TESIS

REPARTO DE CARGAS EN LA SIMULACION Y ANALISIS DE REDES ELECTRICAS MEDIANTE MICROPROCESADORES EN PARALELO

por

ANTONIO GOMEZ EXPOSITO

Ingeniero Industrial por la E.T.S. de I.I.

de la Universidad de Sevilla

presentada en la

ESCUELA TECNICA SUPERIOR DE INGENIEROS INDUSTRIALES

de la

UNIVERSIDAD DE SEVILLA

para la obtención del

GRADO DE DOCTOR INGENIERO INDUSTRIAL

SEVILLA, JUNIO DE 1985.

INTRODUCCION.

PLANTEAMIENTO Y OBJETIVO DE LA TESIS.

Hoy día no se concibe un sistema eléctrico de potencia, sin la presencia de los computadores digitales, tanto en lo que se refiere al control y seguimiento de su funcionamiento, como en la planificación, amén de otras funciones auxiliares de gestión. Y ésto es así desde la propia aparición de los mismos [61], puesto que una de sus primeras aplicaciones tecnológicas fue en este área, lo cual no resulta extraño si se tienen en cuenta las sólidas bases matemáticas y numéricas que Kirchhoff, Heaviside y otros habían dado a la ingeniería eléctrica.

Desde esos comienzos, el esfuerzo principal se ha dedicado a incrementar el tamaño del sistema bajo estudio, y a reducir los tiempos, en paralelo al crecimiento de las redes y su mayor interconexión.

A lo largo de esta memoria (véase capítulo 1), y en otras publicaciones, queda patente la utilidad y el carácter básico del problema del reparto de cargas en numerosos aspectos del análisis de los sistemas de potencia. Las importantes mejoras que supusieron, entre los años 1965-1968, la introducción del método de Newton-Raphson, y las técnicas de eliminación ordenada para la solución de sistemas de ecuaciones no lineales, fueron definitivas, y potenciaron el uso de los computadores digitales frente a los antiguos analizadores de redes.

En la actualidad, podemos estar atravesando una etapa parecida de adaptación a las nuevas tecnologías. Mientras se acaban de poner a punto los ordenadores de la última generación, los microprocesadores, cada vez más potentes y económicos, y numerosos circuitos especiales, se están utilizando en campos hasta ahora reservados como mínimo a los miniordenadores.

Si en los años 60 se puede decir que el mayor avance se produjo en el "software", en los últimos 5 años el "hardware" ha evolucionado tan rápidamente, que es preciso un replanteamiento, y un esfuerzo mucho mayor, en los aspectos algorítmicos y de programación, para adaptarse a dichos avances y poder sacarles un rendimiento adecuado.

Las nuevas tendencias apuntan a la resolución en paralelo del problema, fundamentalmente con dos variantes: Bien con varios microprocesadores trabajando simultáneamente de forma sincronizada, o bien conectando al ordenador principal un procesador especializado con varias unidades aritméticas de multiplicación y suma, y con estructura "pipeline"[82].

La primera variante requiere un cambio radical de filosofía en los algoritmos para aprovechar la arquitectura desarrollada, ya que hasta ahora se trabajaba secuencialmente. La segunda variante puede no requerir grandes cambios, pues existen incluso compiladores en FORTRAN que generan código para esos procesadores, realizándose el paralelismo internamente, de forma transparente al usuario.

Se piensa que en esta misma década se conseguirán circuitos VLSI, especializados en la resolución de sistemas de ecuaciones [96], que permitirán reducir el tiempo de cálculo en dos órdenes de magnitud respecto a los que se consiguen actualmente con los mejores algoritmos, haciendo posible la solución en tiempo real del problema de la estabilidad transitoria, o del flujo de cargas óptimo, por poner dos ejemplos significativos.

En la estimación de estado se ha avanzado más en este terreno, probablemente por el caracter distribuido que presentan las medidas, siendo más fácil una jerarquización y por tanto el paralelismo llevado a sus últimas consecuencias (separación geográfica de procesadores). Para el problema del reparto de cargas ya han aparecido diversas publicaciones [55,82,96], esperándose que aumenten sustancialmente en el futuro.

A petición de la Cía. Sevillana de Electricidad, dentro de los planes de investigación de UNESA, se inició en el Departamento de Electrónica y Automática la realización de un prototipo de simulador de redes para entrenamiento de operadores de Centros Provinciales de Maniobra. Esto fue el "leitmotiv" de los trabajos que se describen en esta memoria, puesto que fue preciso disponer de un programa para el cálculo "on-line" del flujo de cargas.

Después se ha profundizado en algunos aspectos, obteniéndose un algoritmo original y eficiente para la resolución en paralelo del reparto de cargas por el método desacoplado rápido.

Para ello se han utilizado microprocesadores de 16 bits, trabajando en un bus estándard. De esa forma se tienen resueltos numerosos problemas de comunicaciones de datos, y a la vez el coste es muy bajo, en comparación a la otra alternativa ya mencionada.

SUMARIO DE LOS CAPITULOS POSTERIORES.

El capítulo 1 es una introducción básica al problema del reparto de cargas, situándolo en el contexto general del análisis de sistemas de potencia.

El capítulo 2 hace una recopilación detallada de los algoritmos más importantes propuestos desde hace más de 20 años.

En el capítulo 3 se hace mención a una serie de aspectos, que no dejan de ser importantes, pero que se apartan de la línea principal del problema. Completan la visión de los capítulos 1 y 2.

El capítulo 4 se dedica exclusivamente al tema de las matrices vacías o dispersas, como pieza clave de la solución de numerosos problemas de ingeniería, y en particular de las redes eléctricas.

Se describen algunas de las numerosas experiencias realizadas en el capítulo 5. El lenguaje utilizado en estas pruebas es el FORTRAN, y la máquina el miniordenador PDP-11/34 de Digital.

Se propone en este capítulo una simplificación original al método desacoplado rápido, que funciona bien en la mayoría de los casos estudiados.

El capítulo 6 aborda la solución en paralelo del problema. Tanto el algoritmo de ordenación de matrices, como la solución del sistema de ecuaciones, son originales, y aplicables en otros contextos.

Por último, en el capítulo 7, se obtienen brevemente las conclusiones del trabajo, y algunas sugerencias.

El método de Newton se detalla en el apéndice 2.

Los diferentes capítulos están escritos de forma que se puedan leer individualmente sin ningún problema, especialmente el capítulo 4.

Una lectura rápida de la tesis podría consistir en el capítulo 1, los párrafos 2.1, 2.2.11, 2.2.17, 2.2.23, 4.4, y los capítulos 5, 6 y 7.

AGRADECIMIENTOS.

Quiero agradecer a D. Leopoldo G. Franquelo su acertada orientación en la consecución de la tesis, sus consejos y sus discusiones.

Al Colegio de Ingenieros Industriales de Andalucía Occ., agradezco su estímulo y su apoyo económico, que me permitió una mayor dedicación a la tesis.

A D. Javier Aracil, D. Carlos Izquierdo, D. Jose Luís Calvo, y demás compañeros del Depto., les expreso mi agradecimiento por su colaboración y correcciones a la redacción provisional.

A la Cía. Sevillana de Electricidad, que me ha suministrado los datos de algunas redes utilizadas como test.

A mi esposa e hija, por cederme generosamente gran parte de su tiempo durante estos años. A mi esposa también por la edición del texto y la confección de las figuras.

I N D I C E

1. EL PROBLEMA DEL REPARTO DE CARGAS.

1.1 INTRODUCCION.

Hacer un reparto de cargas ("load flow" o "power flow" en terminología sajona), consiste en determinar las condiciones de operación en régimen permanente de un sistema de potencia. Ello conlleva:

- La formulación de un modelo matemático apropiado.
- La especificación de una serie de restricciones en los nudos del sistema.
- La resolución numérica del problema.
- El cálculo de magnitudes secundarias, necesarias para la completa definición del régimen actual de trabajo.

Las cargas se especifican normalmente por la potencia activa y reactiva que consumen, supuestas independientes de la tensión y la frecuencia. Para un conjunto dado de cargas, existen infinitas combinaciones con las que los generadores pueden atender esas cargas más las pérdidas. Atendiendo a la experiencia de los operadores de los centros de control, se actúa sobre los distribuidores de las turbinas para lograr un reparto económico entre los generadores. Las tensiones se mantienen constantes en las barras de generación con los reguladores de tensión, actuando sobre la excitación de los mismos. En otras barras se puede controlar la tensión con una inyección apropiada de potencia reactiva, utilizando bancos de condensadores o actuando sobre las tomas de transformadores próximos.

Al realizar el flujo de cargas se calculan las tensiones complejas en todos los nudos, y como consecuencia, los flujos de activa y reactiva en las líneas de transmisión, las pérdidas totales del sistema, y la potencia reactiva necesaria para mantener la tensión constante en los nudos que se requiera.

La red se presenta por parámetros lineales, bilaterales y concentrados, aunque debido a las restricciones en los nudos, el problema se hace no lineal, lo que implica que la solución deba ser iterativa.

A los métodos numéricos utilizados, se les exige una serie de propiedades, a saber [104].

- Rapidez de cálculo, especialmente importante en aplicaciones "on line", interactivas, o cuando sea preciso resolver múltiples casos.
- Requerimiento pequeño de memoria, sobretodo para sistemas muy grandes, o cuando se utilizan miniordenadores para aplicaciones en tiempo real, donde no son aceptables las técnicas de "overlays".
- Solución fiable y precisa en la resolución de problemas difíciles, en tiempo real, o análisis de contingencias.
- Versatilidad, o capacidad para manejar casos convencionales y especiales (p.e. ajuste de tomas de transformadores) así como la posibilidad de insertarse en problemas más complejos.
- Simplicidad o facilidad de codificación del programa.

1.2 SIMBOLOGIA Y NOMENCLATURA.

En términos generales se seguirá la siguiente notación:

$E_i = V_i \angle\theta_i =$ $= e_i + jf_i$	Tensión nodal compleja.
I_i	Corriente nodal inyectada.
$S_i = P_i + jQ_i =$ $= (P_g - P_l) + j(Q_g - Q_l)$	Potencia neta inyectada en el nudo i.
$\Delta S_i = \Delta P_i + j\Delta Q_i$	Error residual en la potencia.
$\Delta E_i = \Delta V_i \angle\Delta\theta_i =$ $= \Delta e_i + j\Delta f_i$	Corrección a la tensión nodal.
$Y = G + jB$	Matriz de admitancia de nudos.
Z	Matriz de impedancias de nudos.
J	Matriz jacobiano.
n	Normalmente número de nudos.
X_{ij}	Reactancia entre los nudos i, j.
$\theta_{ij} = \theta_i - \theta_j$	Desfase entre los nudos i, j.
r	Normalmente nudo de referencia ($\theta = 0$).
$P_{ij} + jQ_{ij}$	Flujo de potencia de i a j por la línea que los une.

Los superíndices "e"(o "esp"), "c"(o "cal") y * denotan respectivamente magnitudes especificadas, calculadas y complejas conjugadas. Los subíndices g y l denotan generación y carga.

En muchos casos, sobretodo para mantener la notación original de una referencia bibliográfica, se utilizará la letra V en lugar de la E para especificar la tensión compleja. Distinguiremos entonces los módulos encerrándolos entre barras verticales, como es usual, salvo que el contexto evite la posibilidad de error, en cuyo caso se omitirán las barras.

Otros superíndices que los mencionados indicarán normalmente iteraciones.

Sobre la marcha se introducirá la notación más particular.

1.3 FORMULACION BASICA.

Para una red con n nudos, las ecuaciones nodales se escriben:

$$I = Y\,E$$

$$I_i = \sum_{j=1}^{n} Y_{ij}\,E_j \qquad i=1,2...n \tag{1.1}$$

En cada nudo se debe cumplir además la restricción:

$$S_i = E_i\,I_i^{*} \tag{1.2}$$

siendo S_i la potencia inyectada neta (generación menos consumo). Sustituyendo (1.2) en (1.1):

$$S_i = E_i \sum_{j=1}^{n} Y^*_{ij} \cdot E^*_j \qquad i=1,2...n \tag{1.3}$$

o bien

$$S^*_i = P_i - jQ_i = E^*_i \sum_{j=1}^{n} Y_{ij} E_j \qquad i=1,2...n \tag{1.4}$$

Las n ecuaciones complejas anteriores son la fórmula general para el reparto de cargas, pudiéndose descomponer en las 2n ecuaciones siguientes:

$$\begin{aligned} P_i &= V_i \sum_{j=1}^{n} V_j (G_{ij} \text{Cos}\, \vartheta_{ij} + B_{ij}\, \text{Sen}\, \vartheta_{ij}) \\ Q_i &= V_i \sum_{j=1}^{n} V_j (G_{ij} \text{Sen}\, \vartheta_{ij} - B_{ij}\, \text{Cos}\, \vartheta_{ij}) \end{aligned} \tag{1.5}$$

o, expresando las tensiones en forma cartesiana:

$$\begin{aligned} P_i &= e_i \sum_{j=1}^{n} (G_{ij} e_j - B_{ij} f_j) + f_i \sum_{j=1}^{n} (G_{ij} f_j + B_{ij} e_j) \\ Q_i &= f_i \sum_{j=1}^{n} (G_{ij} e_j - B_{ij} f_j) - e_i \sum_{j=1}^{n} (G_{ij} f_j + B_{ij} e_j) \end{aligned} \tag{1.6}$$

Se pueden hacer las siguientes consideraciones sobre las ecuaciones anteriores:

- Son algebraicas, al trabajarse con un modelo estático.
- Son no lineales, lo que imposibilita una solución analítica.
- La suma de todas las P_i, constituye la diferencia entre la potencia generada y la consumida, que deben ser forzosamente las pérdidas en la red.
- Del mismo modo la suma de las Q_i será la potencia reactiva absorbida (o generada) en la red.
- Los desfases, en coordenadas polares, aparecen como diferencias.
- Conocidas P_i, Q_i, tendremos el mismo número de ecuaciones que de incógnitas, y podremos encontrar (al menos en teoría) una solución, que generalmente no será única, aunque sólo una de ellas tenga interés físico (véase capitulo 3).

Entre las variables del problema podemos distinguir los siguientes grupos:

- Las potencias activas y reactivas demandadas por las cargas, determinadas por el usuario, e imposibles de controlar (grupo de variables incontrolables).
- Las variables de control o independientes, sobre las que podemos actuar directamente, como son las potencias activas y reactivas generadas, las tomas de regulación, etc.

- Las variables de estado o dependientes, cuyo valor se obtiene a partir de las variables de control. Pueden ser las tensiones en gran parte de los nudos, los flujos de potencia por las líneas, etc.

Una vez conocida la carga demandada (primer grupo de variables) y con unos valores supuestos de las variables de control, podemos obtener las variables de estado.

Como se ha dicho, para las ecuaciones en coordenadas polares los ángulos aparecen en forma de diferencias, de manera que no podremos calcular los valores de los mismos por separado.

Tampoco podremos especificar todas las potencias generadas, pués el consumo de las líneas no se conoce a priori.

Solventamos ambos inconveniente si en uno de los nudos no especificamos P ni Q generados, y al mismo tiempo fijamos V y θ (tomamos una referencia para los ángulos). El número de incógnitas y el de ecuaciones sigue siendo el mismo, pues a pesar de haberse dado dos grados más de libertad hemos eliminado dos incógnitas.

En base a los comentarios anteriores podemos clasificar las barras del sistema en las siguientes:

- Nudos P-V. Se especifica para dichos nudos la potencia activa P, y la tensión en módulo se mantiene constante mediante el intercambio apropiado de potencia reactiva

(entre ciertos límites). Estas barras son normalmente las de generación, o aquellas subestaciones donde se regula la tensión por baterías de condensadores o compensadores síncronos. Las restricciones para estos nudos son por tanto:

$$P_i^{esp} = R_e\ (E_i . I_i^*)$$
$$V_i = V_i^{esp} = \sqrt{e_k^2 + f_k^2} \qquad (1.7)$$

- Nudos P-Q. Se especifica la potencia activa y reactiva. Corresponden a centros de consumo, donde se supone que los cambios de tensión no afectan a P y Q.

$$P_i + jQ_i = E_i I_i^* \qquad (1.8)$$

- Nudo flotante o de referencia. Se toma como referencia para las fases de tensiones, conociéndose además su módulo.

$$\theta_r = 0$$
$$V_r = V_r^{esp} \qquad (1.9)$$

Se elige como referencia uno de los nudos PV, a ser posible con gran capacidad de generación, o próximo a las interconexiones con otras áreas, asignándosele en la práctica la mayor responsabilidad en el control de la frecuencia.

Los nudos P-Q suponen normalmente entre un 80 y un 90 % del total.

Sea cual sea el procedimiento iterativo que se utilice para obtener la solución (véase capítulo 2), el nudo de referencia se omite en los cálculos, pues conocemos por completo su tensión. Para los nudos P-Q, no conocemos ni V ni ϑ, y es necesario obtener nuevos valores de estas magnitudes en cada iteración. Por último, en los nudos P-V conocemos V y debemos actualizar ϑ. Asímismo calcularemos Q en dichos nudos para contrastarla con los límites de reactiva. Si se excede alguno, daremos a Q el valor de dicho límite para la siguiente iteración y liberaremos V, pasando a ser realmente un nudo P-Q (véase capítulo 3).

Una vez resuelto el problema calcularemos P y Q para el nudo de referencia, mediante las dos ecuaciones no utilizadas en el proceso iterativo. Para acabar de definir el flujo de cargas, se obtendrán las potencias circulantes por las líneas, para lo cual, del modelo en π

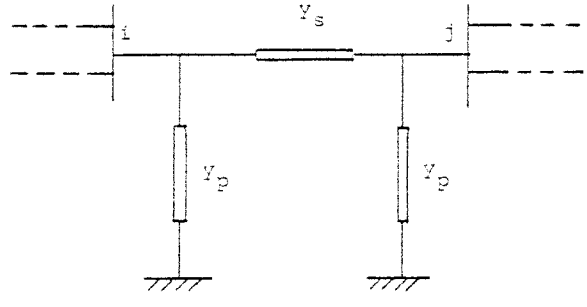


fig 1.1

se deducen las ecuaciones a utilizar.

$$I_{ij} = (E_i - E_j)y_s + E_i y_p \qquad (1.10)$$

$$S_{ij} = E_i \cdot I_{ij}^*$$

Es corriente dar también como resultado las pérdidas totales. Su cálculo se realiza, como ya se ha dicho, simplemente sumando todas las P_i.

Para cualquier procedimiento iterativo es preciso tener unos criterios de convergencia, para decidir cuándo la aproximación de los valores actuales es suficientemente buena.

Los dos criterios más utilizados son:

- La diferencia entre los valores que da para P y Q la iteración v, y los valores especificados, debe ser menor que cierta tolerancia prefijada, es decir:

$$|\Delta P_i| = |P_i^v - P_i^{esp}| \leqslant C \quad \text{para todos los nudos PQ,PV}$$
$$|\Delta Q_i| = |Q_i^v - Q_i^{esp}| \leqslant C' \quad \text{para todos los nudos PQ}$$

- La magnitud del cambio para las tensiones entre dos iteraciones sucesivas, deberá ser menor que cierto valor:

$$|E_i^{v+1} - E_i^v| \leqslant C$$

El primer criterio es preferible, pues la tolerancia especificada tiene una interpretación directa. Supone un límite máximo en el error del flujo de potencia por las líneas.

El segundo criterio se utiliza sólo cuando el algoritmo no calcula ΔP y ΔQ en cada iteración, y por tanto no estan dispo-

nibles esos valores. Es más sensitivo a la velocidad de convergencia del proceso iterativo. Una convergencia muy lenta en un momento dado puede hacernos creer que hemos llegado al final.

Otro criterio más sofisticado, aunque probablemente superfluo, es acotar la suma de los cuadrados de los valores absolutos de los residuos en P y Q:

$$\left| \sum_i \Delta P_i \right|^2 + \left| \sum_i \Delta Q_i \right|^2 \leqslant C$$

La cota del error C puede estar comprendida entre 10^{-2} y 10^{-5} p.u., dependiendo de las cifras significativas deseadas.

1.4 RELACION CON OTROS PROBLEMAS DE LOS SISTEMAS DE POTENCIA.

El gran esfuerzo dedicado por numerosos investigadores al problema del reparto de cargas, se justifica por la utilización de esta rutina básica en un buen número de aplicaciones del computador a los sistemas de potencia en todas sus fases (planificación y operación en tiempo real). Entre las más importantes podemos mencionar:

- Estabilidad transitoria.
- Flujo de cargas óptimo.
- Seguridad de operación en tiempo real.
- Planificación.

- Simulador para entrenamiento de operadores.

Comentaremos brevemente en qué medida están relacionadas esas aplicaciones con el reparto de cargas.

El problema de la estabilidad transitoria queda definido en forma general por dos sistemas de ecuaciones, uno diferencial y otro algebraico [1].

$$\begin{aligned} \dot{x} &= F(x,u) \\ 0 &= G(x,u) \end{aligned} \qquad (1.11)$$

Cada paso de integración del sistema diferencial, se alterna con la solución del sistema algebraico (que es precisamente un flujo de cargas). De ese modo se pueden conocer las tensiones en los nudos en cada etapa del proceso transitorio.

Lo anterior no es válido si se utiliza un método directo, mediante funciones de energía del tipo de Liapunov, en lugar de la simulación número mencionada. De todas formas esos métodos cualitativos no están muy generalizados todavía, y por tanto es necesaria una rutina de cálculo del flujo de cargas con ciertas peculiaridades [104,119].

Dado un conjunto de cargas demandadas, se pueden obtener diferentes flujos de cargas variando las variables de control. El término "flujo de cargas óptimo" se refiere a un estado de trabajo del sistema en el que se minimiza una cierta función, sujeto a

unas restricciones en las variables del problema. El flujo de cargas óptimo más popular es el del despacho económico [33].

Entre las diversas soluciones propuestas a este problema, están aquéllas [26] que utilizan un flujo de cargas convencional dentro del proceso global. Este flujo de cargas se debe resolver por el método de Newton-Raphson (véase capítulo 2), pués la inversa del jacobiano (sensitividad) indica cómo se deben ajustar las variables de control para realizar el siguiente paso en busca del óptimo.

Dy Liacco [30] ha sido uno de los principales impulsores en el análisis del funcionamiento en tiempo real del sistema. A él se debe, entre otras, la clasificación de los estados de trabajo en normal, alerta, emergencia, extremo y restauración, que resulta muy útil al sistematizar las decisiones que se deben tomar en cada momento en un centro de control. Para aclarar el papel que juega el reparto de cargas en este contexto, puede ayudar la siguiente figura, sintetizada de [31] :

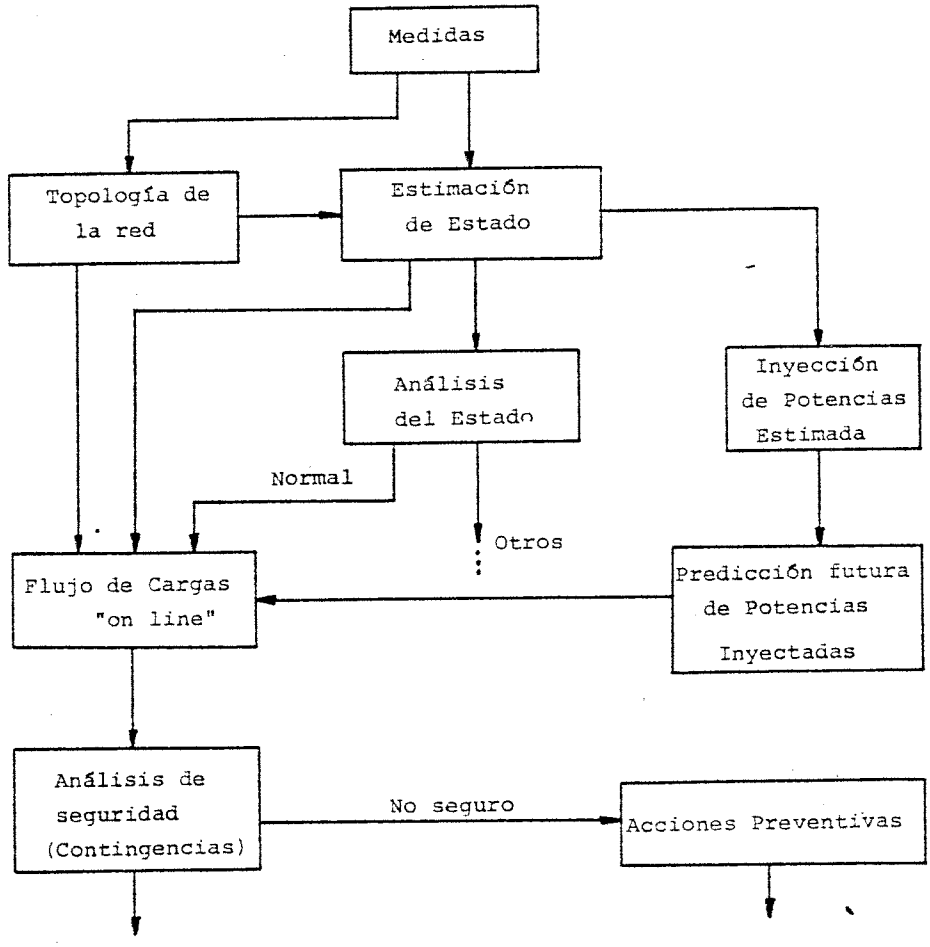


fig 1.2

Se realiza un análisis del estado estimado del sistema. En la mayoría de los casos estaremos en un estado normal (se satisfacen las restricciones de igualdad y desigualdad [31]). Ahora aparece el concepto de seguridad (en un sentido estático). El estado normal será seguro si ante unas hipotéticas contingencias sigue permaneciendo en el estado normal. El operador se debe adelantar a los acontecimientos y prever cómo se comportará el sistema ante lo que pueda ocurrir 15 ó 30 minutos después. Para ello, como se indica en la figura 1.2, dispone como herramientas del flujo de cargas y el análisis de contingencias (capítulo 3).

Otro tema importante al que se dedican muchas horas de ordenador es el de la planificación. En este campo se pueden diferen-

ciar claramente dos tipos de actividades (aunque no desligables totalmente). Por un lado existen unos aspectos no cuantificables, como el impacto subjetivo en el medio ambiente, las implicaciones sociales de una determinada política energética, etc. Y por otro está el desarrollo de modelos probabilísticos (predicción de carga, ciclos climatológicos) que determinan el crecimiento de la capacidad de generación, distribución y reparto. Normalmente, con la particularidad de que ahora no se trabaja en tiempo real, el reparto de cargas juega un papel esencial en la simulación de las diferentes situaciones, englobándose en cierto modo las tres facetas mencionadas anteriormente (estabilidad transitoria, optimización, seguridad) [61].

Recientemente, los simuladores para entrenamiento de operadores están tomando un gran auge, debido a que las incidencias graves en la red son poco frecuentes, afortunadamente, y sin embargo estas personas deben estar suficientemente preparadas para decidir ágil y eficazmente en tales situaciones, dada la importancia creciente de la calidad en el suministro. En [81] se describen las funciones que desempeña el reparto de cargas en estos entrenadores, exigiéndosele fundamentalmente rapidez de respuesta.

2. PRINCIPALES ALGORITMOS PROPUESTOS.

2.1 INTRODUCCION.

La cantidad de algoritmos publicados para la solución del reparto de cargas, ha sido ingente. Comenzaron a aparecer hace casi treinta años, y desde entonces no ha menguado el interés de los investigadores en este tema o en aspectos relacionados, a juzgar por las publicaciones especializadas más recientes.

Prácticamente todos pueden englobarse en la siguiente clasificación:

- Métodos basados en la matriz de admitancias de nudos.
- Métodos basados en la matriz de impedancias de nudos.
- Variantes del método de Newton-Raphson.
- Métodos desacoplados.
- Métodos de segundo orden.
- Procesamiento en paralelo.
- Miscelánea.

Esta clasificación sigue casi exactamente un orden cronológico, con cierto solapamiento entre los diversos grupos, si exceptuamos lógicamente el apartado de miscelánea que engloba a lo no clasificable.

La inmensa mayoría tienen en común la formulación nodal de las ecuaciones, como se ha dado en el capítulo anterior, existiendo casi siempre dos versiones de un mismo algoritmo: la que utiliza coordenadas polares y la de coordenadas cartesianas.

Los métodos iniciales, basados en la matriz de admitancias de nudos, eran apreciados por la poca capacidad de memoria que necesitaban, y por la simplicidad de programación. Eran aplicaciones directas, al caso no lineal, de los procedimientos iterativos que se conocían en el análisis numérico para la solución de sistemas lineales. El más extendido de todos fue el de Gauss-Seidel.

En este grupo podemos incluir los algoritmos descritos en 2.2.1, 2.2.2 y 2.2.3.

La pobre convergencia de estos procedimientos no satisfacía en absoluto cuando se estudiaban sistemas medianamente grandes. Ello condujo a la aparición, casi en paralelo, de los métodos basados en la matriz de impedancias y el de Newton-Raphson, con la ayuda que significó el progresivo aumento de memoria de los computadores.

Los épigrafes 2.2.6, 2.2.7, 2.2.8, 2.2.9 y 2.2.10 describen algoritmos basados en la matriz de impedancias, mientras que diversas variantes del método de Newton se mencionan en 2.2.4, 2.2.5, 2.2.11, 2.2.15 y 2.2.18.

El método de Newton acabó imponiéndose a los basados en la matriz de impedancias, con la explotación de las técnicas de tratamiento de matrices vacias y ordenación óptima (ver capítulo 4), ya que la matriz de impedancias de nudos es llena.

La versión básica del método de Newton se describe en el Apéndice 2.

El esfuerzo, a partir de entonces, se concentró en mejorar la técnica de Newton-Raphson, lo que se logró sustancialmente al aplicársele ciertas propiedades de desacoplo del sistema, permitiendo reducir casi a la mitad el número de ecuaciones a resolver simultáneamente, y evitándose la triangularización del jacobiano en cada iteración. La culminación de estos intentos se conoce con el nombre de "flujo de cargas desacoplado rápido", algoritmo que hasta el presente no ha sido mejorado de forma clara, salvo en casos muy concretos.

En 2.2.12, 2.2.16, 2.2.17, 2.2.26 y 2.2.27 se incluyen los algoritmos más importantes de este grupo.

En lugar de atacar el problema por la vía de la simplificación, como en el caso anterior, se ha intentado hacerlo más recientemente por la vía de la complejidad, tomando un término más en el desarrollo en serie. De esa manera se han publicado numerosos métodos de segundo orden, buscando una mejor convergencia, pero sacrificando a cambio simplicidad de programación y esfuerzo de cálculo. Estos métodos se describen en 2.2.19, 2.2.20, 2.2.21, 2.2.24 y 2.2.25.

En los últimos años, con la progresiva utilización y abaratamiento de los microprocesadores y circuitos complejos VLSI, se están adaptando los algoritmos anteriores, de tipo estrictamente secuencial, al procesamiento en paralelo, lográndose con determi-

nadas arquitecturas tiempos muy bajos, comparables a los obtenidos con supercomputadores vectoriales, pero a un coste mucho menor. Se remite al lector al epígrafe 2.2.23.

Finalmente, algunos métodos, interesantes por su originalidad, no se pueden encuadrar en los grupos anteriores (lo que hemos denominado "miscelánea"). Tal es el caso de los descritos en 2.2.13 y 2.2.14.

2.2 REVISION HISTORICA.

En este epígrafe se pretenden condensar las ideas básicas de los algoritmos que a nuestro juicio pueden tener más relevancia. El criterio seguido para considerar un algoritmo como relevante es subjetivo, y por tanto sujeto a discusión, pero se ha intentado hacer una recopilación bastante completa, de forma que el lector saque una idea clara del estado de la técnica en esta materia y de su proceso evolutivo.

Se ha hecho especial énfasis en los métodos desacoplados, por el propio desarrollo de la tesis, así como en el método de segundo orden en coordenadas cartesianas que parece más rápido [73]. El método de Newton es ya clásico en los libros de texto, y se puede encontrar en el Apéndice 2.

2.2.1 WARD-HALE.

Históricamente, se está de acuerdo en reconocer que esta publicación de Junio de 1956 [120], constituye el primer método totalmente automático para la solución del flujo de cargas, utilizando un computador digital.

Partiendo de las ecuaciones nodales:

$$I_i = \sum_{j=1}^{N} Y_{ij} V_j \qquad i=1...N \qquad (2.1)$$

y de la expresión para la potencia compleja,

$$S_i = V_i I_i^* \qquad i=1...N \qquad (2.2)$$

llegamos a las siguientes ecuaciones

$$P_i^{esp} = \text{Real} \left\{ (V_i+dV_i)(I_i+Y_{ii}dV_i)^* \right\} \qquad (2.3)$$

$$|V_i^{esp}|^2 = |V_i+dV_i|^2 \qquad (2.4)$$

$$Q_i^{esp} = \text{Imag} \left\{ (V_i+dV_i)(I_i+Y_{ii}dV_i)^* \right\} \qquad (2.5)$$

Para los nudos P-V se utilizan la (2.3) y (2.4), mientras que para los nudos P-Q se utilizan la (2.3) y (2.5).

El complejo dV_i es la corrección en tensión que hace que se satisfaga cada pareja de ecuaciones correspondientes, permaneciendo constantes las tensiones de los otros nudos.

Partiendo de los mejores valores hasta el momento, se calcula I mediante (2.1), y a continuación se obtiene una corrección en tensión que anule la diferencia entre las magnitudes calculadas y las especificadas en (2.3), (2.4) ó (2.5).

El proceso se repite para cada nudo, y tantas veces como se requiera hasta obtener la precisión deseada.

Se puede acelerar el procedimiento multiplicando las correcciones por un factor, cuyo óptimo suele estar próximo a 1.4, aunque por supuesto depende del sistema. Factores excesivamente grandes, alrededor de 2, producen divergencia.

2.2.2 GLIMN-STAGG.

Es una simplificación del anterior (Octubre 1957). Los mejores valores de la tensión, se obtienen directamente como función de los valores previos, combinando (2.1) y (2.2) de la siguiente forma [45]:

$$V_{ii} = \frac{(P_i^{esp} - jQ_i^{esp})/V_i^* - \sum_{j \neq i} Y_{ij} V_j}{Y_{ii}} \qquad (2.6)$$

Para un nudo P-V no conocemos Q, y por tanto ésta se sustituye por la correspondiente calculada. Además, la tensión calculada de (2.6) debe ser multiplicada por el factor:

$$|V^{esp}| \; / \; |V|$$

para que el módulo sea el especificado.

Los resultados, en cuanto a iteraciones se refiere, son muy parecidos a los del método anterior con los mismos factores de aceleración.

En los libros de cálculo numérico este método se conoce con el nombre de Gauss-Seidel o desplazamientos sucesivos, presentando en general mejor convergencia que el método de Gauss, Gauss--Jacobi, o desplazamientos simultáneos, cuya diferencia radica en no aprovechar la parte de tensiones que ya se ha actualizado en la presente iteración, sino que se toma el valor de la anterior.

2.2.3 JORDAN.

Utiliza un método de relajación, de los que existen diversas variantes (1958). Se calcula I de (2.1) y de (2.2) y la diferencia forma el llamado residuo [59]. El proceso iterativo acaba cuando este residuo, que se utiliza para hacer las correcciones oportunas en las tensiones, es suficientemente pequeño. El orden en que se procesan las ecuaciones en cada iteración puede ser diferente, si optamos por eliminar el residuo mayor, lo cual hace la programación más complicada que en otros métodos que utilizan la matriz de admitancias de nudos. Al igual que en ellos, la presencia de nudos P-V empeora la convergencia. Aunque las técnicas de relajación no se han aplicado suficientemente al problema del flujo de cargas, no parecen tener ninguna ventaja. La experiencia realizada por Brown y Tinney [20], así lo demostraba al menos.

2.2.4 VAN-NESS.

Este autor aplicó las técnicas llamadas de matrices variacionales (Agosto 1959) [115], que más propiamente comenzaron a llamarse de Newton-Raphson, a partir de las aportaciones fundamentales de Tinney y otros [108].

Sustituyendo (2.1) en (2.2):

$$S_i = V_i \sum_{j=1}^{N} V_j^* Y_{ij}^* \tag{2.7}$$

se forman las diferenciales totales para dar las nuevas ecuaciones linealizadas en coordenadas polares:

$$\begin{aligned} \Delta P_i &= \sum_{j=1}^{N} H_{ij} \Delta \theta_j + \sum_{j=1}^{N} N_{ij} \Delta V_j \\ \Delta Q_i &= \sum_{j=1}^{N} J_{ij} \Delta \theta_j + \sum_{j=1}^{N} L_{ij} \Delta V_j \end{aligned} \tag{2.8}$$

que en forma matricial se escribe:

$$\begin{bmatrix} \Delta P_i \\ \Delta Q_i \end{bmatrix} = \begin{bmatrix} H_{ij} & N_{ij} \\ J_{ij} & L_{ij} \end{bmatrix} \cdot \begin{bmatrix} \Delta \theta_i \\ \Delta V_i \end{bmatrix} \tag{2.9}$$

Se itera como es usual y bien conocido en el método de Newton-Raphson (Apéndice 2).

Van Ness propone diversos métodos para no tener que invertir la matriz de coeficientes, y a la vez tener en cuenta su variación con las tensiones (estos métodos a su vez son iterativos).

2.2.5 VAN NESS-GRIFFIN.

Utilizan la ecuación (2.9) de la forma [116] :

$$\begin{bmatrix} \Delta P_i \\ \Delta Q_i \end{bmatrix} = \begin{bmatrix} H_{ij} & N_{ij} \\ J_{ij} & L_{ij} \end{bmatrix} \cdot \begin{bmatrix} \Delta \vartheta_i \\ \dfrac{\Delta V_i}{V_i} \end{bmatrix} \qquad (2.10)$$

en la que el vector independiente es adimensional. La importancia de este método radica en la solución de (2.10) por eliminación gaussiana en lugar de hacerlo iterativamente. Observaron, lo que se confirmó posteriormente, que el número de iteraciones era independiente del tamaño del sistema (Junio 1961).

2.2.6 HALE-GOODRICH.

En lugar de utilizar:

$$\begin{bmatrix} I_m \\ I_n \end{bmatrix} = \begin{bmatrix} Y_{mm} & Y_{mn} \\ Y_{nm} & Y_{nn} \end{bmatrix} \cdot \begin{bmatrix} V_m \\ V_n \end{bmatrix} \qquad (2.11)$$

donde m son los nudos PV y n los nudos PQ, se propone utilizar [49] :

$$\begin{bmatrix} I_m \\ V_n \end{bmatrix} = \begin{bmatrix} Y^1_{mm} & D_{mn} \\ D_{nm} & Z_{nn} \end{bmatrix} \cdot \begin{bmatrix} V_m \\ I_n \end{bmatrix} \qquad (2.12)$$

donde:

$$Y^{1}_{mm} = Y_{mm} - Y_{mn} \cdot Y^{-1}_{nn} \cdot Y_{nm}$$
$$D_{mn} = Y_{mn} \cdot Y^{-1}_{nn}$$
$$D_{nm} = -Y^{-1}_{nn} \cdot Y_{nm}$$
$$Z_{nn} = Y^{-1}_{nn}$$

El número de iteraciones es menor que en los métodos que utilizaban Y_{nudos}, pero el esfuerzo de cálculo es mayor (Octubre 1959).

2.2.7 BRAMELLER-DENMEAD.

Propusieron básicamente dos variantes (1962) [14]. Llamaron a la primera "sustitución hacia adelante". Si en las ecuaciones (2.1):

$$I_i = Y_{io}V_o + Y_{i1}V_1 + \ldots Y_{in}V_n \qquad (2.13)$$

en las que el subíndice 0 es para el nudo de referencia, obtenemos I'_i tal que:

$$I'_i = I_i - Y_{io}V_o = \sum_{j=1}^{N} Y_{ij}V_j \qquad i=1\ldots.N \qquad (2.14)$$

podemos invertir las (2.14) para dar:

$$V_i = \sum_{j=1}^{N} Z_{ij}I'_j \qquad i=\ldots N \qquad (2.15)$$

en donde I'_i se calcula de (2.2):

$$I'_i = S^{*}_i/V^{*}_i - Y_{io}V_o \qquad (2.16)$$

sustituyéndose en (2.15), y repitiéndose el proceso hasta obtener la convergencia.

El segundo método, una variante del anterior, es llamado "residuo cero de corriente".

Se sustituye (2.16) en (2.15):

$$V_i = Z_{i1}I_1' + \ldots Z_{ii}\left(\frac{S_i^*}{V_i^*} - Y_{io}V_o\right) + \ldots Z_{in}I_n' \qquad (2.17)$$

Multiplicando por V_i^* y agrupando términos:

$$V_i^{\,2} + A_i V_i^* - Z_{ii}S_i^* = 0 \qquad (2.18)$$

donde:

$$A_i = Z_{ii}Y_{io}V_o - \sum_{\substack{j=1 \\ j\neq i}}^{N} Z_{ij}I_j^1$$

De (2.18) se calcula V_i. Para los nudos PV, Q es desconocido y se necesita una tercera ecuación:

$$|V_i|^2 = \left[\mathrm{Re}(V_i)\right]^2 + \left[\mathrm{Im}(V_i)\right]^2 \qquad (2.19)$$

2.2.8 GUPTA-DAVIES.

La idea básica consiste en transformar las cargas y generaciones en admitancias que afectan a la diagonal de Y_{nudos} (Enero 1961) [46]. A partir de aquí, introduciendo la intensidad modificada I' del método anterior, llegamos a una ecuación similar a la (2.15) (con una Z diferente según las transformaciones mencionadas).

$$V_i = \sum_{j=1}^{N} Z_{ij} I'_j \qquad i=1...N \tag{2.20}$$

I'_i viene dada según (2.14) de:

$$I'_i = I_i - Y_{io} V_o \tag{2.21}$$

donde inicialmente I_i vale cero (no hay cargas ni generaciones) y en las siguientes iteraciones es una corriente residual, que tendrá en cuenta el hecho de que las tensiones sobre las que calculamos las admitancias de carga y generación eran estimadas, no exactas. Por tanto I_i se puede considerar como una corriente inyectada que en cada iteración produce las potencias requeridas, es decir:

$$\frac{S^*}{(V^k)^*} = -YV^k + I_i \tag{2.22}$$

donde Y es la carga o generación transformada en admitancia.

$$Y = -\frac{S^*}{|V|^2_{inic}} = -\frac{S^*}{1} \tag{2.23}$$

Sustiyendo (2.23) en (2.22):

$$I_i = \frac{S^*}{(V^k)^*}(1 - |V^k|^2) \qquad (2.24)$$

Proceso a seguir: Con (2.20) se obtienen unos mejores valores para las tensiones. Con (2.24) y (2.21) se obtienen las nuevas I', y así sucesivamente.

Para nudos P-V, se elige un valor para Q en función del de la iteración anterior:

$$Q^k = \frac{Q^{k-1}\,|V|^2}{|V^k|^2} \qquad (2.25)$$

2.2.9 BROWN-CARTER-HAPP-PERSON.

Se puede considerar una extensión del trabajo anterior (1963) [19]. Las cargas y generaciones se convierten en admitancias sólo para los nudos PQ, pero no para los PV. Además se invierte Y_{nudos} completa, en lugar de utilizar el concepto de intensidad modificada I'.

Por tanto, las ecuaciones que se utilizan son:

$$V_i = \sum Z_{ij} I_j \qquad i = 1....N \qquad (2.26)$$

con

$$I_i = (S_i/V_i)^* \qquad (2.27)$$

para nudos PV, y utilizándose la (2.24) para nudos PQ. Proceso a seguir: Se calculan las I_i de (2.27) ó (2.24). Se calcula I_o (referencia) de:

$$I_o = \frac{V_o - \sum_{j=1}^{N} Z_{oj} \cdot I_j}{Z_{oo}} \qquad (2.28)$$

Ahora se obtienen las nuevas tensiones de (2.26), etc. Las tensiones en los nudos PV se deben corregir para tener en cuenta su módulo exigido. Antes de calcular las nuevas I en los nudos PV, es preciso calcular la Q respectiva.

2.2.10 FRERIS-SASSON.

Su interés radica en ser una de las escasas aportaciones a la solución del flujo de cargas utilizando la matriz de impedancia de mallas (Octubre 1968) [35]. El comportamiento de esta formulación es muy semejante a la de las anteriormente mencionadas, basadas en la matriz de impedancia de nudos, puesto que, a pesar de la idea diferente de partida, con las manipulaciones que se hacen se obtienen ecuaciones prácticamente equivalentes.

A la vista de la velocidad de convergencia de los métodos basados en la matriz de admitancias y de impedancias, presentan un método híbrido, en el que se comienza iterando con el método

de las impedancias (5 iteraciones son suficientes) para finalizar con el método de las admitancias. Los resultados son muy discutibles.

Por último proponen un criterio, tras numerosos ensayos, para la elección del nudo de referencia en el método de las impedancias: Se tomará aquel nudo cuya impedancia propia (valor de la diagonal) sea menor, lo cual tiene cierta justificación desde el punto de vista del análisis numérico.

2.2.11 TINNEY-HART.

La contribución importante de este artículo (Noviembre 1967) [108], es mostrar la evidencia de que el método de Newton-Raphson, no sólo funciona bien, sino que es eficiente para problemas de gran dimensión. Esto no había quedado claro en la propuesta inicial de Van Ness y Griffin, que ya apuntaban el crecimiento desorbitado de memoria y tiempo conforme aumentaba el número de nudos, aunque mostraron que la convergencia era muy buena, resolviéndose problemas que no se lograban con el método de Gauss-Seidel o desplazamientos sucesivos.

En este artículo se describe la solución particular de los autores mediante el método de Newton-Raphson (véase Apéndice 2), utilizando las técnicas propuestas por Sato y Tinney [93], que combinaban eficientemente tres factores:

- Almacenamiento compacto de la matriz de admitancias y del jacobiano (altamente vacías).

- Reordenación de los nudos para evitar el posterior llenado del jacobiano en la...

- Eliminación gaussiana del mismo, que se realiza en cada iteración (en lugar de su inversión explícita).

Con este procedimiento se logró que el tiempo de ejecución y la capacidad de memoria requerida, creciese prácticamente proporcional al número de nudos, lo cual hizo que comenzase a hacerse popular y se extendiera su utilización.

A pesar de todo, el método de almacenamiento utilizado no permite la introducción fácil de elementos nuevos en el jacobiano.

El proceso iterativo propuesto también tiene un inconveniente, y es que no se detecta el que se ha conseguido la convergencia hasta el final de la iteración, lo que obliga a realizar una más de las necesarias estrictamente.

Ambos defectos se pueden subsanar fácilmente, y no empañan en absoluto el valor de estas aportaciones que revolucionaron este campo de los sistemas de potencia.

2.2.12 DESPOTOVIC-BABIC-MASTILOVIC.

Presentan el método (Enero 1971) [25], llamado así por los autores, de los "vectores de tensión", que constituye un antecedente claro de los métodos desacoplados posteriores, desde una óptica diferente al método de Newton-Raphson.

Lo describiremos brevemente con nuestra nomenclatura. Escribimos las ecuaciones básicas de la potencia activa y reactiva de la siguiente forma, en la que hemos utilizado una expresión alternativa para el coseno:

$$P_i = V_i \sum_{j=1}^{N} V_j B_{ij} \operatorname{Sen} \theta_{ij} + V_i \sum_{j=1}^{N} V_j G_{ij} \left[1-2\operatorname{Sen}^2 \frac{\theta_{ij}}{2} \right] \qquad (2.29)$$

$$\frac{Q_i}{V_i} = \sum_{j=1}^{N} V_j G_{ij} \operatorname{Sen} \theta_{ij} - \sum_{j=1}^{N} V_j B_{ij} \left[1-2\operatorname{Sen}^2 \frac{\theta_{ij}}{2} \right] \qquad (2.30)$$

Si aproximamos:

$$\begin{aligned} \operatorname{Sen} \theta &\simeq \theta - \theta^3/6 \\ \operatorname{Sen}^2 \theta &\simeq \theta^2/4 \end{aligned} \qquad (2.31)$$

la (2.29) y (2.30) se convierten en ecuaciones de la forma:

$$P_i = V_i \cdot \sum_j V_j B_{ij} \theta_{ij} + p^i \qquad (2.32)$$

$$\frac{Q_i}{V_i} = - \sum_j B_{ij} V_j + q^i \qquad (2.33)$$

en donde los términos p_i, q_i engloban las relaciones no lineales entre P-ϑ y Q-V respectivamente, esto es:

$$p_i = V_i \sum_{j=1}^{N} V_j \cdot G_{ij} - G_{ij} \frac{\vartheta_{ij}^2}{2} - B_{ij} \frac{\vartheta_{ij}^3}{6} \qquad (2.34)$$

$$q_i = \sum_{j=1}^{N} V_j \cdot G_{ij} \vartheta_{ij} - G_{ij} \frac{\vartheta_{ij}^3}{6} + B_{ij} \frac{\vartheta_{ij}^2}{2} \qquad (2.35)$$

Las (2.32) y (2.33) se pueden escribir en forma matricial así:

$$\begin{aligned} P - p &= T.\vartheta \\ \frac{Q}{V} - q &= U.V \end{aligned} \qquad (2.36)$$

con

$$\begin{aligned} T_{ij} &= - V_i V_j B_{ij} \qquad i \neq j \\ T_{ii} &= V_i \cdot \sum_{j \neq i} (V_j B_{ij}) \\ U_{ij} &= - B_{ij} \end{aligned} \qquad (2.37)$$

El proceso iterativo propuesto es el siguiente: Se calcula p,q de (2.34) y (2.35), se obtienen nuevas tensiones de (2.36), y se acaba cuando la diferencia entre p y q para dos iteraciones sucesivas es suficientemente pequeña.

Resulta extraño que los autores no hayan propuesto el algoritmo más lógico a primera vista, en el que se calcula p, se obtiene θ, con estos valores se calcula q y después V, etc., en lugar de la resolución simultánea de las dos ecuaciones de (2.36). En ese caso el método se parecería enormemente al propuesto posteriormente por Stott y Alsac .

2.2.13 GALLOWAY-TAYLOR-HOGG-SCOTT.

Aprovechan la siguiente idea (1970) [36]: En la resolución paso a paso de la respuesta transitoria de un sistema, se llega finalmente a un régimen permanente, que es la solución a un flujo de cargas.

Entonces en un nudo P-V, la restricción en potencia activa se expresa mediante la siguiente ecuación diferencial:

$$\frac{d^2\theta_i}{dt^2} = \frac{\Delta P_i}{H_i} \qquad (2.38)$$

siendo H una "inercia" artificial. Los nudos P-Q se pueden tratar de dos formas. Podemos convertir las cargas en admitancias shunt, que se añaden a las diagonales respectivas. Por otra parte es factible utilizar una ecuación como la (2.38) y otra análoga para Q-V.

$$\frac{d^2V_i}{dt^2} = \frac{\Delta Q_i}{G_i} \qquad (2.39)$$

Las ecuaciones se integran como se haría en un programa de cálculo de respuesta transitoria, en el que la "perturbación" inicial viene dada por el error en la estimación de las tensiones, que hace que $\Delta P, \Delta Q$ no sean nulas.

Para mejorar la convergencia y disminuir el número de pasos de la integración, se introduce un amortiguamiento ficticio en forma discontinua.

A pesar de ser ingenioso, el método no es en absoluto competitivo.

2.2.14 SASSON-TREVIÑO-ABOYTES.

Con anterioridad a este artículo (1971) [92], el primero de los autores había propuesto la resolución del flujo de cargas como un problema de minimización, donde la función objetivo era la suma de los cuadrados de ΔPi, ΔQi. El mínimo global de esta función era la solución buscada. La proposición tenía la virtud de que en muchos lugares ya se tienen rutinas de optimización, que pueden ser utilizadas para este problema ahorrando trabajo, pero no es ventajosa en cuanto a esfuerzo de cálculo, debido fundamentalmente a que las no linealidades culpables de que la solución deba ser iterativa, aumentan considerablemente al elevar al cuadrado. Es decir, conviene retener la linealidad en la medida de lo posible.

En vista de este defecto, los autores propusieron acelerar el método de Newton de la siguiente forma: En cada iteración se

traza una recta n-dimensional que une dos soluciones consecutivas, la cual pasará cerca de la solución si el proceso es convergente. Se hace entonces una búsqueda del mínimo mencionado anteriormente, a lo largo de esa variedad lineal, y una vez encontrado ese punto se toma como el valor acelerado del método de Newton.

Este algoritmo sólo es rentable si casualmente la recta pasa muy próxima a la solución, de otro modo la mejora obtenida con la aceleración no compensa en absoluto el tiempo invertido en la misma.

2.2.15 DUSONCHET-TALUKDAR-SINNOT-EL ABIAD.

Se dividen las ecuaciones del flujo de cargas en dos grupos (1971) [29]: barras PQ y barras PV.

Para el primer grupo, las ecuaciones se escriben de la forma:

$$AX = F_1(X) \qquad (2.40)$$

y utilizan el proceso iterativo de Gauss-Jacobi o desplazamientos simultáneos, al que ellos llaman Point-Jacobi, que resuelve (2.40) como:

$$AX^{n+1} = F_1(X^n) \qquad n = 0,1,2... \qquad (2.41)$$

Para el segundo grupo, nudos P-V, utilizan normalmente el método de Newton, es decir, si escribimos las ecuaciones respectivas como:

$$F_2(X) = 0 \tag{2.42}$$

se itera con:

$$J_n\,(\Delta X^{n+1}) = -\,F_2(X^n) \tag{2.43}$$
$$X^{n+1} = X^n + \Delta X^{n+1}$$

siendo:

$$J = \frac{\partial F_2(X)}{\partial X}$$

La matriz A tiene la ventaja respecto al jacobiano de que es constante, y por tanto no hay que recalcularla y triangularizarla cada vez.

