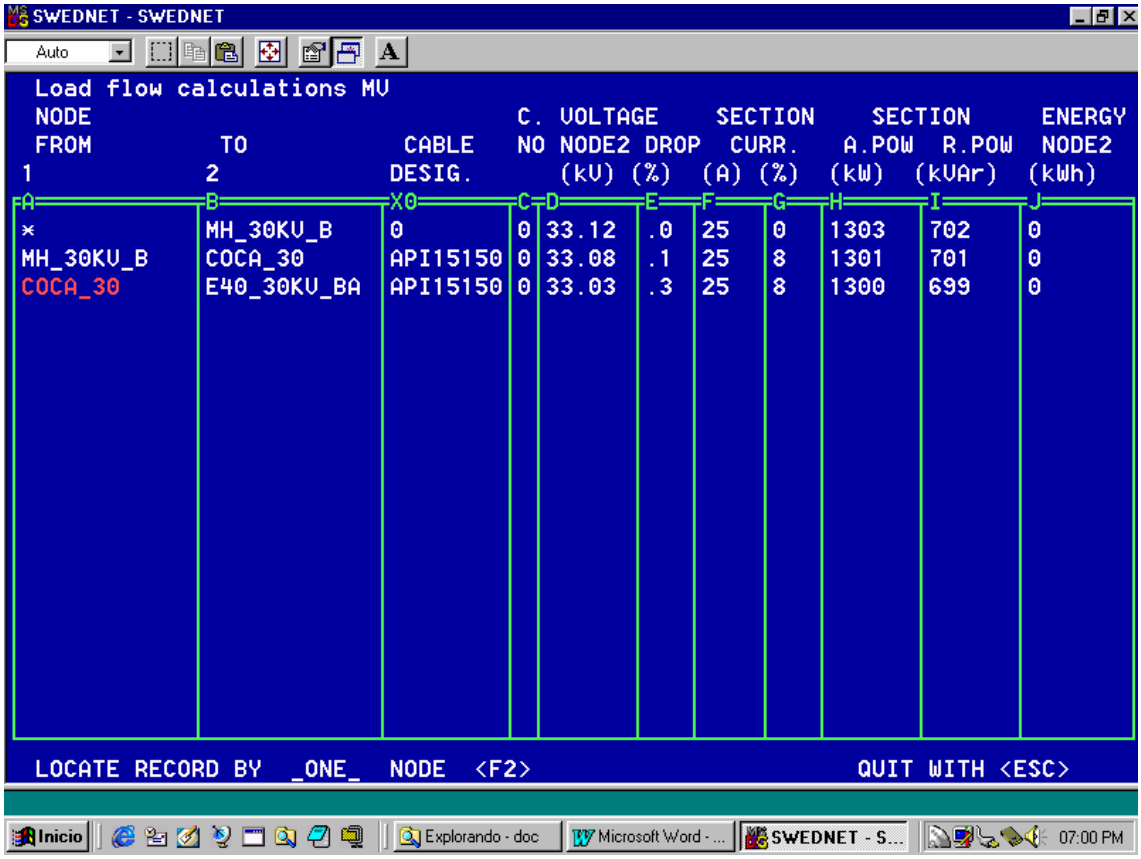


Figura 4.7

4.3.3. *Optimización*

El programa de optimización está realizado para planear redes radiales y anilladas, o realizar cambios en redes existentes. Dadas las demandas de potencia, y las limi-

taciónes técnicas existentes este programa determina la alternativa de menor costo para un proyecto de una red.

4.4. **Pruebas realizadas al FLUCAR 2.0**

A los efectos de verificar el correcto funcionamiento del programa, se ejecutaron distintas corridas en los tres programas comparando los resultados obtenidos.

4.4.1. Corrida 1

Se corrió un primer caso base, formado por un generador, dos líneas y dos barras de carga.

4.4.1.1. Corrida 1 con FLUCAR

```
      A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P        Q        V   delta   Vmin  Vmax  Qmin  Qmax }

              J30.     1    0        0       31.5   0         N     N
              321A     2    -5       -3      31.5   0         N     N
              021A     2    -7       -4      31.5   0         N     N


+CUADRIPOLOSPI
 {                              Z12
                    b1 -------////////------- b2
                       -              -
                       /              /
                  Y13  /              /  Y23
                       /              /
                       -              -
                       N              N
 }


     {          b1      b2      b3      Y13              Z12          Y23          Imax}
     cua001   J30.   321A    N        0           0.1+j0.2        0            0
```

```
cua002    J30.   021A    N        0            0.1+j0.2        0            0
```

Figura 4.8

Dando como resultado lo siguiente:

```
P R O Y E C T O   F L U C A R

                I I E  -  1 9 9 7



DATOS: corrida1.dat

RESULTADOS: corrida1.res


 RESULTADOS:

b: J30.  S:   12.0100048 + j   7.0200096  V:  31.5000000(  0.0000000ø)
b: 321A  S:   -5.0000000 + j  -3.0000000  V:  31.4650327( -0.0404652ø)
b: 021A  S:   -7.0000000 + j  -4.0000000  V:  31.4522927( -0.0578309ø)
 TSGeneracion:  12.0100048 + j   7.0200096
 TSConsumo:     12.0000000 + j   7.0000000

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001   5.0034342 + j   3.0068683
321A->cua001  -5.0000000 + j  -3.0000000

J30.->cua002   7.0065706 + j   4.0131413
021A->cua002  -7.0000000 + j  -4.0000000

Perdidas Joule en los cuadripolos:  0.0100048
```

Figura 4.9

4.4.1.2. *Corrida 1 con EPRI*

```
Caso Base: corrida 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
 a
```

```
 a
BAS
C Definicion de areas:
C
3456789012345678901234567890123456789012345678901234567890123456789
01234567890
C  NNNNNNNNNNnnnnnnnnvvvv SSSSSSSS Z1 Z2 Z3 Z4 Z5 Z6 Z7 Z8 Z9 Z0
VVVVvvvv
C  MJ       E2      31.5        0 MJ                          31.531.5
C
3456789012345678901234567890123456789012345678901234567890123456789
01234567890
C
booonnnnnnnnvvvvzzpppppqqqqqllllccccPPPPpppppQQQQQqqqqqVVVVvvvvNN
NNNNNNtttt%%%
B     MJ      31.5                 15              1000
B     E2      31.5                   -7   -4
B     E32     31.5                    -5   -3
C
**********************************************************************
**********
C
3456789012345678901234567890123456789012345678901234567890123456789
01234567890
C
looonnnnnnnnvvvvvmNNNNNNNNNVVVVisIIIInrrrrrrrxxxxxxxgggggggbbbbbbbbmm
mmCCCCCCCCmaamaa
L     MJ      31.5 E2      31.5                    0.1  0.2
```

Figura 4.10

Dando como resultado:

SUMMARY OF GENERATOR DATA

BUS NAME		MW	ACTUAL MW	DESIRED MVAR	MVAR LIMITS	VOLTAGE VOLTAGE
			PMAX	LOW	HIGH	NOT HELD
E2	31.5	-7.00	0.00	-4.00	(P+Q NO LIMITS)	
MJ	31.5	12.01	15.00	7.02	(P+Q NO LIMITS)	
E32	31.5	-5.00	0.00	-3.00	(P+Q NO LIMITS)	

SUMMARY OF 0 BUSES WITH VOLTAGE OVER 1.05 PER UNIT

SUMMARY OF 0 BUSES WITH VOLTAGE UNDER 0.95 PER UNIT

THERE WERE NO OVERLOADED TRANSMISSION LINES

THERE WERE NO OVERLOADED TRANSFORMERS

THERE WERE NO REGULATING TRANSFORMERS

1Caso Base: prueba 1 asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
SUMMARY REPORT PAGE NO. 2
a OWNER LOSS SUMMARY
a 1/ 1/1970 -6: 0: 0

SUMMARY OF LOSSES BY OWNER IN MW

OWNER LOSSES OWNER LOSSES OWNER LOSSES OWNER LOSSES

0.01

********** TOTAL SYSTEM LOSSES ARE 0.01 MW **********

----------BUS DATA----------X----------LINE DATA----------
--X

NAME-BASE	P.U.	GENERATION	LOAD	SHUNT	ID	TO	LINE FLOWS	LINE LOSSES

ACTUAL KV	ANGLE	MW	MVAR	MW	MVAR	MW	MVAR	BUS NAME	MW	MVAR	MW	MVAR
E2	31.5	0.9985	-7.0	-4.0	0.0	0.0	0.0	0.0 ----------				
	31.5 KV	-0.1						MJ	31.5	-7.0	-4.0	0.01	0.01
MJ	31.5	1.0000	12.0	7.0	0.0	0.0	0.0	0.0 ----------				
	31.5 KV	0.0						E2	31.5	7.0	4.0	0.01	0.01
								E32	31.5	5.0	3.0	0.00	0.01
E32	31.5	0.9989	-5.0	-3.0	0.0	0.0	0.0	0.0 ----------				
	31.5 KV	0.0						MJ	31.5	-5.0	-3.0	0.00	0.01

Figura 4.11

4.4.1.3. Comparación de los resultados obtenidos entre FLUCAR20 y EPRI

BARRA	P_EPRI (MW)	P_FLUC. (MW)	ΔP(%)	Q_EPRI (MVAR)	Q_FLUC. (MVAR)	ΔQ(%)	V_EPRI (kV)	V_FLUC (kV)	ΔV(%)	θ_EPRI (º)	θ_FLUC. (º)	Δθ(%)
J.30	12.01	12.01	**0.00**	7.02	7.02	**0.00**	31.5	31.5	**0.00**	0.00	0.00	**0.00**
021ª	-7.0	-7.0	**0.00**	-4.0	-4.0	**-0.00**	31.45	31.45	**0.00**	-0.1	-0.057	**43.0**
321ª	-5.0	-5.0	**0.00**	-3.0	-3.0	**0.00**	31.47	31.47	**-0.00**	0.0	-0.04	**_____**

Se observa la coincidencia de los resultados. La diferencia mayor es debido al redondeo que realiza el EPRI trayendo como resultado un error apreciable en las fase de las tensiones.

4.4.1.4. *Corrida 1 con SWEDNET*

Dado que los datos de entrada se deben ingresar en distintas bases de datos, no se presentan los mismos en la documentación por creer que no aporta información relevante.

El resultado es el siguiente:

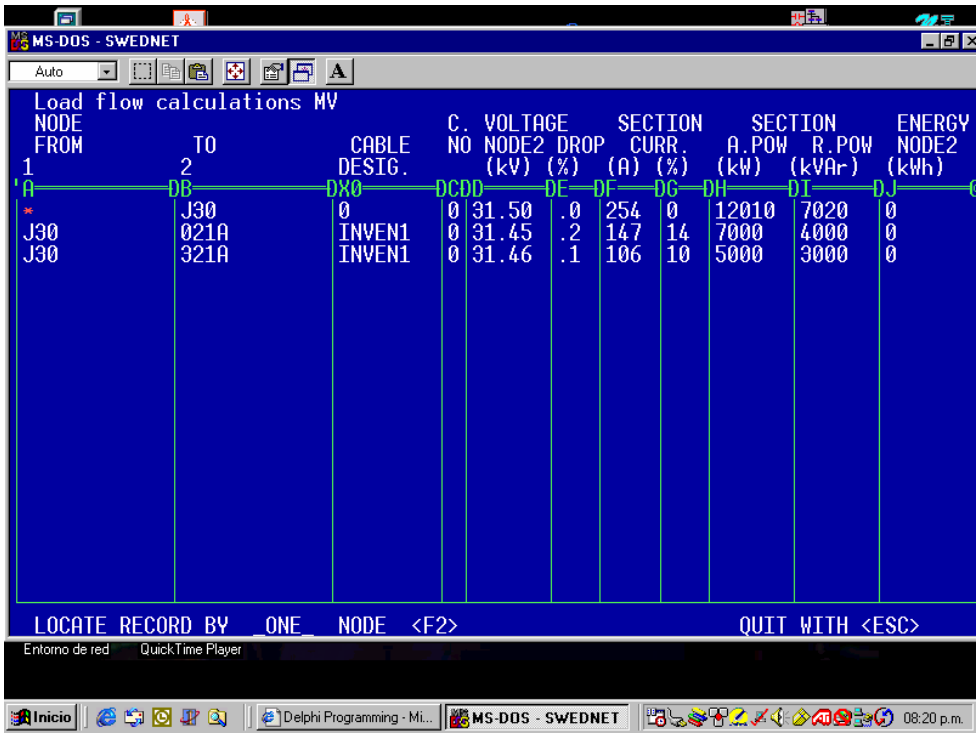


Figura 4.12

4.4.1.5. *Comparación de los resultados obtenidos entre FLUCAR20 y SWEDNET*

BARRA	P_SWED (MW)	P_FLUC. (MW)	ΔP(%)	Q_SWE (MVAR)	Q_FLUC. (MVAR)	ΔQ(%)	V_SWED (kV)	V_FLUC (kV)	ΔV(%)	θ_SWED (º)	θ_FLUC. (º)	Δθ(%)
J.30	12.01	12.01	**0.00**	7.02	7.02	**0.00**	31.5	31.5	**0.00**	S/D	0.00	--------
021ª	-7.0	-7.0	**0.00**	-4.0	-4.0	**-0.00**	31.45	31.45	**0.00**	S/D	-0.057	-------
321ª	-5.0	-5.0	**0.00**	-3.0	-3.0	**0.00**	31.47	31.47	**-0.00**	S/D	-0.04	_____

Se observa que no hay diferencias entre los resultados arrojados por ambos programas. Cabe destacar que Swednet no presenta en los resultados el valor del desfasaje de la tensión.

4.4.2. Corrida 2

Se corrió un segundo caso, consistente en un generador, dos líneas, barras de carga y un transformador.

4.4.2.1. Corrida 2 con FLUCAR 2.0

```
{       CORRIDA2 DEL FLUCAR
        RED FICTICIA


     A R C H I V O  D E  E N T R A D A  D E  D A T O S
}

+BARRAS {Nom   Tipo  P       Q       V   delta   Vmin  Vmax  Qmin  Qmax }

            J30.     1    0       0        31.5  0       N     N
            321A     2    -5      -3       31.5  0       N     N
            021A     2    0       0        31.5  0       N     N
            022A     2    -7      -4        6.3  0       N     N

+CUADRIPOLOSPI
 {                          Z12
                          b1 -------////////------- b2
                             -            -
                             /            /
                        Y13  /            /  Y23
                             /            /
                             -            -
                             N            N
 }


         { b1    b2    b3     Y13          Z12          Y23         Imax}
 cua001  J30.  321A   N       0          0.6+j0.4       0           0
 cua002  J30.  021A   N       0          0.6+j0.3       0           0

+TRAFOS
{
                                    1:n
                                  - -  - -       Zcc
                          Nodo1  ----- -     -     - ----/////---- Nodo2
                                  - -  - -
}

{ Nombre     b1    b2        n    Zcc        Imax }
    traf1    021A  022A    0.2   0+j0.001    0
```

Figura 4.13

```
DATOS: corrida4.dat

RESULTADOS: corrida4.res


 RESULTADOS:

  b: J30.  S:   12.0604801 + j   7.0353517  V:  31.5000000(  0.0000000ø)
b: 321A  S:   -5.0000000 + j  -3.0000000  V:  31.3660968( -0.0115980ø)
 b: 021A  S:    0.0000000 + j   0.0000000  V:  31.3276108(  0.0174760ø)
b: 022A  S:   -7.0000000 + j  -4.0000000  V:   6.2648836(  0.0072584ø)
 TSGeneracion:  12.0604801 + j   7.0353517
 TSConsumo:     12.0000000 + j   7.0000000

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001   5.0207349 + j   3.0138233
321A->cua001  -4.9999996 + j  -2.9999998

J30.->cua002   7.0397453 + j   4.0215284
021A->cua002  -6.9999988 + j  -4.0016552

Perdidas Joule en los cuadripolos:  0.0604817

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
021A->traf1   6.9999952 + j   4.0016553
022A->traf1  -6.9999952 + j  -3.9999992

Perdidas Joule en los transformadores:  0.0000000

 REGULADORES
Perdidas Joule en los reguladores:  0.0000000

Perdidas Joule totales en las l¡neas:  0.0604817
```

Figura 4.14

4.4.2.2. Corrida 2 con EPRI

```
        C
3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
        C
booonnnnnnnnvvvvvzzpppppqqqqqqllllccccPPPPpppppQQQQQqqqqqVVVVvvvvNNNNNNNNtttt%%%
        B   MJ       31.5            15          1000
        B   E2       31.5
        B   E32      31.5                          -5    -3
        B   E2_6     6.3                           -7    -4

        C
********C*************************************************************************
3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
        C
looonnnnnnnnvvvvvmNNNNNNNNNVVVVisIIIInrrrrrrrxxxxxxxgggggggbbbbbbbbmmmmmCCCCCCCCmaa
maa
        L   MJ       31.5   E2           31.5        0.6  0.3
        L   MJ       31.5   E32          31.5        0.6  0.4
        T   E2       31.5   E2_6         6.3               0.001
        ZZ
```

Figura 4.15

```
        SUMMARY OF GENERATOR DATA

                                                    ACTUAL    DESIRED
          BUS NAME        MW       MW      MVAR      MVAR  LIMITS    VOLTAGE   VOLT-
AGE                      PMAX              LOW      HIGH           NOT HELD
        MJ     31.5      12.06    15.00     7.03    (P+Q NO LIMITS)
        E32    31.5      -5.00     0.00    -3.00    (P+Q NO LIMITS)
        E2_6   6.3       -7.00     0.00    -4.00    (P+Q NO LIMITS)

                          SUMMARY OF    0 BUSES WITH VOLTAGE OVER  1.05 PER UNIT

                          SUMMARY OF    0 BUSES WITH VOLTAGE UNDER 0.95 PER UNIT

                                  THERE WERE NO OVERLOADED  TRANSMISSION LINES

                                  THERE WERE NO OVERLOADED TRANSFORMERS

                                  THERE WERE NO REGULATING TRANSFORMERS
        1Caso Base: prueba 1  asdfasfd asdfadf dfaci¢n montevideo j rscxh00.flu
SUMMARY REPORT PAGE NO.   2
        a                                                          OWNER LOSS SUMMARY
        a                                                          1/ 1/1970 -6: 0: 0

                        SUMMARY OF LOSSES BY OWNER IN MW

            OWNER  LOSSES     OWNER  LOSSES     OWNER  LOSSES     OWNER  LOSSES
```

```
                    0.06
              ********** TOTAL SYSTEM LOSSES ARE    0.06 MW **********
         1Caso Base: prueba 1  asdfasfd asdfadf dfaci¿n montevideo j rscxh00.flu                POWER
FLOW REPORT    PAGE NO.   1
         a                                                          ZONE (  ) OF AREA (        )
         a                                                                1/ 1/1970 -6: 0: 0
```

```
         0X------------------------------BUS DATA------------------------------X----------------------LINE
DATA----------------------X
              NAME-BASE  P.U.      GENERATION         LOAD            SHUNT         ID    TO      LINE FLOWS
LINELOSSES
         ACTUAL KV   ANGLE     MW     MVAR     MW     MVAR     MW     MVAR      BUS NAME      MW     MVAR
MW  MVAR  OVL
         E2          31.5     0.9946    0.0      0.0      0.0      0.0      0.0      0.0 ----------------------------------------------------
-
                     31.3 KV    0.0                                                          MJ     31.5    -7.0    -4.0    0.04    0.02
                                                          PHASE SHIFT        0.00 DEG.    E2_6   6.3     7.0     4.0     0.00    0.00
         MJ          31.5     1.0000    12.1     7.0      0.0      0.0      0.0      0.0 ----------------------------------------------------

                     31.5 KV    0.0                                                          E2     31.5    7.0     4.0     0.04    0.02
                                                                                             E32    31.5    5.0     3.0     0.02    0.01
         E32         31.5     0.9958   -5.0     -3.0     0.0      0.0      0.0      0.0 ----------------------------------------------
------
                     31.4 KV    0.0                                                          MJ     31.5    -5.0    -3.0    0.02    0.01
         E2_6        6.3      0.9945   -7.0     -4.0     0.0      0.0      0.0      0.0 ----------------------------------------------
------
                     6.3 KV     0.0                                                          E2     31.5    -7.0    -4.0    0.00    0.00
```

Figura 4.16

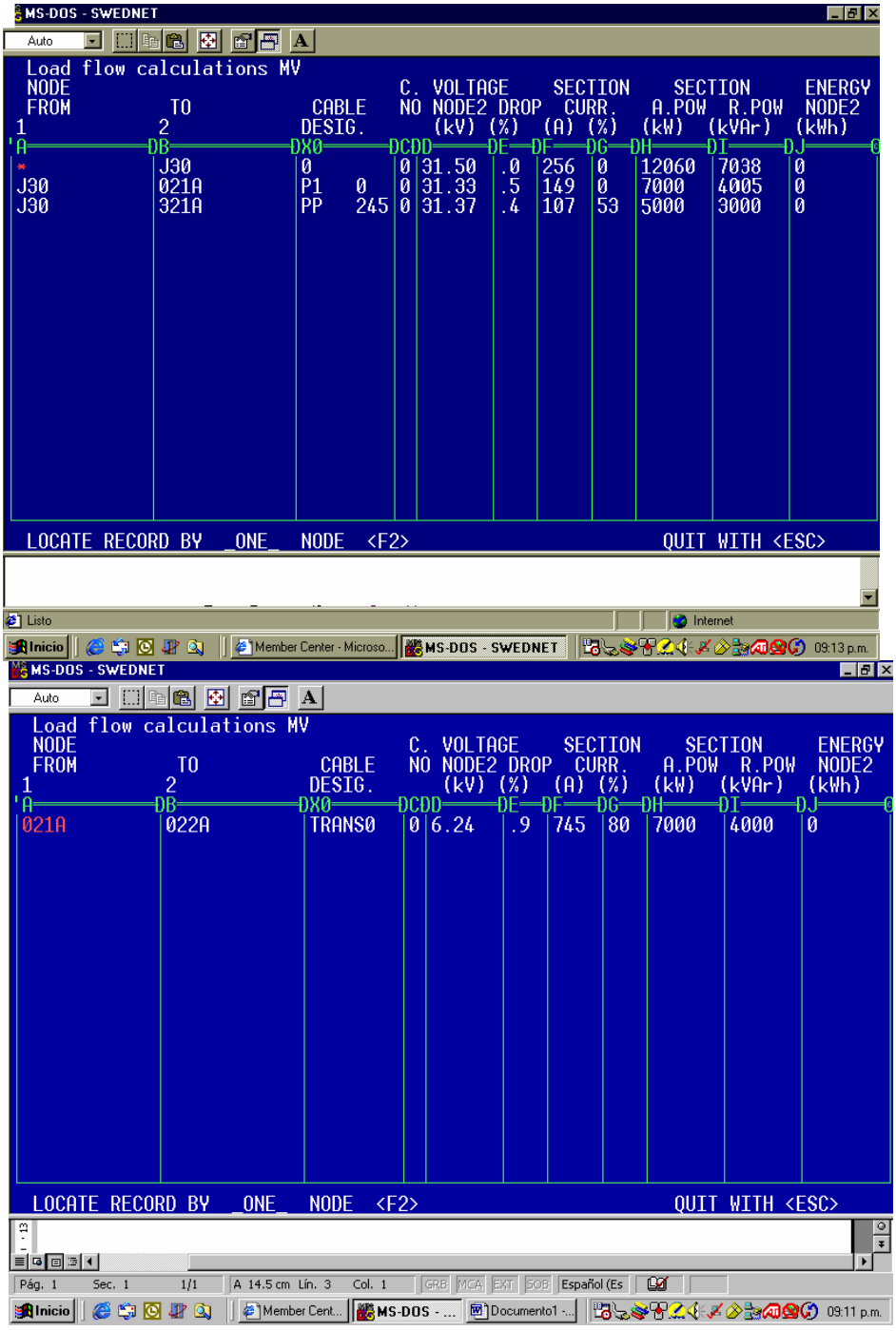
4.4.2.3. Comparación de los resultados obtenidos entre FLUCAR20 y EPRI

BARRA	P_EPRI (MW)	P_FLUC. (MW)	ΔP(%)	Q_EPRI (MVAR)	Q_FLUC. (MVAR)	ΔQ(%)	V_EPRI (kV)	V_FLUC (kV)	ΔV(%)	θ_EPRI (º)	θ_FLUC. (º)	Δθ(%)
J.30	12.06	12.06	**0.00**	7.03	7.04	**-0.14**	31.5	31.5	**0.00**	0.0	0.00	**0.00**
021ª	0.0	0.0	**0.00**	-4.0	-4.0	**-0.00**	31.33	31.33	**0.00**	-0.0	-0.011	**-----**
321ª	-5.0	-5.0	**0.00**	-3.0	-3.0	**0.00**	31.37	31.37	**-0.00**	0.0	-0.04	**------**
022ª	-7.0	-7.0	**0.00**	-4.0	-4.0	**0.00**	6.27	6.27	**-0.01**	-0.0	0.007	**----**

Los valores son iguales. La pequeña diferencia existente es por el redondeo tal como se observara anteriormente y en fases.

4.4.2.4. *Corrida 2 en SWEDNET*

Se obtuvieron los siguientes resultados (Figura 4.17):



4.4.2.5. Comparación de los resultados obtenidos entre FLUCAR20 y SWEDNET

BARRA	P_SWED (MW)	P_FLUC. (MW)	ΔP(%)	Q_SWE (MVAR)	Q_FLUC. (MVAR)	ΔQ(%)	V_SWED (kV)	V_FLUC (kV)	ΔV(%)	θ_SWED (º)	θ_FLUC. (º)	Δθ(%)
J.30	12.06	12.06	**0.00**	7.038	7.04	**0.03**	31.5	31.5	**0.00**	0.00	0.00	**0.00**
021ª	0.0	0.0	**0.00**	-4.0	0.0	**-0.00**	31.33	31.33	**0.00**	--------	-0.011	**-----**
321ª	-5.0	-5.0	**0.00**	-3.0	-3.0	**0.00**	31.37	31.37	**-0.00**	-------	-0.04	**------**
022ª	-7.0	-7.0	**0.00**	-4.0	-4.0	**0.00**	6.24	6.27	**-0.01**	--------	0.007	**----**

4.4.3. Conclusiones

Se observa la coincidencia de los resultados, dentro de cierto margen de error, entre los tres programas.

Comparativamente el funcionamiento de los tres flujos de carga es similar. Lo que se destaca como mayor diferencia, y en lo que se puede hacer mayor hincapié es en la presentación de datos. En FLUCAR y el EPRI, la entrada de datos tiene diferencias menores ya que el EPRI es un programa relativamente viejo. En la salida de datos tampoco se ven diferencias mayores. El mayor cambio observado es en el SWEDNET. Este como mencionamos introduce el entorno Windows, que consideramos como un elemento a mejorar en el FLUCAR. La otra gran diferencia es la base de datos técnica que maneja el SWEDNET. Esto es útil a nivel empresarial, ya que la incorporación de todos los datos en bases de datos tipo .dbf da ciertos beneficios para la reutilizacion de losmismos datos en otras corridas, o su utilización para otros usos en la empresa. Sin embargo analizado esto, no nos parece ventajoso incluirlo en el FLUCAR, ya que el programa puede ser usado para aplicaciones totalmente diferentes a nivel universitario.

CAPITULO 5

5. MÉTODO DE NEWTON RAPHSON MODIFICADO

5.1. Introducción

La tercera versión de FLUCAR, incorpora nuevas herramientas al software existente. En las versiones anteriores, los reguladores de tensión de los transformadores se manipulaban "manualmente" entre corridas. Esta versión incorpora un método modificado para integrar la regulación automática.

La primera versión FLUCAR (v1.0) resolvió el problema de Flujo de Carga de una red de potencia aplicando el método de Newton-Raphson. No estában previstos los transformadores ni las cotas para las tensiones o para las potencias reactivas en las barras. Estas variantes fueron incorporadas en la versión 2.0.

En la publicación 70 TP 160-PWR de la IEEE, se plantea un sistema para la modificación del método de Newton Raphson clásico que se aplica en particular para el ajuste automático del regulador de tensión de los transformadores (taps).

A continuación se realiza un desarrollo del método planteado.

5.2. Estudio del artículo "Automatic Adjustment of Transformer and Phase-Shifter Taps in the Newton Power Flow"

Para un sistema de potencia, la potencia entrante en el nodo k es (balance en el nodo k entre lo aportado por un generador y lo absorbido por una carga) es:

$$P_k + jQ_k = \overline{V}_k \hat{I}_k = \overline{V}_k \sum_{i=1}^{n} \hat{V}_i \hat{Y}_{ki} \qquad \text{k} = 1,2,....,\text{n}$$

Desarrollandolo en forma matricial, la matriz jacobiana quedaría:

$$\begin{bmatrix} P_{1s} \\ \vdots \\ P_{n-1,s} \\ Q_{1s} \\ \vdots \\ Q_{\ell s} \end{bmatrix} - \begin{bmatrix} P_1\left(\left[X_{2n}^0\right]\right) \\ \vdots \\ P_{n-1}\left(\left[X_{2n}^0\right]\right) \\ Q_1\left(\left[X_{2n}^0\right]\right) \\ \vdots \\ Q_\ell\left(\left[X_{2n}^0\right]\right) \end{bmatrix} = \begin{bmatrix} \frac{\partial P_1}{\partial \theta_1} & \cdots & \frac{\partial P_1}{\partial \theta_{n-1}} & \frac{\partial P_1}{\partial V_1} V_1 & \cdots & \frac{\partial P_1}{\partial V_\ell} V_\ell \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial P_{n-1}}{\partial \theta_1} & \cdots & \frac{\partial P_{n-1}}{\partial \theta_{n-1}} & \frac{\partial P_{n-1}}{\partial V_1} V_1 & \cdots & \frac{\partial P_{n-1}}{\partial V_\ell} V_\ell \\ \frac{\partial Q_1}{\partial \theta_1} & \cdots & \frac{\partial Q_1}{\partial \theta_{n-1}} & \frac{\partial Q_1}{\partial V_1} V_1 & \cdots & \frac{\partial Q_1}{\partial V_\ell} V_\ell \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Q_\ell}{\partial \theta_1} & \cdots & \frac{\partial Q_\ell}{\partial \theta_{n-1}} & \frac{\partial Q_\ell}{\partial V_1} V_1 & \cdots & \frac{\partial Q_\ell}{\partial V_\ell} V_\ell \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_{n-1} \\ \frac{\Delta V_1}{V_1} \\ \vdots \\ \frac{\Delta V_\ell}{V_\ell} \end{bmatrix}$$

o dicho de otra manera:

$$\begin{bmatrix} \left[\Delta P\left(\left[X_{2n}^0\right]\right)\right] \\ \left[\Delta Q\left(\left[X_{2n}^0\right]\right)\right] \end{bmatrix} = \begin{bmatrix} \mathrm{H}\left(\left[X_{2n}^0\right]\right) & \mathrm{N}\left(\left[X_{2n}^0\right]\right) \\ J\left(\left[X_{2n}^0\right]\right) & L\left(\left[X_{2n}^0\right]\right) \end{bmatrix} \begin{bmatrix} [\Delta\theta] \\ \left[\frac{\Delta V}{V}\right] \end{bmatrix}$$

Siendo

$$H_{km} = \frac{\partial P_k}{\partial \theta_m}$$

$$N_{km} = \frac{\partial P_k}{\partial V_m} V_m$$

$$J_{km} = \frac{\partial Q_k}{\partial \theta_m}$$

$$L_{km} = \frac{\partial Q_k}{\partial V_m} V_m$$

En los métodos como implementados en Flucar 2.0, para redes con transformadores con regulación de tensión, luego de cada solución para ΔV y $\Delta \theta$, la posición del tap se modifica ya sea manualmente o por un procedimiento matemático. A través de este procedimiento, la nueva posición del tap es la vieja posición más un factor de aceleración que multiplica la tensión deseada menos la tensión obtenida.

Si t_i es la posición del regulador automático de tensión o tap, entonces:

$$t_i^{new} = t_i^{old} + c_i.(V_i^{reg} - V_i)$$

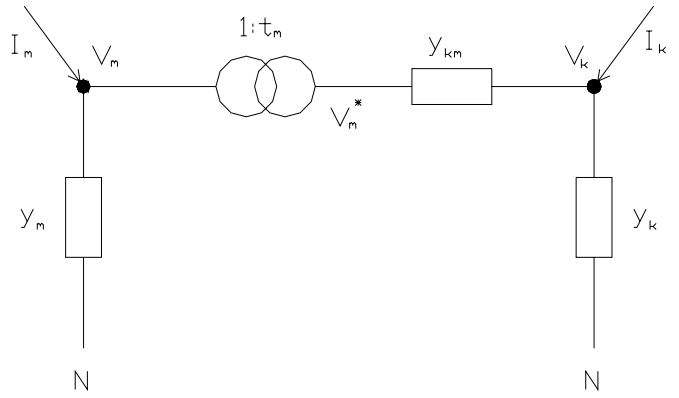
Los factores de aceleración (c_i) son de cálculo empírico y muchas veces causan una lenta resolución o una convergencia impredecible.

La idea de esta publicación es incorporar la posición del tap como variable más que como constantes en las ecuaciones. Así un tranformador Ti se opera para que su tap t_i se ajuste dentro de ciertos limites para lleva V_i al valor V_i de régimen. Para ello, V_i es constante y t_i sustituye a V_i en las ecuaciones como variable. Resumiendo, la idea es que en aquel nodo donde exista un tap se sustituya V_i por t_i.

Por lo tanto N_{km} se convierte en C_{km} y L_{km} en D_{km}, siendo

$$C_{km} = \frac{\partial P_k}{\partial t_m} t_m \quad \text{y} \quad D_{km} = \frac{\partial Q_k}{\partial t_m} t_m$$

En la publicación 70 TP 160-PWR de la IEEE, en la cual se plantea el cambio de variable, se detallan los cambios a realizar en la matriz, y se menciona que el cambio de variable produce solo cambios menores en los elementos dela matriz. Por lo tanto las modificaciónes de programacion serían menores. Consideremos un nodo genérico k y veamos como se modifica el problema:



En el nodo k-ésimo:

$$\bar{I}_k = \bar{V}_k \bar{y}_k + \sum_{i=1, i\neq k, i\neq m}^{n} (\bar{V}_k - \bar{V}_i)\bar{y}_{ki} + (\bar{V}_k - \bar{V}_m{}^*)\bar{y}_{km}$$

$$\bar{I}_k = \bar{V}_k \bar{y}_k + \bar{V}_k \sum_{i=1, i\neq k, i\neq m}^{n} \bar{y}_{ki} - \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{y}_{ki} + \bar{V}_k \bar{y}_{km} - t_m \bar{V}_m \bar{y}_{km}$$

$$\bar{I}_k = \bar{V}_k (\bar{y}_k + \sum_{i=1, i\neq k}^{n} \bar{y}_{ki}) - \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{y}_{ki} - t_m \bar{V}_m \bar{y}_{km}$$

se define :

$$\bar{y}_k + \sum_{i=1, i\neq k}^{n} \bar{y}_{ki} = \bar{Y}_{kk}$$

$$-\bar{y}_{ki} = \bar{Y}_{ki}$$

y entonces :

$$\bar{I}_k = \bar{V}_k \bar{Y}_{kk} + \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{Y}_{ki} + t_m \bar{V}_m \bar{Y}_{km}$$

Por lo que

$$\boxed{\bar{I}_k = \sum_{i=1, i\neq m}^{n} \bar{V}_i \bar{Y}_{ki} + t_m \bar{V}_m \bar{Y}_{km} \qquad k = 1,2,....,n}$$ ecuación 5.1

Si consideramos las ecuaciones de comportamiento de la red

$$\bar{V}_i = V_i e^{j\theta_i} = V_i(\cos\theta_i + j\,\mathrm{sen}\,\theta_i)$$

$$\bar{Y}_{ki} = a_{ki} + jb_{ki}$$

Si substituimos en la ecuación 5.1, vemos que:

$$P_k + jQ_k = V_k V_m t_m (a_{km} - jb_{km}) e^{j(\theta_k - \theta_m)} + V_k \sum_{i=1, i\neq m}^{n} V_i (a_{ki} - jb_{ki}) e^{j(\theta_k - \theta_i)}$$

$$P_k + jQ_k = V_k V_m t_m (a_{km} - jb_{km}) [\cos(\theta_k - \theta_m) + j\,\mathrm{sen}(\theta_k - \theta_m)] + V_k \sum_{i=1, i\neq m}^{n} V_i (a_{ki} - jb_{ki}) e^{j(\theta_k - \theta_i)}$$

y como $C_{km} = \dfrac{\partial P_k}{\partial t_m} t_m$, sustituyendo:

$$C_{km} = \frac{\partial P_k}{\partial t_m} t_m = V_k V_m [\mathrm{a_{km}} \cos(\theta_k - \theta_m) + b_{km} \mathrm{sen}(\theta_k - \theta_m)] \mathrm{t_m}$$

$$D_{km} = \frac{\partial Q_k}{\partial t_m} t_m = V_k V_m [\mathrm{a_{km}} \mathrm{sen}(\theta_k - \theta_m) - b_{km} \cos(\theta_k - \theta_m)] \mathrm{t_m}$$

Si consideramos ahora el nodo m:

$$\bar{\bar{I}}_m = \bar{V}_m \bar{y}_m + \sum_{i=1, i\neq k, i\neq m}^{n} (\bar{V}_m - \bar{V}_i) \bar{y}_{mi} + (\bar{V}_m - \frac{\bar{V}_k}{t_m}) t_m{}^2 \bar{y}_{mk}$$

$$\bar{\bar{I}}_m = \bar{V}_m \bar{y}_m + \bar{V}_m \sum_{i=1, i\neq k, i\neq m}^{n} \bar{y}_{mi} - \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{y}_{mi} + \bar{V}_m \bar{y}_{mk} t_m{}^2 - t_m \bar{V}_k \bar{y}_{mk}$$

$$\bar{\bar{I}}_m = \bar{V}_m (\bar{y}_m + \sum_{i=1, i\neq k, i\neq m}^{n} \bar{y}_{mi} + t_m{}^2 \bar{y}_{mk}) - \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{y}_{mi} - t_m \bar{V}_k \bar{y}_{mk}$$

se define :

$$\bar{y}_m + \sum_{i=1, i\neq k, i\neq m}^{n} \bar{y}_{mi} = \bar{Y}'_{mm}$$

$$-\bar{y}_{mi} = \bar{Y}_{mi}$$

Por lo que:

$$\boxed{\bar{\bar{I}}_m = \sum_{i=1, i\neq k, i\neq m}^{n} \bar{V}_i \bar{Y}_{mi} + \mathrm{t_m} \bar{Y}_{mk} \bar{V}_k + \bar{V}_m \bar{Y}'_{mm} - \bar{V}_m \mathrm{t_m}{}^2 \bar{Y}_{mk}}$$ ecuación 5.2

Si consideramos las ecuaciones de comportamiento de la red

$$\overline{V}_i = V_i e^{j\theta_i} = V_i\left(\cos\theta_i + j\,\mathrm{sen}\theta_i\right)$$
$$\overline{Y}_{ki} = a_{ki} + jb_{ki}$$

Si substituimos en la ecuación 5.2, vemos que:

$$P_m + jQ_m = \overline{V}_m \sum_{i=1, i\neq k, i\neq m}^{n} \overline{V}_i \overline{Y}_{mi} + t_m \overline{V}_m \overline{Y}_{mk} \overline{V}_k + \overline{V}_m{}^2 \overline{Y}'_{mm} - \overline{V}_m{}^2 t_m{}^2 \overline{Y}_{mk}$$

$$P_m + jQ_m = V_k V_m t_m \left(a_{mk} - jb_{mk}\right) e^{j(\theta_m - \theta_k)} + V_m{}^2 \left[a'_{mm} - jb'_{mm} - t_m{}^2 \left(a_{mk} - jb_{mk}\right)\right] + V_m \sum_{i=1, i\neq m, i\neq k}^{n} V_i \left(a_{mi} - jb_{mi}\right) e^{j(\theta_m - \theta_i)}$$

y como $C_{mm} = \dfrac{\partial P_m}{\partial t_m} t_m$, sustituyendo:

$$C_{mm} = \frac{\partial P_m}{\partial t_m} t_m = V_k V_m \left[a_{mk} \cos\left(\theta_m - \theta_k\right) + b_{mk}\,\mathrm{sen}\left(\theta_m - \theta_k\right)\right] t_m - 2 t_m{}^2 a_{mk} V_m{}^2$$

$$D_{mm} = \frac{\partial Q_m}{\partial t_m} t_m = V_k V_m t_m \left[a_{mk}\,\mathrm{sen}\left(\theta_m - \theta_k\right) - b_{km} \cos\left(\theta_m - \theta_k\right)\right] + 2 t_m{}^2 V_m{}^2\, b_{mk}$$

Además, como sucede usualmente, los elementos de la matriz en las cual no existe rama entre los nodos k y m, serán cero. Pero los elementos de la matriz serán cero también, cuando la rama k,m no es la rama en la cual se encuentra el regulador.

Es decir:

$C_{mi} = 0$
$D_{mi} = 0 \qquad \forall \; i, i \neq k$

Por lo tanto, el sistema se formaría de la siguiente manera:

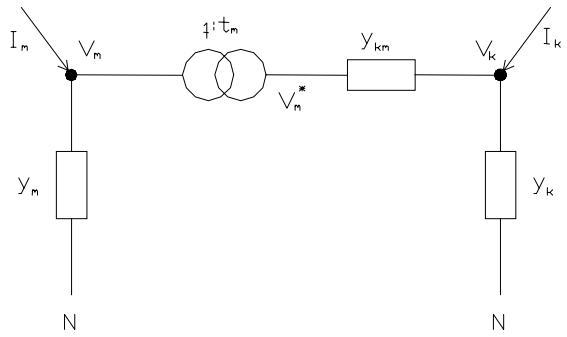
$$\begin{bmatrix} \Delta P_1 \\ \vdots \\ \Delta P_k \\ \vdots \\ \Delta P_m \\ \vdots \\ \Delta P_{n-1} \\ \Delta Q_1 \\ \vdots \\ \Delta Q_k \\ \vdots \\ \Delta Q_m \\ \vdots \\ \Delta Q_{n-1} \end{bmatrix} = \begin{bmatrix} H_{1,1} & \cdots & H_{1,n-1} & N_{1,1} & \cdots & 0 & \cdots & N_{1,n-1} \\ \vdots & & \vdots & \vdots & & 0 & & \vdots \\ H_{k,1} & \cdots & H_{k,n-1} & N_{k,1} & \cdots & C_{k,m} & \cdots & N_{k,n-1} \\ \vdots & & \vdots & \vdots & & 0 & & \vdots \\ H_{m,1} & & H_{m,n-1} & N_{m,1} & & C_{m,m} & & N_{m,n-1} \\ & & & & & 0 & & \\ H_{n-1,1} & & H_{n-1,n-1} & N_{n-1,1} & & 0 & & N_{n-1,n-1} \\ M_{11} & \cdots & M_{1,n-1} & T_{11} & \cdots & 0 & \cdots & T_{1,n-1} \\ \vdots & & \vdots & \vdots & & 0 & & \vdots \\ M_{k,1} & & M_{k,n-1} & T_{k,1} & & D_{k,m} & & T_{k,n-1} \\ \vdots & & \vdots & \vdots & & 0 & & \vdots \\ M_{m,1} & & M_{m,n-1} & T_{m,1} & & D_{m,m} & & T_{m,n-1} \\ \vdots & & \vdots & \vdots & & 0 & & \vdots \\ M_{n-1,1} & \cdots & M_{n-1,n-1} & T_{n-1,1} & \cdots & 0 & \cdots & T_{n-1,n-1} \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_k \\ \vdots \\ \Delta\theta_m \\ \vdots \\ \Delta\theta_{n-1} \\ \dfrac{\Delta V_1}{V_1} \\ \vdots \\ \dfrac{\Delta V_k}{V_k} \\ \vdots \\ \dfrac{\Delta t_m}{t_m} \\ \vdots \\ \dfrac{\Delta V_{n-1}}{V_{n-1}} \end{bmatrix}$$

El artículo señala que las derivadas en los otros casos dan igual que el método clásico, y sólo los elementos que cambian son mostrados. En el punto siguiente se demostrará que además de estás derivadas se modifican otros elementos de la matriz.

5.3. Discrepancias encontrados en el artículo estudiado

Cómo se mencionó anteriormente, el artículo desarrollado por Peterson-Meyer, al realizar la regulación automática modifica solo cuatro elementos de la matriz por cada regulador introducido. En este punto se deducirán todos los elementos de la matriz en los cuales influye el regulador para demostrar que esta aseveración no es correcta.

Siguiendo con razonamiento del punto anterior, si tenemos un regulador entre los nodos k y m, se tiene los siguiente:



La potencia entrante en el nodo k es (balance en el nodo k entre lo aportado por un generador y lo absorbido por una carga)

$$P_k + jQ_k = \overline{V}_k \hat{I}_k \qquad k = 1,2,....,n$$

Para todos los nodos excepto los nodos en los que se encuenta un regulador se cumple:

$$P_k + jQ_k = V_k \sum_{i=1}^{n} V_i \left(a_{ki} - jb_{ki}\right) e^{j(\theta_k - \theta_i)}$$

$$P_k + jQ_k = V_k \sum_{i=1}^{n} V_i \left(a_{ki} - jb_{ki}\right) \left[\cos\left(\theta_k - \theta_i\right) + j\,\mathrm{sen}(\theta_k - \theta_i)\right]$$

Separando en parte real y parte imaginaria se obtienen :

$$P_k = V_k \sum_{i=1}^{n} V_i \left(a_{ki}\cos\left(\theta_k - \theta_i\right) + b_{ki}\,\mathrm{sen}(\theta_k - \theta_i)\right)$$

$$Q_k = V_k \sum_{i=1}^{n} V_i \left(a_{ki}\,\mathrm{sen}\left(\theta_k - \theta_i\right) - b_{ki}\cos(\theta_k - \theta_i)\right)$$

Recordando que m es el nodo en el cual se encuentra el regulador se tiene:

$$P_m + jQ_m = V_k V_m t_m \left(a_{mk} - jb_{mk}\right) e^{j(\theta_m - \theta_k)} + V_m{}^2 \left[a'_{mm} - jb'_{mm} - t_m{}^2 \left(a_{mk} - jb_{mk}\right)\right] + V_m \sum_{i=1, i\neq m, i\neq k}^{n} V_i \left(a_{mi} - jb_{mi}\right) e^{j(\theta_m - \theta_i)}$$

Siendo $\bar{y}_m + \sum_{i=1, i\neq k, i\neq m}^{n} \bar{y}_{mi} = \bar{Y}'_{mm}$ y $\bar{Y}'_{mm} = a'_{mm} + jb'_{mm}$

$-\bar{y}_{mi} = \bar{Y}_{mi}$

Separando en parte real y parte imaginaria se obtienen :

$$P_m = V_k V_m t_m \left(\mathrm{a_{mk}}\cos(\theta_m - \theta_k) + b_{mk}\,\mathrm{sen}(\theta_m - \theta_k)\right) + V_m{}^2\left[a'_{mm} - t_m{}^2 a_{mk}\right] + V_m \sum_{i=1, i\neq m, i\neq k}^{n} V_i\left[\mathrm{a_{mi}}\cos(\theta_m - \theta_i) + b_{mi}\,\mathrm{sen}(\theta_m - \theta_i)\right]$$

$$Q_m = V_k V_m t_m\left[\mathrm{a_{mk}}\,\mathrm{sen}(\theta_m - \theta_k) - b_{km}\cos(\theta_m - \theta_k)\right] + V_m{}^2\left[-b'_{mm} + t_m{}^2 b_{mk}\right] + V_m \sum_{i=1, i\neq m, i\neq k}^{n} V_i\left[\mathrm{a_{mi}}\,\mathrm{sen}(\theta_m - \theta i_i) - b_{ki}\cos(\theta_m - \theta_i)\right]$$

Observando las ecuaciones anteriores se observa que las derivadas parciales coinciden con las realizadas en el artículo estudiado, exceptuando lo siguiente:

$$A_{mk} = \frac{\partial P_m}{\partial \theta_k} = V_m V_k t_m\left[a_{mk}\,\mathrm{sen}(\theta_m - \theta_k) - b_{mk}\cos(\theta_m - \theta_k)\right]$$

$$B_{\mathrm{mk}} = \frac{\partial Q_m}{\partial \theta_k} = -V_k V_m t_m\left[a_{mk}\cos(\theta_m - \theta_k) + b_{mk}\,\mathrm{sen}(\theta_m - \theta_k)\right]$$

$$C_{mk} = \frac{\partial P_m}{\partial V_k} V_k = V_k V_m t_m\left[a_{mk}\cos(\theta_m - \theta_k) + b_{mk}\,\mathrm{sen}(\theta_m - \theta_k)\right]$$

$$D_{mk} = \frac{\partial Q_m}{\partial V_k} V_k = V_k V_m t_m\ \left[a_{mk}\,\mathrm{sen}(\theta_m - \theta_k) - b_{mk}\cos(\theta_m - \theta_k)\right]$$

Recordando que k es el segundo nodo en el cual se encuentra el regulador se tiene:

$$P_k + jQ_k = V_k V_m t_m\left(a_{km} - jb_{km}\right)\left[\cos(\theta_k - \theta_m) + j\,\mathrm{sen}(\theta_k - \theta_m)\right] + V_k \sum_{i=1, i\neq m}^{n} V_i\left(a_{ki} - jb_{ki}\right)e^{j(\theta_k - \theta_i)}$$

Separando en parte real y parte imaginaria se obtienen :

$$P_k = V_k V_m t_m\left[\mathrm{a_{km}}\cos(\theta_k - \theta_m) + b_{km}\,\mathrm{sen}(\theta_k - \theta_m)\right] + V_k \sum_{i=1, i\neq m}^{n} V_i\left[\mathrm{a_{ki}}\cos(\theta_k - \theta_i) + b_{ki}\,\mathrm{sen}(\theta_k - \theta_i)\right]$$

$$Q_k = V_k V_m\left[\mathrm{a_{km}}\,\mathrm{sen}(\theta_k - \theta_m) - b_{km}\cos(\theta_k - \theta_m)\right]\mathrm{t_m} + V_k \sum_{i=1, i\neq m}^{n} V_i\left[\mathrm{a_{ki}}\,\mathrm{sen}(\theta_k - \theta_i) - b_{ki}\cos(\theta_k - \theta_i)\right]$$

Por lo tanto se obtienen las siguientes derivadas parciales diferentes al artículo estudiado:

$$A_{km} = \frac{\partial P_k}{\partial \theta_m} = V_k V_m t_m \left[a_{km} \text{sen}(\theta_k - \theta_m) - b_{km} \cos(\theta_k - \theta_m) \right]$$

$$B_{\text{km}} = \frac{\partial Q_k}{\partial \theta_m} = -V_k V_m t_m \left[a_{km} \cos(\theta_k - \theta_m) + b_{km} \text{sen}(\theta_k - \theta_m) \right]$$

$$C_{km} = \frac{\partial P_k}{\partial V_m} V_m = V_k V_m t_m \left[a_{km} \cos(\theta_k - \theta_m) + b_{km} \text{sen}(\theta_k - \theta_m) \right]$$

$$D_{km} = \frac{\partial Q_k}{\partial V_m} V_m = V_k V_m t_m \left[a_{km} \text{sen}(\theta_k - \theta_m) - b_{km} \cos(\theta_k - \theta_m) \right]$$

Esto ocasiona que el nuevo jacobiano quedará formado de la siguiente manera:

$$\begin{bmatrix} \Delta P_1 \\ \vdots \\ \Delta P_k \\ \vdots \\ \Delta P_m \\ \vdots \\ \Delta P_{n-1} \\ \Delta Q_1 \\ \vdots \\ \Delta Q_k \\ \vdots \\ \Delta Q_m \\ \vdots \\ \Delta Q_{n-1} \end{bmatrix} = \begin{bmatrix} H_{1,1} & \cdots & \cdots & \cdots & \cdots & H_{1,n-1} & N_{1,1} & \cdots & \cdots & \cdots & 0 & \cdots & N_{1,n-1} \\ \vdots & \ddots & & & & \vdots & \vdots & & & & 0 & & \vdots \\ H_{k,1} & \cdots & \cdots & \cdots & A_{k,m} & H_{k,n-1} & N_{k,1} & \cdots & \cdots & \cdots & C_{k,m} & \cdots & N_{k,n-1} \\ \vdots & & & & & \vdots & \vdots & & & & 0 & & \vdots \\ H_{m,1} & \cdots & A_{m,k} & & & H_{m,n-1} & N_{m,1} & & C_{m,k} & & C_{m,m} & & N_{m,n-1} \\ & & & \ddots & & & & & & & 0 & & \\ H_{n-1,1} & & & & & H_{n-1,n-1} & N_{n-1,1} & & & & 0 & & N_{n-1,n-1} \\ M_{11} & \cdots & \cdots & \cdots & \cdots & M_{1,n-1} & T_{11} & \cdots & \cdots & \cdots & 0 & \cdots & T_{1,n-1} \\ \vdots & & & & & \vdots & \vdots & & & & 0 & & \vdots \\ M_{k,1} & & & B_{k,m} & & M_{k,n-1} & T_{k,1} & & & & D_{k,m} & & T_{k,n-1} \\ \vdots & & & & & \vdots & \vdots & & & & 0 & & \vdots \\ M_{m,1} & & B_{m,k} & & & M_{m,n-1} & T_{m,1} & & D_{m,k} & & D_{m,m} & & T_{m,n-1} \\ \vdots & & & & & \vdots & \vdots & & & & 0 & & \vdots \\ M_{n-1,1} & \cdots & \cdots & \cdots & \cdots & M_{n-1,n-1} & T_{n-1,1} & \cdots & \cdots & \cdots & 0 & \cdots & T_{n-1,n-1} \end{bmatrix} \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_k \\ \vdots \\ \Delta\theta_m \\ \vdots \\ \Delta\theta_{n-1} \\ \frac{\Delta V_1}{V_1} \\ \vdots \\ \frac{\Delta V_k}{V_k} \\ \vdots \\ \frac{\Delta t_m}{t_m} \\ \vdots \\ \frac{\Delta V_{n-1}}{V_{n-1}} \end{bmatrix}$$

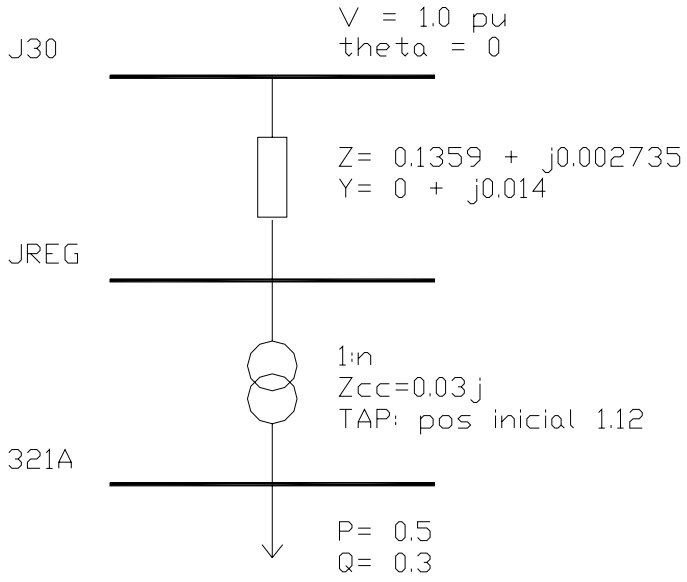
CAPITULO 6

6. PRUEBAS REALIZADAS AL NUEVO METODO INCORPORADO EN FLUCAR V3.0

Para verificar el buen funcionamiento del FLUCAR V3.0 se realizaron diferentes corridas y se compararon con el FLUCAR 2.0 existente en el IIE.

6.1. Prueba 1

Como primera corrida se tomó una red radial formada por 3 barras y un regulador. El esquema unifilar es el siguiente:



El archivo de entrada de datos es el siguiente:

```
{       PRUEBA NUMERO 1 DEL FLUCAR
        RED FICTICIA
     A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P        Q        V   delta   Vmin  Vmax  Qmin  Qmax }

              J30.  1     0        0        1    0    N    N
             321A   2    -0.5     -0.3      1    0    N    N
             JREG   4     0        0       0.95  0    N    N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                        Tipo 2:  Barra de carga (Datos: P,Q)
                        Tipo 3:  Barra de generaci¢n (Datos: P,V)
                        Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor inicial
                        del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites £nicamente
          (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
          flujo sin l¡mites en una barra se debe escribir la letra N (no
          hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con la
          £nica diferencia que se hace un ajuste del regulador para ajustar la
          tensi¢n.  Si una barra con regulador se corre como tipo 2 no se har
          ajuste de la tensi¢n tom ndose el regulador como un trafo de relaci¢n
          constante e igual al valor de n inicial
}
```

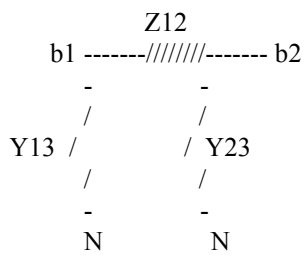
{+IMPEDANCIAS}
{

}

{ Nombre bs bll Z Imax }

{ Z1 S.J. N 0+j1 0 }

+CUADRIPOLOSPI



{ b1 b2 b3 Y13 Z12 Y23 Imax}
cua001 J30. JREG N 0+j0.014030414 0.13587713783+j0.002735222979 0+j0.01403041414

+TRAFOS
{

{ Nombre b1 b2 n Zcc Imax }

{Nota: La corriente m xima admisible por una impedancia, cuadripolo,
otransformador debe anotarse debajo de Imax. En el caso de
no querer establecer restricci¢n sobre la corriente escribir
0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota: n se refiere al valor de la relaci¢n de transformaci¢n de partida.
nmin es el valor m¡nimo de n admisible.
nmax es el valor m ximo de n admisible.
deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre b1 b2 n nmin nmax deltan Zcc Imax }

```
traf002   JREG  321A   1.12  0.95       1.2     0.0005     0+j0.03        0

+TOLERANCIA     0.01

+NITS        50

+FIN.
```

Figura 6.1

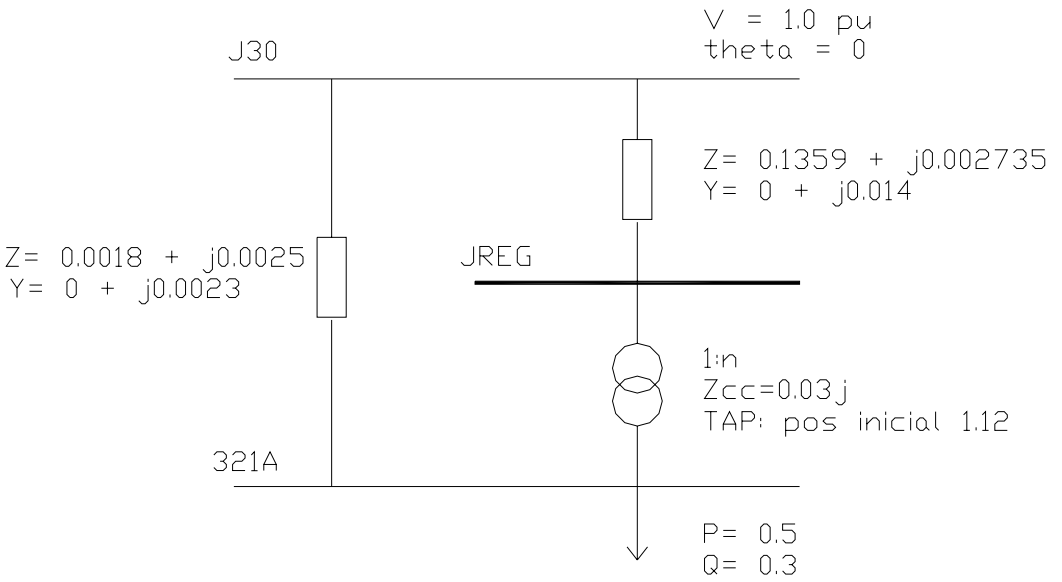
Al realizar esta corrida con el FLUCAR V3.0 dio como resultado que el flujo de carga diverge. Esto sucedío igualmente con el FLUCAR20. Esto se debe a que la convergencia de el flujo de carga resuelto con el método de Newton-Rapshon cuando las redes son radiales no siempre ocurre.

La divergencia de este caso se comprobó matemáticamente observando que la norma del Jacobiano es mayor que uno.

Para eliminar la divergencia se hizo una segunda corrida con una red anillada.

6.2. **Prueba 2**

La segunda corrida consiste en la misma red que el ejemplo anterior con la diferencia de que se le agregó un línea mas para así anillar la red. El esquema unifilar es el siguiente:



El archivo de entrada de datos es el siguiente:

```
{       PRUEBA NUMERO 2 DEL FLUCAR
        RED FICTICIA
```

```
      A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P        Q         V   delta   Vmin  Vmax  Qmin  Qmax }

            J30.    1     0        0          1    0    N    N
            321A    2    -0.5     -0.3        1    0    N    N
            JREG    4     0        0         0.95  0    N    N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                        Tipo 2:  Barra de carga (Datos: P,Q)
                        Tipo 3:  Barra de generaci¢n (Datos: P,V)
                        Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor inicial
                        del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites £nicamente
          (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
          flujo sin l¡mites en una barra se debe escribir la letra N (no
          hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con la
          £nica diferencia que se hace un ajuste del regulador para ajustar la
          tensi¢n.  Si una barra con regulador se corre como tipo 2 no se har
          ajuste de la tensi¢n tom ndose el regulador como un trafo de relaci¢n
          constante e igual al valor de n inicial
}


{+IMPEDANCIAS}
{
                                  Z
                b1---------/////////---------b2

}


                { Nombre   bs     bll    Z      Imax }


           {  Z1     S.J.    N    0+j1      0  }



+CUADRIPOLOSPI
 {                                Z12
                          b1 -------////////------- b2
                             -              -
                             /              /
                        Y13 /              /  Y23
```

```
                                /            /
                                -            -
                                N            N
}


         { b1    b2      b3          Y13                  Z12                                Y23          Imax}
 cua001  J30.   JREG    N    0+j0.014030414  0.13587713783+j0.002735222979   0+j0.01403041414
 cua002  J30.   321A    N    0+j0.023377409  0.0017938322+j0.002519502143    0+j0.023377409



+TRAFOS
{
                                          1:n
                                        - -  - -      Zcc
                              Nodo1  ----- -    -     - ----/////---- Nodo2
                                        - -  - -
 }

{ Nombre     b1   b2      n    Zcc        Imax }


{Nota:   La corriente m xima admisible por una impedancia, cuadripolo,
         otransformador debe anotarse debajo de Imax.  En el caso de
         no querer establecer restricci¢n sobre la corriente escribir
         0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de partida.
         nmin es el valor m¡nimo de n admisible.
         nmax es el valor m ximo de n admisible.
         deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre  b1   b2     n    nmin      nmax    deltan    Zcc        Imax }
traf002   JREG  321A   1.12  0.95       1.2     0.005     0+j0.03       0

+TOLERANCIA     0.01

+NITS        50

+FIN.
```

Figura 6.2

6.2.1. *RESULTADO DEL FLUCAR V3.0*

```
P R O Y E C T O  F L U C A R v 3.0

            I I E  -  2 0 0 1



DATOS: prueba2.dat

RESULTADOS: prueba2.res


SI BIEN LOS CALCULOS SE HACEN EN POR UNIDAD, LOS VALORES DE SALIDA SE
REPRESENTAN EN FUNCION
DE LAS SIGUIENTES BASES:
Potencia base (MVA): 1
Tension base (kV): 1

 RESULTADOS:



  NODO      POTENCIA APARENTE              TENSION
 b: J30.          S:    0.5190 + j    0.2310  V:    1.0000(    0.0000°)
 b: 321A          S:   -0.5000 + j   -0.3000  V:    0.9993(   -0.0027°)
 b: JREG          S:    0.0000 + j    0.0000  V:    0.9500(    0.5939°)
 TSGeneracion:    0.5190 + j    0.2310
 TSConsumo:       0.5000 + j    0.3000

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001     0.3664 + j    0.0658
JREG->cua001    -0.3473 + j   -0.0921

J30.->cua002     0.1526 + j    0.1652
321A->cua002    -0.1525 + j   -0.2117

Perdidas Joule en los cuadripolos:  0.0192104

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
Perdidas Joule en los transformadores:  0.0000000

 REGULADORES
Relación de transformación del regulador traf002:   1.055
JREG->traf002     0.3690 + j    2.3005
321A->traf002    -0.3690 + j   -2.1567
```

```
Perdidas Joule en los reguladores: -0.0000000

Perdidas Joule totales en las líneas:  0.0192104

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 4 converge: TRUE Tiempo:   0.0000s
```

Figura 6.3

6.2.2. *RESULTADO DEL FLUCAR 2.0*

```
P R O Y E C T O   F L U C A R

          I I E  -  1 9 9 7


DATOS: Prueba2.dat

RESULTADOS: Prueba.res


 RESULTADOS:

b: J30.    S:    0.5227811 + j   0.2318109  V:   1.0000000(  0.0000000ø)
b: 321A  S:   -0.5000000 + j  -0.3000000  V:   0.9993187(  0.0009488ø)
b: JREG  S:    0.0000000 + j   0.0000000  V:   0.9455564(  0.6474532ø)
 TSGeneracion:   0.5227811 + j   0.2318109
 TSConsumo:      0.5000000 + j   0.3000000

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001   0.3993820 + j   0.0726445
JREG->cua001  -0.3766880 + j  -0.0987623

J30.->cua002   0.1233991 + j   0.1591664
```

321A->cua002 -0.1233120 + j -0.2057670

Perdidas Joule en los cuadripolos: 0.0227811

POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
Perdidas Joule en los transformadores: 0.0000000

REGULADORES
Relaci¢n de transformaci¢n del regulador traf002: 1.060
JREG->traf002 0.3766880 + j 0.0987624
321A->traf002 -0.3766880 + j -0.0942330

Perdidas Joule en los reguladores: 0.0000000

Perdidas Joule totales en las l¡neas: 0.0227811

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 13 converge: TRUE Tiempo: 0.0000s

Figura 6.4

6.2.3. *COMPARATIVO SOLUCIONES*

En las siguientes tablas se realiza un comparativo entre los resultados obtenidos por ambos metodos:

BARRA	P_FLU30 (PU)	P_FLU20 (PU)	ΔP(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	ΔQ(%)	V_FLU30 (PU)	V_FLU20 (PU)	ΔV(%)	θ_FLU30 (º)	θ_FLU20 (º)	Δθ(%)
J.30	0.519	0.523	**0.77**	0.231	0.232	**0.463**	1.0	1.0	**0.00**	0.00	0.00	**0.00**
JREG	0.0	0.0	**0.00**	0.0	0.0	**-0.00**	0.99	0.99	**0.00**	-0.0027	-0.001	**-0.63**
321A	-0.5	-5.0	**0.00**	-0.3	-0.3	**0.00**	0.95	0.95	**-0.00**	0.5939	0.6475	**0.90**

FLUJO DE CARGA	NUMERO ITERACIONES	TIEMPO(SEG)
FLUCAR30	4	0.0

FLUCAR20	13	0.0

LINEA	P_FLU.30 (PU)	P_FLU.20 (PU)	DIF_P(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	DIF_Q(%)
J30.-JREG.	0.3664	0.3993	**-8.98**	0.0658	0.0726	**-10.33**
J30.-321ª.	0.1526	0.1234	**19.13**	0.1652	0.1591	**3.69**

	FLUCAR30	FLUCAR20	DIFERENCIA PORCENTUAL
Pérdidas en el Sistema (PU)	0.0192	0.0228	**-18.65**

Relación (n) de los reguladores	ZEJERMAN	FLUCAR	DIFERENCIA PORCENTUAL
Trafo JREG-321ª	1.055	1.060	**-0.47**

Al comparar los resultados se observan diferencias entre un metodo y otro. Estas diferencias se acentuan en la potencia entregada por las líneas en las que rondan un 19 %. En la segunda tabla se ve la ventaja del método de Newton-Raphson modificado ya que al FLUCAR20 le llevó 9 iteraciones más para converger al resultado.

6.2.4. PRUEBA 2A

Para obtener una mayor precisión en los resultados el se decidío tomar un delta del tap de 0.005 y un máximo de iteraciones de 200 en el FLUCAR20. Estas son la entrada de datos y los resultados obtenidos:

```
{       PRUEBA NUMERO 2 DEL FLUCAR
        RED FICTICIA


     A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P        Q        V   delta   Vmin  Vmax  Qmin  Qmax }

            J30.    1     0        0          1     0     N     N
            321A    2    -0.5     -0.3        1     0     N     N
            JREG    4     0        0         0.95   0     N     N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                        Tipo 2:  Barra de carga (Datos: P,Q)
                        Tipo 3:  Barra de generación (Datos: P,V)
                        Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber escribirse el valor inicial
```

del cual se quiere que comience la iteraci¢n.

Nota 3: En las columnas de l¡mites deben escribirse dos l¡mites £nicamente (m¡nimo y m ximo de V o de Q). En el caso de querer correr el flujo sin l¡mites en una barra se debe escribir la letra N (no hay l¡mites) en las columnas correspondientes.

Nota 4: Las barras de tipo 4 se comportan como barras de tipo 2 con la £nica diferencia que se hace un ajuste del regulador para ajustar la tensi¢n. Si una barra con regulador se corre como tipo 2 no se har ajuste de la tensi¢n tom ndose el regulador como un trafo de relaci¢n constante e igual al valor de n inicial

}

{+IMPEDANCIAS}
{

```
                    Z
          b1---------/////////---------b2
```

}

{ Nombre bs bll Z Imax }

{ Z1 S.J. N 0+j1 0 }

+CUADRIPOLOSPI

```
 {                          Z12
                    b1 -------/////////------- b2
                       -              -
                       /              /
                  Y13  /              /  Y23
                       /              /
                       -              -
                       N              N
 }
```

	{ b1	b2	b3	Y13	Z12	Y23	Imax}
cua001	J30.	JREG	N	0+j0.014030414	0.13587713783+j0.002735222979	0+j0.01403041414	
cua002	J30.	321A	N	0+j0.023377409	0.0017938322+j0.002519502143	0+j0.023377409	

+TRAFOS
{

```
                                    1:n
                                  - -  - -      Zcc
                  Nodo1  ----- -     -     - ----/////---- Nodo2
```

```
                                        - - - -
 }

{ Nombre     b1    b2      n     Zcc        Imax }


{Nota:   La corriente m xima admisible por una impedancia, cuadripolo,
         otransformador debe anotarse debajo de Imax.  En el caso de
         no querer establecer restricci¢n sobre la corriente escribir
         0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de partida.
         nmin es el valor m¡nimo de n admisible.
         nmax es el valor m ximo de n admisible.
         deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre  b1    b2      n    nmin       nmax    deltan     Zcc        Imax }
traf002   JREG  321A   1.12  0.95        1.2     0.0005     0+j0.03        0

+TOLERANCIA     0.01

+NITS       200

+FIN.
```

Figura 6.5

Esta segunda corrida realizada por el Ing. Mario Vignolo dio el siguiente resultado:

```
P R O Y E C T O   F L U C A R

              I I E   -   1 9 9 7



DATOS: pepito.dat

RESULTADOS: pepito.res


 RESULTADOS:

b: J30.     S:    0.5192935 + j   0.2309959  V:   1.0000000(  0.0000000ø)
b: 321A   S:   -0.5000000 + j  -0.3000000  V:   0.9992524( -0.0025936ø)
b: JREG  S:    0.0000000 + j   0.0000000  V:   0.9499409(  0.5948499ø)
 TSGeneracion:   0.5192935 + j   0.2309959
 TSConsumo:      0.5000000 + j   0.3000000
```

POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
Perdidas Joule en las impedancias: 0.0000000

POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
J30.->cua001 0.3671814 + j 0.0659427
JREG->cua001 -0.3479932 + j -0.0922477

J30.->cua002 0.1521120 + j 0.1650532
321A->cua002 -0.1520068 + j -0.2116254

Perdidas Joule en los cuadripolos: 0.0192935

POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
Perdidas Joule en los transformadores: 0.0000000

REGULADORES
Relaci¢n de transformaci¢n del regulador traf002: 1.055
JREG->traf002 0.3479932 + j 0.0922477
321A->traf002 -0.3479932 + j -0.0883746

Perdidas Joule en los reguladores: 0.0000000

Perdidas Joule totales en las l¡neas: 0.0192935

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 122 converge: TRUE Tiempo: 0.1100s

Figura 6.6

6.2.4.1. Conclusiones

En las siguientes tablas se realiza un comparativo entre los resultados obtenidos por ambos metodos:

BARRA	P_FLU30 (PU)	P_FLU20 (PU)	ΔP(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	ΔQ(%)	V_FLU30 (PU)	V_FLU20 (PU)	ΔV(%)	θ_FLU30 (º)	θ_FLU20 (º)	Δθ(%)
J.30	0.519	0.519	**0.00**	0.231	0.231	**0.00**	1.0	1.0	**0.00**	0.00	0.00	**0.00**
JREG	0.0	0.0	**0.00**	0.0	0.0	**-0.00**	0.99	0.99	**0.00**	-0.0027	-0.0026	**-0.03**
321A	-0.5	-5.0	**0.00**	-0.3	-0.3	**0.00**	0.95	0.95	**-0.00**	0.5939	0.5948	**0.15**

FLUJO DE CARGA	NUMERO ITERACIONES	TIEMPO(SEG)
FLUCAR30	4	0.0
FLUCAR20	122	0.11

LINEA	P_FLU.30 (PU)	P_FLU.20 (PU)	DIF_P(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	DIF_Q(%)
J30.-JREG.	0.3664	0.3672	**-0.22**	0.0658	0.0659	**-0.15**
J30.-321ª.	0.1526	0.1521	**0.32**	0.1652	0.1651	**0.06**

	FLUCAR30	FLUCAR20	DIFERENCIA PORCENTUAL
Pérdidas en el Sistema (PU)	0.0192	0.0193	**-0.52**

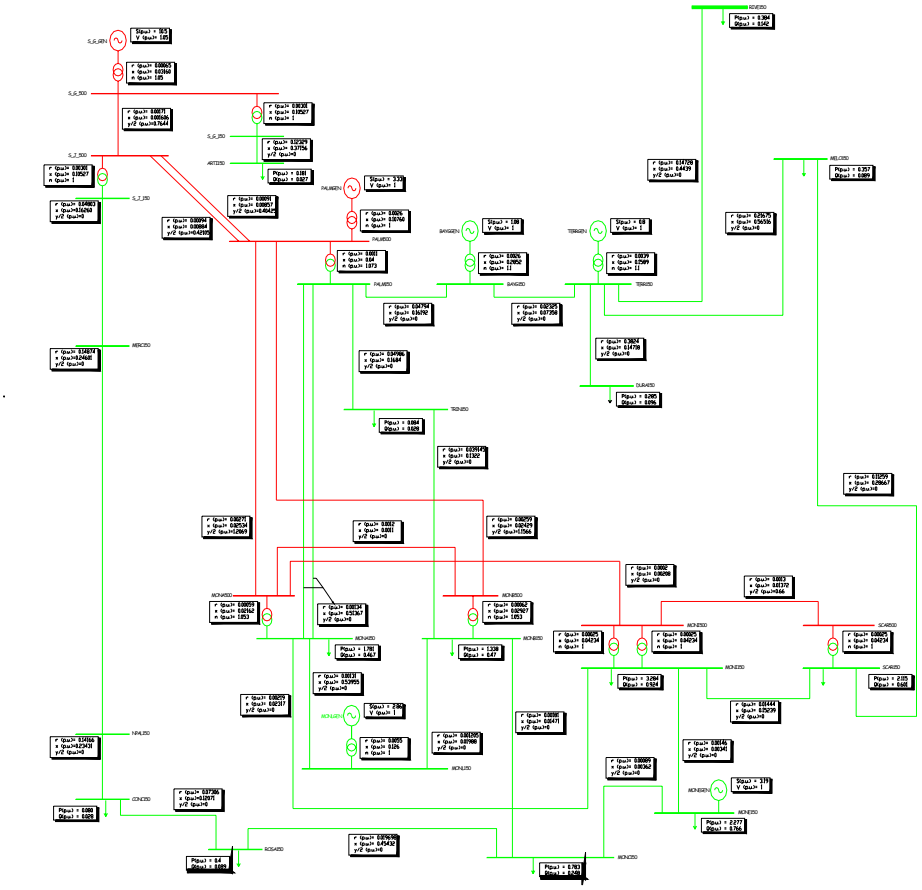
Relación (n) de los reguladores	FLUCAR30	FLU-CAR20	DIFERENCIA PORCENTUAL
Trafo JREG-321ª	1.055	1.055	**0.00**

Como puede apreciarse, los resultados son iguales, a menos de una diferencia porcentual que no supera el 0.5 %. En esta segunda corrida se puede observar la notoria ventaja del metodo de Newthon-Raphson modificado, respecto al metodo clásico, ya que al Flucar 20 le llevó 118 iteraciones más que al Flucar30 converger al resultado.

CAPITULO 7

7. CORRIDA DE CASO PARTICULAR: SISTEMA ELÉCTRICO URUGUAYO

En esta etapa se corrió un ejemplo asignado para probar los cambios realizados en la nueva versión del FLUCAR. El ejemplo planteado consiste en una red **simplificada** del sistema eléctrico uruguayo. Se detalla el esquema unifilar en la figura siguiente:



Para el modelado de los distintos elementos de la red se tuvieron las siguientes consideraciones:

Generadores: Para el tratamiento de los generadores de energía eléctrica se tomaron de forma de tener un generador equivalente con la suma de las potencias aparentes de todas las máquinas en cada usina generadora. Para el modelo equivalente del generador se tomó el mismo como una inyección de potencia activa fija P, y de reactiva Q variable dentro de un rango acotado [Qmin, Qmax] controlada por un valor de consigna de tensión en las barras a las que se conecta el generador.

Generador	No. Unidades	S (MVA)	Pbase	S (pu)
Salto Grande	7	150	100	10.5
Baygorria	3	36	100	1.08
Palmar	3	111	100	3.33
Terra	2	40	100	0.8
Montevideo E	3	4º 62.5	100	3.1875
		5º 100		
		6º 156.25		
Montevideo L	2	142.75	100	

Cargas: Para las cargas a considerar se decidió tomar el peor caso. Para ello se ingresaron las potencias activas y reactivas del pico de invierno pasado. Esta información fue brindada por el Despacho Nacional de Cargas. Se tomo potencia base 100 MVA

Nombre barra carga	Cargas asociadas	P(pu)	Q(pu)	P (pu)	Q (pu)
ARTI150	Arapey	2.9	0.1	0.181	0.027
	T. Gomensoro	5.4	0.3		
	Artigas	9.8	2.3		
MERC150	Mercedes	17.2	4.7	0.172	0.047
NPAL150	Nueva Palmira	15.1	4.2	0.151	0.042
CONCH150	Conchillas	8	2.8	0.080	0.028
RIVE150	Tacuarembó	18.5	6.3	0.384	0.142
	Rivera	19.9	7.9		
MELO150	Valentinez	6.5	0.8	0.457	0.089
	Treinta y Tres	19.9	5.8		
	Melo	13.9	3.8		
	E. Martinez	5.4	1.5		
TRINI150	Trinidad	8.4	2.8	0.084	0.028
DURA150	Durazno	14	4.1	0.285	0.096
	Florida	18.8	5.5		
MONB150	Aguas Corrientes	12.5	5.3	1.338	0.44
	Rodriguez	24.7	9.5		
	Montevideo B	37.1	13.5		
	Las Piedras	43.6	15.2		
	Santiago Vazquez	15.9	3.5		
MONA150	Montevideo A AC	25.9	7.1	1.781	0.467
	Montevideo A AL	3.2	1.4		
	Montevideo N	149	38.2		
MONI150	Montevideo R	8.2	2	3.284	0.924
	Montevideo I	0	0		
	Montevideo H	101	25.2		
	Montevideo F	138.7	43.8		
	Montevideo K	50	13.6		
	Solymar	30.5	7.8		
MONC150	Montevideo C	78.3	24.8	0.783	0.248
MONE150	Montevideo J	68.4	24.3	2.271	0.766
	Montevideo E	158.7	52.3		
SCAR150	Bifurcación	37.9	14.5	2.115	0.601
	San Carlos	6.6	1.8		
	Maldonado	37.2	9.9		
	Punta del Este	27.1	5.2		
	Rocha	24.4	3.9		
ROSA150	Colonia	21.4	6.7	0.559	0.124

	Rosario	24.3	3.9		
	Libertad	9.5	2.4		
	Efice	6.2	2.6		

Líneas: Al tomar un sistema eléctrico simplificado, en algunos casos se tomaron los diferentes tramos de líneas, de forma de unirlos en una línea sola entre barras.

Red de 500 kV

Línea	R (pu)	X (pu)	Y/2 (pu)
SG500-SJ500	0.00171	0.01606	0.7644
SJ500- PALM500 (L1)	0.00094	0.00884	0.42105
SJ500- PALM500 (L2)	0.00091	0.00857	0.40425
MONB500-MONA500	0.0012	0.0011	0
MONA500-MONI500	0.0002	0.00208	0
MONI500-SCAR500	0.01444	0.15239	0
PALM500-MONA500	0.00271	0.02534	1.20698
PALM500-MONB500	0.00259	0.02534	1.156575

Red de 150 kV

Línea	Trámo de línea				Totales de la línea		
	Tramo	R (pu)	X (pu)	Y/2 (pu)	R (pu)	X (pu)	Y/2 (pu)
SG150-ARTI150	SG-arapey	0.03545	0.10683	0	0.12329	0.37156	0
	Arapey-T.gom	0.03304	0.09957	0			
	T.Gom-Artigas	0.0547	0.16516	0			
SJ150-MERC150	S.Javier-F.Bentos	0.03743	0.12673	0	0.04803	0.1626	0
	F.Bentos-Mercedes	0.0106	0.03587	0			
MERC150-NPAL150					0.14874	0.24601	0
NPAL150-CONC150					0.14166	0.23431	0
TERR150-RIVE150	Terra-Tacuarembó	0.08275	0.2494	0	0.14728	0.4439	0
	Tacuarembó-Rivera	0.06453	0.1945	0			
TERR150-MELO150	Terra-Valentinez	0.08694	0.262	0	0.21675	0.56516	0
	Valentinez-T.y Tres	0.06663	0.11275	0			
	T.y Tres-Melo	0.06318	0.19041	0			
PAL150-TRINI150					0.04986	0.1684	0
TERR150-DURA150					0.03824	0.14718	0
TRIN150-MONB150	Trinidad-Rodriguez	0.05226	0.1765	0	0.039145	0.1322	0

	Rodriguez-Mont B	0.02603	0.0879	0			
MONB-MONC	Mont B-Mont C (L1)	0.00362	0.02948	0	0.00181	0.01471	0
	Mont B-Mont C (L2)	0.00362	0.02948	0			
MONI150-SCAR150					0.01444	0.15239	0
MONI150-MONI150	Mont B-Mont H (L1)	0.00085	0.00192	0	0.002078	0.0048	0
	Mont B-Mont H (L2)	0.00059	0.00124	0			
	Mont B-Mont H (L3)	0.00059	0.00124	0			
	Mont H-Mont F (L1)	0.00084	0.00189	0			
	Mont H-Mont F (L2)	0.00076	0.000178	0			
	Mont F-Mont E	0.00116	0.00341	0			
MONC150-MONE150	Mont C-Mont D (L1)	0.00056	0.00454	0	0.00089	0.003615	0
	Mont C-Mont D (L2)	0.00056	0.00454	0			
	Mont D-Mont E (L1)	0.00122	0.00269	0			
	Mont D-Mont E (L2)	0.00122	0.00269	0			
MONA150-MONI150					0.00219	0.02317	0
MONC150-ROSA150	Mont C-S. Vazquez	0.01216	0.02	0	0.19698	0.45432	0
	S. Vazquez-Libertad	0.02584	0.12908	0			
	Libertad-J. Lacaze	0.03929	0.06492	0			
	J. Lacaze- Colonia	0.02603	0.06492	0			
	Colonia-Rosario	0.02603	0.08596	0			
PALM150-MONA150	Palmar-Durazno	0.03824	0.14718	0	0.134007	0.51367	0
	Durazno-Progreso	0.08777	0.33782	0			
	Progreso-Mont A	0.00799	0.02867	0			
MONA150-MONL150	Mont A – AB150T22	0.00063	0.000685	0	0.00131	0.053955	0
	AB150T22-Mont L	0.00063	0.04885	0			
MONA150-MONL150	Mont B – AB150T22	0.00052	0.00425	0	0.001205	0.01998	0
	AB150T22-Mont L	0.00063	0.04885	0			
PALM150-MONA150	Palmar-Florida	0.08783	0.33805	0	0.134	0.51367	
	Florida-Progreso	0.03818	0.14695	0			
	Progreso-Mont A	0.00799	0.02867	0			
PALM150-BAYG150					0.04794	0.16192	0
BAYG150-TERR150					0.02325	0.07358	0
MONI150-SCAR150					0.01444	0.15329	0

Transformadores: Se tomaron los siguientes datos brindados por el Despacho Nacional de Cargas.

Barra 1	Barra 2	N	Impedancia cortocircuito
SJ500	SJ150	1	0.00301+j0.10527
SG500	SG150	1	0.00301+j0.10527

PALM500	PALM150	1.073	0.00110+j0.04
MONB500	MONB150	1.053	0.00062+j0.02927
MONA500	MONA150	1.053	0.00059+j0.02162
MONI500	MONI150	1	0.00025+j0.04234
MONI500	MONI150	1	0.00025+j0.04234
SCAR500	SCAR150	1	0.00025+j0.04234
S.G.GEN	S.G.500	1.05	0.00065+j0.0316
PALMGEN	PALM500	1	0.00260+j0.10760
MONEGEN	MONE150	1.033	0.0055+j0.126
BAYGGEN	BAYG150	1.1	0.0026+j0.2852
TERRGEN	TERR150	1.1	0.0039+j0.2852
MONLGEN	MONL150	1	0.0055+j0.126

Se realizó una primer corrida con la versión 3.0 en la cual no se colocaron reguladores obteniendose los siguientes resultados:

7.1. Entrada de datos

Archivo de entrada de datos:

```
{      SISTEMA ELECTRICO URUGUAYO



           A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom    Tipo  P        Q         V   delta   Vmin  Vmax  Qmin
Qmax }
       S_G_GEN  3    10.5     0         1      0     N     N
       S_G_500  1    0        0         1      0     N     N
       S_J_500  2    0        0         1      0     N     N
       S_G_150  2    0        0         1      0     N     N
       MERC150  2    -0.172   -0.047    1      0     N     N
       NPAL150  2    -0.151   -0.042    1      0     N     N
       CONC150  2    -0.080   -0.028    1      0     N     N
       PALM500  2    0        0         1      0     N     N
       S_J_150  2    0        0         1      0     N     N
       PALM150  2    0        0         1      0     N     N
       ARTI150  2    -0.181   -0.027    1      0     N     N
       TRIN150  2    -0.084   -0.028    1      0     N     N
       DURA150  2    -0.285   -0.096    1      0     N     N
       RIVE150  2    -0.384   -0.142    1      0     N     N
       MELO150  2    -0.357   -0.089    1      0     N     N
       MONA500  2    0        0         1      0     N     N
       MONB500  2    0        0         1      0     N     N
       MONB150  2    -1.338   -0.47     1      0     N     N
       MONA150  2    -1.781   -0.467    1      0     N     N
       MONI500  2    0        0         1      0     N     N
```

```
      MONI150  2    -3.284  -0.924   1      0     N     N
      MONC150  2    -0.783  -0.248   1      0     N     N
      MONL150  2    0       0        1      0     N     N
      ROSA150  2    -0.4    -0.089   1      0     N     N
      MONE150  2    -2.277  -0.766   1      0     N     N
      SCAR500  2    0       0        1      0     N     N
      SCAR150  2    -2.115  -0.601   1      0     N     N
      MONEGEN  3    3.1875  0        1      0     N  N
      PALMGEN  3    3.33    0        1      0     N  N
      MONLGEN  3    2.855   0        1      0     N  N
      TERRGEN  3    0.8     0        1      0     N  N
      BAYGGEN  3    1.08    0        1      0     N  N
      TERR150  2    0       0        1      0     N  N
      BAYG150  2    0       0        1      0     N  N


{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                  Tipo 2:  Barra de carga (Datos: P,Q)
                  Tipo 3:  Barra de generaci¢n (Datos: P,V)
                  Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor
inicial
                  del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites
£nicamente
      (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
      flujo sin l¡mites en una barra se debe escribir la letra N (no
      hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con
la
      £nica diferencia que se hace un ajuste del regulador para
ajustar la
      tensi¢n.  Si una barra con regulador se corre como tipo 2 no se
har
      ajuste de la tensi¢n tom ndose el regulador como un trafo de
relaci¢n
      constante e igual al valor de n inicial
}


+IMPEDANCIAS
{
                        Z
            b1---------/////////---------b2

}


            { Nombre    bs           bll            Z             Imax


                  {Z1      S_G_GEN     S_G_500         0+j0.00001     0
                  Z2      PALMGEN     PALM500         0+j0.00001     0
                  Z3      MONLGEN     MONL150         0+j0.00001     0}
                  Z4      MONEGEN     MONE150         0+j0.00001     0
               {Z5      BAYGGEN     BAYG150         0+j0.00001  0
                  Z6      TERRGEN     TERR150         0+j0.00001     0}
```

```
+CUADRIPOLOSPI
 {                              Z12
                  b1 -------////////------- b2
                       -                  -
                       /                  /
                    Y13  /                  /  Y23
                       /                  /
                       -                  -
                       N                  N
 }
```

	{ b1	b2	b3	Y13	Z12	Y23	Imax}
cua001	S_G_500	S_J_500	N	0+j0.7644	0.00171+j0.01606		
0+j0.7644	0						
cua002	S_J_500	PALM500	N	0+j0.42105	0.00094+j0.00884		
0+j0.42105	0						
cua003	S_J_500	PALM500	N	0+j0.40425	0.00091+j0.00857		
0+j0.40425	0						
cua004	S_J_150	MERC150	N	0+j0	0.04803+j0.16260	0+j0	
0							
cua005	MERC150	NPAL150	N	0+j0	0.14874+j0.24601	0+j0	
0							
cua006	NPAL150	CONC150	N	0+j0	0.14166+j0.23431	0+j0	
0							
cua007	TERR150	RIVE150	N	0+j0	0.14728+j0.4439	0+j0	
0							
cua008	TERR150	MELO150	N	0+j0	0.21675+j0.56516	0+j0	
0							
cua009	PALM150	TRIN150	N	0+j0	0.04986+j0.1684	0+j0	
0							
cua010	TERR150	DURA150	N	0+j0	0.03824+j0.14718	0+j0	
0							
cua011	TRIN150	MONB150	N	0+j0	0.039145+j0.1322	0+j0	
0							
cua012	MONB150	MONC150	N	0+j0	0.00181+j0.01471	0+j0	
0							
cua013	MONB500	MONA500	N	0+j0	0.0012+j0.0011	0+j0	
0							
cua014	MONA500	MONI500	N	0+j0	0.0002+j0.00208	0+j0	
0							
cua015	MONI500	SCAR500	N	0+j0.66	0.0013+j0.01372	0+j0.66	
0							
cua016	MONI150	MONE150	N	0+j0	0.00146+j0.00341	0+j0	
0							
cua017	MONC150	MONE150	N	0+j0	0.00089+j0.003615	0+j0	
0							
cua018	MONA150	MONI150	N	0+j0	0.00219+j0.02317	0+j0	
0							
cua019	MONC150	ROSA150	N	0+j0	0.19698+j0.45432	0+j0	
0							
cua020	PALM500	MONA500	N	0+j1.20698	0.00271+j0.02534		
0+j1.20698	0						
cua021	PALM150	MONA150	N	0+j0	0.00134+j0.51367	0+j0	
0							
cua022	MONA150	MONL150	N	0+j0	0.00131+j0.053955	0+j0	
0							
cua023	MONB150	MONL150	N	0+j0	0.001205+j0.01998	0+j0	
0							
cua024	PALM500	MONB500	N	0+j1.156575	0.00259+j0.02429	0+j0	
0							
cua025	PALM150	MONA150	N	0+j0	0.00134+j0.51367	0+j0	
0							

```
 cua026  S_G_150  ARTI150   N        0+j0         0.12329+j0.37156     0+j0
0
 cua027  PALM150  BAYG150   N        0+j0         0.04794+j0.16192     0+j0
0
 cua028  BAYG150  TERR150   N        0+j0         0.02325+j0.07358     0+j0
0
 cua029  MONI150  SCAR150   N        0+j0         0.01444+j0.15239     0+j0
0
 cua030  SCAR150  MELO150   N        0+j0         0.11259+j0.28667     0+j0
0


+TRAFOS
{
                                          1:n
                                       -  -   -  -        Zcc
                  Nodo1  ----- -      -      - ----/////---- Nodo2
                                       -  -   -  -
 }

{ Nombre      b1       b2         n     Zcc                  Imax }
  traf001     S_J_500  S_J_150    1     0.00301+j0.10527     0
  traf002     S_G_500  S_G_150    1     0.00301+j0.10527     0
  traf003     PALM500  PALM150    1.073 0.00110+j0.04000     0
  traf004     MONB500  MONB150    1.053 0.00062+j0.02927     0
  traf005     MONA500  MONA150    1.053 0.00059+j0.02162     0
  traf006     MONI500  MONI150    1     0.00025+j0.04234     0
  traf007     MONI500  MONI150    1     0.00025+j0.04234     0
  traf008     SCAR500  SCAR150    1     0.00025+j0.04234     0
  traf009     S_G_GEN  S_G_500    1.05  0.00065+j0.03160     0
  traf010     PALMGEN  PALM500    1     0.00260+j0.10760     0
  traf011     MONLGEN  MONL150    1     0.0055 +j0.126       0
{ traf012     MONEGEN  MONE150    1.1    0.002 +j0.126       0}
 traf013     BAYGGEN  BAYG150    1.1   0.0026 +j0.2852      0
  traf014     TERRGEN  TERR150    1.1   0.0039 +j0.1589      0

{Nota:      La corriente m xima admisible por una impedancia,
cuadripolo,
      o transformador debe anotarse debajo de Imax.  En el caso de
      no querer establecer restricci¢n sobre la corriente escribir
      0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de
partida.
      nmin es el valor m¡nimo de n admisible.
      nmax es el valor m ximo de n admisible.
      deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre   b1    b2      n    nmin       nmax    deltan     Zcc
Imax

+TOLERANCIA     0.1

+NITS         20

+FIN.
```

7.2. Resultados

Registrandose los siguientes resultados:

```
            P  R  O  Y  E  C  T  O     F  L  U  C  A  R  v 3.0

                        I  I  E     -     2  0  0  1


DATOS: REDURUF.DAT

RESULTADOS: REDURUF.res


SI BIEN LOS CALCULOS SE HACEN EN POR UNIDAD, LOS VALORES DE SALIDA SE
REPRESENTAN EN FUNCION
DE LAS SIGUIENTES BASES:
Potencia base (MVA): 1
Tension base (kV): 1

 RESULTADOS:

   NODO        POTENCIA APARENTE                  TENSION
b: S_G_GEN  S:     10.5000 + j     3.1267  V:     1.0000(   18.3043°)
b: S_G_500  S:     -7.6507 + j    -1.9140  V:     1.0000(    0.0000°)
b: S_J_500  S:      0.0000 + j     0.0000  V:     1.0207(   -2.4829°)
b: S_G_150  S:      0.0000 + j     0.0000  V:     0.9950(   -1.1175°)
b: MERC150  S:     -0.1720 + j    -0.0470  V:     0.9541(   -8.6278°)
b: NPAL150  S:     -0.1510 + j    -0.0420  V:     0.8941(  -11.7468°)
b: CONC150  S:     -0.0800 + j    -0.0280  V:     0.8735(  -12.8309°)
b: PALM500  S:      0.0000 + j     0.0000  V:     1.0206(   -3.0027°)
b: S_J_150  S:      0.0000 + j     0.0000  V:     1.0004(   -4.9676°)
b: PALM150  S:      0.0000 + j     0.0000  V:     1.0705(   -3.0497°)
b: ARTI150  S:     -0.1810 + j    -0.0270  V:     0.9590(   -4.9589°)
b: TRIN150  S:     -0.0840 + j    -0.0280  V:     1.0320(   -5.8699°)
b: DURA150  S:     -0.2850 + j    -0.0960  V:     1.0237(    0.2122°)
b: RIVE150  S:     -0.3840 + j    -0.1420  V:     0.9033(   -6.8262°)
b: MELO150  S:     -0.3570 + j    -0.0890  V:     0.9379(  -11.2060°)
b: MONA500  S:      0.0000 + j     0.0000  V:     0.9959(   -6.7595°)
b: MONB500  S:      0.0000 + j     0.0000  V:     0.9973(   -6.5770°)
b: MONB150  S:     -1.3380 + j    -0.4700  V:     1.0100(   -7.6657°)
b: MONA150  S:     -1.7810 + j    -0.4670  V:     1.0175(   -8.2317°)
b: MONI500  S:      0.0000 + j     0.0000  V:     0.9964(   -7.1662°)
b: MONI150  S:     -3.2840 + j    -0.9240  V:     0.9969(   -9.1882°)
b: MONC150  S:     -0.7830 + j    -0.2480  V:     1.0003(   -8.9124°)
b: MONL150  S:      0.0000 + j     0.0000  V:     1.0034(   -5.4819°)
b: ROSA150  S:     -0.4000 + j    -0.0890  V:     0.8388(  -20.1964°)
b: MONE150  S:     -2.2770 + j    -0.7660  V:     1.0000(   -8.9783°)
b: SCAR500  S:      0.0000 + j     0.0000  V:     0.9932(   -8.4912°)
b: SCAR150  S:     -2.1150 + j    -0.6010  V:     0.9655(  -12.7166°)
b: MONEGEN  S:      3.1875 + j     1.1308  V:     1.0000(   -8.9764°)
b: PALMGEN  S:      3.3300 + j     0.3288  V:     1.0000(   17.4990°)
b: MONLGEN  S:      2.8550 + j     0.3717  V:     1.0000(   15.4022°)
b: TERRGEN  S:      0.8000 + j     0.3777  V:     1.0000(    8.5073°)
```

```
b: BAYGGEN  S:      1.0800 + j     0.2596  V:     1.0000(   18.5408°)
b: TERR150  S:      0.0000 + j     0.0000  V:     1.0488(    2.2551°)
b: BAYG150  S:      0.0000 + j     0.0000  V:     1.0674(    3.3656°)
 TSGeneracion:    21.7525 + j     5.5953
 TSConsumo:       21.3227 + j     5.9780

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
MONEGEN->Z4     3.1875 + j     1.1308
MONE150->Z4    -3.1875 + j    -1.1307

Perdidas Joule en las impedancias: -0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
S_G_500->cua001     2.5929 + j    -2.2719
S_J_500->cua001    -2.5775 + j     0.8555

S_J_500->cua002     1.0590 + j    -0.5320
PALM500->cua002    -1.0580 + j    -0.3357

S_J_500->cua003     1.0924 + j    -0.5173
PALM500->cua003    -1.0914 + j    -0.3151

S_J_150->cua004     0.4254 + j     0.1716
MERC150->cua004    -0.4153 + j    -0.1374

MERC150->cua005     0.2433 + j     0.0904
NPAL150->cua005    -0.2323 + j    -0.0722

NPAL150->cua006     0.0813 + j     0.0302
CONC150->cua006    -0.0800 + j    -0.0280

TERR150->cua007     0.4142 + j     0.2332
RIVE150->cua007    -0.3840 + j    -0.1420

TERR150->cua008     0.4381 + j     0.0858
MELO150->cua008    -0.3988 + j     0.0166

PALM150->cua009     0.3656 + j     0.1447
TRIN150->cua009    -0.3589 + j    -0.1220

TERR150->cua010     0.2883 + j     0.1087
DURA150->cua010    -0.2850 + j    -0.0960

TRIN150->cua011     0.2749 + j     0.0940
MONB150->cua011    -0.2718 + j    -0.0835

MONB150->cua012     1.5547 + j     0.4914
MONC150->cua012    -1.5500 + j    -0.4530

MONB500->cua013     1.9565 + j    -0.8424
MONA500->cua013    -1.9510 + j     0.8474

MONA500->cua014     3.3343 + j    -0.5467
MONI500->cua014    -3.3320 + j     0.5706

MONI500->cua015     1.6765 + j    -0.5645
SCAR500->cua015    -1.6728 + j    -0.7029

MONI150->cua016    -1.2280 + j    -0.3664
MONE150->cua016     1.2304 + j     0.3720
```

```
MONC150->cua017     0.3200 + j     0.0076
MONE150->cua017    -0.3199 + j    -0.0073

MONA150->cua018     0.8094 + j     0.8318
MONI150->cua018    -0.8066 + j    -0.8016

MONC150->cua019     0.4470 + j     0.1974
ROSA150->cua019    -0.4000 + j    -0.0890

PALM500->cua020     2.7128 + j    -0.4655
MONA500->cua020    -2.6920 + j    -1.7945

PALM150->cua021     0.1918 + j     0.1187
MONA150->cua021    -0.1918 + j    -0.0959

MONA150->cua022    -0.9002 + j     0.3093
MONL150->cua022     0.9014 + j    -0.2621

MONB150->cua023    -1.9035 + j     0.4863
MONL150->cua023     1.9080 + j    -0.4107

PALM500->cua024     2.6949 + j    -0.4316
MONB500->cua024    -2.6754 + j    -0.5899

PALM150->cua025     0.1918 + j     0.1187
MONA150->cua025    -0.1918 + j    -0.0959

S_G_150->cua026     0.1855 + j     0.0405
ARTI150->cua026    -0.1810 + j    -0.0270

PALM150->cua027    -0.7072 + j     0.2746
BAYG150->cua027     0.7313 + j    -0.1933

BAYG150->cua028     0.3461 + j     0.1621
TERR150->cua028    -0.3431 + j    -0.1527

MONI150->cua029     0.4057 + j     0.1794
SCAR150->cua029    -0.4029 + j    -0.1493

SCAR150->cua030    -0.0401 + j     0.1098
MELO150->cua030     0.0418 + j    -0.1056

Perdidas Joule en los cuadripolos:  0.2732790

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
S_J_500->traf001     0.4261 + j     0.1937
S_J_150->traf001    -0.4254 + j    -0.1716

S_G_500->traf002     0.1856 + j     0.0444
S_G_150->traf002    -0.1855 + j    -0.0405

PALM500->traf003     0.0425 + j     0.6719
PALM150->traf003    -0.0421 + j    -0.6568

MONB500->traf004     0.7189 + j     1.4322
MONB150->traf004    -0.7174 + j    -1.3641

MONA500->traf005     1.3088 + j     1.4938
MONA150->traf005    -1.3066 + j    -1.4162
```

```
MONI500->traf006     0.8277 + j    -0.0031
MONI150->traf006    -0.8276 + j     0.0323

MONI500->traf007     0.8277 + j    -0.0031
MONI150->traf007    -0.8276 + j     0.0323

SCAR500->traf008     1.6728 + j     0.7029
SCAR150->traf008    -1.6720 + j    -0.5615

S_G_GEN->traf009    10.5000 + j     3.1267
S_G_500->traf009   -10.4292 + j     0.3135

PALMGEN->traf010     3.3300 + j     0.3288
PALM500->traf010    -3.3009 + j     0.8760

MONLGEN->traf011     2.8550 + j     0.3717
MONL150->traf011    -2.8094 + j     0.6728

BAYGGEN->traf013     1.0800 + j     0.2596
BAYG150->traf013    -1.0773 + j     0.0312

TERRGEN->traf014     0.8000 + j     0.3777
TERR150->traf014    -0.7975 + j    -0.2749

Perdidas Joule en los transformadores:  0.1565376

 REGULADORES
Perdidas Joule en los reguladores:  0.0000000

Perdidas Joule totales en las líneas:  0.4298166

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 3 converge: TRUE Tiempo:   8.1300s
```