Para el caso nuestro, las ecuaciones (2.43) ya se han escrito en (2.10) y están detalladas en los apéndices. Las ecuaciones (2.40) para los nudos PQ son:

$$\sum_{j=1}^{m} Y_{ij}V_j = \frac{P_i - jQ_i}{V_i^*} - \sum_{j=m+1}^{n} Y_{ij}V_j \qquad i = 1\ldots m \tag{2.44}$$

donde los primeros m nudos son de carga, y los últimos hasta n son regulados.

Las ecuaciones anteriores no son más que un caso particular de utilización de Z_{nudos}, en las que en lugar de obtener explícitamente dicha matriz (que no es vacía) se utiliza la tabla de factores de Y_{nudos}.

Cuando utilizamos Z_{nudos} en forma explícita, construida respecto al nudo de referencia en lugar de hacerlo respecto a masa, mediante desplazamientos sucesivos, la convergencia es buena. Por el contrario, al utilizarla en forma implícita según (2.40), la convergencia es mala debido fundamentalmente a los nudos regulados, de ahí que los autores propongan su utilización en la forma (2.44), y resuelvan los nudos regulados por otro procedimiento (en este caso el método de Newton).

El proceso iterativo a seguir sería partiendo de:

$$\begin{aligned} AX' &= F_1(X',X'') \\ 0 &= F_2(X',X'') \end{aligned} \qquad (2.45)$$

el siguiente:

$$\begin{aligned} AX'_{n+1} &= F_1(X'_n,X''_n) \\ J_n\Delta X'_{n+1} &= -F_2(X'_n,X''_n) \\ X''_{n+1} &= X''_n + \Delta X''_{n+1} \end{aligned} \qquad (2.46)$$

Con este algoritmo lo único que está claro es que se ahorra memoria, pero el tiempo es mayor que utilizando solamente el método de Newton, a pesar de los extraños resultados numéricos que presentan los autores.

2.2.16 STOTT.

El autor (Septiembre 1972) [102] presenta la primera mejora realmente significativa al método de Newton, desde que quedó definitivamente planteado por Tinney y sus colegas.

En este artículo, un método puramente matemático es enriquecido con el conocimiento físico del problema, y de ahí su éxito, que culmina con el publicado dos años después en una versión definitiva más rápida .

Se basa en el conocido desacoplo entre los módulos de las tensiones y los desfases, y su fuerte dependencia con la potencia reactiva y activa respectivamente, pudiéndose descomponer el problema de resolver:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix}^k = \begin{bmatrix} H & N \\ J & L \end{bmatrix}^k \cdot \begin{bmatrix} \Delta \theta \\ \Delta V \end{bmatrix}^{k+1} \qquad (2.47)$$

en la resolución de:

$$\Delta P^k = H^k \cdot \Delta \theta^{k+1} \qquad (2.48)$$

$$\Delta Q^k = L^k \cdot \Delta V^{k+1} \qquad (2.49)$$

debido a los pequeños valores numéricos de los términos de N y J. Ello ahorra obviamente capacidad de memoria, y probablemente tiempo de cálculo (dependiendo del caso concreto).

La ecuación (2.49) presenta a veces problemas de convergencia, a cierta distancia de la solución, y se ha encontrado más

conveniente utilizar los residuos de la parte imaginaria de la corriente inyectada, quedando el algoritmo en la forma:

$$\Delta P^k = H^k \cdot \Delta \vartheta^{k+1} \qquad (2.48)$$

$$\Delta I^m = D^m \cdot \Delta V^{m+1} \qquad (2.50)$$

El no escribir los mismos índices de iteración para (2.48) y (2.50) pretende indicar que no es preciso iterar estrictamente de forma consecutiva (2.48) y (2.50), sino que se pueden buscar políticas mejores.

Queda descartado iterar con (2.48) y (2.50) en forma simultánea, lo que no resultaría en absoluto competitivo. Conviene resolver (2.48) en ϑ, sustituir estos valores en el cálculo de ΔI y resolver (2.50) en V, o seguir estrategias similares de desplazamientos sucesivos en bloque.

Tras la primera iteración, los módulos de las tensiones difieren menos del 0,2% del valor verdadero (de media), sin que individualmente sobrepasen el 2%. Los primeros valores para las fases están a menos de 2° del valor verdadero (salvo para fases excesivamente grandes). Estos valores son incluso mejores que para el método de N-R formal, pero para precisiones muy finas se pierde la convergencia cuadrática propia del método.

El hecho de poder iterar por separado tantas veces como se quiera para ϑ y V, facilita y mejora el tratamiento de los ajus-

tes a la solución, como tomas de transformadores, límites de reactiva, etc., ya que estos ajustes afectan casi siempre a uno de los dos subproblemas (activa o reactiva).

2.2.17 STOTT-ALSAC.

Se describe el método (1974) [103] que posteriormente ha pasado a llamarse en los libros de texto "flujo de cargas desacoplado y rápido", como los autores titulan este artículo.

A partir de esta aportación, que probablemente no ha sido superada de forma definitiva hasta el momento, se pudo pensar en resolver problemas de cierto tamaño en tiempo real.

Partiendo de las ecuaciones básicas (2.10) repetidas aquí:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} H & N \\ J & L \end{bmatrix} \cdot \begin{bmatrix} \Delta \vartheta \\ \Delta V/V \end{bmatrix} \qquad (2.51)$$

y aplicando el principio de desacoplo descrito por Stott en el artículo anterior, llegamos a:

$$\Delta P = H \Delta \vartheta \qquad (2.52)$$

$$\Delta Q = L \cdot \Delta V/V \qquad (2.53)$$

Podemos introducir nuevas simplificaciones, justificadas desde el punto de vista físico. En concreto se puede suponer que:

$$\cos \vartheta_{ij} \simeq 1$$

$$G_{ij} \operatorname{Sen} \vartheta_{ij} << B_{ij}$$

$$Q_i << B_{ii} V_i^2$$

con lo que (2.52) y (2.53) se aproximan por:

$$\Delta P = V.B'.V \ . \Delta \vartheta \qquad (2.54)$$

$$\Delta Q = V.B''.V \ . \Delta V/V \qquad (2.55)$$

Los elementos de B' y B" son los de la parte imaginaria de Y_{nudos} cambiada de signo (Apéndice 2).

Hacemos además las siguientes simplificaciones:

a) Omitir en B' la representación de aquellos elementos que afectan predominantemente al flujo de reactiva, como las reactancias shunt y tomas distintas de la nominal.

b) Omitir en B" el efecto de los transformadores con desplazamiento de fase.

c) Pasar los módulos de las tensiones al lado izquierdo de las ecuaciones, y hacer en (2.54) los módulos de las tensiones igual a 1 p.u. para eliminar la influencia de los flujos de reactiva al calcular .

d) Ignorar las resistencias al construir B', lo cual beneficia a la convergencia incluso más de lo que los autores proclaman.

Con las simplificaciones mencionadas (2.54) y (2.55) quedan:

$$\Delta P/V = B' \Delta\theta \qquad (2.56)$$

$$\Delta Q/V = B'' . \Delta V \qquad (2.57)$$

Al ser B' y B" constantes, sólo necesitan construirse y triangularizarse una sola vez, y a partir de ese momento las iteraciones con (2.56) y (2.57) son muy rápidas. Además B" es simétrica, y lo mismo B' si los transformadores con desplazamiento de fase no existen, o se tienen en cuenta por otros medios, lo que implica mucha menos necesidad de memoria.

Tras los experimentos de los autores, el mejor esquema iterativo es aquél que alterna (2.56) y (2.57), lo que ellos llaman esquema (1θ,1V) habiéndose probado con menos éxito (2θ,1V), (2θ,2V), etc.

Un buen método de resolución del flujo de cargas no lo sería tal si no permitiese abordar con facilidad los problemas anexos al mismo, como son los ajustes a la solución, o el análisis de contingencias. Tras la exposición del algoritmo, los autores proponen la forma de abordar estos problemas complementarios desde su óptica, con lo cual el artículo en su conjunto resulta de sumo

valor, y es un ejemplo de lo que debe ser una publicación técnica (véase capítulo 3).

En los resultados numéricos se observa que los sistemas que requerían 3 ó 4 iteraciones, son resueltos con este algoritmo en 4 ó 4 1/2 iteraciones, y puesto que una iteración de este método desacoplado rápido, es unas 5 veces más rápida que la del N-R clásico, se comprende la ventaja del mismo, además del ahorro de memoria.

Se debe incluir aquí, porque constituye un buen complemento, la discusión que E. Hobson hizo del artículo [54]. En las pruebas que él realizó, utilizó la misma matriz para B' y B", lo que puede ser útil cuando se dispone de un ordenador pequeño. Para los nudos PV hacía $\triangle Q = 0$ y no se tenían en cuenta las correcciones para los módulos de las tensiones. Verdaderamente en algunos casos, como en redes de distribución, el efecto de las reactancias shunt es pequeño y B' y B" tienen valores numéricos muy parecidos.

Los resultados que obtuvo fueron satisfactorios, apareciendo problemas sólo cuando las tomas de los transformadores se alejaban excesivamente del valor nominal. Al mismo tiempo, se puso de manifiesto uno de los pocos inconvenientes que tiene el método desacoplado rápido respecto al convencional, y es el empeoramiento de la convergencia cuando existen líneas con una relación alta para R/X, debido a que algunas aproximaciones hechas dejan de ser válidas (véase capítulo 3).

2.2.18 LAHA-BOLLINGER-BILLINGTON-DHAR.

Utilizan una modificación al método de Newton (Agosto 1974) [63] propuesta por K.M. Brown en el año 1967. Esta técnica difiere del método convencional en la formulación del vector residual, que se formaba usando las soluciones estimadas en el paso anterior, y permanecía constante durante la etapa de substitución hacia atrás en la eliminación gaussiana. Con este algoritmo, los valores más recientes de la solución se utilizan para calcular el vector residual a la hora de resolver las siguientes incógnitas (se podría decir que se han incorporado los desplazamientos sucesivos al método de Newton).

Si tenemos el siguiente sistema de ecuaciones no lineales:

$$f_1 (X_1, X_2, \ldots X_n) = 0$$

$$\vdots$$

$$f_n (X_1, X_2, \ldots X_n) = 0$$

la idea básica consiste en:

- Hacer el desarrollo en serie de f_1 hasta los términos de primer orden, alrededor del valor estimado X_o (como se hace en el método de Newton).

- Despejamos de esa ecuación la variable cuya derivada parcial sea mayor que podemos suponer que es la x_1 , obteniéndose:

$$X_1 = b_1(X_2, X_3, \ldots X_n)$$

- Sustituimos ese valor de X_1 en f_2, que ahora tendrá una variable menos. Hacemos también el desarrollo en serie de f_2 y despejamos por ejemplo X_2 (cuya derivada parcial suponemos es la mayor):

$$X_2 = b_2(X_3, X_4 \ldots X_n)$$

- Repetimos el proceso, eliminando una variable cada vez, hasta quedarnos con una sola en la última función. Obtendremos así n funciones tales como:

$$\begin{aligned} g_1 &(X_1, X_2, \ldots X_n) = 0 \\ g_2 &(X_2, X_3, \ldots X_n) = 0 \\ &\vdots \\ g_n &(X_n) = 0 \end{aligned}$$

en donde $g_1 = f_1$.

Estas ecuaciones en forma linealizada quedarían:

$$H\ (X - X^o) = -G^o$$

donde H es triangular superior. Se puede demostrar que los elementos de H son idénticos a los obtenidos por eliminación gaussiana en el jacobiano de F. Por el contrario, el vector G^o difiere bastante del correspondiente al método de Newton, pues las funciones g_i^o se construyen con variables actualizadas, esto es:

$$g_i^o = f_i\ (X_1^{i-1}, X_2^{i-2}, X_3^{i-3}, \ldots X_n^{i-n})$$

con

$$X_k^{i-k} = X_k^o \qquad \text{para } i < K$$

y

$$X_k^j = X_k^o - \left(\frac{\partial g_k}{\partial X_k}\right)^{-1} \left\{ \sum_{i=k+1}^{m=k+j} \frac{\partial g_k}{\partial X_i} \left(X_i^{m-i} - X_i^o\right) + g_k^o \right\}$$

Curiosamente, ocho años después, Tripathy y otros publicaron el mismo algoritmo [112]. La ventaja respecto a éste era que mostraban más datos comparativos, y hacían hincapié en su utilización cuando otros métodos tenían problemas de convergencia. De los resultados mostrados, este algoritmo puede ser competitivo con el método de Newton, ya que cada iteración sólo tarda un 15% más, pero sólo puede aventajar al método desacoplado rápido cuando éste no converge, como sucede con varios ejemplos que han ido surgiendo en la literatura.

2.2.19 SACHDEV-MEDICHERLA.

Constituye este artículo (1977) [89] el comienzo de una serie de ellos que llega hasta la actualidad, y que tienen en común la utilización de un término más en el desarrollo en serie, es decir son métodos de segundo orden. Precisamente éste, por ser el primero no resultó afortunado.

Quedándonos con ese término más, las ecuaciones en cada iteración son ahora:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \cdot \begin{bmatrix} \Delta\vartheta \\ \frac{\Delta V}{V} \end{bmatrix} + \begin{bmatrix} S_1 & S_2 & S_3 \\ S_4 & S_5 & S_6 \end{bmatrix} \cdot \begin{bmatrix} \Delta\vartheta \cdot \Delta\vartheta \\ \Delta V \cdot \Delta V \\ \Delta\vartheta \cdot \Delta V \end{bmatrix} \qquad (2.58)$$

Los valores de los coeficientes se detallan en la publicación original .

Una buena parte de esos coeficientes son numéricamente pequeños y se pueden despreciar. El resto se calculan simplemente en función de los términos del jacobiano, o son iguales, por lo que no suponen un esfuerzo de cálculo excesivo.

Tampoco necesitan capacidad de memoria adicional, como veremos a continuación en la descripción del algoritmo. Las ecuaciones (2.58) son no lineales al aparecer los productos $\Delta\vartheta\,\Delta V$, $\Delta\vartheta\Delta\vartheta$, $\Delta V\Delta V$ y se utilizará el siguiente procedimiento:

1) Calcular ΔP, ΔQ.
2) Ignorando los términos de segundo orden, obtener $\Delta\vartheta$, $\Delta V/V$ (lo que supone una iteración del Newton-Raphson).
3) Utilizando $\Delta\vartheta, \Delta V$ del paso anterior, al calcular los términos de segundo orden en la siguiente ecuación:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} - \begin{bmatrix} S_1 & S_2 & S_3 \\ S_4 & S_5 & S_6 \end{bmatrix} \cdot \begin{bmatrix} \Delta\vartheta\Delta\vartheta \\ \Delta V \Delta V \\ \Delta\vartheta\Delta V \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \cdot \begin{bmatrix} \Delta\vartheta \\ \frac{\Delta V}{V} \end{bmatrix} \qquad (2.59)$$

recalcular el vector $\Delta\theta$, $\Delta V/V$ utilizando el mismo jacobiano triangularizado del paso 2).

4) Realizar el paso 3) hasta que la diferencia entre dos soluciones consecutivas sea suficientemente pequeña, utilizando en el vector residual modificado, los $\Delta\theta$,$\Delta V/V$ de la solución anterior de (2.59).
5) Actualizar las tensiones con los últimos $\Delta\theta$, ΔV.
6) Calcular ΔP, ΔQ, y si no se ha obtenido convergencia volver al paso 2.

Los resultados obtenidos por este procedimiento, ahorran a lo sumo una iteración, o probablemente ninguna, respecto al método de primer orden, por lo que su eficacia resulta dudosa si se tiene en cuenta el proceso sub-iterativo que se requiere entre cada iteración, a pesar de que el cálculo de los coeficientes de segundo orden sea fácil y rápido.

En vista de ello los propios autores propusieron diferentes alternativas, que tenían en común la eliminación del proceso iterativo 4), pero que seguían sin demostrar una clara ventaja respecto a métodos más antiguos, como el desacoplado rápido.

La mejor alternativa parece ser aquella en que, además de eliminar el proceso iterativo 4), los términos de segundo orden sólo se tienen en cuenta en la primera iteración, puesto que luego su efecto resulta insignificante. De ese modo, excepto la primera, las demás iteraciones son idénticas al método de Newton.

2.2.20 IWAMOTO-TAMURA.

El método propuesto (Septiembre 1978) [57] hace uso del hecho de que las ecuaciones del flujo de cargas son cuadráticas, cuando se expresan en coordenadas cartesianas. Por tanto, en el desarrollo en serie, no existen términos a partir del tercero. Las ecuaciones básicas del flujo de cargas en coordenadas cartesianas se escriben como:

$$P_i = \sum_{j=1}^{N} (e_i e_j G_{ij} + e_i f_i B_{ij} + f_i f_j G_{ij} - f_i e_j B_{ij}) \qquad (2.60)$$

$$Q_i = \sum_{j=1}^{N} (f_i e_j G_{ij} + f_i f_j B_{ij} - e_i f_j G_{ij} + e_i e_j B_{ij}) \qquad (2.61)$$

$$|V_i|^2 = e_i^2 + f_i^2 \qquad (2.62)$$

Los miembros de la izquierda son las magnitudes especificadas, que llamaremos Y_s. El problema del flujo de cargas consiste en resolver el sistema cuadrático de ecuaciones:

$$Y_s = A \cdot \begin{bmatrix} X_1 X_1 \\ X_1 X_2 \\ \cdot \\ \cdot \\ \cdot \\ X_n X_n \end{bmatrix} \qquad (2.63)$$

donde X_i son las incógnitas e, f para cada nudo.

Expresando cada valor verdadero X_i por el estimado X_e más una corrección ΔX, se tiene para cada producto $X_i . X_j$:

$$X_i X_j = (X_{ei} + \Delta X_i)(X_{ej} + \Delta X_j) = \\ = X_{ei}X_{ej} + X_{ei}\Delta X_j + \Delta X_i X_{ej} + \Delta X_i \Delta X_j \qquad (2.64)$$

Con lo que (2.63) se descompone en 3 sumandos. El primero es precisamente $Y(X_e)$. El segundo (suma del segundo y el tercero de (2.64)) se puede identificar como $J\Delta X$, siendo J el jacobiano. Y el tercero es análogo al primero pero tomando como variable ΔX en lugar de X_e. Por tanto (2.63) queda:

$$Y_s = Y(X_e) + J\Delta X + Y(\Delta X) \qquad (2.65)$$

que es una formulación equivalente y simplificada del desarrollo en serie exacto hasta el tercer término, alrededor del valor estimado X_e:

$$Y_s = Y(X_e) + J\Delta X + \frac{1}{2} H \begin{bmatrix} \Delta X_1 \Delta X_1 \\ \Delta X_1 \Delta X_2 \\ \vdots \\ \Delta X_n \Delta X_n \end{bmatrix} \qquad (2.66)$$

La ecuación (2.65) se utiliza para iterar como:

$$J\Delta X^{r+1} = Y_s - Y(X_e) - Y(\Delta X^r) \qquad (2.67)$$

En (2.67) $Y_s - Y(X_e)$ es un vector constante, y lo mismo ocurre con J que está calculado para X_e, y que por tanto sólo necesita ser triangularizado una vez al comienzo. Partiendo de $\Delta X = 0$ se utiliza (2.67) sucesivamente para actualizar ΔX. El proceso fina-

liza cuando ΔX^n es igual a ΔX^{n-1}, salvo una tolerancia pequeña, en cuyo caso la solución es $X_e + \Delta X^n$.

Se puede demostrar, como se hace en las discusiones, que el método propuesto se comporta exactamente igual que el método de Newton con pendiente constante, y por tanto requiere más iteraciones (al menos dos más), siendo necesaria una buena estimación inicial.

El método es más rápido que el método de Newton, pero no tanto como los autores postulan, y además necesita más memoria para almacener la tabla de factores del jacobiano (que por cierto es de dimensión mayor que en coordenadas polares).

No existen comparaciones con el método desacoplado rápido , aunque es obvio que el método propuesto es más lento y ocupa mucha más memoria.

2.2.21 ROY.

Esta discusión del artículo anterior merece la pena mencionarla por separado.

El jacobiano, calculado para una estimación inicial de $1\angle 0^\circ$, no es simétrico totalmente, como ocurre en coordenadas polares. Los únicos términos que introducen asimetría son:

$$\frac{\partial P_i}{\partial f_i} = B_{ii} + d_i$$
$$\frac{\partial Q_i}{\partial e_i} = B_{ii} - d_i \qquad (2.68)$$

siendo d la parte imaginaria de la corriente nodal. En lugar de escribir la ecuación (2.67) como:

$$\begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \cdot \begin{bmatrix} \Delta e \\ \Delta f \end{bmatrix} = \begin{bmatrix} \Delta P' \\ \Delta Q' \end{bmatrix} \qquad (2.69)$$

donde $\Delta P'$, $\Delta Q'$ son los residuos modificados de dicha ecuación, podemos sustituir la diagonal i-ésima de J_2 por la diagonal i-ésima de J_3 más $2d_i$, puesto que:

$$\frac{\partial P_i}{\partial f_i} = B_{ii} + d_i = \frac{\partial Q_i}{\partial e_i} + 2d_i$$

Con ello (2.69) se puede escribir como:

$$\begin{bmatrix} J_1 & J_3 \\ J_3 & J_4 \end{bmatrix} \cdot \begin{bmatrix} \Delta e \\ \Delta f \end{bmatrix} = \begin{bmatrix} \Delta P' - 2d\Delta e \\ \Delta Q' \end{bmatrix} \qquad (2.70)$$

en donde el jacobiano ya es simétrico, ahorrándose con ello la mitad de memoria requerida para su almacenamiento, y resultando más rápida cada iteración.

La no simetría que introducen otras causas (nudos PV, tomas de transformadores con desplazamiento, etc.) se puede eliminar por manipulaciones parecidas.

Este autor, además, comprobó que la introducción de factores de aceleración perjudicaba a la convergencia. En [86] insistió básicamente sobre las mismas ideas.

2.2.22 SAUER.

Como las ecuaciones del flujo de cargas en coordenadas rectangulares, son ecuaciones cuadráticas de las potencias en función de las tensiones, se estudian en este artículo (Agosto 1981) [94] dos desarrollos en serie de las tensiones en función de las potencias, por analogía con la ecuación cuadrática escalar. La primera es la serie de Taylor, mientras que la segunda es una serie obtenida de una sucesión de valores, que convergen al punto fijo dado por:

$$X = f(X) \tag{2.71}$$

Esta última serie, además, se puede demostrar que tiene una convergencia similar a la del método de Newton-Raphson con jacobiano constante.

Prescindiremos de la serie de Taylor, que tiene peor convergencia. Si escribimos la ecuación cuadrática escalar en la forma (2.71), ésto es:

$$X = b^{-1}.(c - aX^2) \tag{2.72}$$

el problema de obtener la solución de (2.72), es el de hallar el punto fijo, que es precisamente el límite de la sucesión.

$$X,\ f(X),\ f(f(X)),\ f(f(f(X)))\ \ldots.$$

y que para nuestro caso con x(0) = 0 sería:

$$\begin{aligned} X(1) &= b^{-1}C \\ X(2) &= b^{-1}\left[c - a(b^{-1}c)^2\right] \\ X(3) &= b^{-1}\left[c - a(b^{-1}(c-a(b^{-1}c)^2))^2\right] \end{aligned} \tag{2.73}$$

Dicha solución se puede expresar por una serie de la forma:

$$X = b^{-1}\sum_{i=1}^{\infty} c(i) \tag{2.74}$$

con

$$\begin{aligned} C(1) &= C \\ &\vdots \\ C(i) &= b\left[X(i) - X(i-1)\right] \end{aligned} \tag{2.75}$$

Expresando c(i) en forma recursiva, resulta:

$$C(i) = C(i-1)b^{-1}ab^{-1}C(i-1) - 2\sum_{j=1}^{i-1} C(i-1)b^{-1}ab^{-1}C(j) \tag{2.76}$$

Las ecuaciones del flujo de cargas en forma cuadrática se pueden expresar como:

$$P = JV + \begin{bmatrix} V^t H_1 V \\ \vdots \\ V^t H_n V \end{bmatrix} \tag{2.77}$$

Se puede encontrar una serie equivalente a la (2.74) del caso escalar (si suponemos que J es no singular). Escribiendo (2.77) en la forma (2.72):

$$V = J^{-1} \cdot \left(P - \begin{bmatrix} V^t H_1 V \\ \vdots \\ V^t H_n V \end{bmatrix} \right) \tag{2.78}$$

la serie buscada viene dada por:

$$V = J^{-1} \sum_{i=1}^{\infty} P(i) \tag{2.79}$$

con

$$\begin{aligned} P(1) &= P \\ &\vdots \\ P(i) &= J \left[V(i) - V(i-1) \right] \end{aligned} \tag{2.80}$$

siendo la expresión recursiva de P(i):

$$P(i) = \begin{pmatrix} P(i-1)^t J^{-t} H_1 \\ \vdots \\ P(i-1)^t J^{-t} H_n \end{pmatrix} \cdot \left(J^{-1} P(i-1) - 2 \sum_{j=1}^{i-1} J^{-1} P(j) \right) \tag{2.81}$$

Como ha quedado dicho, la convergencia de la serie (2.79) es la misma que la del método de Newton con jacobiano constante, que viene dado por:

$$\begin{aligned} J\Delta V^{(m)} &= \Delta P = P^{esp} - P^{cal(m)} \\ V^{(m+1)} &= V^{(m)} + \Delta V^{(m)} \end{aligned} \qquad (2.82)$$

Una mirada a las ecuaciones (2.82) y (2.80) nos aclara que la serie (2.79) consta de sumandos P(i) que son precisamente los residuos de las potencias, puesto que:

$$\Delta P(m+1) = J\Delta V^{m} = P^{esp} - P^{cal(m)} \qquad (2.83)$$

Por tanto, cuando cada término del vector P(m) satisfaga el criterio de convergencia, la serie habrá convergido a esa tolerancia.

Con las ecuaciones en la forma anterior, sólo se encontraba convergencia para tensiones próximas a cero, lo que no resultaba útil, y por tanto se redefinieron las tensiones con respecto a la del nudo de referencia:

$$V'_p = V_p - V_{ref}$$

eliminándose ya esos problemas con el nuevo vector de tensiones V'.

Una ventaja de la serie (2.79), es que una vez obtenida una solución para una P dada, se puede obtener con mínimos cálculos la solución para otra P que sea un porcentaje determinado de la anterior (si no cambia la topología de la red) almacenando previamente una serie de coeficientes.

2.2.23 PRITCHARD-POTTLE.

En este artículo, (Enero 1982) [82] se describe la realización del flujo de cargas por el método desacoplado rápido, utilizando una arquitectura especial.

Dicha arquitectura consiste en un procesador especializado en operaciones aritméticas, soportado por un miniordenador, y se presenta como alternativa más económica que los grandes ordenadores vectoriales de desarrollo reciente (CRAY-1, STAR, etc.). Estos ordenadores de costo desorbitado, no se pueden utilizar con toda su eficiencia en el tratamiento de las matrices vacias, y este artículo pretende demostrar la competitividad de otras alternativas en los problemas a gran escala de los sistemas de potencia.

En paralelo al desarrollo de los ordenadores vectoriales, han aparecido en el mercado los llamados "array processors", diseñados para proveer de cálculo en punto flotante a alta velocidad al ordenador al que se conectan. Algunos de ellos, como el utilizado en este caso, son programables (AP- 120B).

Constan de un sumador y un multiplicador flotantes con estructura "pipe- line", (de 2 y 3 segmentos respectivamente), numerosos registros para resultados intermedios, y tres zonas de memoria para almacenamiento de datos e instrucciones. Estos constituyentes están unidos por diferentes caminos que se pueden activar o desactivar por programa, de forma que con un código apropiado es factible realizar en paralelo tareas aparentemente no relacionadas. Por tanto el paralelismo en esta máquina se obtiene directamente por el programa del usuario. Su principal limitación es la velocidad de comunicación de datos con el "host computer", y la dificultad de escribir un código optimizado para él.

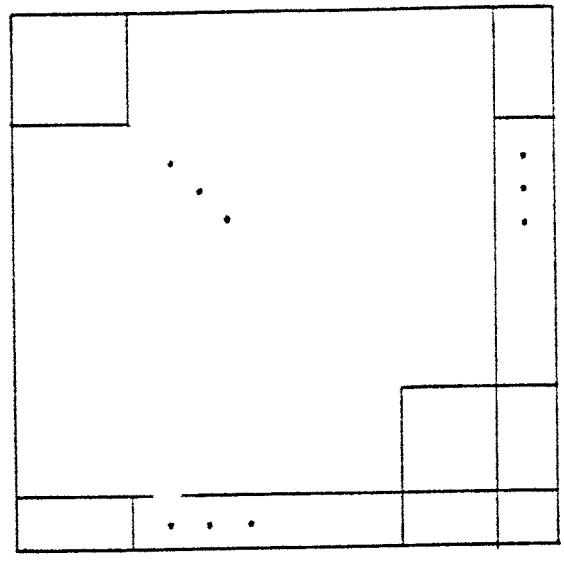
Con la introducción de esta arquitectura paralela, todas las decisiones de carácter algorítmico tomadas para una máquina serie, deben ser reexaminadas. Asímismo queda por decidir qué parte del trabajo hace el computador principal, y cual hace el "array processor". Debido a los problemas de comunicación mencionados, y a tratarse de un miniordenador, se opta por utilizar este último lo menos posible en el proceso iterativo.

Para el sistema de 118 nudos del IEEE se obtiene así un tiempo de 25 mseg. (sólo para el proceso iterativo, y no para la entrada y salida de datos), y se estima por extrapolación que un sistema de 1000 nudos se podría resolver en 0,5 seg.

Puesto que para ese sistema de 118 nudos, se requieren 16 mseg. para transferir los datos necesarios al procesador (con un miniordenador), tanta velocidad de cálculo lograda no tendría excesivo sentido para resolver casos simples de flujo de cargas,

sino en aplicaciones en que dicha rutina formase parte de problemas más complejos, como en un flujo de cargas óptimo para estudios de planificación, o determinación de la seguridad de funcionamiento (análisis de contingencias [55]).

Finalmente, como la estructura tan sofisticada del "array processor" no es aprovechada al máximo, se apunta la posibilidad de utilizar varios circuitos más sencillos y económicos (por ejemplo sin estructura pipe-line) realizados con tecnología VLSI, conectados a un bus común y disponiendo cada uno de una memoria local convencional. Se necesitaría en ese caso una ordenación de los nudos que permitiese asignar una parte del sistema a cada procesador, procurando que la parte en común de todos los subsistemas sea mínima, para aumentar el paralelismo al máximo (véanse capítulos 4 y 6).



2.2.24 El-HAWARY, WELLON.

Se proponen. (Abril 1982) [34] una familia de métodos de segundo orden que difieren entre sí en el valor de un parámetro α. Estudian la convergencia de los diferentes métodos en función de dicho parámetro.

El desarrollo en serie hasta los términos de segundo orden de las ecuaciones del flujo de cargas en coordenadas cartesianas, se puede escribir en forma general así para uno de sus componentes:

$$\Delta f_i = J_i \Delta X + \frac{1}{2} (\Delta X)^t H_i \Delta X \qquad i=1,2... \qquad (2.84)$$

siendo J_i la fila i-ésima del jacobiano, y H_i la matriz hessiana para esa componente f .

$$\frac{\partial^2 f_i}{\partial X_j \partial X_k}$$

La idea básica consiste en reescribir (2.84) en la forma equivalente:

$$\Delta f_i - \alpha_i \left(\frac{1}{2} (\Delta X)^t H_i \Delta X \right) = \left[J_i + (1-\alpha_i)\left(\frac{1}{2}(\Delta X)^t H_i\right)\right]\Delta X \qquad (2.85)$$

Si $\alpha = 1$ toda la corrección debida a los términos de segundo orden afecta a los residuos Δf, mientras que con $\alpha = 0$ la influencia se ejerce exclusivamente sobre el jacobiano.

El proceso iterativo propuesto es semejante al de Sachdev-Medicherla [89], descomponiéndose cada iteración en dos semiiteraciones. Primero se calcula ΔX sin tener en cuenta los términos de segundo orden. A continuación, con los valores ΔX obtenidos, se corrigen los residuos y el jacobiano como indica (2.85), y se obtienen los mejores valores ΔX.

Sus resultados muestran que la mejor convergencia, sucede con $\alpha = 0$, necesitándose la mitad de iteraciones, o algo más, que para el método de Newton-Raphson.

Sin embargo, se necesita mucha más memoria, puesto que es preciso almacenar el jacobiano, además de su tabla de factores, para hacerle las correcciones posteriores, y tampoco se gana nada en tiempo, pues cada semiiteración requiere la factorización del jacobiano, y por tanto lleva prácticamente los mismos cálculos que una iteración del método de Newton-Raphson.

Con $\alpha = 1$, tenemos las ecuaciones utilizadas por el método de Iwamoto Tamura [57], que es mucho más competitivo.

Con $\alpha = 0$ se tiene una de las variantes propuestas por Roy años atrás [86], con algunas diferencias que la hacían más ventajosa.

2.2.25 NAGENDRA-PRAKASA-NANDA.

Puesto que ninguno de los métodos de segundo orden propuestos hasta entonces había mostrado una ventaja clara, los autores presentan una versión mejorada (Septiembre 1982) [73], tomando como base las coordenadas cartesianas, simplemente introduciendo algunas modificaciones. Además, proponen una nueva forma de manejar los nudos PV cuyos límites de reactiva son sobrepasados.

Se parte de las ecuaciones fundamentales en coordenadas rectangulares, dadas por (2.60),(2.61),(2.62).

Supongamos de entrada que no existen nudos PV. Realicemos las siguientes operaciones:

1) Todas las conexiones shunt en cada nudo, se consideran como cargas de impedancia constante, en lugar de incluirlas en Y_{nudos}.
2) Los valores iniciales de las tensiones se hacen igual a la del nudo de referencia (e_s + j0).

Entonces el siguiente desarrollo en serie exacto, equivalente a (2.65):

$$\begin{bmatrix} P^{esp} \\ Q^{esp} \end{bmatrix} = \begin{bmatrix} P^{o} \\ Q^{o} \end{bmatrix} + J. \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} + \begin{bmatrix} SP \\ SQ \end{bmatrix} \qquad (2.86)$$

donde SP, SQ son los términos de segundo orden, y P^{o} y Q^{o} son los valores de P y Q para las tensiones estimadas, se puede escribir

como:

$$\begin{bmatrix} P' \\ Q' \end{bmatrix} = J \cdot \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} + \begin{bmatrix} SP \\ SQ \end{bmatrix} \qquad (2.87)$$

habiendo desaparecido P^o , Q^o debido a los valores iniciales tomados según 2) y definiéndose:

$$P_i' = P_i^{esp} - g_i(e_i^2 + f_i^2) \qquad (2.88)$$

$$Q_i' = Q_i^{esp} - b_i(e_i^2 + f_i^2) \qquad (2.89)$$

según 1). El jacobiano en (2.87) no es el mismo que en (2.86), puesto que no se incluyen las conexiones shunt g+jb en cada nudo, las cuales se tienen en cuenta en (2.88), (2.89).

Precisamente el jacobiano, debido a las modificaciones hechas, tiene una forma particular, escribiéndose (2.87) como:

$$\begin{bmatrix} P' \\ Q' \end{bmatrix} = e_s \begin{bmatrix} B & G \\ -G & B \end{bmatrix} \cdot \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} + \begin{bmatrix} SP \\ SQ \end{bmatrix} \qquad (2.90)$$

Si definimos los residuos como:

$$\begin{bmatrix} RP \\ -RQ \end{bmatrix} = \begin{bmatrix} P' \\ Q' \end{bmatrix} - \begin{bmatrix} SP \\ SQ \end{bmatrix} \qquad (2.91)$$

donde el signo menos de RQ sirve para hacer simétrico el jacobiano, la (2.90) resulta:

$$\begin{bmatrix} RP \\ RQ \end{bmatrix} = e_s \begin{bmatrix} B & G \\ G & -B \end{bmatrix} \cdot \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} \qquad (2.92)$$

La ecuación (2.92) es muy interesante pues permite, teniendo en cuenta los valores de SP, SQ, obtener las siguientes relaciones:

$$SP_i = e_i.(RP_i/e_s) + f_i(RQ_i/e_s) \qquad (2.93)$$

$$SQ_i = f_i.(RP_i/e_s) - e_i(RQ_i/e_s) \qquad (2.94)$$

que son las aportaciones más importantes del artículo, pues permiten calcular fácilmente los términos de segundo orden, sin tener que almacenar los términos del jacobiano.

Los nudos PV se incorporan a (2.92) utilizando (2.62), sin que entremos aquí en detalles del procedimiento a seguir.

Con las consideraciones anteriores, el algoritmo propuesto es el siguiente (para una solución no ajustada).

1) Formar el jacobiano, triangularizarlo y almacenar su tabla de factores. Tomar inicialmente como cero los términos de segundo orden.
2) Calcular P' y Q' de (2.88) y (2.89).
3) Restarle el vector de términos de segundo orden para obtener RP, RQ.
4) Calcular de (2.92) las correcciones Δe y Δf.

5) Calcular los términos de segundo orden de (2.93) y (2.94) e investigar la convergencia. Si no ha habido convergencia ir a 2).

Se puede demostrar que la diferencia entre los términos de segundo orden para una iteración, es igual a ΔP, ΔQ al finalizar la iteración, de ahí que se pueda analizar la convergencia en el paso 5).

La forma de manejar los nudos PV cuyos límites son sobrepasados, es muy semejante a la propuesta por Stott y Alsac (veáse capítulo 3).

Al final se hace una excelente comparación entre el método propuesto y los ya existentes.

El método es más rápido y necesita menos memoria que los de Iwamoto- Tamura y Roy, pero no está clara la comparación con el de Stott y Alsac, aunque parece dar mayor fiabilidad, al no hacer ningún tipo de aproximaciones, en sistemas que convergen mal (sistemas con líneas de alta relación R/X o ramas capacitivas).

2.2.26 BABIC.

En este breve artículo (Mayo 1983) [6], se tratan diversas posibilidades de desacoplo en coordenadas rectangulares, indicándose cuál puede ser la más apropiada.

Además de la expresión clásica de residuos en P y Q:

$$\begin{bmatrix} \frac{\partial(\Delta P)}{\partial f} & \frac{\partial(\Delta P)}{\partial e} \\ \frac{\partial(\Delta Q)}{\partial f} & \frac{\partial(\Delta Q)}{\partial e} \end{bmatrix} \cdot \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} = - \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \qquad (2.95)$$

se puede utilizar para el modelo desacoplado, al igual que se ha hecho en coordenadas polares, los residuos en intensidades (en su parte activa y reactiva):

$$\begin{bmatrix} \frac{\partial(\Delta I_a)}{\partial f} & \frac{\partial(\Delta I_a)}{\partial e} \\ \frac{\partial(\Delta I_r)}{\partial f} & \frac{\partial(\Delta I_r)}{\partial e} \end{bmatrix} \cdot \begin{bmatrix} \Delta f \\ \Delta e \end{bmatrix} = - \begin{bmatrix} \Delta I_a \\ \Delta I_r \end{bmatrix} \qquad (2.96)$$

Tomando como partida (2.95) ó (2.96) y haciendo diferentes simplificaciones, llegamos a ecuaciones desacopladas de la forma:

$$\begin{aligned} B_{11}\ \Delta f &= - b_1 \\ B_{22} \cdot \Delta e &= - b_2 \end{aligned} \qquad (2.97)$$

Dependiendo del grado de simplificación, y de la combinación que se haga entre (2.95) y (2.96), se obtienen hasta 14 posibilidades diferentes que aparecen tabuladas.

Se tabulan asímismo 10 posibilidades de desacoplo en coordenadas polares, partiendo de ecuaciones equivalentes a (2.95) y (2.96).

Se muestran resultados parciales para tres sistemas diferentes, y la conclusión es que la forma de desacoplo más eficiente (la que necesita menor tiempo sin perder fiabilidad) en coordenadas cartesianas es:

$$-B\ \Delta f = \frac{\Delta P}{e}$$
$$-B\ \Delta e = \frac{\Delta Q}{e} \qquad (2.98)$$

y en coordenadas polares la propuesta por Stott-Alsac, y que fue la originaria de todos estos trabajos:

$$-B\ \Delta \theta = \frac{\Delta P}{V}$$
$$-B\ \Delta V = \frac{\Delta Q}{V} \qquad (2.99)$$

El mayor énfasis del artículo se hace en destacar que (2.98) es la mejor forma de desacoplo en coordenadas rectangulares, pero no deja claro qué tipo de coordenadas es más conveniente utilizar, aunque por los resultados (en absoluto estándars) parece indistinto.

2.2.27 HALEY-AYRES.

Se va aún más lejos en el concepto de desacoplo (superdesacoplo), intentando obviar el inconveniente que presentan los métodos desacoplados, que suponen que la relación X/R es alta. En los casos en que ésto no es cierto la convergencia empeora (véase capítulo 3). La hipótesis de X/R alta, se sustituye aquí por la menos restrictiva de que las impedancias de los enlaces entre nudos permanezcan en un cuadrante del plano complejo.

El método consiste [50] en aplicar operadores (unitarios) de rotación a los vectores $[\Delta P, \Delta Q]$ y $[\Delta\theta, \Delta V]$, transformando las ecuaciones de la red de tal manera que las impedancias aparecen como si fuesen casi enteramente reactivas. En las nuevas coordenadas la relación X/R es alta, y se logra artificialmente el desacoplo.

3. ASPECTOS COMPLEMENTARIOS DEL REPARTO DE CARGAS.

3.1.INTRODUCCION.

Además del algoritmo básico del reparto de cargas, cuya evolución desde sus comienzos hemos descrito en el capítulo anterior, existen unos aspectos relacionados con el mismo que han centrado la atención de los investigadores en este área.

Cada uno de ellos es de por sí lo suficientemente amplio como para no poder abordarlo en detalle. Aquí, sin embargo, haremos una breve reseña de los mismos, con el afán de dar una visión global del problema que permita al lector hacerse una idea exacta al respecto.

A pesar de todo, habremos de omitir algunos tópicos importantes, que sólo nos tocan de pasada y disponen de sus propias herramientas, como son los equivalentes estáticos.

Los equivalentes estáticos del sistema de potencia se utilizan cuando sólo se está interesado en una zona en concreto del mismo, trabajándose con un equivalente del resto que lo representa bastante bien si las condiciones de trabajo no varían ostensiblemente.

3.2.VALORES INICIALES.

Cualquier procedimiento iterativo de solución de un sistema de ecuaciones no lineales, necesita conocer una estimación inicial adecuada para obtener la convergencia.

En los textos de análisis numérico se han propuesto numerosos procedimientos para la obtención sistematica de valores iniciales [76]. En el caso del reparto de cargas, afortunadamente, este problema está resuelto para la mayoria de las ocasiones de una forma muy simple.

Normalmente se toman como valores iniciales $V + j0$ para las barras de tensión regulada, y $1 + j0$ para el resto, en ausencia de otros mejores. Estos valores iniciales se conocen en la literatura como "flat start", aunque existe cierto confusionismo al respecto, pués algunos autores dan la misma denominación a la asignación del valor $V_r + j0$ para los nudos no regulados, siendo V_r la tensión del nudo de referencia. La diferencia suele ser mínima, pués $V_r \simeq 1$ por regla general. Además de ser los valores iniciales más comunes, se utilizan para comparar los diferentes algoritmos.

A menudo, cuando se realizan sucesivos repartos de cargas (p.e. análisis de contingencias), se toman como mejores valores conocidos los de la solución anterior. Esto suele ahorrar una iteración, o más aún si se trata de métodos con pobre convergencia.

Los métodos de Newton-Raphson en sus diferentes versiones, dada su fuerte convergencia (cuadrática), no son muy sensibles a los valores iniciales. En algunos casos, poco frecuentes, no se obtiene convergencia con el "flat start", divergiendo con la misma velocidad con la que normalmente convergería. Para esos casos raros, unos valores iniciales más exactos podrían lograr la solución. Las experiencias de ciertos autores [108], indican que realizar una o dos iteraciones previas por un procedimiento de desplazamientos sucesivos (del estilo de Gauss-Seidel), es beneficioso.

Puesto que las estimaciones iniciales de los módulos de las tensiones son mejores que las de los ángulos, se ha utilizado también como estimación para éstos últimos los resultados de un reparto de cargas de potencias activas ("d.c. load flow"). El reparto de cargas activas [121], es una simplificación en la que se ignora la componente reactiva, que en muchas situaciones es secundaria.

Las suposiciones que se hacen son:

a) Angulos pequeños : $\mathrm{Sen}\,\vartheta \simeq 0$; $\mathrm{Cos}\,\vartheta \simeq 1$, $\vartheta_i - \vartheta_k \simeq 0$

b) Módulos igual a 1 p.u.

c) Los transformadores tienen sus tomas nominales.

d) Se desprecian los elementos en derivación de la red.

e) Se desprecian las resistencias.

Y las ecuaciones resultantes:

$$P_i = \sum_{k=1}^{N-1} H_{ik} \cdot \vartheta_k \qquad i=1....N-1 \qquad (3.1)$$

siendo N el nudo de referencia cuyo ángulo se toma como cero. En forma matricial:

$$P = H \cdot \vartheta \qquad (3.2)$$

H es cuadrada y no singular, con la misma estructura que Y_{nudos} eliminando la última fila y columna.

De (3.2) se obtienen los ángulos que se tomarán como valores iniciales.

Stott va aún más lejos en la única publicación conocida que versa exclusivamente sobre el problema de los valores iniciales [101]. Para la obtención de los primeros desfases, propone el flujo de cargas activas mencianado anteriormente.

Para calcular los módulos desconocidos, se suponen de la forma:

$$V_i = 1 + \Delta V_i \qquad (3.3)$$

obteniéndose V_i para cada nudo P-Q de la ecuación matricial:

$$D = L \cdot \Delta V \qquad (3.4)$$

en donde D y L se calculan, con los valores ya hallados para ϑ, de ciertas expresiones [101] que dependen de los datos del problema: P_i, Q_i, V_k en los nudos P-V, etc.

Para ángulos muy grandes el error puede llegar a 5^{o}. El máximo error encontrado, en un caso extremo, fue de 0,013 p.u. para los módulos, siendo normalmente menor de 0,003 p.u.

A juzgar por los resultados anteriores, el procedimiento de Stott puede utilizarse para cálculo rápido aproximado de flujos de carga, en el que sólo se pretenda detectar posibles casos difíciles más que resultados exactos.

Es probable que siguiendo la línea de trabajo anterior, Stott obtuviese sus procedimientos desacoplados [102,103], que dan valores muy similares para la primera iteración (en menor tiempo).

A veces, una mínima diferencia en la elección de los valores iniciales, da lugar a algoritmos bien distintos en su complejidad. Tal es el caso de los dos principales métodos de segundo orden en coordenadas cartesianas. Roy [86] utiliza 1 + j0 (véase capítulo 2) y consigue un jacobiano simétrico trás ciertas manipulaciones. Sin embargo Nagendra et al. [73] toman V_r + j0, con lo que desaparece el primer término del desarrollo en serie, resultando un algoritmo más simple y competitivo.

3.3. AJUSTES A LA SOLUCION.

Un programa de uso industrial para el cálculo del flujo de cargas, debe incorporar unas funciones adicionales a la mera solución del problema básico descrito en el capítulo 1. Entre las

más importantes destacan:

- Límites de reactiva en nudos P-V.
- Límites de tensión en nudos P-Q.
- Transformadores con tomas variables.
- Transformadores de desplazamiento de fase.
- Control de intercambio entre áreas.

Realizar alguno de estos ajustes a la solución implica aumentar el número de iteraciones, y por tanto el tiempo de cálculo, así como la complejidad de programación. Ello obscurece la elegancia del algoritmo básico, y por eso a menudo no se discuten en la literatura, lo cual puede hacer llegar a conclusiones erróneas al comparar diversos algoritmos, pués no todos permiten con la misma facilidad la incorporación de los mismos.

En general, lo que se pretende es controlar una magnitud basándose en uno o varios parámetros, con un determinado criterio de control. Por ejemplo, se puede utilizar la toma variable en carga de un transformador, para controlar la tensión en un nudo específico próximo. Entre cada iteración se va ajustando la toma para conseguir que la tensión se mantenga al valor deseado.

Consideremos un criterio de control simple, en el que un parámetro x varía para mantener una variable y a un valor específico y^e. El proceso de ajuste es una realimentación en la que x se corrige, dentro de ciertos límites, para reducir el error $\triangle y = y^e - y$ a una tolerancia dada. Dicho proceso no debe comenzar hasta que y se conozca con cierta precisión.

El tamaño de la corrección Δx no debe ser muy grande ni muy pequeño. Correcciones muy pequeñas harán que la convergencia final venga dictada por la de este bucle realimentado. Correcciones grandes perturbarán excesivamente la solución en cada iteración, hasta el punto que pueden producir divergencia.

El parámetro x puede ser una magnitud continua o discreta. En el segundo caso, para evitar problemas, se suele ignorar su característica discreta hasta cerca del final, en donde se le da el valor del escalón más próximo.

La relación entre Δx y Δy viene dada por

$$\Delta x = \alpha \cdot \Delta y \qquad (3.5)$$

donde α es la ganancia de la realimentación, cuya elección es crítica para minimizar el número total de iteraciones.

Estos ajustes afectan en menor medida a los algoritmos con pobre convergencia, como los utilizados primitivamente. En estos casos se suele buscar un valor empírico de que haga que la convergencia del proceso de ajuste sea del mismo orden que la del algoritmo en su conjunto.

Para algoritmos del tipo de Newton-Raphson la convergencia se ve afectada bastante, siendo corriente que las iteraciones aumenten a más del doble. En estos casos α es aproximadamente la sensitividad entre x e y , y puede obtenerse un valor empírico como se indica en [104].

Tener en cuenta los límites de reactiva en los nudos P–V es el ajuste más comúnmente incorporado a los programas de flujo de cargas. Cuando en un nudo P–V se alcanza uno de los límites, se pierde la capacidad de mantener la tensión al valor específico V^e. Se utilizan dos procedimientos para forzar la potencia reactiva a caer dentro de los límites: a) El nudo se convierte al tipo PQ, con Q^e igual al límite alcanzado, o b) Se realiza una realimentación como la descrita anteriormente, en la que V^e se ajusta en cada iteración con la corrección:

$$\Delta V^e = \alpha \, . \, \Delta Q \qquad (3.6)$$

El método a) se introduce fácilmente en procesos iterativos basados en las matrices de admitancias o impedancias de nudos. Con el método de Newton, el cambio de tipo de nudo obliga a modificar la estructura del jacobiano, lo cual es crítico en los casos en que se utilice un jacobiano constante, triangularizado una sola vez. Se prefiere entonces el procedimiento b).

En [103] se describe cómo incorporar este procedimiento al flujo de cargas desacoplado rápido. La ecuación (3.6) toma la forma,

$$\Delta V = S \, . \, \Delta Q/V \qquad (3.7)$$

en donde la sensitividad S es el elemento diagonal del nudo en cuestión en la inversa de la matriz B", aumentada con la fila y columna correspondientes a ese nudo (recuérdese que B" es una aproximación del jacobiano). Como se dispone de la tabla de fac-

tores de B", es muy rápida la obtención de S utilizando las propias rutinas del algoritmo (Véase apéndice 3). Este valor de S se puede calcular "off-line" para todos los nudos P-V, y utilizarse sucesivamente mientras no existan cambios importantes en la red, puesto que realmente es un valor aproximado.

En [68] se propone un procedimiento para tratar los límites de reactiva en el método de Newton, que sólo tiene un interés didáctico, puesto que el esfuerzo de cálculo y la capacidad de almacenamiento necesaria son muy altos.

Nagendra et al [73] proponen un procedimiento muy similar al descrito en el Apéndice 3, pero para coordenadas cartesianas, en el que, en lugar de calcular un sólo factor de sensitividad cada vez, se obtienen simultáneamente todos los de los nudos que en esa iteración han sobrepasado sus límites.

Sea cual sea el procedimiento utilizado, debe ser capaz de prever la posibilidad de que un nudo que ha excedido los límites vuelva a estar dentro de ellos. Además, ninguno debe comenzar a actuar hasta que exista cierta convergencia (después de dos iteraciones en el método de Newton y similares).

A veces no se permite que un nudo PQ sobrepase ciertos límites en su tensión, para lo que se dispone de capacidad de generación de reactiva, convirtiéndose el nudo en uno del tipo PV. El problema es el dual del anterior, y sirven las consideraciones hechas con las salvedades lógicas.

Los transformadores con tomas bajo carga controlan normalmente la tensión de un nudo próximo. Inicialmente se consideran de variación continua, y el incremento en la toma viene dado por:

$$\Delta t = \alpha \cdot \Delta V \qquad (3.8)$$

donde $\Delta V = V^e - V$. Suele ser satisfactorio considerar $\alpha = \pm 1$ (dependiendo el signo de en qué lado sea la toma), especialmente si el nudo es una terminación radial.

A veces el transformador con tomas puede regular el flujo de reactiva a través de sí mismo o de líneas vecinas. En ese caso la realimentación se hará con ΔQ.

Peterson y Meyer [79] propusieron una forma de incorporar este ajuste al método de Newton que fue muy bien aceptada, evitando introducir el bucle realimentado descrito anteriormente, lo que aumenta excesivamente el número de iteraciones. La idea consiste en incluir las tomas como variables, en lugar de ser constantes. Así, en el proceso iterativo de la forma

$$\Delta F = J \cdot \Delta X \qquad (3.9)$$

suponiendo que el transformador regula la tensión en el nudo m como indica la figura,

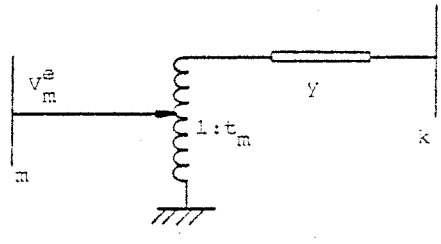


fig. 3.1

se introducirán los siguientes cambios:

a) Sustituir $\Delta V_m/V_m$ por $\Delta t_m/t_m$ en ΔX(ya que $V_m = V_m^e$ = constante).

b) Los términos del jacobiano de la forma:

$$\frac{\partial P_k}{\partial V_m}.V_m \quad , \quad \frac{\partial Q_k}{\partial V_m}.V_m \quad , \quad \frac{\partial P_m}{\partial V_m}.V_m \quad , \quad \frac{\partial Q_m}{\partial V_m}.V_m$$

se sustituyen por estos otros:

$$\frac{\partial P_k}{\partial t_m}.t_m \quad , \quad \frac{\partial Q_k}{\partial t_m}.t_m \quad , \quad \frac{\partial P_m}{\partial t_m}.t_m \quad , \quad \frac{\partial Q_m}{\partial t_m}.t_m$$

(aunque su estructura no cambia).