7.3. **Modificaciones**

Para elevar la tensión de las barras Nueva Palmira y Conchillas se decidió colocar un regulador de tension en San Javier. Se obtuvieron los siguientes resultados:

7.3.1. *Entrada de datos*

Archivo de entrada de datos:

```
{     SISTEMA ELECTRICO URUGUAYO


          A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P       Q        V   delta   Vmin  Vmax  Qmin
Qmax }
      S_G_GEN  3    10.5    0        1      0     N     N
      S_G_500  1    0       0        1      0     N     N
      S_J_500  2    0       0        1      0     N     N
      S_G_150  2    0       0        1      0     N     N
      MERC150  4    -0.172  -0.047   1      0     N     N
      NPAL150  2    -0.151  -0.042   1      0     N     N
      CONC150  2    -0.080  -0.028   1      0     N     N
      PALM500  2    0       0        1      0     N     N
      S_J_150  2    0       0        1      0     N     N
      PALM150  2    0       0        1      0     N     N
      ARTI150  2    -0.181  -0.027   1      0     N     N
      TRIN150  2    -0.084  -0.028   1      0     N     N
      DURA150  2    -0.285  -0.096   1      0     N     N
      RIVE150  2    -0.384  -0.142   1      0     N     N
      MELO150  2    -0.357  -0.089   1      0     N     N
      MONA500  2    0       0        1      0     N     N
      MONB500  2    0       0        1      0     N     N
      MONB150  2    -1.338  -0.47    1      0     N     N
      MONA150  2    -1.781  -0.467   1      0     N     N
      MONI500  2    0       0        1      0     N     N
      MONI150  2    -3.284  -0.924   1      0     N     N
      MONC150  2    -0.783  -0.248   1      0     N     N
      MONL150  2    0       0        1      0     N     N
      ROSA150  2    -0.4    -0.089   1      0     N     N
      MONE150  2    -2.277  -0.766   1      0     N     N
      SCAR500  2    0       0        1      0     N     N
      SCAR150  2    -2.115  -0.601   1      0     N     N
      MONEGEN  3    3.1875  0        1      0     N  N
      PALMGEN  3    3.33    0        1      0     N  N
      MONLGEN  3    2.855   0        1      0     N  N
      TERRGEN  3    0.8     0        1      0     N  N
      BAYGGEN  3    1.08    0        1      0     N  N
      TERR150  2    0       0        1      0     N  N
      BAYG150  2    0       0        1      0     N  N
      SJAVREG  2    0       0        1      0     N     N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                  Tipo 2:  Barra de carga (Datos: P,Q)
                  Tipo 3:  Barra de generaci¢n (Datos: P,V)
                  Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor
inicial
                  del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites
£nicamente
      (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
```

```
      flujo sin l¡mites en una barra se debe escribir la letra N (no
      hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con
la
      £nica diferencia que se hace un ajuste del regulador para
ajustar la
      tensi¢n.  Si una barra con regulador se corre como tipo 2 no se
har
      ajuste de la tensi¢n tom ndose el regulador como un trafo de
relaci¢n
      constante e igual al valor de n inicial
}


+IMPEDANCIAS
{
                        Z
            b1---------/////////---------b2

}


            { Nombre    bs          bll           Z             Imax


                     {Z1      S_G_GEN     S_G_500          0+j0.00001
      0
                     Z2       PALMGEN     PALM500          0+j0.00001
      0
                     Z3       MONLGEN     MONL150          0+j0.00001
      0}
                     Z4       MONEGEN     MONE150          0+j0.00001
      0
                {Z5       BAYGGEN     BAYG150          0+j0.00001  0
                     Z6       TERRGEN     TERR150          0+j0.00001
      0}



+CUADRIPOLOSPI
 {                            Z12
                  b1 -------/////////------- b2
                       -                  -
                       /                  /
                    Y13  /                  /  Y23
                       /                  /
                       -                  -
                       N                  N
 }


         { b1     b2      b3       Y13            Z12            Y23
Imax}
 cua001  S_G_500  S_J_500   N        0+j0.7644    0.00171+j0.01606
0+j0.7644    0
 cua002  S_J_500  PALM500   N        0+j0.42105   0.00094+j0.00884
0+j0.42105   0
 cua003  S_J_500  PALM500   N        0+j0.40425   0.00091+j0.00857
0+j0.40425   0
 cua004  S_J_150  SJAVREG   N        0+j0         0.04803+j0.16260     0+j0
0
```

```
 cua005  MERC150  NPAL150   N        0+j0          0.14874+j0.24601     0+j0
0
 cua006  NPAL150  CONC150   N        0+j0          0.14166+j0.23431     0+j0
0
 cua007  TERR150  RIVE150   N        0+j0          0.14728+j0.4439      0+j0
0
 cua008  TERR150  MELO150   N        0+j0          0.21675+j0.56516     0+j0
0
 cua009  PALM150  TRIN150   N        0+j0          0.04986+j0.1684      0+j0
0
 cua010  TERR150  DURA150   N        0+j0          0.03824+j0.14718     0+j0
0
 cua011  TRIN150  MONB150   N        0+j0          0.039145+j0.1322     0+j0
0
 cua012  MONB150  MONC150   N        0+j0          0.00181+j0.01471     0+j0
0
 cua013  MONB500  MONA500   N        0+j0          0.0012+j0.0011      0+j0
0
 cua014  MONA500  MONI500   N        0+j0          0.0002+j0.00208     0+j0
0
 cua015  MONI500  SCAR500   N        0+j0.66       0.0013+j0.01372     0+j0.66
0
 cua016  MONI150  MONE150   N        0+j0          0.00146+j0.00341     0+j0
0
 cua017  MONC150  MONE150   N        0+j0          0.00089+j0.003615    0+j0
0
 cua018  MONA150  MONI150   N        0+j0          0.00219+j0.02317     0+j0
0
 cua019  MONC150  ROSA150   N        0+j0          0.19698+j0.45432     0+j0
0
 cua020  PALM500  MONA500   N        0+j1.20698    0.00271+j0.02534
0+j1.20698   0
 cua021  PALM150  MONA150   N        0+j0          0.00134+j0.51367      0+j0
0
 cua022  MONA150  MONL150   N        0+j0          0.00131+j0.053955    0+j0
0
 cua023  MONB150  MONL150   N        0+j0          0.001205+j0.01998    0+j0
0
 cua024  PALM500  MONB500   N        0+j1.156575   0.00259+j0.02429    0+j0
0
 cua025  PALM150  MONA150   N        0+j0          0.00134+j0.51367     0+j0
0
 cua026  S_G_150  ARTI150   N        0+j0          0.12329+j0.37156     0+j0
0
 cua027  PALM150  BAYG150   N        0+j0          0.04794+j0.16192     0+j0
0
 cua028  BAYG150  TERR150   N        0+j0          0.02325+j0.07358     0+j0
0
 cua029  MONI150  SCAR150   N        0+j0          0.01444+j0.15239     0+j0
0
 cua030  SCAR150  MELO150   N        0+j0          0.11259+j0.28667     0+j0
0


+TRAFOS
{
                                              1:n
                                          -  -   -  -        Zcc
                    Nodo1  ----- -      -      - ----/////---- Nodo2
                                          -  -   -  -
 }

{ Nombre      b1       b2         n     Zcc                   Imax }
  traf001     S_J_500  S_J_150    1     0.00301+j0.10527      0
  traf002     S_G_500  S_G_150    1     0.00301+j0.10527      0
```

```
  traf003     PALM500  PALM150    1.073 0.00110+j0.04000     0
  traf004     MONB500  MONB150    1.053 0.00062+j0.02927     0
  traf005     MONA500  MONA150    1.053 0.00059+j0.02162     0
  traf006     MONI500  MONI150    1     0.00025+j0.04234     0
  traf007     MONI500  MONI150    1     0.00025+j0.04234     0
  traf008     SCAR500  SCAR150    1     0.00025+j0.04234     0
  traf009     S_G_GEN  S_G_500    1.05  0.00065+j0.03160     0
  traf010     PALMGEN  PALM500    1     0.00260+j0.10760     0
  traf011     MONLGEN  MONL150    1     0.0055 +j0.126       0
{ traf012     MONEGEN  MONE150    1.1    0.002 +j0.126       0}
 traf013     BAYGGEN  BAYG150    1.1   0.0026 +j0.2852      0
  traf014     TERRGEN  TERR150    1.1   0.0039 +j0.1589      0

{Nota:      La corriente m xima admisible por una impedancia,
cuadripolo,
      o transformador debe anotarse debajo de Imax.  En el caso de
      no querer establecer restricci¢n sobre la corriente escribir
      0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de
partida.
      nmin es el valor m¡nimo de n admisible.
      nmax es el valor m ximo de n admisible.
      deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre   b1    b2      n    nmin       nmax    deltan     Zcc
Imax}
traf002   MERC150 SJAVREG    1  0.9      1.2     0.005     0+j0.03
0

+TOLERANCIA     0.001

+NITS         120

+FIN.
```

7.3.2. *Resultado*

Se obtuvieron los siguientes resultados:

```
              P  R  O  Y  E  C  T  O     F  L  U  C  A  R  v 3.0

                       I  I  E     -     2  0  0  1


DATOS: SJAVNUE.DAT

RESULTADOS: SJAVNUE.res


SI BIEN LOS CALCULOS SE HACEN EN POR UNIDAD, LOS VALORES DE SALIDA SE
REPRESENTAN EN FUNCION
```

```
DE LAS SIGUIENTES BASES:
Potencia base (MVA): 1
Tension base (kV): 1

 RESULTADOS:

   NODO       POTENCIA APARENTE                TENSION
b: S_G_GEN  S:     10.5000 + j     3.1267  V:     1.0000(   18.3043°)
b: S_G_500  S:     -7.6522 + j    -1.9114  V:     1.0000(    0.0000°)
b: S_J_500  S:      0.0000 + j     0.0000  V:     1.0207(   -2.4815°)
b: S_G_150  S:      0.0000 + j     0.0000  V:     0.9950(   -1.1175°)
b: MERC150  S:     -0.1720 + j    -0.0470  V:     1.0132(   -9.2613°)
b: NPAL150  S:     -0.1510 + j    -0.0420  V:     0.9434(  -12.2048°)
b: CONC150  S:     -0.0800 + j    -0.0280  V:     0.9239(  -13.1763°)
b: PALM500  S:      0.0000 + j     0.0000  V:     1.0206(   -3.0013°)
b: S_J_150  S:      0.0000 + j     0.0000  V:     0.9999(   -4.9580°)
b: PALM150  S:      0.0000 + j     0.0000  V:     1.0705(   -3.0484°)
b: ARTI150  S:     -0.1810 + j    -0.0270  V:     0.9590(   -4.9589°)
b: TRIN150  S:     -0.0840 + j    -0.0280  V:     1.0320(   -5.8688°)
b: DURA150  S:     -0.2850 + j    -0.0960  V:     1.0237(    0.2132°)
b: RIVE150  S:     -0.3840 + j    -0.1420  V:     0.9033(   -6.8260°)
b: MELO150  S:     -0.3570 + j    -0.0890  V:     0.9378(  -11.2051°)
b: MONA500  S:      0.0000 + j     0.0000  V:     0.9959(   -6.7584°)
b: MONB500  S:      0.0000 + j     0.0000  V:     0.9973(   -6.5759°)
b: MONB150  S:     -1.3380 + j    -0.4700  V:     1.0100(   -7.6647°)
b: MONA150  S:     -1.7810 + j    -0.4670  V:     1.0175(   -8.2307°)
b: MONI500  S:      0.0000 + j     0.0000  V:     0.9964(   -7.1652°)
b: MONI150  S:     -3.2840 + j    -0.9240  V:     0.9969(   -9.1872°)
b: MONC150  S:     -0.7830 + j    -0.2480  V:     1.0003(   -8.9115°)
b: MONL150  S:      0.0000 + j     0.0000  V:     1.0034(   -5.4809°)
b: ROSA150  S:     -0.4000 + j    -0.0890  V:     0.8388(  -20.1964°)
b: MONE150  S:     -2.2770 + j    -0.7660  V:     1.0000(   -8.9774°)
b: SCAR500  S:      0.0000 + j     0.0000  V:     0.9932(   -8.4902°)
b: SCAR150  S:     -2.1150 + j    -0.6010  V:     0.9654(  -12.7157°)
b: MONEGEN  S:      3.1875 + j     1.1323  V:     1.0000(   -8.9755°)
b: PALMGEN  S:      3.3300 + j     0.3291  V:     1.0000(   17.5011°)
b: MONLGEN  S:      2.8550 + j     0.3717  V:     1.0000(   15.4034°)
b: TERRGEN  S:      0.8000 + j     0.3778  V:     1.0000(    8.5085°)
b: BAYGGEN  S:      1.0800 + j     0.2597  V:     1.0000(   18.5422°)
b: TERR150  S:      0.0000 + j     0.0000  V:     1.0488(    2.2561°)
b: BAYG150  S:      0.0000 + j     0.0000  V:     1.0673(    3.3668°)
b: SJAVREG  S:      0.0000 + j     0.0000  V:     0.9529(   -8.5977°)
 TSGeneracion:    21.7525 + j     5.5974
 TSConsumo:       21.3242 + j     5.9754

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
MONEGEN->Z4     3.1875 + j     1.1323
MONE150->Z4    -3.1875 + j    -1.1322

Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
S_G_500->cua001     2.5915 + j    -2.2693
S_J_500->cua001    -2.5761 + j     0.8528

S_J_500->cua002     1.0591 + j    -0.5328
PALM500->cua002    -1.0580 + j    -0.3348

S_J_500->cua003     1.0925 + j    -0.5181
PALM500->cua003    -1.0914 + j    -0.3142
```

```
S_J_150->cua004     0.4240 + j     0.1759
SJAVREG->cua004    -0.4138 + j    -0.1416

MERC150->cua005     0.2757 + j     0.1259
NPAL150->cua005    -0.2623 + j    -0.1039

NPAL150->cua006     0.0812 + j     0.0300
CONC150->cua006    -0.0800 + j    -0.0280

TERR150->cua007     0.4143 + j     0.2332
RIVE150->cua007    -0.3840 + j    -0.1420

TERR150->cua008     0.4380 + j     0.0858
MELO150->cua008    -0.3988 + j     0.0166

PALM150->cua009     0.3656 + j     0.1447
TRIN150->cua009    -0.3589 + j    -0.1219

TERR150->cua010     0.2883 + j     0.1087
DURA150->cua010    -0.2850 + j    -0.0960

TRIN150->cua011     0.2749 + j     0.0939
MONB150->cua011    -0.2718 + j    -0.0835

MONB150->cua012     1.5547 + j     0.4909
MONC150->cua012    -1.5500 + j    -0.4526

MONB500->cua013     1.9565 + j    -0.8426
MONA500->cua013    -1.9510 + j     0.8477

MONA500->cua014     3.3343 + j    -0.5473
MONI500->cua014    -3.3320 + j     0.5712

MONI500->cua015     1.6765 + j    -0.5645
SCAR500->cua015    -1.6728 + j    -0.7028

MONI150->cua016    -1.2280 + j    -0.3674
MONE150->cua016     1.2304 + j     0.3730

MONC150->cua017     0.3200 + j     0.0072
MONE150->cua017    -0.3199 + j    -0.0068

MONA150->cua018     0.8094 + j     0.8314
MONI150->cua018    -0.8066 + j    -0.8013

MONC150->cua019     0.4470 + j     0.1974
ROSA150->cua019    -0.4000 + j    -0.0890

PALM500->cua020     2.7128 + j    -0.4661
MONA500->cua020    -2.6921 + j    -1.7938

PALM150->cua021     0.1918 + j     0.1187
MONA150->cua021    -0.1918 + j    -0.0959

MONA150->cua022    -0.9002 + j     0.3093
MONL150->cua022     0.9014 + j    -0.2620

MONB150->cua023    -1.9035 + j     0.4863
MONL150->cua023     1.9080 + j    -0.4107
```

```
PALM500->cua024     2.6949 + j    -0.4322
MONB500->cua024    -2.6754 + j    -0.5892

PALM150->cua025     0.1918 + j     0.1187
MONA150->cua025    -0.1918 + j    -0.0959

S_G_150->cua026     0.1855 + j     0.0405
ARTI150->cua026    -0.1810 + j    -0.0270

PALM150->cua027    -0.7072 + j     0.2745
BAYG150->cua027     0.7312 + j    -0.1932

BAYG150->cua028     0.3461 + j     0.1621
TERR150->cua028    -0.3431 + j    -0.1526

MONI150->cua029     0.4057 + j     0.1795
SCAR150->cua029    -0.4029 + j    -0.1493

SCAR150->cua030    -0.0401 + j     0.1098
MELO150->cua030     0.0418 + j    -0.1056

Perdidas Joule en los cuadripolos:  0.2754370

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
S_J_500->traf001     0.4246 + j     0.1981
S_J_150->traf001    -0.4240 + j    -0.1759

S_G_500->traf002     0.1856 + j     0.0444
S_G_150->traf002    -0.1855 + j    -0.0405

PALM500->traf003     0.0426 + j     0.6717
PALM150->traf003    -0.0421 + j    -0.6566

MONB500->traf004     0.7189 + j     1.4318
MONB150->traf004    -0.7175 + j    -1.3637

MONA500->traf005     1.3087 + j     1.4934
MONA150->traf005    -1.3066 + j    -1.4159

MONI500->traf006     0.8277 + j    -0.0034
MONI150->traf006    -0.8276 + j     0.0326

MONI500->traf007     0.8277 + j    -0.0034
MONI150->traf007    -0.8276 + j     0.0326

SCAR500->traf008     1.6728 + j     0.7028
SCAR150->traf008    -1.6720 + j    -0.5615

S_G_GEN->traf009    10.5000 + j     3.1267
S_G_500->traf009   -10.4292 + j     0.3135

PALMGEN->traf010     3.3300 + j     0.3291
PALM500->traf010    -3.3009 + j     0.8757

MONLGEN->traf011     2.8550 + j     0.3717
MONL150->traf011    -2.8094 + j     0.6727

BAYGGEN->traf013     1.0800 + j     0.2597
BAYG150->traf013    -1.0773 + j     0.0311
```

```
TERRGEN->traf014     0.8000 + j     0.3778
TERR150->traf014    -0.7975 + j    -0.2750

Perdidas Joule en los transformadores:  0.1565388

 REGULADORES
Relación de transformación del regulador traf002:   0.949
MERC150->traf002    -0.3727 + j     2.0388
SJAVREG->traf002     0.3727 + j    -1.9132

Perdidas Joule en los reguladores: -0.0000000

Perdidas Joule totales en las líneas:  0.4319758

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 8 converge: TRUE Tiempo:  23.8400s
```

Notese en los resultados la elevación de da tensión en las barras de Nueva Palmira y Conchillas.

Este archivo de entrada de datos se utilizó con el FLUCAR 20 para así corroborar los resultados:

7.3.3. *Comprobación Flucar 20*

7.3.3.1. Entrada de datos

Archivo de entrada de datos:

```
{     SISTEMA ELECTRICO URUGUAYO



          A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom   Tipo  P       Q        V   delta   Vmin  Vmax  Qmin
Qmax }
      S_G_GEN  3    10.5    0        1      0     N     N
```

```
      S_G_500  1    0       0        1      0     N     N
      S_J_500  2    0       0        1      0     N     N
      S_G_150  2    0       0        1      0     N     N
      MERC150  4    -0.172  -0.047   1      0     N     N
      NPAL150  2    -0.151  -0.042   1      0     N     N
      CONC150  2    -0.080  -0.028   1      0     N     N
      PALM500  2    0       0        1      0     N     N
      S_J_150  2    0       0        1      0     N     N
      PALM150  2    0       0        1      0     N     N
      ARTI150  2    -0.181  -0.027   1      0     N     N
      TRIN150  2    -0.084  -0.028   1      0     N     N
      DURA150  2    -0.285  -0.096   1      0     N     N
      RIVE150  2    -0.384  -0.142   1      0     N     N
      MELO150  2    -0.357  -0.089   1      0     N     N
      MONA500  2    0       0        1      0     N     N
      MONB500  2    0       0        1      0     N     N
      MONB150  2    -1.338  -0.47    1      0     N     N
      MONA150  2    -1.781  -0.467   1      0     N     N
      MONI500  2    0       0        1      0     N     N
      MONI150  2    -3.284  -0.924   1      0     N     N
      MONC150  2    -0.783  -0.248   1      0     N     N
      MONL150  2    0       0        1      0     N     N
      ROSA150  2    -0.4    -0.089   1      0     N     N
      MONE150  2    -2.277  -0.766   1      0     N     N
      SCAR500  2    0       0        1      0     N     N
      SCAR150  2    -2.115  -0.601   1      0     N     N
      MONEGEN  3    3.1875  0        1      0     N  N
      PALMGEN  3    3.33    0        1      0     N  N
      MONLGEN  3    2.855   0        1      0     N  N
      TERRGEN  3    0.8     0        1      0     N  N
      BAYGGEN  3    1.08    0        1      0     N  N
      TERR150  2    0       0        1      0     N  N
      BAYG150  2    0       0        1      0     N  N
      SJAVREG  2    0       0        1      0     N     N

{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                  Tipo 2:  Barra de carga (Datos: P,Q)
                  Tipo 3:  Barra de generaci¢n (Datos: P,V)
                  Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor
inicial
                  del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites
£nicamente
      (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
      flujo sin l¡mites en una barra se debe escribir la letra N (no
      hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con
la
      £nica diferencia que se hace un ajuste del regulador para
ajustar la
      tensi¢n.  Si una barra con regulador se corre como tipo 2 no se
har
      ajuste de la tensi¢n tom ndose el regulador como un trafo de
relaci¢n
      constante e igual al valor de n inicial
}
```

```
+IMPEDANCIAS
{
                        Z
             b1---------/////////---------b2

}
```

```
             { Nombre    bs           bll            Z             Imax

                      {Z1       S_G_GEN      S_G_500          0+j0.00001
      0
                       Z2       PALMGEN      PALM500          0+j0.00001
      0
                       Z3       MONLGEN      MONL150          0+j0.00001
      0}
                       Z4       MONEGEN      MONE150          0+j0.00001
      0
                 {Z5       BAYGGEN      BAYG150          0+j0.00001  0
                       Z6       TERRGEN      TERR150          0+j0.00001
      0}
```

```
+CUADRIPOLOSPI
 {                            Z12
                   b1 -------/////////------- b2
                        -                    -
                        /                    /
                     Y13  /                    /  Y23
                        /                    /
                        -                    -
                        N                    N
 }
```

```
          { b1     b2      b3       Y13             Z12             Y23
Imax}
 cua001  S_G_500  S_J_500   N        0+j0.7644    0.00171+j0.01606
0+j0.7644    0
 cua002  S_J_500  PALM500   N        0+j0.42105   0.00094+j0.00884
0+j0.42105   0
 cua003  S_J_500  PALM500   N        0+j0.40425   0.00091+j0.00857
0+j0.40425   0
 cua004  S_J_150  SJAVREG   N        0+j0         0.04803+j0.16260     0+j0
0
 cua005  MERC150  NPAL150   N        0+j0         0.14874+j0.24601     0+j0
0
 cua006  NPAL150  CONC150   N        0+j0         0.14166+j0.23431     0+j0
0
 cua007  TERR150  RIVE150   N        0+j0         0.14728+j0.4439      0+j0
0
 cua008  TERR150  MELO150   N        0+j0         0.21675+j0.56516     0+j0
0
 cua009  PALM150  TRIN150   N        0+j0         0.04986+j0.1684      0+j0
0
 cua010  TERR150  DURA150   N        0+j0         0.03824+j0.14718     0+j0
0
 cua011  TRIN150  MONB150   N        0+j0         0.039145+j0.1322     0+j0
0
```

```
 cua012  MONB150  MONC150   N        0+j0         0.00181+j0.01471     0+j0
0
 cua013  MONB500  MONA500   N        0+j0         0.0012+j0.0011       0+j0
0
 cua014  MONA500  MONI500   N        0+j0         0.0002+j0.00208      0+j0
0
 cua015  MONI500  SCAR500   N        0+j0.66      0.0013+j0.01372      0+j0.66
0
 cua016  MONI150  MONE150   N        0+j0         0.00146+j0.00341     0+j0
0
 cua017  MONC150  MONE150   N        0+j0         0.00089+j0.003615    0+j0
0
 cua018  MONA150  MONI150   N        0+j0         0.00219+j0.02317     0+j0
0
 cua019  MONC150  ROSA150   N        0+j0         0.19698+j0.45432     0+j0
0
 cua020  PALM500  MONA500   N        0+j1.20698   0.00271+j0.02534
0+j1.20698   0
 cua021  PALM150  MONA150   N        0+j0         0.00134+j0.51367      0+j0
0
 cua022  MONA150  MONL150   N        0+j0         0.00131+j0.053955    0+j0
0
 cua023  MONB150  MONL150   N        0+j0         0.001205+j0.01998    0+j0
0
 cua024  PALM500  MONB500   N        0+j1.156575  0.00259+j0.02429     0+j0
0
 cua025  PALM150  MONA150   N        0+j0         0.00134+j0.51367     0+j0
0
 cua026  S_G_150  ARTI150   N        0+j0         0.12329+j0.37156     0+j0
0
 cua027  PALM150  BAYG150   N        0+j0         0.04794+j0.16192     0+j0
0
 cua028  BAYG150  TERR150   N        0+j0         0.02325+j0.07358     0+j0
0
 cua029  MONI150  SCAR150   N        0+j0         0.01444+j0.15239     0+j0
0
 cua030  SCAR150  MELO150   N        0+j0         0.11259+j0.28667     0+j0
0

+TRAFOS
{
                                              1:n
                                          -  -   -  -        Zcc
                    Nodo1  ----- -       -       - ----/////---- Nodo2
                                          -  -   -  -
 }

{ Nombre      b1       b2         n     Zcc                  Imax }
  traf001     S_J_500  S_J_150    1     0.00301+j0.10527     0
  traf002     S_G_500  S_G_150    1     0.00301+j0.10527     0
  traf003     PALM500  PALM150    1.073 0.00110+j0.04000     0
  traf004     MONB500  MONB150    1.053 0.00062+j0.02927     0
  traf005     MONA500  MONA150    1.053 0.00059+j0.02162     0
  traf006     MONI500  MONI150    1     0.00025+j0.04234     0
  traf007     MONI500  MONI150    1     0.00025+j0.04234     0
  traf008     SCAR500  SCAR150    1     0.00025+j0.04234     0
  traf009     S_G_GEN  S_G_500    1.05  0.00065+j0.03160     0
  traf010     PALMGEN  PALM500    1     0.00260+j0.10760     0
  traf011     MONLGEN  MONL150    1     0.0055 +j0.126       0
{ traf012     MONEGEN  MONE150    1.1    0.002 +j0.126       0}
 traf013     BAYGGEN  BAYG150    1.1   0.0026 +j0.2852      0
  traf014     TERRGEN  TERR150    1.1   0.0039 +j0.1589      0
```

```
{Nota:      La corriente m xima admisible por una impedancia, cuadripo-
lo,
      o transformador debe anotarse debajo de Imax.  En el caso de
      no querer establecer restricci¢n sobre la corriente escribir
      0. Para los trafos Imax es la del secundario
}
+REGULADORES

{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de
partida.
      nmin es el valor m¡nimo de n admisible.
      nmax es el valor m ximo de n admisible.
      deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre   b1    b2      n    nmin       nmax    deltan     Zcc
Imax}
traf002   MERC150 SJAVREG    1  0.1      1.2     0.0005     0+j0.03
0

+TOLERANCIA     0.001

+NITS         120

+FIN.
```

7.3.3.2. Salida de datos

Obteniendose los siguientes resultados:

```
            P  R  O  Y  E  C  T  O     F  L  U  C  A  R

                     I  I  E     -     1  9  9  7


DATOS: SJAVNUE.dat

RESULTADOS: SJAVNUE.res


 RESULTADOS:

b: S_G_GEN  S:   10.5000000 + j   3.1266850  V:   1.0000000(
18.3043437°)
b: S_G_500  S:   -7.6519325 + j  -1.9113805  V:   1.0000000(
0.0000000°)
b: S_J_500  S:    0.0000000 + j   0.0000000  V:   1.0206936( -
2.4816865°)
b: S_G_150  S:    0.0000000 + j   0.0000000  V:   0.9949604( -
1.1174980°)
b: MERC150  S:   -0.1720000 + j  -0.0470000  V:   0.9996471( -
9.3891741°)
b: NPAL150  S:   -0.1510000 + j  -0.0420000  V:   0.9430363(-
12.2114245°)
b: CONC150  S:   -0.0800000 + j  -0.0280000  V:   0.9235253(-
13.1837045°)
```

```
b: PALM500  S:    0.0000000 + j   0.0000000  V:   1.0205756( -
3.0014954°)
b: S_J_150  S:    0.0000000 + j   0.0000000  V:   0.9999392( -
4.9596290°)
b: PALM150  S:    0.0000000 + j   0.0000000  V:   1.0705014( -
3.0485785°)
b: ARTI150  S:   -0.1810000 + j  -0.0270000  V:   0.9589943( -
4.9588753°)
b: TRIN150  S:   -0.0840000 + j  -0.0280000  V:   1.0319644( -
5.8690373°)
b: DURA150  S:   -0.2850000 + j  -0.0960000  V:   1.0237073(
0.2129840°)
b: RIVE150  S:   -0.3840000 + j  -0.1420000  V:   0.9032782( -
6.8262152°)
b: MELO150  S:   -0.3570000 + j  -0.0890000  V:   0.9378456(-
11.2053502°)
b: MONA500  S:    0.0000000 + j   0.0000000  V:   0.9958722( -
6.7586233°)
b: MONB500  S:    0.0000000 + j   0.0000000  V:   0.9972918( -
6.5761338°)
b: MONB150  S:   -1.3380000 + j  -0.4700000  V:   1.0099978( -
7.6649669°)
b: MONA150  S:   -1.7810000 + j  -0.4670000  V:   1.0174633( -
8.2308993°)
b: MONI500  S:    0.0000000 + j   0.0000000  V:   0.9963708( -
7.1654153°)
b: MONI150  S:   -3.2840000 + j  -0.9240000  V:   0.9969269( -
9.1874356°)
b: MONC150  S:   -0.7830000 + j  -0.2480000  V:   1.0002986( -
8.9116914°)
b: MONL150  S:    0.0000000 + j   0.0000000  V:   1.0033781( -
5.4811077°)
b: ROSA150  S:   -0.4000000 + j  -0.0890000  V:   0.8388237(-
20.1966115°)
b: MONE150  S:   -2.2770000 + j  -0.7660000  V:   0.9999887( -
8.9775925°)
b: SCAR500  S:    0.0000000 + j   0.0000000  V:   0.9931999( -
8.4904752°)
b: SCAR150  S:   -2.1150000 + j  -0.6010000  V:   0.9654428(-
12.7159721°)
b: MONEGEN  S:    3.1875000 + j   1.1323649  V:   1.0000000( -
8.9757662°)
b: PALMGEN  S:    3.3300000 + j   0.3291065  V:   1.0000000(
17.5008546°)
b: MONLGEN  S:    2.8550000 + j   0.3717368  V:   1.0000000(
15.4031445°)
b: TERRGEN  S:    0.8000000 + j   0.3778357  V:   1.0000000(
8.5082338°)
b: BAYGGEN  S:    1.0800000 + j   0.2596890  V:   1.0000000(
18.5419463°)
b: TERR150  S:    0.0000000 + j   0.0000000  V:   1.0488220(
2.2559222°)
b: BAYG150  S:    0.0000000 + j   0.0000000  V:   1.0673358(
3.3665973°)
b: SJAVREG  S:    0.0000000 + j   0.0000000  V:   0.9528770( -
8.6016551°)
 TSGeneracion:  21.7525000 + j   5.5974179
 TSConsumo:     21.3239325 + j   5.9753805

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
```

```
MONEGEN->Z4   3.1874998 + j   1.1323650
MONE150->Z4  -3.1874998 + j  -1.1322506

Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
S_G_500->cua001   2.5917045 + j  -2.2692649
S_J_500->cua001  -2.5763461 + j   0.8527449

S_J_500->cua002   1.0590529 + j  -0.5327972
PALM500->cua002  -1.0580329 + j  -0.3348221

S_J_500->cua003   1.0924503 + j  -0.5180996
PALM500->cua003  -1.0913996 + j  -0.3142162

S_J_150->cua004   0.4242080 + j   0.1759468
SJAVREG->cua004  -0.4140768 + j  -0.1416487

MERC150->cua005   0.2420768 + j   0.0883206
NPAL150->cua005  -0.2321932 + j  -0.0719736

NPAL150->cua006   0.0811932 + j   0.0299736
CONC150->cua006  -0.0800000 + j  -0.0280000

TERR150->cua007   0.4142570 + j   0.2331943
RIVE150->cua007  -0.3840000 + j  -0.1420000

TERR150->cua008   0.4380483 + j   0.0857641
MELO150->cua008  -0.3987896 + j   0.0166002

PALM150->cua009   0.3656315 + j   0.1446662
TRIN150->cua009  -0.3589044 + j  -0.1219457

TERR150->cua010   0.2883001 + j   0.1087017
DURA150->cua010  -0.2850000 + j  -0.0960000

TRIN150->cua011   0.2749044 + j   0.0939457
MONB150->cua011  -0.2718021 + j  -0.0834688

MONB150->cua012   1.5547493 + j   0.4909059
MONC150->cua012  -1.5500326 + j  -0.4525736

MONB500->cua013   1.9564722 + j  -0.8426517
MONA500->cua013  -1.9509971 + j   0.8476705

MONA500->cua014   3.3343132 + j  -0.5473014
MONI500->cua014  -3.3320108 + j   0.5712464

MONI500->cua015   1.6765240 + j  -0.5645138
SCAR500->cua015  -1.6728326 + j  -0.7028003

MONI150->cua016  -1.2280182 + j  -0.3673927
MONE150->cua016   1.2304318 + j   0.3730299

MONC150->cua017   0.3200231 + j   0.0071496
MONE150->cua017  -0.3199320 + j  -0.0067794

MONA150->cua018   0.8094113 + j   0.8314127
MONI150->cua018  -0.8065630 + j  -0.8012784
```

```
MONC150->cua019   0.4470095 + j   0.1974240
ROSA150->cua019  -0.4000000 + j  -0.0890000

PALM500->cua020   2.7128309 + j  -0.4661434
MONA500->cua020  -2.6920549 + j  -1.7937851

PALM150->cua021   0.1918372 + j   0.1187000
MONA150->cua021  -0.1917777 + j  -0.0958886

MONA150->cua022  -0.9002327 + j   0.3092578
MONL150->cua022   0.9013792 + j  -0.2620351

MONB150->cua023  -1.9034708 + j   0.4862657
MONL150->cua023   1.9080301 + j  -0.4106689

PALM500->cua024   2.6949350 + j  -0.4321924
MONB500->cua024  -2.6753917 + j  -0.5891824

PALM150->cua025   0.1918372 + j   0.1187000
MONA150->cua025  -0.1917777 + j  -0.0958886

S_G_150->cua026   0.1854896 + j   0.0405304
ARTI150->cua026  -0.1810000 + j  -0.0270000

PALM150->cua027  -0.7071674 + j   0.2744864
BAYG150->cua027   0.7312395 + j  -0.1931814

BAYG150->cua028   0.3461092 + j   0.1620516
TERR150->cua028  -0.3431285 + j  -0.1526183

MONI150->cua029   0.4057229 + j   0.1794948
SCAR150->cua029  -0.4028632 + j  -0.1493148

SCAR150->cua030  -0.0401386 + j   0.1098039
MELO150->cua030   0.0417896 + j  -0.1056002

Perdidas Joule en los cuadripolos:  0.2720278

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
S_J_500->traf001   0.4248429 + j   0.1981519
S_J_150->traf001  -0.4242080 + j  -0.1759468

S_G_500->traf002   0.1855992 + j   0.0443639
S_G_150->traf002  -0.1854896 + j  -0.0405304

PALM500->traf003   0.0425539 + j   0.6716607
PALM150->traf003  -0.0421384 + j  -0.6565526

MONB500->traf004   0.7189195 + j   1.4318341
MONB150->traf004  -0.7174764 + j  -1.3637029

MONA500->traf005   1.3087388 + j   1.4934160
MONA150->traf005  -1.3066233 + j  -1.4158934

MONI500->traf006   0.8277434 + j  -0.0033663
MONI150->traf006  -0.8275709 + j   0.0325881

MONI500->traf007   0.8277434 + j  -0.0033663
MONI150->traf007  -0.8275709 + j   0.0325881
```

```
SCAR500->traf008   1.6728326 + j   0.7028003
SCAR150->traf008  -1.6719982 + j  -0.5614891

S_G_GEN->traf009  10.5000000 + j   3.1266850
S_G_500->traf009 -10.4292363 + j   0.3135206

PALMGEN->traf010   3.3300000 + j   0.3291065
PALM500->traf010  -3.3008873 + j   0.8757134

MONLGEN->traf011   2.8550000 + j   0.3717368
MONL150->traf011  -2.8094093 + j   0.6727040

BAYGGEN->traf013   1.0800000 + j   0.2596890
BAYG150->traf013  -1.0773488 + j   0.0311298

TERRGEN->traf014   0.8000000 + j   0.3778357
TERR150->traf014  -0.7974771 + j  -0.2750418

Perdidas Joule en los transformadores:  0.1565395

 REGULADORES
Relaci¢n de transformaci¢n del regulador traf002:   0.949
MERC150->traf002  -0.4140768 + j  -0.1353206
SJAVREG->traf002   0.4140768 + j   0.1416487

Perdidas Joule en los reguladores:  0.0000000

Perdidas Joule totales en las l¡neas:  0.4285673

VERIFICACION DE LIMITES:

No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 107 converge: TRUE Tiempo: 278.3100s
```

7.3.3.3. Conclusiones

En las siguientes tablas se realiza un comparativo entre los resultados obtenidos por ambos metodos:

BARRA	P_FLU30 (PU)	P_FLU20 (PU)	ΔP(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	ΔQ(%)	V_FLU30 (PU)	V_FLU20 (PU)	ΔV(%)	θ_FLU30 (º)	θ_FLU20 (º)	Δθ(%)
S_GGEN	10.5	0.519	**0.00**	3.1267	0.231	**0.00**	1.0	1.0	**0.00**	18.3	0.00	**0.00**
S_G_500	-7.6522	-7.6519	**0.00**	-1.9114	-1.9114	**0.00**	1.00	1.00	**0.00**	0.00	0.00	**0.00**
S_J_500	0	0.0000	**0.00**	0	0.0000	**0.00**	1.02	1.02	**0.00**	-2.48	-2.48	**-0.01**
S_G_150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	0.99	**0.00**	-1.12	-1.12	**0.00**
MERC150	-0.172	-0.1720	**0.00**	-0.047	-0.0470	**0.00**	1.01	1.00	**1.34**	-9.26	-9.39	**-1.38**
NPAL150	-0.151	-0.1510	**0.00**	-0.042	-0.0420	**0.00**	0.94	0.94	**0.04**	-12.20	-12.21	**-0.05**
CONC150	-0.08	-0.0800	**0.00**	-0.028	-0.0280	**0.00**	0.92	0.92	**0.04**	-13.18	-13.18	**-0.06**
PALM500	0	0.0000	**0.00**	0	0.0000	**0.00**	1.02	1.02	**0.00**	-3.00	-3.00	**-0.01**
S_J_150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	1.00	**0.00**	-4.96	-4.96	**-0.03**
PALM150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.07	1.07	**0.00**	-3.05	-3.05	**-0.01**
ARTI150	-0.181	-0.1810	**0.00**	-0.027	-0.0270	**0.00**	0.96	0.96	**0.00**	-4.96	-4.96	**0.00**
TRIN150	-0.084	-0.0840	**0.00**	-0.028	-0.0280	**0.00**	1.03	1.03	**0.00**	-5.87	-5.87	**0.00**
DURA150	-0.285	-0.2850	**0.00**	-0.096	-0.0960	**0.00**	1.02	1.02	**0.00**	0.21	0.21	**0.10**
RIVE150	-0.384	-0.3840	**0.00**	-0.142	-0.1420	**0.00**	0.90	0.90	**0.00**	-6.83	-6.83	**0.00**
MELO150	-0.357	-0.3570	**0.00**	-0.089	-0.0890	**0.00**	0.94	0.94	**0.00**	-11.21	-11.21	**0.00**
MONA500	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	1.00	**0.00**	-6.76	-6.76	**0.00**
MONB500	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	1.00	**0.00**	-6.58	-6.58	**0.00**
MONB150	-1.338	-1.3380	**0.00**	-0.47	-0.4700	**0.00**	1.01	1.01	**0.00**	-7.66	-7.66	**0.00**
MONA150	-1.781	-1.7810	**0.00**	-0.467	-0.4670	**0.00**	1.02	1.02	**0.00**	-8.23	-8.23	**0.00**
MONI500	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	1.00	**0.00**	-7.17	-7.17	**0.00**
MONI150	-3.284	-3.2840	**0.00**	-0.924	-0.9240	**0.00**	1.00	1.00	**0.00**	-9.19	-9.19	**0.00**
MONC150	-0.783	-0.7830	**0.00**	-0.248	-0.2480	**0.00**	1.00	1.00	**0.00**	-8.91	-8.91	**0.00**
MONL150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.00	1.00	**0.00**	-5.48	-5.48	**0.00**
ROSA150	-0.4	-0.4000	**0.00**	-0.089	-0.0890	**0.00**	0.84	0.84	**0.00**	-20.20	-20.20	**0.00**
MONE150	-2.277	-2.2770	**0.00**	-0.766	-0.7660	**0.00**	1.00	1.00	**0.00**	-8.98	-8.98	**0.00**
SCAR500	0	0.0000	**0.00**	0	0.0000	**0.00**	0.99	0.99	**0.00**	-8.49	-8.49	**0.00**
SCAR150	-2.115	-2.1150	**0.00**	-0.601	-0.6010	**0.00**	0.97	0.97	**0.00**	-12.72	-12.72	**0.00**
MONEGEN	3.1875	3.1875	**0.00**	1.1323	1.1324	**-0.01**	1.00	1.00	**0.00**	-8.98	-8.98	**0.00**
PALMGEN	3.33	3.3300	**0.00**	0.3291	0.3291	**0.00**	1.00	1.00	**0.00**	17.50	17.50	**0.00**
MONLGEN	2.855	2.8550	**0.00**	0.3717	0.3717	**-0.01**	1.00	1.00	**0.00**	15.40	15.40	**0.00**
TERRGEN	0.8	0.8000	**0.00**	0.3778	0.3778	**-0.01**	1.00	1.00	**0.00**	8.51	8.51	**0.00**
BAYGGEN	1.08	1.0800	**0.00**	0.2597	0.2597	**0.00**	1.00	1.00	**0.00**	18.54	18.54	**0.00**
TERR150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.05	1.05	**0.00**	2.26	2.26	**0.01**
BAYG150	0	0.0000	**0.00**	0	0.0000	**0.00**	1.07	1.07	**0.00**	3.37	3.37	**0.01**
SJAVREG	0	0.0000	**0.00**	0	0.0000	**0.00**	0.95	0.95	**0.00**	-8.60	-8.60	**-0.05**

FLUJO DE CARGA	NUMERO ITERACIONES	TIEMPO(SEG)
FLUCAR30	8	23.84
FLUCAR20	107	278.31

LINEA	P_FLU.30 (PU)	P_FLU.20 (PU)	DIF_P(%)	Q_FLU30 (PU)	Q_FLU20 (PU)	DIF_Q(%)
S_G_500-S_J_500	2.5915	2.5917	**-0.01**	-2.2693	-2.2693	**0.00**
S_J_500-PALM500	1.0591	1.0591	**0.00**	-0.5328	-0.5328	**0.00**
S_J_500-PALM500	1.0925	1.0925	**0.00**	-0.5181	-0.5181	**0.00**
S_J_150-SJAVREG	0.424	0.4242	**-0.05**	0.1759	0.1759	**-0.03**
MERC150-NPAL150	0.2757	0.2756	**0.02**	0.1259	0.1259	**0.01**
NPAL150-CONC150	0.0812	0.0812	**0.01**	0.03	0.0300	**0.09**
TERR150-RIVE150	0.4143	0.4143	**0.01**	0.2332	0.2332	**0.00**
TERR150-MELO150	0.438	0.4380	**-0.01**	0.0858	0.0858	**0.04**
PALM150-TRINI150	0.3656	0.3656	**-0.01**	0.1447	0.1447	**0.02**
TERR150-DURA150	0.2883	0.2883	**0.00**	0.1087	0.1087	**0.00**
TRINI150-MONB150	0.2749	0.2749	**0.00**	0.0939	0.0939	**-0.05**
MONB150-MONC150	1.5547	1.5547	**0.00**	0.4909	0.4909	**0.00**
MONB500-MONA500	1.9565	1.9565	**0.00**	-0.8426	-0.8427	**-0.01**
MONA500-MONI500	3.3343	3.3343	**0.00**	-0.5473	-0.5473	**0.00**
MONI500-SCAR500	1.6765	1.6765	**0.00**	-0.5645	-0.5645	**0.00**
MONI150-MONE150	-1.228	-1.2280	**0.00**	-0.3674	-0.3674	**0.00**
MONC150-MONE150	0.32	0.3200	**-0.01**	0.0072	0.0071	**0.70**
MONA150-MONI150	0.8094	0.8094	**0.00**	0.8314	0.8314	**0.00**
MONC150-ROSA150	0.447	0.4470	**0.00**	0.1974	0.1974	**-0.01**
PALM500-MONA500	2.7128	2.7128	**0.00**	-0.4661	-0.4661	**-0.01**
PALM150-MONA150	0.1918	0.1918	**-0.02**	0.1187	0.1187	**0.00**
MONA150-MONL150	-0.9002	-0.9002	**0.00**	0.3093	0.3093	**0.01**
MONB150-MONL150	-1.9035	-1.9035	**0.00**	0.4863	0.4863	**0.01**
PALM500-MONB500	2.6949	2.6949	**0.00**	-0.4322	-0.4322	**0.00**
PALM150-MONA150	0.1918	0.1918	**-0.02**	0.1187	0.1187	**0.00**
S_G_150-ARTI150	0.1855	0.1855	**0.01**	0.0405	0.0405	**-0.08**
PALM150-BAYG150	-0.7072	-0.7072	**0.00**	0.2745	0.2745	**0.00**
BAYG150-TERR150	0.3461	0.3461	**0.00**	0.1621	0.1621	**0.03**
MONI150-SCAR150	0.4057	0.4057	**-0.01**	0.1795	0.1795	**0.00**
SCAR150-MELO150	-0.0401	-0.0401	**-0.10**	0.1098	0.1098	**0.00**

	P FLUC 30 (PU)	Q FLUC 20 (PU)	DIFERENCIA PORCENTUAL	Q FLUC 30 (PU)	Q FLUC 20 (PU)	DIFERENCIA PORCENTUAL
TSGeneración (PU)	21.7525	21.7525	**0.00**	5.5974	5.5974	**0.00**
TSConsumo (PU)	21.3242	21.3239	**0.00**	5.9754	5.9754	**0.00**

	FLUCAR30	FLUCAR20	DIFERENCIA PORCENTUAL
Pérdidas en el Sistema	0.4319758	0.4285673	**0.79**

(PU)			

Relación (n) de los reguladores	**FLUCAR30**	**FLUCAR20**	**DIFERENCIA PORCENTUAL**
Trafo MERC150-SJAVREG	0.949	0.949	**0.00**

Como puede apreciarse, los resultados son iguales, a menos de una diferencia porcentual que no supera el 1 %. En esta corrida se puede observar la **notoria ventaja del metodo de Newthon-Raphson modificado** respecto al metodo clásico, ya que al Flucar 20 le llevó 117 iteraciones y el Flucar 30 converger en solo 8.

CAPITULO 8

8. OTRAS MEJORAS INTRODUCIDAS EN FLUCAR V3.0

8.1. INTERFASE GRAFICA DE ENTRADA SALIDA DATOS

8.1.1. *Introducción*

Se desarrolló una interfase gráfica para el manejo de Flucar v3.0 desarrollada en Delphi 5.0.