Con ello se logra casi la misma convergencia que en una solución sin ajustar. Sin embargo el procedimiento tiene algunos inconvenientes, entre los que destaca el manejo de los límites en las tomas, que pueden ser sobrepasados ampliamente trás una iteración.

Para métodos que utilizan un jacobiano constante [104], es más interesante ajustar las tomas como indica (3.8), formándose dicha matriz inicialmente con el valor nominal de las tomas.

Recientemente [2] se ha sugerido un procedimiento para ajustar las tomas en el flujo de cargas desacoplado rápido. Consiste en considerar la barra cuya tensión regula la toma, como un nudo PV, aunque estrictamente sea un nudo PQ. Se evalúa, como se hace habitualmente, la potencia reactiva Q necesaria para mantener esa tensión constante. Ese valor de Q es equivalente a un cambio Δt en la relación de transformación. El cambio Δt, a su vez, modi-

fica las tensiones de otros nudos próximos, lo que implica que Q sea ligeramente diferente al primer valor calculado. Se entra entonces en un proceso iterativo hasta obtener un valor Δt exacto, aunque puede ser suficiente con un valor de Δt aproximado (no iterativo).

Un transformador de desplazamiento de fase regula el flujo de potencia activa a su través, o a través de líneas vecinas. Estos dispositivos introducen asimetría en el jacobiano.

Peterson y Meyer [79] realizaron este ajuste por un procedimiento análogo al indicado para el transformador con tomas.

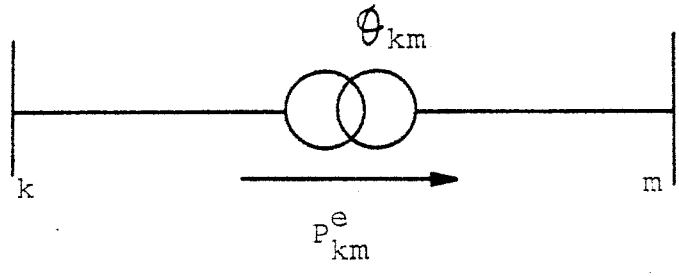


fig. 3.2

Como ahora Pkm no es una variable regular del problema, no se puede hacer el cambio de variables anterior. En lugar de ello se añade la nueva ecuación $P_{km} = P^e_{km} - P_{km}$, siendo ϑ_{km} una nueva variable, y modificándose convenientemente el jacobiano. Si P_{km} se refiere a flujos por líneas diferentes de las que está el transformador, el nuevo elemento introducido en la diagonal puede ser muy pequeño o cero, lo que crearía nuevos problemas.

De nuevo, para algoritmos que utilizan un jacobiano constante y simétrico, conviene representar el transformador de despla-

zamiento de fase como inyecciones de potencia en los nudos, en lugar de incluirlo en los términos del jacobiano, y realizar el ajuste por el procedimiento de realimentación que indica (3.8).

El último ajuste que vamos a considerar es el del intercambio de potencia activa entre áreas, que se debe mantener a un valor determinado. Se suele asignar la responsabilidad de mantener el intercambio a un generador específico. En ese caso, trás cada iteración, se suman las potencias de todas las líneas de intercambio y se obtiene el error ΔP_I. La potencia que da el generador responsable se debe modificar en la cantidad

$$\Delta P_G = \alpha \cdot \Delta P_I \qquad (3.10)$$

Es satisfactorio utilizar $\alpha = 1$ en el método de Newton [104]. Para este algoritmo, Britton [16] propuso una técnica similar a la ya mencionada de Peterson y Meyer para transformadores con tomas.

Si son varios generadores los que deben mantener el intercambio, se distribuirá proporcionalmente a su capacidad la magnitud del error.

3.4. ANALISIS DE CONTINGENCIAS.

Existen aplicaciones específicas en las que se requieren resolver repetidamente un reparto de cargas, con alguna modificación en la red o en las restricciones de los nudos.

Pasaremos por alto dos de las aplicaciones más importantes, como son el flujo de cargas óptimo y el estudio de la estabilidad dinámica por integración numérica. Nos detendremos sin embargo en una tercera aplicación, denominada análisis de contingencias.

El análisis de contingencias es un estudio de lo que ocurre en el sistema, partiendo de un caso base, ante determinados cambios en la red, en las generaciones o en las cargas.

Se utiliza en planificación ("off-line") o en el control actual de la operación del sistema ("on-line"). En ambos casos se puede obtener información sobre cuál es el grado de seguridad de una determinada configuración del sistema, detectándose posibles dificultades.

Por su propia naturaleza no se requiere excesiva precisión en los resultados, haciéndose más hincapié en la velocidad de cálculo, debido al gran número de casos que habrá que resolver. Como valores iniciales se pueden utilizar los del caso base. La realización de ajustes dependerá de las necesidades concretas, no siendo precisos por regla general.

Cualquier método moderno rápido, en especial los desacoplados, puede servir para estos estudios, realizando sólo las iteraciones precisas. Han aparecido algunas publicaciones expresamente dedicadas al tema, de las cuales mencionaremos las más destacadas.

Peterson, Tinney y Bree [80] propusieron un algoritmo que era una recomposición, adaptada a este problema, de otros anteriores [25,110,111]. Mediante ciertas simplificaciones en las ecuaciones que definen el reparto de cargas, se llega a las siguientes ecuaciones desacopladas:

$$A \cdot \theta = P' + P'' \qquad (3.11)$$

$$C.V = Q' + Q'' \qquad (3.12)$$

Aunque las matrices A y C son variables con las tensiones, se calculan sólo para el caso base y se triangularizan una sola vez.

Un cambio en una línea entre los nudos k,m modificará A en los elementos kk,mm,km,mk sólamente (y en la misma cantidad a_{km}).

Dicho cambio se puede expresar como:

$$\Delta A = a_{km} \cdot M \cdot M^t \qquad (3.13)$$

donde M es un vector nulo salvo los elementos k y m que valen 1 y -1 respectivamente.

Los nuevos ángulos se pueden calcular de :

$$(A + \Delta A)(\vartheta + \Delta\vartheta) = P' + P'' \qquad (3.14)$$

Teniendo en cuenta (3.11) los incrementos en los ángulos se obtienen de:

$$\begin{aligned} (A + \Delta A)\,\Delta\vartheta = \Delta A \,.\, \vartheta &= a_{km}(\vartheta_k - \vartheta_m)M \\ \Delta\vartheta &= (A + \Delta A)^{-1} \,.\, a_{km}(\vartheta_k - \vartheta_m)M \end{aligned} \qquad (3.15)$$

La inversa de $A + \Delta A$ en la ecuación anterior se puede obtener por la fórmula de Sherman - Morrison [97]

$$(A + a_{km}\, MM^t)^{-1} = A^{-1} - (M^t A^{-1} M + \frac{1}{a_{km}})^{-1} . A^{-1} . M . M^t . A^{-1} \qquad (3.16)$$

que a pesar de su aparente complejidad, es eficiente si se utiliza la tabla de factores de A, en lugar de A^{-1} en forma explícita, puesto que el resto de las operaciones involucran sólo escalares o al vector M. Esto es un caso particular de una técnica general para invertir matrices modificadas, muy utilizada en análisis numerico.

Análogamente se procedería con la ecuación (3.12) para calcular los nuevos módulos de tensiones.

A partir de las ecuaciones (3.11) y (3.15) para $P-\vartheta$, y de las correspondientes para Q-V, se proponen diferentes esquemas iterativos, según que la contingencia ocurra en una línea o en un

generador. Generalmente se obtienen buenos resultados con sólo dos iteraciones en P-θ

y una en Q-V.

Stott y Alsac [103] en la presentación del método desacoplado rápido, proponen un procedimiento análogo adaptado a su propio algoritmo. Se trata ante todo de no tener que recalcular la tabla de factores de B' y B", que consume tiempo, para lo cual también utilizan (3.16).

Sachdev e Ibrahim [87] cambiaron de técnica. La puesta fuera de servicio de una línea es simulada por adecuadas inyecciones de potencia en los nudos, utilizando la matriz de sensitividad. A continuación se expondrán las ideas básicas de su algoritmo.

Las ecuaciones del flujo de cargas, linealizadas alrededor del estado básico, relacionan las pequeñas variaciones del vector de estado con el vector de control

$$J \, \Delta X = \Delta U \qquad (3.17)$$

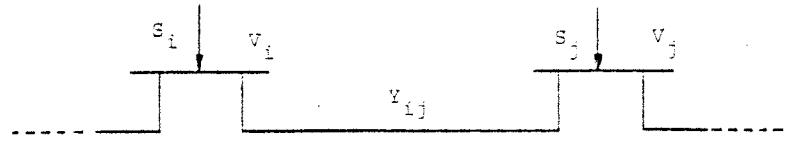
A veces se escribe la relación inversa

$$S \, . \, \Delta U = \Delta X \qquad (3.18)$$

donde S es la matriz de sensitividad, cuyos términos no es preciso calcular explícitamente.

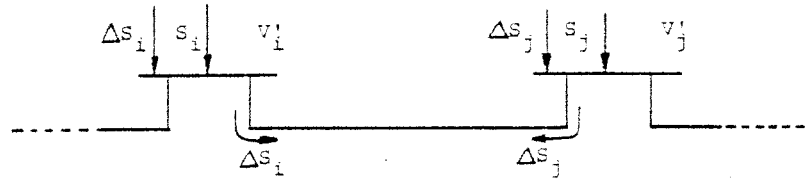
El cambio en la generación o carga de un nudo induce un nuevo estado que se obtiene fácilmente de (3.18) (realmente se opera con (3.17) pues se dispone del jacobiano triangularizado). Todos los elementos de U son nulos salvo el del nudo en cuestión, suponiendo que el nudo oscilante absorba todo el cambio. Habrá otros elementos no nulos si varios generadores comparten los efectos de la modificación. Esta contingencia es siempre la más fácil de estudiar.

Supongamos ahora que queremos estudiar las modificaciones que introduce la eliminación de una línea o transformador. Las tensiones en el estado final, de los nudos en los que incide la línea, no cambiarán si la línea se reconecta, y se inyecta en los nudos las potencias que fluirían por la línea. Tendríamos un sistema con la misma configuración pero comportándose como si la línea no existiera. Esta idea queda representada en las siguientes figuras:



(a) Configuración Inicial

(b) Configuración final



(c) Configuración final equivalente

fig. 3.3

El jacobiano permanece inalterado, salvo los cambios despreciables debido a los nuevos valores V_i', V_j'. Es inmediato obtener una relación no lineal entre $\Delta S_i, \Delta S_j$ y V_i', V_j' [87]. Existe además otra relación entre las partes real e imaginaria de ΔS_i, ΔS_j y los cambios habidos en las tensiones $\Delta V_i = V_i' - V_i$ y $\Delta V_j = V_j' - V_j$ que se obtiene simplemente entresacando las oportunas ecuaciones de (3.18). Entre los dos conjuntos de ecuaciones se obtiene iterativamente ΔS_i y ΔS_j, con la precisión deseada. Finalmente se obtienen todos los cambios en las tensiones con (3.18) (o (3.17)).

Los mismos autores presentaron esta técnica en una versión para coordenadas cartesianas [88].

Posteriormente, Mamandur y Berg [69] utilizaron la misma idea de simular la apertura de una línea por modificaciones en las potencias. En lugar de calcular de forma iterativa ΔV_i, ΔV_j, en base a su relación no lineal con $\Delta S_i, \Delta S_j$ (variaciones en las inyecciones de los nudos), se calculan en forma más sistemática y directa. Para ello se aprovecha su relación (linealizada), con los cambios habidos en las potencias que fluyen por el resto de líneas, que aún permanecen conectadas al par de nudos. La ecuación principal, diferente a la del método anterior es :

$$\begin{bmatrix} \Delta P_{ia} \\ \Delta Q_{ia} \\ \Delta P_{ib} \\ \Delta Q_{ib} \end{bmatrix} = T \begin{bmatrix} \Delta \vartheta \\ \Delta V \end{bmatrix} \qquad (3.19)$$

En dicha ecuación, ΔS_{ia} es el cambio resultante en la potencia total de las líneas que quedan en el nudo i, como consecuencia de la apertura de la línea objeto de estudio. ΔS_{ib} es el equivalente para el nudo j. Esos valores se obtienen fácilmente, sabiendo que en cada nudo las líneas restantes deben satisfacer la potencia neta inyectada, que no ha cambiado. Los términos de T son de la forma

$$\frac{\partial P_{ij}}{\partial \vartheta}, \qquad \frac{\partial Q_{ij}}{\partial V}$$

y por tanto son análogos a los del jacobiano. Para cambios de tensión muy grandes, este método es probablemente menos exacto que el anterior, pués en el fondo es una versión linealizada.

Pai et al. [77] propusieron una forma nueva de resolver el flujo de cargas, haciendo especial hincapié en su utilización para el análisis de contingencias. Transforman los nudos de carga en admitancias equivalentes, quedando una red reducida con sólo nudos P-V. Los ángulos ϑ_g de estos nudos se calculan de forma semejante al algoritmo desacoplado rápido, esto es:

$$\Delta P/V = B \cdot \Delta \vartheta_g \qquad (3.20)$$

Con los nuevos ángulos se calculan las tensiones en los nudos PQ de la expresión

$$Y_{11} \cdot V_1 = -Y_{1g} \cdot V_g \qquad (3.21)$$

donde Y_{11}, Y_{1g} son submatrices, modificadas convenientemente, de la matriz de admitancias de nudos. El proceso se repite hasta ob-

tener la convergencia deseada.

El análisis de contingencias lo realizan análogamente a como se hace en las referencias [80,103]. Por los resultados mostrados parece ser tan competitivo como el de [103].

Por último, Hulskamp et al. [55] utilizan una arquitectura no convencional para el análisis de contingencias. Esta arquitectura ha sido ensayada ya con éxito en numerosas aplicaciones de los sistemas eléctricos de potencia, como puede ser un reparto de cargas convencional (descrito en el capítulo 2 [82]), y parece ser una alternativa relativamente barata para la solución en tiempo real de sistemas muy grandes.

Consiste, como se recordará, en un procesador de alta velocidad de cálculo aritmetico, llamado "array processor", conectado como periférico de un miniordenador.

En el capítulo 2 se describía brevemente la composición de ese procesador, así como sus ventajas e inconvenientes. El procesamiento de los datos se debe hacer en forma de vectores de la mayor dimensión posible, para aprovechar el paralelismo interno del procesador, resultando operaciones muy eficientes.

El análisis de contingencias propuesto por estos autores está basado en el algoritmo desacoplado rápido de Stott y Alsac [103]. Se realiza enteramente en el "array processor", para evitar transmitir excesivos datos al ordenador principal, reservándose este último para la solución del reparto de cargas en el es-

tado básico, y la responsabilidad de la interfase con los demás perifericos, que el otro no puede asumir.

Exponen con detalle, y de forma clara, cómo disponer los datos en forma vectorial, especialmente la rutina de cálculo de los residuos de potencia ΔP, ΔQ, que según sus resultados consume una parte importante del tiempo (58 %).

En definitiva, todos los algoritmos propuestos, buscan una solución de compromiso entre un reparto de cargas exacto, costoso en tiempo si se pretenden analizar muchos casos, y el llamado flujo de cargas activas, que siendo muy rápido (no iterativo), no da información sobre el problema reactivo, y la que da no suele ser lo suficientemente precisa.

3.5. CONVERGENCIA, ESTABILIDAD Y UNICIDAD DE LAS SOLUCIONES.

Las tres propiedades que dan título al epígrafe, conforman una serie de aspectos teóricos ligados a todo problema numérico, que en el caso que nos ocupa no han sido profusamente estudiados.

El máximo interés, como lo demuestra la literatura sobre el tema, se ha centrado en el desarrollo de métodos numéricos cada vez más eficientes. En comparación, se ha investigado poco la parte analítica del problema del reparto de cargas.

La explicación puede ser debida, al hecho real de que en la práctica pocas veces aparecen dificultades que necesiten ser ana-

lizadas teóricamente. No se ha dedicado por tanto gran esfuerzo a ello.

El único aspecto que se salva de estas consideraciones es el de la convergencia, precisamente porque es el factor más importante en la elección de un método que ofrezca fiabilidad, y porque además se han encontrado casos para los cuales ningún método conocido converge, lo que ha motivado a los investigadores en este sentido.

3.5.1 CONVERGENCIA.

La convergencia se diferencia de las otras dos cuestiones, en que va unida al método de solución, más que a la solución en sí misma. Por tanto se debe analizar en el contexto de un algoritmo concreto, independientemente de ciertas consideraciones generales comunes a todos.

En el capítulo 1 ya se indicaron los criterios utilizados para determinar cuándo se ha producido la convergencia. Para comparar dos algoritmos en este tema, es preciso basarse en el mismo criterio para obtener conclusiones válidas.

Una medida de la velocidad de convergencia se puede obtener de la siguiente ecuación,

$$\left| X^{(h+1)} - X \right| = K \left| X^{(h)} - X \right|^{n} \qquad (3.22)$$

donde X es el valor verdadero, y $X^{(h)}$ indica el valor en la iteración h. Para que exista convergencia k debe ser menor que 1. Si

n=1 la convergencia es lineal, y si n=2 la convergencia es cuadrática.

Considérese, en general, el problema de resolver el sistema no lineal F(x)=y. Todos los procedimientos iterativos calculan la solución x a través de la fórmula recursiva:

$$X^{k+1} = \varphi(X^k) \tag{3.23}$$

generándose una secuencia de puntos x^1, x^2,... que convergen a x. Por tanto, para ese valor x, se cumplirá que

$$X = \varphi(X) \tag{3.24}$$

El que la solución de F(x)-y=0 deba satisfacer (3.24) conduce a la relación:

$$F(x) - y = H(x)(x - \varphi(x)) \tag{3.25}$$

donde H(x) es una matriz no singular cuyos elementos son función de x. Despejando $\varphi(x)$ de (3.25) y sustituyendo en (3.23) obtenemos la siguiente fórmula recursiva general:

$$X^{k+1} = X^k - H(x)^{-1} (F(x^k)-y) \tag{3.26}$$

Los diferentes algoritmos propuestos se diferencian entre sí en la elección de H(x), llamada matriz de la iteración.

Los métodos primitivos, basados en la matriz de admitancias de nudos (Gauss-Seidel), tienen una convergencia muy pobre, y muy dependiente de diversos factores. La técnica más común para ace-

lerar la convergencia en estos casos, es la de corregir en exceso los valores estimados [76].

$$x^{(h+1)} = x^{(h)} + \omega . \Delta x \qquad (3.27)$$

denominándose a esta versión del Gauss-Seidel, el método de sobrerelajaciónes sucesivas (SOR). A pesar de que x es un complejo, ω es un número real, que se obtiene empíricamente y depende del sistema en concreto, aunque teóricamente se demuestra que debe estar comprendido entre 0 y 2 para que se produzca convergencia. Utilizar un valor ω complejo dificulta las cosas, pero se puede demostrar asimismo que el valor ω que minimiza el número de iteraciones es complejo. El orden en que se van procesando las ecuaciones también afecta a la convergencia.

Por el contrario, aquellos métodos de desplazamientos sucesivos basados en la matriz de impedancias, no son sensibles a factores de aceleración, presentando mucha mejor convergencia. La no generalización de su uso se debe a otros motivos, principalmente la ocupación de memoria.

En ambos casos, la elección del nudo de referencia afecta en cierto grado a la convergencia [35].

En la presentación del método de Newton por Tinney y Hart [108], quedaron patentes sus ventajas frente a los procedimientos anteriores, en lo que atañe a la convergencia. Estas se pueden plasmar en los siguientes puntos:

a) Número de iteraciones independiente del tamaño del sistema (convergencia fuertemente cuadrática).

b) Más tolerancia a la elección del nudo de referencia.

c) No requiere factores de aceleración.

d) Es capaz de resolver problemas difíciles, en los que fallan otros métodos, como son:

- Sistemas radiales.
- Lineas cortas y largas terminando en un mismo nudo.
- Sistemas fuertemente cargados, lo que implica desfases muy grandes en las líneas.
- Sistemas con reactancias negativas, como condensadores en serie.

Este método se obtiene de la ecuación (3.26) haciendo H(x) igual al jacobiano de F(x) para $X=X^k$.

Diversas aproximaciones del método de Newton utilizan un jacobiano constante, lo que por supuesto empeora la convergencia, haciéndolas más dependientes de los valores iniciales tomados.

El método desacoplado rápido [103] introduce además otras simplificaciones. A pesar de ello sigue teniendo convergencia geométrica, y es bastante fiable tomando como valores iniciales $1\angle 0^{\circ}$. No obstante, para soluciones muy precisas requiere iteraciones adicionales. En este algoritmo existen tres componentes con sus propias características de convergencia: a) La solución del problema P-ϑ, b) La solución del problema Q-V y c) Los efectos de los cambios de V y ϑ, en la definición de las funciones $\Delta P/V$ y $\Delta Q/V$ respectivamente.

La convergencia de este algoritmo para el sistema de 118 nudos del IEEE se muestra en la siguiente figura:

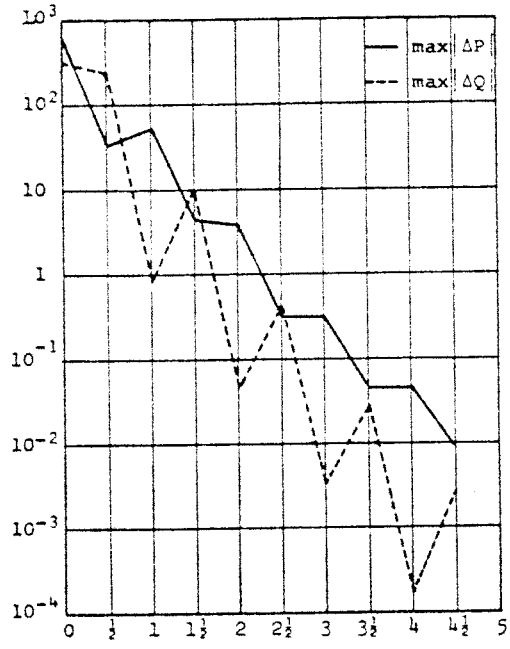


fig. 3.4

Cada vez que se actualizan los nuevos ϑ, disminuimos lógicamente ΔP, pero aumentamos ΔQ. El efecto análogo entre V y ΔP es menos pronunciado. Ello es debido a que las pérdidas activas debidas al flujo de reactiva son normalmente menores que las pérdidas reactivas debidas al flujo de activa. El esquema iterativo que produce mejor convergencia es (1ϑ ,1V) [103].

Dada la gran expectación que ocasionó la publicación de este algoritmo, se hicieron numerosos ensayos para comprobar realmente su fiabilidad, dado que era muy rápido.

Se detectó que la convergencia empeoraba apreciablemente cuando existían líneas con una relación R/X alta. Esta situación se puede presentar en líneas de tensión muy alta con reactancia

capacitiva en serie para compensación, en líneas de distribución, donde la resistencia es más importante, o en cables. Ello es debido a que las aproximaciones hechas por el algoritmo son menos exactas.

Dy Liaco y Ramarao [32] ensayaron con éxito una solución para estos casos. Consiste en introducir un nudo ficticio en medio de la línea problemática, dividiéndola así en dos. En una mitad se coloca una reactancia negativa X", y en la otra mitad se coloca la impedancia total de la línea original, más la reactancia X" cambiada de signo. Por tanto una de las líneas resultantes tiene una relación R/X nula, y la otra muy pequeña. Cuando la línea es radial este método da buen resultado, aunque parece que no ocurre lo mismo si forma parte de un bucle.

Wu [124] presentó años más tarde un estudio teórico de la convergencia del método desacoplado rápido. Primeramente demostró que las ecuaciones básicas de este algoritmo son un caso particular de la ecuación (3.26), con una determinada matriz H(x). Este resultado le permitió aplicar toda la teoría desarrollada en análisis numérico, a este caso concreto. Sus principales conclusiones fueron:

a) Unas condiciones suficientes para la convergencia del proceso iterativo. La comprobación de estas condiciones se puede hacer utilizando los datos de la red y los resultados de la primera iteración.
b) La existencia y unicidad de la solución al reparto de cargas en una determinada región.

c) Una estimación del error trás cada iteración, entre el valor calculado y el verdadero.

d) Un aumento de la relación máxima R/X en el sistema dificulta la convergencia, si permanecen constantes, los otros datos.

Estos resultados no pasan de tener un interés teórico, dada su difícil aplicabilidad real (en lo que respecta a las condiciones suficientes para la convergencia). Nagendra et al [72], con el afán de comparar su eficiente método de segundo orden con el desacoplado rápido, utilizaron la misma técnica que Wu para demostrar que el método de Gauss-Seidel (en sus dos versiones de admitancias e impedancias), el método de Newton, el método desacoplado rápido, y su propio método de segundo orden, son casos particulares de la ecuación general (3.26). Para cada procedimiento calcularon la matriz H(x).

Ello puso de manifiesto que todos los métodos son a fin de cuentas aproximaciones del método de Newton-Raphson. Aunque no profundizaron excesivamente, concluyeron que su nueva técnica de segundo orden era más fiable que la de Stott y Alsac para sistemas de difícil convergencia, porque introducía menos aproximaciones en el jacobiano.

Braess y Grebe [13] dieron también una condición suficiente para la convergencia del método de Newton, y de aquellos basados en la matriz de impedancias, en una determinada región. Esta condición es excesivamente conservadora, y no muy útil desde un punto de vista práctico, especialmente en sistemas muy grandes, pues exige conocer los elementos de la matriz de impedancias. Resulta

interesante la forma en que propusieron resolver sistemas muy cargados, para los que no se obtiene convergencia. Consiste en considerar primeramente sólo una fracción de la carga y los parámetros de la red. La solución de este problema se toma como valor inicial para otro con una fracción mayor, y así sucesivamente hasta considerar la carga y parámetros completos.

Recientemente, Nagendra et al. [74] han propuesto la utilización de un criterio empírico para la determinación de la convergencia del algoritmo desacoplado rápido, a diferencia del propuesto por Wu que es netamente teórico. Asimismo, calculan en forma aproximada el número de iteraciones que serán necesarias para obtener la solución hasta la tolerancia deseada.

Se calcula el radio espectral s, de la matriz de iteración particular de ese algoritmo H(x). El radio espectral es, por definición, el módulo del mayor autovalor de la matriz. Si $s<1$, para ese sistema, el algoritmo desacoplado rápido convergerá. Además, la velocidad de convergencia viene dada aproximadamente por $-\log(s)$, de forma que el número de iteraciones se puede estimar de la ecuación:

$$n = - \log(|u|/|v|) / \log(s) \qquad (3.28)$$

donde u es el máximo valor de ΔP, ΔQ trás la primera iteración y v es la tolerancia deseada para dar por finalizado el proceso.

La constante s no se debe calcular explícitamente, lo que sería ineficaz, sino que se estima empíricamente sin necesidad de obtener la matriz de iteración H(x). De todas formas se contem-

plan muchos casos posibles para el cálculo de s, según el tipo de red, lo que resta elegancia al procedimiento.

Para finalizar, se hará una última consideración acerca de la convergencia [105]. Se debe tener cuidado con no exigir una tolerancia excesivamente precisa en la resolución del problema del reparto de cargas, especialmente si el tamaño de palabra del ordenador no es muy grande. Ello podría aumentar artificialmente el número de iteraciones, debido a los errores de redondeo, o incluso podría no conseguirse la tolerancia pedida. Si no se utilizan números reales de doble precisión, no se debe exigir como criterio de convergencia menos de 10^{-4} ó 10^{-5} p.u.

3.5.2 ESTABILIDAD Y UNICIDAD.

Entraremos ahora brevemente a tratar el tema de la unicidad y estabilidad de las soluciones al reparto de cargas. Todos los trabajos publicados al respecto, en su mayoria muy recientes, obtienen conclusiones para sistemas cuyos nudos son del tipo PV, y en los que se desprecian las pérdidas. Se han hecho mínimas conjéturas para sistemas con pérdidas.

Existe una gran laguna entre las publicaciones de tipo eminentemente pragmático, y aquellas otras más teóricas que ahora estamos considerando, y deberá pasar cierto tiempo hasta que los resultados de estas últimas tengan una aplicabilidad clara. El mayor inconveniente puede provenir del hecho de que se trabaja por separado en ambos campos, con equipos de personas, técnicas,e intereses diferentes, sin aparente coordinación.

Considérese una red sin pérdidas de n+1 barras PV, en la que la barra n+1 es la de referencia ($\vartheta_{n+1}=0$). Las ecuaciones del flujo de cargas son:

$$P_i = \sum_{j=1}^{n+1} \alpha_{ij} \, Sen(\vartheta_i - \vartheta_j) \qquad i = 1,2,...n+1 \tag{3.29}$$

$$\alpha_{ij} = cte. = V_i V_j B_{ij} \gtrless 0$$

donde la ecuacion n+1 no se utiliza al ser redundante: $p_1+p_2+...p_{n+1}=0$. En forma vectorial:

$$P = f(\vartheta) \tag{3.30}$$

La función f es periódica, y por tanto restringimos el dominio de variación de cada componente al intervalo $[-\pi, \pi]$.

Al hablar de la estabilidad, nos referimos a la estabilidad en el sentido de Liapunov. Las soluciones al reparto de cargas, serán los puntos de equilibrio de las ecuaciones diferenciales que gobiernan la dinámica de la red ("swing equations"):

$$M_i \ddot{\vartheta}_i + D_i \dot{\vartheta}_i = P_i - f_i(\vartheta) \qquad i=1...n \tag{3.31}$$

En la ecuación anterior, M_i, D_i son las constantes de inercia y amortiguamiento del generador i-ésimo. El estado $(\vartheta, \dot{\vartheta})$ es un equilibrio si $\dot{\vartheta}=0$ y $f(\vartheta)=p$. Si la ecuación (3.31) se linealiza alrededor de un equilibrio, los autovalores del sistema linealizado marcan la pauta para la determinación de la estabilidad en el sentido de Liapunov. El vector ϑ es estable si y solo si

$F(\vartheta)>0$, es decir, si el jacobiano $F(\vartheta)$ es definido positivo (si todos sus autovalores son positivos).

Si hacemos $D_i=0$ llegamos a la misma conclusión. El sistema es hamiltoniano (conservativo), y se puede definir una función potencial V tal que $\nabla V=f(\vartheta)$ y su hessiano

$$\frac{\partial^2 V}{\partial\vartheta_i\,\partial\vartheta_j}$$

determina la estabilidad. La ventaja es que así podemos aplicar la teoria de Morse para clasificar los puntos críticos.

El análisis de la estabilidad de las soluciones al reparto de cargas, nos mete de lleno por tanto en las técnicas cualitativas desarrolladas por Poincaré y el propio Liapunov, y continuadas por importantes matematicos de este siglo.

En contra de lo que se había venido creyendo, Korsak [62] presentó un ejemplo en el que existían dos soluciones estables, para un mismo vector de demandas. Arapostathis et al. [4], demostraron que la región estable puede ser desconexa, y que en el ejemplo de Korsak ambas soluciones estables pertenecían a componentes conexas diferentes. Además, para pasar en forma continua de una a otra, es inevitable atravesar zonas de inestabilidad. Una de las componentes conexas de la región estable, llamada componente principal, incluye el origen ($\vartheta_i=0$). Esta componente es muy importante desde el punto de vista práctico, pues para que las líneas no sobrepasen su capacidad de transmisión, el vector ϑ debe caer en dicha componente principal, normalmente.

De las dos soluciones del ejemplo de Korsak (un bucle de 5 nudos) sólo una cae en la componente principal y físicamente no se puede alcanzar la otra. Como indicaron Nagendra et al. [72] ningún algoritmo numérico convergerá a la segunda solución con los valores iniciales que usualmente se toman (θ=0).

Otras conclusiones de Arapostathis et al. fueron:

- La ecuación (3.30) tiene a lo sumo una solución en la componente principal.
- Si dicha ecuación tiene alguna solución estable, entonces tiene una solución estable en la componente principal.
- El número de soluciones no estables es mayor que las estables.
- Existen vectores de demanda p que son satisfechos por soluciones inestables pero no por estables.

Aparte del análisis global de la solución, se adentraron en las propiedades locales, estudiando mediante la teoría de bifurcaciones los cambios cualitativos en el plano de fase al variar el vector p.

Si nos alejamos de los límites de la región estable, existirá un mayor margen de estabilidad, como se comprende intuitivamente. Una medida de este margen lo puede dar el menor autovalor de $F(\theta)$. Se produce un punto de bifurcación cuando $F(\theta)=0$.

No es inseparable el estudio local y global de la solución, puesto que el comportamiento global viene estructurado fundamen-

telmente por las bifurcaciones que ocurren.

Otros autores, principalmente Baillieul y Byrnes [7,8], profundizaron en el tema. Para ello, mediante el cambio de variables:

$$\begin{aligned} X_i &= \text{Sen}\,(\vartheta_1 - \vartheta_i) \\ y_i &= \text{Cos}\,(\vartheta_1 - \vartheta_i) \qquad i = 2...n \\ X_{n+1} &= \text{Sen}\,\vartheta_1 \\ y_{n+1} &= \text{Cos}\,\vartheta_1 \end{aligned} \tag{3.32}$$

convierten las ecuaciones trascendentales (3.29) en ecuaciones de polinomios, para cuyo estudio disponen de diversas teorías matemáticas (geometría algebraica, intersección en espacios proyectivos, teoría de eliminación para la obtención de raices de polinomios, etc.).

Uno de sus resultados más espectaculares, es que el número de soluciones complejas para inyecciones de potencias pequeñas, en una red sin pérdidas de n nudos PV, son:

$$\binom{2n-2}{n-1}$$

que tiende a 2^{2n} para n grande. De ellas, al menos 2^{2n-1} son reales.

El caso concreto de 3 barras (2 generadores frente a una barra de potencia infinita) ha sido profusamente estudiado. Comenzaron haciéndolo Távora y Smith [107], que hicieron populares sus figuras para unos valores concretos de los parámetros de las

ecuaciones. Arapostathis y Varaiya [5], mostraron que el comportamiento de tal sistema, puede ser mucho más complejo que el de un solo generador conectado a una barra de potencia infinita. En este último caso, se sabe que con suficiente amortiguamiento se puede lograr completamente la estabilidad, mientras que con 3 máquinas el sistema puede no ser estable, independientemente del amortiguamiento. Esto es importante, porque muchas veces se pretende generalizar el comportamiento cualitativo de una máquina frente a un sistema de potencia infinita.

Baillieul y Byrnes [10,11], mediante el cambio de variables mencionado con anterioridad, lograron determinar la dependencia del número y la naturaleza de las soluciones con los parámetros del problema (p_i, θ_{ij}). Una vez hecho el cambio de variables, es factible, en principio, establecer una solución explícita a las soluciones del flujo de cargas, con la potente teoría de ecuaciones polinómicas. Para el caso de 3 barras lo realizaron utilizando el MACSYMA (lenguaje de simulación), de lo contrario sería materialmente imposible manejar las expresiones que aparecen al ir eliminando las variables entre ecuaciones.

Los mismos autores [9] inician el estudio del sistema con pérdidas. La principal dificultad al hacerlo radica en que la noción de solución estable no viene ya caracterizada por ser $F(\theta)>0$.

Muestran que la solución al flujo de cargas es casi siempre única, o al menos existen muchas menos que para el sistema sin pérdidas. Si las resistencias son suficientemente pequeñas, gran

parte de las conclusiones obtenidas permanecen válidas para el caso con pérdidas.

Resulta extraño que nadie haya analizado el comportamiento de las ecuaciones del flujo de cargas en coordenadas cartesianas (véase capítulo 1). En ese caso se obtienen directamente ecuaciones cuadráticas, sin necesidad de hacer el cambio de variables sugerido por Baillieul y Byrnes, con la ventaja de un significado físico más patente.

3.6.SOLUCION DEL SISTEMA DE ECUACIONES LINEALES.

Junto a las técnicas de matrices vacías, que se comentarán en el capítulo siguiente, la solución eficiente del sistema de ecuaciones lineales que aparece en cada iteración, son los aspectos más importantes a la hora de llevar a la práctica, traducido en forma de programa, un algoritmo de reparto de cargas basado en cualquiera de las variantes del método de Newton.

El mejor algoritmo puede no resultar operativo, si no se hace una programación adecuada de esas facetas, que constituyen casi la totalidad del tiempo invertido en la solución del problema global (sin tener en cuenta las entradas y salidas de datos).

Nos interesa entonces conocer los procedimientos que son rentables para la obtención del vector X de la ecuación:

$$AX = B \tag{3.33}$$

siendo A una matriz cuadrada no singular de nxn y B y X vectores

de dimensión n (si B y X fuesen a su vez matrices, se trataría una columna de cada vez). No son rentables, por ejemplo, la inversión explícita de A o la regla de Cramer.

El más clásico y básico de ellos es la eliminación gaussiana. Consiste en la realización de una serie de modificaciones en los elementos de A y B, hasta lograr que A sea triangular superior.

El proceso se puede describir como sigue [76]. En la etapa k-ésima tenemos un sistema "reducido"

$$A^{(k)}.X = B^{(k)} \tag{3.34}$$

donde $A^{(1)}=A$ y $B^{(1)},=B$, y en general $A^{(k)}$ y $B^{(k)}$ son de la forma:

$$A^{(k)} = \begin{bmatrix} a_{11}^{(k)} & \cdots & & a_{1n}^{(k)} \\ & \ddots & & \\ & & a_{kk}^{(k)} \cdots & a_{kn}^{(k)} \\ & O & & \\ & & a_{nk}^{(k)} & a_{nn}^{(k)} \end{bmatrix} ; \quad B^{(k)} = \begin{bmatrix} b_1^{(k)} \\ \vdots \\ \\ b_n^{(k)} \end{bmatrix} \tag{3.35}$$

Para obtener el siguiente sistema reducido se supone que:

$$a_{kk}^{(k)} \neq 0$$

y los elementos de $A^{(k+1)}$ y $B^{(k+1)}$ se definen como:

$$a_{ij}^{(k+1)} = \begin{cases} a_{ij}^{(k)} & \text{si} \quad i \leqslant k \quad o \quad j<k \\ a_{ij}^{(k)} - m_{ik}a_{kj}^{(k)} & i=k+1\ldots n,\ j=k\ldots n \end{cases} \qquad (3.36)$$

$$b_i^{(k+1)} = \begin{cases} b_i^{(k)} & i \leqslant k \\ b_i^{(k)} - m_{ik}b_k^{(k)} & i=k+1\ldots n \end{cases} \qquad (3.37)$$

donde:

$$m_{ik} = a_{ik}^{(k)}/a_{kk}^{(k)} \qquad i=k+1\ldots n$$

El efecto de estos cálculos es introducir ceros en las posiciones de la columna K-ésima por debajo de la diagonal; por tanto, después de n-1 etapas, la matriz resultante $A^{(n)}$ es triangular superior, completándose el proceso llamado de eliminación hacia adelante. Para obtener a partir de $A^{(n)}$ la solución, realizamos la llamada substitución hacia atrás, ésto es, resolvemos el sistema triangular $A^{(n)}X=B^{(n)}$ mediante:

$$X_n = b_n^{(n)}/a_{nn}^{(n)}$$

$$X_i = (b_i^{(n)} - a_{i,i+1}^{(n)}X_{i+1} - \ldots - a_{i,n}^{(n)}X_n)/a_{ii}^{(n)} \qquad (3.38)$$

$$i=n-1,n-2\ldots 1$$

Al implementarse en un ordenador, los elementos calculados según (3.36), (3.37) y (3.38) se escriben encima de sus predecesores, con lo que al final el vector B contendrá la solución X. Además, los elementos de la forma

$$a_{ik}^{(k+1)} \qquad k<i\leqslant n$$

que se hacen nulos según (3.36), no es preciso calcularlos explícitamente. Basta con ignorarlos como se hace en (3.38).

Como casi siempre las divisiones consumen más tiempo que las multiplicaciones, el número total de aquellas se puede reducir a n si se almacenan los inversos de los elementos diagonales (pivotes), en lugar de ellos mismos.

Una versión muy conocida es la eliminación de Gauss-Jordan (diagonalización). El método es similar al anterior, pero ahora los elementos por encima de la diagonal se eliminan tambien, de manera que no es precisa la sustitución hacia atrás. En cuanto a número de operaciones y eficiencia son equivalentes, utilizándose preferentemente la triangularización.

Ambos procedimientos son rentables, cuando el sistema de ecuaciones hay que resolverlo cada vez para valores diferentes de la matriz A. Sin embargo, en muchos casos, como ocurre en los métodos de solución del reparto de cargas con jacobiano constante, la misma matriz A se utiliza para valores diferentes del vector B. Entonces conviene utilizar alguno de los algoritmos denominados descomposición o factorización triangular, que básicamente lo que hacen es una eliminación gaussiana en la que se guardan de alguna manera, para su posterior utilización, las operaciones que se están ejecutando, con lo que trás una sola realización del proceso, se pueden resolver sucesivos problemas con vectores B diferentes.

Estos algoritmos [122] se basan en el hecho de que una matriz A no singular se puede descomponer en la forma

$$A = LU \tag{3.39}$$

donde L es triangular inferior y U triangular superior (denominadas a veces tablas de factores). Si conocida A, pretendemos calcular L y U de (3.39) veremos que existen n grados de libertad (n^2+n incógnitas y n^2 ecuaciones). La única diferencia real entre los métodos que veremos, es la forma en que especifican esas incógnitas de más. Unos toman la diagonal de L o U como la unidad; en otros se exige que los elementos diagonales de L y U sean iguales, etc.

El proceso de solución del sistema de ecuaciones lineales consta de tres etapas:

1) Factorización de la matriz A:

$$A = L \cdot U$$

2) Obtención de Y de la ecuación:

$$LY = B \tag{3.40}$$

3) Finalmente, obtención de X de la ecuación:

$$UX = Y \tag{3.41}$$

La equivalencia entre la eliminación gaussiana y la descomposición triangular es tal, que el vector Y es el mismo que el B trás el proceso de reducción de A. Por tanto, la segunda etapa

corresponde a la eliminación hacia adelante, y la tercera a la substitución hacia atrás. Obsérvese que la etapa 1 sirve para diferentes vectores B, y no hay que repetirla mientras no se produzcan cambios en A.

Los elementos de L y U se pueden acomodar en el espacio ocupado inicialmente por A. La diferencia significativa con la eliminación gaussiana, es que la parte de A que va cayendo en desuso conforme avanza el proceso de eliminación, se utiliza ahora para almacenar L.

La otra diferencia es, que no se precisa ir evaluando todos los coeficientes intermedios de la forma $a_{ij}^{(k)}$, sino que los términos de L y U se calculan directamente, como veremos a continuación. Aunque ésto no suponga ahorro de memoria, puede ahorrar tiempo de acceso a la misma, y reducir los errores de redondeo. Por ello se les suele denominar métodos de eliminación compacta.

El método de Doolittle [122] da el valor 1 a los elementos de la diagonal de L. Los coeficientes de L y U vienen dados por (para k=1).

$$l_{11} = 1$$
$$u_{1j} = a_{1j} \qquad j = 1...n \tag{3.42}$$

y para k=2...n

$$l_{kk} = 1$$

$$l_{kj} = \frac{1}{u_{jj}} \left(a_{kj} - \sum_{m=1}^{j-1} l_{km} . u_{mj} \right) \qquad j=1...k-1$$

$$u_{kj} = a_{kj} - \sum_{m=1}^{k-1} l_{km} \cdot u_{mj} \qquad j=k...n \qquad (3.43)$$

El orden de los cálculos es como indica la figura:

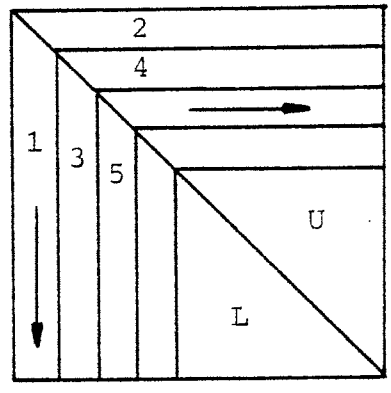


fig. 3.5

Es decir, la primera fila de L, la primera fila de U, segunda fila de L, etc.

Por el contrario, el método de Crout (llamado a veces método de Cholesky generalizado) asigna el valor 1 a la diagonal de U. Los valores de sus coeficientes son (para k=1):

$$u_{11} = 1$$
$$l_{i1} = a_{i1} \qquad i=1...n \qquad (3.44)$$

y para k=2....n

$$
\begin{aligned}
u_{kk} &= 1 \\
l_{ik} &= a_{ik} - \sum_{m=1}^{k-1} l_{im} \cdot u_{mk} \qquad i=k\ldots n \\
u_{kj} &= \frac{1}{l_{kk}} \left(a_{kj} - \sum_{m=1}^{k-1} l_{km} \cdot u_{mj}\right) \quad j= k+1\ldots n
\end{aligned}
\tag{3.45}
$$

Los elementos se calculan en el orden que indica la figura:

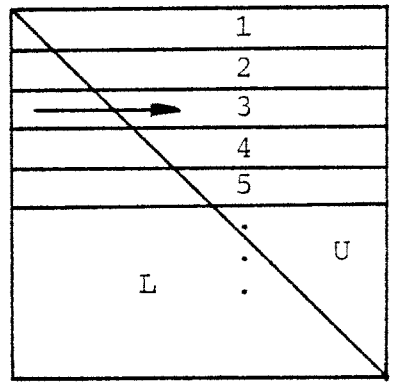


fig. 3.6

es decir, primera columna de L, primera fila de U, etc.

Para el caso particular de matrices simétricas, el método más popular es el de Cholesky, que descompone la matriz A en:

$$A = L \cdot L^t \tag{3.46}$$

A la ecuación anterior se llega por el siguiente proceso: La factorización general de A en LU destruye su simetría, pero se observa que

$$l_{ij} = \frac{u_{ji}}{u_{jj}} \qquad (j<i) \tag{3.47}$$

lo que quiere decir que se cumple la relación $U=D.L^t$ siendo D una matriz diagonal cuyos términos valen u_{ii}.

Por tanto A queda de la forma:

$$A = L.D.L^t \tag{3.48}$$

Ahora podemos descomponer D en dos matrices y agregarlas a L y L^t, es decir

$$A = L.D^{1/2}.D^{1/2}.L^t = L_1 \cdot L_1^t \tag{3.49}$$

llegándose a la expresión (3.46).

Si además de ser simétrica, A es definida positiva, los elementos de D son todos positivos (ya que son precisamente los autovalores de A), con lo que no aparecerán números complejos en las raices cuadradas de (3.49).

Los coeficientes de L tal que $A=L.L^t$ son:

$$l_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2 \right)^{1/2} \tag{3.50}$$

$$l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} \cdot l_{jk} \right) \qquad j<i$$

La descomposición de Cholesky se puede llevar a cabo con casi la mitad de cálculos y de memoria que en el caso asimétrico.

Su inconveniente es el tener que realizar n raices cuadradas, que suelen consumir más tiempo que otras operaciones aritmeticas (Véase Apéndice 4).

En ocasiones, conviene resolver (3.33) en forma particionada, como se indica a continuación:

$$AX = \begin{pmatrix} D & G \\ B & E \end{pmatrix} \cdot \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \qquad (3.51)$$

(se trabajará sólo con dos bloques, sin pérdida de generalidad, pués E o D se pueden particionar a su vez recursivamente hasta que se desee).

La solución clásica de (3.51) sería:

$$\begin{aligned} X_2 &= (E-BD^{-1}G)^{-1}(b_2-BD^{-1}b_1) \\ X_1 &= D^{-1}(b_1-GX_2) \end{aligned} \qquad (3.52)$$

Sin embargo, la forma práctica de resolver (3.51) es utilizando una triangularización por bloques, ya que no es rentable calcular inversas de matrices como indica (3.52). La factorización de A se puede realizar así:

$$A = L_a U_a = \begin{pmatrix} L & 0 \\ W & L_e \end{pmatrix} \cdot \begin{pmatrix} U & V \\ 0 & U_e \end{pmatrix} \qquad (3.53)$$

donde:

$$
\begin{aligned}
D &= LU \\
V &= L^{-1}G \\
W &= BU^{-1} \\
L_e U_e &= E - W.V
\end{aligned}
\tag{3.54}
$$

Con A puesta en la forma (3.53), la obtención de X se hace por un proceso de substitución hacia atrás por bloque, en los siguientes pasos:

$$
\begin{aligned}
Ly_1 &= b_1 \\
b_2^1 &= b_2 - W.y_1 \\
L_e y_2 &= b_2^1 \\
U_e x_2 &= y_2 \\
y_1^1 &= y_1 - Vx_2 \\
Ux_1 &= y_1^1
\end{aligned}
\tag{3.55}
$$

Es inmediato ver que el resultado de este proceso es el mismo que el obtenido de (3.52).

Bunch y Rose [21], demostraron que el número de operaciones aritméticas requeridas para realizar esta triangularización en bloques, es el mismo que para descomponer A globalmente en L_a U_a. Los párrafos siguientes describirán sin embargo, situaciones en las que se puede obtener alguna ventaja con las particiones.

Para matrices llenas, el proceso de partición no ahorra tiempo respecto a la solución con la matriz completa, pero permi-

te resolver sistemas más grandes cuando está limitada la capacidad de memoria. Además, si la submatriz D se compone a su vez de bloques diagonales:

$$A = \left(\begin{array}{ccc|c} D_1 & & & \\ & D_2 & & G \\ & & \ddots & \\ & & D_n & \\ \hline & B & & E \end{array}\right) \qquad (3.56)$$

y E no es muy grande, se puede realizar un procesamiento en paralelo del problema (véanse los próximos capítulos).

También se utiliza esta técnica cuando se dispone de la tabla de factores de A, y se pretende factorizar una matriz A' que es una modificación simple de A (por ejemplo la adición de una fila y columna). En ese caso una parte importante de las operaciones ya está hecha.

Para matrices vacías, George [42] presentó otras dos factorizaciones alternativas a la (3.53), demostrando que bajo ciertas hipótesis, relativas a la estructura de G y B, eran más eficientes (en operaciones aritméticas y almacenamiento).

Bunch y Rose [21] hicieron además la importante observación, de que si B,G,L y U son suficientemente vacías comparadas con W y V, se pueden ahorrar operaciones en la substitución hacia atrás utilizando W y V sólo implícitamente a través de sus definiciones. Por ejemplo, el segundo proceso indicado por (3.55)

$$b_2' = b_2 - Wy_1 \qquad (3.57)$$

se puede escribir como:

$$\begin{aligned} Uz &= y_1 \\ \bar{z} &= Bz \\ b_2' &= b_2 - \bar{z} \end{aligned} \qquad (3.58)$$

ya que, según la tercera ecuación de (3.54),

$$W = BU^{-1}$$

Mediante (3.58) se ahorra también almacenamiento, pués W puede ser mucho más llena que B.

4. MATRICES VACIAS.

4.1 INTRODUCCION Y CAMPOS DE APLICACION.

Se puede considerar que una matriz es vacía o dispersa ("sparse" en inglés) cuando se obtiene alguna ventaja teniendo en cuenta el porcentaje y/o la distribución de elementos nulos. Hay casos en los que el número de elementos no nulos es alto, y sin embargo su posición, en forma de bloques por ejemplo, acarrea algún beneficio en su tratamiento como matriz vacía.

La ventaja más evidente del tratamiento de matrices vacías es el ahorro en memoria, que permite estudiar sistemas más grandes y modelos más precisos. Además, al realizar las operaciones, es más fácil organizarlas de manera que se eviten las sumas de elementos nulos y las multiplicaciones por la unidad.

El principal objetivo deseado al trabajar con matrices vacías, es procurar que sigan siendo lo más vacías posible al manipularlas, lo cual depende como veremos de una adecuada ordenación de las ecuaciones y las variables. Esta idea básica es tan simple que, precisamente por eso y a pesar de las técnicas desarrolladas por Gauss, Jacobi y otros hace más de un siglo, fue completamente pasada por alto por matemáticos e ingenieros durante casi dos décadas de utilización de computadores, hasta comienzo de los años 60 en que Tinney et al. en la Bonneville Power Administration comenzaron a redescubrirla.

Las matrices vacías aparecen cuando existe una relativa ausencia de interconexión entre las variables que intervienen en un problema, y a menudo, salvo en casos puramente matemáticos, es evidente su presencia por el propio aspecto físico del mismo.

Piénsese como ejemplo en una red de distribución de aguas como la de la siguiente figura:

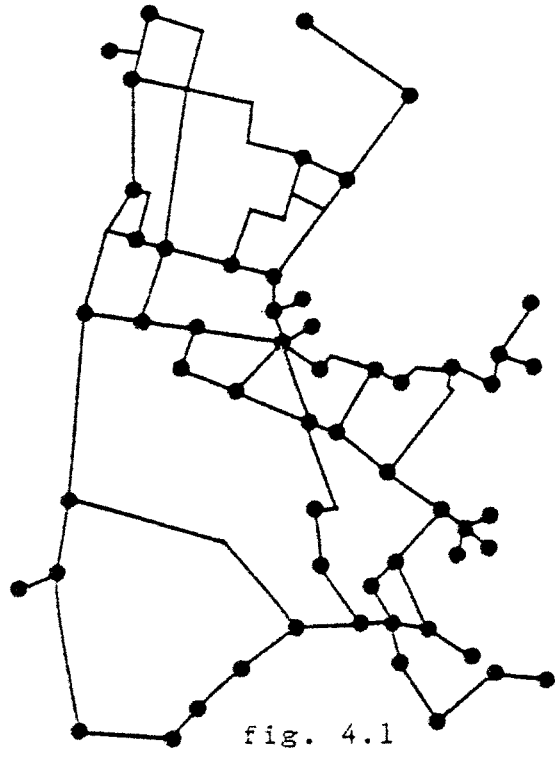


fig. 4.1

A pesar de que, en teoría, cada nudo puede estar conectado a todos los demás por una tubería, la realidad es que la media es de 1'26 conexiones/nudo, y esta estructura física debe ser aprovechada en la medida de lo posible al resolver problemas relativos a ese sistema.