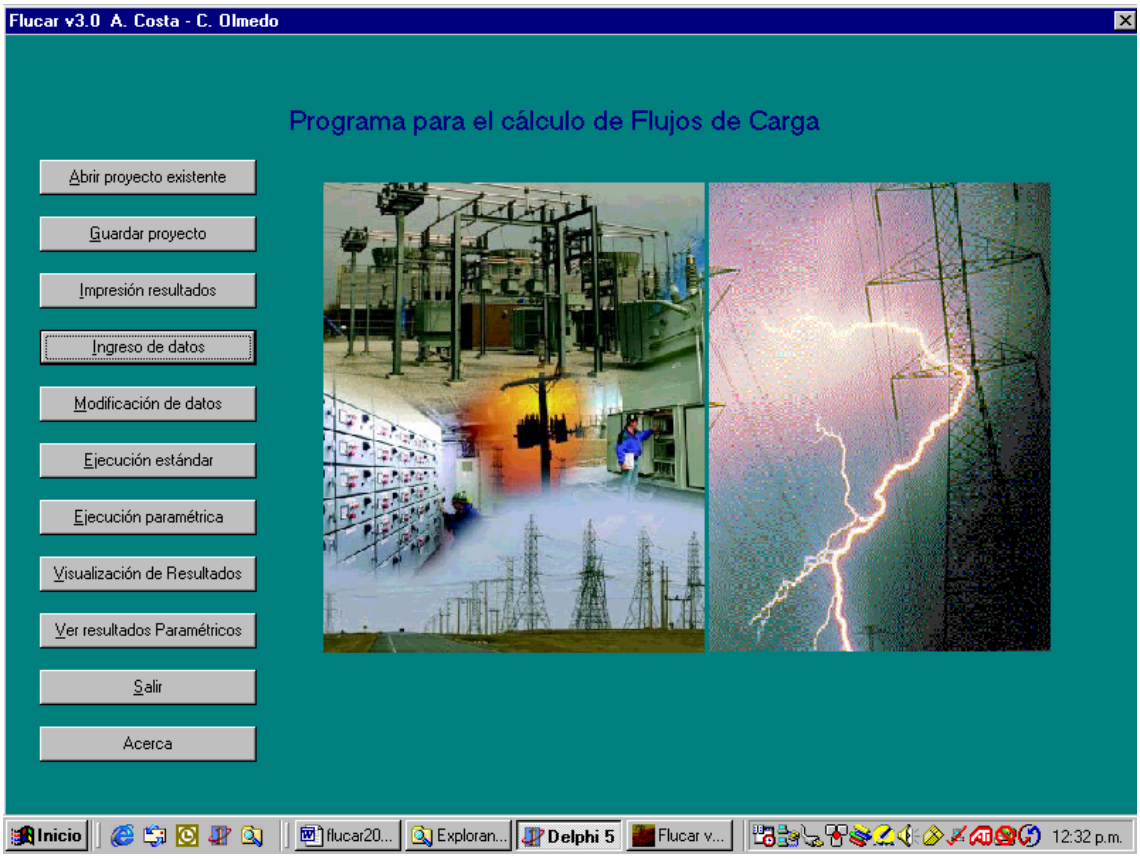
Este programa incorpora un manejo totalmente diferente de datos, estructuras e incluye nuevos tipos de datos que no estaban incluidos en Pascal. Cada ventana que abrimos es una FORMA del tipo Tform. En cada FORMA hay un número de acciones que

son detectadas y a las cuales se puede asociar una acción, por ejemplo: Mover el Raton, Clic con botón del ratón, al Cerrar una ventana, etc. entre varias posibles. Existe un gran número de elementos para captura de datos que ya están implementados. La inclusión en una FORMA es simple y con ella se autoejecutan todas las acciones para permitir el reconocimiento del elemento insertado. Por ejemplo se puede ingresar un Botón (BUTTON), un campo para permitir al usuario ingresar valores (EDIT), un campo donde se restringa el valor que puede ingresar el usuario a una lista (LISTBOX) que debe seleccionar con el ratón, una lista con ingreso de datos combinado ratón o escritura (COMBOBOX), botón de acción del tipo OK, SAVE, etc (BITBTN), entre varios otros. A cada uno de estos elementos se le puede asociar procedimientos, acciones, controles, etc del mismo tipo que descrito en las FORMAS. Cada uno de los elementos se le puede asociar ayuda (HINT), color, estilo, texto a desplegar (CAPTION) entre otros. A botones y formas se le pueden asociar iconos (ICON) y cada ventana tiene un gran número de parámetros que se pueden definir opcionalmente. Es importante acotar que la creación automática de cada uno de estos elementos incluye un número de opciones estándar que nos aseguran que la aplicación al menos lo reconocerá. Evidentemente la personalización para la aplicación es lo que interesa en cada caso.

Cada uno de los datos ingresados, por ejemplo en un campo EDIT son del tipo dinámico Tcaption. El manejo de la información a posteriori requiere la conversión de los datos a formatos conocidos. El manejo de strings es nuevo al igual que el manejo de reales entre otros. Los tipos de datos que maneja Delphi son diferentes: Boolean, byte, char, double, float, handle_t, hyper, long, short, small, wchar_t. Por tanto un nuevo manejo de datos es necesario para conversión de strings a reales (flotas), etc.

Delphi es una herramienta muy poderosa para manejo de entorno gráfico reconociendo y cargando automáticamente casi todo tipo de archivos gráficos (BMP, etc.). También permite el manejo de aplicaciones con bases de datos, acceso a internet, audio entre muchos más.

8.1.2. *Pantalla Principal*

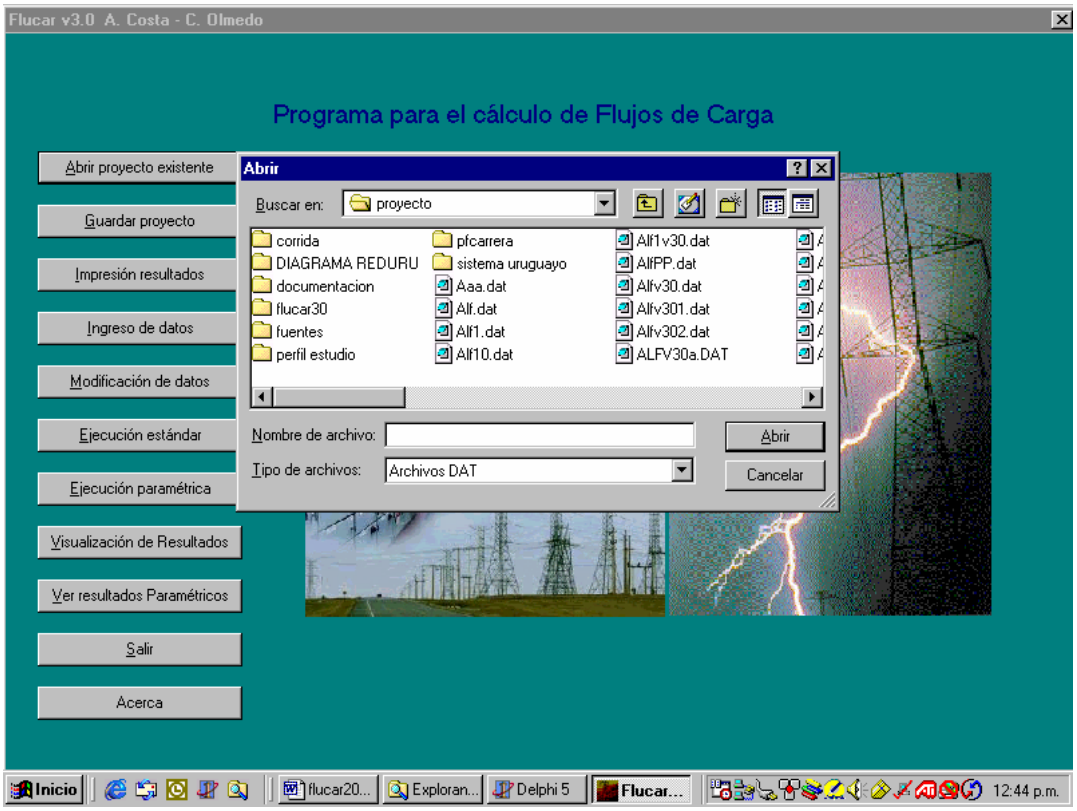


El usuario al ejecutar el programa encuentra un menú de opciones disponibles que iremos analizando. Posicionándonos con el ratón sobre cada botón se despliega una ayuda. A su vez cada botón tien un carácter de acceso rápido. Por ejemplo Abrir archivo existente tiene subrayada la A que permite el acceso rápido con SHIFT+A. El procedimiento se inicia con la apertura de un archivo existente (.DAT)

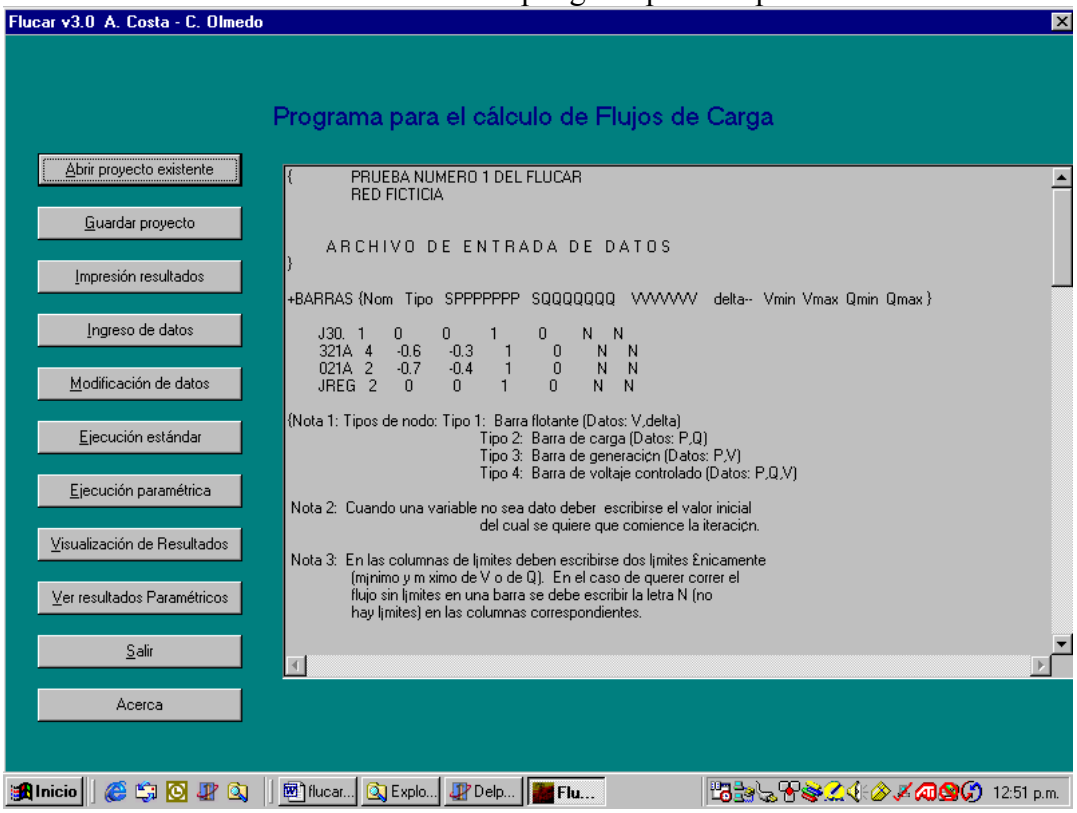
8.1.2.1. Abrir proyecto existente

Esta opción permite Abrir un archivo de datos (DAT) ya creado. El archivo DAT a diferencia de el caso de FLUCAR2.0 debe tener un formato determinado para permitir que Delphi tome valores.

Se despliega una ventana de selección de archivos donde la extensión (.DAT) ya está pre seleccionada típico de aplicaciones entorno Windows.

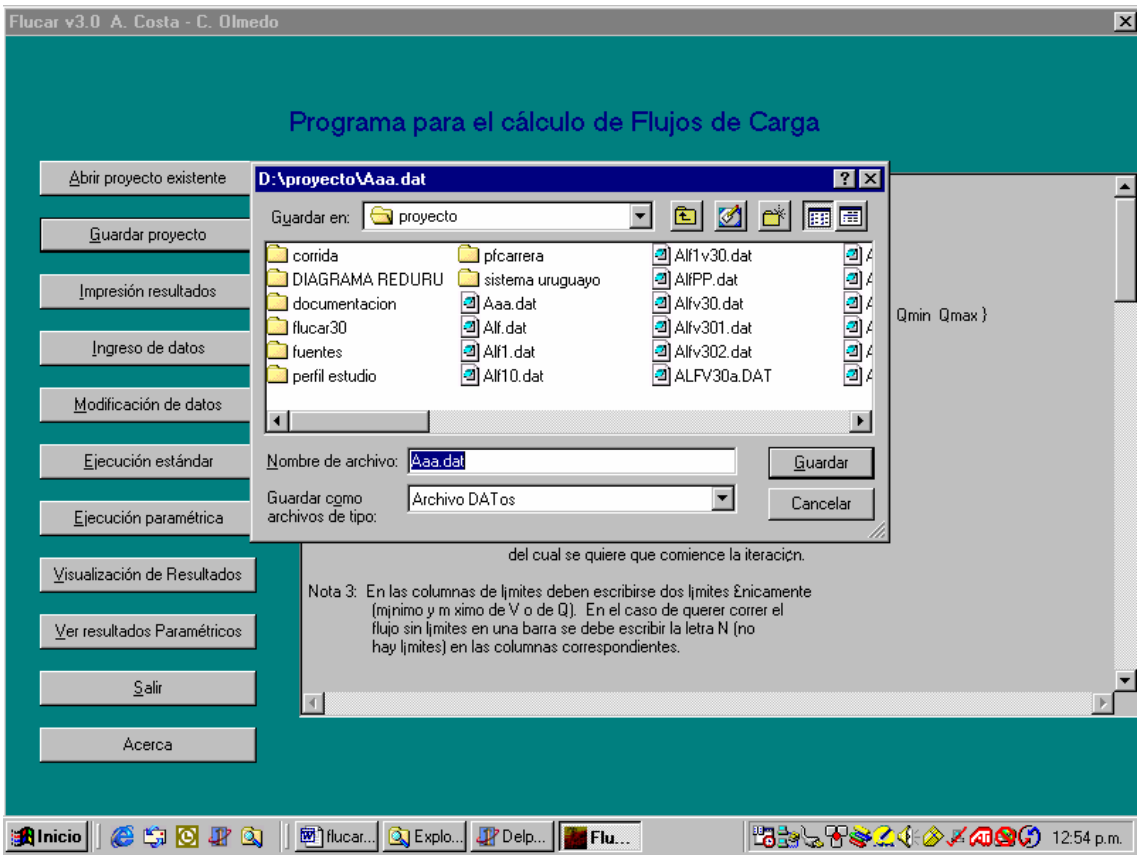


Seleccionado el archivo el mismo se despliega en pantalla para su visión



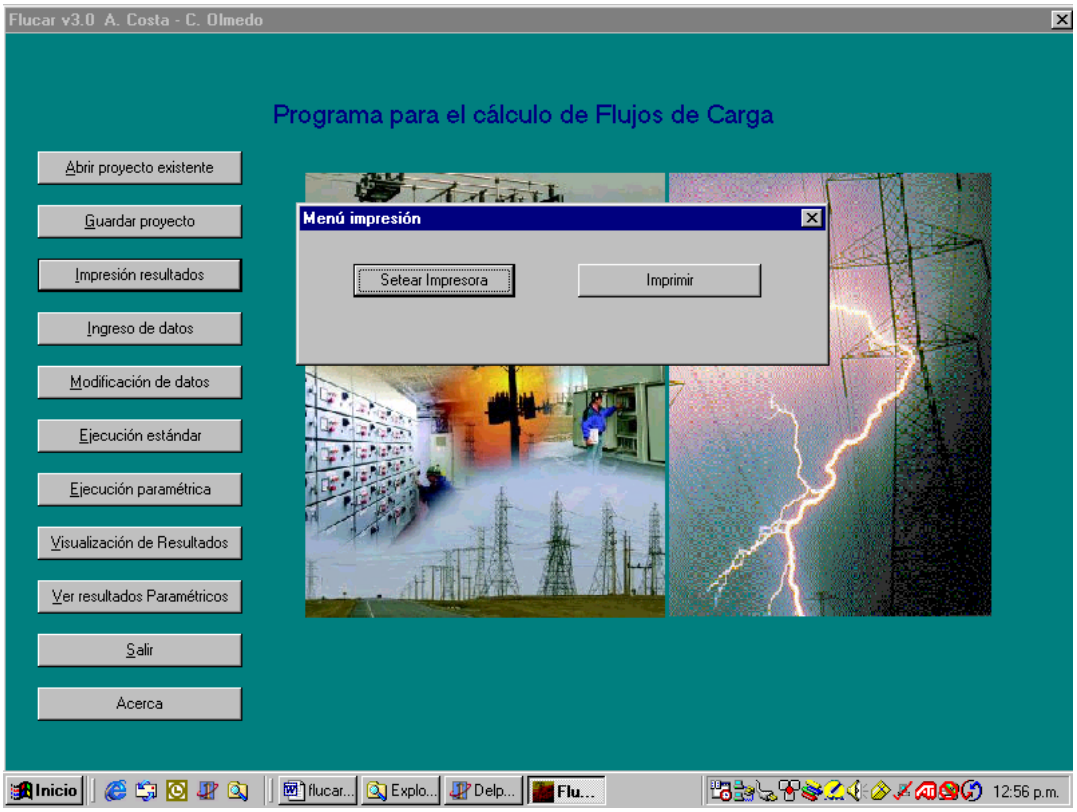
8.1.2.2. *Guardar proyecto*

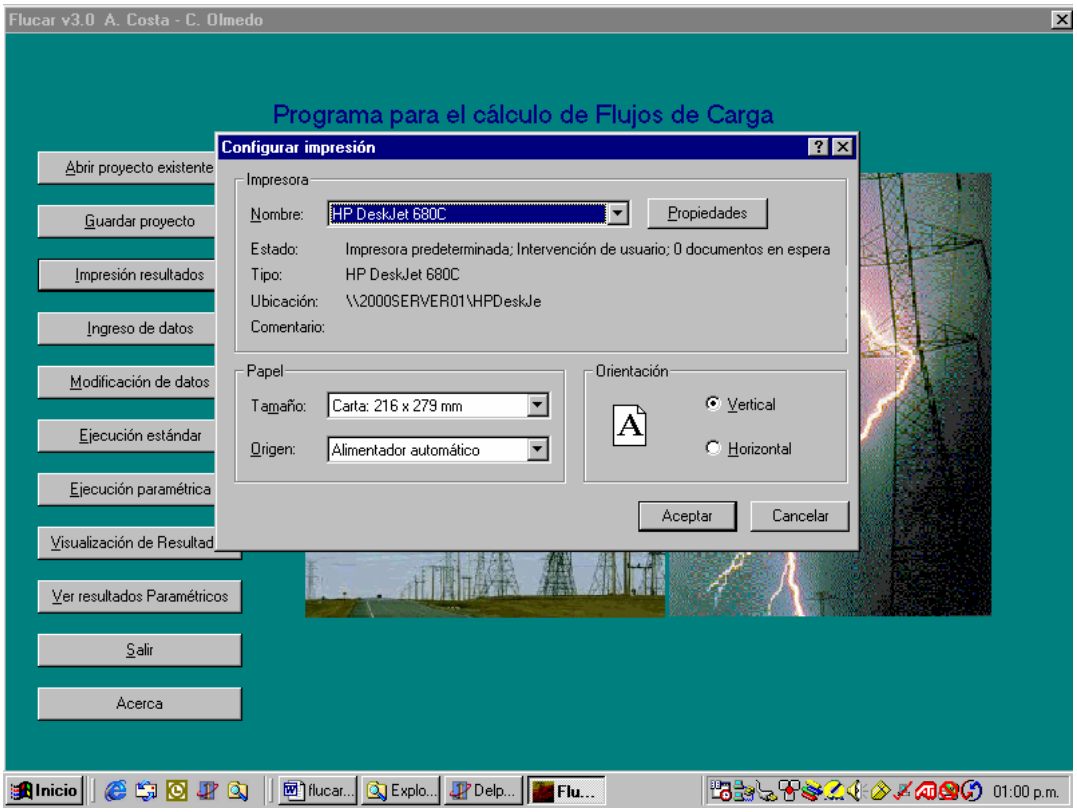
Abre la venta que permite seleccionar el nombre con que se guardará el proyecto. Luego la aplicación retorna a la pantalla principal.



8.1.2.3. *Impresión de resultados*

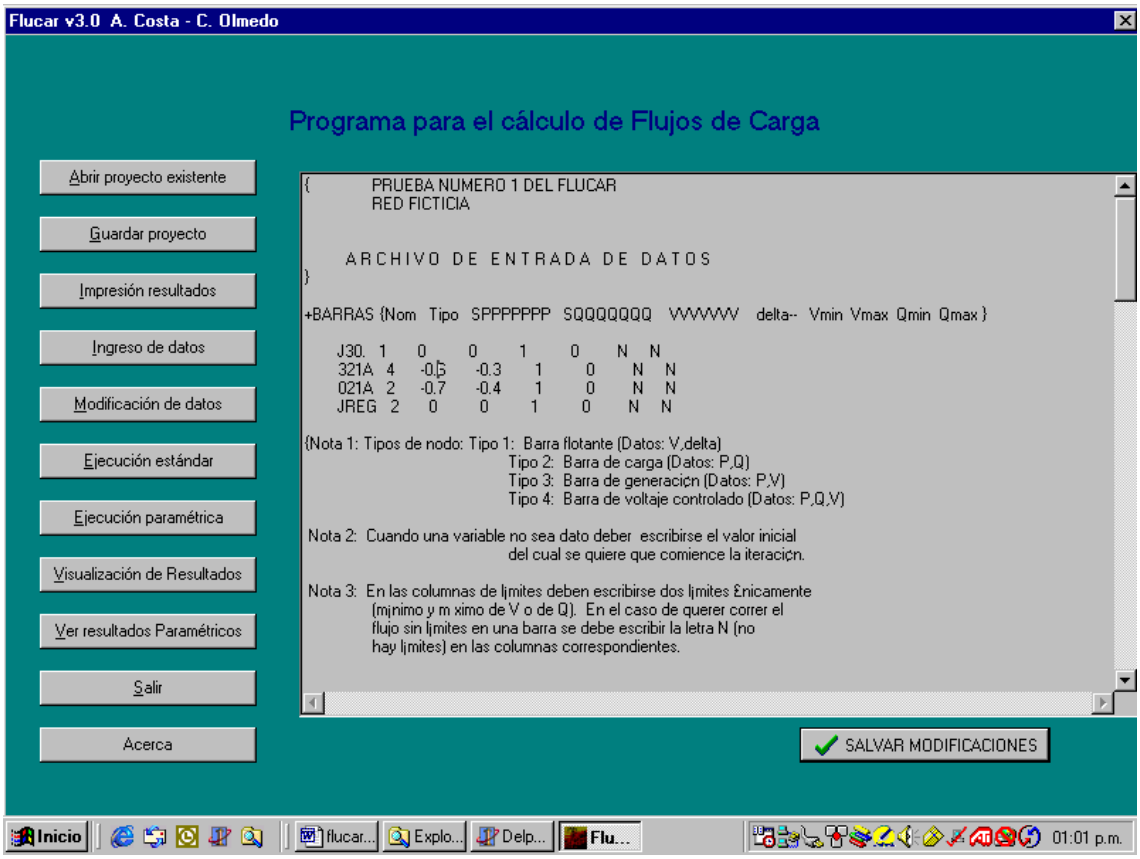
Se abre la pantalla que permite setear la impresora o directamente imprimir. El seteo de impresora es la clásica selección de impresora y características de impresión a las que estamos acostumbrados en toda aplicación Windows. Imprimir ya imprime lo deseado.


Flucar v3.0 A. Costa - C. Olmedo
Programa para el cálculo de Flujos de Carga
Abrir proyecto existente
Guardar proyecto
Impresión resultados
Ingreso de datos
Modificación de datos
Ejecución estándar
Ejecución paramétrica
Visualización de Resultados
Ver resultados Paramétricos
Salir
Acerca
Menú impresión
Setear Impresora
Imprimir
Inicio
flucar...
Explo...
Delp...
Flu...
12:56 p.m.


Flucar v3.0 A. Costa - C. Olmedo
Programa para el cálculo de Flujos de Carga
Abrir proyecto existente
Guardar proyecto
Impresión resultados
Ingreso de datos
Modificación de datos
Ejecución estándar
Ejecución paramétrica
Visualización de Resultad
Ver resultados Paramétricos
Salir
Acerca
Configurar impresión
Impresora
Nombre:
HP DeskJet 680C
Propiedades
Estado:
Impresora predeterminada; Intervención de usuario; 0 documentos en espera
Tipo:
HP DeskJet 680C
Ubicación:
\\2000SERVER01\HPDeskJe
Comentario:
Papel
Tamaño:
Carta: 216 x 279 mm
Origen:
Alimentador automático
Orientación
Vertical
Horizontal
Aceptar
Cancelar
Inicio
flucar...
Explo...
Delp...
Flu...
01:00 p.m.

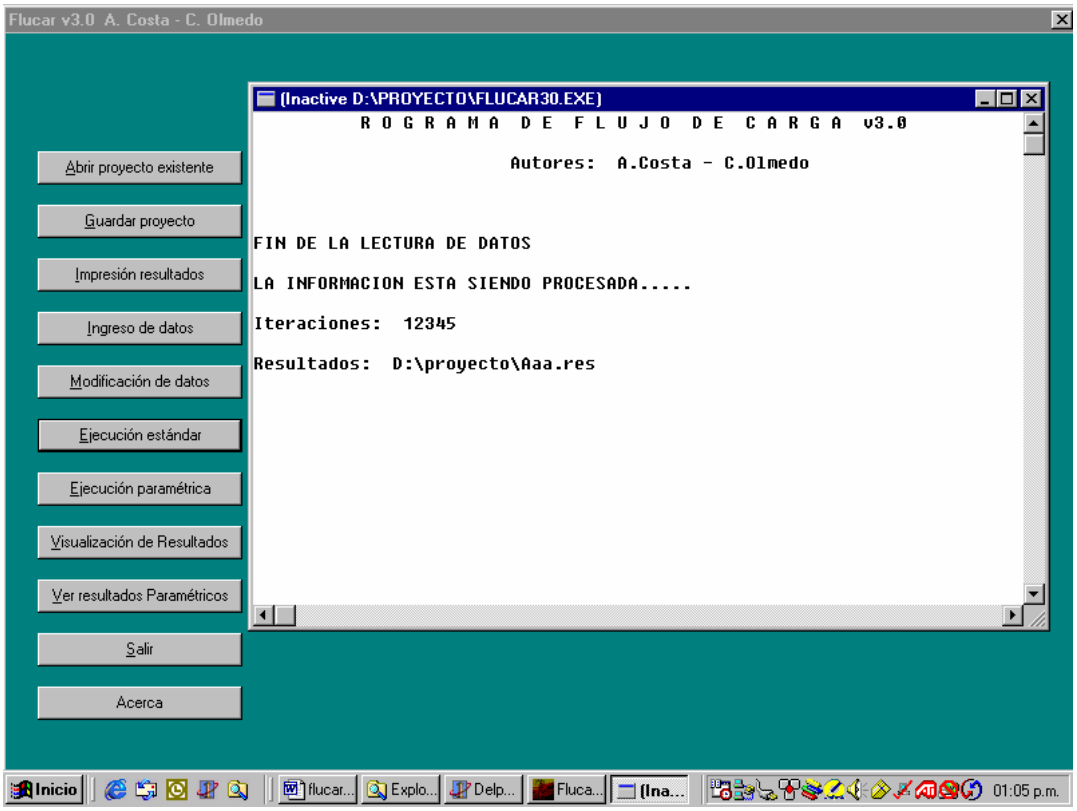
8.1.2.4. *Modificación de datos*

A diferencia del caso Abrir proyecto acá se pueden modificar valores y luego salvar las modificaciones. Es importante notar que se cuenta con siete posiciones para ingreso de datos para cada valor P, Q, V o delta. Previo a esto en los casos de P o Q se tiene un espacio para ingreso de signo. El formato de datos se puede ver en la línea superior +BARRAS. El espacio entre datos es fijo y no debe variarse.

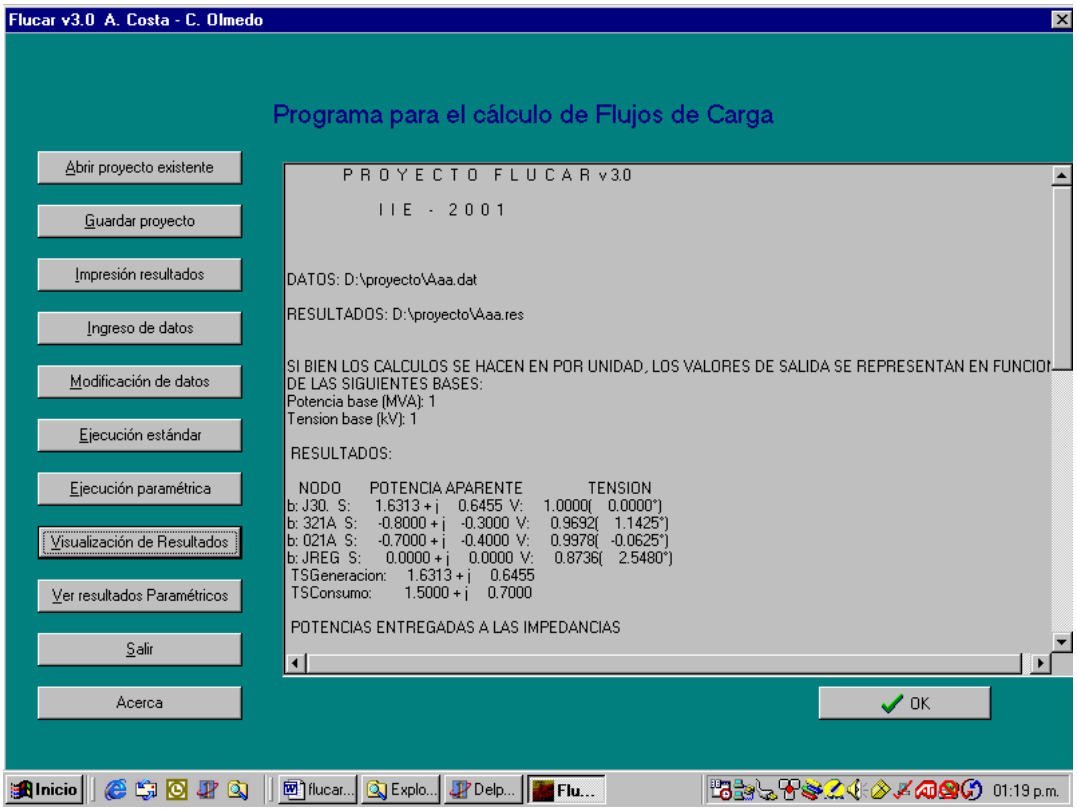


8.1.2.5. *Ejecución estándar*

Al incorporar la corrida de flujo de cargas paramétrica se cuenta con la opción de corrida de flujo de carga estándar. Este abre una ventana donde corre el FLUCAR3.0. Una vez terminado queda abierto para que el usuario se entere que su aplicación terminó de correr.

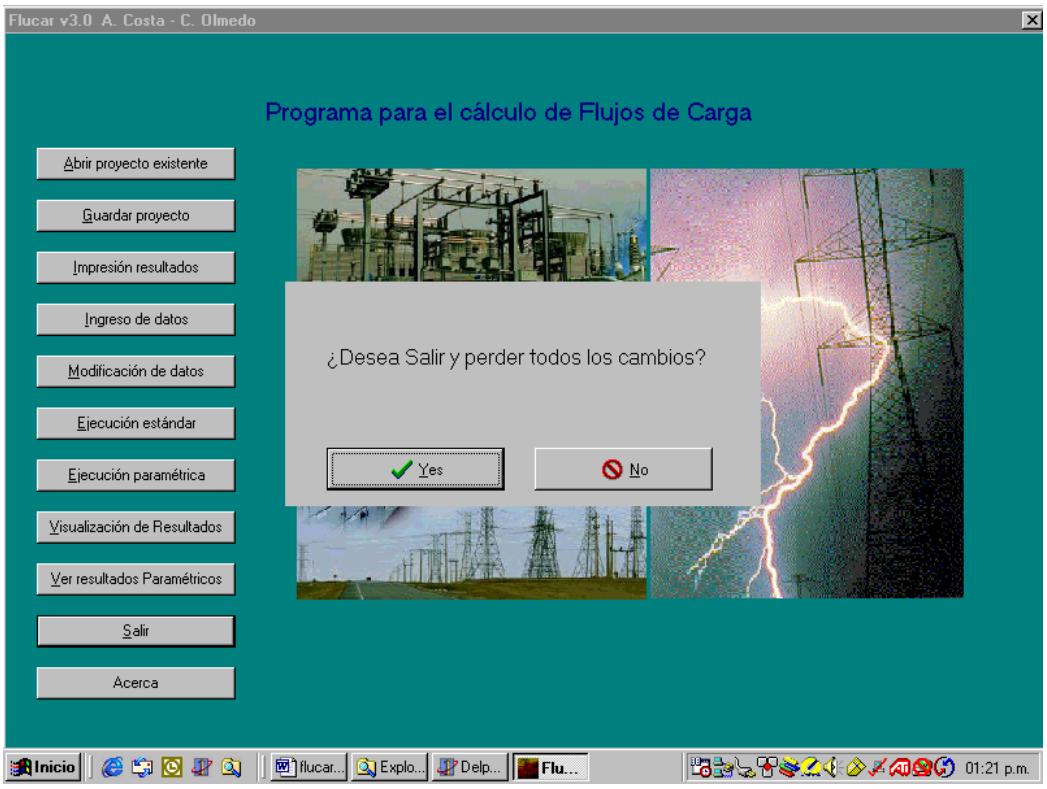

Flucar v3.0 A. Costa - C. Olmedo
(Inactive D:\PROYECTO\FLUCAR30.EXE)
ROGRAMA DE FLUJO DE CARGA v3.0
Autores: A.Costa - C.Olmedo
FIN DE LA LECTURA DE DATOS
LA INFORMACION ESTA SIENDO PROCESADA.....
Iteraciones: 12345
Resultados: D:\proyecto\Aaa.res
Abrir proyecto existente
Guardar proyecto
Impresión resultados
Ingreso de datos
Modificación de datos
Ejecución estándar
Ejecución paramétrica
Visualización de Resultados
Ver resultados Paramétricos
Salir
Acerca
Inicio
flucar...
Explo...
Delp...
Fluca...
(Ina...
01:05 p.m.

8.1.2.6. *Visualización resultado corrida estandar*

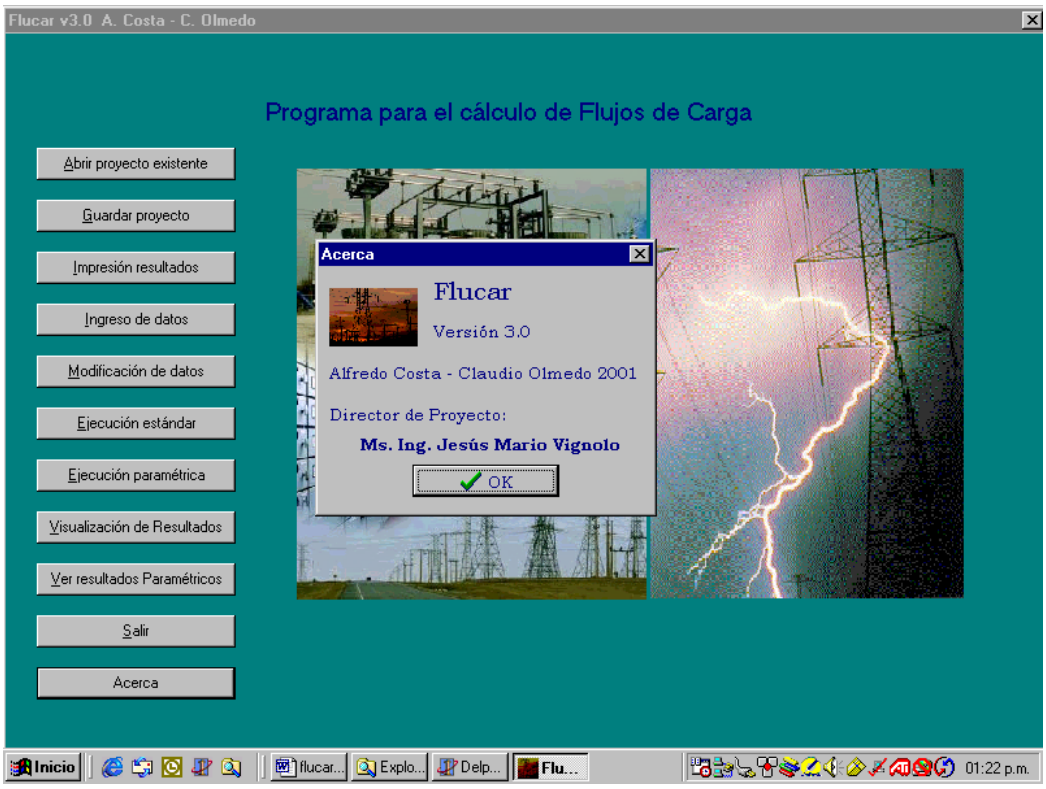


Permite observar el archivo *.res de salida del flujo de carga estándar. Se entiende por estándar el flujo de carga no paramétrico.

8.1.2.7. *Salida del programa*

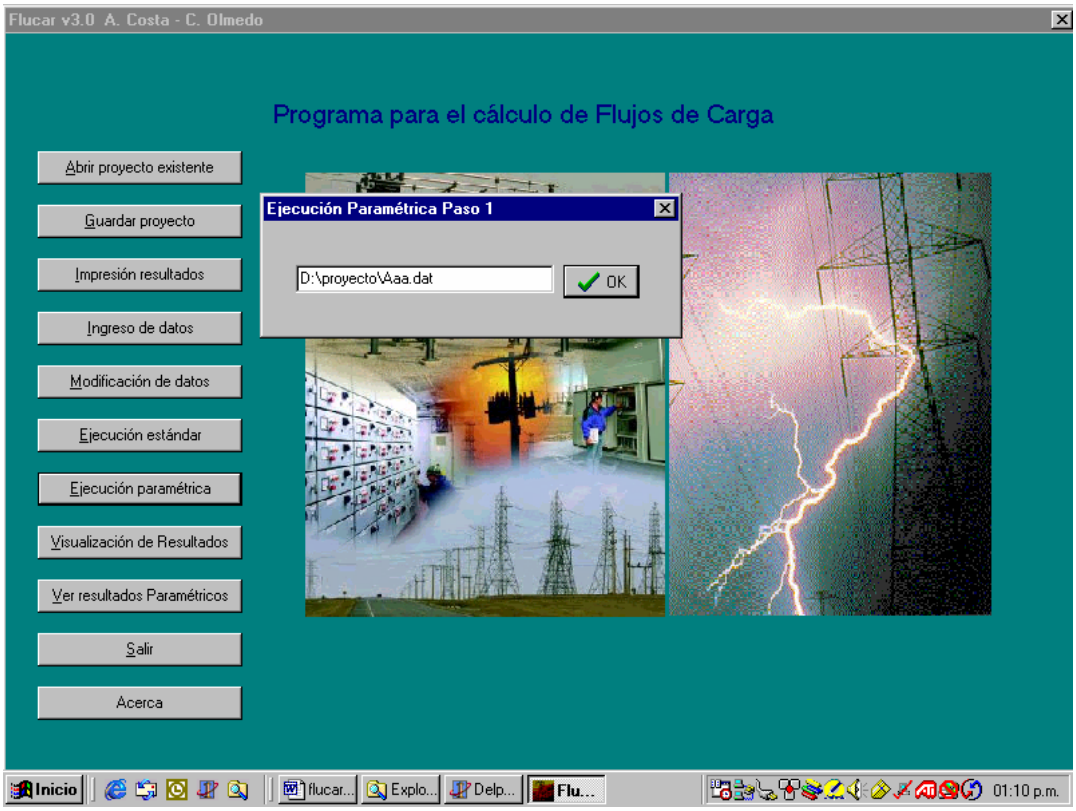


8.1.2.8. *Acerca*

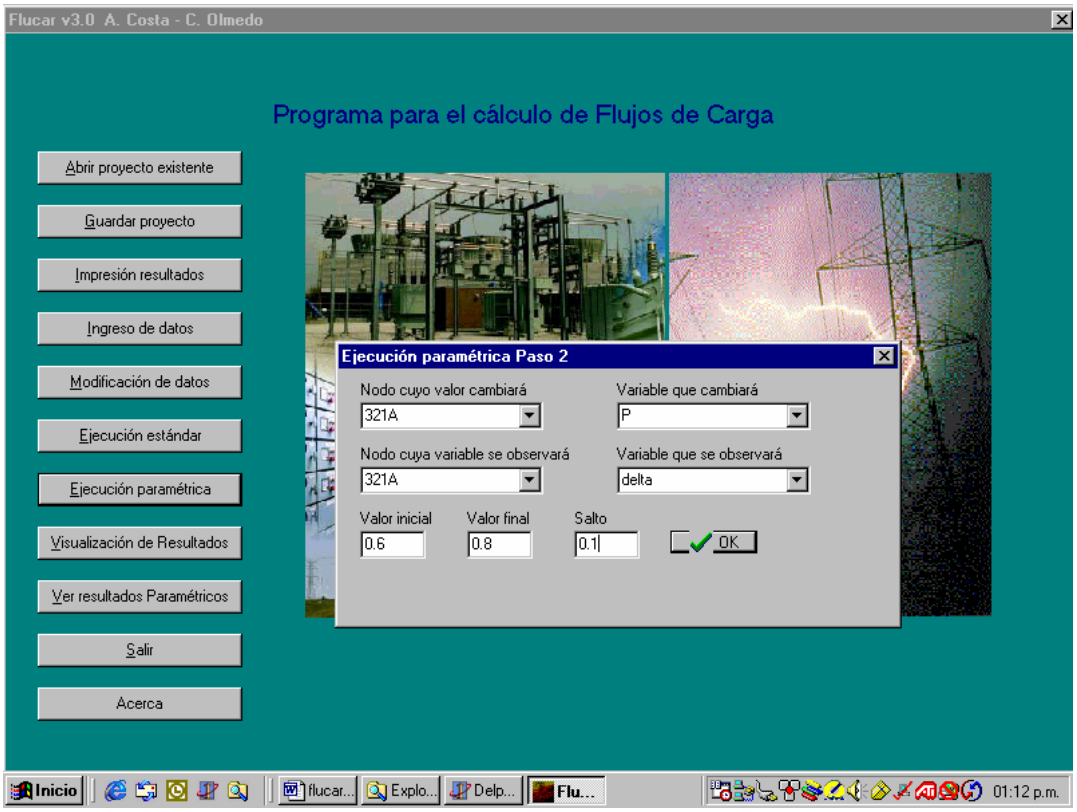

Flucar v3.0 A. Costa - C. Olmedo
Programa para el cálculo de Flujos de Carga
Abrir proyecto existente
Guardar proyecto
Impresión resultados
Ingreso de datos
Modificación de datos
Ejecución estándar
Ejecución paramétrica
Visualización de Resultados
Ver resultados Paramétricos
Salir
Acerca
Acerca
Flucar
Versión 3.0
Alfredo Costa - Claudio Olmedo 2001
Director de Proyecto:
Ms. Ing. Jesús Mario Vignolo
OK
Inicio
flucar...
Explo...
Delp...
Flu...
01:22 p.m.

8.2. Ejecución Paramétrica

Se realizó una aplicación que permite variar un parámetro determinado de la red: Potencia Activa, Potencia Reactiva,Tensión o desfasaje de un nodo y observar en forma gráfica la variación de otro parámetro cualquiera: Potencia Activa, Potencia Reactiva, Tensión o desfasaje de un nodo cualquiera. La aplicación permite ingresar el rango de variación de la variable y el salto de dicha variación, por ejemplo: Qmin a Qmáx con salto DeltaQ.



Primero se pide confirmación de archivo de origen de datos.

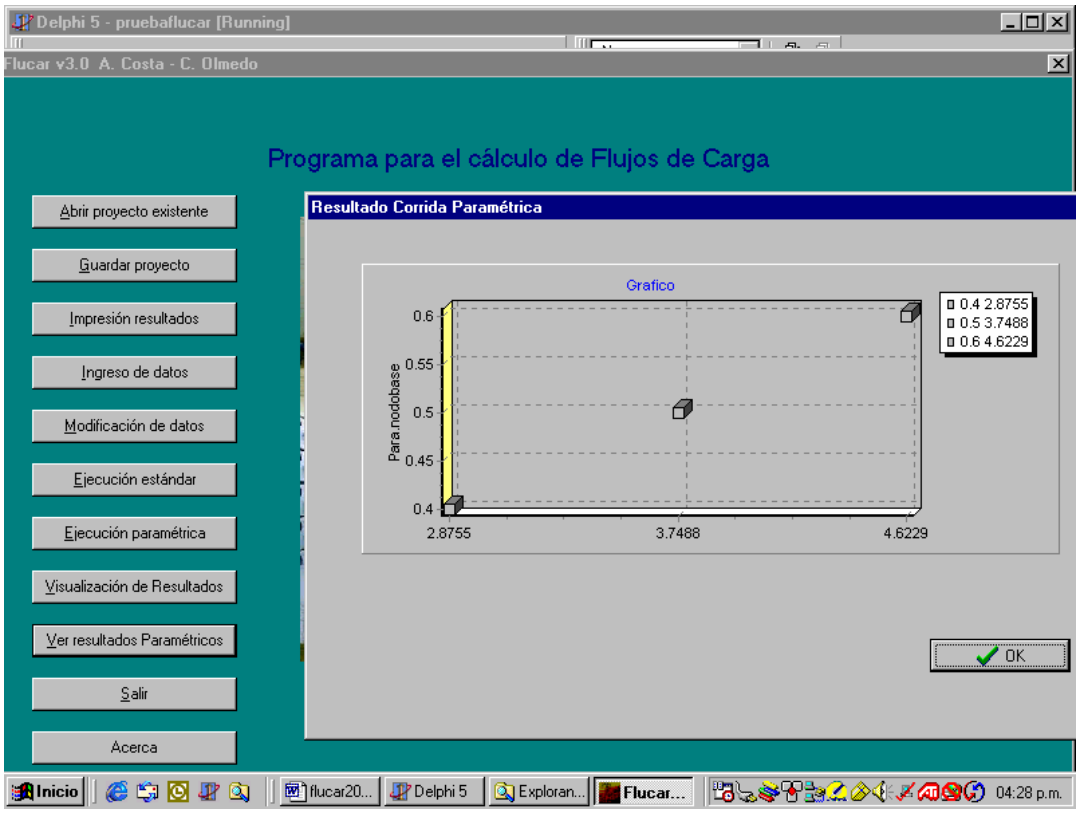


Se ingresa el nodo base cuyo valor cambiará , el parámetro que cambiará de dicho nodo (P,Q,V o delta), el nodo resultado cuyo valor se observará y la variable de dicho nodo que se desea observar (P,Q,V o delta). Se debe ingresar el rango de valores entre los que cambiará. En este caso arriba mostrado P variará entre 0.6 y 0.8 con salto de 0.1.

Una vez que se termina la ejecución paramétrica se vuelve a la pantalla principal.

8.2.1. *Visualización de resultados corrida paramétrica*

Se abre una pantalla donde se presenta una gráfica. Al hacer clic con el boton izquierdo sobre el gráfico este se actualiza y presenta los resultados de las corridas realizadas.



El grafico presenta valores en un cuadro en un lateral y las variaciones de los parametros en función de los resultados obtenido

ANEXO I

En el presente anexo se detalla una corrida con el Flucar 30 con mas de un regulador, para así mostrar el correcto funcionamiento del mismo.