Las consideraciones anteriores pueden dar una idea al lector del vasto campo de aplicación de las matrices vacías. En [28] se puede encontrar una relación bastante exhaustiva, y aquí sólo

mencionaremos las más clásicas en el campo de la ingeniería, a saber:

- Redes eléctricas
- Circuitos electrónicos
- Análisis de estructuras
- Vibraciones
- Redes hidráulicas
- Transportes
- Transferencia de calor, etc.

En todas las aplicaciones anteriores, uno se enfrenta a alguno o algunos de los siguientes problemas matemáticos:

- Ecuaciones lineales y no lineales.
- Ecuaciones diferenciales ordinarias y parciales
- Autovalores
- Combinatoria
- Programación dinámica, etc.

Y aunque la idea de partida es la misma, se han desarrollado herramientas específicas para cada caso. Así, una ordenación de las variables que puede ser conveniente para la resolución de un problema de ecuaciones diferenciales por diferencias finitas, puede no serlo en absoluto para un sistema eléctrico de potencia.

La información disponible sobre matrices vacías está muy diseminada, como consecuencia de su distinto origen. Existen además muy pocos libros, la mayoría de los cuales son el resultado escrito de simposiums o congresos.

En lo que sigue se analizará con cierta profundidad el problema de las matrices vacías, enfocadas fundamentalmente al aná-

lisis de los sistemas eléctricos de potencia.

4.2 CONCEPTOS BASICOS PRELIMINARES Y NOMENCLATURA

Considérese una matriz M (en general no simétrica). Podemos definir un grafo G(M) = (X,E), en el que X es un conjunto de vértices o nudos y E un conjunto de arcos como sigue: Se asocia un vertice $x_i \in X$ a cada fila i de M, y la pareja ordenada $(x_i, x_j) \in E$ (un arco de x_i a x_j en el grafo G) si $m_{ij} \neq 0$ y $i \neq j$. Este grafo se denomina orientado por existir un orden en la pareja de nudos en que inciden sus arcos.

Asímismo los nudos se consideran ordenados o numerados, mediante una aplicación biyectiva entre $\{1,2....N\}$ y X, siendo N = X . Esta ordenación es precisamente, junto a las técnicas de almacenamiento, la principal razón de ser de toda la literatura sobre matrices vacías.

El mayor inconveniente al tratar una matriz vacía, aparece en su eliminación gaussiana o triangularización, puesto que este proceso conlleva la aparición en diverso grado de nuevos elementos ("fill-in") que, además de complicar el almacenamiento, pueden dar lugar a que el resultado no permita el calificativo de matriz vacía. El grado de llenado para una misma matriz depende fuertemente de la ordenación de los nudos en el grafo, que será precisamente el orden en que se van eliminando las filas de la matriz (pivote). Es indistinto desde ahora hablar de matriz o grafo.

Las matrices con las que nos veremos involucrados en el problema del flujo de cargas o son simétricas, o tienen al menos una simetría estructural, aunque no numérica. Desde el punto de vista de la triangularización y la ordenación de los nudos, es indistinta una simetría estructural o númerica, hasta tal punto que casi siempre se representan las matrices con "X" en las posiciones de los elementos distintos de cero.

La siguiente figura representa una matriz no simétrica y otra simétrica:

(a) (b)

fig. 4.2

Sus grafos respectivos son:

(a) (b)

fig. 4.3

En el caso de matrices simétricas, siempre que aparezca el arco (x_i,x_j) lo hará tambien (x_j,x_i), y por tanto no aporta nada

el considerar una orientación en los arcos. Se obtiene así el grafo no orientado, que sólo indica conexión entre nudos. Para la matriz (b) anterior su grafo no orientado es:

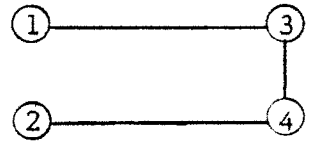
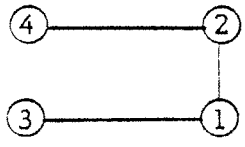


fig 4.4

Obsérvese de paso que con la ordenación de nudos anterior, no aparecen elementos nuevos en la factorización triangular, mientras que con esta otra (justo la inversa)



X	X	X	
X	X	F	X
X	F	X	F
	X	F	X

fig 4.5

aparecen cuatro elementos nuevos marcados con F en la matriz ("fill-in"). A pesar de su pequeña dimensión, el ejemplo da una idea clara de la importancia de una adecuada ordenación, lo cual resulta vital para sistemas muy grandes, como los que aparecen en redes de potencia. A partir de ahora nos referiremos a grafos no orientados.

Si para matrices llenas, la ordenación de las filas y columnas viene determinada por la elección del pivote más grande, para evitar singularidades y errores de redondeo, para matrices vacías la elección se hará buscando mantener el menor grado posible de

llenado, y más aún cuando, como es nuestro caso, la matriz tiene diagonal dominante y no existen problemas para dicha elección (que se tomará siempre en la diagonal, para mantener la simetría).

Cambiar la numeración de los nudos en el grafo G(M), equivale a multiplicar la matriz M por una matriz de permutación P del modo:

$$P.M.P^{t} \qquad (4.1)$$

Los elementos P_{ij} valen 1 si el nudo i pasa a renumerarse como j y cero en cualquier otro caso. La matriz P para pasar de la ordenación del grafo de la fig. 4.4 a la de la fig. 4.5 es

$$P = \begin{pmatrix} & & & & 1 \\ & & & 1 & \\ & & 1 & & \\ & 1 & & & \\ 1 & & & & \end{pmatrix}$$

No es preciso considerar explícitamente la matriz P en la práctica, ni realizar las operaciones indicadas en (4.1). Mediante un solo vector se puede realizar un direccionamiento indirecto, de tal manera que el nudo i es realmente el que indica la posición i-ésima de dicho vector. Para el ejemplo anterior dicho vector es:

$$\begin{pmatrix} 4 & 3 & 2 & 1 \end{pmatrix}$$

El grafo reducido G_i se obtiene quitando el nudo i y los arcos que en él inciden, y añadiendo arcos nuevos entre los nudos en que incidían los arcos borrados, si no eran adyacentes (dos nudos son adyacentes si están unidos por un arco).

La principal propiedad del grafo reducido así definido, es que representa precisamente a la matriz trás haber realizado la eliminación gausiana sobre la fila i . Además, el número de arcos nuevos aparecidos en el grafo, representa los nuevos elementos distintos de cero aparecidos en la matriz (teniendo en cuenta que cada arco no orientado representa un elemento y su simétrico). De nuevo utilizamos el ejemplo de la fig. 4.5, dibujándolo en las diferentes etapas de la eliminación ordenada:

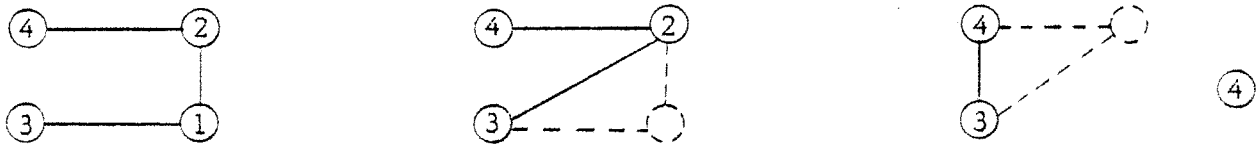


fig 4.6

Los dos nuevos arcos que han aparecido al eliminar los nudos 1 y 2 corresponden a las "F" de la matriz de la fig. 4.5.

Recuérdese que la etapa de triangularización de la matriz para la fila i, consiste en realizar las transformaciones elementales necesarias para eliminar los elementos por debajo de la diagonal i (eliminación por columnas). Ello explica la aparición

de los elementos "F".

Como contrapartida, las diferentes etapas para el mismo grafo con la ordenación de la fig. 4.4 son:

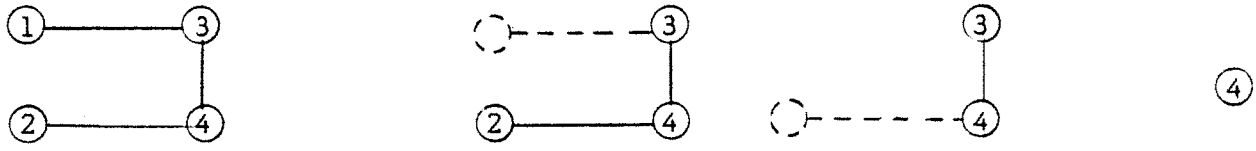


fig 4.7

observándose que no aparece ningún arco nuevo. A la matriz equivalente al grafo de las figuras anteriores se le llama matriz de eliminación perfecta, porque existe al menos una ordenación para la que no se presentan nuevos elementos en la triangularización.

Para nosotros una ordenación óptima será aquella que introduzca el menor número posible de elementos nuevos en la triangularización, por razones obvias.

Llamaremos grafo llenado (o lleno) al grafo representativo de la matriz ya triangularizada.

Introduciremos otras definiciones básicas, salvo las más conocidas o elementales, como grado de un nudo, camino, bucle, grafo conexo, subgrafo, etc.

Considérese el grafo no orientado G(M) = (X,E) ya definido anteriormente. Para un subconjunto de nudos YCX, se define el conjunto adyacente a Y como:

$$\text{Ady}\ (Y) = \left\{ x \in X - Y \mid \{x,y\} \in E \text{ para algún } y \in Y \right\}$$

La distancia d(x,y) entre dos nudos x,y es la longitud (número de arcos) del camino más corto posible entre dichos nudos. Se define la excentricidad de un nudo x como:

$$e(x) = \max\{d(x,y) \mid y \in X\}$$

y el diámetro de un grafo

$$d(G) = \max \left\{ e(x) \mid x \in X \right\}$$

Un nudo x se dice que es periférico si su excentricidad es la misma que el diámetro del grafo. Un nudo pseudoperiférico es aquel cuya excentricidad está próxima al diámetro.

Un árbol radical es un árbol construido a partir de un nudo particular llamado la raiz del árbol. Si (r,s)...(x,y) es el camino (único) desde la raiz r al nudo y, a x se le denomina el padre de y. Esta relación de paternidad caracteriza completamente al árbol radical, de manera que tal árbol puede ser representado convenientemente utilizando N posiciones de memoria (si el árbol tiene N nudos), asignando a cada posición el padre del nudo respectivo.

Una ordenación monótona para un árbol radical es aquella que numera siempre a un nudo antes que a su padre. La raiz entonces será numerada la última.

Sea el conjunto S C X. El nudo $y \in S$ se dice alcanzable desde un nudo x a través de S si existe un camino $(x, v_1, v_2, \ldots, v_k, y)$ de x a y tal que $v_i \in S$ $1 \leqslant i \leqslant k$. El conjunto alcanzable de x a través de S, llamado Alc(x,S) se define como:

$$Alc(x,S) = \left\{ y \in X-S \mid y \text{ es alcanzable desde } x \text{ a través de } S \right\}$$

S puede ser vacío, y los caminos de longitud 1.

Si $S = \phi$ se tiene que $Alc(x, \phi) = Ady(x)$.

Una partición P del grafo G es el conjunto

$$P = Y_1, Y_2, \ldots Y_p$$

con

$$\bigcup_{i=1}^{p} Y_i = X$$

$$Y_i \cap Y_j = \phi \qquad i \neq j$$

Una ordenación se dice que es compatible con la partición P si para cada $Y \in P$ sus nudos están enumerados consecutivamente.

Definimos el grafo cociente de G con respecto a una partición P, al siguiente grafo:

$$G/P = (P, \mathcal{E})$$

donde $(Y_i, Y_j) \in \mathcal{E}$ si y sólo si $Ady(Y_i) \cap Y_j \neq \phi$

Cuando a su vez G/P es un árbol, lo llamamos árbol cociente. Se dice que el árbol cociente es maximal si P no admite una partición Q más fina que dé un árbol cociente. Escrito formalmente: Si no existe una partición $Q = \{Z_1, Z_2 \ldots Z_q\}$ tal que $p<q$ y para cada i, $Z_i \subset Y_k$ para algún $1 \leqslant k \leqslant p$. Para no confundir ideas considérese el siguiente ejemplo:

La partición $P = \{Y_1, Y_2\}$ es maximal, y sin embargo existe una partición Q que resulta en un árbol con tres miembros $\{(2), (1,3), (4)\}$, pero no satisface las condiciones anteriores, pués (1,3) no está incluido en Y_1 ni en Y_2.

En [42] se da una condición suficiente para que una partición sea maximal (es decir, genere un árbol cociente maximal).

Estaremos interesados principalmente en una clase particular de árbol cociente, la denominada estructura de niveles. Formalmente una estructura de niveles de un grafo G=(X,E) es una partición de X

$$L = \{L_o, L_1 \ldots L_k\}$$

tal que:

$$\text{Ady}\ (L_i) \subset L_{i-1} \cup L_{i+1} \qquad i=1....k-1$$

$$\text{Ady}\ (L_o) \subset L_1$$

$$\text{Ady}\ (L_k) \subset L_{k-1}$$

El número k es la longitud o profundidad de la estructura. La cantidad

$$\max\left\{ |L_i| \ ,\ i = 1 \ ...k \right\}$$

se denomina anchura de L. El árbol cociente G/L es una simple cadena.

En la práctica produciremos estructuras de niveles radicales, para las que, si x es la raiz:

$$L_o\ (x) = x$$

$$L_i\ (x) = \text{Ady}\left(\bigcup_{j=o}^{i-1} L_j(x) \right) \qquad i = 1\ .....k(x)$$

k(x) será la excentricidad de x. La siguiente figura es un ejemplo de estructura de nivel radical:

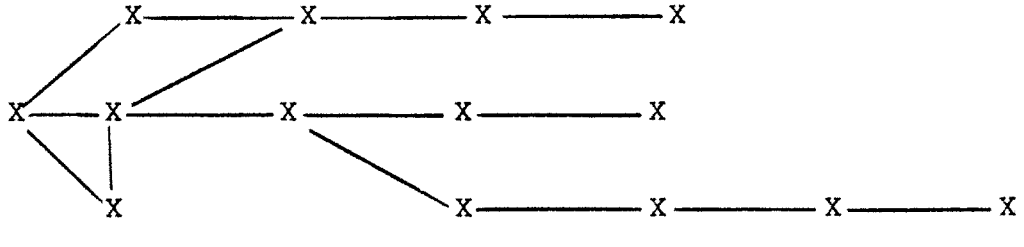


fig. 4.9

El hecho de que el grafo cociente G/L sea una cadena, hace que en la matriz asociada, particionada según cada nivel L_i, y ordenada de forma compatible con la partición, aparezca una estructura de bloques tridiagonal, como en el siguiente ejemplo (que no es del tipo radical):

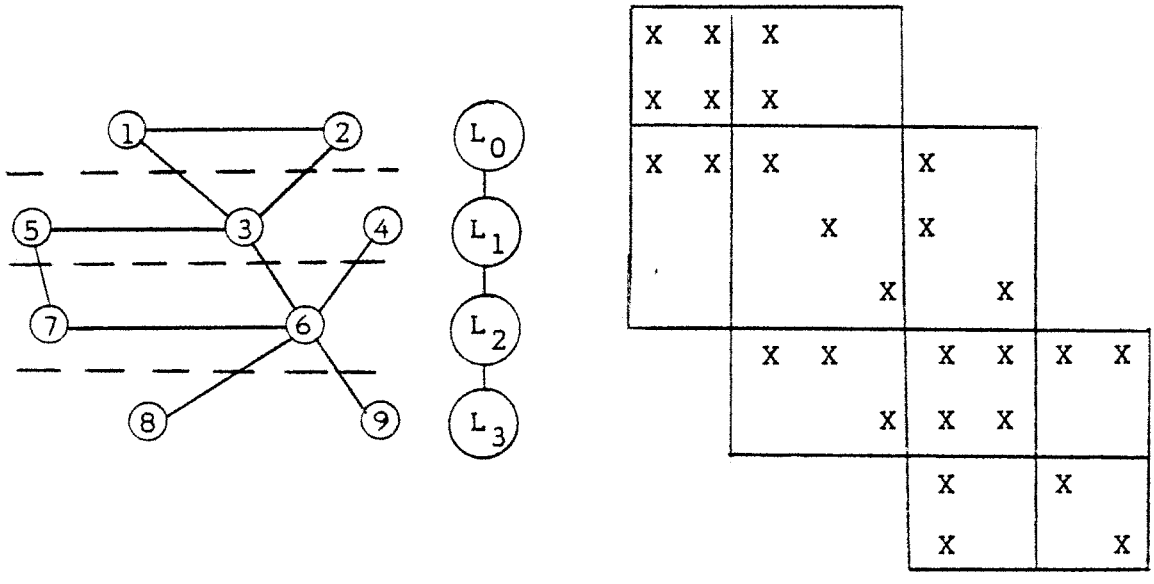


fig 4.10

El ejemplo evidencia la utilidad de los conceptos introducidos anteriormente para el tratamiento de las particiones en matrices vacías. Algunas otras definiciones se darán cuando se necesiten.

Por último, para un tipo especial de ordenación, como es la ordenación en banda que será muy importante para nosotros, conviene familiarizarse con los siguientes conceptos.

Para una matriz A (simétrica) de N x N se define el ancho de banda B como:

$$B = \max_i \beta_i \tag{4.2}$$

donde β_i es la anchura de la fila i dada por:

$$\beta_i = i - f_i \qquad i=1....N \tag{4.3}$$
$$f_i = \min \left\{ j \quad : a_{ij} \neq 0 \right\}$$

El perfil P se define como:

$$P = \sum_{i=1}^{n} \beta_i \tag{4.4}$$

Siendo w_i el número de columnas activas para la fila i (una columna j es activa en una fila i si $j>i$ y existe un elemento distinto de cero en esa columna para una fila $k \leqslant i$) se define el frente de onda W de la matriz A como

$$W = \max_i \; w_i \tag{4.5}$$

Puesto que A es simétrica se cumple que:

$$P = \sum_{i=1}^{n} \beta_i = \sum_{i=1}^{n} w_i \tag{4.6}$$

La envoltura se define como:

$$Env = \left\{ (i,j) \; : \; f_i \leqslant j \leqslant i \right\} \tag{4.7}$$

y la envoltura traspuesta:

$$Tenv = \left\{ (i,j) \; : \; j \leqslant i \quad y \; \exists \; k \geqslant i \quad \middle| \quad a_{kj} \neq 0 \right\} \tag{4.8}$$

Se cumple que

$$|Env| = P + N \quad ; \quad B \geq W \tag{4.9}$$

Definiremos formalmente el grado de llenado de una matriz A al que antes se aludió. Siendo A simétrica se puede factorizar como $A = LL^t$ (véase el capítulo 3). El llenado de A se define como:

$$Llen\ (A) = \left\{ (i,j) : \ a_{ij} = 0 \ \ y \ \ l_{ij} \neq 0 \right\} \tag{4.10}$$

Se cumple además que [67]:

$$Llen\ (A) \subseteq Env\ (A) \tag{4.11}$$

lo cual es un resultado muy interesante, pués nos garantiza que no aparecerán elementos nuevos en la triangularización de A fuera de la envoltura (parece conveniente entonces que A tenga una envoltura pequeña).

Aclaremos todas las definiciones anteriores con un ejemplo;

G(A)

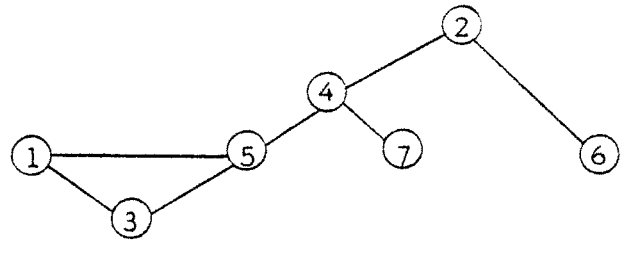


fig 4.11

La matriz A del grafo anterior es:

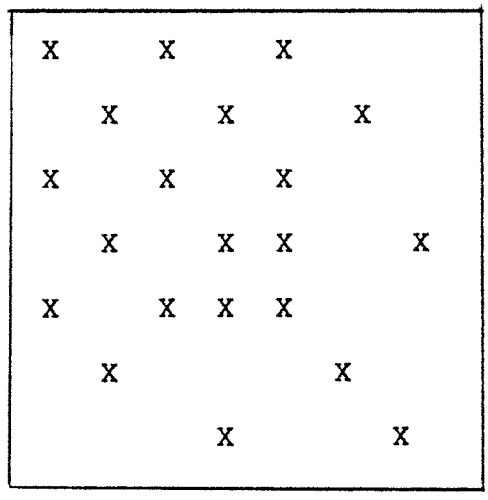


fig. 4.12

y los valores de los parámetros anteriores para cada fila:

i	f_i	β_i	w_i
1	1	0	2
2	2	0	4
3	1	2	3
4	2	2	3
5	1	4	2
6	2	4	1
7	4	3	0

B = 4 ; P = 15 ; W = 4

|Env| = 22 ; |Tenv| = 20

fig. 4.13

La matriz L tal que $A = LL^t$ será :

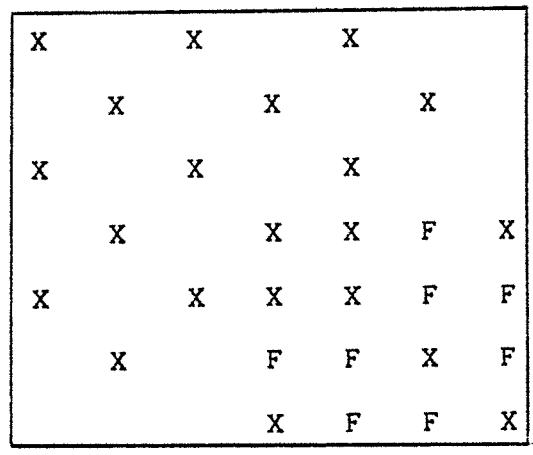


fig. 4.14

cumpliéndose la relación dada por (4.11).

4.3 ALMACENAMIENTO DE MATRICES VACIAS

Para comprender la necesidad de almacenar en una matriz vacía sólo los elementos no nulos, considérese el siguiente ejemplo: Una matriz de dimensión 1000, necesitaría 10^6.n posiciones de memoria si la almacenásemos en su totalidad, siendo n las posiciones que necesita un solo elemento. Si suponemos que existen una media de 3 elementos no nulos por fila, lo que es bastante realista, se necesitarían 3000.n posiciones de memoria. En realidad se necesitan algunas más, pués además del conjunto de elementos no nulos de la matriz, llamado a veces conjunto primario, se necesitan unos conjuntos secundarios auxiliares de números enteros que permitan reconocer esos elementos.

De cualquier forma la diferencia es bien notable. La capacidad de memoria requerida en el segundo caso puede ser satisfecha por muchos ordenadores domésticos, mientras que en el primer caso

sería necesaria la utilización de memoria auxiliar, salvo para ordenadores muy grandes.

Además de la capacidad de memoria, está el tiempo de cálculo, que sería mayor en el primer caso, aunque se utilizase la técnica de testear el valor del elemento para ver si es nulo antes de ser operado.

No existe un esquema óptimo para almacenar la estructura de una matriz vacía. Cada esquema debe ajustarse al caso particular, aunque todos deben cumplir dos requisitos previos: Que se acceda fácilmente a los elementos de la matriz, y que la capacidad de memoria requerida se mantenga baja. Obviamente existe un conflicto entre los dos objetivos.

Básicamente existen unas técnicas entre las cuales elegir, dependiendo de la estructura de la matriz y de las operaciones que haya que realizar con ella, aunque luego cada programador las particulariza, en función de la máquina de que disponga y del tipo de lenguaje en que programe.

Dichas técnicas se pueden clasificar de la siguiente manera:

- Para matrices con estructuras irregulares
    - Almacenamiento estático.
        a) Identificación binaria.
        b) "Hashing".
        c) Almacenamiento compacto o sistemático.
    - Almacenamiento dinámico.

d) Almacenamiento aleatorio por coordenadas.

e) Almacenamiento aleatorio con enlaces.

- Para matrices con cierta regularidad en su estructura:
  - Matrices en banda diagonal
  - Matrices con banda de anchura variable.

El grupo de matrices sin una estructura aparente es el que más nos interesa, desde el punto de vista de las redes eléctricas. Si sólo estamos interesados en acceder a los elementos de la matriz, sin que vayan a aparecer elementos adicionales durante la operación, se utilizará un almacenamiento estático, llamado así porque no está diseñado para permitir fácilmente cambiar la estructura de la matriz. Por el contrario, se utilizará algún tipo de almacenamiento dinámico cuando se está interesado en añadir cómodamente elementos nuevos.

La identificación binaria consiste en almacenar la estructura de la matriz a base de ceros y unos (un bit por cada elemento). Es difícil de implementar en lenguajes de alto nivel, pero puede ser interesante si se dispone de instrucciones de manejo de bits individualmente.

La siguiente matriz:

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2.6 & 0 & 3.1 \\ -1.2 & 0.3 & 0 & 0 & 2.3 \end{bmatrix}$$

fig. 4.15

Almacenaría su estructura como:

$$\underbrace{0\ 0\ 0\ 0\ 0}_{\text{fila 1}} \qquad \underbrace{0\ 0\ 1\ 0\ 1}_{\text{fila 2}} \qquad \underbrace{1\ 1\ 0\ 0\ 1}_{\text{fila 3}}$$

y los elementos no nulos en un conjunto unidimensional:

$$\{2.6,\ \ 3.1,\ \ -1.2,\ \ 0.3,\ \ 2.3\}$$

Con las técnicas de "hashing"[98], muy conocidas en otros ámbitos, la posición de un elemento en el conjunto primario se calcula como una función predeterminada de su fila y columna. Dicha función variará según el número de elementos que se estima que habrá por fila. La técnica se complementa con un procedimiento para manejar colisiones, pués combinaciones diferentes de fila y columna pueden dar lugar a una misma posición. Este procedimiento puede ser interesante si se logra un buen compromiso entre el número de colisiones y la memoria reservada inútilmente.

El llamado almacenamiento compacto o sistemático es bastante popular [93]. Almacena los elementos secuencialmente por filas (o columnas). Para acceder a ellos se utilizan dos vectores enteros. Uno de la misma dimensión que el anterior y que contiene la columna (o fila) del elemento correspondiente. Otro de dimensión igual al número de filas más uno, que indica dónde comienza cada fila (o columna). El último elemento de este vector se utiliza para saber dónde acaba la última fila (o columna). Por ejemplo, la matriz anterior en un esquema por filas quedaría:

$$VAL = \{\ 2.6\ ,\quad 3.1\ ,\quad -1.2\ ,\quad 0.3,\quad 2.3\}$$

$$COL = \{\ 3\ ,\quad 5\ ,\quad 1\ ,\quad 2\ ,\quad 5\}$$

$$COMFI = \{\ 0\ ,\quad 1\ ,\quad 3\ ,\quad 6\}$$

donde el 0 de COMFI indica que no existen elementos en la primera fila. El número de elementos para una fila m que contenga alguno viene dado por COMFI(m+1)-COMFI(m). Obsérvese que sería muy lento añadir elementos nuevos o suprimir alguno, por ello decimos que es un almacenamiento estático.

Este procedimiento admite muchas variantes, que van desde intercalar elementos mudos o ficticios que indican comienzos de fila, por ejemplo, hasta guardar doble información en un mismo número entero, multiplicando por un factor arbitrariamente grande una de las informaciones y sumándosela a la otra.

Dentro de los almacenamientos dinámicos, el más inmediato puede ser el almacenamiento aleatorio por coordenadas. Los elementos se van almacenando en el orden en que se van leyendo, que en general será arbitrario, guardándose además sus coordenadas bien en forma conjunta o por separado. Una de las ventajas es que los elementos extras se añaden simplemente sin perturbar a los demás. Para la matriz anterior:

$$VAL = \{\ 0.3\ ,\quad 3.1\ ,\quad -1.2\ ,\quad 2.6\ ,\quad 2.3\}$$

$$FIL = \{\ 3\ ,\quad 2\ ,\quad 3\ ,\quad 2\ ,\quad 3\}$$

$$COL = \{\ 2\ ,\quad 5\ ,\quad 1\ ,\quad 3\ ,\quad 5\}$$

En el caso concreto de formación de una matriz de admitancias de nudos de una red, los elementos de la diagonal, que se ven involucrados en sumas, conviene almacenarlos en un vector aparte.

El acceso a los elementos en este tipo de almacenamiento se hace por búsqueda, y es muy poco eficaz. Sin embargo, una de las operaciones que se pueden realizar muy fácilmente, es la multiplicación de la matriz por un vector. La eliminación gaussiana es muy difícil de llevar a cabo, puesto que estamos interesados en filas enteras, no en elementos aislados. Por este motivo no se utiliza este almacenamiento en la solución de sistemas de ecuaciones lineales por dicho procedimiento. El sistema de ecuaciones se podría resolver en ese caso por un método iterativo como el del gradiente conjugado, que sólo utiliza multiplicaciones como la mencionada anteriormente.

Para soslayar la dificultad de búsqueda de un elemento, y seguir manteniendo la aleatoriedad en el almacenamiento, que es muy útil como hemos visto a la hora de modificar la matriz, se recurre al almacenamiento con enlaces, eligiendo según el caso alguno o algunos de los siguientes [28]:

- puntero al siguiente elemento de la fila
- puntero al siguiente elemento de la columna
- puntero al elemento anterior de la fila
- puntero al elemento anterior de la columna.

Además conviene disponer un vector que indique dónde comienza cada fila (o columna). Cuantos más enlaces se utilicen, más fácil será acceder a un elemento en concreto, bien por filas o por columnas, aunque la capacidad de memoria requerida será mayor, como quedó dicho al principio.

Para una eliminación gaussiana por filas, es suficiente con un enlace por filas apuntando al elemento siguiente. Escribamos el ejemplo de la matriz anterior en este caso:

$$\text{VAL} = \left\{ 0.3 \ , \quad 3.1 \ , \quad -1.2 \ , \quad 2.6 \ , \quad 2.3 \right\}$$

$$\text{ENLA} = \left\{ \quad 5 \ , \quad -1 \quad , \quad 1 \quad , \quad 2 \quad , \quad -1 \right\}$$

$$\text{COMFI} = \left\{ \quad 0 \ , \quad 4 \quad , \quad 3 \right\}$$

Obsérvese que un número negativo en los enlaces indica que ese elemento es el último de la fila. Esa posición se puede utilizar, si se desea, para almacenar el número de elementos de la fila. El 0 en COMFI indica, como antes, que no hay elementos en esa fila.

El acceso a cada elemento es algo más lento, debido al direccionamiento indirecto que hay que utilizar. Además, para que el método sea eficiente, se debe utilizar cuando toda la matriz esté en memoria central.

A pesar de ello, en los casos en que la matriz deba sufrir modificaciones, éste será probablemente el mejor procedimiento.

En el epígrafe siguiente veremos que ciertos ordenamientos de las filas de la matriz, colocan sus elementos alrededor de la diagonal en forma de banda. Este ordenamiento es útil cuando el problema físico que representa la matriz (o su grafo) posee cierta regularidad en la relación entre sus variables (p.e. elementos finitos).

En estos casos existen almacenamientos eficientes en forma de banda que no necesitan conjuntos auxiliares para localizar a los elementos de la matriz [58].

Considérese la siguiente matriz simétrica con anchura de banda constante:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & & \\ a_{21} & a_{22} & a_{23} & a_{24} & \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ & a_{42} & a_{43} & a_{44} & a_{45} \\ & & a_{53} & a_{54} & a_{55} \end{bmatrix}$$

fig. 4.16

Sólo se utiliza la mitad inferior en el almacenamiento, y se almacenan elementos ficticios al principio para hacer que cada fila tenga el mismo número de ellos, de la siguiente manera:

$$VAL = \left\{ X, X, a_{11} \mid X, a_{21}, a_{22} \mid a_{31}, a_{32}, a_{33} \mid a_{42}, a_{43}, a_{44} \mid a_{53}, a_{54}\ a_{55} \right\}$$

En general, el elemento $a_{ij}(j \leqslant i)$ estará almacenado en la posición ib-i+j del conjunto VAL, siendo b el número de elementos por fila.

Esta es sólo una de las muchas soluciones posibles. Otra puede ser almacenar la matriz en un conjunto de dos dimensiones (n x b), utilizando los mismos elementos mudos que anteriormente. El elemento $a_{ij}(j \leqslant i)$ ocupará ahora la posición (i,b-i+j).

Si la anchura de banda es variable, no es ventajoso almacenar la matriz como anteriormente se ha hecho, puesto que se necesitarían guardar ceros en las filas cuya anchura fuese pequeña, para mantener la regularidad. En ese caso se puede hacer un almacenamiento con anchura de banda variable, necesitándose entonces un conjunto auxiliar que apunte a las direcciones de la diagonal. Tómese como ejemplo la siguiente matriz:

$$A = \begin{pmatrix} a_{11} & & & & a_{15} \\ & a_{22} & a_{23} & & a_{25} \\ & a_{32} & a_{33} & & a_{35} \\ & & & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{pmatrix}$$

fig. 4.17

almacenada como:

$$VAL = \{ a_{11} \mid a_{22} \mid a_{32}, a_{33} \mid a_{44} \mid a_{51}, a_{52}, a_{53}, a_{54}, a_{55} \}$$
$$DIA = \{ 1 , 2 , 4 , 5 , 10 \}$$

El elemento a_{ij} ($j \leq i$) se localizará en la posición DIA(i)+j-i del conjunto VAL, y el número de elementos en la fila i(i>1) se obtendrá como DIA(i)-DIA(i-1). Este esquema fue propuesto por Jennings [58].

Por último, se han utilizado algunas técnicas de almacenamiento que están íntimamente relacionadas con el método de ordenación presentado conjuntamente. Tal es el caso de la disección anidada y otras semejantes que en sucesivos epígrafes se estudiarán más detenidamente.

4.4 ORDENACION OPTIMA

En el epígrafe de conceptos básicos sobre matrices vacías, se puso de manifiesto mediante un ejemplo simple, la importancia que la ordenación de los nudos tenía en el resultado final de la matriz, una vez triangularizada. Se definió tambien lo que se entendía por una ordenación óptima.

Se seguirá suponiendo que la matriz tiene estructura simétrica, o lo que es lo mismo que trabajaremos con grafos no orientados.

Cuando realizamos la eliminación en matrices vacías simétricas, el objetivo normal es elegir los pivotes en una secuencia tal, que el número de operaciones aritmeticas y/o el número de elementos nuevos aparecidos durante dicha eliminación sean mínimos. Los pivotes se elegirán de la diagonal, para no destruir la

simetría.

Algunos autores [15] han mostrado de forma experimental, que las matrices ordenadas aleatoriamente se llenan muy rápidamente conforme avanza la eliminación, deduciéndose incluso fórmulas teóricas que indican dicho crecimiento.

Idealmente sería deseable obtener la solución al problema de encontrar el óptimo en algún sentido entre todas las ordenaciones posibles, cuya esencia combinatoria lo hace irrealizable en la práctica. Por ello el calificativo de "optima" para la ordenación es sólo una forma de hablar, y los algoritmos que se presentarán a continuación son aproximaciones al óptimo para ciertos tipos de problemas.

La mayoría de las técnicas de ordenación se pueden clasificar en dos grupos: O bien utilizan como dato la estructura de la matriz inicial, sin preocuparse de etapas intermedias, o bien los pivotes se van eligiendo mediante un análisis de la matriz reducida, como se encuentra en la etapa actual, lo que exige una simulación del proceso de triangularización que permita conocer los sucesivos estadios de la matriz. Al primer grupo se les suele llamar ordenaciones "a priori" o estáticas, y al segundo ordenaciones dinámicas.

La mayoría de estas técnicas, o por lo menos las más populares, fueron descritas hace dos décadas por Tinney et al. [93,109,110], pero no han perdido en absoluto su vigencia, como lo demuestran las publicaciones más recientes [66].

En el campo de los sistemas eléctricos, los esquemas más generalizados son los tres que siguen, denominados respectivamente esquema 1, esquema 2 y esquema 3 de Tinney, debido a la gran contribución de este autor. Cada uno de ellos aumenta en complejidad, tiempo de ejecución y optimalidad.

Esquema 1. Es una ordenación "a priori". Se numeran los nudos de menor a mayor grado, es decir, empezando por los que sólo tienen un nudo conectado, y acabando con los que más nudos tienen. Este esquema no tiene en cuenta los efectos subsecuentes del proceso de eliminación. La única información que se necesita es una lista con el número de elementos distintos de cero para cada fila de la matriz original. Si dos nudos tienen el mismo grado se elige arbitrariamente uno de ellos.

Esquema 2. Se numeran los nudos de tal manera que en cada paso del proceso de eliminación, la próxima fila a operar sea la que menos nudos tiene conectados. Este esquema requiere simular el proceso de eliminación, y a fin de cuentas aplica el esquema 1, pero a la matriz reducida en lugar de la original.

Esquema 3. Se numeran las filas (o nudos) de manera que en cada paso del proceso, la siguiente fila a operar sea la que introduzca menos elementos nuevos distintos de cero. Por este procedimiento, además de simular la eliminación, es preciso en cada etapa ensayar todos los nudos que restan y averiguar qué alternativa introduce menos elementos.

Para los dos últimos esquemas, que son ordenaciones dinámicas, se necesita como entrada una lista con las columnas que para cada fila tienen un valor distinto de cero.

Las ventajas comparativas de cada uno de ellos están influenciadas por la topología de la red, su tamaño, y el número de soluciones buscadas. La única virtud del esquema 1 es su simplicidad y velocidad. Para la matriz de admitancias de nudos, el esquema 2 es lo suficientemente mejor que el esquema 1 como para justificar el tiempo requerido para su ejecución. No se puede decir lo mismo del esquema 3 respecto al 2, por lo menos para redes eléctricas.

Avanzando más pasos en el esquema 3 nos acercamos al óptimo. Es decir, si en un determinado momento, existe empate entre dos filas en lo que respecta al número de elementos nuevos que se introducirían si se eliminara esa fila, se puede indagar uno o más pasos hacia adelante para averiguar qué fila es la mejor. Estudiando los pasos suficientes en cada etapa ordenaremos el grafo óptimamente.

Stott [100] ha hecho una excelente comparación de estos métodos, en lo que respecta a tiempos de ordenación, número de elementos nuevos aparecidos, operaciones aritméticas necesarias para la solución del sistema de ecuaciones correspondiente, y tiempo invertido en dicha solución. La conclusión es que el esquema 2 es el mejor para los sistemas ensayados, y en ésto coinciden casi todos los conocedores del tema.

En [75] se indica hasta dónde es óptimo cada uno de los ordenamientos. El esquema 1 es óptimo para grafos en estrella y para cualquiera que tenga menos de cinco nudos. A partir de aquí no se puede garantizar nada, dependiendo de la configuración. El esquema 2 garantiza la optimalidad para cualquier grafo en árbol, grafos que no contengan más de un bucle, y cualquier grafo con menos de seis nudos u ocho líneas. El esquema 3 tiene las mismas garantías que el 2, pero la última de ellas se eleva a nueve nudos o diez líneas. Esta cifra asciende a cualquier grafo con once nudos si avanzamos dos pasos en el esquema 3.

Para finalizar con este grupo básico de ordenamientos en redes eléctricas, se dirá que el esquema 2 se conoce en otros ámbitos con el nombre de "mínimo grado", y tambien como criterio de Markowitz simplificado. El criterio de Markowitz [70] es también muy popular pero para matrices no simétricas, donde el pivote no se elige necesariamente de la diagonal. Propone elegir como pivote en cada etapa, el elemento que minimice el producto del número de elementos no nulos de su fila, por el número de elementos no nulos de su columna (sin contar él mismo en cada caso).

Para la siguiente matriz, el elemento (3,4) no tiene otros elementos en su columna, y por tanto el mencionado producto vale cero (como indica la matriz central):

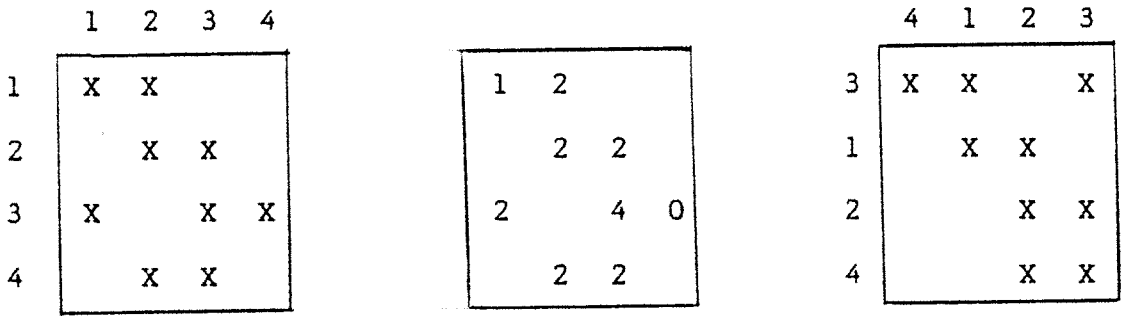


fig.4.18

Utilizando el criterio de Markowitz la matriz quedaría ordenada como la del extremo derecho, observándose que no aparece ningún elemento nuevo en la factorización (con la ordenación natural aparecerían 2, en las posiciones (3,2) y (4,4)).

El producto utilizado por este criterio, es precisamente el número de operaciones (sumas + multiplicaciones), y la máxima cantidad de elementos nuevos que se puedan crear al utilizar el elemento en cuestión como pivote. El comprender el significado de ese producto, que se utiliza como factor de mérito, explica porqué este criterio (y su versión para matrices simétricas que hemos llamado esquema 2) da tan buenos resultados y está tan generalizado.

Otro tipo diferente, pero igualmente importante, de ordenaciones lo constituyen las ordenaciones en banda. Todas ellas tienen en común el que disponen los elementos no nulos agrupados en una banda más o menos ancha alrededor de la diagonal principal. En función del método de almacenamiento que se adopte podemos utilizar dos tipos diferentes de algoritmos: Los que minimizan la anchura de banda de la matriz y los que minimizan el perfil (o

envoltura, que viene relacionada con el perfil segun (4.9)). Existe una fuerte correlación entre ambas ideas, pues una matriz con una banda estrecha, no puede tener un perfil excesivamente grande, y viceversa, pero ello no quiere decir que ambos mínimos se obtengan con la misma ordenación.

Como la anchura de cada fila no será exactamente igual a la anchura de la matriz, sino que será menor, conviene utilizar un almacenamiento del tipo de anchura variable, para no tener que almacenar los elementos nulos. En ese caso es preferible realizar una ordenación que minimice el perfil. Si se pretende por el contrario simplificar la programación, y se utiliza un almacenamiento del tipo de banda constante, interesa que la anchura de la matriz sea mínima, para intentar almacenar el menor número posible de elementos nulos. Estas consideraciones se ilustrarán con el siguiente ejemplo:

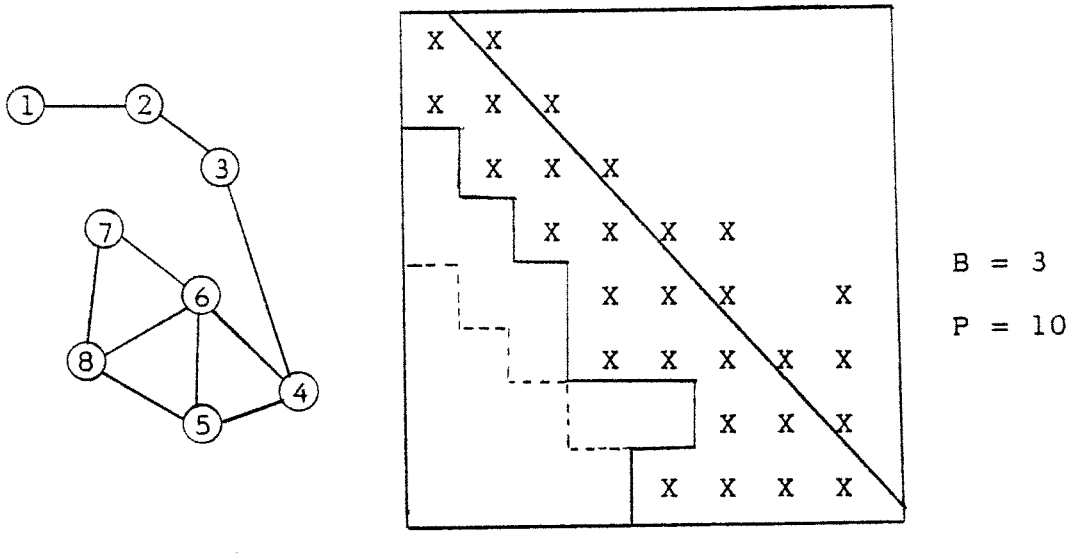


fig. 4.19

Con la ordenación anterior, un almacenamiento de banda constante sería antieconómico, pués almacenaria 8 elementos nulos (por cada semi-matriz). Sería conveniente un almacenamiento de banda variable. Si el mismo grafo lo ordenamos de esta otra forma:

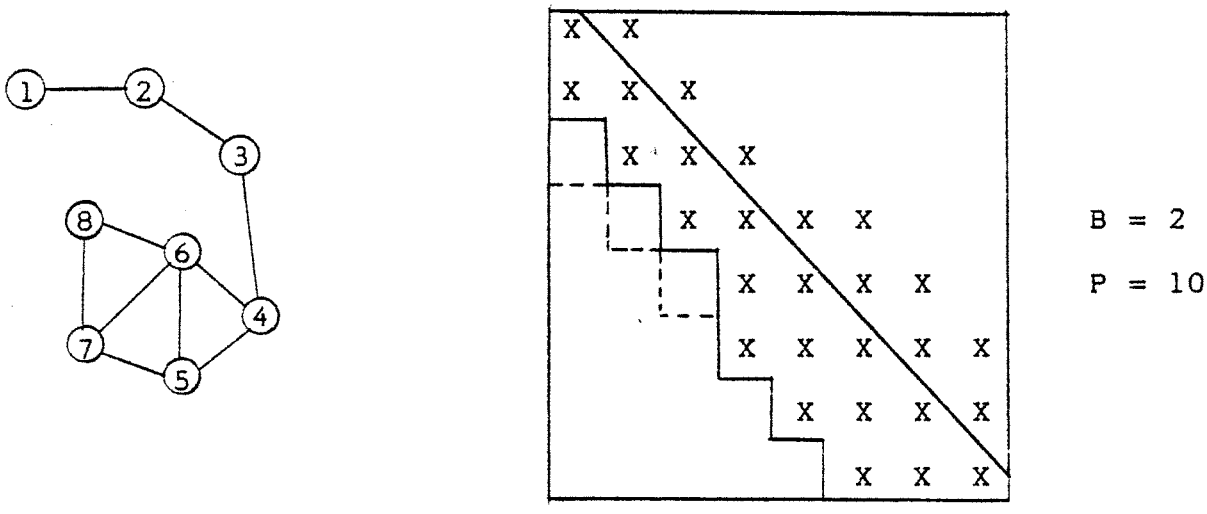


fig. 4.20

resulta que la anchura de banda (B) ha disminuido de 3 a 2, y un almacenamiento de banda constante es interesante pues sólo almacenamos 3 elementos extras (y además no se necesita ningún conjunto de punteros). El perfil es mínimo en los dos casos, pero la anchura de banda sólo es mínima en el segundo, aunque entre uno y otro ordenamiento sólo han cambiado los dos últimos nudos.

De nuevo se muestra patente la necesidad de una adecuada ordenación.

Entre los algoritmos que minimizan la anchura de banda, el más popular es sin duda el de Cuthill-McKee [23]. El algoritmo de CM es como sigue:

a) Para cada nudo con grado suficientemente bajo, generar su estructura de nivel radical y calcular su anchura.

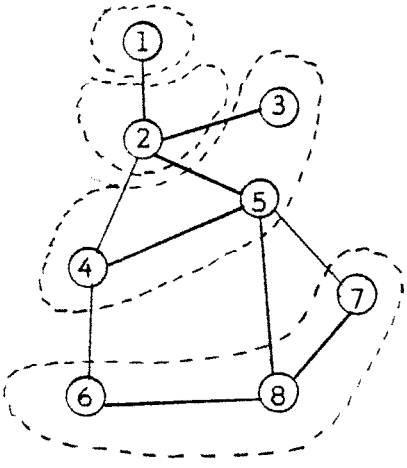
b) Para cada estructura anterior que tenga anchura mínima, ordenar los nudos, nivel por nivel, por el procedimiento siguiente:

   b1) La raiz es el número 1.
   b2) Para los siguientes niveles, numerar primero los vértices adyacentes al nudo con el número más bajo del nivel anterior, en orden de grado creciente. A continuación los adyacentes al vértice siguiente del nivel anterior, y así sucesivamente.

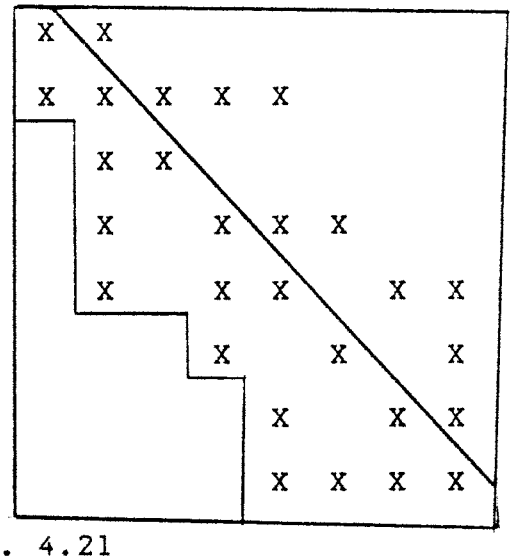
c) Calcular la anchura de banda de cada ordenamiento anterior y quedarse con aquel que la de menor.

A menudo en la literatura aparece el algoritmo sólo como el paso b), dándose por descontado que ya se conoce la raiz que da una anchura mínima. Con lógica se elige como raiz un nudo con la máxima excentricidad posible (periférico), que dará una estructura de mayor profundidad, y por tanto (puesto que el número de elementos totales es constante) la anchura media para cada nivel será menor. Encontrar un nudo periférico no es trivial, habiéndose trabajado mucho al respecto, y se opta entonces por elegir un nudo con las mínimas interconexiones posibles, lo que en muchos casos es satisfactorio aunque no resulte una banda mínima (ésto se compensa con el ahorro de tiempo en la ordenación).

El grafo de la figura 4.20 está ordenado según este algoritmo. Asimismo lo está el de la figura siguiente:



A_c=



B = 3

P = 14

$|Llen(A_c)|$=8

fig. 4.21

En la matriz se ha señalado la envoltura. Un grafo ordenado por este procedimiento cumple ciertas propiedades que se demuestran en [67] . Entre ellas están las siguientes:

a) No existen filas reentrantes, ésto es, si i<j entonces $f_i \leq f_j$ (se dice también que la envoltura es monótona). Esta propiedad es muy interesante, pues la matriz (considerando sólo la mitad inferior) aparece como una serie de triángulos superpuestos, los cuales no se necesitan en su totalidad en la eliminación gaussiana. Para matrices muy grandes sólo es preciso tener en memoria una parte de ellas en un momento dado.

b) La envoltura de la matriz correspondiente ya triangularizada es llena, es decir, todos los elementos que inicialmente

eran 0 dejan de serlo. Esto implica que para aquellos grafos cuya banda resultante sea muy ancha, como ocurre por regla general en las redes eléctricas, el número final de elementos puede llegar a ser muy alto.

En su tesis doctoral, George [37] hizo notar que simplemente invirtiendo el orden de los nudos, se conseguía una mayor eficiencia. En concreto, el perfil o envoltura era menor en muchos casos, aunque por supuesto la banda era la misma. Se denominó a este algoritmo el método de Cuthill-McKee invertido, y es muy utilizado industrialmente en combinación con el esquema de almacenamiento de Jennings. Para el ejemplo anterior el resultado sería:

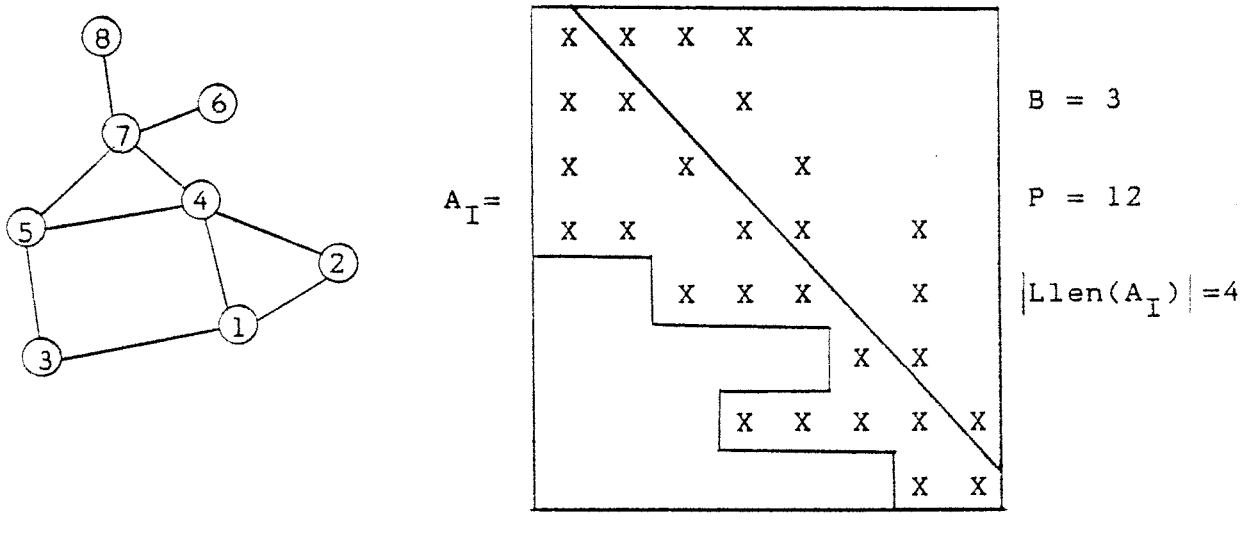


fig. 4.22

El llenado final es exactamente la mitad, necesitándose por tanto menos operaciones aritméticas y menos capacidad de memoria.