Entrada de datos:

```
{      SISTEMA ELECTRICO URUGUAYO



          A R C H I V O   D E   E N T R A D A   D E   D A T O S
}

+BARRAS {Nom    Tipo  P        Q         V   delta   Vmin  Vmax  Qmin
Qmax }
      S_G_GEN  3    10.5     0         1      0     N     N
      S_G_500  1    0        0         1      0     N     N
      S_J_500  2    0        0         1      0     N     N
      S_G_150  2    0        0         1      0     N     N
      MERC150  4    -0.172   -0.047    1      0     N     N
      NPAL150  2    -0.151   -0.042    1      0     N     N
      CONC150  2    -0.080   -0.028    1      0     N     N
      PALM500  2    0        0         1      0     N     N
      S_J_150  2    0        0         1      0     N     N
      PALM150  2    0        0         1      0     N     N
      ARTI150  2    -0.181   -0.027    1      0     N     N
```

```
      TRIN150  2    -0.084  -0.028   1      0     N     N
      DURA150  2    -0.285  -0.096   1      0     N     N
      RIVE150  4    -0.384  -0.142   1      0     N     N
      MELO150  2    -0.357  -0.089   1      0     N     N
      MONA500  2    0       0        1      0     N     N
      MONB500  2    0       0        1      0     N     N
      MONB150  2    -1.338  -0.47    1      0     N     N
      MONA150  2    -1.781  -0.467   1      0     N     N
      MONI500  2    0       0        1      0     N     N
      MONI150  2    -3.284  -0.924   1      0     N     N
      MONC150  2    -0.783  -0.248   1      0     N     N
      MONL150  2    0       0        1      0     N     N
      ROSA150  2    -0.4    -0.089   1      0     N     N
      MONE150  2    -2.277  -0.766   1      0     N     N
      SCAR500  2    0       0        1      0     N     N
      SCAR150  2    -2.115  -0.601   1      0     N     N
      MONEGEN  3    3.1875  0        1      0     N  N
      PALMGEN  3    3.33    0        1      0     N  N
      MONLGEN  3    2.855   0        1      0     N  N
      TERRGEN  3    0.8     0        1      0     N  N
      BAYGGEN  3    1.08    0        1      0     N  N
      TERR150  2    0       0        1      0     N  N
      BAYG150  2    0       0        1      0     N  N
      SJAVREG  2    0       0        1      0     N     N
      RIVEREG  2    0       0        1      0     N     N
{Nota 1: Tipos de nodo: Tipo 1:  Barra flotante (Datos: V,delta)
                  Tipo 2:  Barra de carga (Datos: P,Q)
                  Tipo 3:  Barra de generaci¢n (Datos: P,V)
                  Tipo 4:  Barra de voltaje controlado (Datos: P,Q,V)

 Nota 2:  Cuando una variable no sea dato deber  escribirse el valor
inicial
                  del cual se quiere que comience la iteraci¢n.

 Nota 3:  En las columnas de l¡mites deben escribirse dos l¡mites
£nicamente
      (m¡nimo y m ximo de V o de Q).  En el caso de querer correr el
      flujo sin l¡mites en una barra se debe escribir la letra N (no
      hay l¡mites) en las columnas correspondientes.

 Nota 4:  Las barras de tipo 4 se comportan como barras de tipo 2 con
la
      £nica diferencia que se hace un ajuste del regulador para
ajustar la
      tensi¢n.  Si una barra con regulador se corre como tipo 2 no se
har
      ajuste de la tensi¢n tom ndose el regulador como un trafo de
relaci¢n
      constante e igual al valor de n inicial
}


+IMPEDANCIAS
{
                        Z
            b1---------/////////---------b2

}
```

```
            { Nombre    bs          bll           Z             Imax

                    {Z1      S_G_GEN     S_G_500          0+j0.00001
      0
                    Z2       PALMGEN     PALM500          0+j0.00001
      0
                    Z3       MONLGEN     MONL150          0+j0.00001
      0}
                    Z4       MONEGEN     MONE150          0+j0.00001
      0
                {Z5      BAYGGEN     BAYG150          0+j0.00001  0
                    Z6       TERRGEN     TERR150          0+j0.00001
      0}


+CUADRIPOLOSPI
 {                            Z12
                  b1 -------/////////------- b2
                       -                   -
                       /                   /
                    Y13  /                   /  Y23
                       /                   /
                       -                   -
                       N                   N
 }


          { b1     b2      b3       Y13            Z12            Y23
Imax}
 cua001  S_G_500  S_J_500   N        0+j0.7644    0.00171+j0.01606
0+j0.7644    0
 cua002  S_J_500  PALM500   N        0+j0.42105   0.00094+j0.00884
0+j0.42105   0
 cua003  S_J_500  PALM500   N        0+j0.40425   0.00091+j0.00857
0+j0.40425   0
 cua004  S_J_150  SJAVREG   N        0+j0         0.04803+j0.16260     0+j0
0
 cua005  MERC150  NPAL150   N        0+j0         0.14874+j0.24601     0+j0
0
 cua006  NPAL150  CONC150   N        0+j0         0.14166+j0.23431     0+j0
0
 cua007  TERR150  RIVEREG   N        0+j0         0.14728+j0.4439      0+j0
0
 cua008  TERR150  MELO150   N        0+j0         0.21675+j0.56516     0+j0
0
 cua009  PALM150  TRIN150   N        0+j0         0.04986+j0.1684      0+j0
0
 cua010  TERR150  DURA150   N        0+j0         0.03824+j0.14718     0+j0
0
 cua011  TRIN150  MONB150   N        0+j0         0.039145+j0.1322     0+j0
0
 cua012  MONB150  MONC150   N        0+j0         0.00181+j0.01471     0+j0
0
 cua013  MONB500  MONA500   N        0+j0         0.0012+j0.0011       0+j0
0
 cua014  MONA500  MONI500   N        0+j0         0.0002+j0.00208      0+j0
0
 cua015  MONI500  SCAR500   N        0+j0.66      0.0013+j0.01372      0+j0.66
0
 cua016  MONI150  MONE150   N        0+j0         0.00146+j0.00341     0+j0
0
```

```
 cua017  MONC150  MONE150   N        0+j0         0.00089+j0.003615    0+j0
0
 cua018  MONA150  MONI150   N        0+j0         0.00219+j0.02317     0+j0
0
 cua019  MONC150  ROSA150   N        0+j0         0.19698+j0.45432     0+j0
0
 cua020  PALM500  MONA500   N        0+j1.20698   0.00271+j0.02534
0+j1.20698   0
 cua021  PALM150  MONA150   N        0+j0         0.00134+j0.51367      0+j0
0
 cua022  MONA150  MONL150   N        0+j0         0.00131+j0.053955    0+j0
0
 cua023  MONB150  MONL150   N        0+j0         0.001205+j0.01998    0+j0
0
 cua024  PALM500  MONB500   N        0+j1.156575  0.00259+j0.02429    0+j0
0
 cua025  PALM150  MONA150   N        0+j0         0.00134+j0.51367     0+j0
0
 cua026  S_G_150  ARTI150   N        0+j0         0.12329+j0.37156     0+j0
0
 cua027  PALM150  BAYG150   N        0+j0         0.04794+j0.16192     0+j0
0
 cua028  BAYG150  TERR150   N        0+j0         0.02325+j0.07358     0+j0
0
 cua029  MONI150  SCAR150   N        0+j0         0.01444+j0.15239     0+j0
0
 cua030  SCAR150  MELO150   N        0+j0         0.11259+j0.28667     0+j0
0



+TRAFOS
{
                                          1:n
                                       -  -   -  -        Zcc
                  Nodo1  ----- -      -      - ----/////---- Nodo2
                                       -  -   -  -
 }

{ Nombre      b1       b2         n     Zcc                  Imax }
  traf001     S_J_500  S_J_150    1     0.00301+j0.10527     0
  traf002     S_G_500  S_G_150    1     0.00301+j0.10527     0
  traf003     PALM500  PALM150    1.073 0.00110+j0.04000     0
  traf004     MONB500  MONB150    1.053 0.00062+j0.02927     0
  traf005     MONA500  MONA150    1.053 0.00059+j0.02162     0
  traf006     MONI500  MONI150    1     0.00025+j0.04234     0
  traf007     MONI500  MONI150    1     0.00025+j0.04234     0
  traf008     SCAR500  SCAR150    1     0.00025+j0.04234     0
  traf009     S_G_GEN  S_G_500    1.05  0.00065+j0.03160     0
  traf010     PALMGEN  PALM500    1     0.00260+j0.10760     0
  traf011     MONLGEN  MONL150    1     0.0055 +j0.126       0
{ traf012     MONEGEN  MONE150    1.1    0.002 +j0.126       0}
 traf013     BAYGGEN  BAYG150    1.1   0.0026 +j0.2852      0
  traf014     TERRGEN  TERR150    1.1   0.0039 +j0.1589      0

{Nota:      La corriente m xima admisible por una impedancia,
cuadripolo,
      o transformador debe anotarse debajo de Imax.  En el caso de
      no querer establecer restricci¢n sobre la corriente escribir
      0. Para los trafos Imax es la del secundario
}
+REGULADORES
```

```
{Nota:  n se refiere al valor de la relaci¢n de transformaci¢n de
partida.
      nmin es el valor m¡nimo de n admisible.
      nmax es el valor m ximo de n admisible.
      deltan es el paso porcentual: n_nuevo= n_viejo*(1 +/- deltan).
}

{Nombre   b1    b2      n    nmin       nmax    deltan     Zcc
Imax}
traf002   MERC150 SJAVREG    1  0.1      1.2     0.005     0+j0.03
0
traf006   RIVE150 RIVEREG    1  0.1        1.2     0.005     0+j0.03
0


+TOLERANCIA     0.1

+NITS         100

+FIN.
```

Obteniendose el siguiente resultado:

```
             P  R  O  Y  E  C  T  O     F  L  U  C  A  R  v 3.0

                      I  I  E     -     2  0  0  1


DATOS: sjavyriv.dat

RESULTADOS: sjavyriv.res


SI BIEN LOS CALCULOS SE HACEN EN POR UNIDAD, LOS VALORES DE SALIDA SE
REPRESENTAN EN FUNCION
DE LAS SIGUIENTES BASES:
Potencia base (MVA): 1
Tension base (kV): 1

 RESULTADOS:

   NODO       POTENCIA APARENTE                 TENSION
b: S_G_GEN  S:     10.5000 + j     3.1267  V:     1.0000(   18.3043°)
b: S_G_500  S:     -7.6836 + j    -1.9323  V:     1.0000(    0.0000°)
b: S_J_500  S:      0.0000 + j     0.0000  V:     1.0211(   -2.4542°)
b: S_G_150  S:      0.0000 + j     0.0000  V:     0.9950(   -1.1175°)
b: MERC150  S:     -0.1720 + j    -0.0470  V:     1.0130(   -9.1935°)
b: NPAL150  S:     -0.1510 + j    -0.0420  V:     0.9434(  -12.1343°)
b: CONC150  S:     -0.0800 + j    -0.0280  V:     0.9239(  -13.1058°)
b: PALM500  S:      0.0000 + j     0.0000  V:     1.0209(   -2.9667°)
b: S_J_150  S:      0.0000 + j     0.0000  V:     1.0042(   -4.9014°)
b: PALM150  S:      0.0000 + j     0.0000  V:     1.0706(   -2.9773°)
b: ARTI150  S:     -0.1810 + j    -0.0270  V:     0.9590(   -4.9589°)
b: TRIN150  S:     -0.0840 + j    -0.0280  V:     1.0320(   -5.8114°)
b: DURA150  S:     -0.2850 + j    -0.0960  V:     1.0246(    0.5702°)
b: RIVE150  S:     -0.3840 + j    -0.1420  V:     1.0062(   -6.3753°)
b: MELO150  S:     -0.3570 + j    -0.0890  V:     0.9379(  -11.0395°)
```

```
b: MONA500  S:      0.0000 + j     0.0000  V:     0.9960(   -6.7150°)
b: MONB500  S:      0.0000 + j     0.0000  V:     0.9974(   -6.5330°)
b: MONB150  S:     -1.3380 + j    -0.4700  V:     1.0101(   -7.6188°)
b: MONA150  S:     -1.7810 + j    -0.4670  V:     1.0176(   -8.1848°)
b: MONI500  S:      0.0000 + j     0.0000  V:     0.9965(   -7.1210°)
b: MONI150  S:     -3.2840 + j    -0.9240  V:     0.9970(   -9.1411°)
b: MONC150  S:     -0.7830 + j    -0.2480  V:     1.0003(   -8.8649°)
b: MONL150  S:      0.0000 + j     0.0000  V:     1.0034(   -5.4352°)
b: ROSA150  S:     -0.4000 + j    -0.0890  V:     0.8389(  -20.1487°)
b: MONE150  S:     -2.2770 + j    -0.7660  V:     1.0000(   -8.9306°)
b: SCAR500  S:      0.0000 + j     0.0000  V:     0.9933(   -8.4426°)
b: SCAR150  S:     -2.1150 + j    -0.6010  V:     0.9655(  -12.6577°)
b: MONEGEN  S:      3.1875 + j     1.1209  V:     1.0000(   -8.9288°)
b: PALMGEN  S:      3.3300 + j     0.3262  V:     1.0000(   17.5299°)
b: MONLGEN  S:      2.8550 + j     0.3712  V:     1.0000(   15.4478°)
b: TERRGEN  S:      0.8000 + j     0.3715  V:     1.0000(    8.8576°)
b: BAYGGEN  S:      1.0800 + j     0.2579  V:     1.0000(   18.7995°)
b: TERR150  S:      0.0000 + j     0.0000  V:     1.0497(    2.6095°)
b: BAYG150  S:      0.0000 + j     0.0000  V:     1.0678(    3.6304°)
b: SJAVREG  S:      0.0000 + j     0.0000  V:     0.9635(   -8.5514°)
b: RIVEREG  S:      0.0000 + j     0.0000  V:     0.9087(   -5.6657°)
 TSGeneracion:    21.7525 + j     5.5744
 TSConsumo:       21.3556 + j     5.9963

 POTENCIAS ENTREGADAS A LAS IMPEDANCIAS
MONEGEN->Z4     3.1875 + j     1.1209
MONE150->Z4    -3.1875 + j    -1.1207

Perdidas Joule en las impedancias:  0.0000000

 POTENCIAS ENTREGADAS A LOS CUADRIPOLOS
S_G_500->cua001     2.5600 + j    -2.2902
S_J_500->cua001    -2.5448 + j     0.8715

S_J_500->cua002     1.0457 + j    -0.5233
PALM500->cua002    -1.0447 + j    -0.3451

S_J_500->cua003     1.0787 + j    -0.5083
PALM500->cua003    -1.0776 + j    -0.3248

S_J_150->cua004     0.4201 + j     0.1397
SJAVREG->cua004    -0.4107 + j    -0.1081

MERC150->cua005     0.2751 + j     0.1254
NPAL150->cua005    -0.2619 + j    -0.1035

NPAL150->cua006     0.0812 + j     0.0300
CONC150->cua006    -0.0800 + j    -0.0280

TERR150->cua007     0.3850 + j     0.2281
RIVEREG->cua007    -0.3582 + j    -0.1474

TERR150->cua008     0.4443 + j     0.0866
MELO150->cua008    -0.4040 + j     0.0185

PALM150->cua009     0.3672 + j     0.1445
TRIN150->cua009    -0.3604 + j    -0.1216

TERR150->cua010     0.2883 + j     0.1087
DURA150->cua010    -0.2850 + j    -0.0960
```

```
TRIN150->cua011     0.2764 + j     0.0936
MONB150->cua011    -0.2733 + j    -0.0830

MONB150->cua012     1.5545 + j     0.4942
MONC150->cua012    -1.5498 + j    -0.4559

MONB500->cua013     1.9539 + j    -0.8397
MONA500->cua013    -1.9484 + j     0.8447

MONA500->cua014     3.3292 + j    -0.5410
MONI500->cua014    -3.3269 + j     0.5649

MONI500->cua015     1.6727 + j    -0.5634
SCAR500->cua015    -1.6690 + j    -0.7044

MONI150->cua016    -1.2278 + j    -0.3592
MONE150->cua016     1.2302 + j     0.3648

MONC150->cua017     0.3198 + j     0.0105
MONE150->cua017    -0.3197 + j    -0.0101

MONA150->cua018     0.8096 + j     0.8342
MONI150->cua018    -0.8067 + j    -0.8040

MONC150->cua019     0.4470 + j     0.1974
ROSA150->cua019    -0.4000 + j    -0.0890

PALM500->cua020     2.7085 + j    -0.4607
MONA500->cua020    -2.6878 + j    -1.8007

PALM150->cua021     0.1928 + j     0.1188
MONA150->cua021    -0.1927 + j    -0.0958

MONA150->cua022    -0.9003 + j     0.3098
MONL150->cua022     0.9014 + j    -0.2626

MONB150->cua023    -1.9034 + j     0.4862
MONL150->cua023     1.9080 + j    -0.4106

PALM500->cua024     2.6906 + j    -0.4265
MONB500->cua024    -2.6711 + j    -0.5960

PALM150->cua025     0.1928 + j     0.1188
MONA150->cua025    -0.1927 + j    -0.0958

S_G_150->cua026     0.1855 + j     0.0405
ARTI150->cua026    -0.1810 + j    -0.0270

PALM150->cua027    -0.7291 + j     0.2813
BAYG150->cua027     0.7546 + j    -0.1950

BAYG150->cua028     0.3227 + j     0.1624
TERR150->cua028    -0.3201 + j    -0.1540

MONI150->cua029     0.4044 + j     0.1794
SCAR150->cua029    -0.4016 + j    -0.1494

SCAR150->cua030    -0.0452 + j     0.1120
MELO150->cua030     0.0470 + j    -0.1075
```

```
Perdidas Joule en los cuadripolos:  0.2730826

 POTENCIAS ENTREGADAS A LOS TRANSFORMADORES
S_J_500->traf001     0.4205 + j     0.1602
S_J_150->traf001    -0.4199 + j    -0.1397

S_G_500->traf002     0.1856 + j     0.0444
S_G_150->traf002    -0.1855 + j    -0.0405

PALM500->traf003     0.0241 + j     0.6787
PALM150->traf003    -0.0237 + j    -0.6633

MONB500->traf004     0.7173 + j     1.4357
MONB150->traf004    -0.7158 + j    -1.3674

MONA500->traf005     1.3070 + j     1.4971
MONA150->traf005    -1.3049 + j    -1.4194

MONI500->traf006     0.8271 + j    -0.0007
MONI150->traf006    -0.8269 + j     0.0299

MONI500->traf007     0.8271 + j    -0.0007
MONI150->traf007    -0.8269 + j     0.0299

SCAR500->traf008     1.6690 + j     0.7044
SCAR150->traf008    -1.6682 + j    -0.5636

S_G_GEN->traf009    10.5000 + j     3.1267
S_G_500->traf009   -10.4292 + j     0.3135

PALMGEN->traf010     3.3300 + j     0.3262
PALM500->traf010    -3.3009 + j     0.8784

MONLGEN->traf011     2.8550 + j     0.3712
MONL150->traf011    -2.8094 + j     0.6732

BAYGGEN->traf013     1.0800 + j     0.2579
BAYG150->traf013    -1.0774 + j     0.0327

TERRGEN->traf014     0.8000 + j     0.3715
TERR150->traf014    -0.7975 + j    -0.2693

Perdidas Joule en los transformadores:  0.1564759

 REGULADORES
Relación de transformación del regulador traf002:   0.960
MERC150->traf002    -0.3646 + j     1.6751
SJAVREG->traf002     0.3646 + j    -1.5892

Relación de transformación del regulador traf006:   0.904
RIVE150->traf006    -0.3775 + j     3.2723
RIVEREG->traf006     0.3775 + j    -2.9508

Perdidas Joule en los reguladores:  0.0000000

Perdidas Joule totales en las líneas:  0.4295585

VERIFICACION DE LIMITES:
```

```
No hubo violaciones de l¡mites en las barras.

No hubo violaciones de l¡mites en las impedancias.

No hubo violaciones de l¡mites en los cuadripolos.

No hubo violaciones de l¡mites en los transformadores.

No hubo violaciones de l¡mites en los reguladores.

NITs: 3 converge: TRUE Tiempo:   8.9000s
```

ANEXO II

1. RESOLUCION DEL FLUJO DE CARGA MEDIANTE SOFTWARE

1.1. Introducción

El lenguaje utilizado para la programación del flujo de carga fue Turbo Pascal 7.0. Para la realización del software se tuvo en consideración la existencia de un software previo propiedad del IIE utilizado en las versiones 1 y 2 del FLUCAR. Este debió ser estudiado y en algunos casos modificado agregando rutinas nuevas que aplicamos la versión 3.

Utilizamos el álgebra de complejos implementada en la unidad ALGEBRAC, el trabajo con ecuaciones implementado en la unidad ECUACS, el trabajo con sistemas de ecuaciones implementado en la unidad USISTEMA y el álgebra de vectores y matrices implementado en la unidad MATCPX.

Muchas de éstas debieron ser modificadas para poder implementar lo requerido generando pues las unidades SERVICE2, FUN4, etc.

1.2. Reordenamiento de las unidades Pascal

Se cambió el nombre de las unidades utilizadas por la versión anterior del programa de Flujo para preservar los originales. Las unidades que cambiaron de nombre son:

- MATADM por MATADM1
- FUN1 por FUN4
- SERVICE1 por SERVIC2
- NR1 por NR2
- SALIDA por SALIDA3

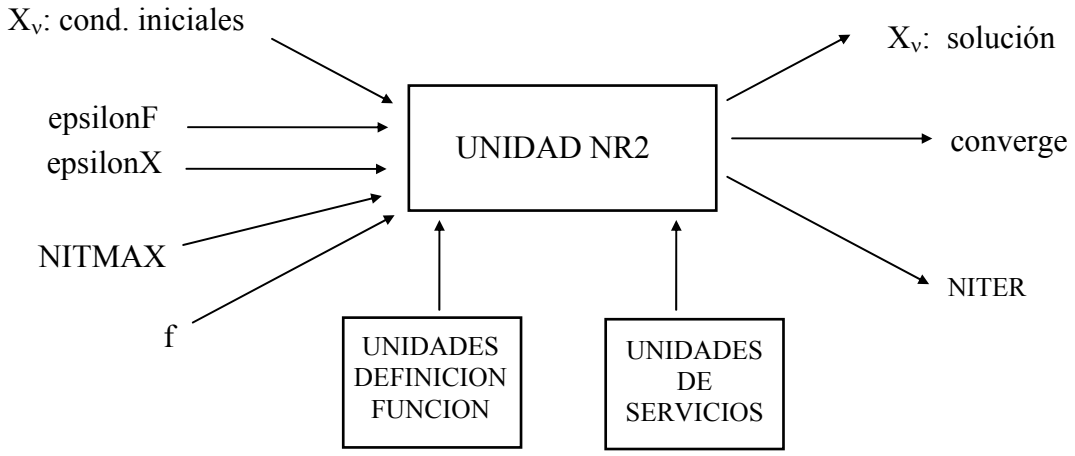
1.3. Implementación del método de Newton-Raphson

Dado que puede ocurrir que existan o no reguladores, debimos implementar el método de Newton-Raphson en la unidad NR2 siguiendo los conceptos teóricos planteados en el punto **2.** En el caso que existan reguladores se deben incluír según lo descrito en el punto **4.** del presente documento**.** NR2 puede utilizarse para aplicar el método de Newton-Raphson a cualquier función contenga o no reguladores.

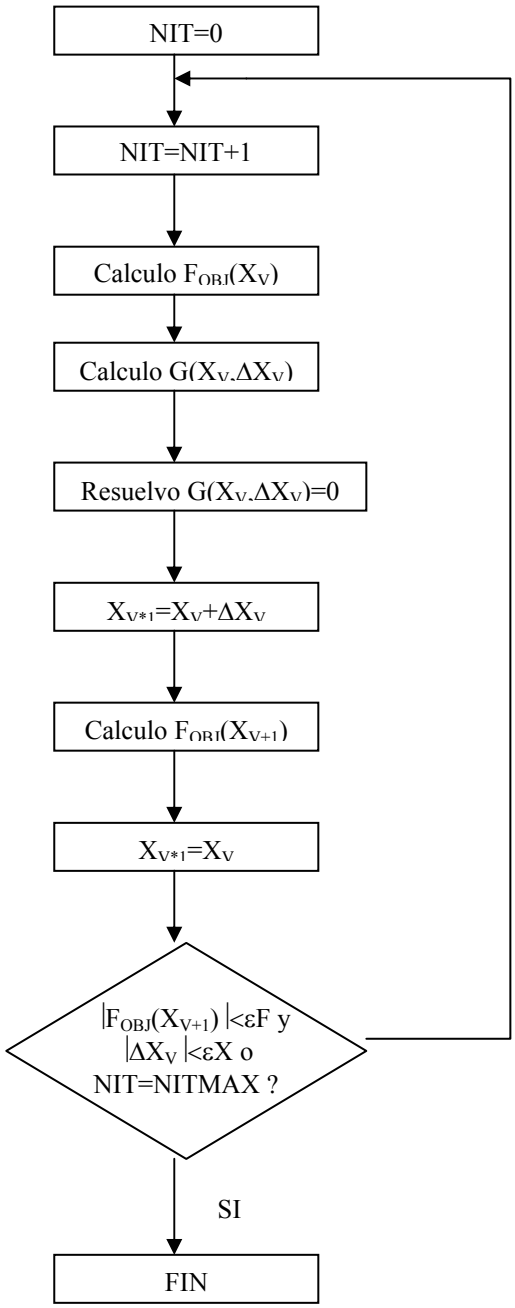
Se implementó un procedimiento cuyas variables de entrada son el vector $\mathbf{X_v}$ donde se ingresan las condiciones iniciales, **epsilonF**: el ε para ΔF, **epsilon X**: el epsilon para ΔX, **f**: factor de reducción del paso y **NITMAX**: número máximo de iteraciones admitidas. Las variables de salida son: $\mathbf{X_v}$ donde se devuelve el vector solución, la variable **converge**: es TRUE si se cumple la condición de parada con los epsilon y **NITER**: devuelve el número total de iteraciones que se ejecutaron.

La función objetivo F y las derivadas parciales para el cálculo del jacobiano deben programarse en cada caso particular en una unidad independiente que NR debe poder utilizar. En el caso particular del problema de flujo de carga dicha unidad la designamos FUN4.

El esquema básico de la unidad NR2 se representa a continuación:



El diagrama de bloques del método de Newton-Raphon implementado es el siguiente:

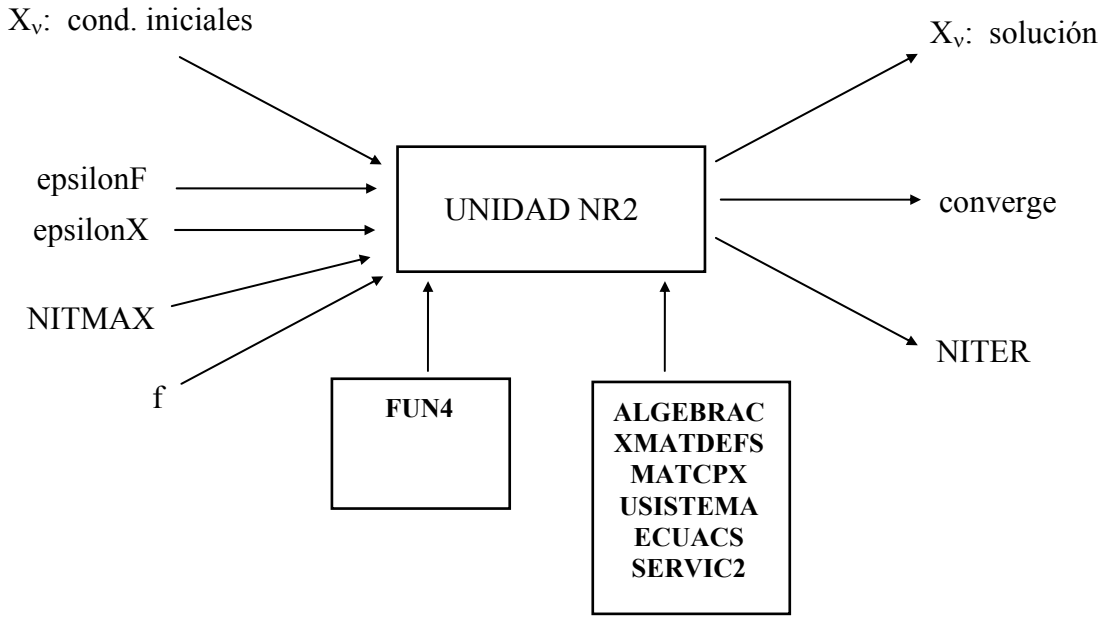


NOTA: F_{OBJ} es la función ΔF introducida en **2.** y

$$G(X_v,\Delta X_v) = [\Delta X_v] - \boxed{J_f(X_v)}\,[\Delta X_v]$$

Para la programación de esta unidad utilizamos algunas unidades de la biblioteca de unidades del IIE: ALGEBRAC, XMATDEFS, MATCPX,USISTEMA y ECUACS.

Por otra parte modificamos dos unidades más de uso de NR2: FUN4, donde como ya dijimos implementamos la función particular del problema de Flujo de Carga y sus derivadas parciales y SERVIC2. El esquema básico de la unidad resulta entonces:



1.3.1. UNIDAD NR2

El procedimiento newtonraphson toma como variables de entrada el vector Xv: condiciónes iniciales para iniciar la iteración, epsilonX: condición de parada para el paso DX,epsilonF: condición de parada para la función F, f: factor de reducción del paso DX
y NITMAX: número máximo de iteraciones admitidas.

Las variables de salida son: Xv donde se devuelve la solución del problema, NITER: número de iteraciones ejecutadas por el método y la variable converge: devuelve
TRUE si la iteración termina por condición de parada dada por los epsilon.

Se modificó la versión existente del flujo de carga anterior. Se le quitó la segunda iteración que realizaba la unidad anteriormente, dado que es innecesaria para el metodo de Newton-Raphson modificado. A continuación transcribimos la unidad:

```
{+doc
+NOMBRE: NR2
+CREACION: 1997
+AUTORES: M. Petrucelli-M. Vigñolo
+REGISTRO:
+TIPO: Unidad Pascal
+OPCIONES DE COMPILACION: Existen las siguientes opciones de compilacion:

        1. DEB_NR: Se puede utilizar para debuggear NR. Va mostrando en
                   pantalla los cálculos que va realizando
        2. EVOLUCION: Si se utiliza genera  la evolucion del paso DX y de
                    la funci¢n objetivo F y la va guardando en dos ectores

+PROPOSITO: Implementaci¢n genérica del método de Newton-Raphson y su
modificación de existir taps para resolución de sistemas de ecuaciones no lineales
+PROYECTO: FLUCAR (Flujo de carga NR)

+MODIFICACION: 08/2000
+AUTOR:  A. Costa-C. Olmedo
+DESCRIPCION: Se le suprimio la segunda iteración implementada en el NR
.realizada para no dar el paso completo.

-doc}



unit  NR2;

interface
uses
                algebrac,XMatDefs, matCPX,usistema,ecuacs,fun4,
servic2{},barrs2,regulado,TyVS2,Matreal;

procedure newtonraphson(var Xv:PVectComplex;var NITER:longint;var
converge:boolean;
                        epsilonX,epsilonF,f:Nreal; NITMAX:longint);

{El procedimiento newtonraphson toma como variables de entrada el vector Xv: cond.
iniciales para iniciar la iteración, epsilonX: condición de parada para el paso DX,
epsilonF: condición de parada para la función F, f: factor de reducción del paso DX
y NITMAX: número máximo de iteraciones admitidas.
Las variables de salida son: Xv donde se devuelve la solución del problema, NITER:
número de iteraciones ejecutadas por el método y la variable converge: devuelve
TRUE si la iteración termina por condición de parada dada por los epsilon}
```

```
procedure calcularvector(var PV:PVectComplex; SJ:PSistema);

{Toma los t,rminos independientes de las ecuaciones del sistema SJ y los almacena
en orden en el vector PV}

implementation

procedure calcularvector(var PV:PVectComplex; SJ:PSistema);

var
        p:PEcuacion;
        k:word;
        numec:integer;

begin
        numec:=SJ^.NumerodeEcuaciones;
        for k:=1 to numec do
        begin
                p:=SJ^.NecPtr(k);
                PV^.pon_e(k,p^.constante);
        end;
end;

        procedure newtonraphson(var Xv:PVectComplex;var NITER:longint;var
                                        converge:boolean;
                        epsilonX,epsilonF,f:Nreal; NITMAX:longint);

var
        rc:complex;
        Xvmas1,DXv,FXv,FXvmas1:PVectComplex;
        sistJ:PSistema;
        nel:word;
        NIT, NIT2:longint;
        b, condreg:boolean;
        histdx,histfx: PVectComplex;

        l:integer;
        nB,nbc,nbcr:word;
        tcompleja:complex;
        t:Nreal;


begin
        {$define deb_nr}
```

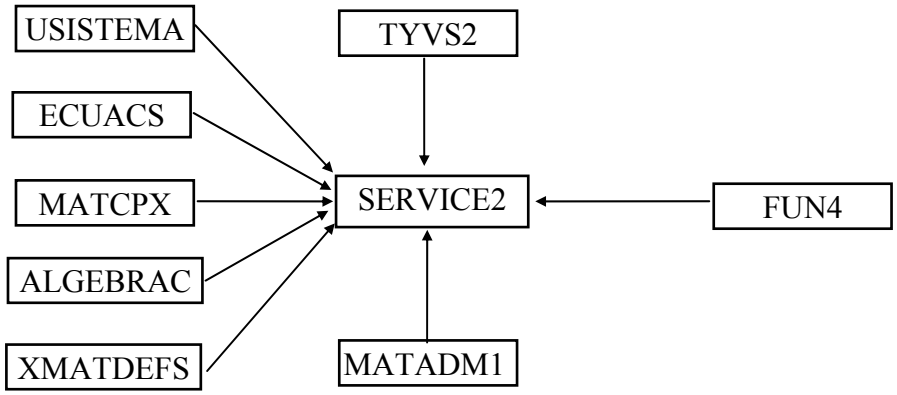
```
{{$define evolucion}
nB:=NBarras;
nBC:=NBarrasdeCarga;
nBcR:=NBarrasconregulador;

NIT:=0;
rc.r:=f;
rc.i:=0;
converge:=false;
{creo los vectores que se usan internamente en la unidad}
nel:=Xv^.n;
New(Xvmas1,Init(nel));
New(DXv,Init(nel));
New(FXv,Init(nel));
New(FXvmas1,Init(nel));
{$IFDEF EVOLUCION}
{creo los vectores que se usan si corre EVOLUCION}
New(histDX,Init(NITMAX));
New(histFX,Init(NITMAX));
{$ENDIF}
{creo el sistema de ecuaciones que se usa en la unidad}
New(sistJ,initcrearsistema(nel));
{comienzo de la iteracion}
writeln('Iteraciones:  ');
repeat
        NIT:=NIT+1;
        NIT2:=0;
        write(NIT);
        {$IFDEF DEB_NR}
        writeln('principal',NIT);
        Xv^.mostrarvector;
        muestrarelreg2(Xv);
        readln;
        {$ENDIF}
        calcularFXv(FXv,Xv);
        {$IFDEF DEB_NR}
        write('Fxv');
        fxv^.mostrarvector;
        readln;
        {$ENDIF}
         calcularfunG(sistJ,Xv);
        {$IFDEF DEB_NR}
        writeln('calcule fung');
        sistJ^.muestrasistema;
        readln;
```

```
{$ENDIF}
sistJ^.EliGaussPivPar;
{$IFDEF DEB_NR}
writeln('muestro sistema luego del pivoteo');
sistJ^.muestrasistema;
readln;
{$ENDIF}
if nit=1 then
begin
        calcularvector(DXv,sistJ);
        sistJ^.Borrartodo;
end;
{$IFDEF DEB_NR}
writeln('cálculo de DXv');
sistJ^.muestrasistema;
write('Dxv');
Dxv^.mostrarvector;
readln;
{$ENDIF}
calcXvmas1(Xvmas1,Xv,Dxv);
{$IFDEF DEB_NR}
write('Xvmas1');
Xvmas1^.mostrarvector;
readln;
{$ENDIF}
calcularFXv(FXvmas1,Xvmas1);
{$IFDEF DEB_NR}
write('Fxvmas1');
FXvmas1^.mostrarvector;
b:=FXvmas1^.cond_epsilon(epsilonF);
writeln('epsilonf',b);
readln;
{$ENDIF}
{$IFDEF EVOLUCION}
calcularevolucion(histDX,histFX,NIT,
DXv^.NormEuclid,FXvmas1^.NormEuclid);
mostrarevolucion(histDX,histFX,NIT,1);
{$ENDIF}
        writeln('DXv');
        Dxv^.mostrarvector;
        readln;
        DXv^.PorComplex(rc);
        writeln('DXv*rc');
        Dxv^.mostrarvector;
        readln;
```

```
                                {$ENDIF}
                                calcXvmas1(Xvmas1,Xv,DXv);
                                calcularFXv(FXvmas1,Xvmas1);
                                {$IFDEF DEB_NR}
                                writeln('Xv');
                                Xv^.mostrarvector;
                                writeln('Xvmas1');
                                Xvmas1^.mostrarvector;
                                readln;
                                writeln( NIT:12, NIT2:12,'FXVmas1',
                                        FXvmas1^.NormEuclid,' dx: ',
                                        DXv^.NormEuclid);
                                writeln('FXVmas1:',FXvmas1^.NormEuclid,'FXv:',
                                        FXv^.NormEuclid);
                                readln;
                                {$ENDIF}

                        {end;   }
{
                (DXv^.cond_epsilon(epsilonX))) then converge:=true;
                NITER:=NIT;

                {Libero memoria}
                Dispose(DXv,Done);
                Dispose(FXv,Done);
                Dispose(FXvmas1,Done);
                {$IFDEF DEB_NR}
                write(memavail,'bytes libres');
                readln;
                {$ENDIF}
                sistJ^.destruirsistema;
                {$IFDEF DEB_NR}
                write(memavail,'bytes libres');
                readln;
                {$ENDIF}

end;

end.
```

1.3.2. UNIDAD SERVICE2

SERVICE2 es una unidad de servicios a la unidad NR2 y al programa principal. Consta de varios procedimientos: calcularevolucion y mostrarevolución son procedimientos propios de uso de NR2 y son genéricos para cualquier función a la que se le aplique el método. En cambio calcXvmas1, si bien es utilizado por NR2, está pensado únicamente para el problema de Flujo de Carga. Por otra parte calcularXv, actualizarbarras y reordenarbarras son procedimientos de uso exclusivo del programa pricipal.



Se modificó la unidad de servicios existente en el flujo anterior, se le quitaron los procedimientos ajustaregulador y ajustaregulador2 dado que los reguladores se ajustan automaticamente en el nuevo método. Se agregó muestrarelreg2 para tener salida de la relación de regulación ahora presente en el vector Xv. Se cambió CalcularXv en caso de regulador, ya que la tensión permanece fija y se debe sacar de los datos originales, mientras que la fase es móvil y se debe sacar de Xv. Se agregó opción de debuggeo a calcularXvmas1 para tener impresión de datos parciales. El procedimiento actualizarbarras incluye ahora el hecho de que V es fijo y no esta dentro de Xv cuando hay regulador.

```
{+doc
+NOMBRE: SERVIC2
+CREACION: 1997
+AUTORES: M. Petrucelli-M. Vigñolo
+REGISTRO:
+TIPO: Unidad Pascal
+OPCIONES DE COMPILACION:
+PROPOSITO: Unidad de servicios para la unidad NR y el
                                programa FLUCAR
```

```
+PROYECTO: FLUCARV3 (Flujo de carga NR)
+MODIFICACIÓN: 08/2001
+AUTORES:  A. Costa-C. Olmedo
+DESCRIPCION: Se le suprimio el procedimiento ajusta regulador, dado que el
regulador es ajustado automaticamente.
-doc}



unit servic2;

interface

uses
        {$I XObjects},usistema,algebrac,XMatDefs,MatCPX,TyVs2,barrs2,
        regulado,fun4,links1,matadm1;


procedure calcularXv(PV:PVectComplex; B,R:PCollection;
                                                nB,nBC,nBcR:integer);

{Este procedimiento calcula el vector de variables del flujo de carga
 a partir de los datos de las barras B y de los reguladores R.
 PV^ es el vector calculado con las fases y los módulos de las tensiones.
 PN^ es el vector calculado con las relaciones de transformación de los
 reguladores.
 nB, nBC y nBcR es el número de barras, el número de barras de carga y
 el número de barras con regulador respectivamente que deben ser
 pasadas como datos al procedimiento}

procedure calcXvmas1(var Xvmas1,Xv,DXv:PVectComplex);

{Calcula el valor siguiente del vector de variables del sistema de ecuaciones
de Newton-Raphson (Xvmas1) a partir del valor previo (Xv) y del paso (DXv) }

procedure calcularevolucion(var histDX,histFX:PVectComplex;
                        NIT:integer; nDX,nFX:Nreal);

{Va guardando en histDX las sucesivas normas del paso de la iteración de NR
(nDX)
y en histFX las sucesivas normas de la función objetivo (nFX). NIT es el número
de la iteración correspondiente.}

procedure mostrarevolucion (histDX,histFX:PVectComplex;
                        Niter,factor:integer);
```

```
        {Muestra los vectores calculados histDX e histFX. Niter es el número total de
        iteraciones realizadas y factor es un número que divide a Niter para determinar
        cuántos elementos de los vectores deben ser mostrados}

        procedure actualizarbarras(var B:PCollection; S:PSistema;
                                   Xv:PVectComplex; nB,nBC,nBcR:integer);

        { Actualiza a partir de los datos del vector de variables del Flujo (Xv) y
         de las admitancias del sistema (S) los campos de las barras (B). nB y nBC
         son el n£mero de barras y el n£mero de barras de carga del sistema
         respectivamente}

        procedure reordenarbarras(BO,B:PCollection);

        {Vuelve al lugar original las barras que al comienzo se ordenaron según el
        tipo. BO son las barras ordenadas. B son las barras originales tal como
        fueron ingresadas}



        procedure muestrarelreg;

        procedure muestrarelreg2(xv:pvectcomplex);

        implementation

procedure calcXvmas1(var Xvmas1, Xv,DXv:PVectComplex);

var
        k:integer;
        res1,res2,res3,res4:complex;
        nel:word;
        DXvaux,DXvaux1:PVectComplex;
        res5,res6: Nreal;

begin
                {{$define deb_xvmas1}
                nel:=DXv^.n;
                New(DXvaux,Init(nel));
                DXvaux^.copy(DXv^);
                for k:=1 to (Nbarrasdecarga+NBarrasconregulador) do
                begin
                        DXv^.e(res1,k+Nbarras-1);
                        Xv^.e(res2,k+Nbarras-1);
```

```
                    res3:=pc(res1,res2)^;
                    DXv^.pon_e(k+Nbarras-1,res3);
                    {$ifdef deb_xvmas1}
                    writeln('calculo de xvmas1');
                    writeln('elemento ',k);
                    writeln('res1 ',res1.r,res1.i);
                    writeln('res2 ',res2.r,res2.i);
                    writeln('res3 ',res3.r,res3.i);
                    {$endif}
                end;
                Xvmas1^.sumvect(Xv^,Dxv^);
                DXv^.copy(DXvaux^);
                Dispose(DXvaux,done);

end;

        procedure calcularXv( PV:PVectComplex; B,R:PCollection;
                                                        nB,nBC,nBcR:integer);

        var
                k,l:integer;
                fasek,modVk,nc:complex;


        begin
                for k:=1 to (nB-1) do
                begin
                        fasek.r:=fase(PBarra(B^.At(k-1))^.V);
                        fasek.i:=0;
                        PV^.pon_e(k,fasek);
                end;
                for k:=1 to nBC do
                begin
                        modVk.r:=mod1(PBarra(B^.At(k-1))^.V);
                        modVk.i:=0;
                        PV^.pon_e((k+nB-1),modVk);
                end;
                for k:=1 to nBcR do
                begin
                        modVk.r:=PRegulador(Reguladores^.At(k-1))^.n;
                        modVk.i:=0;
                        PV^.pon_e((k+nBC+nB-1),modVk);
                        l:=k+nBC;
                end;
```

```
end;

procedure calcularevolucion(var histDX,histFX: PVectComplex;
                            NIT:integer;nDX,nFX:Nreal);

var
        norDXc,norFXc:complex;

begin
        norDXc.r:=nDX;
        norDXc.i:=0;
        histDX^.pon_e(NIT,norDXc);
        norFXc.r:=nFX;
        norFXc.i:=0;
        histFX^.pon_e(NIT,norFXc);
end;

procedure mostrarevolucion (histDX,histFX:PVectComplex;
                            Niter,factor:integer);

var
        k,N:integer;
        xr:Nreal;
        x:complex;

begin
        writeln(' ');
        writeln(' ');
        writeln('histDX');
        N:=Niter div factor;
        for k:=1 to N do
        begin
                histDX^.e(x,k*factor);
                xr:=x.r;
                write(' ');
                write(factor*k);
                write(' ');
                write(xr:6:4);
        end;
        writeln(' ');
        writeln('histFX');
        for k:=1 to N do
        begin
                histFX^.e(x,factor*k);
                xr:=x.r;
```

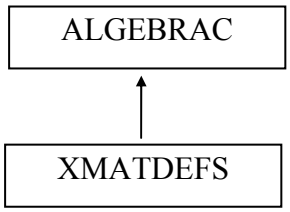
```
                write(' ');
                write(factor*k);
                write(' ');
                write(xr:6:4);
            end;
            writeln(' ');
        end;


        procedure actualizarbarras(var B:PCollection; S:PSistema;
                                                    Xv:PVectComplex;
nB,nBC,nBcR:integer);

        var
            k:word;
            modVnuevo,deltanuevo,Vrnuevo,Vinuevo:Nreal;

        begin
            for k:=1 to (nBC{+nBcR}) do   {para las barras de carga
            cuyos datos son P y Q, solo modifico V y delta}
            begin
                modVnuevo:=complex(Xv^.pte(nB-1+k)^).r;
                deltanuevo:= complex(Xv^.pte(k)^).r;
                Vrnuevo:=modVnuevo*cos(deltanuevo);
                Vinuevo:=modVnuevo*sin(deltanuevo);
                PBarra(B^.At(k-1))^.V.r:=Vrnuevo;
                PBarra(B^.At(k-1))^.V.i:=Vinuevo;
            end;
            for k:=nBC+1 to (nBC+nBcR) do   {para las barras que tienen regulador
V es
            fijo varía solo teta.}
            begin
                modVnuevo:=mod1(Pbarra(B^.At(k-1))^.V);
                deltanuevo:= complex(Xv^.pte(k)^).r;
                Vinuevo:=modVnuevo*sin(deltanuevo);
                PBarra(B^.At(k-1))^.V.i:=Vinuevo;
            end;
            for k:=nBC+nBcR+1 to nB-1 do     {para las barras de generacion y
            voltaje controlado modifico delta y calculo Q}
            begin
                modVnuevo:=mod1(PBarra(B^.At(k-1))^.V);
                deltanuevo:=complex(Xv^.pte(k)^).r;
                Vrnuevo:=modVnuevo*cos(deltanuevo);
                Vinuevo:=modVnuevo*sin(deltanuevo);
                PBarra(B^.At(k-1))^.V.r:=Vrnuevo;
```

```
                PBarra(B^.At(k-1))^.V.i:=Vinuevo;
                PBarra(B^.At(k-1))^.S.i:=reacbarr(Xv,B,S,k,nB,nBC,nBcR);
        end;
        PBarra(B^.At(nB-1))^.S.r:=actibarr(Xv,B,S,nB,nB,nBC,nBcR);
        PBarra(B^.At(nB-1))^.S.i:=reacbarr(Xv,B,S,nB,nB,nBC,nBcR);
end;

procedure reordenarbarras(BO,B:PCollection);
var
r,k:integer;
begin
for r:=1 to NBarras do
        begin
                k:=1;
                while (k<= NBarras)and(r<>PBarra(BO^.At(k-1))^.Nro) do
                inc(k);
                PBarra(B^.At(r-1))^.nombre:=PBarra(BO^.At(k-1))^.nombre;
                PBarra(B^.At(r-1))^.restriccion:=PBarra(BO^.At(k-1))^.
                restriccion;
                PBarra(B^.At(r-1))^.S:=PBarra(BO^.At(k-1))^.S;
                PBarra(B^.At(r-1))^.V:=PBarra(BO^.At(k-1))^.V;
        end;
end;


{       end; }

          procedure muestrarelreg;

          var
                  k: integer;

          begin
    for k:=1 to NReguladores do
    with PRegulador(Reguladores^.At(k-1))^ do
    begin
          writeln('Regulador: ', k-1,' ',n);
    end;
end;


procedure muestrarelreg2(xv:Pvectcomplex);
var
  k,nb, nbc: integer;
  modvic:complex;
```

```
  modvi:Nreal;

begin
   nB:=Nbarras;
   nBC:=Nbarrasdecarga;
   for k:=1 to NReguladores do
   begin
         Xv^.e(modVic,k+Nbc+nB-1);
         modVi:=modVic.r;
         writeln('modvi sin regulador ',modvi);readln;
   end;

end;
end.
```

1.3.3. UNIDAD ALGEBRAC

ALGEBRAC es una de las unidades de la biblioteca de unidades del IIE. Su esquema básico es el que se muestra:



ALGEBRAC implementa el álgebra de los números complejos. Un número complejo es un **record** cuyos campos son la parte real y la parte imaginaria del mismo. Están implementadas las operaciones básicas entre complejos así como también la escritura en pantalla y en archivo.

Reproducimos a continuación:

```
(*****************************************************************
*)
(*  Unidad de complejos                    15/08/89/MEEP/RCH *)
(*****************************************************************
*)
unit   AlgebraC;
interface
uses
      xMatDefs;
```

```
type
        TAspectos = ( CA_Rectangulares, CA_GradosDecimales,
                        CA_Radianes, CA_GradosMinutos );
const
        CA_Campo_Defecto= 11;
        CA_Campo:integer= CA_Campo_Defecto;
        CA_Decs:integer= CA_Campo_Defecto-4;
        CA_ASPECTO: TAspectos = CA_Rectangulares;



Type
        complex = record
                r , i :NReal;
        end;
        pcx = ^complex;
{ La siguiente definici¢n es solo para hacer TypeCast NO PARA definir
variables. }
        VGComplex = array[1..3000] of complex;
        PVGComplex = ^VGComplex;

const {Constantes complejas m s usadas }
        complex_NULO: complex = (r:0; i:0);
        complex_UNO: complex = (r:1; i:0);
        complex_j: complex = (r:0; i:1);


        function   sc  (x,y:complex):pcx; (* suma*)
        function   rc  (x,y:complex):pcx; (* resta *)
        function   pc  (x,y:complex):pcx; (*producto*)
        function   dc  (x,y:complex) :pcx;     (* divicion*)
        function   ppc (x,y:complex):pcx; (* paralel *)
        function   cc  (x  :complex):pcx; (* conjugado*)
        function   prc (a:NReal;x:complex):pcx; (* NReal * complejo *)
        function   numc( a , b:NReal)     :pcx; (* numc:=^(a+j b)   *)
        function        invc( x: complex):pcx; (* inverso*)

        { funcion raiz cuadrada (PRINCIPAL) }
        function raizc( x: complex): pcx;

        { funciones Hiperbolicas }
        function chc( x: complex): pcx;
        function shc( x: complex): pcx;
        function        mod1(x: complex)     :NReal;(* modulo *)
```

```
(* fase en Radianes *)
function  fase(x: complex):    NReal;
function   mod2( x :complex)     :NReal;(* modulo^2 *)

(* Escriben el complejo en la salida estnadar segun CA_ASPECTO *)
procedure wc(x:complex);               (* write(complex) *)
procedure wcln(x:complex);             (* writeln(complex)*)

(* Escriben el complejo en un archivo de texto
 segun el ASPECTO especificado*)
procedure wtxtc(var f: text; x:complex; Aspecto: TAspectos);
procedure wtxtcln(var f: text; x:complex; Aspecto: TAspectos);
(*writeln(complex)    *)
```

1.3.4. UNIDAD XMATDEFS

XMATDEFS es una de las unidades de la biblioteca de unidades del IIE. Contiene definiciones corrientes para uso en matemáticas. No utiliza ninguna otra unidad.