Liu y Sherman [67] demostraron que esta ordenación es siempre al menos tan buena como la original en lo que respecta a los factores mencionados anteriormente. La envoltura de una matriz con esta ordenación es la envoltura traspuesta de la matriz ordenada con el algoritmo primitivo (recuérdense las definiciones), y lo que se demostró fue precisamente que:

$$\begin{aligned} &Env\ (A_I) = Tenv\ (A_c) \subseteq Env\ (A_c) \\ &\left|Env\right|(A_I) \leqslant \left|Env\right|(A_c) \end{aligned} \qquad (4.12)$$

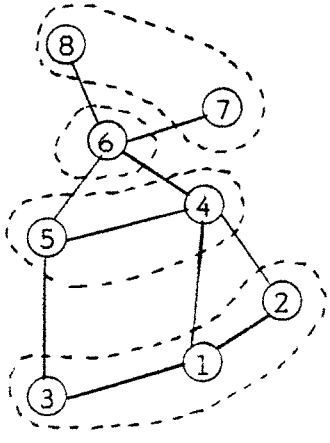
y como corolario:

$$\left|Llen\right|(A_I) \leqslant \left|Llen\ (A_c)\right| \qquad (4.13)$$

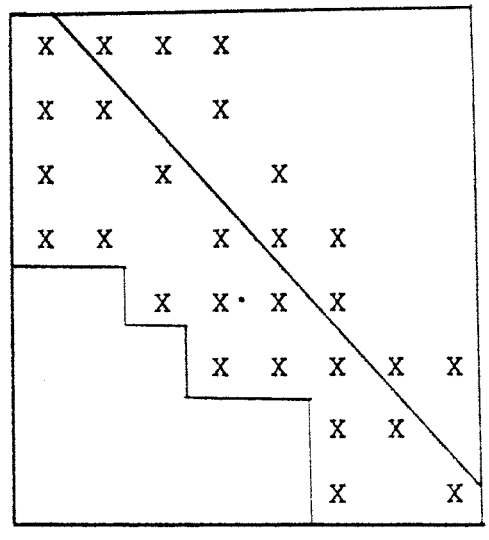
Con esta ordenación no se cumple la propiedad a) que se comentó anteriormente, pero ésto no resulta un gran inconveniente si se utiliza un almacenamiento de banda variable.

Un algoritmo alternativo al de Cuthill-McKee fue propuesto por Gibbs et al. [44], con vistas a reducir tanto la anchura de banda como el perfil. Es el resultado de la combinación de tres algoritmos, que intentan obviar los inconvenientes del de Cuthill-McKee. Primeramente se buscan los extremos de un pseudodiámetro. Se combinan las dos estructuras de nivel que se pueden generar desde cada extremo, en una sola, que no tiene porqué ser radical, buscando minimizar su anchura (no confundir con anchura de banda). Se ordenan los nudos dentro de cada nivel por un procedimiento semejante al de Cuthill-McKee.

El resultado es menos intuitivo y simple, pero parece más rápido, y da por regla general envolturas y bandas más pequeñas (aunque los ejemplos mostrados provienen todos de elementos finitos). Sería preferible utilizar una versión simplificada del algoritmo de Cuthill-McKee que es mucho más facil de programar. El ejemplo que hemos venido manejando quedaría de la siguiente forma:



A_G =



B = 3

P = 13

$|Len(A_G)|$ = 6

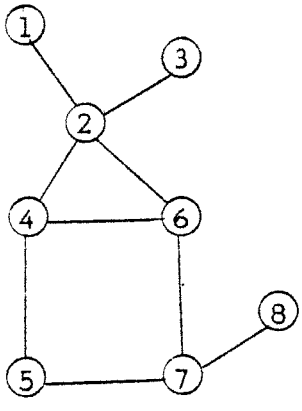
fig. 4.23

En este caso concreto el resultado es intermedio entre el de C-M y su inverso, en lo que al perfil se refiere, aunque la banda es la misma (en la izquierda se han señalado los diferentes niveles).

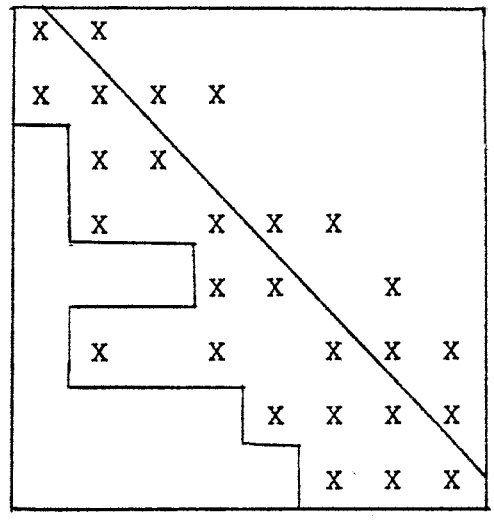
King [60] propuso un algoritmo específicamente para reducir la envoltura. Se numera primero un vértice de grado mínimo (como se puede hacer en una versión simplificada del de C-M). El grafo queda dividido en tres partes: Los vértices ya numerados (parte 1), los vértices adyacentes a los de la parte 1 (parte 2), y el

resto (parte 3). En cada etapa se numera el vértice de la parte 2 que hace que menos vértices de la parte 3 pasen a ser de la parte 2.

Nuestro ejemplo se ordenaría así:



A_k =



B = 4

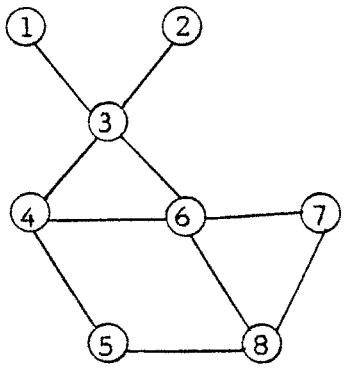
P = 13

$|Llen(A_k)|$ =6

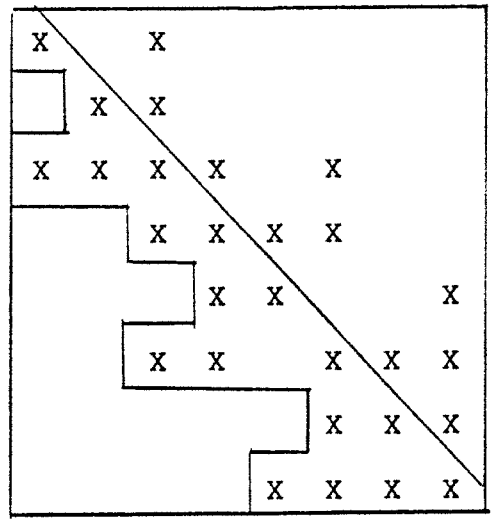
fig. 4.24

Se observa que no se minimiza la banda, aunque el perfil es menor que para el de C-M.

Levy [65] propone no restringir la búsqueda en la elección del siguiente nudo a los de la parte 2, produciéndose a veces un perfil menor, aunque el algoritmo es más costoso de llevar a cabo. Casualmente este algoritmo produce una ordenación óptima en nuestro ejemplo:



A_1 =



B = 3

P = 11

$|Llen(A_1)| = 2$

fig. 4.25

George-Liu [42] propusieron un algoritmo que en cierto modo sintetiza los de CM y Gibbs et al. Este algoritmo, que llamaron "arbol cociente refinado", está a medio camino entre los 3 esquemas básicos que se describieron al comienzo, cuya finalidad es minimizar el llenado de la matriz y las operaciones aritméticas, para lo que se precisa un almacenamiento complejo, y los esquemas de minimización de banda y/o perfil, que no necesitan una estructura compleja para almacenar la matriz, pero a cambio no explotan los ceros interiores a esa estructura, por lo que no resultan muy eficientes cuando la forma del grafo es irregular, como suele ocurrir con una red eléctrica.

El método del árbol cociente refinado sigue manteniendo una estructura de almacenamiento relativamente simple, pero además supera a los ordenamientos en banda o perfil para problemas con forma más irregular.

El algoritmo, que no parece oportuno detallarlo aquí, genera una estructura de nivel a partir de un nudo pseudoperiférico, como hacen Gibbs et al. En las definiciones vimos que esta estructura era un árbol cociente, pero que no tiene porqué ser maximal. Estos autores proponen una forma de refinar el árbol cociente hasta hacerlo maximal.

Las particiones de ese árbol maximal se ordenan monótonamente, y los nudos dentro de cada partición se ordenan por un procedimiento generalizado del de CM invertido. A continuación se muestra la estructura de nivel creada desde un nudo periférico, su árbol cociente refinado, y la ordenación de los nudos, para el grafo de la figura 4.10.

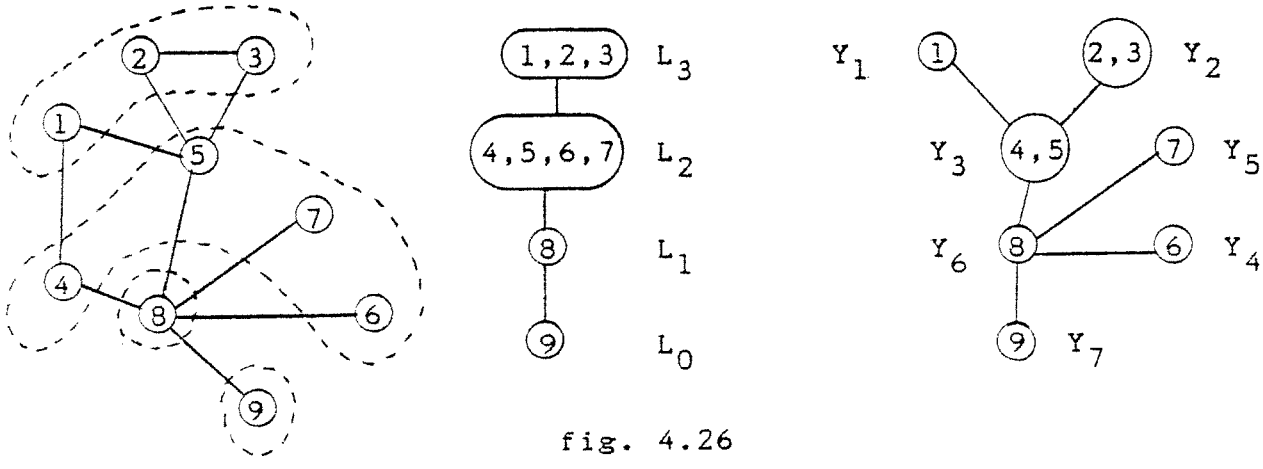


fig. 4.26

Este algoritmo es inseparable de la forma en que los autores almacenan la matriz y resuelven el sistema de ecuaciones. La partición $P=\{Y_1, Y_2 \ldots Y_p\}$ genera los bloques correspondientes en la matriz. Los bloques diagonales se almacenan siguiendo el esquema de Jennings, y el resto de los bloques por un almacenamiento compacto bastante convencional. La solución del problema, utilizando

la triangularización de Cholesky, tambien se hace por bloques, para lo cual resulta muy ventajoso el hecho de que las particiones constituyan un árbol cociente monótonamente ordenado. Esta idea constituye una generalización, para un árbol cociente, del hecho demostrable [42] de que en un árbol monótonamente ordenado el llenado de su matriz es cero (por tanto no aparecerán bloques nuevos).

George y Liu presentaron además otro algoritmo [43], que intentaba ahorrar memoria mediante una partición fina de la matriz, y almacenando las particiones resultantes en forma diferente según que fuesen o no un bloque de la diagonal.

Al igual que antes ocurría con un árbol monótonamente ordenado, se intenta ahora explotar otra idea simple. La idea se condensa en el siguiente lema:

Sea $j>i$. El elemento a_{ij} pertenece a la matriz A ya triangularizada, si y sólo si el nudo X_j de su grafo es alcanzable a través del conjunto $\{X_1, X_2 \ldots X_{i-1}\}$.

Expresado en un lenguaje menos formal: Los arcos del grafo lleno (el que representa a la matriz triangularizada), serán de la forma (X_i, X_j), $j>i$ en donde X_j es adyacente a X_i (arco de la matriz primitiva), o es alcanzable a través de un camino formado por nudos numerados (eliminados) antes que X_i.

Supongamos que hemos descompuesto un grafo G en los tres subgrafos siguientes:

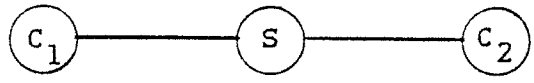


fig. 4.27

El conjunto S se denomina separador, y se dice que se ha hecho una disección en G. Por el lema anterior, no existirán en el grafo resultante de la triangularización, arcos de la forma (X_i,X_j) con $X_i \in C_1$ y $X_j \in C_2$, si numeramos S al final.

Esta idea se puede extender recursivamente para definir lo que se llama una disección anidada:

Se hace una primera división en el grafo. Para cada una de las componentes conexas resultantes se realiza otra división, y así sucesivamente. Una ordenación anidada es aquella que en un determinado nivel de profundidad numera los nudos de los separadores los últimos.

Los autores proponen un algoritmo automático de disección y ordenación anidadas, que combinado con un almacenamiento muy similar al del árbol cociente refinado, resulta bastante eficaz en lo que respecta al ahorro de memoria, aunque computacionalmente es costoso.

La pauta que se sigue en la disección, es la de obtener separadores pequeños que desconecten el grafo en dos o más componentes de aproximadamente igual tamaño.

El algoritmo fue en principio presentado para estructuras rectangulares de elementos finitos [39] donde es más eficiente, y

posteriormente se generalizó a estructuras irregulares [43].

A continuación se muestra un grafo tal y como quedaría ordenado por este procedimiento, señalándose los separadores:

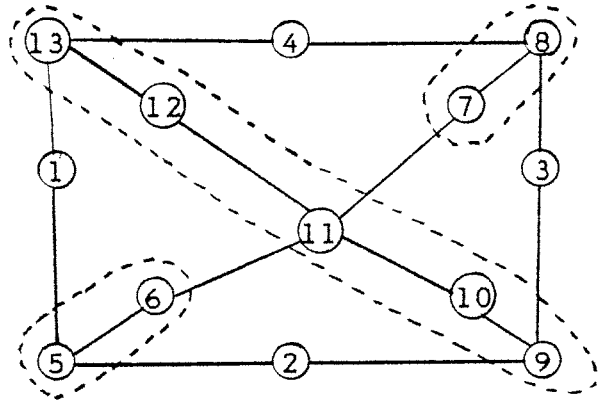
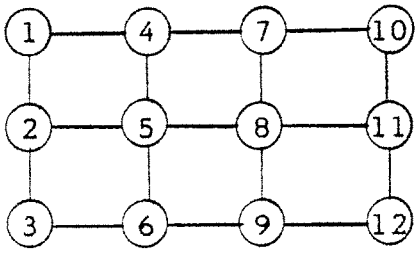


fig. 4.28

George presentó otro algoritmo más simple en la misma línea que éste, tanto en su versión para estructuras regulares [38] como irregulares [41]. Se puede denominar disección en un solo sentido ("one-way dissection"), y aunque menos eficiente que el anterior, resulta a veces competitivo por su rapidez.

Por último, para estructuras muy regulares, existe una ordenación eficiente que se puede realizar manualmente, llamada ordenación frontal, y que consiste en ir barriendo secuencialmente el grafo por su frente menor, como en el siguiente ejemplo:



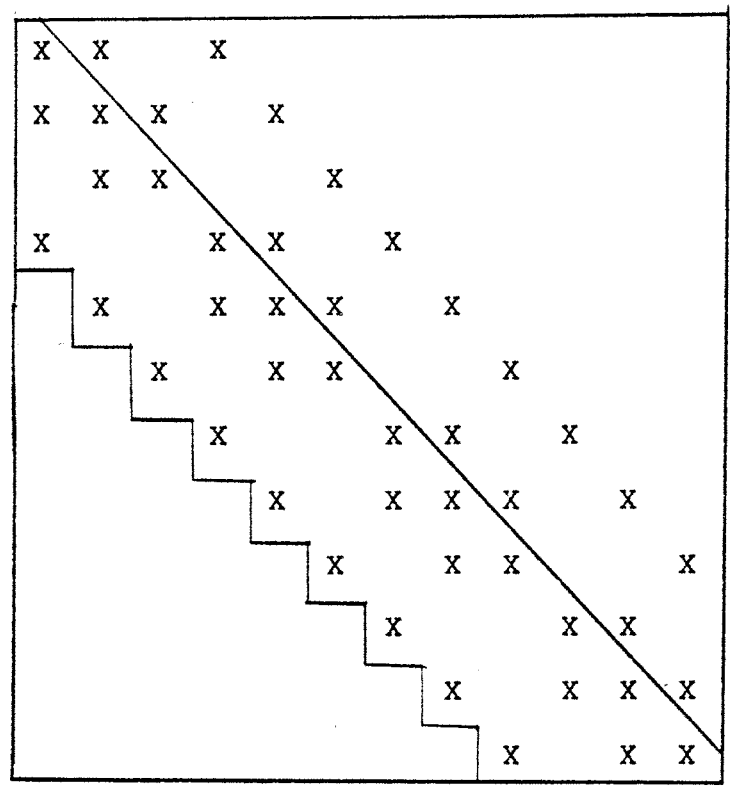


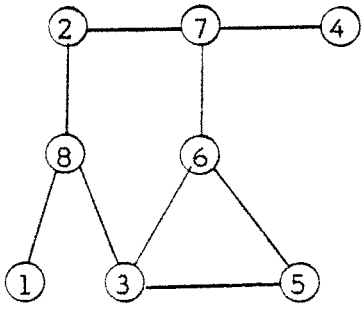
fig. 4.29

Todos los esquemas en banda o análogos mencionados con anterioridad, tienen su origen en la resolución de problemas estructurales por elementos finitos (aviones, barcos, etc.) en los que existe cierta regularidad en los grafos. Los problemas que aparecen en los sistemas eléctricos de potencia, se caracterizan precisamente por la ausencia de una estructura concreta, apareciendo zonas radiales y zonas malladas, dependiendo en cierto grado del nivel de tensión en que nos movamos. Por ello no tiene mucho interés el seguir un algoritmo en forma exacta, dado que ninguno de ellos (salvo quizás el del árbol cociente refinado) responde todo lo satisfactoriamente que sería de desear.

Se han propuesto entonces en la literatura [100] ordenaciones en banda que son simplificaciones de las anteriores. Básica-

mente estas ordenaciones son dos: La primera intenta agrupar los elementos de la matriz alrededor de la diagonal principal, y es una versión simple del algoritmo de CM. Partiendo de un nudo de mínimo grado, se van ordenando consecutivamente los nudos adyacentes a los ya ordenados, sin seguir un criterio especial. Como ejemplo puede servir la figura 4.19.

La segunda es en cierto modo novedosa. Intenta agrupar los elementos no nulos alrededor de la diagonal secundaria (arriba derecha- abajo izquierda). Comienza con un nudo de grado mínimo. Sus adyacentes se numeran los últimos. El siguiente nivel se numera nuevamente al principio, y así sucesivamente, como en este ejemplo:



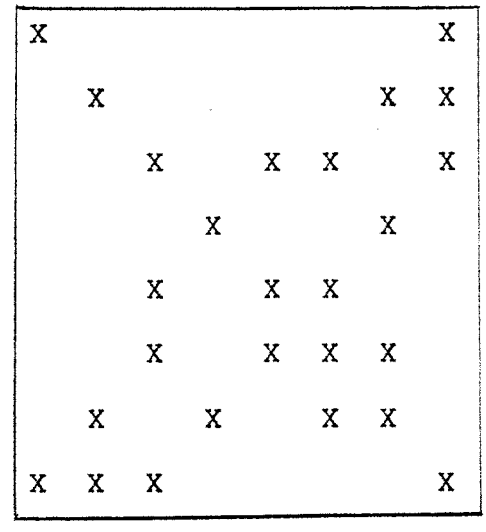


fig. 4.30

Ambos algoritmos son duales en su efecto sobre los grafos. El primero, como hemos visto en varios ejemplos, funciona bien para grafos encadenados y mal para grafos radiales (o en estre-

11a). Lo contrario sucede con el segundo.

Bajo ciertas condiciones, puede interesar no seguir completamente un algoritmo determinado, y numerar al final ciertas filas de la matriz, aunque afecten perjudicialmente al llenado de la misma. Entre esas condiciones podemos señalar:

1) Se sabe que los cambios en la matriz original ocurrirán sólo en determinadas filas. Numerando estas filas al final, sólo habrá que recalcular esa porción de la tabla de factores cuando ocurra un cambio.
2) Se sabe que los cambios en el vector independiente ocurrirán en unos determinados elementos. Si las filas correspondientes se numeran al final, sólo habrá que realizar una mínima parte de la eliminación hacia adelante al producirse un cambio.
3) La matriz es asimétrica sólo en ciertas filas. Se ahorra almacenamiento aprovechando la simetría de las otras filas si éstas se dejan para el final.

Duff et al. [27] compararon el funcionamiento del algoritmo de disección anidada, con el del esquema 2 de Tinney, para problemas de elementos finitos. El resultado fue desfavorable para el último, tanto en lo que respecta a número de operaciones como en el llenado de la matriz. Teóricamente es difícil predecir el comportamiento del esquema 2, pues depende de la forma en que se resuelvan los empates, que son muchos, al optar entre un nudo u otro. Los únicos datos los aportan los ejemplos.

Las conclusiones de Duff et al. no deben engañarnos. Con posterioridad, Lewis y Poole [66], presentaron sus resultados de aplicar cinco formas de ordenación a diversos problemas de sistemas eléctricos, uno de los cuales es un estándard del IEEE (118 nudos). Los algoritmos utilizados fueron

- Esquema 2 de Tinney (E2T).
- Cuthill-McKee invertido (CMI).
- Disección anidada (DA).
- Arbol cociente refinado (ACR).
- Disección en un solo sentido (DUSS).

Y las conclusiones de los autores:

1) La superioridad de E2T sobre los demás algoritmos es abrumadora. Uno de los ejemplos era tan grande (5300 barras), que sólo pudo ser resuelto por este algoritmo. En otras pruebas publicadas sobre elementos finitos, no estaba claro cuál era el peor y el mejor.
2) CMI no es eficiente para problemas eléctricos (resultan bandas muy anchas).
3) ACR es excesivamente lento, a causa del gran número de bloques que genera. Si el nivel de descomposición no fuese tan fino, sus resultados mejorarían.
4) De los tres últimos procedimientos (que producen pequeños bloques en la matriz) el más prometedor para futuros estudios es el de DA. La necesidad de memoria y el número de operaciones en este algoritmo depende grandemente del tamaño de los primeros separadores. Idealmente, el

primer separador debería ser muy pequeño, y dividir la matriz en dos partes casi iguales. En la práctica, el algoritmo toma el separador en la mitad, independientemente de su tamaño. Para redes algo irregulares, esta estrategia no es óptima.

4.5 PARTICIONES EN MATRICES VACIAS.

Un tipo particular de ordenamientos, intenta disponer los nudos de forma que se puedan identificar en la estructura de la matriz bloques grandes, con pocas interconexiones entre ellos. Estos ordenamientos tendrán más o menos éxito, según que la red original en la que se basa la matriz se componga realmente de subredes con poca conexión. A veces las subredes son tan evidentes a simple vista, que los datos de entrada ya se pueden introducir agrupados convenientemente. Otras veces no existen en absoluto agrupamientos ("clusters") naturales, y a pesar de ello puede convenir utilizar alguna de estas ordenaciones, por los motivos que se verán más adelante. La siguiente figura es un ejemplo de partición en matrices:

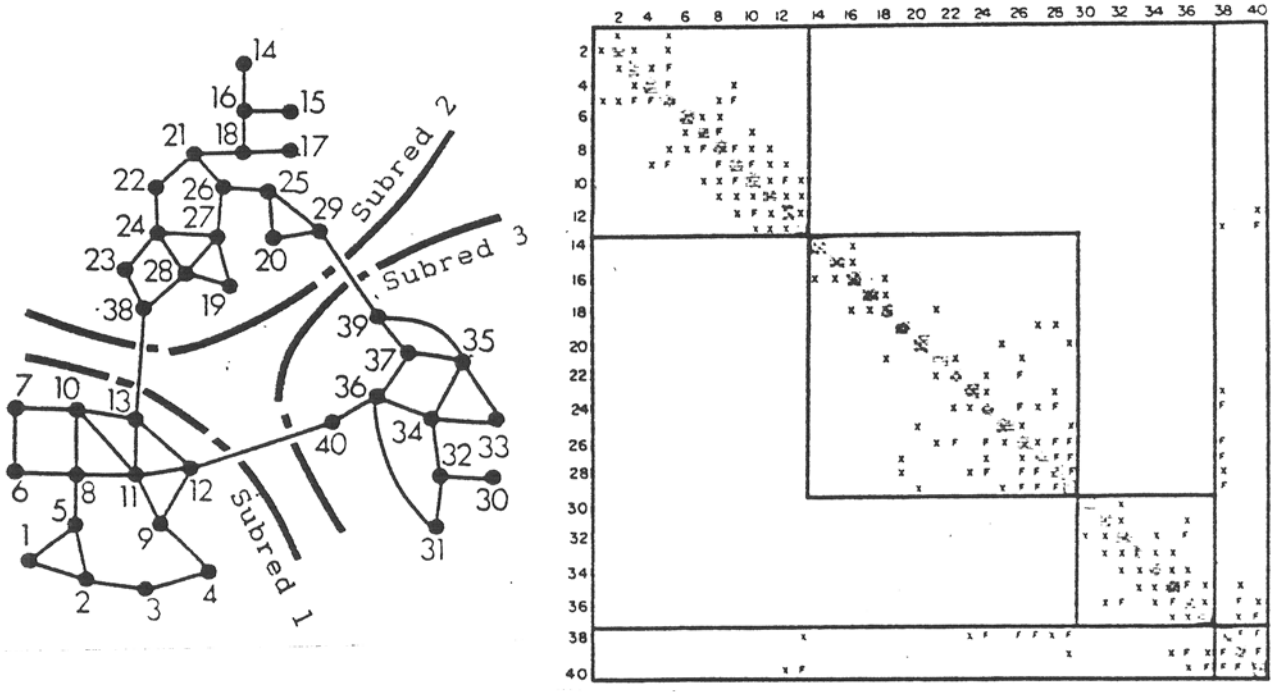


fig. 4.31

La literatura acerca de la solución de ecuaciones lineales por particiones es abundantísima, pero en la mayoría de las publicaciones el problema que se aborda es el de la solución numérica de un sistema ya particionado, dándose por sentado que se han identificado los subgrupos en la matriz, bien manualmente o por algún procedimiento autómatico.

El problema teórico de la partición óptima de cualquier matriz, para minimizar una cierta función de coste, es formidable, y no existe una solución satisfactoria para ello, siendo todavía una materia abierta al estudio. Existen sin embargo algunas aportaciones heurísticas que, dependiendo de la naturaleza del problema, intentan minimizar el número de nudos de interconexión, el número de ramas de interconexión, el coste total de las ramas de interconexión (el coste puede ser por ejemplo la admitancia), o la "distancia" entre "centroides" de cada grupo, etc.

Los campos de aplicación son bastantes diversos: descomposición de redes eléctricas y circuitos electrónicos, "lay-out" de circuitos impresos e integrados, agrupamientos estadísticos de datos, investigación operativa, cálculos de estructuras, etc, aunque en el último mencionado suele ser fácil una identificación visual de los grupos.

Como es lógico sólo nos interesará el primer campo de aplicación, y el criterio en concreto que se seguirá será el de minimizar el número de nudos de interconexión.

El introducir particiones en las matrices ofrece las siguientes ventajas:

1) Un cambio en una submatriz sólo afecta a dicha matriz y a la interconexión, pero no al resto de submatrices (de especial interés al recalcular la tabla de factores).
2) Si trás una solución del problema, se hace un cambio en el vector independiente en las filas correspondientes a una partición, sólo habrá que repetir las operaciones que involucran a esa partición y a la interconexión.
3) Si la capacidad de memoria está limitada, se puede proceder con una submatriz en cada instante ("overlays").
4) Permite procesamiento en paralelo del problema, asignando un procesador a cada partición, lo cual entronca con las últimas tendencias.

Cuando se pretende aprovechar las ventajas 3) ó 4) se exige además al algoritmo que límite el número de nudos por cada partición, o que limite el número de particiones, respectivamente.

El primer algoritmo aparecido, realmente eficiente, es debido a Ogbuobiri, Tinney y Walker [75]. Dada la importancia de este tema, y para que se pueda captar la complejidad del procedimiento, se describirá con ciertos detalles.

Para el grafo en cuestión se construye una tabla de tres entradas, que se explicarán con el ejemplo siguiente:

INDICE	NUDO	CONJUNTO ADYACENTE
9	1(3)	2(4), 3(4), 7(4)
16	2(4)	3(4), 7(4), 4(4), 8(4)
16	3(4)	7(3), 4(2), 8(3), 5(3)
8	4(2)	7(1), 8(1), 5(1), 6(1)
3	7(1)	8(0), 5(0), 6(0)
0	8(0)	5(0), 6(0)
0	5(0)	6(0)
0	6(0)	

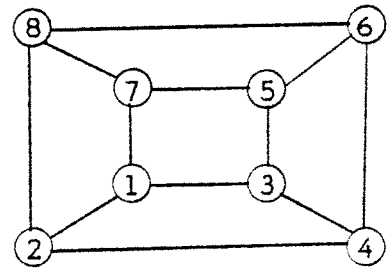


fig. 4.32

En la columna central aparecen secuencialmente todos los nudos del grafo, empezando por uno arbitrario. En la columna de la derecha aparece el conjunto de nudos adyacentes a los ya considerados. El número que aparece entre paréntesis en la columna central es la valencia del nudo. La valencia de un conjunto de nudos es el número de nuevos arcos que aparecen al eliminar del grafo esos nudos simultáneamente. Los números entre paréntesis de la columna de la derecha son las valencias respectivas de eliminar el nudo correspondiente junto al nudo de la columna central. El índice de la columna izquierda es el producto de la valencia del

nudo central por el número de elementos adyacentes que se encuentran a la derecha. Los nudos de la columna central se van eligiendo entre los nudos de menor valencia de la fila anterior. El primer nudo tomado arbitrariamente se llama semilla.

Para explicar cómo se pueden detectar grupos de nudos fuertemente interconectados se considerará el siguiente ejemplo, sacado de [75] y en el que los "clusters" resultan obvios a simple vista (por comodidad no se han escrito las valencias):

INDICE	NUDO	CONJUNTO ADYACENTE
2	1	2, 9,
9	9	2, 10, 14,
24	2	10, 14, 5, 7
72	7	10, 14, 5, 4, 6, 8
50	5	10, 14, 4, 6, 8
40	8	10, 14, 4, 6, 3
20	6	10, 14, 4, 3
9	3	10, 14, 4
2	4	10, 14
→ 12	14	10, 11, 13, 15
8	15	10, 11, 13, 12
3	12	10, 11, 13
0	13	10, 11
0	11	10
0	10	

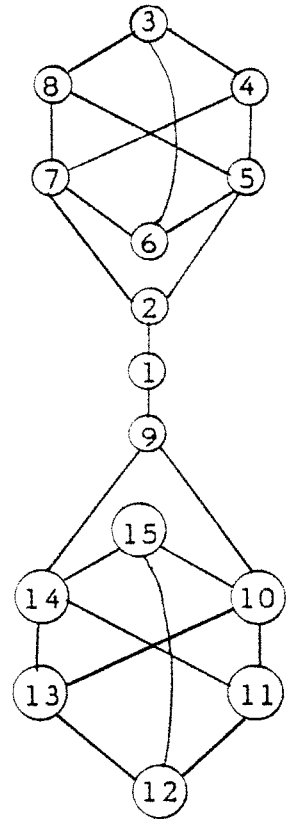


fig. 4.33

La tabla construida de esa manera cumple ciertas propiedades, como son:

1) Su extremo derecho (línea de puntos) tiene forma ondulada con al menos un pico.
2) A ambos lados de cada pico existe una subida y caida monótonas, que indican respectivamente la entrada y salida de un conjunto de nudos fuertemente interconectados.

   Si nos encontramos en el último pico se observa que:

3) La subida y bajada de este pico coincide asímismo con la última subida y bajada de la columna de índices (señalada con una flecha).
4) El grupo de nudos abarcado por este último pico forma un "cluster" (racimo). En este caso $\{14, 15, 12, 13, 11, 10\}$.
5) Los nudos que forman la interfase del cluster son los de la fila anterior a la última subida. En este caso $\{10,14\}$.

   Los demás son nudos interiores.

De haber comenzado la tabla con el nudo 3, el nudo 9 hubiera formado parte también del grupo. Este es el llamado "efecto semilla" que hay que intentar compensar para evitar obtener grupos parciales. Para ello se consideran como candidatos cada uno de los nudos adyacentes al grupo parcial. Si la inclusión de ese nudo, hace que el índice que se obtiene en la fila anterior al grupo, sea menor que el valor actual, el nudo forma parte del mismo. Eso ocurriría en el ejemplo anterior con el nudo 9.

Una vez detectado ese "cluster" completamente, se elimina del grafo y se procede nuevamente con el resto, hasta que no queda más que uno. En una misma pasada se pueden identificar varios "clusters".

Este procedimiento es muy lento, pues calcular la valencia de un conjunto de nudos es labor ardua. Los autores propusieron un método simplificado que parece funcionar bien en muchos casos. La simplificación consiste en utilizar como índice el número de nudos de la columna de la derecha, lo que evita calcular la valencia, pero entonces la elección entre los nudos adyacentes, de cuál será el siguiente nudo, es arbitraria, lo que puede dar resultados erróneos muchas veces.

Siete años más tarde, Sangiovanni-Vincentelli et al. [90], volvieron a investigar sobre este procedimiento simplificado, intentando corregir sus defectos sin perder velocidad.

Conviene hacer una observación gráfica para comprender la bondad del método:

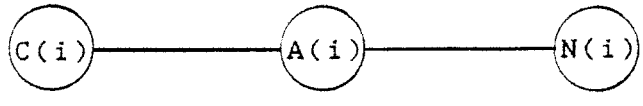


fig. 4.33

Para la etapa i, C(i) es el conjunto de nudos ya considerados, A(i) son los nudos adyacentes a C(i), y N(i) son el resto de nudos.

Obviamente C(i) y N(i) no son adyacentes. Si en algún momento A(i) es pequeño (cuello de botella), entonces C(i) y N(i) forman "clusters".

Tal como Ogbuobiri et al. plantearon el algoritmo simplificado, existían dos grados de libertad: Elegir el nudo inicial, y elegir el siguiente nudo en cada etapa.

Las mejoras propuestas consisten en lo siguiente:

a) El próximo nudo a elegir entre los de A(i) es aquel que da un valor menor para $|A(i+1)|$ (recuérdese que el algoritmo de King [60] para reducir la envoltura seguía exactamente la misma política).

b) El nudo de comienzo será un nudo de grado mínimo, lo que está de acuerdo con la lógica del paso anterior.

Además se le añaden ciertos refinamientos, que tengan en cuenta que cada "cluster" no sobrepase un número determinado de nudos, y que tampoco un "cluster" sea excesivamente pequeño. Algunos ejemplos mostrados, procedentes de circuitos electrónicos, son interesantes.

5. EXPERIENCIAS REALIZADAS.

Aparte de los algoritmos clásicos (Newton, desacoplado rápido, Gauss-Seidel) se han hecho numerosas pruebas que no aparecen en la literatura, para intentar lograr alguna mejora.

Gran parte de las mismas han resultado infructuosas (probablemente por éso no hayan sido nunca publicadas) comentándose aquí algunas para evitar que alguien las repita. Sin embargo, en una de las últimas, descrita en 5.6, las mejoras han sido significativas, a pesar de que su simplicidad parecía indicar lo contrario.

Todas las pruebas que se describen en este capítulo se han llevado a cabo en un PDP-11/34. Los programas se han escrito en FORTRAN, salvo la mayoría de las rutinas, especialmente en las últimas versiones, que se han realizado en ensamblador.

5.1 SISTEMAS UTILIZADOS COMO TEST.

Se ha procurado recopilar datos de la mayor cantidad posible de sistemas, de manera que presenten características diferentes entre sí.

Como sistemas estándard se han utilizado los de 14, 30, 57 y 118 nudos del IEEE, cuyos resultados son bien conocidos y permiten garantizar la ausencia de errores en el programa en concreto que se está probando. Los datos de estos sistemas se pueden en-

contrar en numerosas fuentes [35, 56, 78]. La principal característica de estos sistemas es su buen comportamiento ante los diversos métodos iterativos, por lo que son insuficientes para sacar conclusiones de los casos que nos podemos encontrar en la realidad.

El prototipo de simulador de redes para entrenamiento de operadores que, como se comentó en el capítulo introductorio, se había desarrollado en nuestro Departamento, se piensa ampliar hasta abarcar una red provincial de distribución, que es el campo de dominio de los Centros Provinciales de Maniobra que se están instalando en Andalucía. Este hecho, junto a las características propias de tales redes, fundamentalmente su alta relación R/X que empeora la convergencia de los algoritmos, como ya hemos visto, hace que nos hayamos interesado particularmente en tales redes.

Se han utilizado las de las provincias de Málaga y Sevilla, de 29 y 38 nudos respectivamente, siendo esta última la más importante en número de nudos de Andalucía. Estas redes son de 66 Kv, constituyendo las cargas en cada nudo la red de 20 Kv, e inyectándose potencia desde la red de 132 Kv (o a veces desde la de 220 Kv.).

Los datos de estas redes se encuentran en los Apéndices. En las figuras 6.11 y 6.13 se representan los grafos respectivos, que demuestran que las redes a este nivel de tensión están más malladas de lo que a veces se cree.

Se ha tomado de [112] un sistema de 13 nudos, como un caso típico de mal comportamiento ante los métodos numéricos, puesto que presenta dos ramas capacitivas. La figura 6.8 muestra el grafo de este sistema, que sólo tiene un bucle.

Recientemente hemos incorporado otros dos sistemas más. Uno es la red de la Saskatchewan Power Corporation (SPC), Canadá, aparecido primero en [87], y mencionado con posterioridad en distintas publicaciones. Tiene 26 nudos y 32 líneas, y los datos se refieren a una situación de fuerte carga.

El otro es la red de Nueva Inglaterra, de 39 nudos. Sus datos se encuentran en los Apéndices, y es un sistema utilizado usualmente en problemas de estabilidad transitoria.

5.2 MATRICES VACIAS.

Se han desarrollado expresamente para este trabajo rutinas de tratamiento de matrices vacías, ya que las librerías disponibles, de carácter general, sólo incorporaban matrices en banda como única opción. Estas rutinas permiten crear un elemento aislado de la matriz, crear elementos consecutivos de una fila, acceder a sus elementos de ambas formas (aisladamente o por filas), borrarlos, multiplicar una fila entera por un valor, factorizar la matriz, resolver el sistema de ecuaciones lineales, etc. Y todo ello tanto para matrices reales, complejas, como matrices que podríamos denominar estructurales, es decir, matrices en las que sólo almacenamos la fila y columna del elemento pero no su

valor, que no nos interesa (tal es el caso de las operaciones que se realizan para determinar el orden de los nudos).

Salvo para el método de Newton, programado inicialmente utilizando un almacenamiento compacto, siguiendo las ideas de Tinney y Hart [108], el resto de algoritmos implementan un almacenamiento de tipo aleatorio, más eficaz, mediante punteros al siguiente elemento de la fila.

Se puede seguir la explicación con la figura 5.1. Los dos primeros bytes apuntan a la primera posición disponible para ser utilizada. A continuación siguen N punteros que indican la dirección del primer elemento de cada fila. Después se almacenan en un orden aleatorio los elementos de la matriz. Por cada elemento se almacena su columna, el valor, y la dirección del que le sigue en la fila. El número de bytes (B) que ocupará el valor será 4, 8 ó 0 dependiendo del tipo de matriz (real, compleja o estructural). Un cero en un puntero indica que este elemento es el último de la fila (o que esa fila no tiene elementos).

El total de memoria necesaria para almacenar una matriz de N filas con M elementos será pués:

$$\text{Número de bytes} = 2N + 2 + M(4+B)$$

que depende de B (tipo de matriz).

Los punteros se pueden almacenar, si se desea, en forma relativa a la dirección base de la matriz. Ello tiene dos ventajas. Por una parte la matriz se puede mover de una zona de memoria a

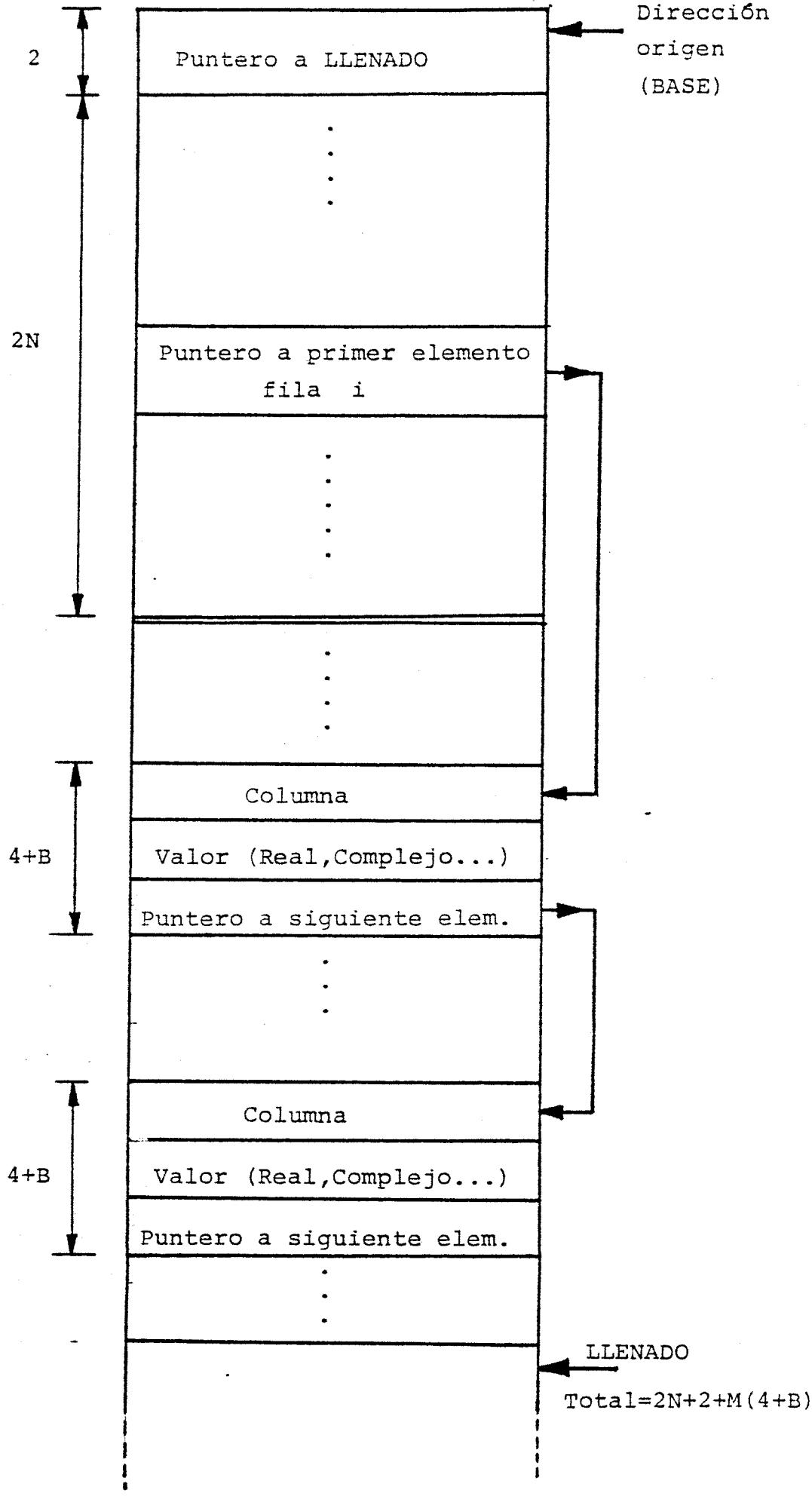

2
Puntero a LLENADO
Dirección origen (BASE)
2N
Puntero a primer elemento fila i
Columna
4+B
Valor (Real,Complejo...)
Puntero a siguiente elem.
Columna
4+B
Valor (Real,Complejo...)
Puntero a siguiente elem.
LLENADO
Total=2N+2+M(4+B)

Fig. 5.1

otra sin ningún cambio (es reubicable). Por otra ahorramos memoria si esta matriz está situada fuera de los primeros 64 K de memoria, donde una dirección absoluta no cabe en dos bytes.

Si el procesador utilizado tiene un direccionamiento eficiente, esta disposición no presenta ningún inconveniente.

Una vez que se dispone de estas rutinas, programar un algoritmo determinado es relativamente simple.

Como están realizadas en lenguaje ensamblador, las sentencias de FORTRAN se limitan prácticamente al manejo de ficheros (entradas de datos, salidas de resultados) y llamadas a las subrutinas apropiadas.

5.3. METODOS PRIMITIVOS.

Dado el poco esfuerzo que requería, se ha programado el algoritmo de Gauss-Seidel, tal como se describe en Elgerd [33]. Para nosotros su interés no radicaba en el propio algoritmo, sino en su posible combinación con otros, aunque en este épigrafe se darán los resultados de su utilización en forma aislada, que se recogen en la siguiente tabla:

NUDOS	14	29	30	38	57	118
F.Acel.	1.65	1.8	1.75	1.8	1.75	1.9
Iteraciones	22	67	34	48	70	115
tiempo (seg)	1.2	6	4.5	6	15.5	48

figura 5.2

Los resultados son muy parecidos a los dados en [35], en lo que respecta a los sistemas del IEEE. Las mayores diferencias están en los factores de acelereción que son un tanto altos (1.8 frente a 1.6). Pueden ser debidas al criterio de convergencia que hemos utilizado ($|\Delta P|$, $|\Delta Q|$ <.0001 p.u.) que no es usual en estos métodos.

Los tiempos no deben tomarse en forma literal, sino aproximada, ya que no se ha intentado optimizar estas versiones, de las que nos interesaba fundamentalmente el número de iteraciones.

Este número se ve que aumenta proporcionalmente a N. Como los cálculos por iteración también crecen aproximadamente con N, el esfuerzo total será proporcional a N^2, lo que resulta inadmisible para sistemas de cierta dimensión.

La red de distribución de 29 nudos requiere un número excesivo de iteraciones. La de 13 nudos no es resoluble por este procedimiento. Por último, la de 39 nudos satisface el criterio de convergencia para 162 iteraciones, pero si se sigue iterando vuelven a aumentar los valores ΔP, ΔQ por encima de la toleran-

cia (solución oscilante).

Pensando en mejorar esos resultados, se ha programado una versión análoga a la segunda variante de Bramaller y Denmead [14], descritas en 2.2.7, pero utilizando la matriz de admitancias en lugar de la de impedancias. La podemos denominar "residuo cero de tensión".

Manipulando la expresión

$$S_i = V_i \sum_{j=1}^{N} Y^*_{ij} \cdot V^*_j \qquad (5.1)$$

se llega a

$$A.\ V_i^2 + B\ .\ V_i + C = 0 \qquad (5.2)$$

con

$$A = Y^*_{ii} = G_{ii} - jB_{ii}$$

$$B = \sum_{\substack{j=1 \\ j\neq i}}^{N} Y^*_{ij}.V^*_j = a_i + jb_i \qquad (5.3)$$

$$C = - S_i = - P_i - jQ_i$$

De la expresión compleja (5.2), se puede obtener el módulo y la fase de V_i, en base a los mejores valores de las tensiones de los otros nudos.

Por este procedimiento, en principio más potente, se obtuvieron los siguientes resultados:

NUDOS	14	29	30	38	39	57	118
F.Aceler.	1.65	1.8	1.8	1.8	1.8	1.7	1.9
Iteraciones	22	70	41	48	87	69	110
Tiempo (seg)	1.88	8.5	5.46	7.88	14.5	17.52	57.42

figura 5.3

El número de iteraciones es sensiblemente el mismo que antes, lográndose no obstante la convergencia correcta para el sistema de 39 nudos en 87 iteraciones. Sin embargo el tiempo total es mayor, lo cual es lógico dado que las expresiones a evaluar en cada iteración son más complejas.

La conclusión es que ninguno de estos procedimientos, ni presumiblemente otros semejantes, resulta competitivo para sistemas grandes.

5.4. METODO DE NEWTON-RAPHSON.

Este procedimiento fué el primero que se programó en este trabajo. Por entonces no estaban disponibles las rutinas de matrices vacías mencionadas, que posteriormente se harían con carácter general. Por ese motivo, y para investigar la bondad de otros tipos de almacenamiento, se utilizó el que en el capítulo 4 denominamos compacto, donde todos los elementos de una misma fila están consecutivos, y en un vector se almacena la posición de comienzo de cada fila.

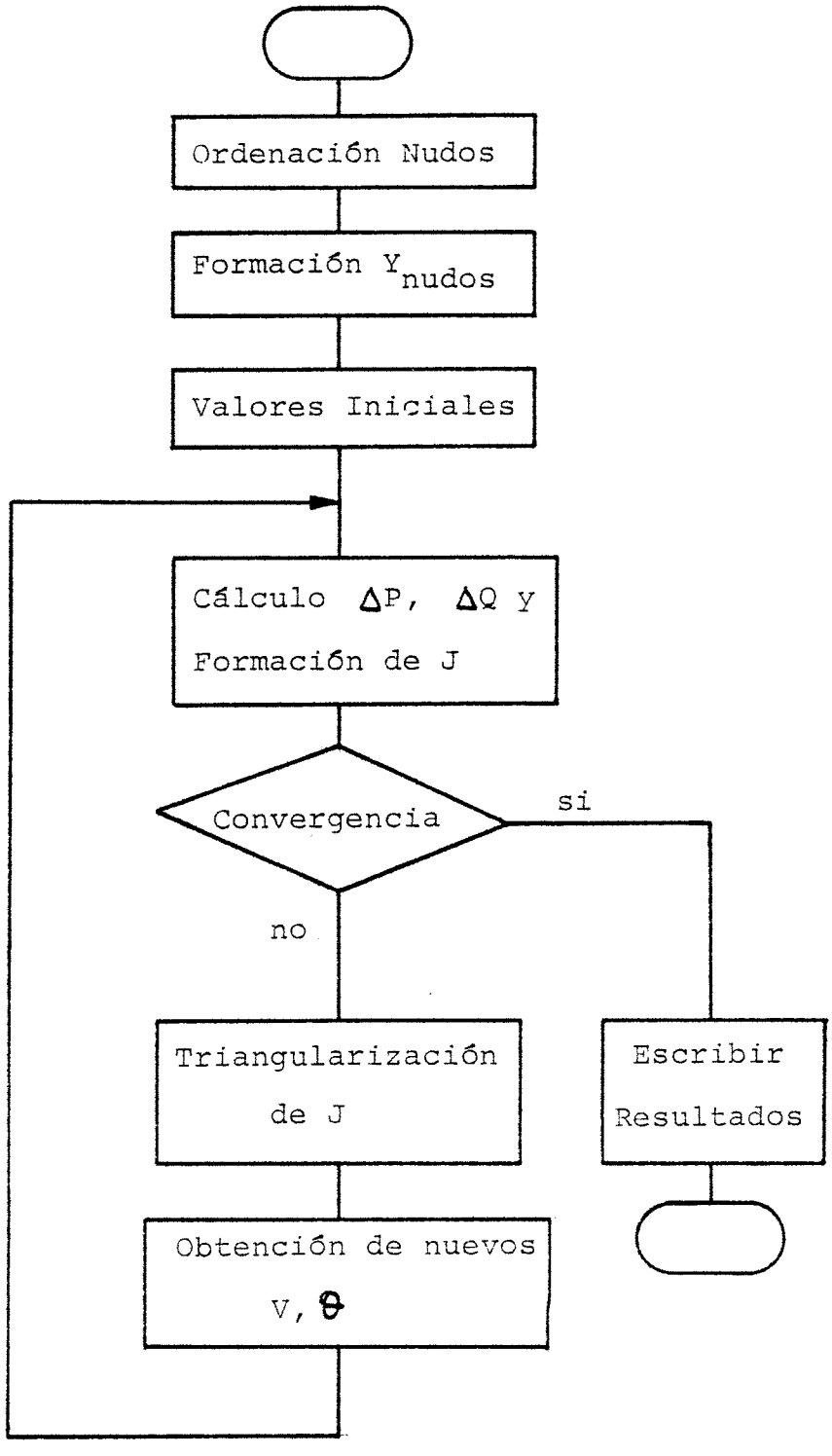

Ordenación Nudos
Formación Y_{nudos}
Valores Iniciales
Cálculo ΔP, ΔQ y
Formación de J
Convergencia
si
no
Triangularización
de J
Escribir
Resultados
Obtención de nuevos
V, θ

Fig. 5.4

La figura 5.4 muestra el diagrama de flujo utilizado, que no es exactamente el propuesto por Tinney y Hart [108], cuyo principal inconveniente era (véase 2.2.11) que la convergencia no se detectaba hasta el final de la presente iteración, con lo que se necesitaba una más de las necesarias realmente.