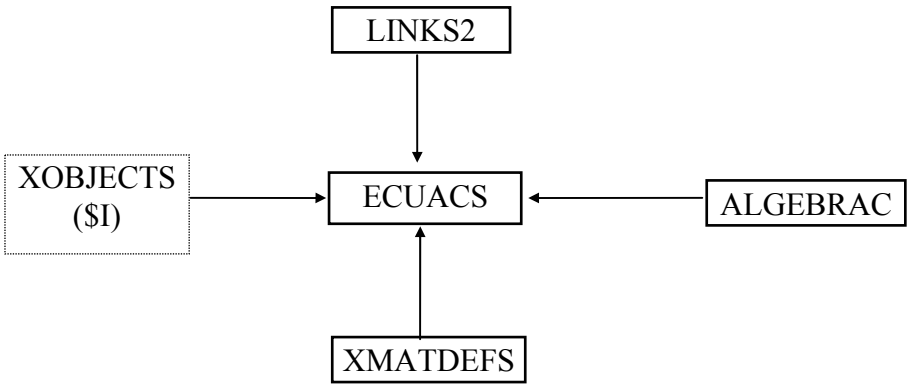
Reproducimos a continuación:

```
unit xMatDefs;

interface

type
{$IFOPT N+}
        NReal = extended;
{$ELSE}
        NReal = real;
{$ENDIF}
  PNReal = ^NReal;

{$IFNDEF ENTERO_LONGINT}
        NEntero = integer;
{$ELSE}
        NEntero = longint;
{$ENDIF}
        PNEntero = ^NEntero;

{ Redefinicion de los timpos estandar para que el compilador detecte
como un error el uso indebido de los mismos }
        extended = boolean;
        real = boolean;

{ Las siguientes constantes son calculadas en el auto-arranque de
la unidad por lo que no es aconsejable utilizarlas en los procedimientos
de auto-arranque de otras unidades pues de hacerlo habrá que tener
}
var
        AsumaCero: NReal; {EPSILON de la maquina en cuentas con NReal }
        DosPi: NReal;           { 2*Pi }

{ abs(x) < AsumaCero }
function EsCero( x: NReal ): boolean;
```

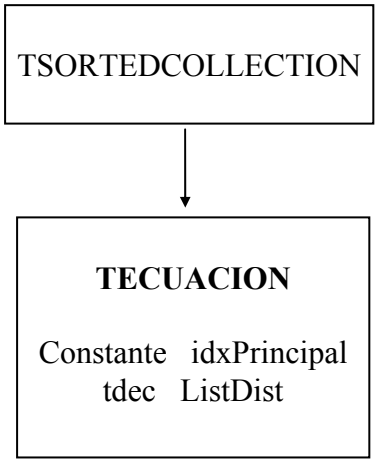
```
{ Casi0:= Abs(x)< xCero.
xCero debe ser un numero positivo }
function Casi0( x: NReal; xCero: NReal): boolean;
```

1.3.5. UNIDAD ECUACS

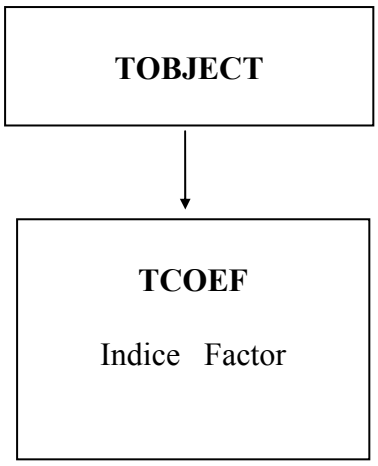
ECUACS es una de las unidades de la biblioteca de unidades del IIE. El esquema básico de la unidad es el que se muestra:



ECUACS define la clase de objetos Tecuacion. Un tipo Tecuacion es una lista de ordenada (**TSortedCollection**) de otros objetos que son los coeficientes de la ecuación. El árbol genealógico de la clase **TEcuacion** es el siguiente:



El árbol genealógico de la clase **TCoef** es el siguiente:



A continuación reproducimos la misma:

```
{+doc
+NOMBRE: ECUACS
+CREACION:16.6.93
+MODIFICACION:17.2.97 proyecto flucar
+AUTORES:rch
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO: Definicion del objeto ecuacion.
+PROYECTO: Flujo de Carga (001)

+REVISION: 11/96
+AUTOR:  MARCELO PETRUCELLI- MARIO VIGNOLO
+DESCRIPCION: Se agregaron la función ValCte y el procedimiento BorrarConstante
                a los  métodos de TEcuacion.
-doc}

unit Ecuacs;
interface
uses
        {$I xObjects}
        , Links, AlgebraC, xmatDefs;

type
        TipoDeEcuacion = (e_nodo, e_corte, e_relten);

type
```

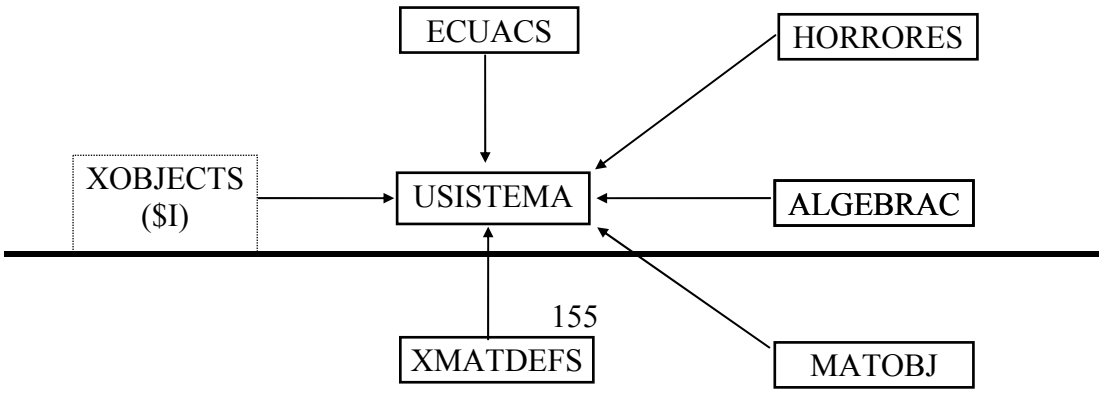
```
PCoef = ^TCoef;

{Clase de Coeficientes complejos. Un coeficiente es un
factor complejo asociado a una variable compleja}
TCoef = object(TObject)

        Indice: TIndice;   {Es el identificador de la varible asociada
                           al coeficiente}

        Factor: complex;   {Es el valor del coeficiente propiamente dicho}
        constructor Init(xIndice:TIndice; xfactor: complex);
        destructor done; virtual;
        procedure acum( xc: complex );   {factor:= factor + xc}
        procedure borrar; { factor := 0 }
        procedure AcumCalc( var Resultado: complex ); {Resultado:= Resultado
                                                + Factor*VariableAsociada}
        constructor Load( var S: TStream );
        procedure Store( var S: TStream ); virtual;
end;

TListDist = object( TSortedCollection )
        constructor init( Alimit, Adelta: integer );
        constructor load( var s: TStream );
        function KeyOf( Item: Pointer): Pointer; virtual;
        function Compare(Key1, Key2: Pointer): Integer; virtual;
end;

PEcuacion = ^TEcuacion;

{TEcuacion es un objeto que implementa una ecuacion como una lista
de coeficientes}
TEcuacion = object( TSortedCollection )
        Constante: complex;   {Es el término independiente de la ecuación}
        idxPrincipal: TIndice; { señala el indice diagonal }
        tdec: TipoDeEcuacion;
        ListDist: TListDist;
        constructor Init;
        {Se inicializa sin coeficientes. El termino constante se inicializa
        a 0}

        procedure Acumular( kindice: TIndice; xFactor: complex);
        {Acumula (xFactor) en el coeficiete asociado a la variable de indice
        (kindice). Primero se busca el coeficiente asociado a (kindice), si
        no se encuentra se agrega}
```

```
        procedure AcumularConstante( xsum: complex );
        {Suma al término independiente de la ecuación xsum}

        function ValCoef(var ResCoef: complex; kindice: TIndice ): boolean;
        {Devuelve en ResCoef, el valor del coeficiente asociado a la
        variable con indice (kindice). El resultado de la funcion es (true)
        cuando la ecuación disponía de un coeficiente para esa variable y es
        (false) cuando no. En cualquiera de los casos el valor devuelto en
        (ResCoef) es el correcto. El resultado de la funcion puede usarse
        para saber si el valor del coeficiente es un CERO absoluto por no
        existir directamente en la ecuación}

        procedure ValCte(var ResCte:complex);
        procedure BorrarCoef( kindice: TIndice);
        {Borra el coeficiente de la ecuacíon. Significa poner el coeficiente
        en CERO}

        procedure BorrarConstante;
        {Borra el término independiente de la ecuación}

        procedure BorrarTodo;
        {Pone todos los coeficientes a CERO incluyendo el termino constante
        NO AFECTA LOS DEMAS PARAMETROS DE LA ECUACION, tales
        como el indice
        principal, el tipo de ecuacion y la lista de distribucion}

        procedure CalcNC( var Resultado: complex );
        {Calcula la suma de los productos de los coeficientes de la ecuacion
         por el valor de sus respectivas variables asociadas y le suma el
         campo (Constante)}

        constructor Load( var S: TStream );
        procedure Store( var S: TStream ); virtual;
        procedure AgregarListDist( nec: TIndice; factor: complex );
        function MaxIndVar: TIndice;    {mayor indice de variable involucrado }

private
        function KeyOf( Item: Pointer): Pointer; virtual;
        {retorna la direccion del campo (Indice) del coeficiente  apuntado
        por Item}

        function Compare(Key1, Key2: Pointer): Integer; virtual;
        {Permite ordenar los coeficientes de una ecuacion por orden creciente
        de las varibles asociadas a los mismos. (Key1 y Key2) seran resultados
```

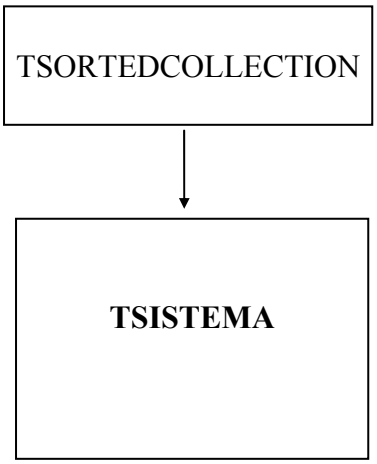
```
                de llamadas a (KeyOf) y por lo tanto son punteros a objetos del tipo
                TIndice. El resultado es (-1) si el TIndice apuntado por Key1 es mayor
                que el apuntado por Key2, es (0) si es iguales y (+1) si es menor}

        end;

        { TDisR, esta pensado para registrar los lugares en donde todo lo
        que se intente agregar a los coeficientes de una ecuación debe ir a
        agregarse }

        PDisR = ^TDisR;
        TDisR = object( TObject )
                nec: TIndice;
                Factor: complex;
                constructor Init( xNEc: TIndice; xFactor: complex );
                constructor Load( var s: TStream );
                procedure Store( var s: TStream  );
  end;

function EliminarVariable(var EcDestino, EcEliminadora: TEcuacion;
                          kIndVar: TIndice ): integer;
{Elimina de EcDestino la variable de índice kIndVar usando la EcEliminadora}

procedure CombinarEcuaciones( var EcDestino, EcOrigen: TEcuacion;
                                  mult: complex );
{Suma a la ecuacion (EcDestino) la ecuacion (EcOrigen) multiplicada
por el multiplicador (mult) }
```

1.3.6. UNIDAD USISTEMA

USISTEMA es una de las unidades de la biblioteca de unidades del IIE. El esquema básico de la unidad es el que se muestra:



USISTEMA define la calse **TSistema** (Sistema de ecuaciones). Un sistema de ecuaciones es una lista ordenada (**TSortedCollection**) de otros objetos que son las ecuaciones del sistema. El árbol genealógico de la clase **TSistema** es el siguiente:



A continuación reproducimos la misma:

```
{+doc
+NOMBRE: usistema
+CREACION: 18.12.94
+MODIFICACION: 13.02.97 proyecto flucar
+AUTORES: rch
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO: encabezamiento estandar
+PROYECTO: redes

+REVISION:  2/97
+AUTOR:   MARCELO PETRUCELLI - MARIO VIGNOLO
+DESCRIPCION:    Se agregaron los procedimientos initcrearsistema, destruirsistema,
                 AcumularConstante, ValCte, BorrarConstante, BorrarTodo y
                 EliGaussPivPar
-doc}

unit usistema;
```

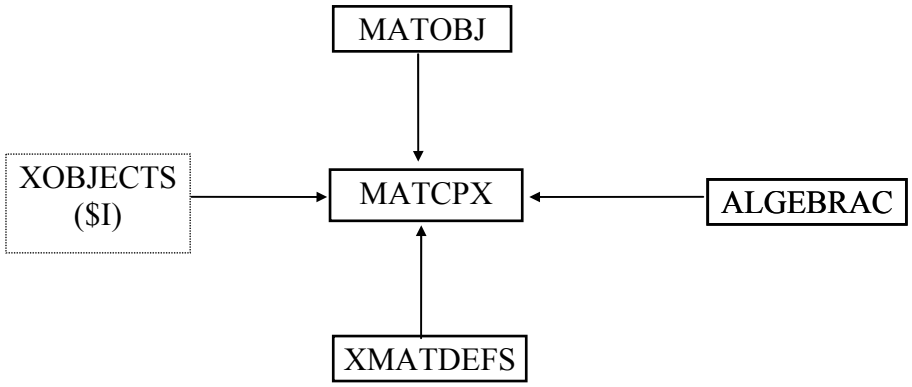
```
interface

uses
        ecuacs, {$I xObjects},XMatDefs, AlgebraC, horrores,MatObj;

type

        TIndice= integer;
        PSistema= ^TSistema;
        {TSistema es una clase de objetos que implementa los sistemas de ecuaciones
        como listas de objetos de tipo TEcuacion}

        TSistema= object(TSortedCollection )
                constructor Init( Limite, Incremento: integer );
                {Init de TSistema como TSortedCollection}

                constructor initcrearsistema(nec:word);
                {Crea un sistema con nec ecuaciones}

                destructor done;virtual;
                destructor destruirsistema;
                {destruye completamente un sistema de ecuaciones}

                procedure Acumular( nec, ncol: TIndice; xFactor: complex);
                {Acumula (xFactor) en el coeficiete asociado a la variable de indice
                (ncol) de la ecuacion (nec). Primero se busca el coeficiente asociado
                a (ncol) en la ecuacion (nec), si no se encuentra se agrega}

                procedure AcumularConstante(nec:TIndice;xFactor:complex);
                {Suma al término independiente de la ecuación (nec) xsum}

                function NumeroDeEcuaciones: integer;
                { Numero de ecuaciones en el sistema }

                function MaxIndVar: integer;
                { Maximo indice de variable involucrado }
                function ChequearFormacionDiagonal: boolean;
                { Chequeo que cada ecuacion tenga variable diagonal
                correspondiente}

                function ValCoef(var ResCoef: complex; nec, ncol:
                TIndice ): boolean;
                {Devuelve en ResCoef, el valor del coeficiente asociado a la
                variable con indice (ncol) de la ecuacion (nec). El resultado de
                la funcion es (true) cuando la ecuación disponía de un coeficiente
```

```
para esa variable y es (false) cuando no. En cualquiera de los casos
el valor devuelto en (ResCoef) es el correcto. El resultado de la
funcion puede usarse para saber si el valor del coeficiente es un
CERO absoluto por no existir directamente en la ecuación}

procedure ValCte(var ResCte: complex; nec:TIndice);
{Devuelve en ResCte el término independiente de la ecuación (nec)}

procedure BorrarCoef( nec, ncol: TIndice);
{Borra el coeficiente (ncol) de la ecuacíon (nec). Significa poner el
coeficiente en CERO}

procedure BorrarTodo;
{Pone todos los coeficientes a CERO incluyendo el termino constante
NO AFECTA LOS DEMAS PARAMETROS DE LA ECUACION, tales
como el indice principal, el tipo de ecuacion y la lista de distribucion}

procedure BorrarConstante(nec: TIndice);
{Borra el término independiente de la ecuación (nec)}

procedure Calc( var Resultado: complex; nec:TIndice );
{Calcula la suma de los productos de los coeficientes de la ecuacion
(nec) por el valor de sus respectivas variables asociadas y le suma el
campo (Constante)}

constructor Load( var S: TStream );
procedure Store( var S: TStream ); virtual;
function NEcPtr( nec: TIndice): PEcuacion;
{Devuelve el puntero a la ecuacion con idxprincipal igual a (nec)}

{
procedure NotificarCambioEcCorte( npos, nneg: TIndice );
}
procedure Pon(nEc, nVar: integer; Y: complex );

procedure PonY( n1, n2: integer; xy: complex);
procedure PonIG(       nneg, npos: TIndice;
        nindep: TIndice; { Numero de la funcion independiente asociada
}
        IgVal: complex );

procedure PonIgV( negIg, posIg, negV, posV: integer;
                ay: complex);
procedure PonZVG(
        nneg, npos: TIndice;
```

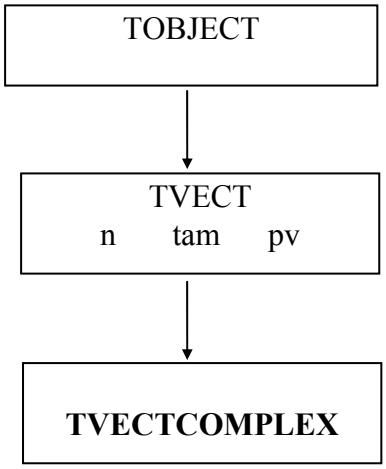
```
                nindep: TIndice; {Numero de func. indep. asociada }
                ZVal, EVal: complex );

        procedure PonTrafoZcc(
                PriPos, PriNeg, SecPos, SecNeg: TIndice;
                Zcc, n: complex );
        {
        procedure PonVG(nneg, npos: TIndice; EVal: complex );
        procedure PonTI(ppos, pneg, spos, sneg: TIndice;
        vs_d_vp: complex );
         }

        function EliminarVariable(
                EcDestino, EcEliminadora: TIndice;
                kIndVar: TIndice ): integer;

        procedure CombinarEcuaciones(
                EcDestino, EcOrigen: TIndice;
                mult: complex );

        function EliminacionDiagonalCompleta: integer;
        { Toma las ecuaciones de a una y elimina la variable asociada
        al elemento diagonal del resto de las ecuaciones. Cuando se completa
        el procedimiento se a resulto el sistema para las variables asociadas
        a la diagonal }

        procedure EliGaussPivPar;
        {dado un sistema,hace eliminacion  gaussiana con pivoteo
        parcial y lo resuelve quedando la solucion en la cte de cada ecuacion}

        procedure muestrasistema;
        {Muestra el sistema de ecuaciones}



private
        function KeyOf( Item: Pointer): Pointer; virtual;
        function Compare(Key1, Key2: Pointer): Integer; virtual;
    end;
```

1.3.7. UNIDAD MATCPX

MATCPX es una de las unidades de la biblioteca de unidades del IIE. El esquema básico de la unidad es el que se muestra:



MATCPX define la clase **TVectComplex** (Vectores complejos). Un tipo **TVectComplex** es un descendiente de la clase **TVect** definida en la unidad MATOBJ. El árbol genealógico de la clase **TVectComplex** es el siguiente:



A continuación reproducimos la misma:

+NOMBRE: MatCPX
+CREACION: 1994
+MODIFICACION: 3/97
+AUTORES: rch

```
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO: definicion del objeto matriz de complejos.
+PROYECTO: rchlib

+REVISION:  3/97
+AUTOR:  MARCELO PETRUCELLI - MARIO VIGNOLO
+DESCRIPCION:    Se agregaron los procedimientos sumvect y mostrarvector y la
                 función cond_epsilon
-doc}
unit MatCPX;
interface
uses
        {$I xobjects}, xMatDefs,MatObj, AlgebraC;

type

        PVectComplex = ^TVectComplex;

        {Se implementa la clase de vectores complejos como una clase descendiente
        de TVect}
        TVectComplex = object(TVect)
                        constructor Init(ne:word);
                        {Crea un vector de numeros complejos de largo (ne)}

                        constructor Ventana( ne: word; var x );
                        constructor Load( var S: TStream );
                        procedure e(var res: complex; k:word);
                        {Devuelve en (res) el elemento del vector que ocupa el lugar
                        k}

                        procedure pon_e(k:word; var x:Complex);
                        {Asigna al elemento que ocupa el lugar (k) en el vector el
                        valor dado por x}

                        procedure acum_e(k:word; var x: Complex);
                        {Suma al elemento que ocupa el lugar (k) en el vector el
                        valor dado por x}

                        procedure PEV(var res: complex; var y :TVectComplex);
                        {Implementa el producto escalar de dos vectores devolviendo
                        el resultado en (res)}

                        procedure PorComplex(var  r: complex );
                        {Implementa el producto de un vector por un escalar}
```

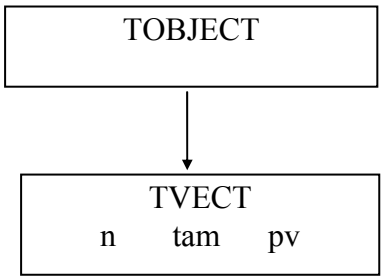
```
                    procedure sum(var y:TVectComplex);
                    {Suma de dos vectores.  El resultado queda en el vector
                    que llama al método}

                    procedure sumComplexPV(var  r: complex;var
                    x:TVectComplex);
                    function ne2:NReal; {norma euclideana al cuadrado }
                    function normEuclid:NReal; {norma euclideana}
                    function normMaxAbs:NReal;
                    function normSumAbs:NReal;
                    procedure Copy(var x:TVectComplex);
                    {Copia el vector (x) en el vector que llama al método}

                    procedure Ceros; virtual;
                    {Toma el vector que llama al método y lo llena con ceros}

                    function cond_epsilon(tol:NReal):boolean;
                    {Si todos los elementos del vector son menores que (tol)
                    el resultado de la función es true}

                    procedure sumvect (V,dV:TVectComplex);
                    {Efectua la suma de V y dV guardando el resultado en el vector
                    que llama al método}

                    procedure mostrarvector;
                    {Despliega el vector en pantalla}
        end;

const
        RVectComplex: TStreamRec = ( ObjType: 4015;
         VmtLink: Ofs(TypeOf(TVectComplex)^);
         Load:    @TVectComplex.Load;
         Store:   @TVectComplex.Store
        );

type
        PComplex = ^TMatComplex;
        TMatComplex = object(TMat)
                constructor Init(filas,columnas:word);
                constructor Load( var S: TStream );
                procedure e(var res: complex; k,j: word);
                procedure pon_e(k,j: integer; x: Complex);
                procedure acum_e(k,j:integer; x: Complex);
                procedure Mult(a,b:TMatComplex);
```

```
                procedure Traza( var res:Complex );
                procedure Deter( var res:Complex );
                procedure Escaler(var res: complex; var i: TMatComplex);
                procedure CopyColVect(var Y: TVectComplex; J: Integer);
                procedure Inv;
                procedure Ceros; virtual;
                procedure CerosFila( kfil: integer);
                procedure WriteM;
        end;

const
        RMatComplex: TStreamRec = ( ObjType: 4016;
         VmtLink: Ofs(TypeOf(TMatComplex)^);
         Load:    @TMatComplex.Load;
         Store:   @TMatComplex.Store
        );
procedure RegisterMatReal;
```

1.3.8. UNIDAD MATOBJ

MATOBJ es una de las unidades de la biblioteca de unidades del IIE. El esquema básico de la unidad es el que se muestra:

MATOBJ define la calse **TVect** (Vectores genéricos). Un tipo **TVect** es un descendiente de la clase **TObject** definida en la unidad **xObjects** de Pascal. El árbol genealógico de la clase **TVect** es el siguiente:



A continuación reproducimos la misma:

```
{------------------------------------------------------ RCH/90 --
 Ing. Ruben Chaer                                  M.E.E.P.
 Dpto. Maquinas Electricas y Electronica de Potencia
 Av. J. Herrera y Reissig No 565
-----------------------------------------------------------------}


{+doc
+NOMBRE: MatObj
+CREACION: 1.4.1990
+MODIFICACION:  5/97 Se agregaron algunos comentarios - Proyecto FLUCAR -
+AUTORES: rch
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO: Definición de objetos abstractos TVect y TMat
```

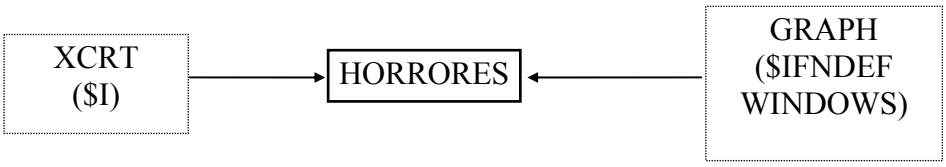
```
+PROYECTO: rchlib

+REVISION: 9.02.93. Se implemento lo necesario para salvarlos
                    usando TStream.
+AUTOR:
+DESCRIPCION:

-doc}


unit MatObj;
interface
uses
        {$I xObjects}, Horrores;

type


PVect = ^TVect;

{Se implementan los vectores como una clase de objetos descendientes de TObject}

TVect=object(TObject)
        n,tam:word;
        {n es el numero de elementos del vector; tam es el espacio de memoria que
        ocupa cada elemento}

        pv:pointer;
        {pv es un puntero al primer elemento del vector}

        constructor Init(ne,te:word);
        {Crea un vector con (ne) elementos, es decir un espacio en memoria de tamaño
        ne*te}

        constructor Ventana( ne, xtam: word; var x );
        constructor Load(var arch:TStream);
        procedure Store(var arch:TStream); virtual;
        procedure Print; virtual;
        function pte(k:word):pointer;
        {Devuelve un puntero al elemento que ocupa el lugar (k) en el vector)
        procedure inc(var p:pointer);
        procedure dec(var p:pointer);
        procedure igual(var x:TVect);
        destructor Done; virtual;
        function IndiceEnRango( k: word ): boolean;
```

```
        function IndiceDe( p: pointer ): word;
        procedure SHLnk( nk: integer );
        procedure SHRnk( nk: integer );
        procedure SHLX(var X);
        procedure SHRX(var X);
        { Incrementa la dimension del vector en (nmas) elementos. Para hacerlo
        pide memoria para el vector ampliado y copia los datos existentes y
        libera la memoria ocupada anteriormente. Cual quier referencia
        a direcciones de memoria (punteros) al vector aterior son inválidas
        luego de la ampliacion. Por lo tanto si utiliza este metodo NO
        utilize punteros para referenciar elementos del vector en forma
        permanente }
        procedure IncDimension( nmas: integer );
end;


const
  RVect: TStreamRec = (       ObjType: 4003;
         VmtLink: Ofs(TypeOf(TVect)^);
         Load:    @TVect.Load;
         Store:   @TVect.Store
        );

type
PMat = ^TMat;
TMat=object(TVect)
        nf,nc:word;
        constructor Init(filas,columnas,BytesPorElemento:word);
        constructor Load(var s: TStream);
        procedure Store(var s: TStream); virtual;

        function pte(k,j:word):pointer;
        procedure incol(var p:pointer);
        procedure decol(var p:pointer);
        procedure IntercambieFilas(k,j:word);
        procedure igual(var x:TMat);
        destructor Done; virtual;
        function IndiceEnRangoColumnas( j: word ): boolean; virtual;
        function pFila( fila: integer ): pointer; virtual;
        procedure Trasponer;
end;

const
  RMat: TStreamRec = (        ObjType: 4004;
         VmtLink: Ofs(TypeOf(TMat)^);
```

```
        Load:   @TMat.Load;
        Store:  @TMat.Store
       );

procedure RegisterMatObj;
```

1.3.9. UNIDAD HORRORES

HORRORES es una de las unidades de la biblioteca de unidades del IIE. El esquema básico de la unidad es el que se muestra:



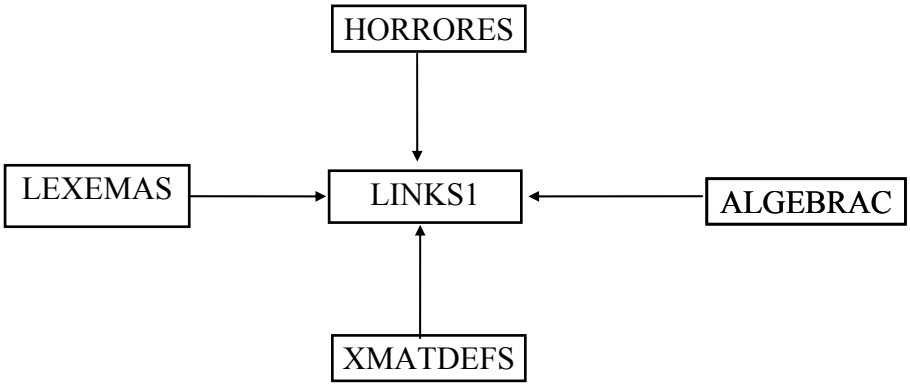
HORRORES es una unidad que unifica el tratamiento de errores.

A continuación reproducimos la misma:

```
{+doc
+NOMBRE:horrores
+CREACION:1.1.90
+AUTORES:rch
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:Servicio de unificacion del tratamiento de errores.
+PROYECTO:rchlib

+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit Horrores;
interface
uses
        {$I xCRT};

procedure Error(x:string);
{Escribe en pantalla un mensaje de error y agrega (x) al mensaje}
```

1.3.10. UNIDAD LINKS1

LINKS1 es una de las unidades de la biblioteca Pascal del IIE. Mostramos un esquema a continuación seguido de la interfase de la unidad.




```
{+doc
+NOMBRE:  LINKS1
+CREACION: 8/97
+AUTORES:  MARIO VIGNOLO
+MODIFICACION:
+REGISTRO:
+TIPO:  Unidad Pascal
+PROPOSITO: Esta unidad es un parche para permitir llamadas a
                                unidades no referenciadas para evitar referencias
circulares
+PROYECTO:

+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit Links1;
interface
uses
        AlgebraC, Lexemas, xMatDefs, Horrores;

type

        TIndice = Integer;

        TFunc_Indice= function(var r: string; var rescod: integer):TIndice;
```

```
        TFunc_BarraPtr = function ( k: TIndice): pointer;



var
        { Esta funcion se define el la unidad TYVS2, la cual se encarga de
        hacer el LINK en la inicializacion. }
        Func_IndiceDeNodo: TFunc_Indice;
        Func_BarraPtr: TFunc_BarraPtr;


function LeerNReal(var a: TFlujoLetras; var resultado: NReal):integer;
function LeerNInteger( var a: TFLujoLetras; var resultado: integer): integer;
function LeerNComplex( var a: TFLujoLetras; var resultado: complex): integer;

implementation

function LeerNReal(var a: TFlujoLetras; var resultado: NReal):integer;
label Check1;
var
        negativo:boolean;
        res: integer;
        r: string;
begin
        negativo:= false;
check1:
        getlexema(r,a);
        if r='N' then
        begin
                LeerNReal:= 114;
                exit;
        end;


        if r='-' then
        begin
                negativo:= true;
                goto check1;
        end;

        if r='+' then goto check1;
        val(r, resultado, res);
        if res <> 0 then error('convirtiendo a real');
        if negativo then resultado := -resultado;
        LeerNReal:= res;
```

```
end;

function LeerNInteger( var a: TFLujoLetras; var resultado: integer): integer;
label Check1;
var
        negativo:boolean;
        res: integer;
        r: string;
begin
        negativo:= false;
check1:
        getlexema(r,a);

        if r='-' then
        begin
                negativo:= true;
                goto check1;
        end;

        if r='+' then goto check1;
        val(r, resultado, res);
        if res <> 0 then error('convirtiendo a integer');
        if negativo then resultado := -resultado;
        LeerNInteger:= res;
end;

function LeerNComplex( var a: TFLujoLetras; var resultado: complex): integer;
label Check1;
var
        cs: string;
        negativo:boolean;
        res: integer;
        LeyendoParteReal: boolean;
        r: string;
begin
        negativo:= false;
        LeyendoParteReal:= true;
        cs:='';
check1:

        getlexema(r,a);

        if r='-' then
        begin
                negativo:= true;
```

```
            cs:=cs+' '+r;
            goto check1;
        end;

        if r='+' then
        begin
            cs:=cs+' '+r;
            goto check1;
        end;

        if leyendoParteReal then
        begin
            { Parte Real }
            val(r, resultado.r, res);
            if negativo then resultado.r:= -resultado.r;
            if res <> 0 then error('convirtiendo a ParteReal');
            leyendoParteReal:= false;
            negativo:= false;
            cs:='';
            goto check1;
        end
        else { Parte Imaginaria }
        if res = 0 then
        begin
            if (pos('j',r)=1)or (pos('i',r)=1) then
            begin
                delete(r,1,1);
                if length(r) = 0 then getlexema(r,a);
                val(r, resultado.i, res);
                if negativo then resultado.i:= -resultado.i;
                if res <> 0 then error('convirtiendo a ParteImaginaria');
                cs:='';
            end
            else
            begin
                r:= cs+' '+r;
                PutLexema(r, a);
                resultado.i:=0;
            end;
        end;
        LeerNComplex:= res;
end;

end.
```

1.3.11. UNIDAD TDFS0

TDFS0 es una de las unidades de la biblioteca Pascal del IIE. Define el conjunto de restricciones posibles para las barras:

```
unit TDEFS0;
interface

type
        TStr8 = string[8];
        TPosiblesRestriccionesDeNodo = ( cf_P, cf_Q, cf_V, cf_delta );
        TRestriccionDeNodo = Set Of TPosiblesRestriccionesDeNodo;
        PSTR = ^string;
implementation

end.
```

1.3.12. UNIDAD LEXEMAS

LEXEMAS es una de las unidades de la biblioteca Pascal del IIE. Resuelve el flujo de letras y lexemas. Mostramos un esquema de la unidad y a continuación reproducimos su interfase:


```
{+doc
+NOMBRE: lexemas
+CREACION:1.1.92
+AUTORES:rch
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:FLujo de Letras y Lexemas.
+PROYECTO:rchlib

+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit Lexemas;
interface

uses
        {$I xCRT},
        {$I XObjects};

const
        EliminarComentariosLlaves:boolean = true;

        err_FinArchivo = 1;
        err_BULLleno = 2;
  err_LetraNoAutorizada = 3;

        ESC = #27;
        CEOF = chr(27);
```

```
        LF = chr(10);
        CR = chr(13);
        TAB = chr(9);
        Transparentes = [' ', LF, CR, TAB];
  {$IFDEF WINDOWS}
        Letras = ['a'..'z','A'..'Z','0'..'9','_',"",'á','é','í','ó','ú','ñ','Ñ','?'];
  {$ELSE}
        Letras = ['a'..'z','A'..'Z','0'..'9','_',"",' ',',','¡','¢','£','¤','¥','?'];
  {$ENDIF}
  Cualificadores = ['.','\'];
        Separadores = ['(',')', ',' , ';','[',']',':','|' ];
        Operadores = ['+','-','*','=','/','<','>'];

const
        LongKeyBuffer = 150;

type

        TFlujoLetras = object
                pf: PStream;
                KeyBuffer : array[0..LongKeyBuffer-1]of char;
    BufferReadIndex,BufferWriteIndex:integer;

    constructor Init( var XF: TStream );
                procedure TomoLetra( var c: char);
                procedure DevuelvoLetra( c: char);
                procedure EsperarLetra(xc:char);
        end;

procedure GetLexema( var lexema: string; var FlujoLetras: TFlujoLetras );
procedure PutLexema( var lexema: string; var FlujoLetras: TFlujoLetras );

function BuscarLexema( lexema: string; var FlujoLetras: TFlujoLetras;
                                IgnorarMaMi: boolean): boolean;
function UpStr( s: string ):string;
function UpCase( c: char ):char;

var
        lx_Error : procedure ( codigo: byte);

procedure DefErrProc( codigo: byte);
```

1.4. Modelado del problema utilizando objetos

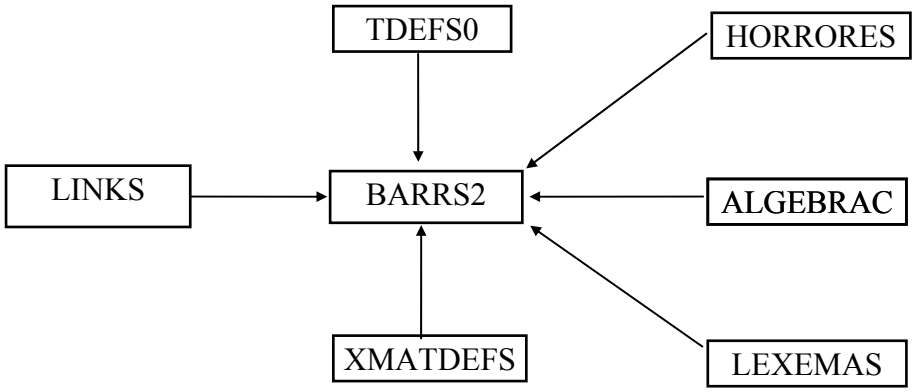
De acuerdo a lo expuesto en el punto 2., los elementos existentes en el problema que nos estamos planteando son las **barras**, las **impedancias, cuadripolos, los transformadores** y **reguladores**. Por lo tanto resulta intuitivo imaginarnos un escenario donde los objetos que "actuen" sean estos cinco. Un regulador es, a los efectos de este programa de flujo de carga, un transformador en el cual la relación **n** de transformación es variable.

Como una de las consignas de nuestro proyecto fue utilizar el flujo de carga FLUCAR proveniente de la biblioteca de software del IIE, entonces usamos como base del FLUCARV3.0 las clases de objetos creadas en el FLUCAR2.0 (que resuelve el problema de flujo de carga a partir del método de Newton-Raphson clásico. FLUCAR utiliza las unidades BARRS2, IMPDS1, CUADRIPO1, TRAFOS1,REGULADO donde están implementadas respectivamente las clases de objetos barras, impedancias, cuadripolos, transformadores y reguladores. Veamos cada una de estas unidades.

1.4.1. UNIDAD BARRS2

Define el tipo de objetos **Tbarra**. Un tipo **Tbarra** es una clase de objetos cuyos campos son: el **nombre** de la barra,el tipo de barra, una **restricción** (recordamos que una barra puede ser *flotante*, de *carga* o de *generación y voltaje controlado, o con regulador*), la **potencia** S (entrante a la barra, S = P + j Q), un valor de **tensión** en la barra V (número complejo) y un valor de la corriente máxima admisible **Imax**.. A partir de los métodos implementados un tipo **Tbarra** puede crearse pasandole el valor de cada uno de sus campos al constructor Init o leyendo estos valores de un archivo usando el constructor LeerdeFljLetras. Además utilizando el procedimiento WriteTXT es posible que una barra despliegue sus campos en un archivo de texto.

El esquema básico de la unidad BARRS es el que se muestra a continuación.




```
{+doc
+NOMBRE: BARRS2
+CREACION: 9/97
+MODIFICACION:
+AUTORES: MARIO VIGNOLO
+REGISTRO:
+TIPO: Unidad Pascal
+PROPOSITO: Definicion del objeto Barra
+PROYECTO: FLUCAR
+REVISION:2001
+AUTOR: A.Costa-C.Olmedo
+DESCRIPCION:
-doc}

unit barrs2;

interface

uses
        {$I Xobjects},Horrores, xMatDefs, Lexemas, AlgebraC, Links1;

type

        TStr8 = string[8];
        TPosiblesRestriccionesDeNodo = ( cf_P, cf_Q, cf_V, cf_delta );

        TRestriccionDeNodo = Set Of TPosiblesRestriccionesDeNodo;
```

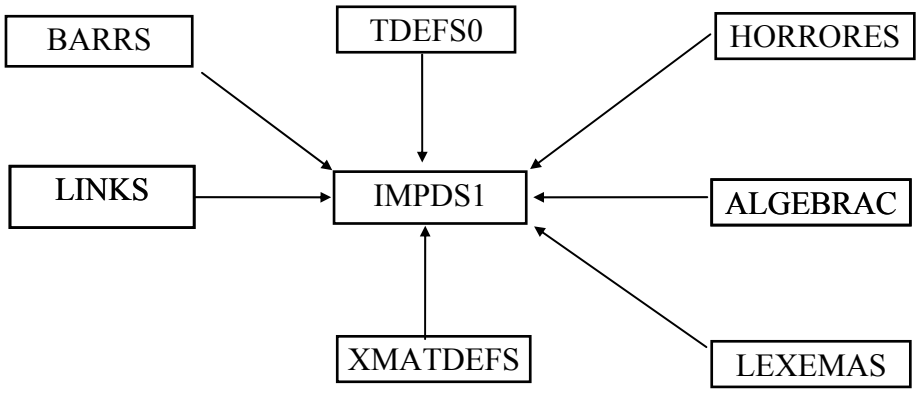
```
        PBarra = ^TBarra;
        TBarra = object(TObject)
                Nombre: TStr8;
                Nro: TIndice; {1,2,3 o 4}
                Restriccion: TRestriccionDeNodo;
                S, V: complex;
                Limitemin, Limitemax: char;
                Rmin,Rmax: NReal;
                constructor Init(xNombre: string; xtipo: TRestriccionDeNodo; xP, xQ,
                xV, xDelta: NReal );
                constructor LeerDeFljLetras( var a: TFlujoLetras; var r: string;
                                                var tipodeBarra: TRestriccionDeNodo);
                procedure WriteTXT( var f: text);
        end;


        { (vk-vj)^2 }
        function ModV2( k,j:integer):Nreal;


implementation

function ModV2( k,j:integer):Nreal;
begin
        ModV2:= mod2(rc(PBarra(func_BarraPtr(k-1))^.V, PBarra(func_BarraPtr(j-
1))^.V)^);
end;

procedure TBarra.WriteTXT( var f: text);
begin
        write(f,'b: ',Nombre,'  S:  ');
        wtxtc(f,S,CA_Rectangulares);
        write(f,'  V: ');
        wtxtc(f,V,CA_GradosDecimales);
end;

constructor TBarra.Init(
                        xNombre: string;
                        xtipo: TRestriccionDeNodo;
                        xP, xQ, xV, xDelta: NReal ); { los que no tengan sentido 0 }
begin
        Nombre:= Copy(xNombre,1,8);

        Restriccion:= xtipo;
        S:= numc( xP, xQ)^;
```

```
        V:= numc( xV*cos(xDelta), xV*sin(xDelta))^;

end;


constructor TBarra.LeerDeFljLetras( var a: TFlujoLetras; var r: string; var tipodeBarra:
TRestriccionDeNodo);

var
        P, Q, xV, Delta: NReal;
        res,T: integer;

begin
        { Nombre ID de la Barra }
        if length(r) > 8 then Nombre := Copy(r, 1,8)
        else Nombre:=r;
        { Restriccion y Datos }
        restriccion := [];
        res:=LeerNInteger(a,T);
        if T=1 {Barra flotante} then restriccion:=restriccion + [cf_V,cf_delta]
        else
        if T=2 {Barra de carga} then restriccion:=restriccion + [cf_P,cf_Q]
        else
        if T=3 {Barra de g y v cont.} then restriccion:=restriccion + [cf_P,cf_V]
        else
        if T=4 {Barra con regulador} then restriccion:=restriccion + [cf_P,cf_Q,cf_V]
        else
        horrores.error('Los tipos de barra solo pueden ser 1, 2, 3 o 4');
        res:=LeerNReal(a, P);
        res:=LeerNReal(a, Q);
        res:=LeerNReal(a, xV);
        res:=LeerNReal(a, delta);
        res:=LeerNReal(a,Rmin);
        if res=114 then limitemin:='N'
        else limitemin:='S';
        res:=LeerNReal(a,Rmax);
        if res=114 then limitemax:='N'
        else limitemax:='S';
        TipodeBarra:=restriccion;
        Init(Nombre,restriccion,P,Q,xV,Delta);
end;

end.

{+doc
```

1.4.2. UNIDAD IMPDS1

Define el tipo de objetos **TImpedancia**. Un tipo **TImpedancia** es una clase de objetos cuyos campos son: el **nombre** de la impedancia, los nodos entre los cuales se encuentra (**Nodo1, Nodo2),** el valor **Z** (número complejo) de la misma y la corriente máxima admisible **Imax**. A partir de los métodos implementados un tipo **TImpedancia** puede crearse pasándole los valores de cada uno de sus campos usando el constructor Init o leyendo dichos valores de un archivo usando el constructor LeerdeFljLetras. Además utilizando la función Perdidas es posible determinar el valor de las pérdidas en la impedancia.

El esquema básico de la unidad IMPDS es el que se muestra a continuación.




```
{+doc
+NOMBRE: Impds1
+CREACION: 8/97
+AUTORES: MARIO VIGNOLO
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:  Definición del objeto Impedancia.
+PROYECTO: FLUCAR
+REVISION:2001
+AUTOR:A.Costa-C.Olmedo
+DESCRIPCION:
-doc}

unit Impds1;
```

```
interface
uses
        {$I XObjects},
        Horrores, xMatDefs, Lexemas, AlgebraC, TDEfs0,Barrs2,Links1;

type
        PImpedancia=^TImpedancia;
        TImpedancia = object(TObject)
                Nombre: TStr8;
                Nodo1, Nodo2: integer;
                Z: complex;
                Imax: NReal;
                constructor Init(
                        xNombre: string;
                        xNod1, xNod2: integer;
                        xZ: complex; xImax: NReal);
                constructor LeerDeFljLetras( var a: TFlujoLetras; var r: string);
                procedure TransfS(var S12,S21,Scon,I: complex); virtual;
        end;

implementation

constructor TImpedancia.Init(
                        xNombre: string;
                        xNod1, xNod2: integer;
                        xZ: complex; xImax: NReal);

begin
        Nombre:= Copy(xNombre,1,8);
        Nodo1:= xNod1;
        Nodo2:= xNod2;
        Z:= xZ;
        Imax:= xImax;
end;

procedure TImpedancia.TransfS(var S12,S21,Scon,I: complex);

var
        I12,I21:complex;

begin
        if (nodo1<>0) and (nodo2<>0) then
        begin
                I12:=dc(rc(PBarra(func_BarraPtr(Nodo1-1))^.V,
                                PBarra(func_BarraPtr(Nodo2-1))^.V)^,Z)^;
```

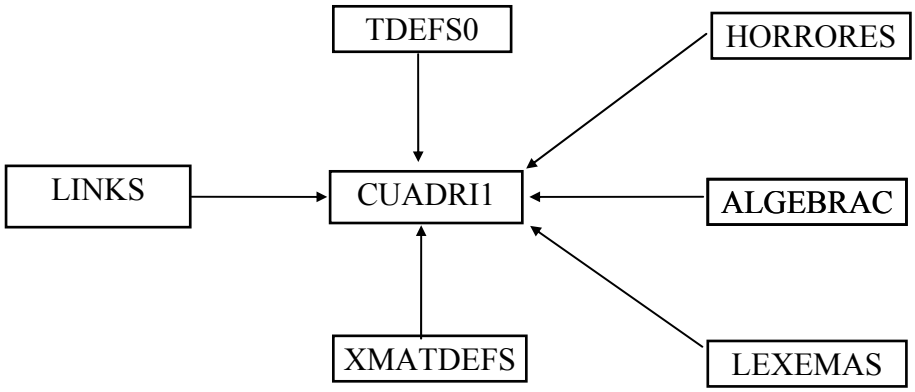
```
                S12:=pc(PBarra(func_BarraPtr(Nodo1-1))^.V,cc(I12)^)^;
                I21:=dc(rc(PBarra(func_BarraPtr(Nodo2-1))^.V,
                                PBarra(func_BarraPtr(Nodo1-1))^.V)^,Z)^;
                S21:=pc(PBarra(func_BarraPtr(Nodo2-1))^.V,cc(I21)^)^;
                I:=I12;
        end
        else
        begin
                if nodo1=0 then
                begin
                        S12:=complex_nulo;
                        I21:=dc(PBarra(func_BarraPtr(Nodo2-1))^.V,Z)^;
                        S21:=pc(PBarra(func_BarraPtr(Nodo2-1))^.V,cc(I21)^)^;
                        I:=I21;
                end
                else
                begin
                        S21:=complex_nulo;
                        I12:=dc(PBarra(func_BarraPtr(Nodo1-1))^.V,Z)^;
                        S12:=pc(PBarra(func_BarraPtr(Nodo1-1))^.V,cc(I12)^)^;
                        I:=I12;
                end;
        end;
        Scon:=sc(S12,S21)^; {Potencia aparente consumida}
end;



constructor TImpedancia.LeerDeFljLetras( var a: TFlujoLetras; var r: string);

var
        res: integer;

begin
        { Nombre ID de la Impedancia }
        if length(r) > 8 then Nombre := Copy(r, 1,8)
        else Nombre:=r;

        { Datos }
        getlexema(r,a);
        Nodo1:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        getlexema(r,a);
        Nodo2:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
```

```
        res:= LeerNComplex(a, Z);
        res:= LeerNReal(a, Imax);
end;

end.
```

1.4.3. UNIDAD CUADRI1

Define el tipo de objetos **TCuadripoloPi**. Un tipo **TCuadripoPi** es una clase de objetos cuyos campos son: el **nombre** del cuadripolo Pi, los nodos entre los cuales se encuentra (**Nodo1, Nodo2, Nodo3)** , los valores **Y13**, **Z12**, **Y23** (números complejos) de las admitancias e impedancia del mismo y la corriente máxima admisible **Imax**. A partir de los métodos implementados un tipo **TCuadripoPi** puede crearse pasándole los valores de cada uno de sus campos usando el constructor Init o leyendo dichos valores de un archivo usando el constructor LeerdeFljLetras. Además utilizando la función PerteneceBarra es posible determinar si una barra determinada es uno de los vértices del cuadripolo.