Los resultados de este método, aplicados a nuestros sistemas, se reflejan en la figura 5.5.

En dicha figura, la fila inmediatamente inferior al número de nudos, indica el número de elementos distintos de cero del jacobiano triangularizado, sin contar la diagonal. Esa cifra da una idea de la capacidad de memoria necesaria para almacenar las matrices.

Las restantes filas son tiempos medidos en segundos, excepto el número de iteraciones. Este número ha sido 3 siempre, salvo para el sistema de 13 nudos que requiere una más. El criterio de convergencia fue $|\Delta P|$, $|\Delta Q| < 0.0001$ p.u.

Conviene aclarar que el número de iteraciones mencionado corresponde a soluciones sin ajustar. Si se tuviesen en cuenta los límites de reactiva en los nudos PV, ese número podría aumentar bastante, así como el tiempo por iteración, al tener más elementos el jacobiano. Como ejemplo, la red de 30 nudos del IEEE requiere 5 iteraciones, y su jacobiano llega a tener 442 elementos (frente a 3 iteraciones y 426 elementos para la solución no ajustada). En este caso sólo un nudo sobrepasaba sus límites.

Newton-Raphson.

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Elementos no diagon.	68	152	212	260	426	334	538	1000	-
Ordenación	.1	.16	.26	.3	.36	.44	.54	1.48	-
Form. Mat.	.38	.64	1.12	1.18	1.28	1.56	1.58	2.68	-
Proce. Ite.	.34	.62	.78	1.0	2.04	1.32	2.56	6.70	-
Total	.98	1.62	2.44	2.84	4.18	3.84	5.36	12.30	-

Iteracion.	4	3	3	3	3	3	3	3	-
Tiempo / Iter.	.085	.206	.26	.333	.68	.44	.85	2.23	-

fig. 5.5

Los tiempos de ordenación de nudos y de formación de la matriz de admitancias, no son representativos de dichos procesos, pués incluyen la lectura de los datos del disco, que al realizarse con formato, y no directamente en binario, consumen mucho tiempo.

La red de 118 nudos no pudo resolverse con los 32 K de memoria que el sistema operativo limita para cada tarea (hubiera sido precisa la utilización de "overlays").

Los tiempos requeridos para el proceso iterativo, sin embargo, no están afectados por el acceso a disco, y sí son significativos.

La principal conclusión es que, a pesar de que el número de iteraciones es muy bajo y constante, no se puede utilizar esta técnica para trabajar en tiempo real, dado los tiempos tan altos resultantes (aunque no se haya intentado optimizar el programa). Sin embargo, dada su fiabilidad y buena convergencia, conviene reservarlo pata trabajos "off-line".

El método de almacenamiento compacto se desechó desde este momento, pués demostró ser poco eficiente.

5.5 DESACOPLADO RAPIDO.

Este método se programó incorporándole todas las técnicas posibles que lo hicieran más rápido y con menos ocupación de memoria. Entre ellas se pueden destacar:

- Almacenamiento de los datos en disco sin formato, y aumento del "buffer" de lectura para que sólo fuese preciso un acceso al mismo.
- Se tuvo en cuenta la simetría de las matrices, en todas las etapas del algoritmo, almacenándose en forma aleatoria.
- En lugar de utilizar las funciones trigonométricas de la librería, se optó por aproximarlas con los primeros términos del desarrollo en serie

$$\text{Sen } X = X - X^3/6$$

$$\text{Cos } X = 1 - X^2/2 + X^4/24$$

que producen en la práctica los mismos resultados, ya que el argumento (desfase entre nudos adyacentes) es relativamente pequeño. Así, con solo 5 multiplicaciones y 3 sumas (o restas) se obtienen simultáneamente las dos funciones.
- Todas las rutinas de tratamiento de matrices, comentadas en 5.2, así como las de cálculo de ΔP y ΔQ se realizaron en ensamblador.
- La triangularización de las matrices no se llevó a cabo mediante el algoritmo de Cholesky, como habitualmente aparece en la literatura, sino por otro que evita rea-

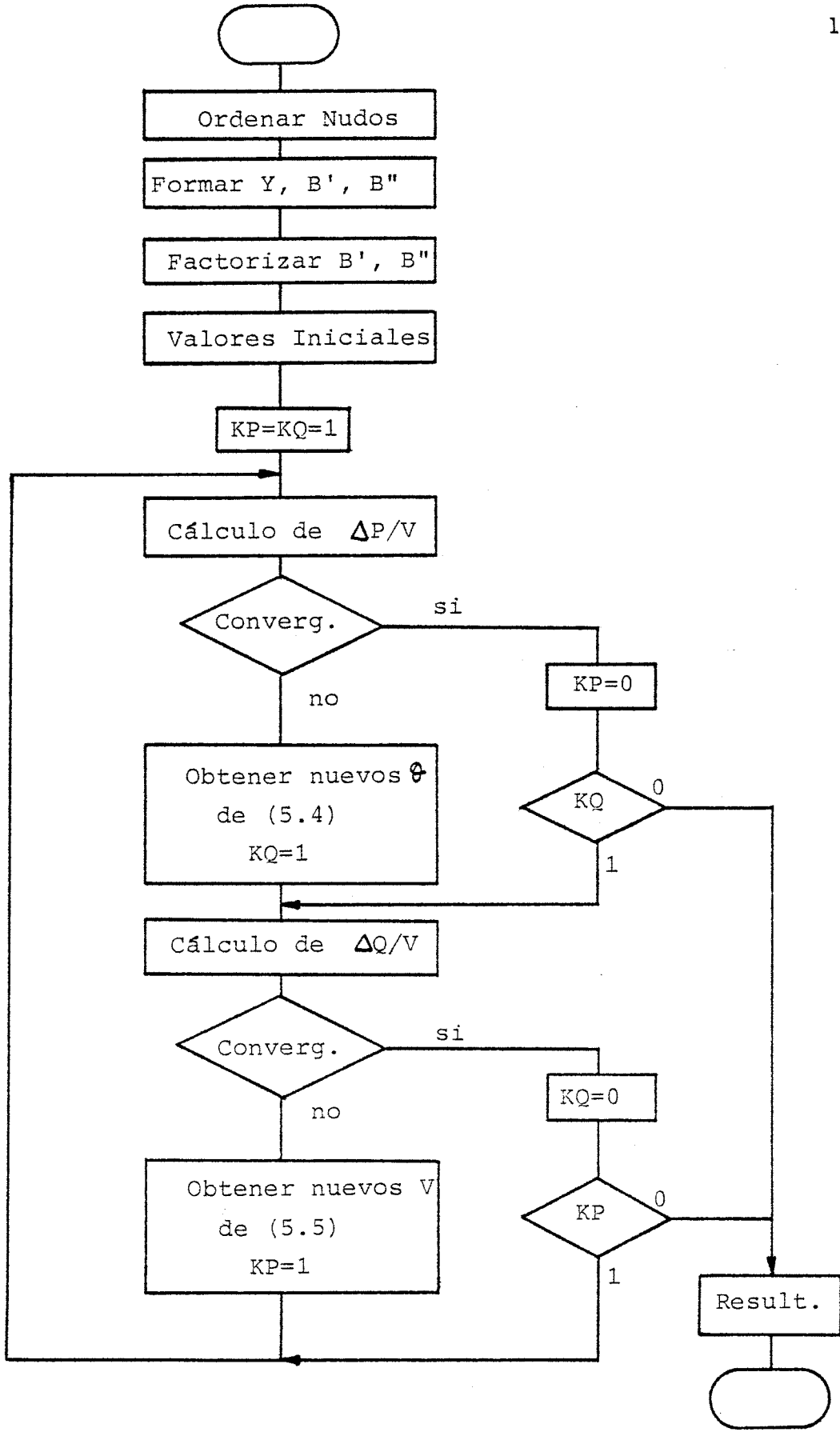

Ordenar Nudos
Formar Y, B', B"
Factorizar B', B"
Valores Iniciales
KP=KQ=1
Cálculo de ΔP/V
Converg.
si
no
KP=0
KQ
0
1
Obtener nuevos θ
de (5.4)
KQ=1
Cálculo de ΔQ/V
Converg.
si
no
KQ=0
KP
0
1
Obtener nuevos V
de (5.5)
KP=1
Result.

fig. 5.6

lizar raices cuadradas, y que puede considerarse como una modificación del de Doolittle para el caso simétrico (véase el párrafo 3.6 y el apéndice 4).

El algoritmo en su conjunto se ha programado siguiendo el diagrama de flujo de la publicación original [103], que se puede ver en la figura 5.6, y que resuelve alternadamente las ecuaciones

$$\Delta P/V = B' \Delta\theta \tag{5.4}$$

$$\Delta Q/V = B'' \Delta V \tag{5.5}$$

hasta que ambas han convergido. El criterio de convergencia es nuevamente $|\Delta Q|, |\Delta P| < 0.0001$ p.u., entendiéndose así en lo sucesivo.

Los resultados obtenidos con este procedimiento se muestran en la figura 5.7.

En comparación con el método de Newton, el número de iteraciones ha aumentado, especialmente en las redes de distribución de 29 y 38 nudos, lo cual era de esperar, dada su alta relación R/X (véase 3.5).

A pesar de todo, los tiempos son mucho menores, si bien es cierto que el método anterior no se ha intentado optimizar. En igualdad de condiciones, una iteración del método de Newton equivale a unas 5 del método desacoplado aproximadamente, debido a que en este último se trabaja con dos matrices que en su conjun-

Desacoplado Rápido (PDP-11/34).

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	.020	.040	.080	.100	.100	.140	.140	.280	.820
Form. Matr.	.180	.260	.360	.380	.380	.460	.420	.640	1.260
Factorizac.	.02	.02	.02	.02	.02	.02	.02	.04	.08
Proc. Iter.	.1	.12	.22	.42	.26	.6	.44	.6	1.140
Total	.32	.44	.68	.92	.76	1.22	1.02	1.560	3.30

Iteraciones	5 1/2	4	4 1/2	7 1/2	3 1/2	8	5 1/2	4 1/2	4 1/2
Tiempo / Itera.	.0181	.03	.048	.056	.074	.075	.08	.133	.2533

fig. 5.7

to, y aprovechando su simétria, suponen casi la cuarta parte del jacobiano, y además sólo se triangularizan una vez.

Más importante que el tiempo total, es el tiempo invertido en el proceso iterativo, pues una vez formadas las matrices, los cambios en la red sólo provocarán pequeñas modificaciones en las mismas, que se pueden realizar rápidamente, no siendo precisa probablemente una reordenación de los nudos.

Por tanto, atendiendo a esos tiempos, parece viable utilizar este algoritmo en tiempo real, incluso en máquinas de poca potencia de cálculo como es la nuestra, y hasta sistemas de cierta dimensión (150 nudos).

Los tiempos unitarios (por cada iteración) pueden ser útiles si pretendemos extrapolar los tiempos de solución para sistemas más grandes que los utilizados. Como muestra la figura 5.8, estos tiempos crecen de forma lineal (la recta dibujada se ha obtenido por un ajuste de mínimos cuadrados).

Por último, las figuras 5.9 y 5.10 muestran gráficamente la velocidad de convergencia de este algoritmo para dos sistemas bien diferentes, de 30 y 38 nudos respectivamente.

El sistema de 30 nudos presenta una convergencia más igualada entre ΔP y ΔQ. Por el contrario, el hecho de que se requieran 8 iteraciones para la red de 38 nudos, se debe a la pobre convergencia que presenta ΔP.

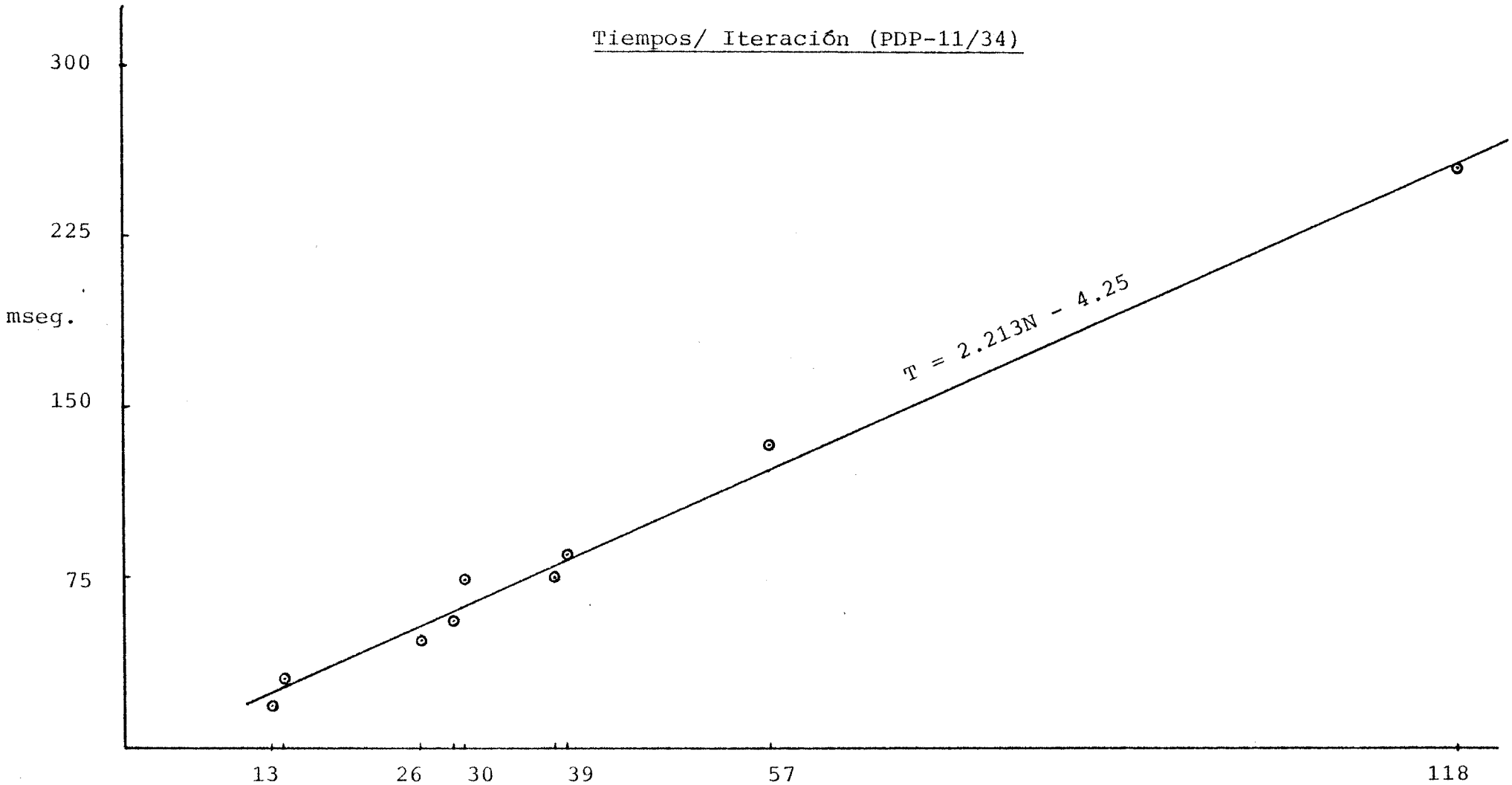

Tiempos/ Iteración (PDP-11/34)
300
225
mseg.
150
75
T = 2.213N - 4.25
13
26
30
39
57
118

fig. 5.8

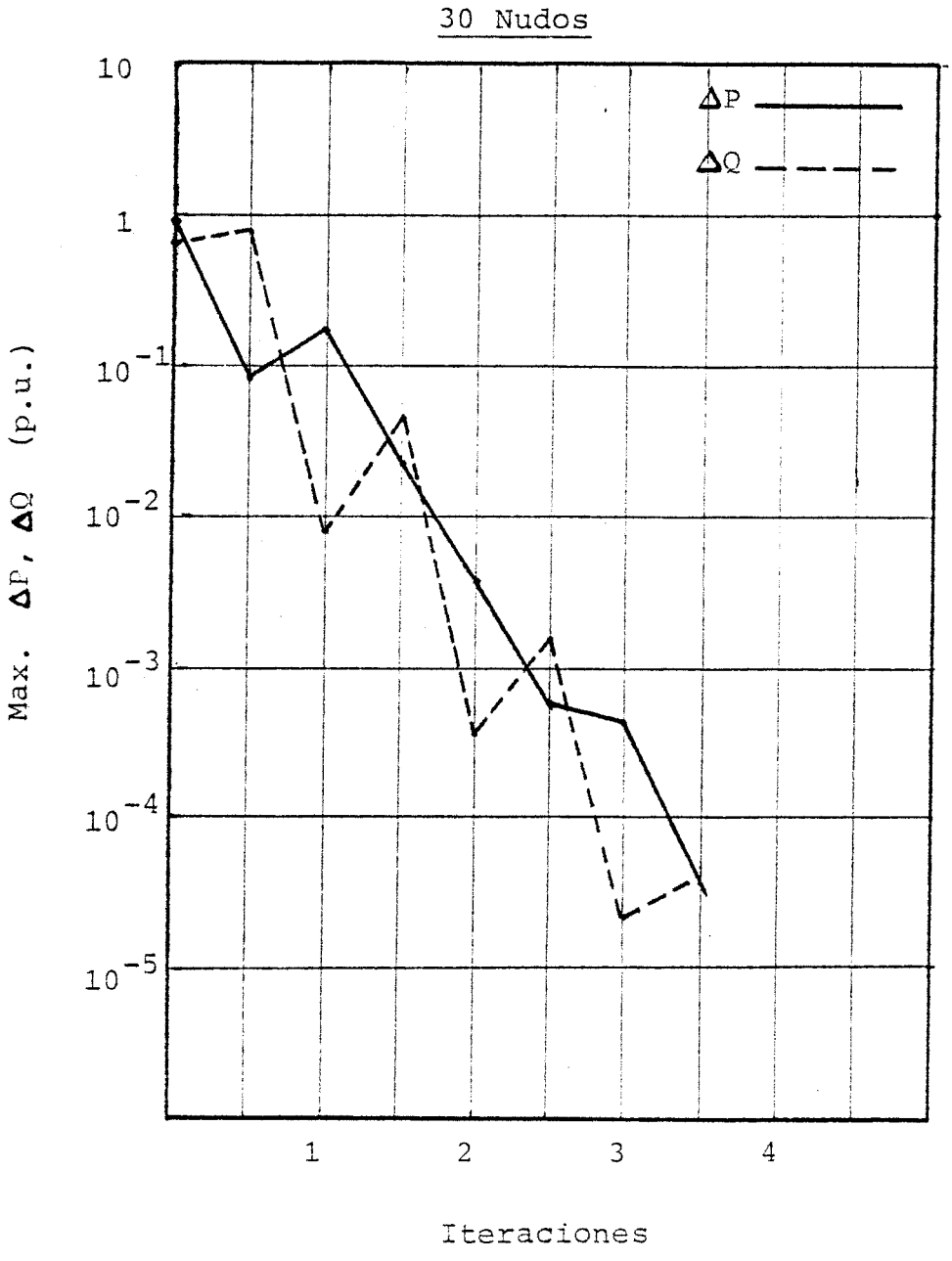

30 Nudos
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
Max. ΔP, ΔQ (p.u.)
ΔP
ΔQ
1
2
3
4
Iteraciones

fig. 5.9

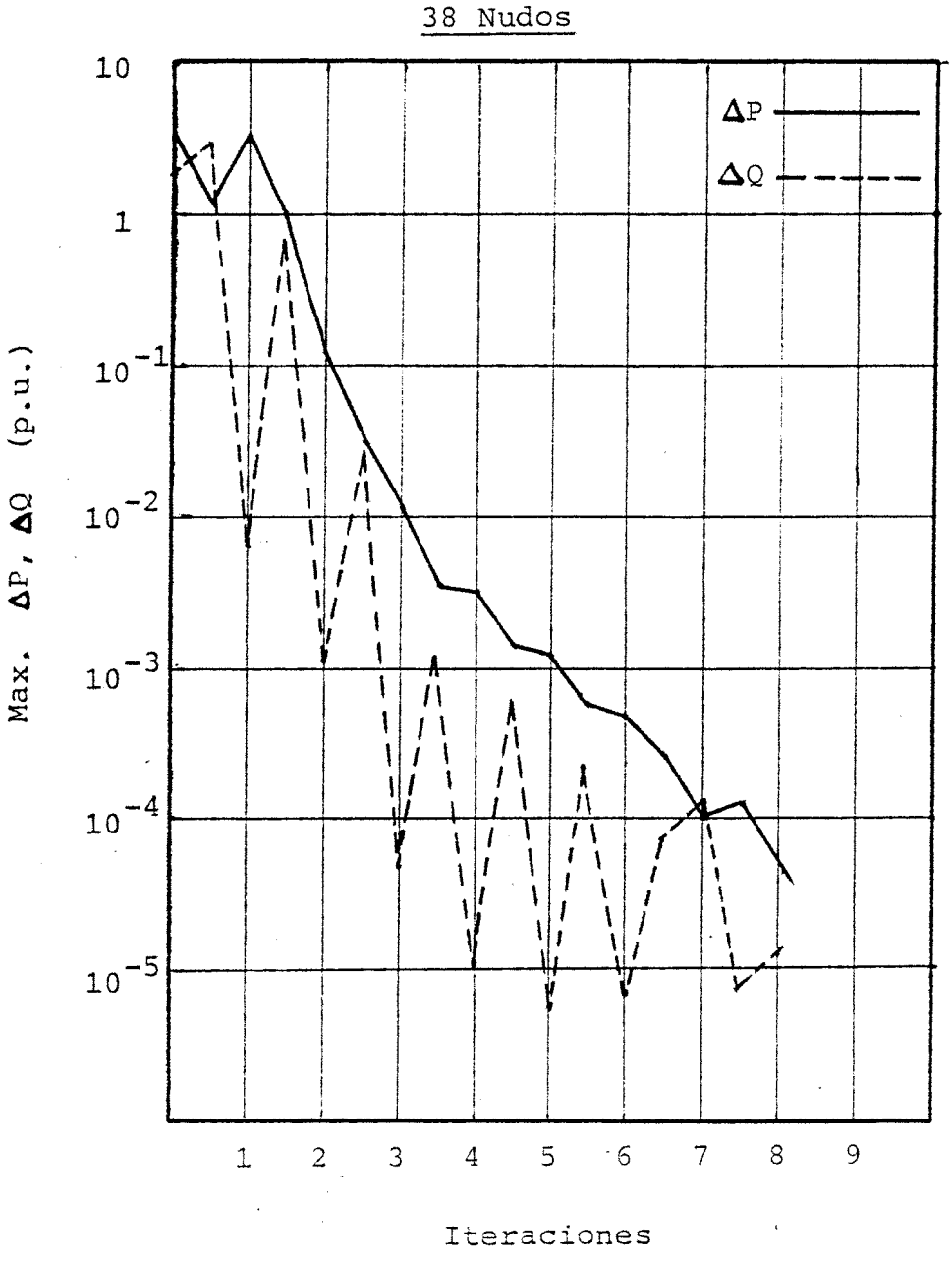

38 Nudos
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
ΔP
ΔQ
Max. ΔP, ΔQ (p.u.)
1 2 3 4 5 6 7 8 9
Iteraciones

fig. 5.10

En ambos sistemas se observa una mayor influencia de los cambios de ΔP sobre ΔQ, que a la inversa, sobretodo en el de 38 mudos. Es decir, cuando iteramos con (5.4) para obtener nuevos θ, disminuimos ΔP (lógicamente), pero aumentamos ΔQ. El efecto contrario es mucho menos pronunciado.

Para indagar el efecto que la relación R/X tiene sobre la convergencia en este algoritmo, se hizo una prueba sencilla con el sistema de 14 nudos del IEEE. La rama que une los nudos 1 y 2 tiene una impedancia serie de 0.01938+j0.05917. Se modificó la resistencia de esta rama al valor 0.05938, es decir, prácticamente se igualó a la reactancia. En otra línea cualquiera esta modificación hubiera supuesto poco frente al resto de la red, pero expresamente se tomó ésta porque es una de las dos que unen el nudo de referencia 1 al resto del sistema, como indica la figura 5.11,

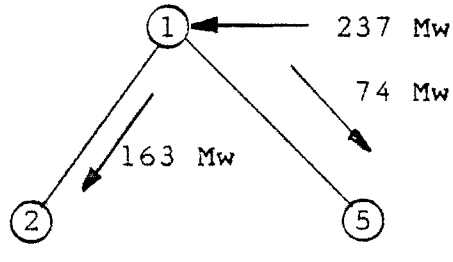


figura 5.11

y casi las 3/4 partes de la potencia activa que se consume, se suministran a través de esa línea, y por tanto resulta crítica. El resultado es que se trasvasa más potencia por la línea 1-5 en detrimento de la 1-2

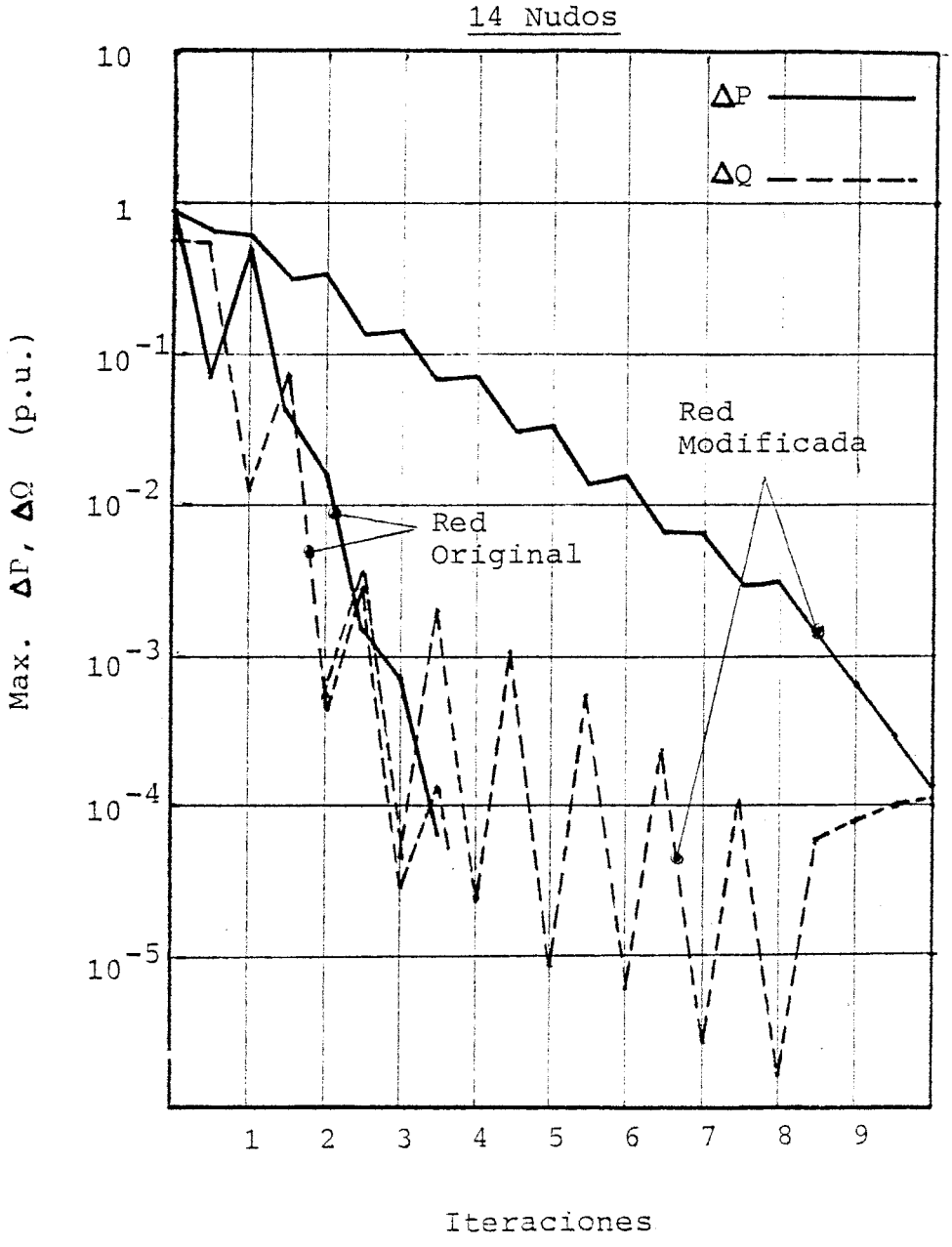

14 Nudos
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
ΔP
ΔQ
Red
Modificada
Red
Original
Max. ΔΓ, ΔΩ (p.u.)
1
2
3
4
5
6
7
8
9
Iteraciones

fig. 5.13

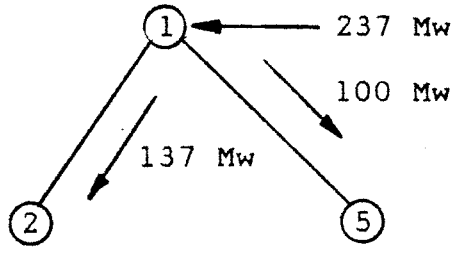


figura 5.12

y las fases de los nudos de toda la red aumentan en unos 3^{o}. Asimismo el número de iteraciones pasa de 4 a 11, como indica la figura 5.13, donde se compara la convergencia en ambos casos. Se observa que ΔQ no se aparta de la original hasta el final del proceso, mientras que ΔP converge muy lentamente, incluso más que en la red de 38 nudos, a pesar de que en esta red no es sólo una línea la que tiene una relación R/X alta, sino la mayoría de ellas. Ello se explica por el carácter estratégico de esta línea, como ha quedado dicho.

La situación presentada es apropiada para aplicarle el algoritmo de Dy Liacco y Ramarao [32] (véase 3.5). Se introduce un nudo ficticio (15) entre 1 y 2, como indica la figura 5.14

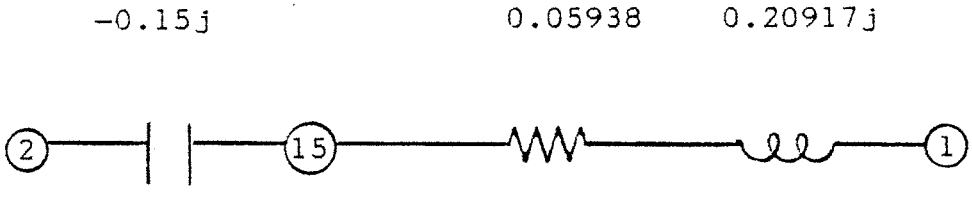


figura 5.14

de forma que el enlace equivalente entre 1 y 2 sea el mismo que anteriormente. Esta simple modificación disminuye el número de iteraciones a 6 1/2 . En la figura 5.15 se puede comparar la convergencia en ΔP para los tres casos: a) Red original, b) Red de resistencia alta en 1-2, c) Red del caso b) pero con un nudo ex-

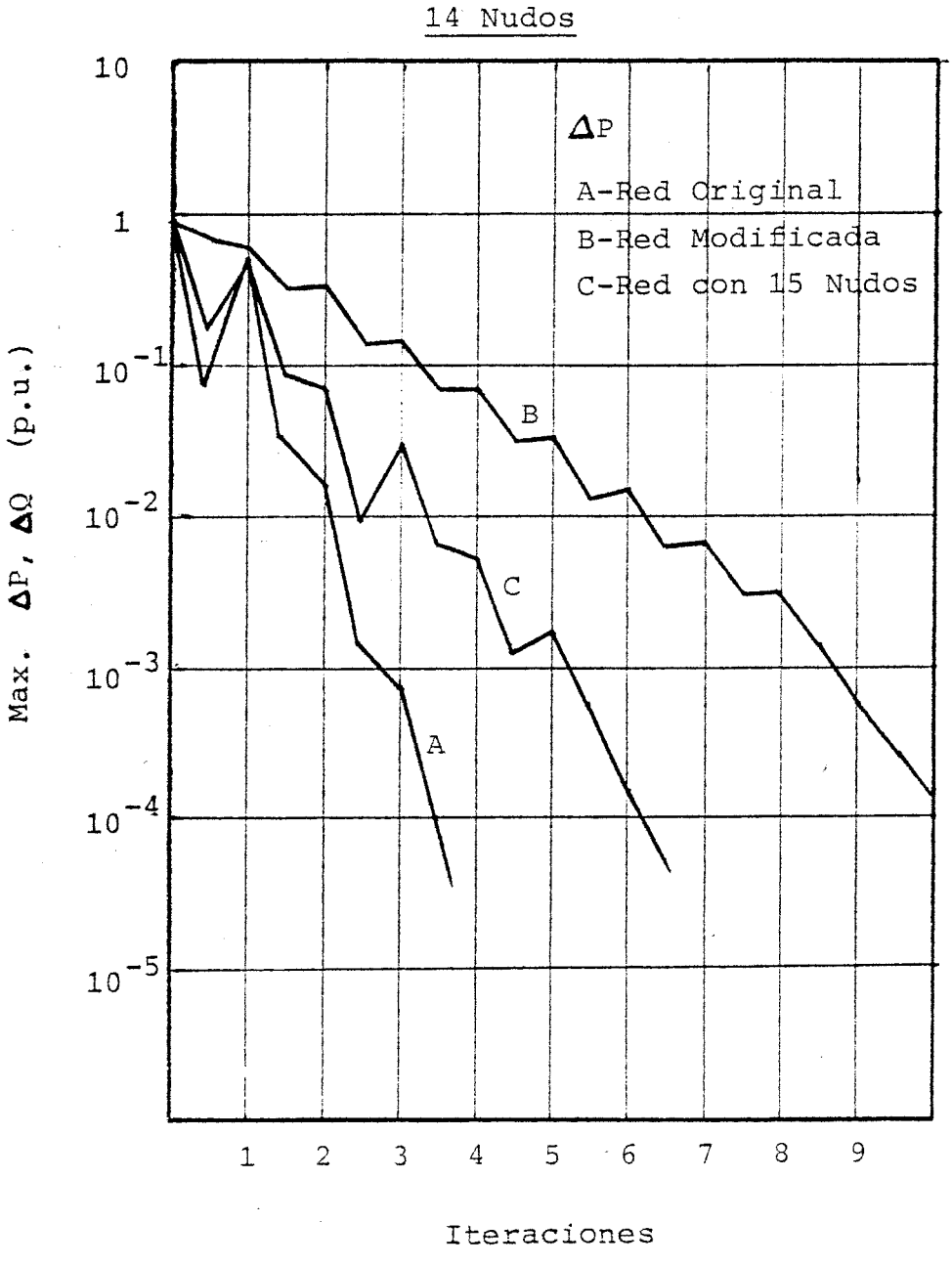

14 Nudos
ΔP
A-Red Original
B-Red Modificada
C-Red con 15 Nudos
Max. ΔP, ΔQ (p.u.)
10
1
10^-1
10^-2
10^-3
10^-4
10^-5
A
B
C
1 2 3 4 5 6 7 8 9
Iteraciones

fig. 5.15

tra entre 1 y 2.

Se observa que este algoritmo mejora el comportamiento de la convergencia en ΔP. Se estaría tentado en aumentar más aún la reactancia capacitiva entre 2 y 15, pero entonces se perjudica notablemente el comportamiento de ΔQ, con oscilaciones mucho mayores, aunque no se haya representado ΔQ en la figura 5.15 para no perder claridad.

En opinión de muchos autores, el método desacoplado rápido es el más eficiente de cuantos existen en la actualidad. Los resultados que se mostraron en la figura 5.7 corroboran esa opinión.

La única duda que se puede presentar es en relación con el algoritmo de segundo orden en coordenadas cartesianas, presentado por Nagendra et al. [73] y descrito someramente en 2.2.25.

Llegados a este punto quedaban dos opciones básicamente: Programar el algoritmo de Nagendra y compararlo con el de Stott y Alsac, o bien continuar profundizando en este último para intentar obtener alguna mejora. Realmente todos los métodos de segundo orden, incluido el de Nagendra, surgieron por ese motivo, como competidores directos del algoritmo desacoplado rápido, pero por la vía de la complejidad en lugar de la simplificación.

Se optó por la segunda, dado el atractivo y la sencillez del algoritmo desacoplado frente a los de segundo orden, a pesar de arriesgarnos al fracaso, puesto que dicho algoritmo está muy de-

purado desde su misma publicación.

En lo que resta de este epígrafe se mencionarán brevemente algunos de los intentos infructuosos.

5.5.1 ALGUNAS VARIANTES DEL METODO DESACOPLADO.

Las figuras 5.9, 5.10, 5.13 y las respectivas para los otros sistemas, nos indican que ΔQ cae por debajo de ΔP, y por tanto es ΔP la que limita casi siempre la velocidad de convergencia. Por tanto, o bien mejoramos la convergencia en ΔP para disminuir el número de iteraciones, o bien simplificamos aún más la obtención de ΔV en función de ΔQ, de manera que, aunque empeore la convergencia del problema reactivo, el número de iteraciones no aumente, pero sin embargo disminuya el tiempo total gracias a esa simplificación.

Método 1. Actualizar los ángulos de la ecuación (5.4) y los módulos de la ecuación (5.2), reescritas aquí nuevamente.

$$\Delta P/V = B' . \Delta\theta \qquad (5.6)$$

$$AV_i^2 + BV_i + C = 0 \qquad (5.7)$$

El mejor esquema resultó de iterar una vez en (5.6), seguido de 2 veces en (5.7), utilizándose factores de aceleración en esta última. La virtud de este procedimiento era el evitar la formación y triangularización de B" en (5.5), lo que ahorraba memoria

y en principio tiempo. Sin embargo, la pobre convergencia de (5.7), especialmente para las redes más grandes, ralentizaba también la de (5.6), y el número total de iteraciones siempre era 2 ó 3 más que en el algoritmo original.

Método 2. Utilizar el método de Brown, descrito en 2.2.18 y que se había aplicado con cierto éxito al método de Newton-Raphson [63], en la solución de (5.6). Se trataba de mejorar la convergencia del subproblema activo, resolviendo el problema reactivo en la forma usual, según la ecuación (5.5).

Se probaron diversas variantes, a la vista de las gráficas de la convergencia, como utilizar el método de Brown sólo en la primera iteración, y continuar después normalmente; o justo al revés, empezar normalmente, y sólo al final aplicar la técnica de Brown. Pero salvo casos muy aislados, el número de iteraciones siempre fue mayor o igual, con el agravante de que cada solución de la ecuación (5.6) requiere más tiempo, puesto que las componentes individuales del vector ΔP se deben calcular por separado, no aprovechándose la simetría de la matriz de admitancias.

Método 3. En el primer método desacoplado que publicó Stott [102], que simplemente ignoraba las matrices no diagonales del jacobiano, sugirió utilizar la parte imaginaria de la intensidad en lugar de ΔQ, porque parecía comportarse mejor lejos de la solución (véase 2.2.16). Aquí se ha intentado incorporar la misma idea al método rápido. La ecuación que sustituye a (5.5) es

$$\Delta I = D \cdot \Delta V \qquad (5.8)$$

donde D, a diferencia de B", no es constante, y por tanto no se puede formar y triangularizar sólo una vez antes del proceso iterativo, con $\theta = 0$. En lugar de ello se pospone su formación hasta iterar una vez con (5.6), y ya con los θ obtenidos, que no están demasiado lejos de los definitivos, se forma D y se triangulariza. La matriz D ya no se modifica en sucesivas iteraciones de (5.8), lo que constituye lógicamente una aproximación adicional.

Los resultados obtenidos tampoco fueron buenos, aumentando siempre al menos media iteración, y siendo más lenta de resolver la ecuación (5.8) que la (5.5), debido a la formación de ΔI.

Método 4. Dado que el tiempo invertido en triangularizar B' no es tan alto (véase la figura 5.7), puede ser interesante, especialmente en los sistemas que requieren un gran número de iteraciones, volver a triangularizar B' cuando se conocen unos

θ mejores que los iniciales ($\theta = 0$). Ello supondría introducir menos simplificaciones en B', definiéndose sus elementos por

$$B'_{ij} = - V_j B_{ij} \cos \theta_{ij} \qquad (5.9)$$

El proceso a seguir es : Realizar una iteración completa del método rápido, con las ecuaciones (5.4), (5.5). Con los valores obtenidos de V, θ, recalcular B' según (5.9) y factorizarla. Continuar en la forma usual, sin recalcular más veces B'.

Lo que se logra así es utilizar una tangente más exacta a partir de la primera iteración. Por tanto, en buena lógica, se

Método 4.

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	.04	.04	.08	.1	.12	.16	.12	.28	.82
Form. Matr.	.18	.28	.38	.4	.42	.48	.5	.72	1.44
Factorizac.	.02	.02	.02	.02	.04	.06	.04	.1	.18
Proc. Iter.	.1	.12	.22	.42	.28	.5	.34	.46	1.04
Total	.34	.46	.7	.94	.86	1.2	1.	1.56	3.48

Iteraciones	5 1/2	4 1/2	4 1/2	7 1/2	4 1/2	7 1/2	4 1/2	4	4 1/2
Tiempo / Itera.	.018	.0266	.048	.056	.0622	.066	.075	.115	.231

fig. 5.16

deberían ahorrar iteraciones. La figura 5.16 muestra los resultados de este procedimiento. Comparándola con la 5.7 observamos que el número de iteraciones permanece igual para los sistemas de 13, 26, 29 y 118 nudos, que aumenta para los de 14 y 30, y que disminuye para los de 38, 39 y 57, aunque en estos últimos los tiempos no han mejorado, porque el ahorro de iteraciones se compensa con el cálculo y triangularización de B' por segunda vez.

De los 4 métodos descritos, éste es sin duda el que mejor resultados aporta. Aparentemente, aunque puede ser debido a la casualidad, funciona mejor para sistemas grandes, según se desprende del párrafo anterior. En general, parece presentar mejor convergencia que el método convencional para sistemas difíciles, aunque esta afirmación debe ser contrastada más a fondo.

Otras pruebas realizadas, no catalogables en ninguno de estos 4 métodos, han consistido en jugar con los valores iniciales para los desfases. Como el nudo de referencia suele ser uno de los que más potencia activa genera, el resto de nudos tendrá mayoritariamente un desfase negativo. Incluso es factible, dado que los nudos de carga son mayoría frente a los de generación, obtener un desfase medio, considerando una carga equivalente en un hipotético "centro de gravedad" del sistema. La red de 14 nudos es ideal para estas pruebas, puesto que casi toda la carga se inyecta en el nudo de referencia, que está en la periferia del grafo. Para los otros nudos, el valor medio de los ángulos es de -13^{o}, con una desviación típica de $\pm 3^{o}$.

Sin embargo, tomando como valores iniciales para las fases -13^{o} en lugar de 0^{o}, el número de iteraciones no disminuyó, por lo que se decidió abandonar estos trabajos.

5.6 SIMPLIFICACION DEL METODO DESACOPLADO RAPIDO.

En este epígrafe se describirá otra modificación original del algoritmo desacoplado rápido, que ha funcionado bien para la casi totalidad de sistemas ensayados.

Se motivará al lector primeramente con el caso escalar, por su fácil visualización gráfica. Considérese la figura 5.17.

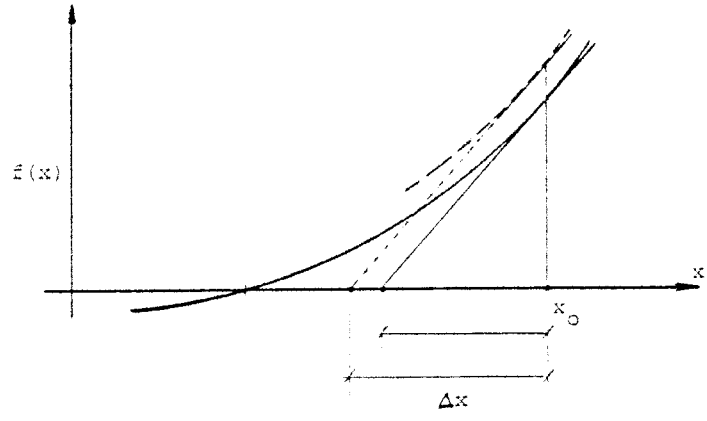


fig. 5.17

Pretendemos obtener la solución de $f(x)=0$, a través del método de Newton, a saber:

$$- J_o \cdot \Delta X = f(x_o) \qquad (5.10)$$

$$X = X_o + \Delta X \qquad (5.11)$$

Las soluciones clásicas para ahorrar tiempo han sido:

a) Partir de un mejor valor inicial X_o, de forma que esté más próximo a la solución. La obtención de este valor inicial no debe ser muy costosa, para que no contreste el ahorro de tiempo que resulta al disminuir el número de iteraciones.
b) Realizar aproximaciones en el jacobiano, de manera que disminuya el tiempo por iteración. Las aproximaciones deben ser tales que no aumente excesivamente el número de iteraciones, para que el tiempo total sea menor.

En el caso concreto del flujo de cargas, la solución a) no ha dado prácticamente ningún resultado. Por el contrario la solución b) ha sido muy fructífera, dando lugar a los métodos desacoplados que han demostrado ser muy eficientes.

Si en la ecuación (5.10) modificamos la función f, la solución obtenida será lógicamente diferente. Ahora bien, podemos aproximar f sólo al comienzo del proceso iterativo, y utilizar su definición exacta cuando nos acercamos al final. En concreto (fig. 5.17), si en la primera iteración, tomamos f como indica la línea discontinua, el resultado será un ΔX mayor, y habremos acelerado la solución.

El comportamiento no es tan simple en el caso n-dimensional, como ya sabemos, pero se ha intentado aplicar esta idea de la forma siguiente:

Considérese la ecuación (5.4), repetida aquí:

$$\Delta P/V = B' \cdot \Delta\theta \tag{5.12}$$

donde $\Delta P/V$ se define por:

$$\frac{\Delta P_i}{V_i} = \frac{P_i^{esp}}{V_i} - \sum_{j=1}^{n} V_j \, (G_{ij} \cos \theta_{ij} + B_{ij} \operatorname{Sen} \theta_{ij}) \tag{5.13}$$

Para la primera iteración $\theta = 0$, de manera que $\cos \theta_{ij} = 1$ y $\operatorname{Sen} \theta_{ij} = 0$, y por tanto podemos evitar estos cálculos para dicha iteración, como ya los autores indicaron en su publicación original [103], quedando:

$$\frac{\Delta P_i}{V_i} = \frac{P_i^{esp}}{V_i} - \sum_{j=1}^{n} V_j \, G_{ij} \tag{5.14}$$

Podemos simplificar aún más (5.14) mediante ciertas aproximaciones:

a) Considerar $V_i = 1$, incluso para los nudos P-V, donde V puede ser diferente de la unidad.

b) Suponer que

$$\sum_{j=1}^{n} G_{ij} = 0$$

lo cual es bastante realista, puesto que G_{ii} es la suma cambiada de signo de los elementos no diagonales, cuan-

do se utilizan parámetros concentrados para el modelo de líneas.

Con estas aproximaciones (prácticamente sólo la a), puesto que la b) no lo es), la (5.14) queda:

$$\frac{\Delta P_i}{V_i} = P_i^{esp} \qquad (5.15)$$

Si tenemos en cuenta que el tiempo requerido en cada iteración se dedica principalmente al cálculo de ΔP y ΔQ, se concluye que, aunque la utilización de (5.15) no ahorre iteraciones, el tiempo total será menor al no necesitarse realizar operaciones para el primer cálculo de ΔP.

La figura 5.18 muestra los resultados obtenidos al aplicar esta simplificación.

Sorprendentemente (compárese con la fig. 5.7) se ahorran iteraciones en las redes de 14, 29, 38 y 39 nudos. Las demás redes requieren el mismo número de iteraciones, como era de esperar dada la ligera aproximación hecha, salvo la red de 118 nudos que necesita media iteración extra.

Aunque en los tiempos puede haber cierto error, dadas las características del sistema operativo, se puede concluir que en todos los sistemas excepto en el de 118 nudos, los tiempos han disminuido.

Desacoplado Rápido Simplificado.

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	.02	.04	.08	.1	.12	.14	.14	.3	.84
Form. Matr.	.18	.28	.36	.38	.38	.46	.44	.62	1.28
Factorizac.	.02	.02	.02	.02	.02	.02	.02	.06	.08
Proc. Iter.	.1	.07	.2	.41	.24	.54	.38	.56	1.34
Total	.32	.41	.66	.91	.76	1.16	.98	1.54	3.54

Iteraciones	5 1/2	3 1/2	4 1/2	7	3 1/2	7	5	4 1/2	5
Tiempo / Itera.	.018	.02	.044	.0585	.068	.077	.076	.124	.268

fig. 5.18

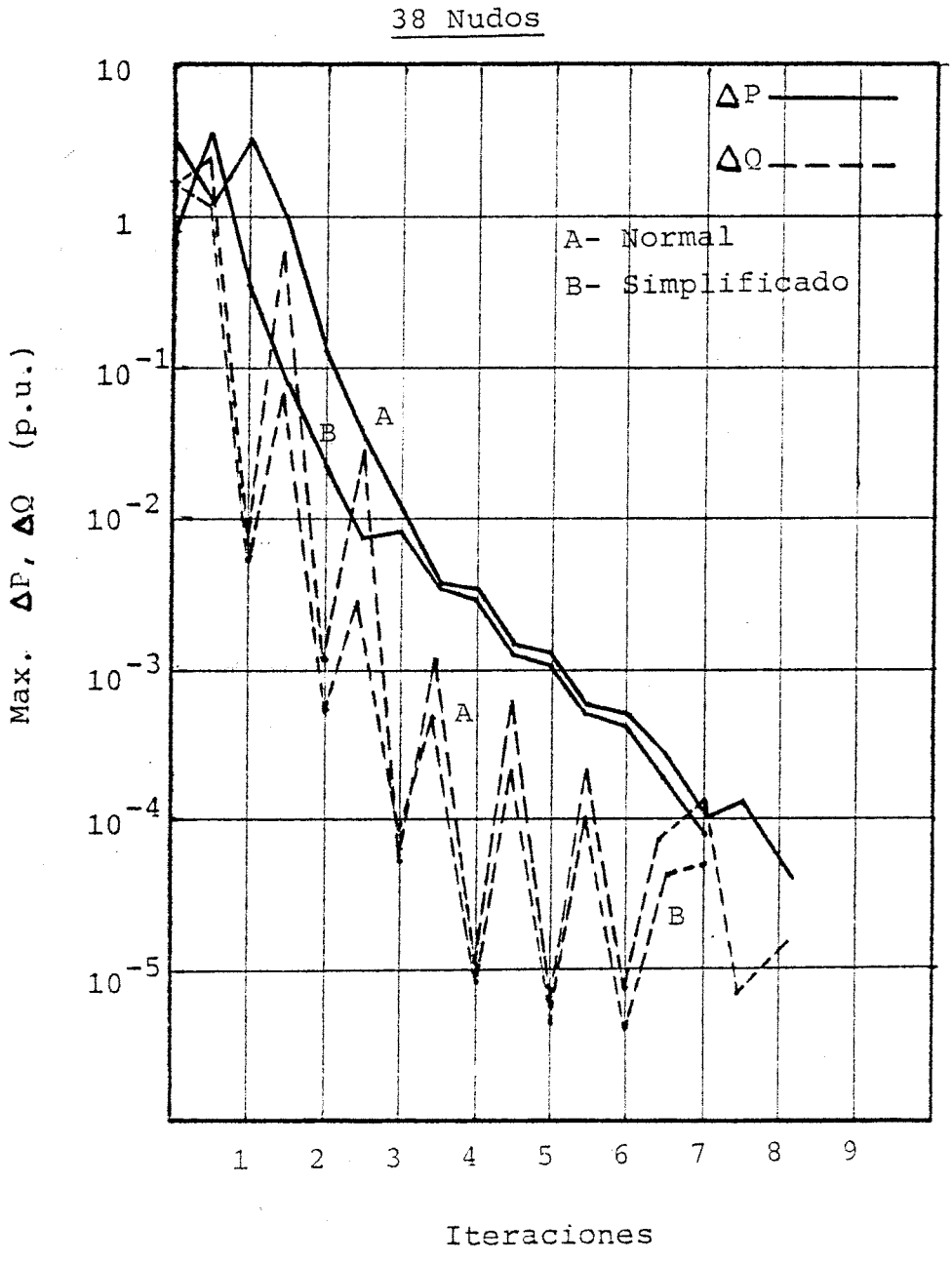

38 Nudos
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
ΔP
ΔQ
A- Normal
B- Simplificado
A
B
A
B
1 2 3 4 5 6 7 8 9
Max. ΔP, ΔQ (p.u.)
Iteraciones

fig. 5.19

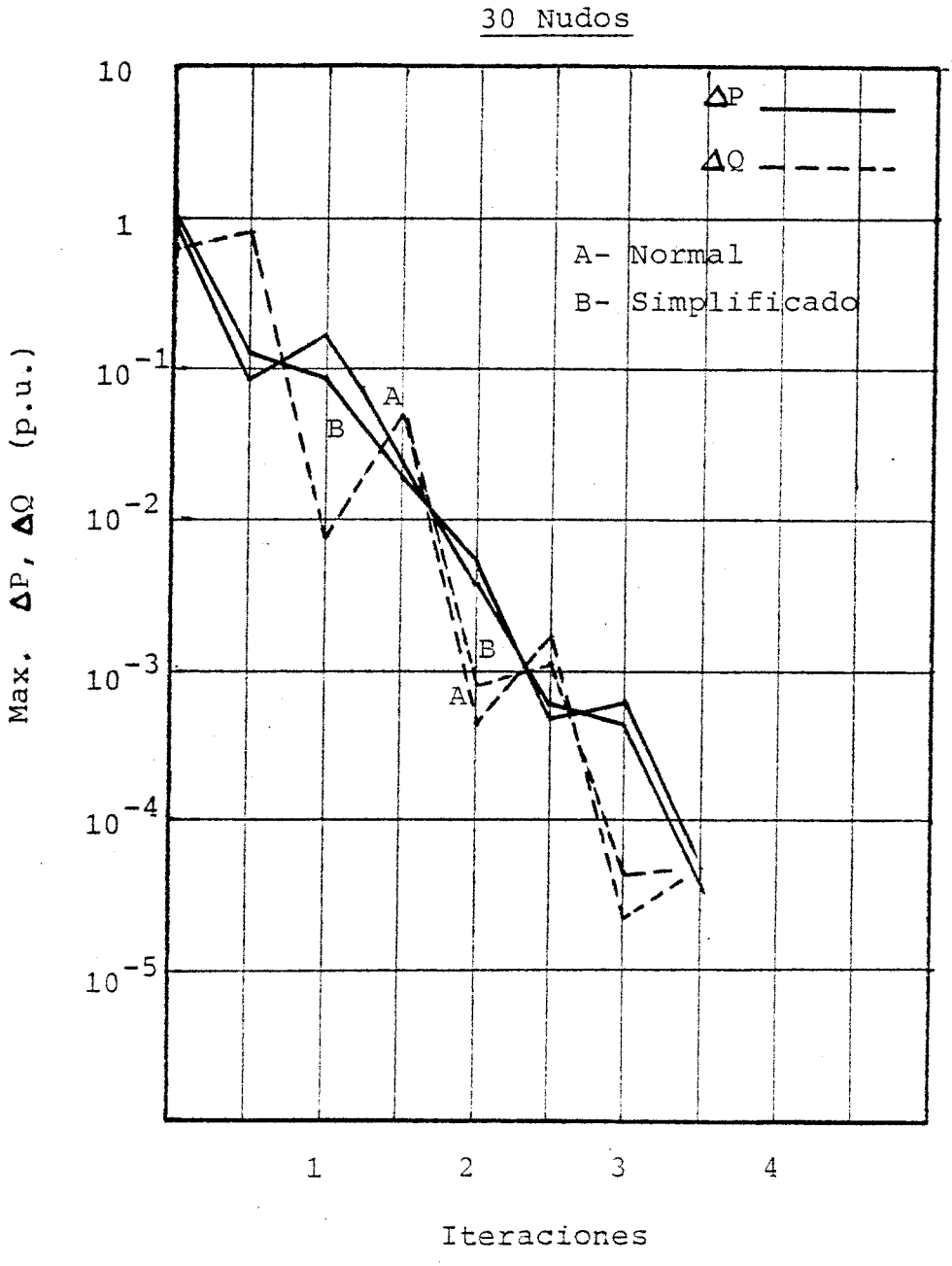

30 Nudos
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
ΔP
ΔQ
A- Normal
B- Simplificado
A
B
B
A
Max. ΔP, ΔQ (p.u.)
1
2
3
4
Iteraciones

fig. 5.20

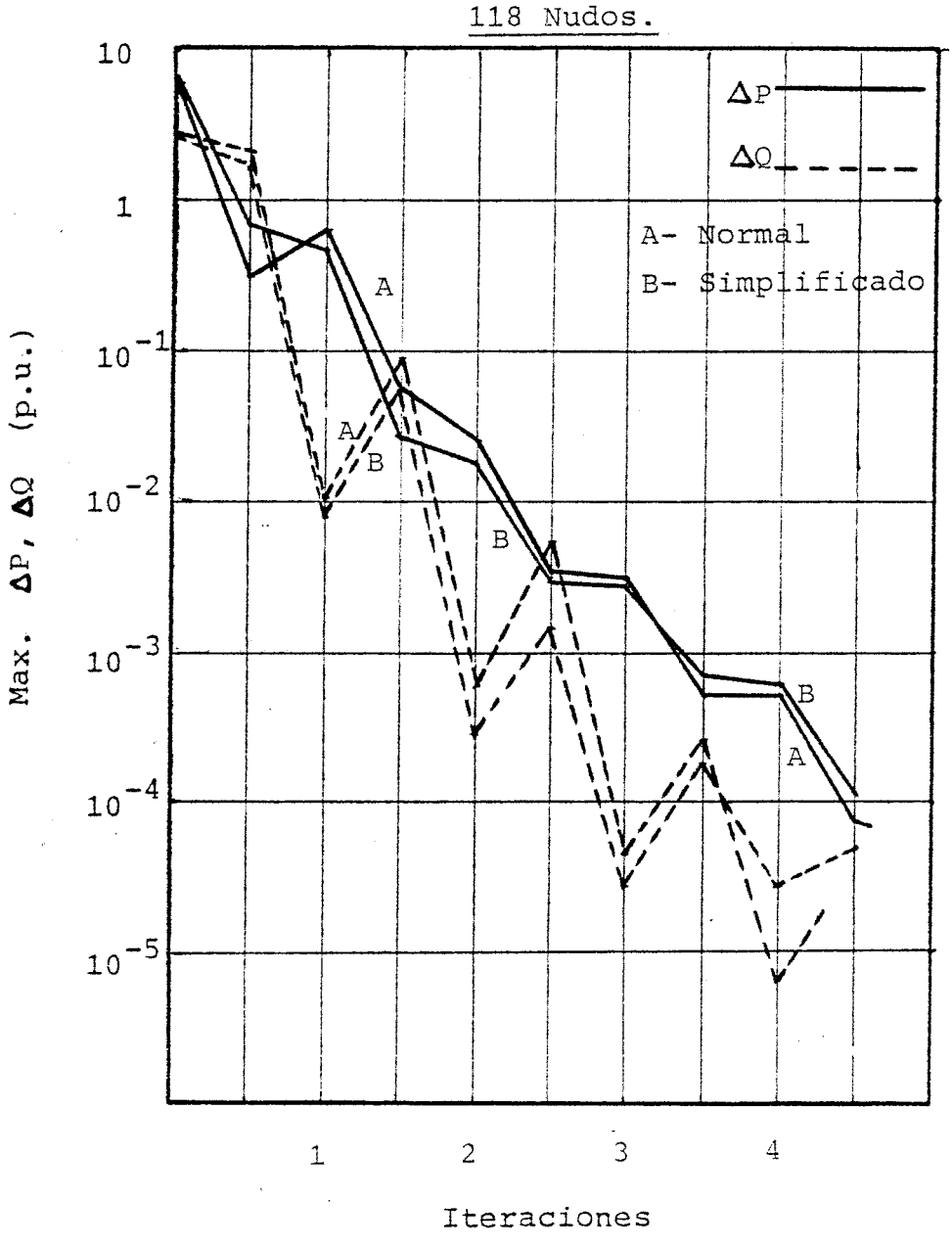

118 Nudos.
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
ΔP
ΔQ
A- Normal
B- Simplificado
A
B
A
B
B
A
1
2
3
4
Iteraciones
Max. ΔP, ΔQ (p.u.)

fig. 5.21

Las figuras 5.19, 5.20, y 5.21 muestran la convergencia de esta simplificación en comparación con el método rápido normal, para tres casos diferentes. El primero es la red de 38 nudos, donde se ahorra una iteración completa. El segundo es la red de 30 nudos, donde no se ahorra nada, lo cual es difícil de lograr dada su buena convergencia (tres iteraciones y media). Y el tercero es la red de 118 nudos, la única donde se pierde media iteración, de forma muy ajustada según indica la figura 5.21.

Las tres tienen en común que la simplificación mejora la convergencia al principio del proceso, pero que al final ambas curvas tienden a igualarse, especialmente las de la potencia activa. Ello indica que se debe profundizar más en este sentido, pués si se logra darle una explicación física, se podrán conseguir mejoras más sustanciosas.

Una explicación matemática, dentro de la cautela que debe tenerse al generalizar un resultado escalar a una dimensión mayor, es la que se dió al principio de este epígrafe en base a la figura 5.17.

Finalmente, la figura 5.22 muestra la red de 29 nudos, cuyo comportamiento es extraordinariamente parecido al de la red de 38 nudos. Recuérdese que ambas son redes de distribución.

29 Nudos.

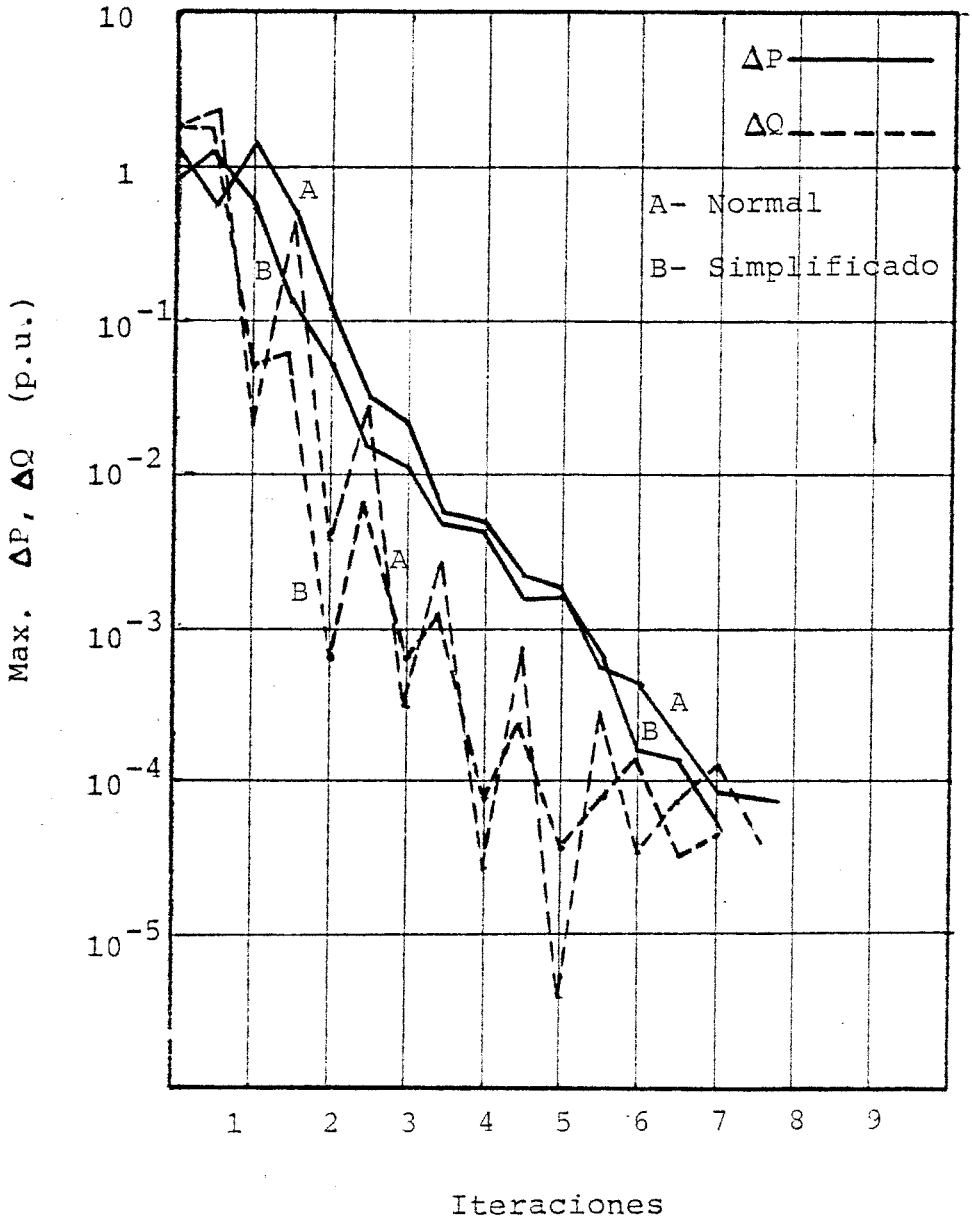


fig. 5.22

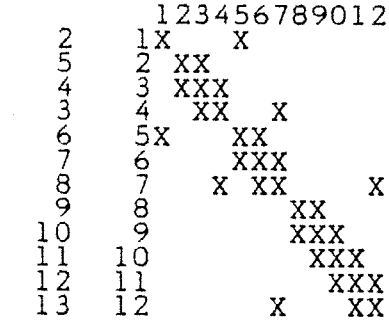
5.7 ORDENACIONES.

Para acabar este capítulo, mostraremos la eficacia de diferentes ordenamientos en las redes eléctricas.

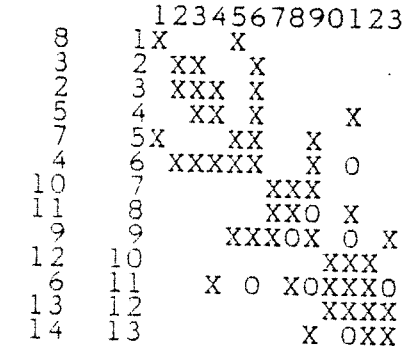
Las figuras 5.23 a 5.28 inclusive, muestran las matrices B' para todas las redes estudiadas, ordenadas según el esquema Tinney-2 (capítulo 4). Como se recordará, la matriz B' no incluye el nudo de referencia y sus conexiones, y ésta es su única diferencia estructural con la matriz de admitancias de nudos. Por tanto, en los algoritmos de ordenación, se ignorará siempre dicho nudo y las líneas que en él inciden. La matriz B" es una submatriz de B', que resulta de eliminar todas las filas y columnas correspondientes a nudos PV. La ordenación óptima para dicha matriz no coincide necesariamente con la de B', pero no es rentable realizar dos ordenaciones diferentes, de forma que siempre nos basaremos en B' que es de dimensión mayor.

En las mencionadas figuras, una "X" indica un elemento original de la matriz, mientras que una "O" refleja un elemento que ha aparecido durante la triangularización. Así se puede captar visualmente la eficacia de la ordenación, simplemente por inspección de la cantidad de letras "O" presentes en la matriz.

La primera columna de la izquierda muestra la ordenación original de los nudos, que se debe recordar para mostrar posteriormente los resultados.

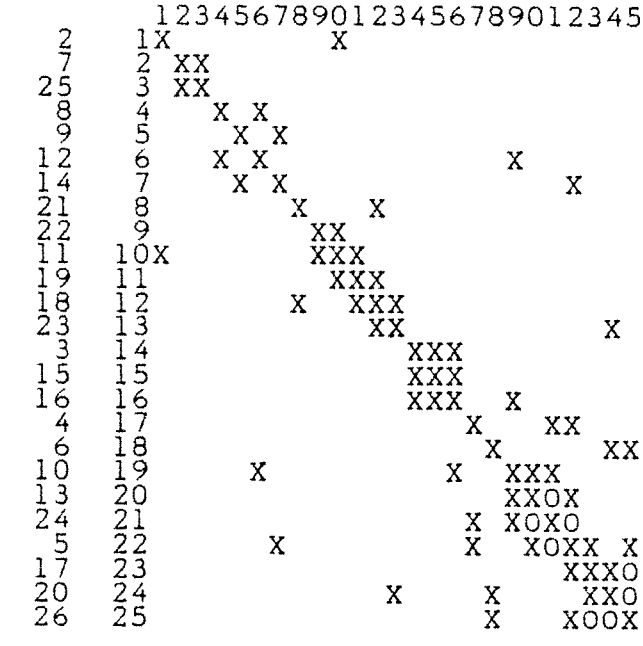


ELEMEN. NO DIAG. = 22

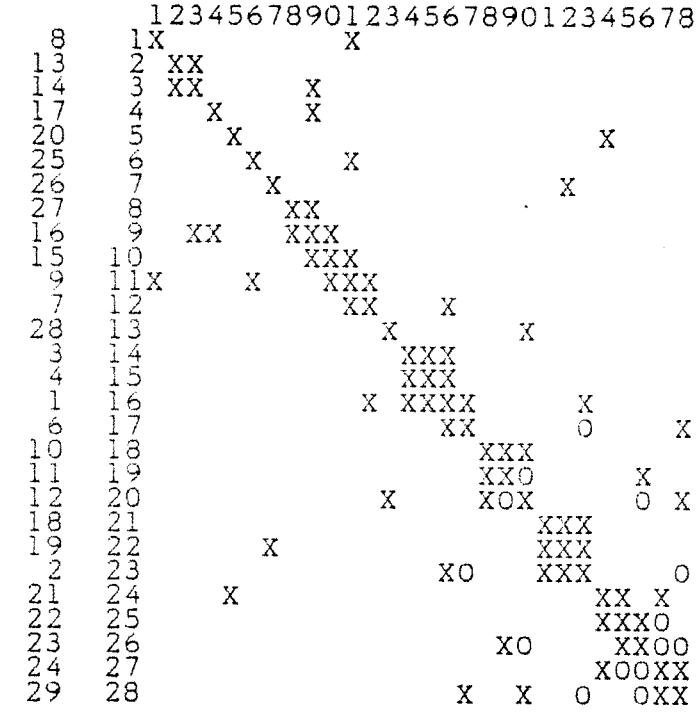


ELEMEN. NO DIAG. = 44

fig. 5.23

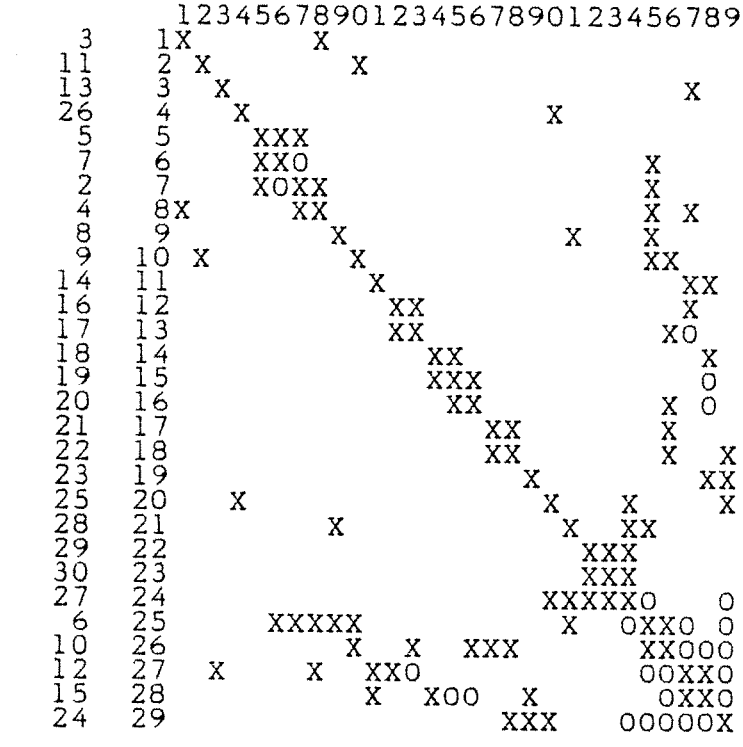


ELEMEN. NO DIAG. = 60

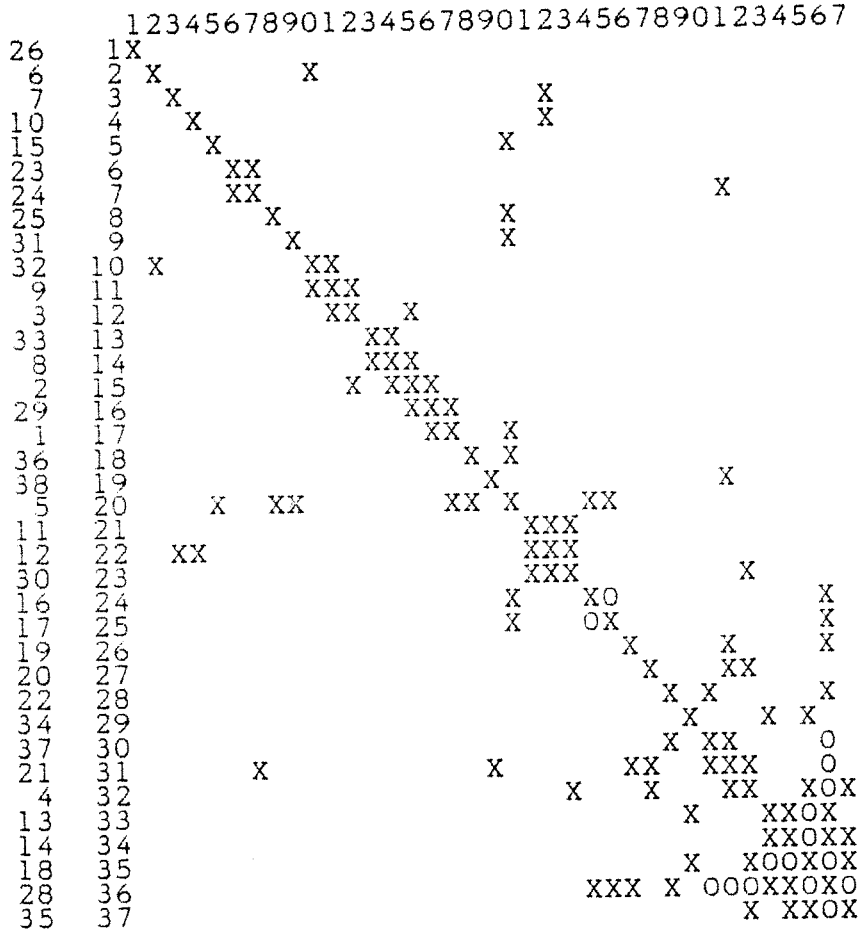


ELEMEN. NO DIAG. = 74

fig. 5.24

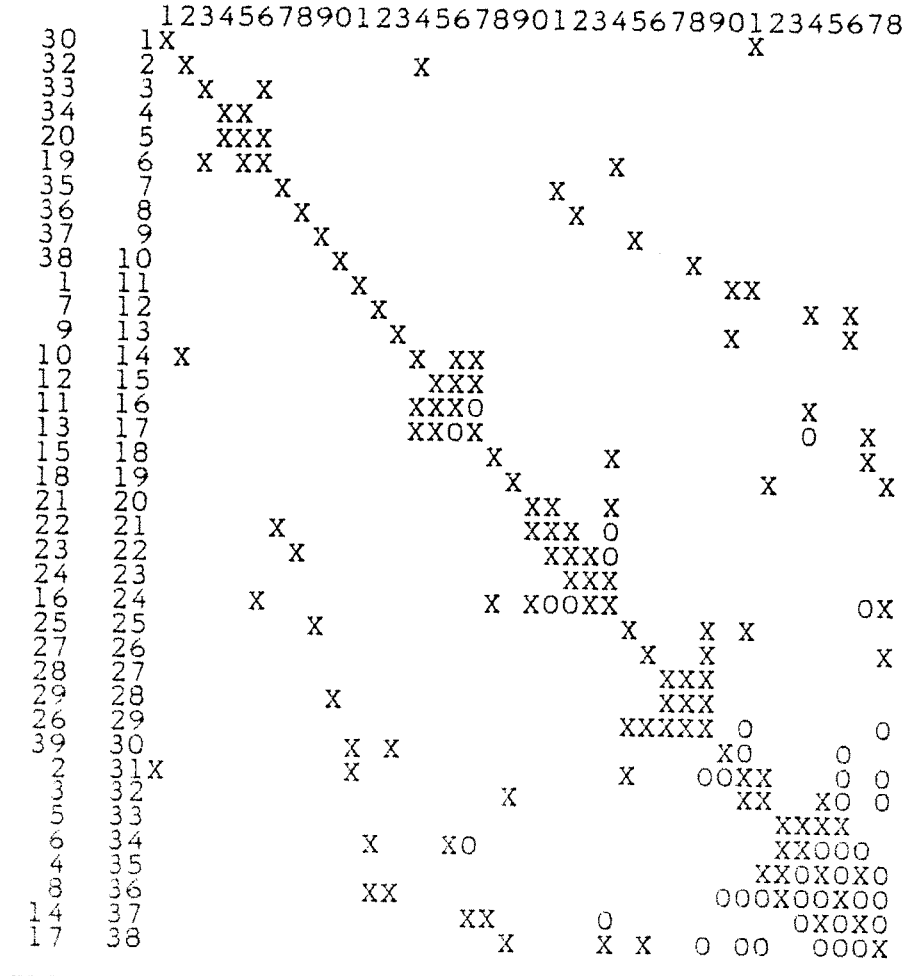


ELEMEN. NO DIAG. = 104



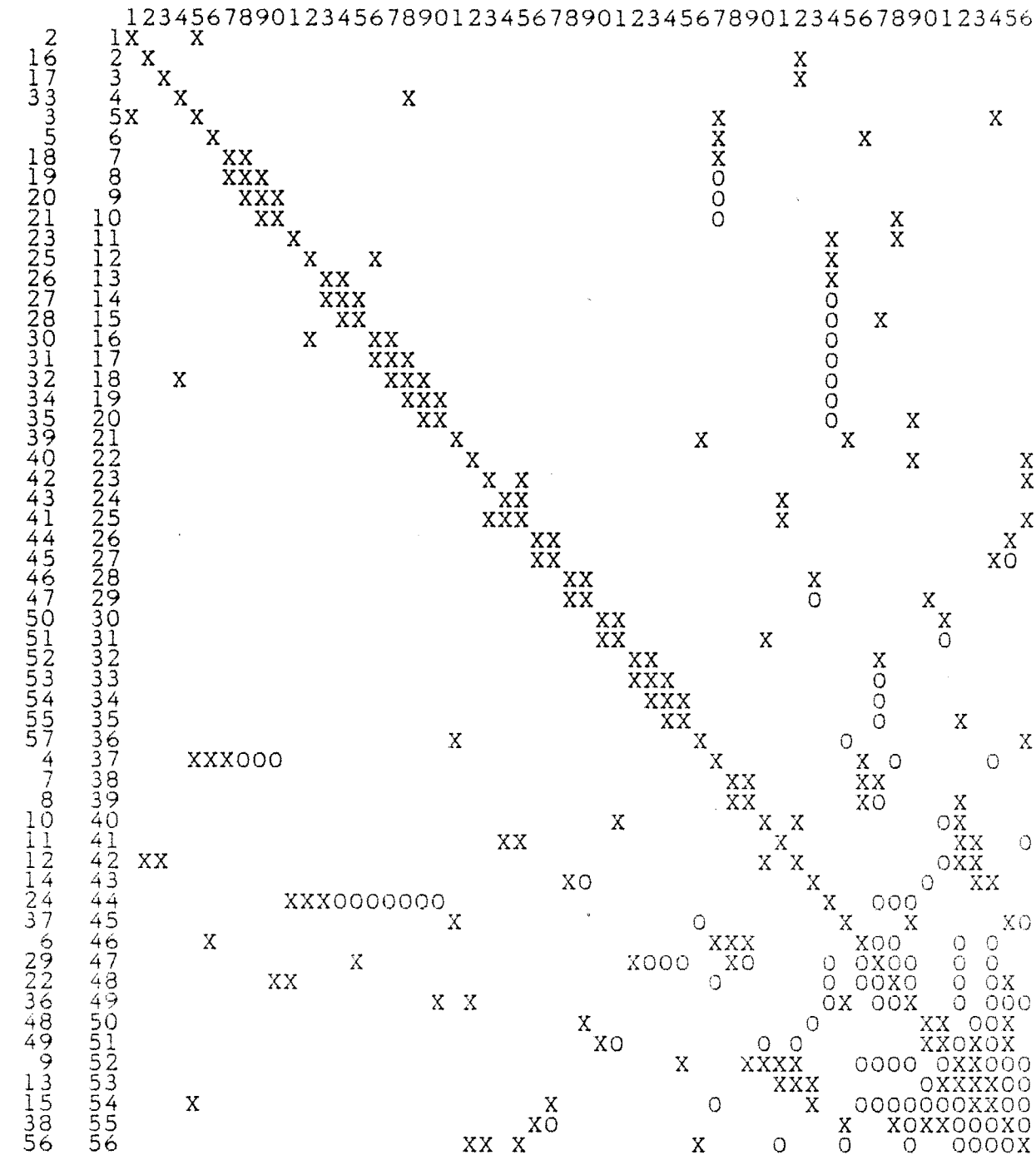
ELEMEN. NO DIAG. = 102

fig. 5.25



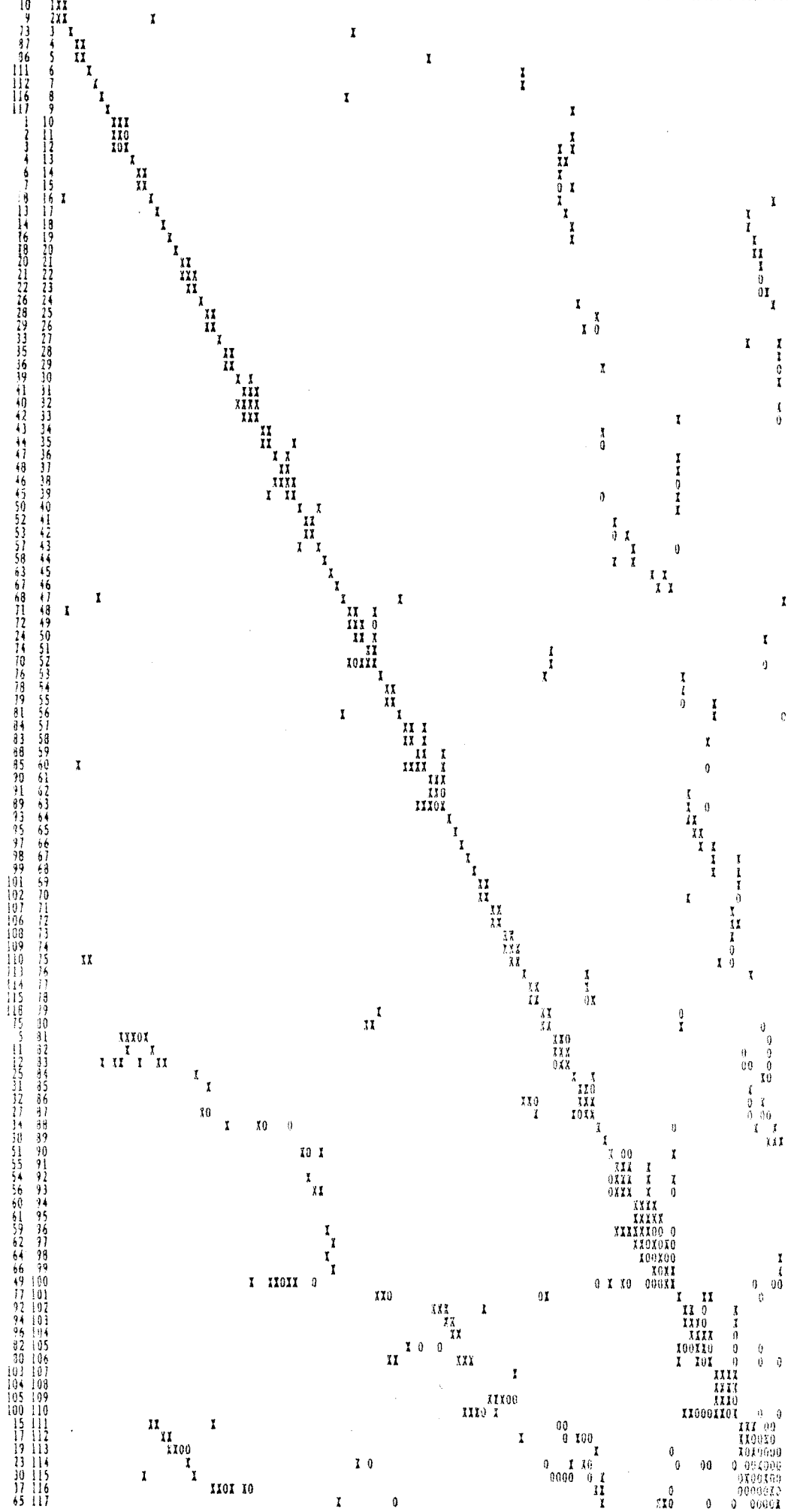
ELEMEN. NO DIAG. = 132

fig. 5.26



ELEMEN. NO DIAG. = 258

fig. 5.27



ELEMEN. NO DIAG. = 506

El número total de elementos, sin contar los de la diagonal, se muestra junto a cada matriz. En realidad, sólo se almacena la mitad superior de la matriz, y por tanto el número de elementos hay que dividirlo por dos, si pretendemos calcular la capacidad de memoria necesaria.

Para el caso concreto de ordenación según el esquema 2 de Tinney, se observa un llenado escaso de la matriz, de manera que resulta muy eficiente.

Esta afirmación resulta más evidente si se comparan las figuras 5.23 a 5.28, con las 5.29 a 5.31, donde se muestran las redes de 13, 26 y 39 nudos ordenadas en forma de banda (siguiendo el algoritmo de Cuthill-Mckee simplificado).

Los tres sistemas se representan antes y después de su triangularización. Se puede observar el cumplimiento de algunos lemas enunciados en el capítulo 4, entre ellos los que afirmaban que no aparecían elementos nuevos fuera de la banda, y que toda la envoltura se llenaba, de ahí que si la envoltura y/o el ancho de banda son grandes, esta ordenación no es en absoluto eficiente.

Y ése es precisamente el caso de las redes eléctricas, a pesar de que en esas figuras sólo se han mostrado los sistemas que presentaban una banda menor.

Como caso intermedio entre las dos ordenaciones, tenemos la que agrupa los elementos no nulos a lo largo de la diagonal se-

```
        123456789012
 2    1XX
 6    2XXX
 7    3 XXX
 8    4  XXXX
 3    5   XX X
13    6   X X X
 4    7    X X X
12    8     X X X
 5    9      X X
11   10       X XX
10   11          XXX
 9   12            XX

ELEMEN. NO DIAG. =    22
```

```
        123456789012
 2    1XX
 6    2XXX
 7    3 XXX
 8    4  XXXX
 3    5   XXOX
13    6   XOXOX
 4    7    XOXOX
12    8     XOXOX
 5    9      XOXO
11   10       XOXX
10   11          XXX
 9   12            XX

ELEMEN. NO DIAG. =    32
```

fig. 5.29

```
        1234567890123456789012345
  2    1XX
 11    2XXXX
 19    3 XX X
 22    4 X X
 18    5  X XXX
 21    6    XX
 23    7    X XX
 20    8      XXXX
  6    9       XX X
 17   10       X X X
 26   11        X XX
  5   12         XXXXXX
  4   13           XX  X
 13   14           X X  X
 14   15           X  X  X
 24   16            X  XX
 10   17              X XX XX
  9   18               X  X
 12   19                 X X X
 16   20                 X  X XX
  8   21                   X X
  3   22                    X XX
 15   23                    X XX
  7   24                        XX
 25   25                        XX
```

ELEMEN. NO DIAG. = 58

```
        1234567890123456789012345
  2    1XX
 11    2XXXX
 19    3 XXOX
 22    4 XOX0
 18    5  XOXXX
 21    6    XX0
 23    7    XOXX
 20    8      XXXX
  6    9       XX0X
 17   10       XOXOX
 26   11        XOXX
  5   12         XXXXXX
  4   13           XX0OX
 13   14           XOXO0X
 14   15           XOOXO0X
 24   16            X00XX0
 10   17              XOXXOXX
  9   18               X00X00
 12   19                 XOXOX
 16   20                 X00XOXX
  8   21                   XOX00
  3   22                    XOXX
 15   23                    XOXX
  7   24                        XX
 25   25                        XX
```

ELEMEN. NO DIAG. = 112

fig. 5.30

```
                1234567890123456789012345678901234567890
 30    1XX
  2    2XXXXX
230     1    3  XX   X
  3     4  X  X   XX
 25     5  X   X     XX
 39     6   X   X      X
  4     7    X   X      XX
 18     8    X     X       X
 26     9     X     X       XXX
 37    10     X      X
  9    11      X      X        X
  5    12       X      X       XX
 14    13       X       X        XX
 17    14        X       XX        X
 27    15         X      XX
 28    16         X        XX
 29    17         X        XX       X
  8    18          XX       X       X
  6    19          X         X      XX
 13    20           X         X       XX
 15    21           X          XX
 16    22            X         XX       XXX
 38    23                      X        X
  7    24                      XX       X
 11    25                      X        XXX
 10    26                       X       XX      X
 12    27                       X       X X
 19    28                         X       X    XX
 21    29                         X         X     X
 24    30                         X          X     X
 32    31                                X      X
 20    32                                X     X    X
 33    33                                X      X
 22    34                                 X       XX  X
 23    35                                  X      XX   X
 34    36                                         X     X
 35    37                                          X   X
 36    38                                           X   X
```

ELEMEN. NO DIAG. = 90

```
                1234567890123456789012345678901234567890
 30    1XX
  2    2XXXXX
  1    3  XX00X
  3    4  X0X00XX
 25    5  X00X000XX
 39    6   X00X0000X
  4    7    X00X0000XX
 18    8    X000X00000X
 26    9     X000X00000XXX
 37   10     X0000X0000000
  9   11      X0000X000000X
  5   12       X0000X00000XX
 14   13       X00000X000000XX
 17   14        X00000XX000000X
 27   15         X0000XX0000000
 28   16         X000000XX00000
 29   17         X000000XX00000X
  8   18           XX00000X00000X
  6   19            X000000X0000XX
 13   20             X000000X00000XX
 15   21             X0000000XX00000
 16   22              X000000XX00000XXX
 38   23                  X00000X0000000
  7   24                   XX0000X000000
 11   25                    X00000XXX000
 10   26                     X0000XX0000X
 12   27                     X0000X0X0000
 19   28                       X00000X000XX
 21   29                       X000000X0000X
 24   30                       X0000000X0000X
 32   31                            X0000X0000
 20   32                              X000X000X
 33   33                              X0000X000
 22   34                               X0000XX0X
 23   35                                X000XX00X
 34   36                                  X000X00
 35   37                                    X00X0
 36   38                                     X00X
```

ELEMEN. NO DIAG. = 384

fig. 5.31

cundaria.

En las figuras 5.32, 5.33 y 5.34 aparecen ordenadas por este procedimiento las redes de 29, 30 y 39 nudos respectivamente. Se observa claramente que los nuevos elementos aparecen fundamentalmente en la parte inferior derecha.

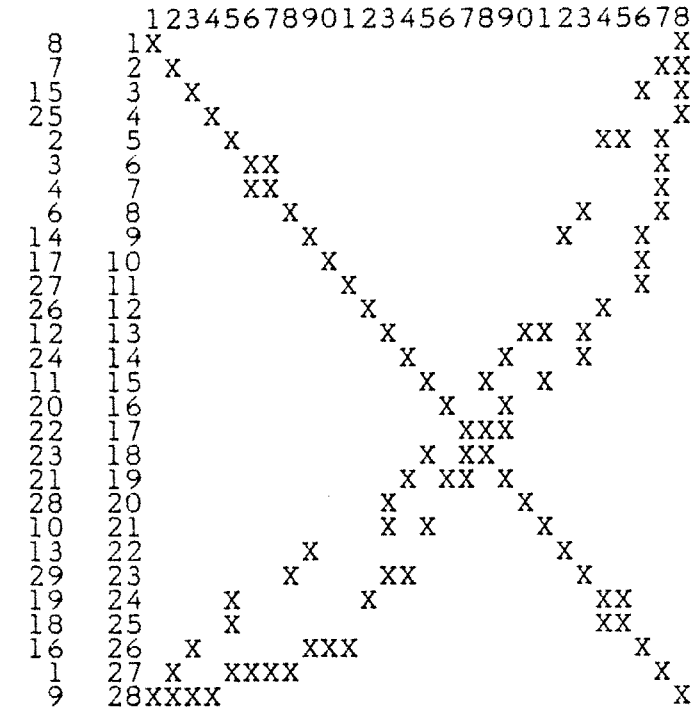
Aunque el esquema 2 de Tinney es el más complejo de programar, en comparación con los otros dos procedimientos, y el que más tiempo requiere, el ahorro posterior es tal que justifica sobradamente el tiempo invertido en la ordenación previa.

Se han implementado también los esquemas 1 y 3 de Tinney. El esquema 3 no introduce mejoras sustanciales frente al 2, y sin embargo complica la programación. El esquema 1 es deseable por su simplicidad, aproximadamente la misma que la de una ordenación en banda, pero sus resultados son peores que los del esquema 2, especialmente cuando el sistema es grande, a juzgar por los datos de la figura 5.35, donde se muestran los elementos de B' antes de triangularizarla, y después, para los esquemas 1 y 2 de Tinney.

De los algoritmos descritos en este capítulo, el de Newton se programó utilizando el esquema 3 de Tinney, y el resto el esquema 2.

Finalmente, se puede observar en la figura 5.23, que en la red de 13 nudos no aparecen elementos nuevos, cuando se ordena por el esquema 2. Esta red es un árbol si prescindimos del nudo de referencia, y dicho esquema tiene la propiedad de ordenar óp-

timamente cualquier árbol, no así el esquema 1.


1234567890123456789012345678
ELEMEN. NO DIAG. = 60

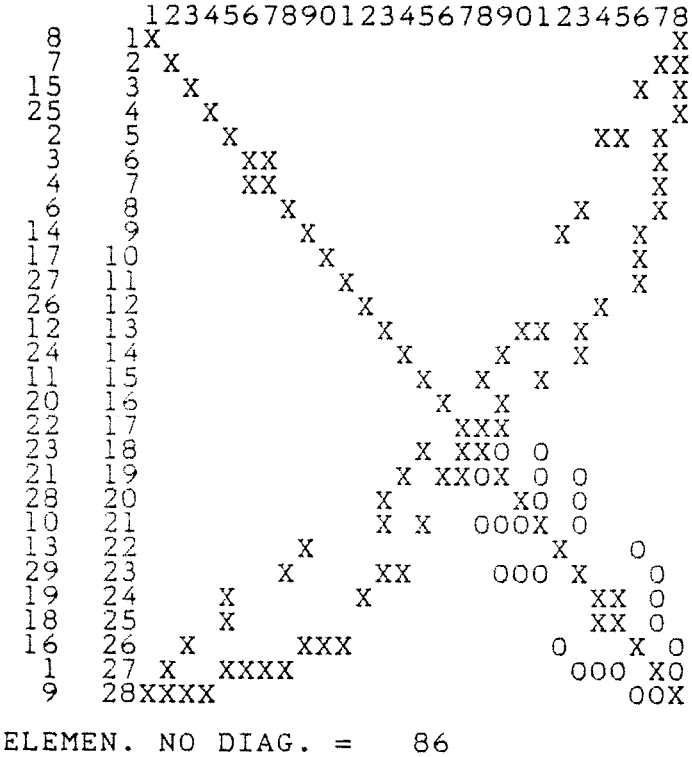
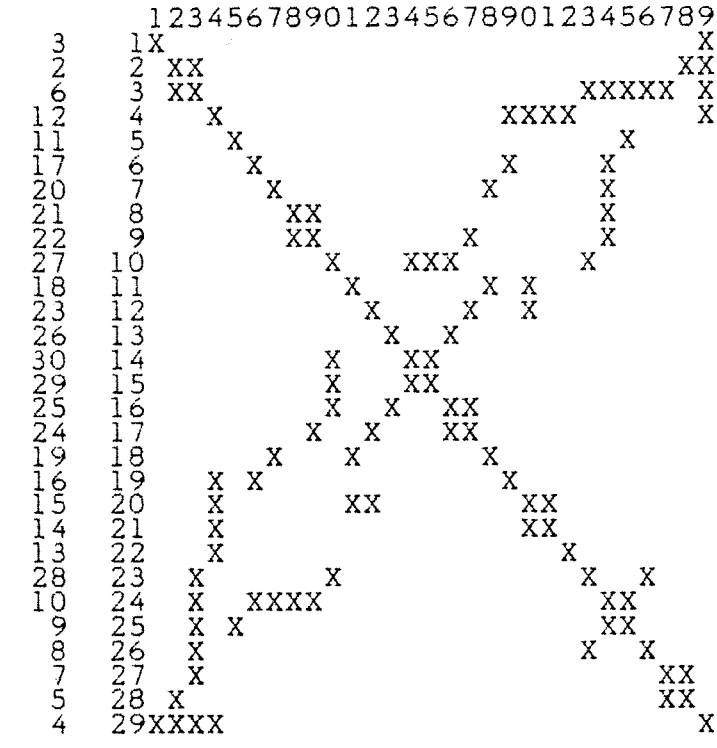
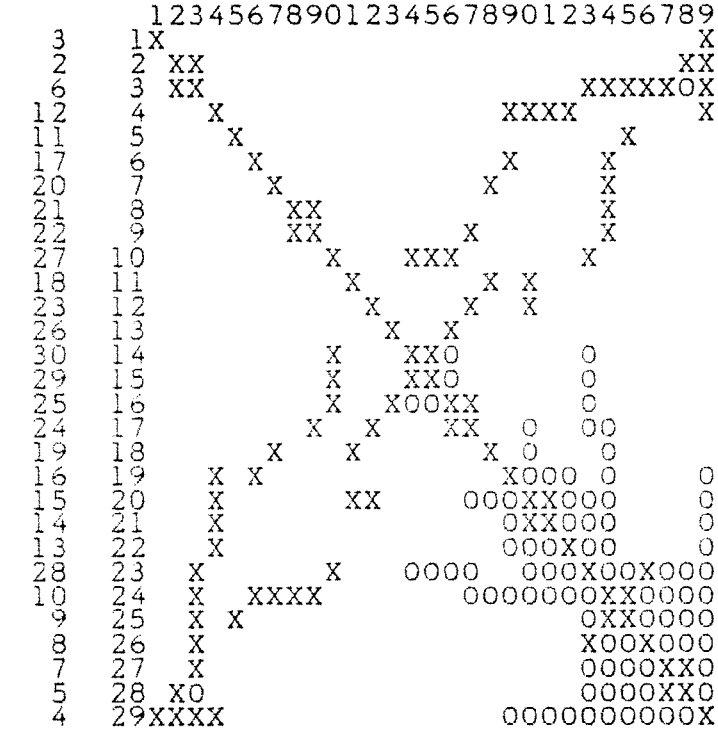

1234567890123456789012345678
ELEMEN. NO DIAG. = 86

fig. 5.32

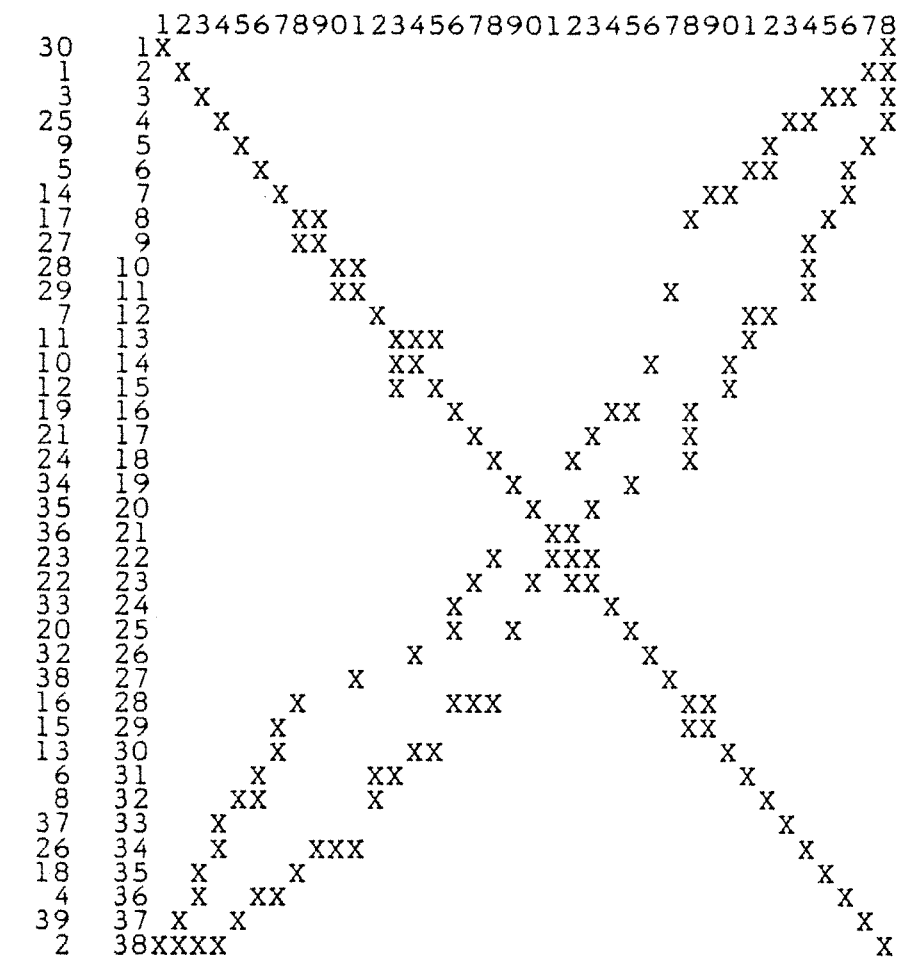


ELEMEN. NO DIAG. = 78

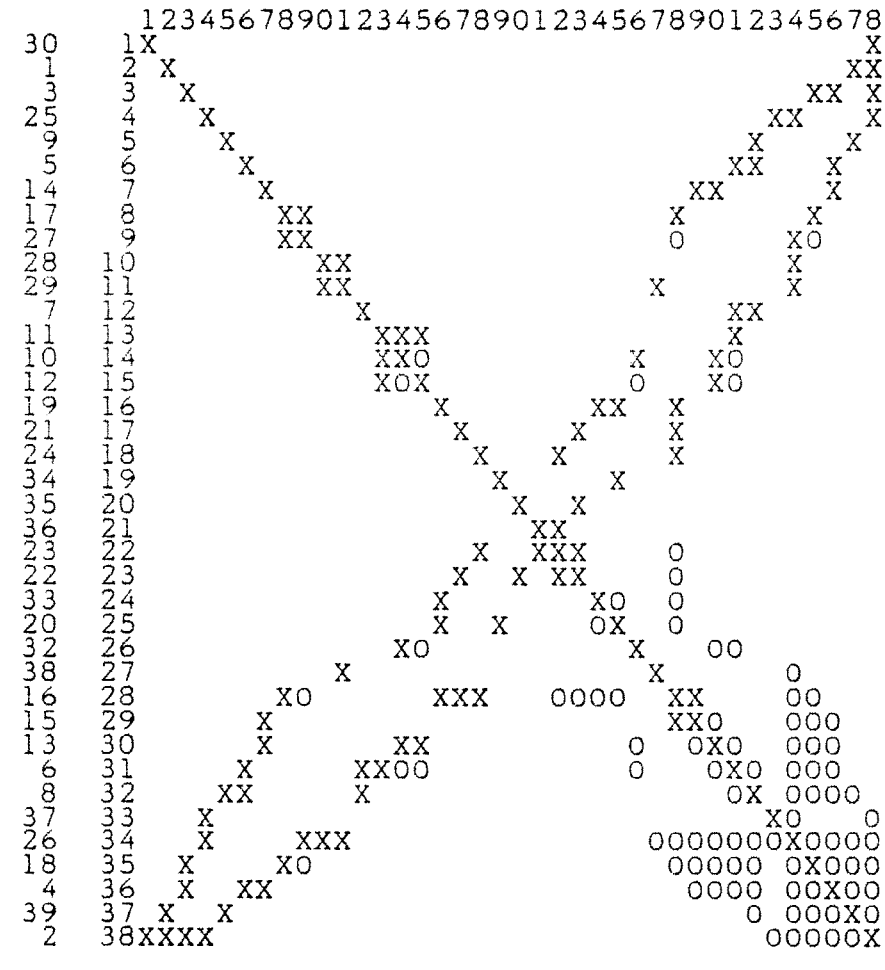


ELEMEN. NO DIAG. = 168

fig. 5.33



ELEMEN. NO DIAG. = 90



ELEMEN. NO DIAG. = 178

fig. 5.34

N°de Elementos de B'

Nudos / Elementos	13	14	26	29	30	38	39	57	118
Sin Triang.	22	36	52	60	78	86	90	148	346
Esquema 2	22	44	60	74	104	102	132	258	506
Esquema 1	24	46	68	80	110	118	138	262	672

fig. 5.35

6. ALGORITMO PROPUESTO PARA LA SOLUCION EN PARALELO.

6.1 INTRODUCCION Y MOTIVACION.

En los últimos años se ha detectado un enorme interés en la solución en paralelo de sistemas de ecuaciones lineales.

Wing y Huang [123] han mostrado desde un punto de vista téorico, que se dispone de suficiente paralelismo en la solución de sistemas de ecuaciones dispersas, como para lograr un considerable aumento en la velocidad. Si se pudiese conseguir en la práctica ese nivel de funcionamiento, se mejorarían en gran medida los métodos actuales.

Los procesadores del tipo instrucción simple datos múltiples (SIMD), se han utilizado para resolver el flujo de cargas [82, 55]. El paralelismo se logra a nivel de la instrucción, pudiendo ser incluso transparente al usuario si se dispone de un compilador de alto nivel. Exigen disponer las operaciones en forma de vectores, para aprovechar su estructura interna [55], lo que en numerosas ocasiones les hace poco atractivos cuando se trabaja con matrices vacías.

La otra alternativa, los procesadores de instrucciones múltiples-datos múltiples (MIMD), permiten realizar un paralelismo en el sentido más estricto de la palabra. Tal es el caso de una estructura de multiprocesador, donde varios procesadores están conectados por uno o varios buses.

A esta última categoría pertenece el trabajo que se va a describir en este capítulo, donde parece que se está concentrando el máximo esfuerzo.

Hasta la fecha, los algoritmos propuestos se encuadran en dos grupos. Uno, que podemos denominar esquemas en bloque, particiona a la matriz de forma que cada bloque está relacionado con un procesador [118]. El otro, que llamaremos esquemas elementales, tiene en cuenta el paralelismo que existe al considerar los elementos individuales de la matriz. Si se observa cuidadosamente la secuencia de operaciones elementales que tienen lugar en la triangularización de la matriz, se verá que se puede establecer una relación de orden parcial entre ellas, de forma que en un mismo instante se podrán estar realizando varias simultáneamente, aunque este número será variable a lo largo del tiempo, dependiendo de la estructura de la matriz [123].

En el segundo caso, el número de procesadores suele ser mayor, y es precisa una reasignación constante de recursos a tareas. En determinados momentos sobrarán recursos y en otros faltarán.

El paralelismo por bloques, no plantea problemas de reasignación de tareas, que está resuelto desde el momento en que cada procesador se dedica a una zona en concreto de la matriz.

Dado que el número de procesadores disponibles no es alto, y que el paralelismo elemental presenta todavia ciertos problemas, que hacen que los resultados publicados sean casi siempre de si-

mulación, se optó por investigar el paralelismo por bloques, que permitiría más probablemente su realización física, como así ha sido.