El esquema básico de la unidad CUADRIPO es el que se muestra a continuación.




```
{+doc
+NOMBRE: Cuadri1
+CREACION: 8/97
+AUTORES: MARIO VIGNOLO
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:  Definición del objeto Cuadripolo.
+PROYECTO: FLUCAR

+REVISION:
+AUTOR:
```

```
+DESCRIPCION:
-doc}

unit cuadri1;

interface
uses
        {$I XObjects},
        Horrores, xMatDefs, Lexemas, AlgebraC, TDEfs0,Barrs2,Links1;




{ Implementacion de los cuadripolos PI.

                Nodo1  ---_ZZZZZZ_--- Nodo2
                           Y       Y
 S12 ->                    Y       Y                    <-S21
                           Y       Y
                 N          ----------       N
}



type
        PCuadripoloPi=^TCuadripoloPi;
        TCuadripoloPi = object(TObject)
                Nombre: TStr8;
                Nodo1, Nodo2, Nodo3: integer;
                Y13, Z12, Y23: complex;
                Imax: NReal;
                constructor Init(
                        xNombre: string;
                        xNod1, xNod2, xNod3: integer;
                        xY13, xZ12, xY23: complex;xImax: NReal);
                constructor LeerDeFljLetras(
                        var a: TFlujoLetras;
                        var r: string);

                function PerteneceBarra(
                        kBarra: integer): boolean;
                procedure TransfS(var S12,S21,Scon,I: complex);
                {S12: potencia aparente entrante por el nodo 1, S21: potencia
```

```
                aparente entrante por el nodo 2, Scon: potencia aparente consumida
                por el cuadripolo, I: I12 si mod(I12)<mod(I21) y I21 si mod(I21)>
                mod(I12), I12: corriente entrante por el nodo 1 e I21 corriente
                entrante por el nodo 2}
        end;


implementation


constructor TCuadripoloPi.Init(
                        xNombre: string;
                        xNod1, xNod2, xNod3: integer;
                        xY13, xZ12, xY23: complex; xImax: NReal);

begin
        Nombre:= Copy(xNombre,1,8);

        Nodo1:= xNod1;
        Nodo2:= xNod2;
        Nodo3:= xNod3;
        Y13:= xY13;
        Z12:= xZ12;
        Y23:= xY23;
        Imax:=xImax;
end;

function TCuadripoloPi.PerteneceBarra( kBarra: integer): boolean;
begin
        PerteneceBarra:=kBarra in [Nodo1, Nodo2, Nodo3];
end;

procedure TCuadripoloPi.TransfS(var S12,S21,Scon,I:complex);

var
        I1,I2,I12,I21: complex;

begin
                I12:= pc(rc(PBarra(Func_BarraPtr(Nodo1-1))^.V,
                PBarra(Func_BarraPtr(Nodo2-1))^.V)^,invc(Z12)^)^;{(V1-V2)/Z12}
                I1:=pc(PBarra(Func_BarraPtr(Nodo1-1))^.V,Y13)^;
                I12:=sc(I12,I1)^;
                S12:=pc(PBarra(Func_BarraPtr(Nodo1-1))^.V,cc(I12)^)^;
                I21:= pc(rc(PBarra(Func_BarraPtr(Nodo2-1))^.V,
                PBarra(Func_BarraPtr(Nodo1-1))^.V)^,invc(Z12)^)^;{(V1-V2)/Z12}
```

```
                I2:=pc(PBarra(Func_BarraPtr(Nodo2-1))^.V,Y23)^;
                I21:=sc(I21,I2)^;
                S21:=pc(PBarra(Func_BarraPtr(Nodo2-1))^.V,cc(I21)^)^;

                Scon:=sc(S12,S21)^; {C lculo de la potencia consumida
                                                        en el cuadripolo}
                if mod1(I12)>mod1(I21) then I:=I12 else I:=I21;
end;

constructor TCuadripoloPi.LeerDeFljLetras( var a: TFlujoLetras; var r: string);
var
        res: integer;

begin
        { Nombre ID de la Impedancia }
        if length(r) > 8 then Nombre := Copy(r, 1,8)
        else Nombre:=r;

        { Datos }
        getlexema(r,a);

        Nodo1:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        getlexema(r,a);
        Nodo2:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        getlexema(r,a);
        Nodo3:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');

        res:= LeerNComplex(a, Y13);
        res:= LeerNComplex(a, Z12);
        res:= LeerNComplex(a, Y23);
        res:= LeerNReal(a,Imax);
end;

end.
```

1.4.4. UNIDAD TRAFOS1

Define el tipo de objetos **TTrafo**. Un tipo **Ttrafo** es una clase que tiene caracteristicas similares a la clase Timpedancia y TcuadripoloPI, cuyos campos son: el **nombre** del trafo, los nodos entre los cuales se encuentra (**Nodo1, Nodo2)** el valor **Zcc** (nú-

mero complejo) de la impedancia de cortocircuito del mismo, la relación de transformación **n,** y la corriente máxima admisible **Imax**.

A continuación mostramos la interfase de la unidad:

```
{+doc
+NOMBRE: Trafos1
+CREACION: 8/97
+AUTORES: MARIO VIGNOLO
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:  Definición de la clase TTrafo.
+PROYECTO: FLUCAR
+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit Trafos1;

interface
uses
        {$I XObjects},
        Horrores, xMatDefs, Lexemas, AlgebraC, TDEfs0,Barrs2,Links1;

type
        PTrafo=^TTrafo;
        TTrafo = object(TObject)
                Nombre: TStr8;
                Nodo1, Nodo2: integer;
                n: NReal;
                Zcc: complex;
                Imax: NReal;
                constructor Init(
                        xNombre: string;
                        xNod1, xNod2: integer;
                        xn: NReal; xZcc:complex; xImax: NReal);
                constructor LeerDeFljLetras( var a: TFlujoLetras; var r: string);
                destructor done; virtual;
                procedure TransfS(var S12,S21,Scon,I: complex);
        end;
```

```
        {
                                        1:n
                                    - - - - -       Zcc
                        Nodo1  ----- -      -      - ----/////---- Nodo2
                                    - -  - -

        }

implementation

constructor TTrafo.Init(xNombre: string; xNod1, xNod2: integer;
                        xn: NReal; xZcc: complex; xImax: NReal);

begin
        Nombre:= Copy(xNombre,1,8);

        Nodo1:= xNod1;
        Nodo2:= xNod2;
        n:=xn;
        Zcc:= xZcc;
        Imax:=xImax;
end;

procedure TTrafo.TransfS(var S12,S21,Scon,I: complex);

var
        I1_2,I1,I2:complex;

begin
        I1_2:=dc(rc(prc(n,PBarra(func_BarraPtr(Nodo1-1))^.V)^,   {(nV1-V2)/Zcc}
                                PBarra(func_BarraPtr(Nodo2-1))^.V)^,Zcc)^;
        I1:=prc(n,I1_2)^;
        S12:=pc(PBarra(func_BarraPtr(Nodo1-1))^.V,cc(I1)^)^;
        I2:=prc(-1,I1_2)^;
        S21:=pc(PBarra(func_BarraPtr(Nodo2-1))^.V,cc(I2)^)^;
        I:=I2;
        Scon:=sc(S12,S21)^; {Potencia aparente consumida}

end;

constructor TTrafo.LeerDeFljLetras( var a: TFlujoLetras; var r: string);

var
        res: integer;
```

```
begin
        { Nombre ID del Trafo}
        if length(r) > 8 then Nombre := Copy(r, 1,8)
        else Nombre:=r;

        { Datos }
        getlexema(r,a);
        Nodo1:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        getlexema(r,a);
        Nodo2:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        res:= LeerNReal(a,n);
        res:= LeerNComplex(a, Zcc);
        res:= LeerNReal(a,Imax);

end;

destructor TTrafo.Done;
begin
{ por ahora nada}
end;

end.
```

1.4.5. UNIDAD REGULADO

La introducción de los reguladores en el software se realizó a partir de la creación de una nueva clase de objetos: **TRegulador**. Esta nueva clase tiene características similares a la clase **TTrafo**.

La clase **TRegulador** así como los métodos que utiliza están definidos en la unidad REGULADO la cual se transcribe a continuación:

```
{+doc
+NOMBRE: Regulado
+CREACION: 9/97
+AUTORES: MARIO VIGNOLO
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:  Definición de la clase Regulador.
+PROYECTO: FLUCAR
+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit Regulado;

interface
uses
        {$I XObjects},
        Horrores, xMatDefs, Lexemas, AlgebraC, TDEfs0,Barrs2,Links1;
type
        PRegulador=^TRegulador;
        TRegulador = object(TObject)
                Nombre: TStr8;
                Nodo1, Nodo2: integer;
                n,nmin,nmax,delta_n: NReal;
                Zcc: complex;
                Imax: NReal;
                constructor Init(
                        xNombre: string;
                        xNod1, xNod2: integer;
                        xn,xnmin,xnmax,xdelta_n: NReal; xZcc:complex; xImax: NReal);

                constructor LeerDeFljLetras( var a: TFlujoLetras; var r: string);
                procedure TransfS(var S12,S21,Scon,I: complex);
        end;
```

```
        {
                                        1:n
                                    - - - -       Zcc
                        Nodo1  ----- -     -     - ----/////---- Nodo2
                                    - - - -

        }

implementation

constructor TRegulador.Init(
                        xNombre: string;
                        xNod1, xNod2: integer;
                        xn,xnmin,xnmax,xdelta_n: NReal; xZcc: complex; xImax:NReal);


begin
        Nombre:= Copy(xNombre,1,8);
        Nodo1:= xNod1;
        Nodo2:= xNod2;
        n:= xn;
        nmin:=xnmin;
        nmax:=xnmax;
        delta_n:=xdelta_n;
        Zcc:= xZcc;
        Imax:=xImax;
end;

procedure TRegulador.TransfS(var S12,S21,Scon,I: complex);

var
        I1_2,I1,I2:complex;

begin
        I1_2:=dc(rc(prc( n,PBarra(func_BarraPtr(Nodo1-1))^.V)^,   {(nV1-V2)/Zcc}
                                PBarra(func_BarraPtr(Nodo2-1))^.V)^,Zcc)^;
        I1:=prc(n,I1_2)^;
        S12:=pc(PBarra(func_BarraPtr(Nodo1-1))^.V,cc(I1)^)^;
        I2:=prc(-1,I1_2)^;
        S21:=pc(PBarra(func_BarraPtr(Nodo2-1))^.V,cc(I2)^)^;
        I:=I2;
        Scon:=sc(S12,S21)^; {Potencia aparente consumida}

end;
```

```
constructor TRegulador.LeerDeFljLetras( var a: TFlujoLetras; var r: string);
var
        res: integer;

begin
        { Nombre ID del Trafo}
        if length(r) > 8 then Nombre := Copy(r, 1,8)
        else Nombre:=r;

        { Datos }
        getlexema(r,a);
        Nodo1:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        getlexema(r,a);
        Nodo2:= Func_IndiceDeNodo(r,res);
        if res <> 0 then error(r+'  nombre no valido');
        res:= LeerNReal(a,n);
        res:= LeerNReal(a,nmin);
        res:= LeerNReal(a,nmax);
        res:= LeerNReal(a,delta_n);
        res:= LeerNComplex(a, Zcc);
        res:= LeerNReal(a,Imax);
end;

end.
```

1.4.6. UNIDAD MATADM1

En la versión anterior del flujo la incorporación de los diferentes elementos vistos anteriormente en la matriz de admitancia se realizaba con el procedimento MATADM. En la nueva versión del flujo, la incorporación de los reguladores se incorpora de manera similar a una impedancia con el procedimiento PonTr, dados que la modificación de la impedancia de cortocircuito para el lado del primario o secundario del regulador se realiza directamente en las ecuaciones de la red.

La unidad MATADM se sustituye por la unidad MATADM1. A continuación se transcribe la unidad MATADM1:

```
{+doc
+NOMBRE:  MATADM1
+CREACION:
+MODIFICACION:  8/01
```

```
+AUTORES:  MARIO VIGNOLO
+REGISTRO:
+TIPO:  Unidad Pascal
+PROPOSITO:  Procedimientos para construir la matriz de admitancias
+PROYECTO: FLUCAR

+REVISION:2001
+AUTOR:A.Costa-C.Olmedo
+DESCRIPCION:Se agregó el procedimiento para colocar reguladores

-doc}

unit matadm;

interface
uses
        XMatDefs,AlgebraC,Barrs2, Impds1, Cuadri1, Trafos1,
        Regulado, TyVs2;

procedure FormarSistema;
{Crea la matriz de admitancias Yki del flujo de carga. Recordar
que Yki=-yki; Ykk=yk+sum(yki), con sum variando i de 1 a NBarras
y i<>k}

procedure PonT( n1, n2: integer; xy: complex; n: NReal);
{Agrega un transformador en la matriz de admitancias}

procedure PonTR( n1, n2: integer; xy: complex; n: NReal);
{Agrega un transformador en la matriz de admitancias}

procedure SacarT( n1, n2: integer; xy: complex; n: NReal);
{Sumo todas las constantes a la matriz de admitancias cambiandoles
el signo. De esta forma si repito consecutivamente con los mismos
par metros PonT y SacarT el resultado final en no haber colocado
ning£n transformador}


implementation

procedure Pon(nEc, nVar: integer; XY: complex );


begin
        if (nEc<> 0)and(nVar<>0) then Admitancias^.acumular(nEc,nVar,XY);
end;
```

```
procedure PonY( n1, n2: integer; xy: complex);
var
        menosy: complex;
begin
        menosy:= prc(-1,xy)^;
        pon(n1,n1,xy);
        pon(n1,n2, menosy);
        pon(n2,n1, menosy);
        pon(n2,n2, xy);
end;

procedure PonT( n1, n2: integer; xy: complex; n: NReal);
var
        xyt,menosy, menosyt: complex;
begin
        menosy:= prc(-1,xy)^;
        xyt:= prc(sqr(n),xy)^;
        menosyt:= prc(n,menosy)^;
        pon(n1,n1,xyt);
        pon(n1,n2, menosyt);
        pon(n2,n1, menosyt);
        pon(n2,n2, xy);
end;

procedure PonTR( n1, n2: integer; xy: complex; n: NReal);
var
        xyt,menosy, menosyt: complex;
begin
        menosy:= prc(-1,xy)^;
        xyt:= prc(sqr(n),xy)^;
        menosyt:= prc(n,menosy)^;
        {pon(n1,n1,xyt);
        pon(n1,n2, menosyt);
        pon(n2,n1, menosyt);
        pon(n2,n2, xy);}
        pon(n1,n1,xy);
        pon(n1,n2, menosy);
        pon(n2,n1, menosy);
        pon(n2,n2, xy);
end;

procedure SacarT( n1, n2: integer; xy: complex; n: NReal);

var
```

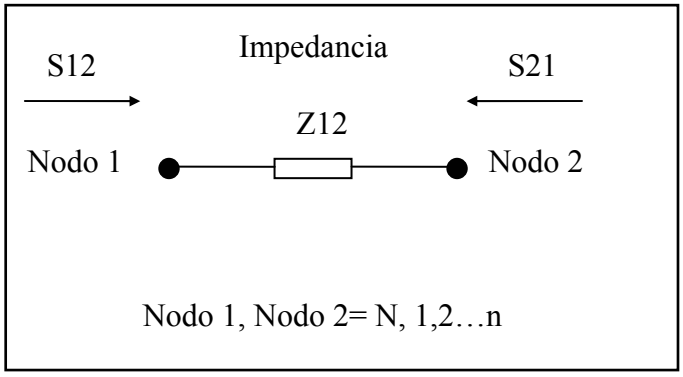
```
        xyt,menosy, menosyt: complex;
begin
        menosy:= prc(-1,xy)^;
        xyt:= prc(sqr(n),xy)^;
        menosyt:= prc(n,menosy)^;
        pon(n1,n1, prc(-1,xyt)^);
        pon(n1,n2, prc(-1,menosyt)^);
        pon(n2,n1, prc(-1,menosyt)^);
        pon(n2,n2, prc(-1,xy)^);
end;



procedure FormarSistema;

var
        k: integer;
        n: NReal;
        N1, N2: TIndice;
        y, z: complex;
        xy: complex;

begin
        { Colocar Impedancias }
        for k:= 1 to NImpedancias do
        begin
                z:= PImpedancia(Impedancias^.At(k-1))^.z;
                y:= prc( 1/mod2(z), cc(z)^)^;
                N1:= PImpedancia(Impedancias^.At(k-1))^.Nodo1;
                N2:= PImpedancia(Impedancias^.At(k-1))^.Nodo2;
                ponY(N1, N2, y);

        end;

        { Colocar CuadripolosPi }
        for k:= 1 to NCuadripolosPi do
        begin
                y:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Y13;
                N1:=PCuadripoloPi(Cuadripolos^.At(k-1))^.Nodo1;
                N2:=PCuadripolopi(Cuadripolos^.At(k-1))^.Nodo3;
                {writeln('valor de y13 ',y.r);    readln;
                writeln('valor de y13 ',y.i); readln;}
                pony(N1, N2, y);
                z:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Z12;
```

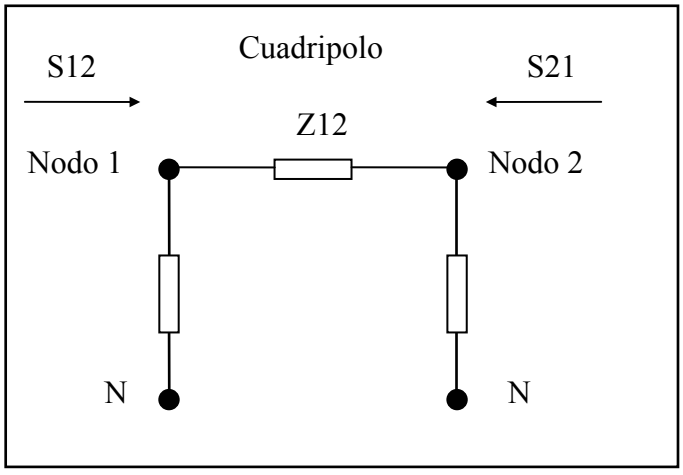
```
                y:= prc( 1/mod2(z), cc(z)^)^;
                N1:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Nodo1;
                N2:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Nodo2;
            {   writeln('valor de y de z ',y.r);    readln;
                writeln('valor de y de z ',y.i); readln;}
                pony(N1, N2, y);
                y:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Y23;
                N1:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Nodo2;
                N2:= PCuadripoloPi(Cuadripolos^.At(k-1))^.Nodo3;
{               writeln('valor de y23 ',y.r);    readln;
                writeln('valor de y23 ',y.i); readln;}
                pony(N1, N2, y);

        end;

        { Colocar Trafos }
        for k:= 1 to NTrafos do
        begin
                z:= PTrafo(Trafos^.At(k-1))^.Zcc;
                y:= prc( 1/mod2(z), cc(z)^)^;
            {   writeln('valor de y ',y.r);    readln;
                writeln('valor de y ',y.i); readln;    }
                N1:= PTrafo(Trafos^.At(k-1))^.Nodo1;
                N2:= PTrafo(Trafos^.At(k-1))^.Nodo2;
                n:= PTrafo(Trafos^.At(k-1))^.n;
                ponT(N1, N2, y, n);
        end;

        { Colocar Reguladores }
        for k:= 1 to NReguladores do
        begin
                z:= PRegulador(Reguladores^.At(k-1))^.Zcc;
                y:= prc( 1/mod2(z), cc(z)^)^;
            {   writeln('valor de y ',y.r);    readln;
                writeln('valor de y ',y.i); readln;}
                N1:= PRegulador(Reguladores^.At(k-1))^.Nodo1;
                N2:= PRegulador(Reguladores^.At(k-1))^.Nodo2;
                n:= PRegulador(Reguladores^.At(k-1))^.n;
                ponTR(N1, N2, y, n);
        end;

        writeln('forme el sistema ok');
end;

end.
```

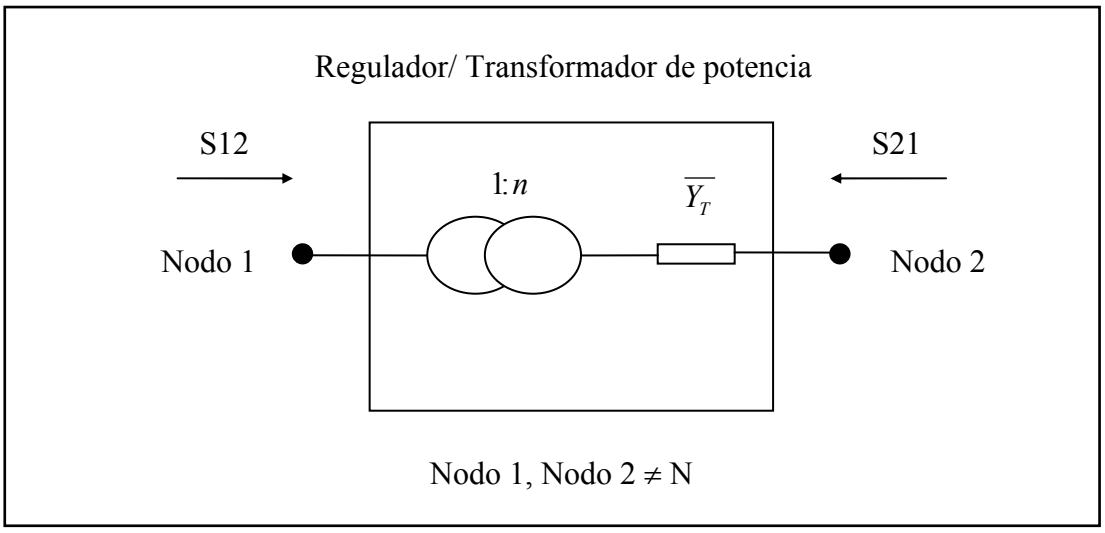
1.5. Cálculo de los flujos de potencia a través de las líneas

El programa de Flujo calcula los flujos de potencia a través de las impedancias, los cuadripolos, los transformadores y los reguladores. El cálculo se realiza a través del procedimiento **TransfS**, el cual está implementado como método de las clases **TImpedancia**, **TCuadripoloPi**, **TTrafo** y **Tregulador**. Este procedimiento utiliza como variables **S12** (potencia aparente que fluye desde el nodo 1 hacia el nodo 2), **S21** (potencia aparente que fluye desde el nodo 2 hacia el nodo 1), **Scon** (potencia aparente consumida en el elemento correspondiente) e **I** (corriente a través del elemento). Para el caso de los reguladores **I** es, de los dos casos posibles (I12 o I21), la de mayor módulo.

Los flujos mencionados son como se muestran en las siguientes figuras:







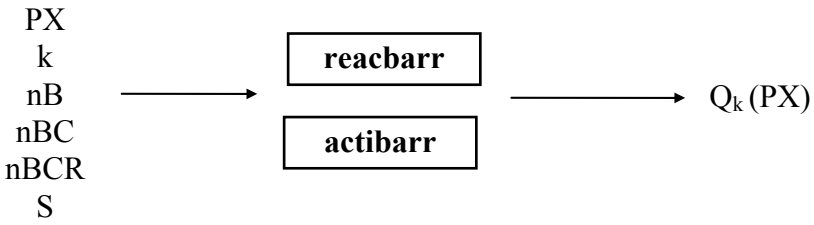
1.6. La función objetivo del flujo de carga

Tanto la función objetivo del flujo de carga como el sistema de ecuaciones lineales asociado a ella se calculaban en la versión anterior del flujo en la unidad FUN1.Esta fue modificada por FUN4, dado que como se vió anteriormente la función objetivo como el cálculo de las derivadas parciales se modifican en la nueva versión del flujo. La función objetivo es evaluada a través del procedimiento **calcularFXv**.al que se le debe pasar el vector de variables Xv. El resultado es devuelto en la variable FXv:

El sistema de ecuaciones calculado a partir de la matriz jacobiana de la función objetivo es calculado por el procedimiento **calcularfunG**. El procedimiento recibe como dato el vector de variables Xv y devuelve el sistema de ecuaciones resultante en la variable SJ:

Además de estos dos procedimientos la unidad FUN4 cuenta con dos funciones más: **reacbarr** y **actibarr** que calculan la potencia entrante reactiva y activa respectivamente de una barra (k) a partir del vector de variables PX, de los datos de las barras (B), y de

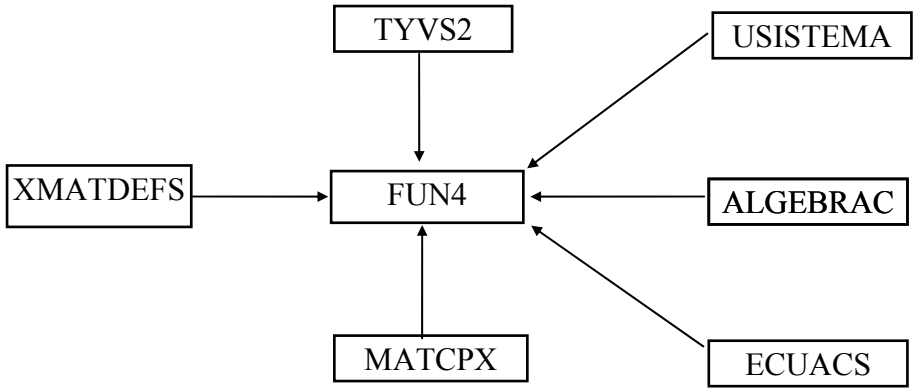
la matriz de admitancias (S). Además se deben dar a las funciones el número de barras (nB), el número de barras de carga (nBC), y el número de barras con regulador (nBCR) como datos:



Además de estas funciones ya existentes se agregó una función booleana auxiliar buscoregulador que devuelve true si encuentra un regulador entre 2 nodos.

Se modificó actibarr y reactibarr para adaptarse a los cambios generados por el nuevo método de Newton Rapshon sacando las tensiones de los datos originales en los casos de los reguladores.

A continuación mostramos un esquema de la unidad y luego el fuente:




```
{+doc
+NOMBRE:  FUN4
+CREACION:  8/97
+MODIFICACION: 2001
+AUTORES:  A.Costa - C.Olmedo
+REGISTRO:
+TIPO:
+PROPOSITO:         C lculo de la fuci¢n objetivo del Flujo de Carga y
                              de su Jacobiano.  Calculo de gradientes.
+PROYECTO:  FLUCAR
```

```
+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

{Opciones de compilación:
DERIVADAS:  Con esta opción calcula la matrices de derivadas DXY y DXYP.
DEB_DERIVADAS:  Con esta opción se puede debuggear el cálculo de las derivadas.
}


unit fun4;

interface
uses
        {$I XObjects},TyVs2,XMatDefs,MatCPX,MatReal,barrs2, usistema,
algebrac,regulado, links1;

procedure calcularFXv(var FXv: PVectComplex; Xv: PVectComplex);
{Calcula la funci¢n objetivo a partir del vector de variables Xv}

procedure calcularfunG(var SJ: PSistema; Xv:PVectComplex);
{Calcula el sistema de ecuaciones lineales que surge de la aproximación
de primer orden de la funci¢n objetivo. Tambien calcula DXY y DGYP,
matrices que contienen las derivadas parciales de las variables inc¢gnita
del flujo respecto de las variables dato del flujo}

{Incorporo la posicion del tap como variable dentro de la matriz como elemento}

function reacbarr(PX:PVectComplex; B:PCollection;
                  S:PSistema; k:TIndice;nB,nBC,nBcR:integer):Nreal;
{Calcula la potencia reactiva entrante al nodo k
  el agregar def_rb ayuda debugeo}

function actibarr(PX:PVectComplex; B:PCollection;
                  S:PSistema; k:TIndice; nB,nBC,nBcR:integer):Nreal;
{Calcula la potencia activa entrante al nodo k}

function buscoregulador(PX:PvectComplex;k,j:integer;var
encontre:boolean;nB,nBC,nBcR:integer):boolean;
{busca reguladores entre los nodos k y j y devuelve true encontre
  el agregar define deb_Brclo da valores de nodos y demas}

implementation
```

```
function buscoregulador(PX:PvectComplex;k,j:integer;var
encontre:boolean;nB,nBC,nBcR:integer):boolean;
var
        tcompleja:complex;
        t:NReal;
        l,nodo1,nodo2:integer;
        encontreuno:boolean;
begin

                encontreuno:=false;
                for l:=1 to nBcR+1 do
                begin
                while (encontreuno=false) and (l<>nBcR+1) do
                        begin
                                {{$DEFINE deb_brclo}
                                nodo2:=PRegulador(Reguladores^.At(l-1))^.Nodo2;
                                nodo1:=PRegulador(Reguladores^.At(l-1))^.Nodo1;
                                if (k=nodo1) and (j=nodo2) then encontreuno:=true;
                                {$ifdef deb_brclo}
                                writeln('procedimiento buscandoregulador');
                                writeln('nodo2 ',nodo2);readln;
                                writeln('nodo1 ',nodo1);readln;
                                writeln('booleana ',encontreuno);
                                writeln('valor de l ',l);readln;
                                {$endif}
                                l:=l+1;
                        end; {del while}
                end;
                {$ifdef deb_brclo}
                writeln('fin procedimiento buscando regulador');
                readln;
                {$endif}
                if encontreuno=true then buscoregulador:=true
                else buscoregulador:=false;
end;


function reacbarr(PX: PVectComplex; B:PCollection;
                        S:PSistema; k:TIndice;nB,nBC,nBcR:integer):Nreal;
                                        {calcula la potencia reactiva
                                        en la barra k a partir de los
                                        valores de tension y las
                                        impedancias de la red}
```

```
var
        modVkc,modVjc,fasekc,fasejc,res,tcompleja:complex;
        sum,gki,bki,modVk,modVj,fasek,fasej,t,bkk,gkk:Nreal;
        j:TIndice;
        n,l:integer;
        Val,uno,dos,tres:boolean;

begin
        {{$define deb_rb}
        sum:=0;
        uno:=false;{variable usada si existe regulador entre k y j}
        dos:=false; {usada si k es regulador }
        tres:=false;  {reserva}
        if (k<=nbc) then
        begin
                PX^.e(modVkc,k+nB-1);
                modVk:=modVkc.r;
        end
        else
        begin
                modVk:=mod1(PBarra(B^.At(k-1))^.V);
                if (k<>nB) then dos:=true;     {el nodo k es barra tipo 4}
        end;
        if k<>nB then
        begin
                PX^.e(fasekc,k);
                fasek:=fasekc.r;
        end
        else
        begin
                fasek:=fase(PBarra(B^.At(k-1))^.V);
        end;
        for j:=1 to NBarras do
        begin
                Val:=S^.Valcoef(res,k,j);
                bki:=res.i;
                gki:=res.r;
                if ( j<=nBC{+nBcR}) then
                begin
                        PX^.e(modVjc,j+nB-1);
                        modVj:=modVjc.r;
                end
                else
                begin
                        modVj:=mod1(PBarra(B^.At(j-1))^.V);
```

```
                end;
                if j<>nB then
                begin
                        PX^.e(fasejc,j);  {2002 ver si sería j o j-1}
                        fasej:=fasejc.r;
                end
                else
                begin
                        fasej:=fase(PBarra(B^.At(j-1))^.V);
                end;
                {$IFDEF DEB_rb}
                writeln('procedimiento potencia reactiva');
                writeln('nodos ',k,j);
                writeln('modvk= ',modvk);
                writeln('modvj= ',modvj);
                writeln('fasek= ',fasek);
                writeln('fasej= ',fasej);
                writeln('bki ',bki);
                writeln('gki ',gki);
                writeln('k es regulador? ',dos);
                writeln('busco regulador ',k,j);
                {$ENDIF}
                if (dos=false) then    {el nodo k es regulador}
begin
                        uno:=buscoregulador(PX,j,k,uno,nB,nBC,nBcR);
                        {$ifdef deb_rb}
                        writeln('kj es regulador? ',uno);
                        {$endif}
                        if (uno=true) then
                        begin
                                Px^.e(tcompleja,j+nB-1); {k por j}
                                t:=tcompleja.r;
                                bki:=bki*t;
                                gki:=gki*t;
                                {$ifdef deb_rb}
                                writeln('t ',t);
                                writeln('bki ',bki);
                                writeln('gki ',gki);
                                readln;
                                {$endif}
                        end;
                end
                else if (dos=true) then
                begin
```

```
                        uno:=buscoregulador(PX,k,j,uno,nB,nBC,nBcR); {de-
vuelve T si reg entre j y k}
                        {$ifdef deb_rb}
                         writeln('existe regulador j y k? ',uno);
                         readln;
                         {$endif}
                        if (uno=true) then
                        begin
                                Px^.e(tcompleja,k+nB-1); {k por j}
                                t:=tcompleja.r;
                                bki:=bki*t;
                                gki:=gki*t;
                                {Val:=S^.Valcoef(res,k,k);
                                bkk:=res.i;
                                gkk:=res.r;}
                                sum:=sum+bki*t*modvk*modvk-
sqr(modvk)*(bki/t);
                                {$ifdef deb_rb}
                                writeln('agregue ',bki*t*sqr(modvk),' y ',-
(bki/t)*sqr(modvk));
                                writeln('t ',t);
                                writeln('bki*t ',bki);
                                writeln('gki*t ',gki);
                                readln;
                                {$endif}
                        end;
                end;
                {$ifdef deb_rb}
                writeln('suma parcial ',sum);
                writeln('mas ',modVk*modVj*(gki*sin(fasek-
                fasej)-bki*cos(fasek-fasej)));
                {$endif}
                sum:=sum+ modVk*modVj*(gki*sin(fasek-
                fasej)-bki*cos(fasek-fasej));
                {$IFDEF deb_rb}
                writeln('suma parcial ',sum);
                writeln('fin de reactibarr');
                readln;
                {$ENDIF}
        end; {end for}
        reacbarr:=sum;
        end;

function actibarr(PX:PVectComplex; B:PCollection;
        S:PSistema; k:TIndice; nB,nBC,nBcR:integer):Nreal;
```

```
                                        {calcula la potencia activa
                                        en la barra k a partir de los
                                        valores de tension y las
                                        impedancias de la red}

var
    modVkc,modVjc,fasekc,fasejc,res,tcompleja:complex;
    sum,gki,bki,modVk,modVj,fasek,fasej,t,bkk,gkk:Nreal;
    j:TIndice;
    n:integer;
    Val,uno,dos,tres:boolean;

    begin
        {{$define deb_ab}
        uno:=false;{variable usada si existe regulador entre k y j}
        dos:=false; {usada si k es regulador }
        tres:=false;  {reserva}

        sum:=0;
        {mantengo original
        if ( k<=nBC+nBcR) then}
        if (k<=nbc) then
        begin
            PX^.e(modVkc,k+nB-1);
            modVk:=modVkc.r;
        end
        else
        begin
            modVk:=mod1(PBarra(B^.At(k-1))^.V);
            if k<>nB then dos:=true;
        end;
        if k<>nB then
        begin
            PX^.e(fasekc,k);
            fasek:=fasekc.r;
        end
        else
        begin
            fasek:=fase(PBarra(B^.At(k-1))^.V);
        end;
        for j:=1 to NBarras do
        begin
            Val:=S^.Valcoef(res,k,j);
            bki:=res.i;
            gki:=res.r;
```

```
                    {if ( j<=nBC+nBcR) then}
                    if (j<=nbc) then
                    begin
                            PX^.e(modVjc,j+nB-1);  {modVj:=X(j+nB-1)}
                            modVj:=modVjc.r;
                    end
                    else
                    begin
                            modVj:=mod1(PBarra(B^.At(j-1))^.V);
                    end;
                    if j<>nB then
                    begin
                            PX^.e(fasejc,j);
                            fasej:=fasejc.r;
                    end
                    else
                    begin
                            fasej:=fase(PBarra(B^.At(j-1))^.V);
                    end;

                    {$IFDEF DEB_ab}
                    writeln('procedimiento potencia activa');
                    writeln('nodos ',k,j);
                    writeln('modvk= ',modvk);
                    writeln('modvj= ',modvj);
                    writeln('fasek= ',fasek);
                    writeln('fasej= ',fasej);
                    writeln('bki ',bki);
                    writeln('gki ',gki);
                    writeln('k es regulador? ',dos);
                    writeln('busco regulador ',k,j);
                    {$ENDIF}
                    if (dos=false) then    {el nodo k es regulador}
                    begin
                            uno:=buscoregulador(PX,j,k,uno,nB,nBC,nBcR);
{devuelve T si reg entre k y j}
                            {$ifdef deb_ab}
                            writeln('kj es regulador? ',uno);
                            {$endif}
                            if (uno=true) then
                            begin
                                    Px^.e(tcompleja,j+nB-1);  {k por j}
                                    t:=tcompleja.r;
                                    bki:=bki*t;
                                    gki:=gki*t;
```

```
                                {$ifdef deb_ab}
                                writeln('t ',t);
                                writeln('bki*t ',bki);
                                writeln('gki*t ',gki);
                                readln;
                                {$endif}
                        end;
                end
                else if (dos=true) then
                begin
                        uno:=buscoregulador(PX,k,j,uno,nB,nBC,nBcR);
{devuelve T si reg entre j y k}
                        {$ifdef deb_ab}
                        writeln('existe regulador j y k? ',uno);
                        readln;
                        {$endif}
                        if (uno=true) then
                        begin
                                Px^.e(tcompleja,k+nB-1); {k por j}
                                t:=tcompleja.r;
                                bki:=bki*t;
                                gki:=gki*t;
                                sum:=sum-gki*t*modvk*modvk+modvk*gki/t;
                                {$ifdef deb_ab}
                                writeln('agregue ',-gki*t*sqr(modvk),' y
',gki*modvk/t);
                                writeln('t ',t);
                                writeln('bki ',bki);
                                writeln('gki ',gki);
                                writeln('gkk ',gkk);
                                readln;
                                {$endif}
                        end;
                end;
                {$ifdef deb_ab}
                writeln('suma parcial ',sum);
                writeln('mas ',modVk*modVj*(gki*cos(fasek-
                fasej)+bki*sin(fasek-fasej)));
                {$endif}
                sum:=sum+modVk*modVj*(gki*
                cos(fasek-fasej)+bki*
                sin(fasek-fasej));
                {$IFDEF deb_ab}
                writeln('suma parcial ',sum);
                writeln('fin de actibarr');
```

```
                readln;
                {$ENDIF}
            end;{end for}
            actibarr:=sum;
        end;


procedure calcularFXv(var FXv: PVectComplex; Xv: PVectComplex);

        var
            k:integer;
            difactk,difreack:complex;
            B: PCollection;
            S: PSistema;
            nB,nBC,nBcR:word;

        begin
            {{$define deb_fo}
            B:=BARRASORDENADAS;
            S:=Admitancias;
            nB:=NBarras;
            nBC:=NBarrasdeCarga;
            nBcR:=NBarrasconregulador;

            for k:=1 to (nB-1) do
            begin
                difactk.r:=PBarra(B^.At(k-1))^.S.r-
actibarr(Xv,B,S,k,nB,nBC,nBcR);
                {$IFDEF DEB_fo}
                writeln('procedimiento calculo fxv');
                writeln('valor de k ',k);
                writeln('Pbarra ',PBarra(B^.At(k-1))^.S.r);readln;
                writeln('actibarr ',actibarr(Xv,B,S,k,nB,nBC,nBcR));readln;
                writeln;
                {$ENDIF}
                difactk.i:=0;
                FXv^.pon_e(k,difactk);
            end;
            for k:=1 to (nBC+nBcR) do
            begin
                difreack.r:=PBarra(B^.At(k-1))^.S.i-
reacbarr(Xv,B,S,k,nB,nBC,nBcR);
                {$IFDEF DEB_fo}
                writeln('procedimiento calculo fxv');
                writeln('valor de k ',k);
```

```
                writeln('Qbarra ',PBarra(B^.At(k-1))^.S.i);readln;
                writeln('reacbarr ',reacbarr(Xv,B,S,k,nB,nBC,nBcR));readln;
                writeln;
                {$ENDIF}
                difreack.i:=0;
                FXv^.pon_e((k+nB-1),difreack);
        end;
        {$IFDEF DEB_fo}
        writeln('fin procedimiento calculo fxv');
        {$endif}
    end;

procedure calcularfunG(var SJ:PSistema; Xv:PVectComplex);

    var
        k,i:TIndice;
        kaux,iaux,nJ,ind,m,l,nodo1,nodo2,barraconregulador,resultado:integer;

    Jki,DXYki,DGYPki,gkiJ,bkiJ,bikj1,gikj1,difactk,difreack,modVk,modVinuevo,
modVi,fasek,fasei,faseinueva,t,q:NReal;

    Jkic,ctek,modVkc,modVic,fasekc,faseic,modVinuevoc,faseinuevac,modVkcaux,
modVicaux,res,aux,tcompleja:complex;
        nB,nBC,nBcR:word;
        Val, esinvertible,encontreuno,kestipo4,iestipo4,diag: boolean;
        f:text;
        s:string;

    begin
        nB:=NBarras;
        nBC:=NBarrasdeCarga;
        nBcR:=NBarrasconregulador;

        {{$define deb_fg}
        for k:=1 to nB-1+nBC+nBcR do {recorro las ecuaciones
                                        una a una}
        begin
        for i:=1 to nB-1+nBC+nBcR do {para cada ecuacion
                                        recorro las variables}
        begin
            {segun la posicion en la matriz J asigno el
            valor de la derivada que corresponda}

            {$ifdef deb_fg}
            writeln('valor de k ',k);
```

```
                writeln('valor de i ',i);
                readln;
                {$endif}
                {Derivadas parciales dPk/dthetai}
                if (k<=nB-1) and (i<=nB-1) then
                 begin
                        Val:=Admitancias^.Valcoef(res,k,i);
                        bkiJ:=res.i;
                        gkiJ:=res.r;
                        {$ifdef deb_fg}
                        writeln('bkij ',bkiJ);writeln('gkij ',gkiJ);
                        readln;
                        {$endif}
                        if (k<=nBc) then
                        begin
                                Xv^.e(modVkc,k+nB-1);
                                modVk:=modVkc.r;
                                {$ifdef deb_fg}
                                writeln('sin regulador modvk ',modvk);
                                readln;
                                {$endif}
                        end
                        else {if ((k>nBc) and (k<=nBC+nBCR))
                        then  }
                        begin

modVk:=mod1(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                                {$ifdef deb_fg}
                                writeln('con regulador modvk ',modvk);
                                readln;
                                {$endif}

                        end;
                        if k<>nB then
                        begin
                                Xv^.e(fasekc,k);
                                fasek:=fasekc.r;
                                {$ifdef deb_fg}
                                writeln('fasek ',fasek);
                                readln;
                                {$endif}
                        end
                        else
                        begin
```

```
fasek:=fase(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                                    {$ifdef deb_fg}
                                    writeln('fasek nodo referencia ',fasek);readln;
                                  {$endif}
                                end;
                                {selecciono segun i}
                                if (i<=nBc) then
                                begin
                                    Xv^.e(modVic,i+nB-1);
                                    modVi:=modVic.r;
                                    {$ifdef deb_fg}
                                    writeln(' mod vi sin regulador ',modvi);readln;
                                    {$endif}
                                end
                                else
                                begin

modVi:=mod1(PBarra(BARRASORDENADAS^.At(i-1))^.V);
                                    {$ifdef deb_fg}
                                    writeln(' mod vi con regulador ',modvi);
                                    readln;
                                    {$endif}
                                end;
                                if i<>nB then
                                begin
                                    Xv^.e(faseic,i);
                                    fasei:=faseic.r;
                                    {$ifdef deb_fg}
                                    writeln('fasei ',fasei);
                                    readln;
                                    {$endif}
                                end
                                else
                                begin
                                    fasei:=fase(PBarra(BARRASORDENADAS^.At(i-
1))^.V);
                                    {$ifdef deb_fg}
                                    writeln('fasei nodo referencia ',fasei);
                                    readln;
                                    {$endif}
                                end;

                                 encontreuno:=false;
                                    for l:=1 to nBcR+1 do
```

```
                                        begin
                                                {$ifdef deb_fg}
                                                writeln('valor de l ',l);readln;
                                                {$endif}
                                                while (encontreuno=false) and
(l<>nBcR+1) do
                                                begin

        nodo2:=PRegulador(Reguladores^.At(l-1))^.Nodo2;

        nodo1:=PRegulador(Reguladores^.At(l-1))^.Nodo1;
                                                        if i=nodo1 then encontreuno:=true;
                                                        {$ifdef deb_fg}
                                                        writeln('nodo2 ',nodo2);readln;
                                                        writeln('nodo1 ',nodo1);readln;
                                                        writeln('booleana ',encontreuno);
                                                        {$endif}
                                                        l:=l+1;
                                                end; {del while}
                                                {$ifdef deb_fg}
                                                if encontreuno=true then
                                                begin
                                                   writeln('existe regulador entre nodo
',nodo1,' y ',nodo2);readln;
                                                end;
                                                {$endif}
                                        end;
                if (k=i)then
                                begin
                                        {$ifdef deb_fg}
                                        writeln('Pase por k=i dP/dteta');
                                        writeln('bkij ',bkij);writeln('modvk ',modvk);
                                        writeln('modvk cuadrado ',sqr(modvk));
                                        writeln('pot
reactiva',reacbarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR));
                                        {$endif}

                                        Jki:=-bkiJ*sqr(modVk)-

        reacbarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                                        DXYki:=Jki;


                                        {$ifdef deb_fg}
                                        writeln('jki (k=i) ',jki);readln;
```

```
                                writeln('dxyki ',dxyki);readln;
                                {$endif}
                        end
                        else
                          begin
                                {$ifdef deb_fg}
                                writeln('Pase por k distinto i dP/dteta');
                              {$endif}

                              if ((k=nodo1) and (i=nodo2)) or ((i=nodo1)
and (k=nodo2)) then
                              begin
                                  Xv^.e(tcompleja,nodo1+nB-1);
                                  t:=tcompleja.r;
                                  bkiJ:=bkiJ*t;
                                  gkiJ:=gkiJ*t;
                                  {$ifdef deb_fg}
                                  writeln('valor de bkij*t',bkiJ);
                                  writeln('valor de gkiJ*t',gkiJ);
                                  {$endif}
                              end;

                                 Jki:=modVk*
                                 modVi*(gkiJ*
                                 sin(fasek
                                 -fasei)-bkiJ*
                                 cos(fasek-
                                 fasei));
                                 DXYki:=Jki;
                                 {$ifdef deb_fg}
                                 writeln('jki (k<>i) ',jki);readln;
                                 writeln('dxyki ',dxyki);readln;
                                 {$endif}
                          end;{else}
                  end;

          {Derivadas parciales dQk/dthetai}
                if (k>nB-1) and (i<=nB-1) then
                        begin
                                kaux:=k;
                                k:=k-(nB-1);
                                Val:=Admitancias^.Valcoef(res,k,i);
                                bkiJ:=res.i;
                                gkiJ:=res.r;
                                {$ifdef deb_fg}
```

```
                                writeln('bkij ',bkiJ);writeln('gkij ',gkiJ);readln;
                              {$endif}
                                {selecciono la extraccion de datos}
                                {seleccion segun k}
                                if (k<=nBc) then
                                begin
                                        Xv^.e(modVkc,k+nB-1);
                                        {modVk:=X(k+nB-1).r}
                                        modVk:=modVkc.r;
                                        {$ifdef deb_fg}
                                        writeln('modvk sin regulador
',modvk);readln;
                                        {$endif}
                                end
                                else
                                begin

        modVk:=mod1(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                                        {$ifdef deb_fg}
                                        writeln('modvk con regulador
',modvk);readln;
                                        {$endif}
                                end;
                                Xv^.e(fasekc,k);
                                fasek:=fasekc.r;  {no importa poner condicion if
porque
                                                  la variable llegaría a lo sumo a
nbc+nbcR
                                                  por lo que siempre saco la fase de
aca}
                                {$ifdef deb_Fg}
                                writeln('fasek ',fasek);readln;
                                {$endif}

                                {seleccion segun i}
                                if (i<=nBC) then
                                begin
                                        Xv^.e(modVic,i+nB-1);
                                        modVi:=modVic.r;
                                        {$ifdef deb_fg}
                                        writeln('modvi sin regulador
',modvi);readln;
                                    {$endif}
                                end
                                else
```

```
                                        begin

        modVi:=mod1(PBarra(BARRASORDENADAS^.At(i-1))^.V);
                                                {$ifdef deb_fg}
                                                writeln('modvi con regulador
',modvi);readln;
                                            {$endif}
                                        end;
                                        if i<>nB then
                                        begin
                                                Xv^.e(faseic,i);
                                                fasei:=faseic.r;
                                                {$ifdef deb_fg}
                                                writeln('fasei ',fasei);readln;
                                            {$endif}
                                        end
                                        else
                                        begin

        fasei:=fase(PBarra(BARRASORDENADAS^.At(i-1))^.V);
                                                {$ifdef deb_fg}
                                                writeln('fasei de referencia ',fasei);readln;
                                                {$endif}

                                        end;

                                    encontreuno:=false;
                                        for l:=1 to nBcR+1 do
                                        begin
                                                {$ifdef deb_fg}
                                                writeln('valor de l ',l);readln;
                                                {$endif}
                                                while (encontreuno=false) and
(l<>nBcR+1) do
                                                begin

        nodo2:=PRegulador(Reguladores^.At(l-1))^.Nodo2;

        nodo1:=PRegulador(Reguladores^.At(l-1))^.Nodo1;
                                                        if i=nodo1 then encontreuno:=true;
                                                        {$ifdef deb_fg}
                                                        writeln('nodo2 ',nodo2);readln;
                                                        writeln('nodo1 ',nodo1);readln;
                                                        writeln('booleana ',encontreuno);
                                                        {$endif}
```

```
                                        l:=l+1;
                                    end; {del while}
                                    {$ifdef deb_fg}
                                    if encontreuno=true then
                                    begin
                                       writeln('existe regulador entre nodo
',nodo1,' y ',nodo2);readln;
                                    end;
                                    {$endif}
                                end;

                                if (k=i) then
                                begin
                                    {$ifdef deb_fg}
                                    writeln('Pase por dQ/dteta k=i');
                                  {$endif}
                                    Jki:=-gkiJ*sqr(modVk)+

        actibarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                                    DXYki:=Jki;
                                    {$ifdef deb_fg}
                                    writeln('jki ',jki);readln;
                                    writeln('dxyki ',dxyki);readln;
                                    {$endif}
                                end
                                else
                                begin
                                    {$ifdef deb_fg}
                                    writeln('Pase por dQ/dteta k distinto i');
                                    {$endif}
                                    if ((k=nodo1) and (i=nodo2)) or ((k=nodo2)
and (i=nodo1)) then
                                    begin
                                      Xv^.e(tcompleja,nodo1+nB-1);
                                      t:=tcompleja.r;
                                      bkiJ:=bkiJ*t;
                                      gkiJ:=gkiJ*t;
                                      {$ifdef deb_fg}
                                      writeln('valor de bkij*t',bkiJ);
                                      writeln('valor de gkiJ*t',gkiJ);
                                      {$endif}
                                    end;

                                    Jki:=-modVk*
                                    modVi*(gkiJ*
```

```
                                        cos(fasek-
                                        fasei)+bkiJ*
                                        sin(fasek-
                                        fasei));
                                        DXYki:=Jki;
                                        {$ifdef deb_fg}
                                        writeln('jki ',jki);readln;
                                        writeln('dxyki ',dxyki);readln;
                                        {$endif}
                                end;{else}
                                k:=kaux;
                        end;

                {Derivadas parciales dPk/dVi y dPk/dti*ti ?}
                if (k<=nB-1) and (i>nB-1) then
                        begin
                                iaux:=i;
                                i:=i-(nB-1);
                                Val:=Admitancias^.Valcoef(res,k,i);
                                bkiJ:=res.i;
                                gkiJ:=res.r;
                                {$ifdef deb_fg}
                                writeln('bkij ',bkij);writeln('gkij ',gkij);readln;
                                {$endif}

                                {selecciono la extraccion de datos}
                                {seleccion segun k}
                                if (k<=nBC) then
                                begin
                                        Xv^.e(modVkc,k+nB-1);
                                        modVk:=modVkc.r;
                                        {$ifdef deb_fg}
                                        writeln('modvk sin regulador
',modvk);readln;
                                        {$endif}
                                end
                                else
                                begin

        modVk:=mod1(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                                        {$ifdef deb_fg}
                                        writeln('modvk con regulador
',modvk);readln;
                                    {$endif}
                                end;
```

```
                                    if (k<>nB) then
                                    begin
                                            Xv^.e(fasekc,k);
                                            fasek:=fasekc.r;
                                            {$ifdef deb_fg}
                                            writeln('fasek ',fasek);readln;
                                            {$endif}
                                    end
                                    else
                                    begin

        fasek:=fase(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                                            {$ifdef deb_fg}
                                            writeln('fasek de referencia ',fasek);readln;
                                            {$endif}
                                    end;
                                    {seleccion segun i}
                                    if (i<=nBc) then
                                    begin
                                            Xv^.e(modVic,i+nB-1);
                                            modVi:=modVic.r;
                                            {$ifdef deb_fg}
                                            writeln('modvi sin regulador
',modvi);readln;
                                        {$endif}
                                    end
                                    else
                                    begin

        modVi:=mod1(PBarra(BARRASORDENADAS^.At(i-1))^.V);
                                            {$ifdef deb_fg}
                                            writeln('modvi referencia ',modvi);readln;
                                            {$endif}
                                    end;
                                    Xv^.e(faseic,i);
                                    fasei:=faseic.r; {i llega a nbc+nbcr asi que esta
bien                                                sacar la fase de Xv. El nodo de
referencia
                                                    no esta incluido}
                                    {$ifdef deb_fg}
                                    writeln('fasei ',fasei);
                                    readln;
                                {$endif}
                                    encontreuno:=false;
                                    for l:=1 to nBcR+1 do
```

```
                                    begin
                                        {$ifdef deb_fg}
                                        writeln('valor de l ',l);readln;
                                        {$endif}
                                        while (encontreuno=false) and
(l<>nBcR+1) do
                                        begin

        nodo2:=PRegulador(Reguladores^.At(l-1))^.Nodo2;

        nodo1:=PRegulador(Reguladores^.At(l-1))^.Nodo1;
                                                if i=nodo1 then encontreuno:=true;
                                                {$ifdef deb_fg}
                                                writeln('nodo2 ',nodo2);readln;
                                                writeln('nodo1 ',nodo1);readln;
                                                writeln('booleana ',encontreuno);
                                                {$endif}
                                                l:=l+1;
                                        end; {del while}
                                        {$ifdef deb_fg}
                                        if encontreuno=true then
                                        begin
                                           writeln('existe regulador entre nodo
',nodo1,' y ',nodo2);readln;
                                        end;
                                        {$endif}
                                    end;


                                  {si en la diagonal y no es nodo tipo 4 el k del jkk}
                                    if (k=i) and (encontreuno=false) then
                                    begin
                                        {$ifdef deb_fg}
                                      {writeln('Pase por dP/dV k=i');}
                                      {$endif}
                                        Jki:=gkiJ*sqr(modVk)+

        actibarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                                        DXYki:=Jki/(modVk);
                                        {$ifdef deb_fg}

        q:=actibarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                                        writeln('jki ',jki);readln;
                                        writeln('pot activa ',q);readln;
                                        writeln('dxyki ',dxyki);readln;
```

```
                                        {$endif}
                                end
                                else if (k=i) and (encontreuno=true) then
                                begin
                                                {$ifdef deb_fg}
                                                writeln('Pase por dP/dt*t k=i');
                                                {$endif}
                                                Xv^.e(tcompleja,nodo1+nB-1);
                                                t:=tcompleja.r;
                                                {$ifdef deb_fg}
                                                writeln('valor del tap ',t);readln;
                                              {$endif}

        Val:=Admitancias^.Valcoef(res,nodo1,nodo2);
                                                bikJ1:=res.i;
                                                gikJ1:=res.r;
                                                {$ifdef deb_fg}
                                                writeln('bikj ',bikj1);writeln('gikj1
',gikj1);readln;
                                                {$endif}

                                                Xv^.e(modVinuevoc,nodo2+nB-1);
                                                modVinuevo:=modvinuevoc.r;
                                                Xv^.e(faseinuevac,nodo2);
                                                faseinueva:=faseinuevac.r;
                                                {$ifdef deb_fg}
                                                writeln('modvk ',modvk,fasek);
                                                writeln('modvi
',modvinuevo,faseinueva);
                                                {$endif}

                                                Jki:=sqr(t)*gikJ1*sqr(modVi)*(-2)+
                                                modVinuevo*modVk*t*(gikJ1*
                                                cos(fasek-faseinueva)+bikJ1*
                                                sin(fasek-faseinueva));
                                                DXYki:=Jki/t;{modvk}
                                                {$ifdef deb_fg}
                                                writeln('jki ',jki);readln;
                                                writeln('dxyki ',dxyki);readln;
                                                {$endif}
                                end
                                else if ((encontreuno=true) and (k=nodo2)) or
((i=nodo2) and (k=nodo1))
                                then{ (i=nodo1)}{ver condicion}
                                begin
```

```
                                        {$ifdef deb_fg}
                                        writeln('Pase por dP/dt*t k distinto i');
                                    {$endif}
                                        Xv^.e(tcompleja,nodo1+nB-1);
                                        t:=tcompleja.r;
                                        {$ifdef deb_fg}
                                        writeln('valor del tap ',t);readln;
                                        {$endif}

        Val:=Admitancias^.Valcoef(res,nodo2,nodo1);
                                        bikJ1:=res.i;
                                        gikJ1:=res.r;
                                        {$ifdef deb_g}
                                        writeln('bikj ',bikj1);writeln('gikj1
',gikj1);readln;
                                    {$endif}

                                        Jki:=t*modVk*modVi*(gikJ1*
                                        cos(fasek-fasei)+bikJ1*
                                        sin(fasek-fasei));
                                        {DXYki:=Jki/(modVi);}
                                        DXYki:=Jki/t;{modvi}
                                        {$ifdef deb_fg}
                                        writeln('jki ',jki);readln;
                                        writeln('dxyki ',dxyki);readln;
                                        {$endif}
                                end
                                else
                                begin
                                        {$ifdef deb_fg}
                                        writeln('Pase por dP/dV k distinto i');
                                    {$endif}
                                        Jki:=modVK*modVi*(gkiJ*
                                        cos(fasek-fasei)+bkiJ*
                                        sin(fasek-fasei));
                                        DXYki:=Jki/(modVi);
                                        {$ifdef deb_fg}
                                        writeln('jki ',jki);readln;
                                        writeln('dxyki ',dxyki);readln;
                                        {$endif}
                                 end;


                            i:=iaux;
```

```
                end;


            {Derivadas parciales dQk/dVi - dQk/dti*ti}
            if (k>nB-1) and (i>nB-1) then
                begin
                    iaux:=i;
                    kaux:=k;
                    i:=i-(nB-1);
                    k:=k-(nB-1);
                    Val:=Admitancias^.Valcoef(res,k,i);
                    bkiJ:=res.i;
                    gkiJ:=res.r;
                {$ifdef deb_fg}
                    writeln('bkij ',bkij);writeln('gkij ',gkij);readln;
                {$endif}
                    {selecciono extraccion de datos}
                    {selecciono segun k}
                    if (k<=nBC) then
                    begin
                        Xv^.e(modVkc,k+nB-1);
                        modVk:=modVkc.r;
                        {$ifdef deb_fg}
                        writeln('vk sin regulador ',modvk);readln;
                        {$endif}
                    end
                    else
                    begin

modVk:=mod1(PBarra(BARRASORDENADAS^.At(k-1))^.V);
                        {$ifdef deb_fg}
                            writeln('vk con regulador ',modvk);readln;
                        {$endif}
                    end;

                    Xv^.e(fasekc,k);
                    fasek:=fasekc.r;

                {$ifdef deb_fg}
                    writeln('fasek ',fasek);readln;
                    {$endif}
                    {selecciono segun i}

                    if (i<=nBC) then
                    begin
```

```
                                        Xv^.e(modVic,i+nB-1);
                                        modVi:=modVic.r;
                                        {$ifdef deb_fg}
                                        writeln('vi sin regulador ',modvi);readln;
                                        {$endif}
                                    end
                                    else
                                    begin

        modVi:=mod1(PBarra(BARRASORDENADAS^.At(i-1))^.V);
                                        {$ifdef deb_fg}
                                        writeln('vi con regulador ',modvi);readln;
                                        {$endif}
                                    end;

                                    Xv^.e(faseic,i);
                                    fasei:=faseic.r;
                                {$ifdef deb_fg}
                                    writeln('fasei ',fasei);readln;
                                    {$endif}



                                    encontreuno:=false;
                                    for l:=1 to nBcR+1 do
                                    begin
                                        {$ifdef deb_fg}
                                        writeln('valor de l ',l);readln;
                                        {$endif}
                                        while (encontreuno=false) and (l<nBcR+1)
do
                                        begin

        nodo2:=PRegulador(Reguladores^.At(l-1))^.Nodo2;

        nodo1:=PRegulador(Reguladores^.At(l-1))^.Nodo1;
                                            if i=nodo1 { k=nodo2} then
encontreuno:=true;
                                        {$ifdef deb_Fg}
                                            writeln('nodo2 ',nodo2);readln;
                                            writeln('nodo1 ',nodo1);readln;
                                            writeln('booleana
',encontreuno);readln;
                                        {$endif}
                                            l:=l+1;
```

```
                                        end; {del while}
                                        {$ifdef deb_fg}
                                        if encontreuno=true then
                                        begin
                                        writeln('encontre regulador entre ',nodo1,' y
',nodo2);readln;
                                        end;
                                        {$endif}
                                end;



                                if (k=i) and (encontreuno=false)then
                                begin
                                        {$ifdef deb_fg}
                                        writeln('PAse por k=i dQ/dV');
                                        {$endif}
                                        Jki:=-bkiJ*sqr(modVk)+

        reacbarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                                        DXYki:=Jki/(modVk);
                                        {$ifdef deb_fg}
                                        writeln('pot reactiva
',reacbarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR));
                                        writeln('Jki ',jki);readln;
                                        {$endif}
                                end
                                else if (k=i) and (encontreuno=true) then
                                begin
                                        {$ifdef deb_fg}
                                           writeln('PAse por k=i dQ/dt*t');
                                           {$endif}
                                           Xv^.e(tcompleja,nodo1+nB-1);
                                           t:=tcompleja.r;
                                           {$ifdef deb_fg}
                                           writeln('valor del regulador ',t);
                                           {$endif}

        Val:=Admitancias^.Valcoef(res,nodo1,nodo2);
                                           bikJ1:=res.i;
                                           gikJ1:=res.r;
                                           {$ifdef deb_fg}
                                           writeln('bikj1 ',bikj1);writeln('gikj1
',gikj1);
                                           {$endif}
```

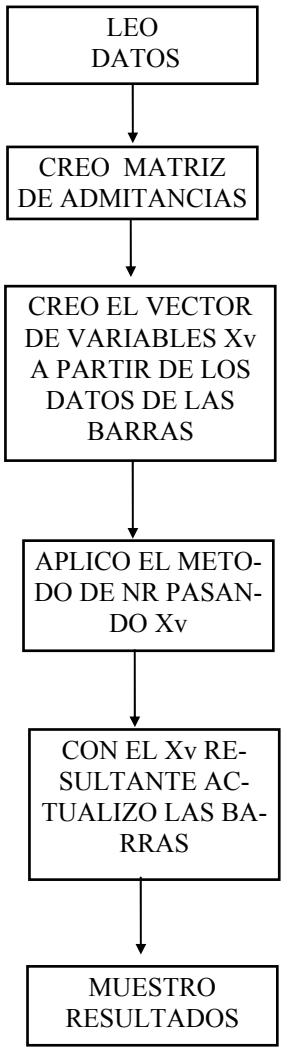
```
                                        Xv^.e(modVinuevoc,nodo2+nB-1);
                                        modVinuevo:=modvinuevoc.r;

                                        Xv^.e(faseinuevac,nodo2);
                                        faseinueva:=faseinuevac.r;
                                        {$ifdef deb_fg}
                                        writeln('modvk ',modvk,fasek);
                                        writeln('modvi
',modvinuevo,faseinueva);
                                        {$endif}


                                        Jki:=sqr(t)*bikJ1*sqr(modVk)*(2)+
                                        modVinuevo*modVk*t*(gikJ1*
                                        sin(fasek-faseinueva)-bikJ1*
                                        cos(fasek-faseinueva));
                                        DXYki:=Jki/t;{modvk}
                                    {$ifdef deb_fg}
                                        writeln('jki ',jki);readln;
                                        writeln('dxyki ',dxyki);readln;
                                        {$endif}

                                    { por si me equivoco 21/11
                                        Jki:=sqr(t)*bkiJ*sqr(modVi)*2+
                                        modVi*modVk*t*(gkiJ*
                                        sin(fasek-fasei)-bkiJ*
                                        cos(fasek-fasei));
                                        DXYki:=Jki/(modVk); }
                                end
                                else if ((encontreuno=true) and (k=nodo2)) or
((i=nodo2) and (k=nodo1))
                                then {ver condicion}
                                begin
                                        {$ifdef deb_fg}
                                        writeln('Pase por k distinto i dQ/dt*t');
                                        {$endif}
                                        Xv^.e(tcompleja,nodo1+nB-1);
                                        t:=tcompleja.r;
                                    {$ifdef deb_fg}
                                        writeln('valor de regulador ',t);readln;
                                    {$endif}

                            Val:=Admitancias^.Valcoef(res,nodo2,nodo1);
                                        bikJ1:=res.i;
```

```
                                    gikJ1:=res.r;
                                    Jki:=t*modVk*modVi*(gikJ1*
                                    sin(fasek-fasei)-bikJ1*
                                    cos(fasek-fasei));
                                    DXYki:=Jki/t; {modvi}
                                {$ifdef deb_fg}
                                    writeln('jki ',jki);readln;
                                    writeln('dxyki ',dxyki);readln;
                                    {$endif}
                            end
                            else
                             begin
                                    {$ifdef deb_fg}
                                    writeln('Pase por k distinto i dQ/dV');
                                    {$endif}
                                    Jki:=modVk*
                                    modVi*(gkiJ*
                                    sin(fasek-fasei)-bkiJ*
                                    cos(fasek-fasei));
                                    DXYki:=Jki/(modVi);
                                {$ifdef deb_fg}
                                    writeln('jki ',jki);readln;
                                    writeln('dxyki ',dxyki);readln;
                                    {$endif}

                                    {luego cambio los jki si hay regulador}
                             end;{else}
                    i:=iaux;
                    k:=kaux;
                    end;

            Jkic.r:=Jki;
            Jkic.i:=0;

            SJ^.Acumular(k,i,Jkic); {agrego el coeficiente i
                                        en la ecuacion k}
            {$IFDEF DERIVADAS}

            {Genero la matriz SJ en formato TMatR}
            DXY^.pon_e(k,i,DXYki);

            {$ENDIF}


end; {end del for i}
```

```
        {Calculo de los terminos independientes}
        if k<=nB-1 then
        begin
                difactk:=PBarra(BARRASORDENADAS^.At(k-1))^.S.r-
                actibarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                ctek.r:=-difactk;
                ctek.i:=0;
            {$ifdef deb_fg}
                writeln('calculo termino independiente');
                writeln('diferencia activa ',ctek.r);readln;
            {$endif}
                SJ^.AcumularConstante(k,ctek);
        end
        else
        begin
                kaux:=k;
                k:=k-(nB-1);
                difreack:=PBarra(BARRASORDENADAS^.At(k-1))^.S.i-
                reacbarr(Xv,BARRASORDENADAS,Admitancias,k,nB,nBC,nBcR);
                ctek.r:=-difreack;
                ctek.i:=0;
                {$ifdef deb_fg}
                writeln('calculo de termino independiente');
                writeln('diferencia reactiva ',ctek.r);readln;
                {$endif}
                SJ^.AcumularConstante(kaux,ctek);
                k:=kaux;
        end;

        end;{end del for k}

        {$IFDEF DERIVADAS}

        {Inversi¢n de la matriz jacobiana}
        esinvertible:=DXY^.Inv;
        if not(esinvertible) then
        horrores.error('Calculo de derivadas: Matriz no invertible en
CALCULARFUNG');

end.
{+doc
```

1.7. El programa principal: flucar 3.0

FLUCAR 3.0 maneja los procedimientos creados en la unidades que expusimos y los va llamando de forma tal de resolver el problema general en forma correcta. El esquema lógico básico de FLUCAR 3.0 es el que se muestra:



A partir de este diagrama de bloques elaboramos el programa FLUCAR 3.0:

```
{+doc
+NOMBRE:  FLUCAR30
+CREACION:  5/01
+MODIFICACION:
+AUTORES:  Alfredo Costa – Claudio Olmedo
```

```
+REGISTRO:
+TIPO:  Programa Pascal
+PROPOSITO:         Resoluci¢n del problema de flujo de carga utilizando el
              método de Newton-Raphson modificado

-doc}

{
        El programa resuleve el Flujo de Carga mediante el m,todo de Newton-
Raphson en coordenadas polares de un sistema en el que se pueden tener
barras de carga (se fija P y Q), barras de generacion y voltaje controlado
(se fija V y P), barras donde se controle la tension mediante un regulador
(se fija P, Q y V) y una barra flotante (con V y delta fijos).
A partir de la matriz de impedancias de la red y de los datos iniciales
de las barras se genera la matriz de Newton-Raphson Jo(inicial). Las
variables que no son datos se inicializan de acuerdo a los valores iniciales
ingresados en el archivo de datos de entrada. Luego se resuelve el sistema Jo
y a partir de los resultados se calcula el nuevo sistema J1.
El proceso se repite hallando los sucesivos Jk hasta verificar la condicion
de parada. Esta consiste en que DFX<=toleranciaF y DX<=toleranciaX
 o que el numero de iteraciones supere la cantidad de MAXNIT.
        Este programa lee los datos de un archivo de texto que debe
organizarse de la siguiente manera: VER ARCHIVO PRUEBA.DAT.  El archivo que
debe pasarse como par metro debe tener la extensi¢n .DAT.  Los resultados
del flujo apareceran un archivo del mismo nombre pero con la extensi¢n .RES.
}


program flucar30;

uses
        xMatDefs, TYVS2, EntDat, AlgebraC, cronomet, servic2, usistema,
        {$I xCRT},
        NR2,MatCPX, barrs2, impds1, MatAdm,salida3, checklim,strings;

var
  Cronometro: Crono;
  NombreArch,NombreArchent,NombreArchsal,extensionent,
  extensionsal: string;
  f:text;
  NIT:longint;
  convergencia:boolean;
  Xv,Xn,Xvsubcero:PVectComplex;
  epsx,epsf:NReal;
  Pbase:integer;
```

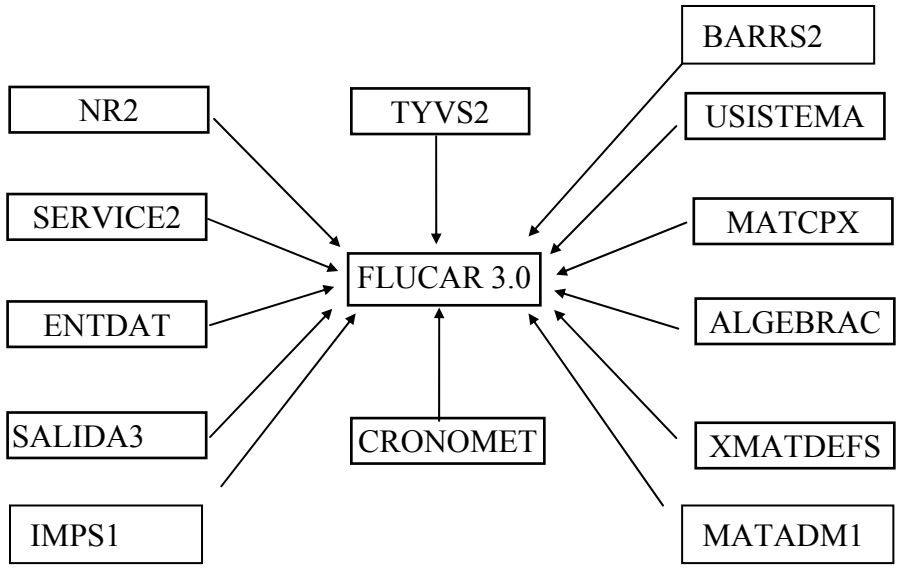
```
  Vbase:integer;
  pp:integer;


procedure Help;
begin
        writeln;
        writeln('=========================================>PROYEC
TO FLUCAR 01');
        writeln(' FLUCAR VER 3.0');
        writeln('_____________________________________________');
        writeln(' Sintaxis: ');
        writeln('             FLUCAR30 NomArch ');
        writeln;
        writeln(' NombArch : Nombre del archivo con las definiciones del sistema sin
extensi¢n');
        writeln('');
        writeln('Los resultados se almacenan en el archivo del mismo nombre con
extensi¢n .RES');
        halt(1);
end;

 begin
        {$define deb_nr}
        clrscr;
        if ParamCount < 1 then help;
        NOmbreArch:= ParamStr(1);
        extensionent:='.dat';
        extensionsal:='.res';
      {NombreArchent:=NombreArch+extensionent;}
        NombreArchent:=NombreArch;
        pp:=length(NombreArch)-3;
        Delete(Nombrearch,pp,4);
        {modificacion realizada para simplificar conectica con Delphi}
        NombreArchsal:=NombreArch+extensionsal;
        New(BarraFlotante);
        New(Barras,Init(500,10));
        New(BarrasdeCarga,Init(500,10));
        New(BarrasdeGenyVcont,Init(500,10));
        New(Barrasconregulador,Init(500,10));
        New(BarrasOrdenadas,Init(500,10));
        New(Impedancias,Init(500,10));
        New(Cuadripolos,Init(500,10));
        New(Trafos,Init(500,10));
        New(Reguladores,Init(500,10));
```

```
Pbase:=1;
Vbase:=1;

writeln('        R O G R A M A  D E  F L U J O  D E  C A R G A  v3.0');
writeln;
writeln('                    Autores:  A.Costa - C.Olmedo  ');
writeln;
writeln;
writeln;

LeerDatos(NombreArchent);
writeln;
writeln('LA INFORMACION ESTA SIENDO PROCESADA.....');
writeln;



New(Admitancias,initcrearsistema(NBarras));
FormarSistema;
{$IFDEF DEB_NR}
Admitancias^.muestrasistema;
{$ENDIF}


New(Xv,Init(NBarras+NBarrasdeCarga+NBarrasconregulador-1));
New(Xvsubcero,Init(NBarras+NBarrasdeCarga+NBarrasconregulador-1));
calcularXv(Xv,BarrasOrdenadas,Reguladores,NBarras,NBarrasdeCarga,
                                    NBarrasconregulador);
Xvsubcero^.copy(Xv^);
NIT:=0;
epsx:=TOLERANCIA/10;
epsf:=TOLERANCIA;
Cronometro.Borre;
Cronometro.Cuente;
newtonraphson(Xv,NIT,convergencia,epsx,epsf, 1,MAXNITs);
Cronometro.Pare;
actualizarbarras(BarrasOrdenadas,Admitancias,Xv,Nbarras,Nbarrasdecarga,
NBarrasconregulador);

writeln;
writeln;
writeln('Resultados:  ',NombreArchsal);
writeln;
writeln;
```

```
        writeln;

        {Escritura de resultados}
        Assign(f,NombreArchsal);
        Rewrite(f);
        writeln(f,'          P R O Y E C T O  F L U C A R v 3.0');
        writeln(f);
        writeln(f,'                I I E - 2 0 0 1    ');
        writeln(f);
        writeln(f);
        writeln(f);
        writeln(f,'DATOS: ',NombreArchent);
        writeln(f);
        writeln(f,'RESULTADOS: ',NombreArchsal);
        writeln(f);
        writeln(f);
        writeln(f,'SI BIEN LOS CALCULOS SE HACEN EN POR UNIDAD, LOS
VALORES DE SALIDA SE REPRESENTAN EN FUNCION');
        writeln(f,'DE LAS SIGUIENTES BASES:');
        writeln(f,'Potencia base (MVA): ',Pbase);
        writeln(f,'Tension base (kV): ',Vbase);
        writeln(f);
        writeln(f,' RESULTADOS: ');
        writeln(f);
        writeln(f,' NODO      POTENCIA APARENTE              TENSION');
        reordenarbarras(BarrasOrdenadas,Barras);
        writeBarras(f,Barras,pbase,vbase,Xv);
        writeFlujosDePotencia(f,Pbase,Vbase,Xv);
        writeln(f);
        writeln(f,'VERIFICACION DE LIMITES:');
        writeln(f);
        checkbarrs(f);
        checkimpds(f);
        checkcuadri(f);
        checktrafos(f);
        checkreg(f);
        writeln(f,'NITs: ',NIT,' converge: ',convergencia, ' Tiempo: ',
        Cronometro.Cuenta:8:4,'s');
        close(f);
        {$IFDEF DEB_FLU}
        write(memavail,'bytes libres');
        readln;
        {$ENDIF}
        Dispose(BarraFlotante,done);
        Dispose(Barras,done);
```

```
BarrasdeCarga^.DeleteAll;
BarrasdeGenyVcont^.DeleteAll;
Barrasconregulador^.DeleteAll;
BarrasOrdenadas^.DeleteAll;
Dispose(Impedancias,done);
Dispose(Cuadripolos,done);
Dispose(Trafos,done);
Dispose(Reguladores,done);
Admitancias^.destruirsistema;
{$IFDEF DEB_FLU}
write(memavail,'bytes libres');
readln;
{$ENDIF}


end.
```

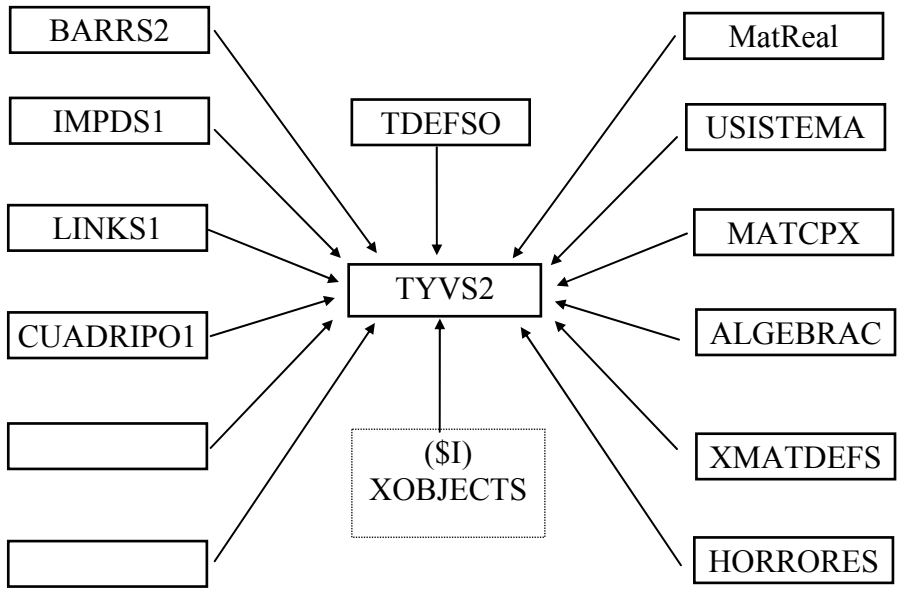
Las unidades que utiliza FLUCAR 3.0 pueden verse en el esquema que se muestra a continuación:



1.7.1. UNIDAD TYVS2

TYVS2 es una de las unidades de la biblioteca Pascal del IIE. Contiene la definición de las variables soporte del problema. Son variables que son vistas tanto desde el programa principal como desde todas las unidades que este utiliza.

Las unidades usadas por TYVS2 pueden verse en el esquema que mostramos a continuación:




```
unit TyVs2;

interface
uses
        {$I xObjects},AlgebraC,xmatdefs, MatCPX, MatReal, usistema,
        Links1, Barrs2, Impds1,Cuadri1,horrores;

type

        TIndice=integer;

var
        Barras : PCollection;
        BarrasdeCarga: PCollection;
        BarrasdeGenyVCont: PCollection;
```

```
        Barrasconregulador: PCollection;
        BarraFlotante: PBarra;
        BarrasOrdenadas: PCollection;
        ParametrosOpt: PCollection;
        Impedancias : PCollection;
        Cuadripolos: PCollection;
        Trafos: PCollection;
        Reguladores: PCollection;
        Admitancias: PSistema;
        DXY, DGYP, DXYP: PMatR;
        Tolerancia: NReal;
        MAXNITs: integer;
        NombreArchent,NombreArchsal: string;



var
        NBarras,NBarrasordenadas, NBarrasdecarga,NBarrasdegenyvcont,
        NImpedancias, NCuadripolosPi, NTrafos, NBarrasconregulador,
        NReguladores, NParametros: integer;

implementation

function  IndiceDeNodo(var r: string; var rescod: integer):TIndice; far;
{Dada una barra a través de su nombre, devuelve la posición en la
lista de BARRASORDENADAS comenzando en 1}

var
        k: integer;
begin
        if r='N' then
        begin
                IndiceDeNodo:=0;
                rescod:=0
        end
        else
        begin
                k:= 0; {los ¡ndices en la TCollection comienzan en 0}
                while (k<= Nbarras-1)
                and(r<>PBarra(BarrasOrdenadas^.At(k))^.Nombre)
                do inc(k);
                if k>NBarras-1 then rescod:= -1 {si no encuentra el nombre dado
                                                devuelve rescod=-1}
                else
                begin
```

```
                    rescod:= 0;
                    IndiceDeNodo:= k+1; {La numeración que yo elijo es: Neutro del
                                         sistema = barra 0; restantes barras numera-
                                         das desde 1 en adelante}
                end;
        end;
end;


function BarraPtr( k: integer): pointer; far;
{Dada una barra a través de su posición en la lista, devuelve un puntero
a la barra}

begin
        BarraPtr:=Barrasordenadas^.At(k);
end;



begin  {begin de implementation}

        Links1.Func_IndiceDeNodo:= IndiceDeNodo;
        Links1.Func_BarraPtr:= BarraPtr;

end.
```

1.7.2. UNIDAD CRONOMET

CRONOMET es una de las unidades de la biblioteca Pascal del IIE. Esta unidad esta hehca para medir tiempos. Implementa la clase de objetos **crono.** Un **crono** es un cronómetro que puede contar, parar de contar y ponerse en cero.

A continuación reproducimos su interfase:

```
{+doc
+NOMBRE: CRONOMET
+CREACION:1.1.90
+AUTORES:rch
+MODIFICACION:
+REGISTRO:
+TIPO: Unidad Pascal.
+PROPOSITO:def. objeto (crono). Sive de cronometro para medir tiempos.
+PROYECTO:rchlib
```

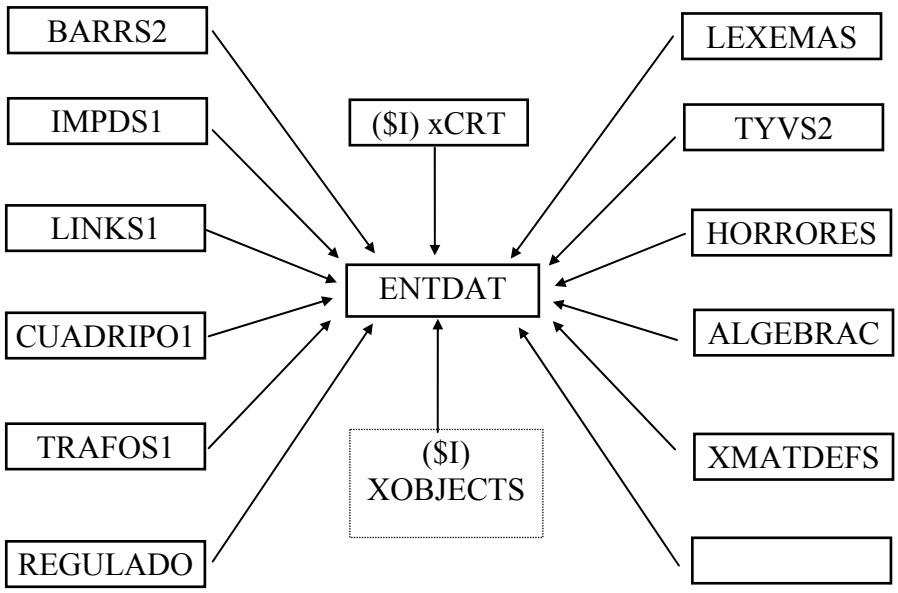
```
+REVISION:
+AUTOR:
+DESCRIPCION:
-doc}

unit cronomet;

interface (*****************************)
(* Devuelve el tiempo transcurrido en segundos     *)
uses
        xMatDefs,
        {$I xCRT},
        {$I xDOS};

type
        crono = object
                tac,tarr:NReal;
                procedure borre;
                procedure pare;
                procedure cuente;
                function cuenta:NReal;
                end;
```

1.8. *LA ENTRADA Y SALIDA DE DATOS*

1.8.1. ENTRADA DE DATOS

Para la entrada y salida de datos se realizó una interfase de entorno Delphi. Sin embargo la interfase gráfica desarrollada en Delphi invoca a Flucar3.0, relacionándose a través de archivos de text tipo DAT y RES, por tanto tratamao por un lado la entrada y salida desde el punto de vista de Flucar3.0 y de Delphi por separado.

La entrada y salida de datos se realiza en Flucar3.0 mediante dos unidades **ENTDAT** y **SALIDA3**.Para mejorar la interacción con el usuario, la introducción de los datos se realiza en entorno windows. Luego del cual se forma el archivo de entrada .DAT que alimentará al flujo de carga. Los cambios a nivel de usuario serán detallados mas adelante por lo cuál en este punto nos remitiremos a la explicación de las dos unidades mencionadas.**ENTDAT** es una unidad que proviene de la versión anterior del flujo, y la cual no fue modificada. A continuación se transcribe la unidad ENTDAT:




```
+doc
+NOMBRE:  ENTDAT
+CREACION:
+MODIFICACION:  8/97
+AUTORES:  MARIO VIGNOLO
+REGISTRO:
+TIPO:  Unidad Pascal
```

```
+PROPOSITO:  Implementaci¢n de la entrada de datos para FLUCAR 1.1
+PROYECTO: FLUCAR

+REVISION: 2001
+AUTOR: A.Costa-C.Olmedo
+DESCRIPCION:
-doc}

unit entdat;

interface
uses
        horrores, Lexemas, {$I xObjects}, AlgebraC, xMatDefs,
        TYVS2, Links1, Barrs2, Impds1, Cuadri1, Trafos1, Regulado, {$I xCRT};

var
        EsperoID: boolean;

procedure LeerDatos( ArchDef: string);

implementation

procedure LEERBARRAS( var a: TFlujoLetras; var r: string; var tipodeBarra:
TRestriccionde Nodo);
var

i: integer;
PBarraN:PBarra;

begin
                NBARRAS:=0;
                NBarrasdecarga:=0;
                NBarrasdegenyvcont:=0;
                NBarrasconregulador:=0;
                GetLexema( r, a);
                while (r<>'+') do
                begin { leeo una barra }
                        { Ide }
                        New(PBarraN); {Creo un nuevo puntero}
                        inc(NBARRAS);
                        PBarraN^.LeerDeFljLetras(a,r,tipodeBarra);
                        PBarraN^.Nro:=NBarras;
                        Barras^.Insert(PBarraN);
                        if tipodeBarra = [cf_P, cf_Q]  then
                                begin
```

```
                        inc(NBARRASDECARGA);
                        BarrasdeCarga^.Insert(PBarraN);
                      end;
                if tipodeBarra = [cf_P, cf_V] then
                      begin
                        inc(nbarrasdegenyvcont);
                        BarrasdeGenyVcont^.Insert(PBarraN);
                      end;
                if tipodeBarra = [cf_P, cf_Q, cf_V] then
                      begin
                        inc(nbarrasconregulador);
                        Barrasconregulador^.Insert(PBarraN);
                      end;
                if tipodeBarra = [cf_V, cf_delta] then
                      begin
                            Barraflotante^.Init(PBarraN^.Nombre,
                            PBarraN^.restriccion,
                            PBarraN^.S.r,PBarraN^.S.i,mod1(PBarraN^.V),
                            fase(PBarraN^.V)*180/pi);
                            Barraflotante^.Nro:=PBarraN^.Nro;
                      end;
          GetLexema( r, a);
          end; {while}
          Nbarrasordenadas:=0;
          for i:=1 to Nbarrasdecarga do
                begin
                      inc(nbarrasordenadas);
                      BarrasOrdenadas^.Insert(PBarra(BarrasdeCarga^.At(i-
1)));
                end;
                for i:=1 to Nbarrasconregulador do
                begin
                      inc(nbarrasordenadas);
                      BarrasOrdenadas^.Insert(
                      PBarra(Barrasconregulador^.At(i-1)));
                end;
          for i:=1 to nbarrasdegenyvcont do
                begin
                      inc(nbarrasordenadas);
                      BarrasOrdenadas^.Insert(
                      PBarra(BarrasdeGenyVcont^.At(i-1)));
                end;
          inc(nbarrasordenadas);
          BarrasOrdenadas^.Insert(BarraFlotante);
          end;
```

```
procedure LEERIMPEDANCIAS( var a: TFlujoLetras; var r: string);

var
        PImpedanciaN: PImpedancia;

begin
                NImpedancias:=0;
                GetLexema( r, a);
                while r<>'+' do
                begin { leeo una Impedancia }
                        { Ide }
                        New(PImpedanciaN); {Creo un nuevo puntero}
                        inc(NImpedancias);
                        PImpedanciaN^.LeerDeFljLetras( a, r);
                        Impedancias^.Insert(PImpedanciaN);
                        GetLexema( r, a);
                end; { while }
end;

procedure LEERCUADRIPOLOSPI( var a: TFlujoLetras; var r: string);

var
        PCuadripoloPiN: PCuadripoloPi;


begin
                NCuadripolosPi:=0;
                GetLexema( r, a);
                while r<>'+' do
                begin { leeo un cuadripolo }
                        { Ide }
                        New(PCuadripoloPiN); {Creo un nuevo puntero}
                        inc(NCuadripolosPi);
                        PCuadripoloPiN^.LeerDeFljLetras( a, r);
                        Cuadripolos^.Insert(PCuadripoloPiN);
                        GetLexema( r, a);
                end; { while }
end;

procedure LEERTOLERANCIA( var a: TFlujoLetras; var r: string);
var
        res: integer;
begin
                res:= LeerNReal(a, Tolerancia);
                if res <> 0 then error('leyendo tolerancia');
```

```
                getlexema(r,a);
end;

procedure LEERNITS( var a: TFlujoLetras; var r: string);
var
        res: integer;
begin
                res:= LeerNInteger(a, MAXNITs);
                if res <> 0 then error('leyendo iteraciones');
                getlexema(r,a);
end;

procedure LEERTRAFOS( var a: TFlujoLetras; var r: string);

var
        PTrafoN: PTrafo;

begin
                NTrafos:=0;
                GetLexema( r, a);
                while r<>'+' do
                begin { leeo un Trafo }
                        { Ide }
                        New(PTrafoN); {Creo un nuevo puntero}
                        inc(NTrafos);
                        PTrafoN^.LeerDeFljLetras( a, r);
                        Trafos^.Insert(PTrafoN);
                        GetLexema( r, a);
                end; { while }
end;


procedure LEERREGULADORES( var a: TFlujoLetras; var r: string);

var
        PReguladorN: PRegulador;

begin
                NReguladores:=0;
                GetLexema( r, a);
                while r<>'+' do
                begin { leeo un Regulador }
                        { Ide }
                        New(PReguladorN); {Creo un nuevo puntero}
                        inc(NReguladores);
```

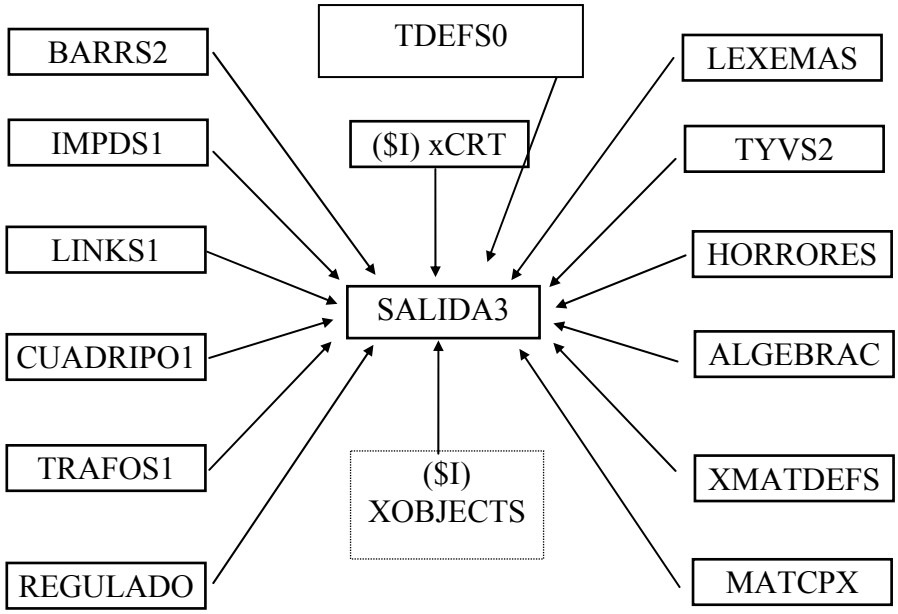
```
                    PReguladorN^.LeerDeFljLetras( a, r);
                    Reguladores^.Insert(PReguladorN);
                    GetLexema( r, a);
                end; { while }
end;


procedure LeerDatos( ArchDef: string);
var
        f: TBufStream;
        r: string;
        a: TFlujoLetras;
        tipodeBarra: TRestriccionDeNodo;
        finlectura:boolean;
        {$IFDEF WINDOWS}
        pstr: PCHAR;
        {$ENDIF}

begin
        {$IFNDEF WINDOWS}
        f.INit(ArchDef, StOpenRead, 512);
        {$ELSE}
        pstr:=@ArchDef[1];
        ArchDef:=ArchDef+#0;
        f.INit(pstr, StOpenRead, 512);
        {$ENDIF}

        if f.status <> stOk then
        begin
                writeln(' Error abriendo archivo');
                writeln(' st: ',f.status,'   errinfo: ',f.errorinfo);
                readln;
                halt(1);
        end;
        a.init(f);
        finlectura:= false;
        EsperoID:= false;
        repeat
                GetLexema( r, a);
                if EsperoID then
                begin
                        EsperoID := false;
                        if r = 'BARRAS' then
                                LEERBARRAS(a,r,tipodeBarra)
                        else if r = 'IMPEDANCIAS' then
```

```
                    LEERIMPEDANCIAS(a,r)
                else if r = 'CUADRIPOLOSPI' then
                    LEERCUADRIPOLOSPI(a, r)
                else if r= 'TRAFOS' then
                    LEERTRAFOS(a,r)
                else if r= 'REGULADORES' then
                    LEERREGULADORES(a,r)
                else if r = 'TOLERANCIA' then
                    LEERTOLERANCIA(a,r)
                else if r = 'NITS' then
                    LEERNITS(a,r)
                else if r = 'FIN.' then
                    finlectura := true;
            end;
            if r='+' then EsperoID := true;
            {write(r);} { caracteres no procesados }
        until finlectura;
        writeln('FIN DE LA LECTURA DE DATOS');
        f.done;
end;
end.
```

1.8.2. SALIDA DE DATOS

La salida de datos fue modificada respecto al la versión del flujo anterior, dado que en la barra regulada, la tensión proviene de los datos originales, también se modificó la salida de datos de los reguladores, dado que ahora la posición de los reguladores se encuentra dentro del vector Xv.

A continuación se transcribe la unidad SALIDA3:




```
{+doc
+NOMBRE:  SALIDA3
+CREACION:
+MODIFICACION:  8/97
+AUTORES:  MARIO VIGNOLO
+REGISTRO:
+TIPO:  Unidad Pascal
+PROPOSITO:  Procedimientos para mostrar resultados
+PROYECTO: FLUCAR

+REVISION:   06/2001
+AUTOR:      COSTA - OLMEDO
+DESCRIPCION:CORRECCION Y AMPLIACION

-doc}

unit salida3;

interface
uses
        Horrores, Lexemas, {$I xObjects}, AlgebraC, xMatDefs,
        Barrs2, Impds1, Cuadri1, Trafos1, Regulado, TYVS2, Links1, TDefs0,
        {$I xCRT}, MAtCPX;
```

```
procedure WriteBarras(var f:text; Bar-
ras:PCollection;pbase:integer;vbase:integer;Xv:Pvectcomplex);
procedure WriteFlujosDePotencia(var f:text;Pbase,Vbase:integer;Xv:Pvectcomplex);

implementation

procedure WriteBarras(var
f:text;Barras:PCollection;pbase:integer;vbase:integer;Xv:PVectComplex);
var
        k,l: integer;
        TSG, TSC, fasekc: complex;
        Tension:complex;

begin
        TSG:= numc(0,0)^;
        TSC:= numc(0,0)^;

        for k:= 1 to (NBARRAS-Nbarrasconregulador) do
        begin
                PBarra(Barras^.At(k-1))^.WriteTXT(f,Pbase,vbase);
                if PBarra(Barras^.At(k-1))^.S.r>0 then
                TSG.r:=(TSG.r+PBarra(Barras^.At(k-1))^.S.r)
                else TSC.r:=(TSC.r-PBarra(Barras^.At(k-1))^.S.r);
                if PBarra(Barras^.At(k-1))^.S.i>0 then
                TSG.i:=(TSG.i+PBarra(Barras^.At(k-1))^.S.i)
                else TSC.i:=(TSC.i-PBarra(Barras^.At(k-1))^.S.i);
                writeln(f);
        end;



        for k:=1 to (Nbarrasconregulador) do
        begin
                PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-
1))^.WriteTXT1(f,Pbase,vbase);
                Tension.r:=PBarra(BarrasOrdenadas^.At(Nbarras-
Nbarrasconregulador+k-2))^.V.r;
                Xv^.e(fasekc,Nbarras-Nbarrasconregulador+k-1);
                Tension.i:=fasekc.r;

                wtxtc1(f,Tension,CA_desplegar,Pbase,Vbase);

                if PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-1))^.S.r>0 then
                TSG.r:=(TSG.r+PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-
1))^.S.r)
```

```
                else TSC.r:=(TSC.r-PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-
1))^.S.r);
                if PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-1))^.S.i>0 then
                TSG.i:=(TSG.i+PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-
1))^.S.i)
                else TSC.i:=(TSC.i-PBarra(Barras^.At(Nbarras-Nbarrasconregulador+k-
1))^.S.i);
                writeln(f);
        end;
        write(f,' TSGeneracion: ');wtxtcln1(f,TSG,CA_Rectangulares,Pbase,Vbase);
        write(f,' TSConsumo:    ');wtxtcln1(f,TSC,CA_Rectangulares,Pbase,Vbase);
end;

procedure WriteFlujosDePotencia(var f:text;Pbase,Vbase:integer;Xv:PVectcomplex);
var
        k,variable,l: integer;
        Perdidas,Perdidasimp,Perdidascuad,Perdidastrafo,
        Perdidasregulador, Perdidastot,modVclo: NReal;
        S,S1,S12,S2,S21,Scon,I,modVkclo: complex;
begin
        writeln(f);
        writeln(f,' POTENCIAS ENTREGADAS A LAS IMPEDANCIAS ');
        Perdidas:=0;
        for k:= 1 to NImpedancias do
  with PImpedancia(Impedancias^.At(k-1))^ do
        begin
                TransfS(S12,S21,Scon,I);
                if (nodo1<>0) and (nodo2<>0) then
                begin
                        write(f,PBarra(BarrasOrdenadas^.At(Nodo1-1))^.Nombre,'-
>',Nombre,' ');
                        wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
                        write(f,PBarra(BarrasOrdenadas^.At(Nodo2-1))^.Nombre,'-
>',Nombre,' ');
                        wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
                        writeln(f);
                end
                else
                begin
                        if nodo1=0 then
                        begin
                                write(f,'N ','->',Nombre,' ');
                                wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
                                write(f,PBarra(BarrasOrdenadas^.At(Nodo2-
1))^.Nombre,'->',Nombre,' ');
```

```
                        wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
                        writeln(f);
                    end
                    else
                    begin
                        write(f,PBarra(BarrasOrdenadas^.At(Nodo1-
1))^.Nombre,'->',Nombre,' ');
                        wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
                        write(f,'N ','->',Nombre,' ');
                        wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
                        writeln(f);
                    end;
                end;
        Perdidas:=Perdidas+Scon.r*Pbase;
        end; {with}
        Perdidasimp:=Perdidas;
        writeln(f,'Perdidas Joule en las impedancias: ',Perdidasimp:10:7);
        writeln(f);

        writeln(f,' POTENCIAS ENTREGADAS A LOS CUADRIPOLOS ');
        Perdidas:=0;
        for k:= 1 to NCuadripolosPi do
  with PCuadripoloPi(Cuadripolos^.At(k-1))^ do
        begin
            TransfS(S12,S21,Scon,I);
            write(f,PBarra(BarrasOrdenadas^.At(Nodo1-1))^.Nombre,'->',Nombre,'
');
            wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
            write(f,PBarra(BarrasOrdenadas^.At(Nodo2-1))^.Nombre,'->',Nombre,'
');
            wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
            writeln(f);
            Perdidas:=Perdidas+Scon.r*Pbase;
         end; {with}
         Perdidascuad:=Perdidas;
        writeln(f,'Perdidas Joule en los cuadripolos: ',Perdidascuad:10:7);
        writeln(f);
        writeln(f,' POTENCIAS ENTREGADAS A LOS TRANSFORMADORES ');
        Perdidas:=0;
        for k:= 1 to NTrafos do
        with PTrafo(Trafos^.At(k-1))^ do
        begin
            TransfS(S12,S21,Scon,I);
            write(f,PBarra(BarrasOrdenadas^.At(Nodo1-1))^.Nombre,'->',Nombre,'
');
```

```
                wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
                write(f,PBarra(BarrasOrdenadas^.At(Nodo2-1))^.Nombre,'->',Nombre,'
');
                wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
                writeln(f);
                Perdidas:=Perdidas+Scon.r*Pbase;
         end; {with}
         Perdidastrafo:=Perdidas;
         Perdidastot:=Perdidasimp+Perdidascuad+Perdidastrafo;
        writeln(f,'Perdidas Joule en los transformadores: ',Perdidastrafo:10:7);
        writeln(f);

        writeln(f,' REGULADORES ');
        Perdidas:=0;
        for k:= 1 to NReguladores do
        with PRegulador(Reguladores^.At(k-1))^ do
        begin
                Xv^.e(modVkclo,(Nbarras+NBarrasdecarga+k-1));
                modVclo:=modVkclo.r;
                writeln(f,'Relación de transformación del regulador -1',Nombre,':
',modvclo:7:3);
                TransfS(S12,S21,Scon,I);
                write(f,PBarra(BarrasOrdenadas^.At(Nodo1-1))^.Nombre,'->',Nombre,'
');
                wtxtcln1(f,S12,CA_Rectangulares,Pbase,Vbase);
                write(f,PBarra(BarrasOrdenadas^.At(Nodo2-1))^.Nombre,'->',Nombre,'
');
                wtxtcln1(f,S21,CA_Rectangulares,Pbase,Vbase);
                writeln(f);
                Perdidas:=Perdidas+Scon.r*Pbase;
         end; {with}

         Perdidasregulador:=Perdidas;
         Perdidastot:=Perdidasimp+Perdidascuad+Perdidastrafo+Perdidasregulador;
        writeln(f,'Perdidas Joule en los reguladores: ',Perdidasregulador:10:7);
        writeln(f);
        writeln(f,'Perdidas Joule totales en las líneas: ',Perdidastot:10:7);


end;
end.
```