La orientación de la tesis hacia estos derroteros, también vino motivada por el hecho de que no existe todavia un algoritmo realmente eficiente para la partición de matrices. Usualmente, los algoritmos de solución en paralelo de sistemas de ecuaciones, dan por descontado que se conoce una partición de la matriz en bloques. En este sentido se remite al lector al párrafo 4.5.

6.2 ALGORITMO DE DESCOMPOSICION DE LA MATRIZ.

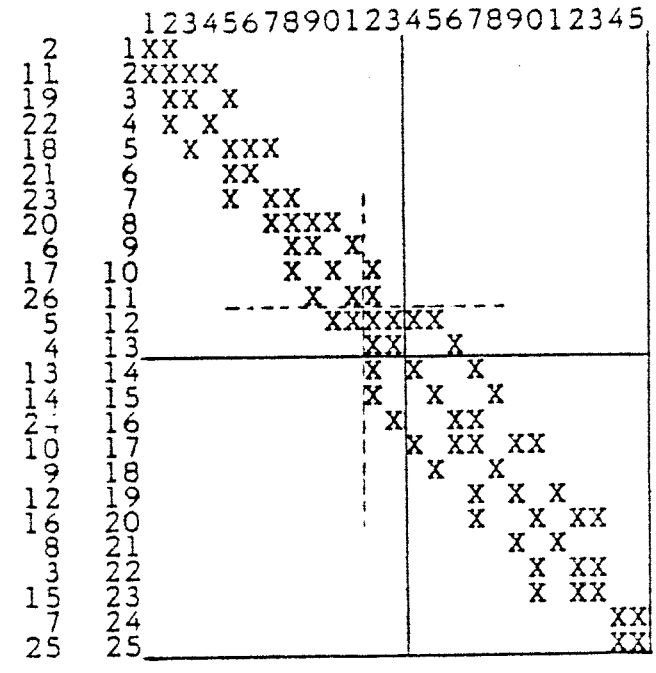
La idea en la que se basa este algoritmo heurístico es muy simple, y surgió en el transcurso de las numerosas pruebas que se hicieron con la ordenación de las matrices.

Inicialmente, se intentaba descomponer una matriz simétrica en la forma:

$$A = \begin{pmatrix} D & G \\ B & E \end{pmatrix}$$

donde D y E son matrices cuadradas. No se buscaba resolver en paralelo el problema Ax=b, sino que, como se indica en el epígrafe 3.6, se intentaba ahorrar operaciones y memoria resolviendo el sistema por bloques, lo cual se consigue bajo ciertas condiciones, si G y B son muy vacías.

Intuitivamente, se pensó que G y B serían muy vacías si ordenábamos A en forma de banda. La figura 6.1 muestra esta idea para la matriz de la red de 26 nudos:



ELEMEN. NO DIAG. = 58

fig.6.1

D	G
B	E

Con los bloques señalados, resulta que B y G son realmente vacías (sólo tienen 3 elementos). El siguiente paso fue inmediato, al darse cuenta de que, si hacemos que D tenga dos filas menos, en beneficio de E, como indican las líneas de trazos, entonces B y G son aún más vacías (sólo 2 elementos).

Por tanto, lo que se hace por este procedimiento, es buscar, para una matriz ordenada en banda un punto de corte próximo a la mitad, pero no forzosamente en la mitad, que minimice el número de elementos que quedan en los bloques no diagonales.

Pero este algoritmo se puede utilizar de un modo bien diferente, si se presta atención a la figura 6.1. En efecto, en lugar

de minimizar el número de elementos de los bloques G y B, se puede minimizar, prácticamente con el mismo esfuerzo, el número de nudos de D que tienen conexión con E. Dichos mínimos no tienen porqué coincidir, como se observa en la anterior figura. El mínimo de elementos en G y B se consigue partiendo las matrices entre los nudos 11 y 12 (línea de trazos), mientras que el mínimo de nudos de D que son frontera con E se consigue partiendo la matriz entre los nudos 12 y 13 (justamente entre las dos líneas). En ese caso, sólo el nudo 12 queda como nexo de unión entre los bloques D y E.

Si ahora eliminamos el nudo frontera 12, la matriz original quedará descompuesta en dos submatrices, que serán aproximadamente iguales si no nos hemos alejado mucho del centro en la búsqueda del mínimo.

Este logro hizo desviarse la atención de los trabajos iniciales, para concentrarse en la explotación del algoritmo para la resolución en paralelo del sistema de ecuaciones Ax=b, como se explica en el epígrafe siguiente.

Se observa cierta analogía de este procedimiento con los descritos en [75, 90], pero partiendo de unos presupuestos totalmente diferentes, y de una forma mucho más simple, sin necesidad además de que existan "clusters" naturales (véase 4.5). El algoritmo de Ogbuobiri et al. [75] es muy costoso en tiempo. El de Sangiovanni-Vincentelli et al. [90], una simplificación del anterior, es más rápido, pero necesita refinamientos especiales para no perder eficacia en el resultado final. En ambos pueden identi-

ficarse un número excesivo de subgrupos muy pequeños, que obligaría a regruparlos si no se admiten tantos.

El algoritmo definitivo, descrito más formalmente, es como sigue:

1) Ordenar la matriz A en forma de banda. Para ello no es preciso seguir un algoritmo complejo, como puede ser el de Cuthill-Mckee [23] o el de Gibbs et al [44], ya que no se exige estrictamente una banda estrecha. Hemos utilizado una versión sencilla y más rápida, a saber:

1.a) Elegir un nudo de grado mínimo, que se numerará el primero.

1.b) Numerar, consecutiva y arbitrariamente, los nudos adyacentes al que tenga el número más bajo de los ya numerados, hasta agotarlos todos.

El algoritmo de Cuthill-Mckee pone más cuidado en la elección del primer nudo (nudo periférico), y para minimizar la banda, impone un orden en la numeración de los nudos adyacentes a los ya numerados, eligiendo primero los de menor grado, en lugar de hacerlo arbitrariamente. Esto introduce una complicación, que sin embargo en nuestro caso no mejora el resultado final, a juzgar por las pruebas realizadas.

2) Calcular, para cada fila i, el cardinal del conjunto:

$$C(i) = \left\{ K<i \ : \ \exists \ a_{kj} \neq 0, \ j \geq i \right\}_{i = m \ldots M}$$

Es decir, para todas las filas comprendidas entre m y M, se calcula un índice $|C(i)|$, que indica el número de nudos anteriores a i, que tienen alguna conexión con nudos posteriores a i, o con el propio i. Las constantes m y M determinan el margen de búsqueda, alrededor de las posiciones centrales de la matriz, del cuello de botella.

3) Determinar la fila i_o con $|C(i)|$ mínimo. Si existe empate se toma la más proxima al centro.

4) Los nudos pertenecientes a $C(i_o)$, son precisamente los de la interconexión, y se numeran arbitrariamente los últimos.

5) Los nudos anteriores a i_o, salvo los del apartado 4, forman una submatriz, y los posteriores a i_o, incluido él mismo, forman la otra, que estarán conectadas sólo por los nudos de $C(i_o)$. Los nudos de estas dos submatrices se numeran finalmente siguiendo el esquema 2 de Tinney, con la única restricción de que se deben mantener dentro de su bloque respectivo.

Es importante dar un valor adecuado a m y M. Un valor muy pequeño para m y muy grande para M, puede dar lugar a dos submatrices de tamaño dispar. Por el contrario, un valor muy igualado entre m y M (alrededor de N/2) logrará dos submatrices muy igualadas, pero puede producir un cuello de botella excesivo. Para la aplicación que haremos del algoritmo, es preferible que ocurra lo primero, dentro de unos límites razonables. Para ello suele ser suficiente con tomar

$$m = 0.5\ N$$

$$M = 0.6\ N$$

aunque el rango se puede ampliar si N es grande.

Las figuras 6.2 a 6.7 muestran cómo descompone este algoritmo las matrices B' de las redes que hemos venido estudiando.

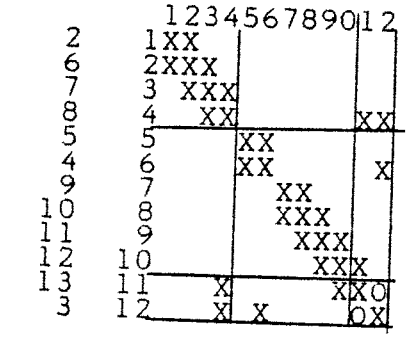
En las figuras 6.8 a 6.13 se pueden ver los grafos de los sistemas más pequeños, señalándose los nudos que constituyen la interconexión. El nudo de referencia y sus conexiones se han mostrado a trazos, indicando que no se deben tener en cuenta en lo tocante a B', aunque lógicamente sí en la matriz de admitancias de nudos.

6.3 ALGORITMO DE SOLUCION EN PARALELO.

Se describirá el algoritmo utilizado para resolver en paralelo el problema del reparto de cargas, por el método desacoplado rápido. Como diagrama de bloques sigue siendo válido el de la figura 5.6, aclarándose aquí cómo se pueden ejecutar en paralelo los distintos apartados de que consta.

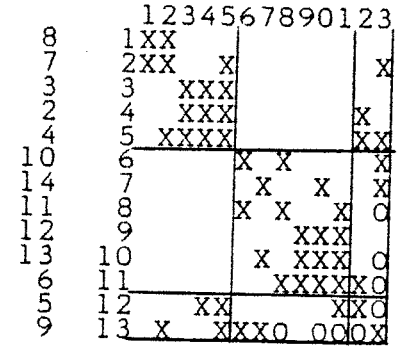
Para la ejecución de los algoritmos, se utilizarán dos procesadores, que denominaremos desde ahora P1 y P2, siendo P2 el que llevará la principal responsabilidad, y el que dispone de los datos del sistema.

BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 5 11



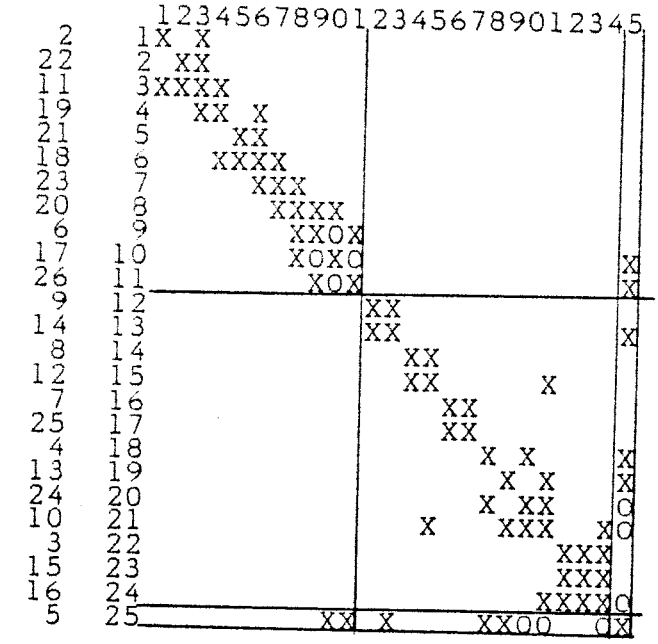
ELEMEN. NO DIAG. = 24

BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 6 12



ELEMEN. NO DIAG. = 44

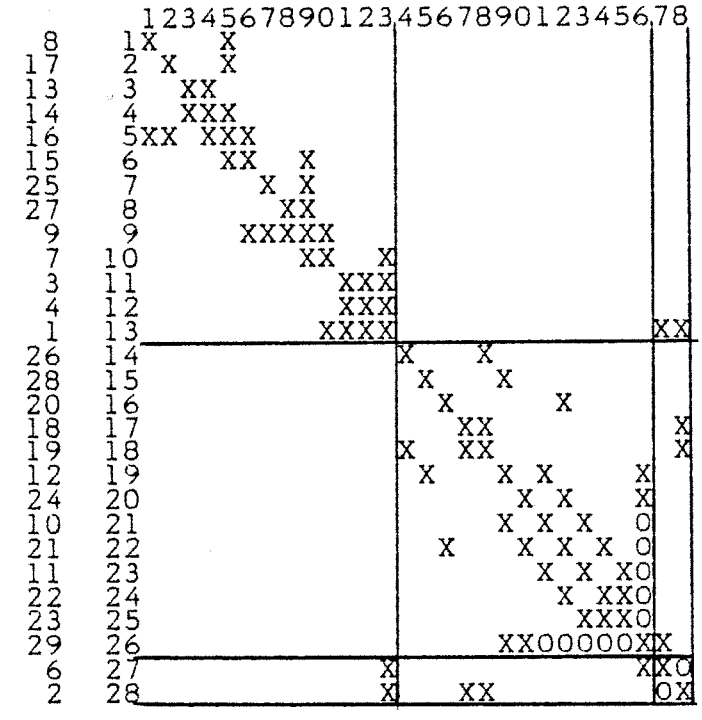
BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 12 25



ELEMEN. NO DIAG. = 62

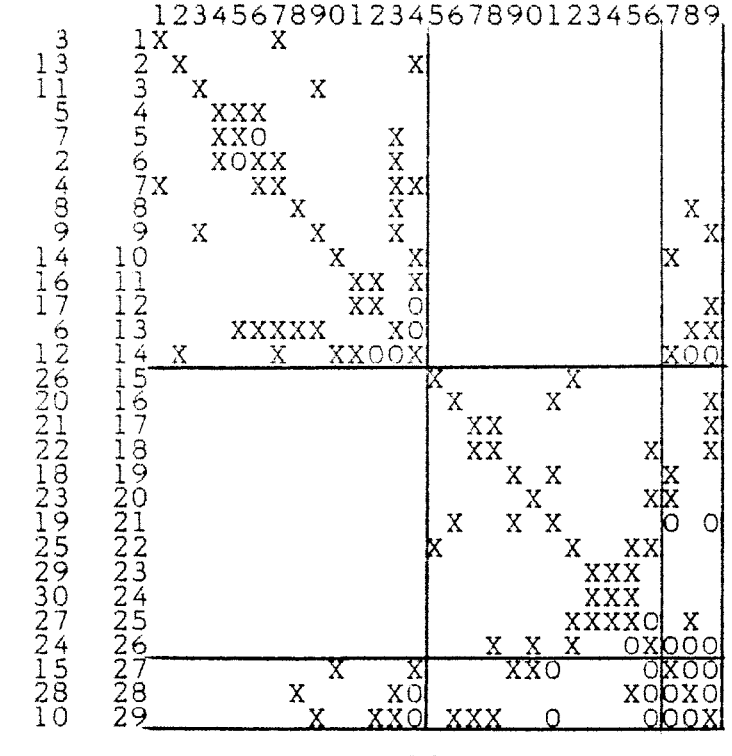
fig. 6.2

BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 14 27



ELEMEN. NO DIAG. = 72

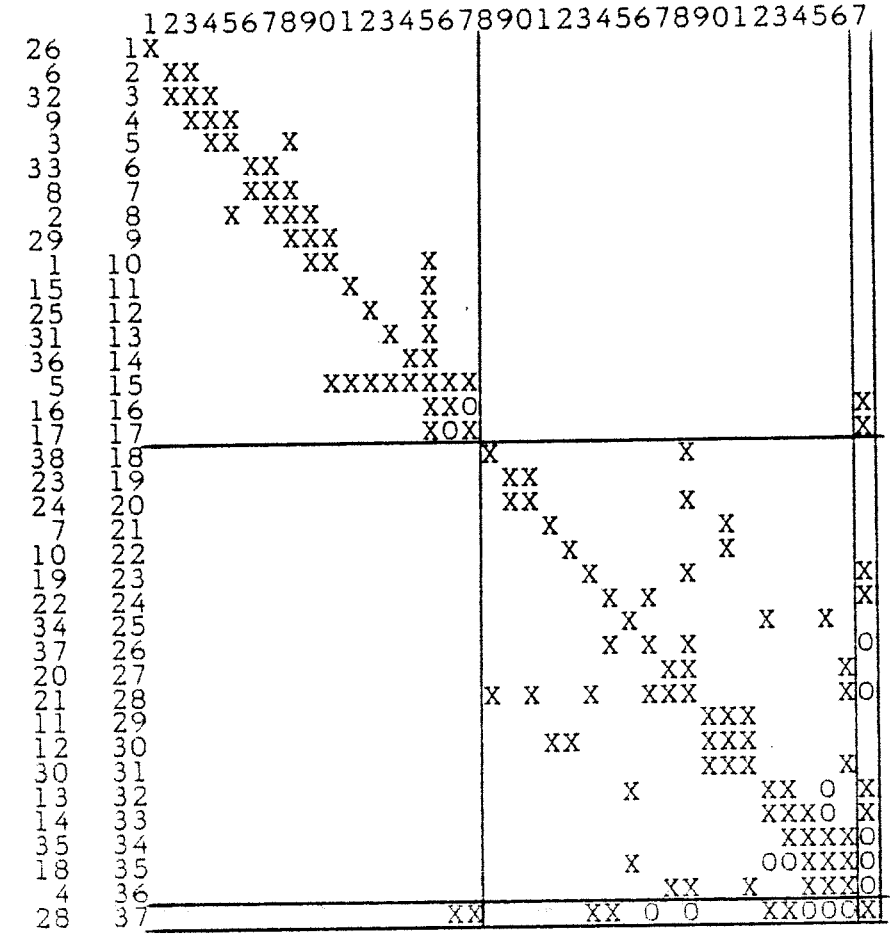
BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 15 27



ELEMEN. NO DIAG. = 106

fig. 6.3

BUSQUEDA ENTRE 50 Y 60%
LIMITES SUBMATRICES: 18 37

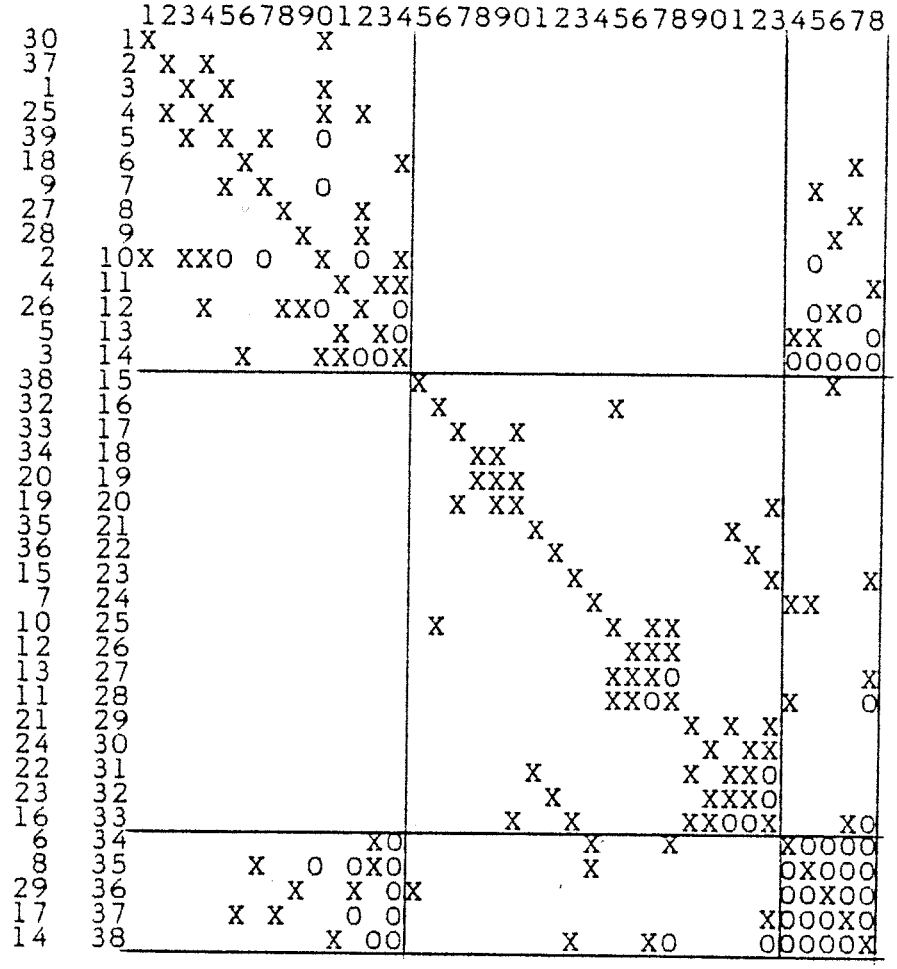


ELEMEN. NO DIAG. = 102

fig. 6.4

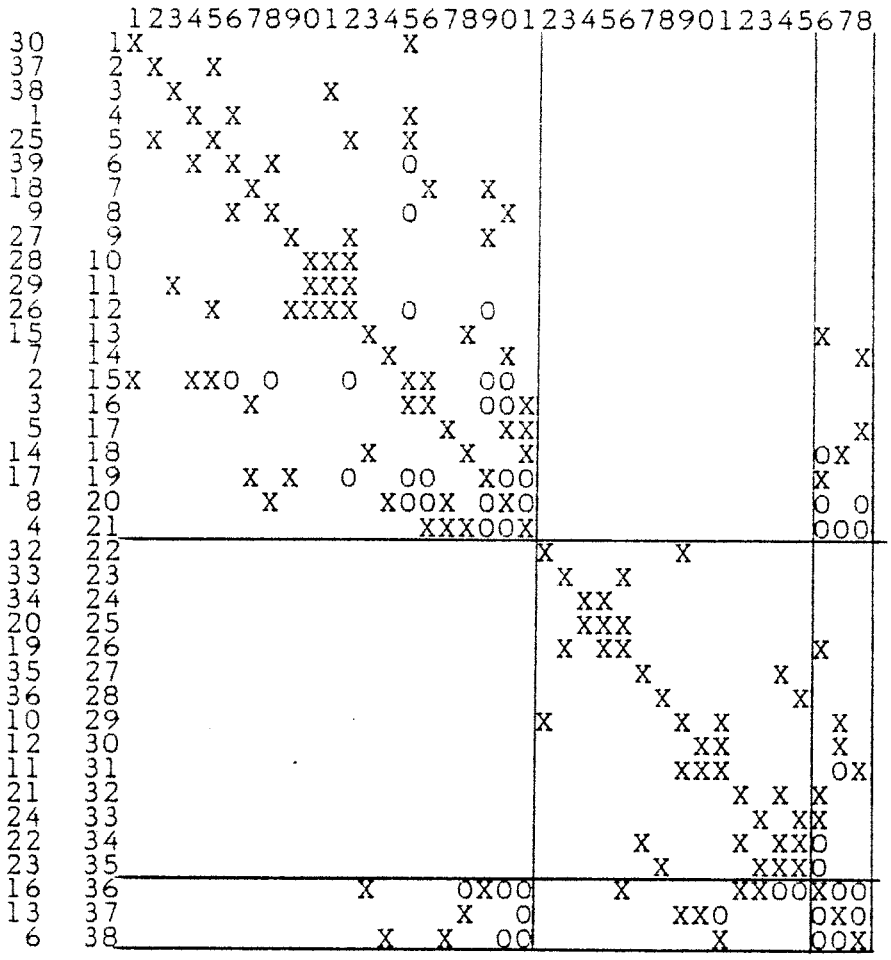
LIMITES SUBMATRICES: 15 34

248



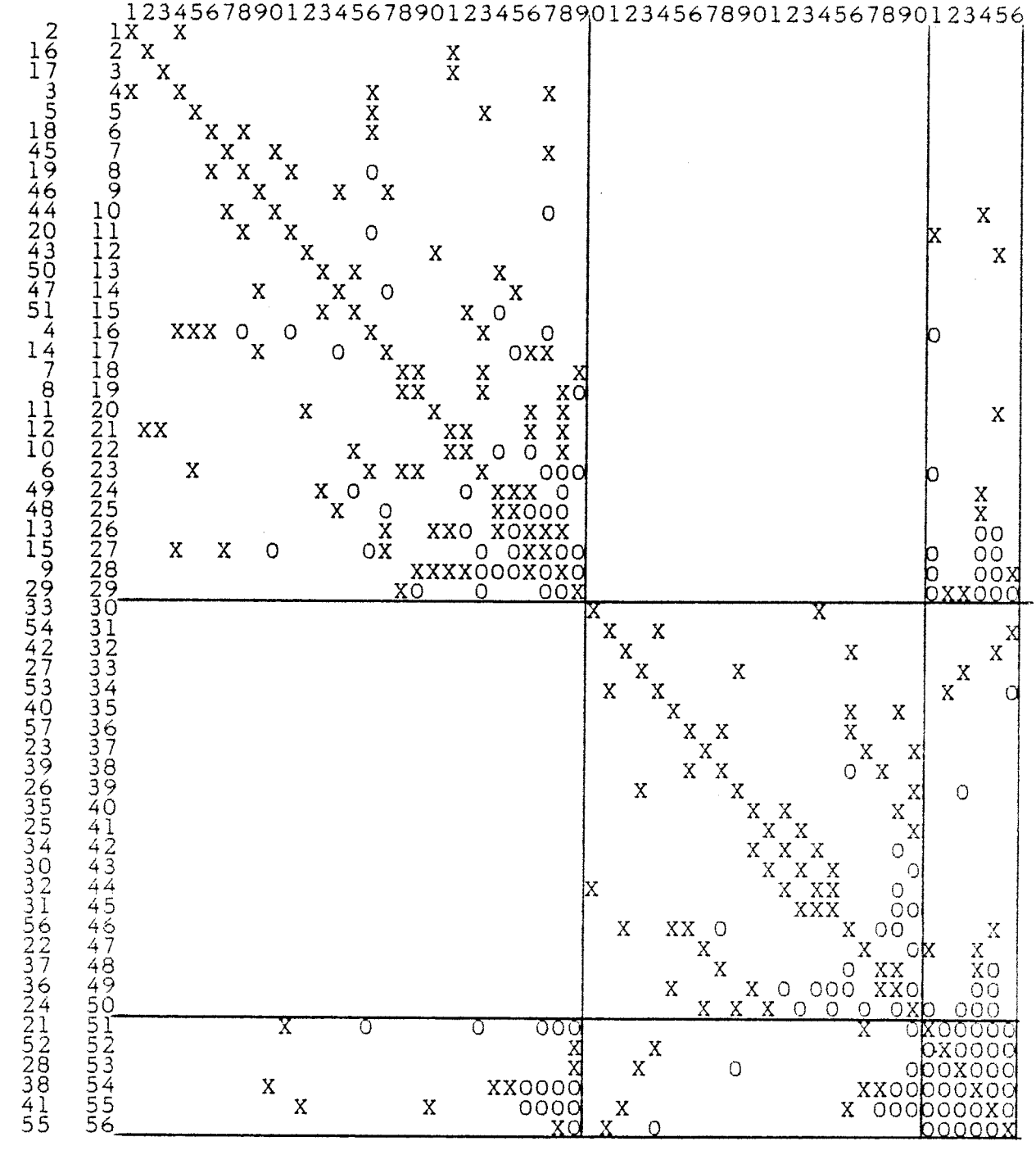
ELEMEN. NO DIAG. = 148

BUSQUEDA ENTRE 39 Y 66%
LIMITES SUBMATRICES: 22 36



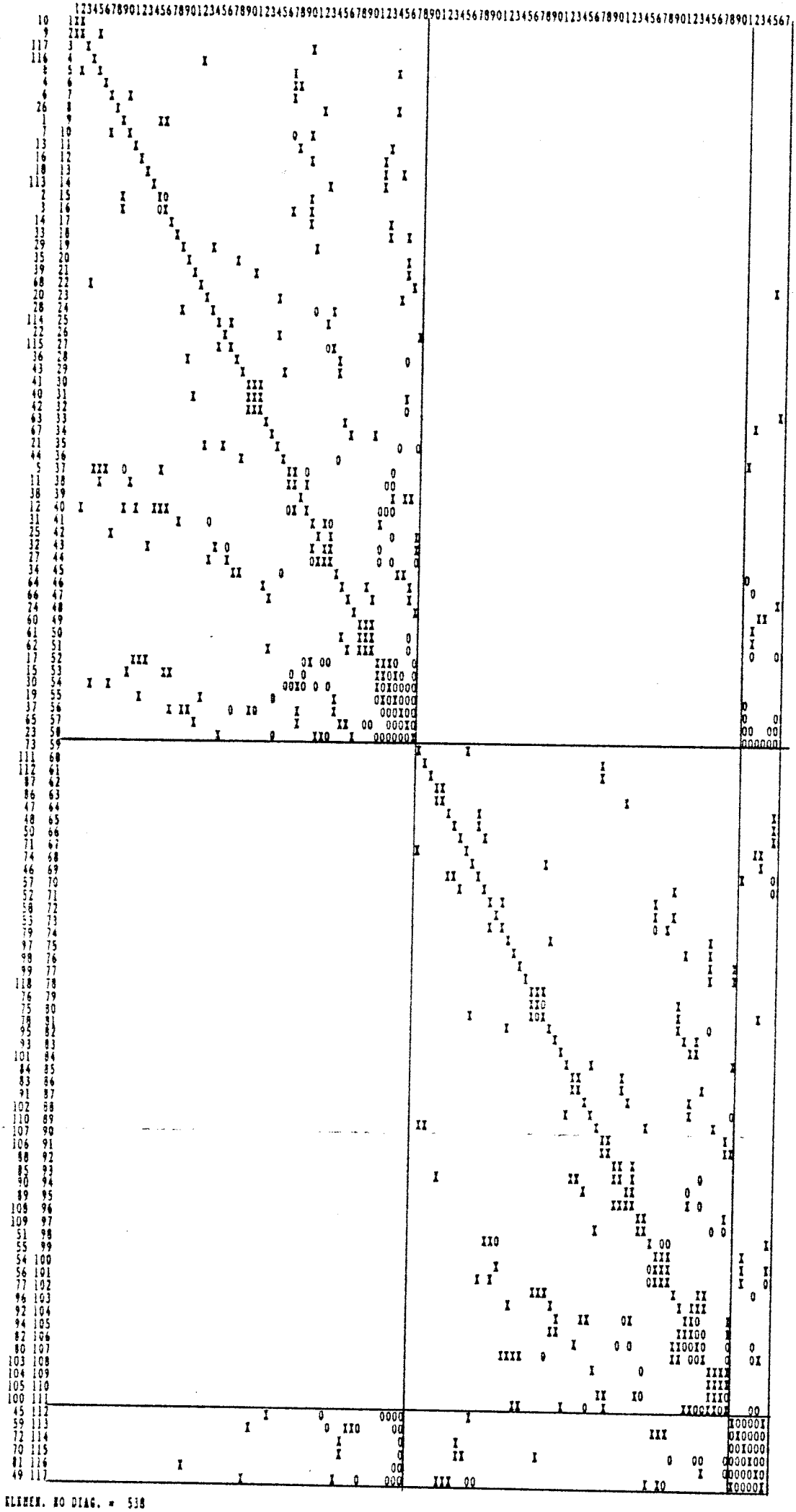
ELEMEN. NO DIAG. = 136

BUSQUEDA ENTRE 39 Y 66%
LIMITES SUBMATRICES: 30 51



ELEMEN. NO DIAG. = 284

fig. 6.6



ELEMEN. NO DIAG. = 538

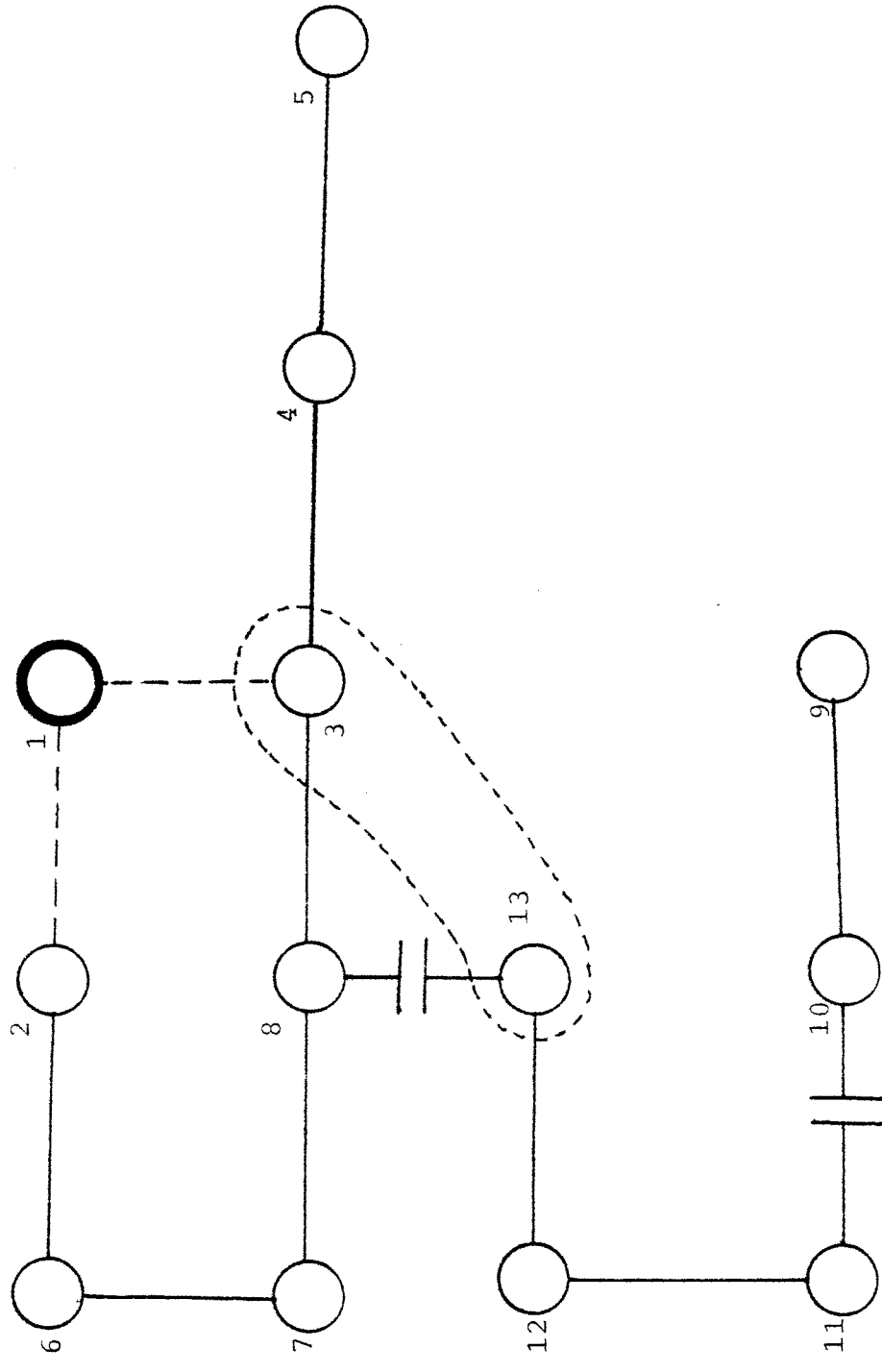


fig. 6.8

14 Nudos (IEEE)

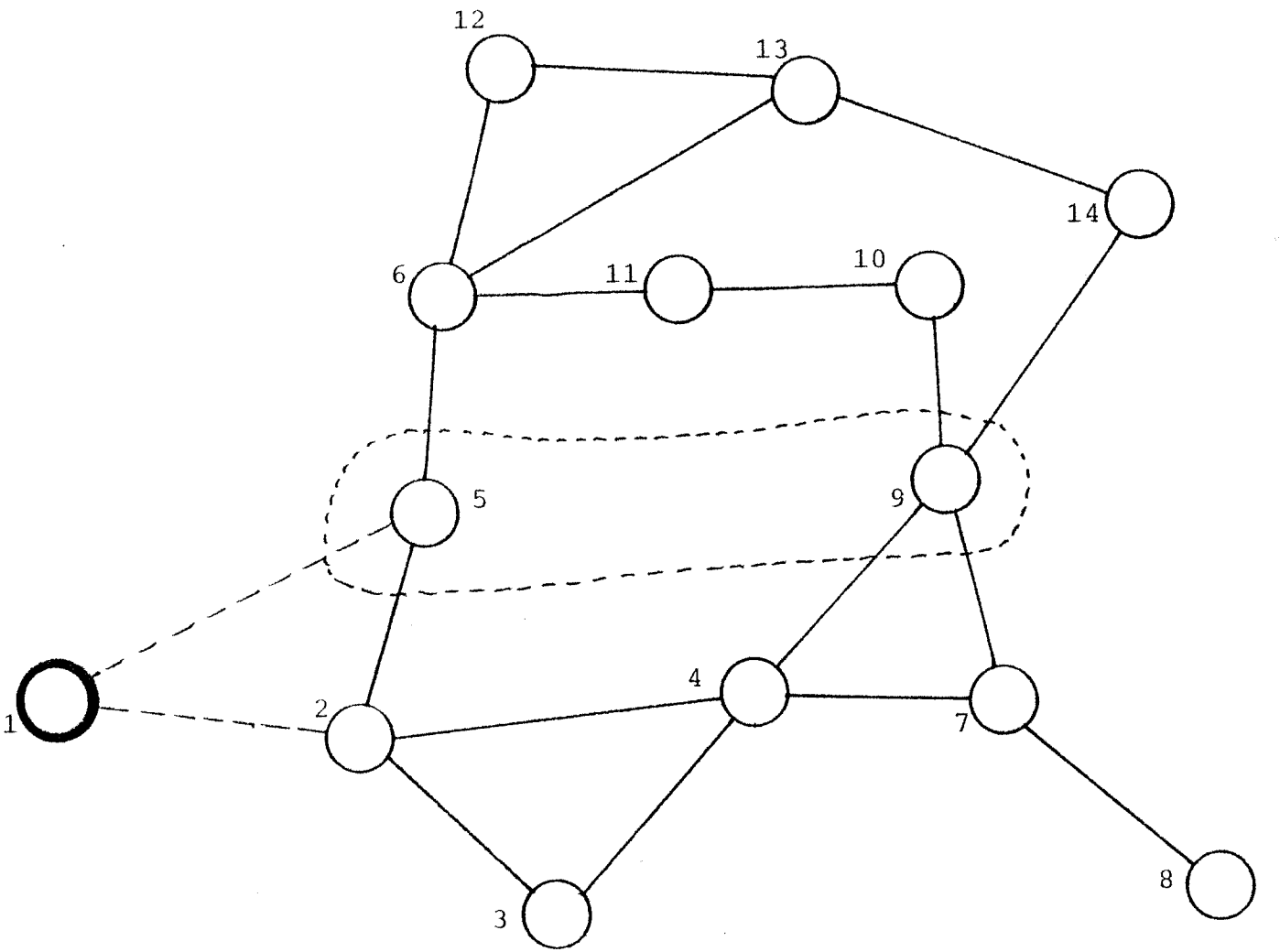


fig. 6.9

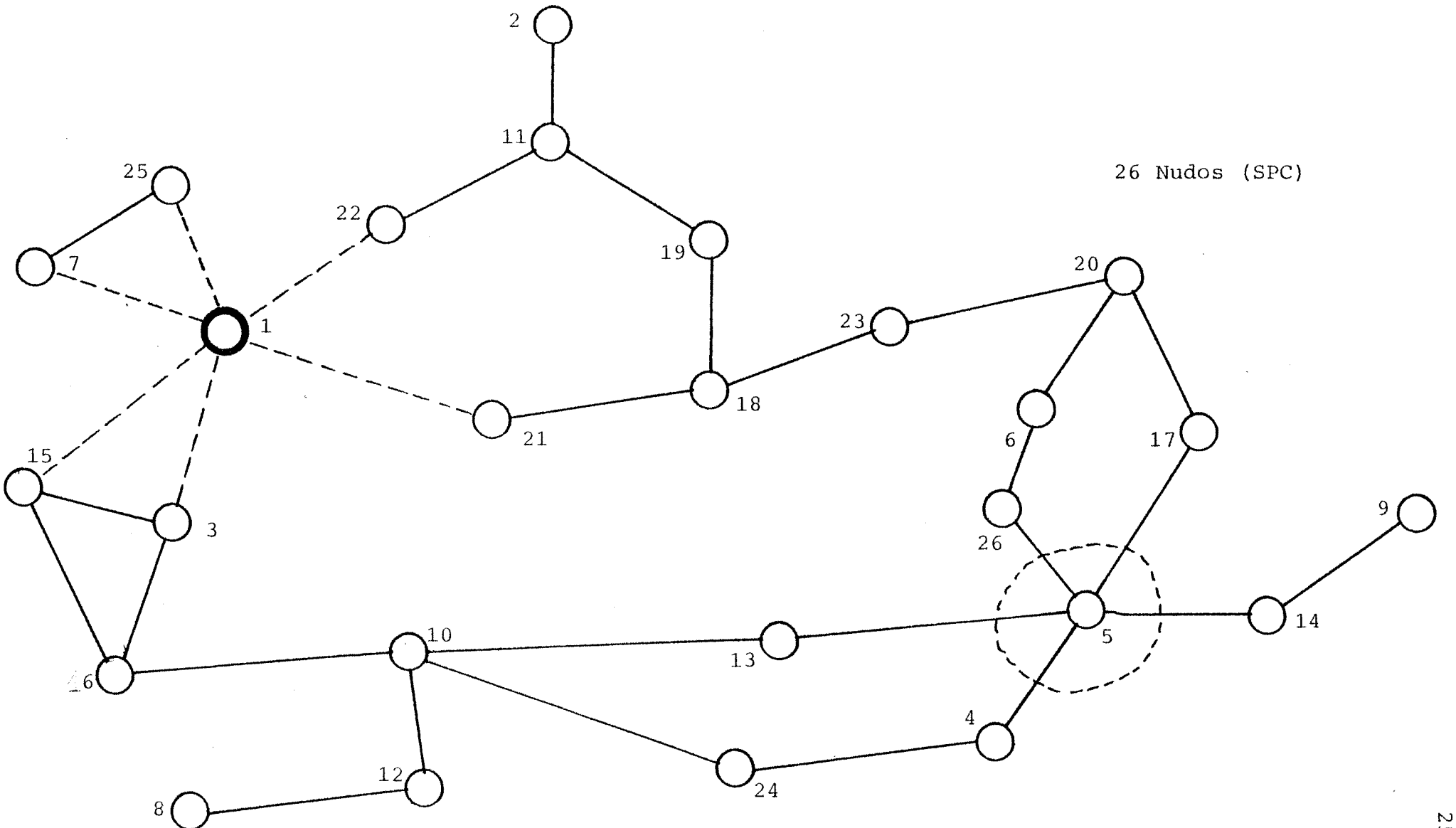
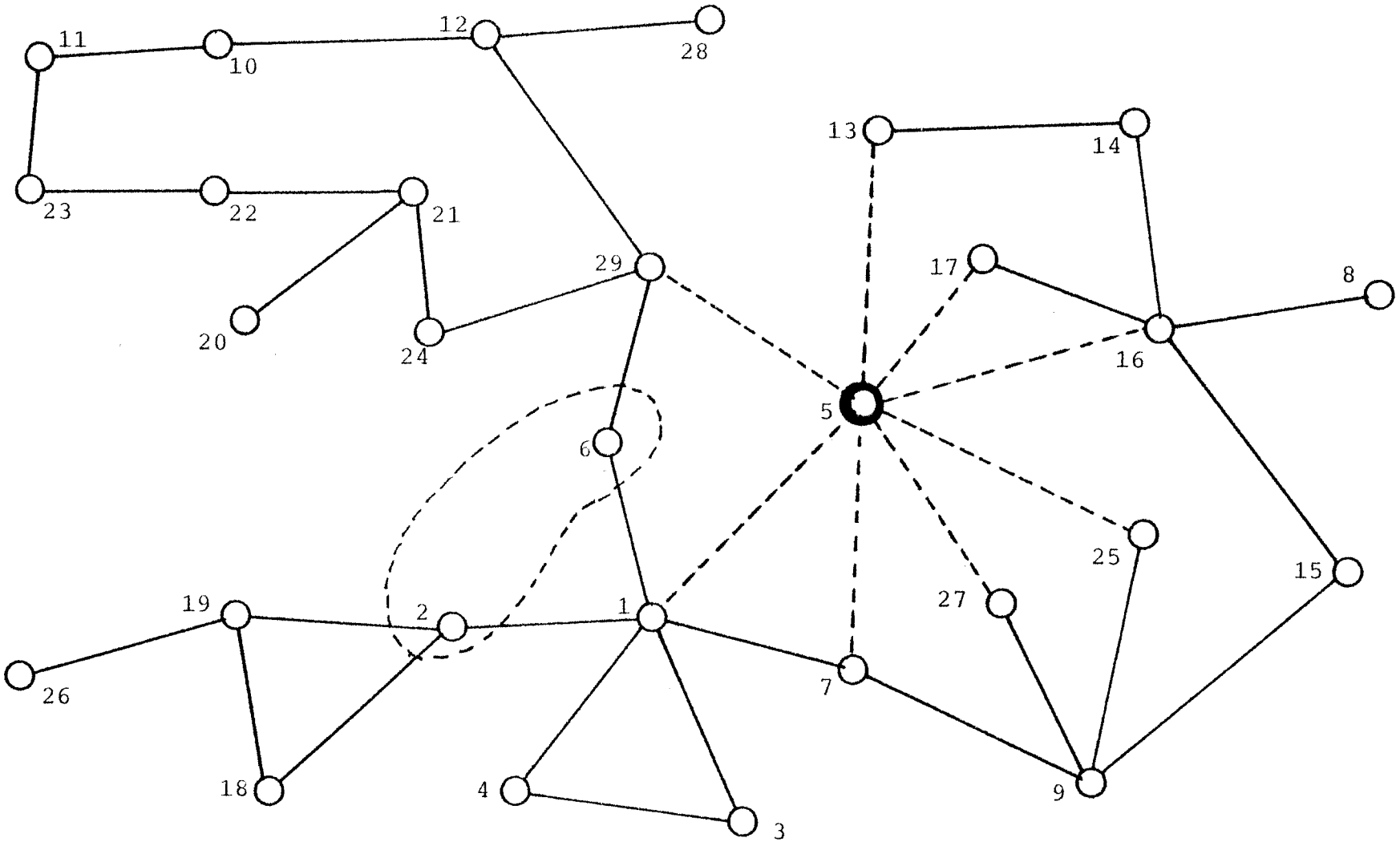

26 Nudos (SPC)
2
11
25
22
19
7
1
20
23
18
21
6
17
15
3
26
9
5
14
10
13
16
4
12
24
8

fig. 6.10



29 Nudos (Málaga)

fig. 6.11

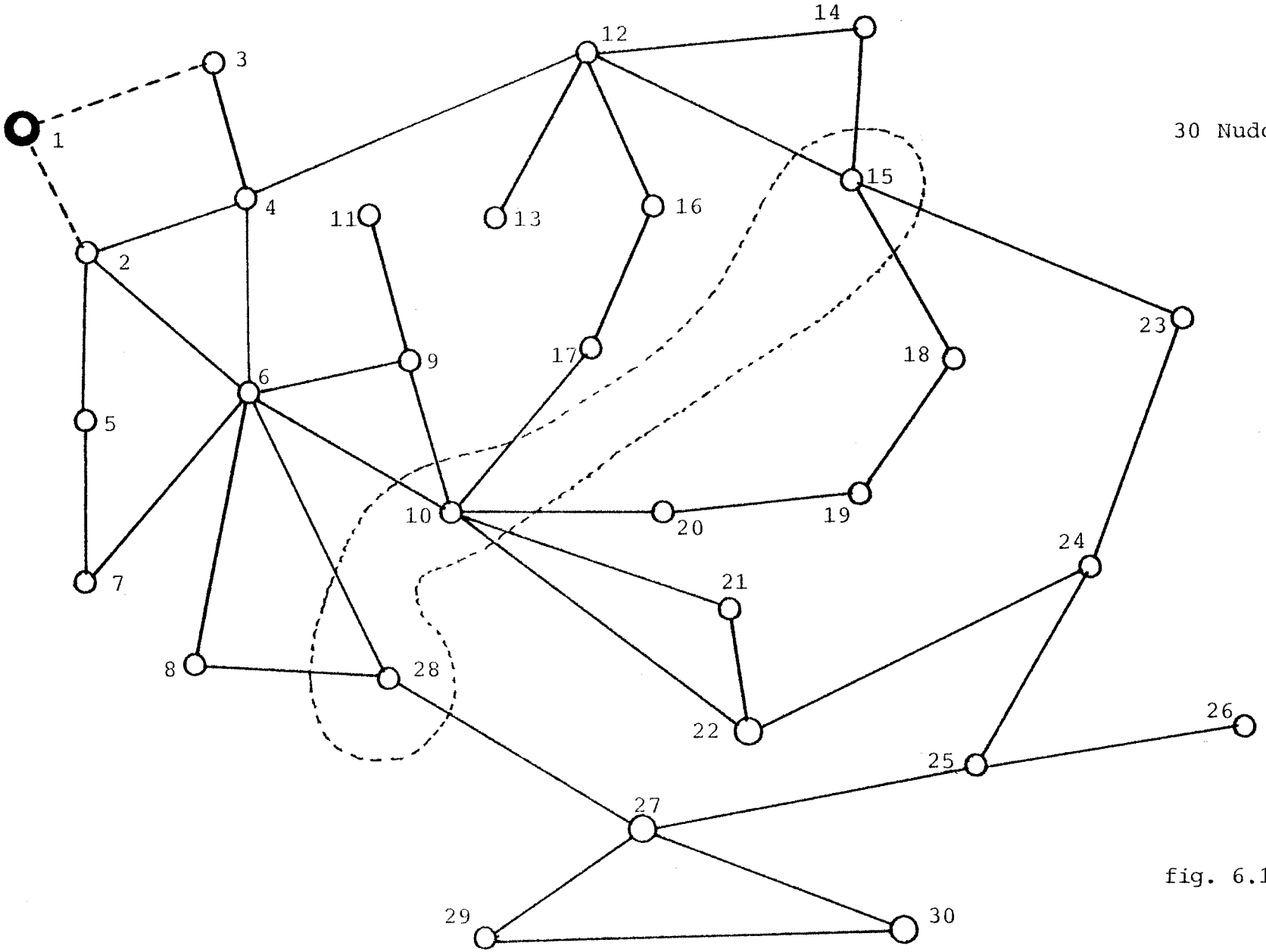
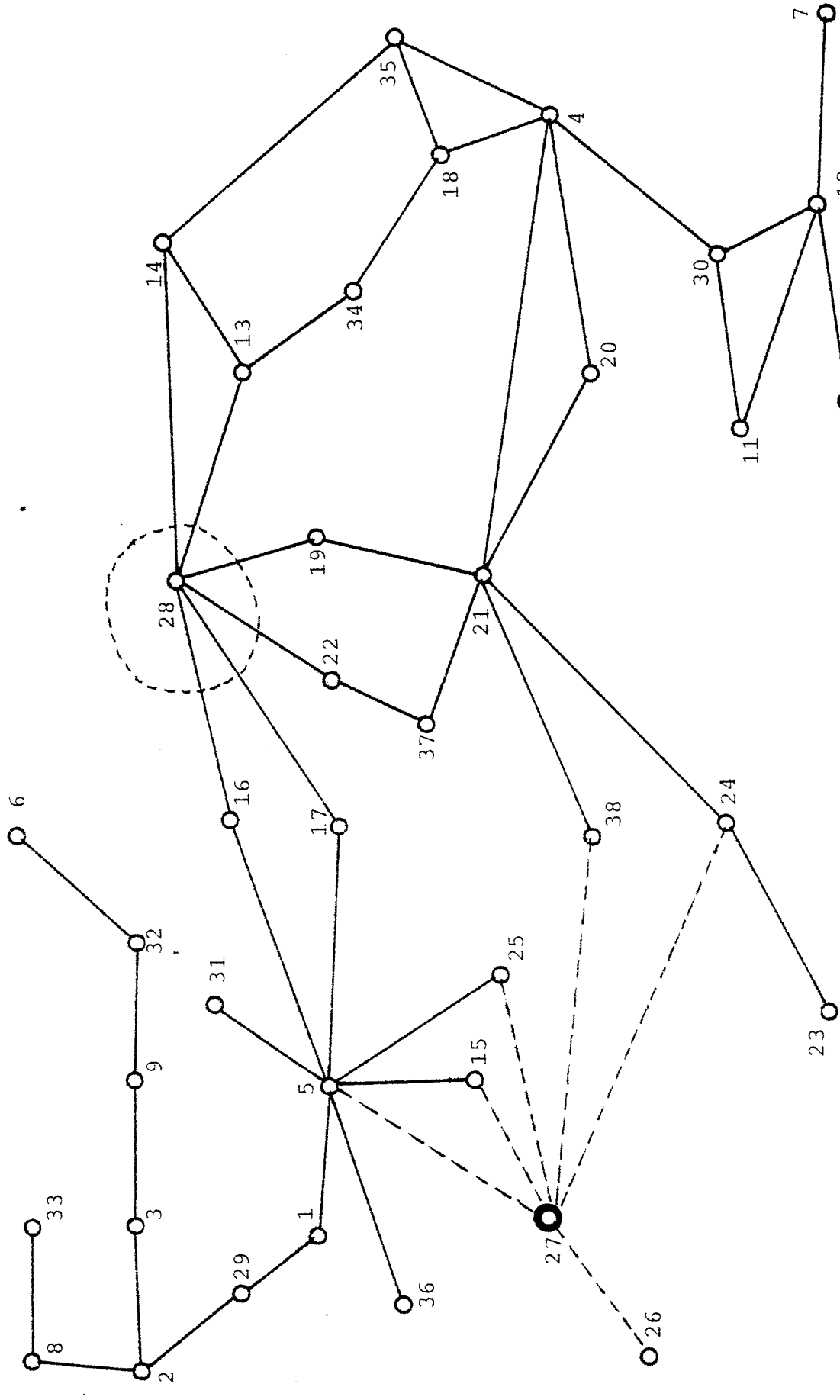

30 Nudos (IEEE)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

fig. 6.12



38 Nudos (Sevilla)

fig. 6.13

1) Ordenación. Este proceso se realiza siguiendo el algoritmo descrito en el epígrafe anterior. Con ello introducimos las siguientes particiones en las matrices B', B" y de admitancias de nudos:

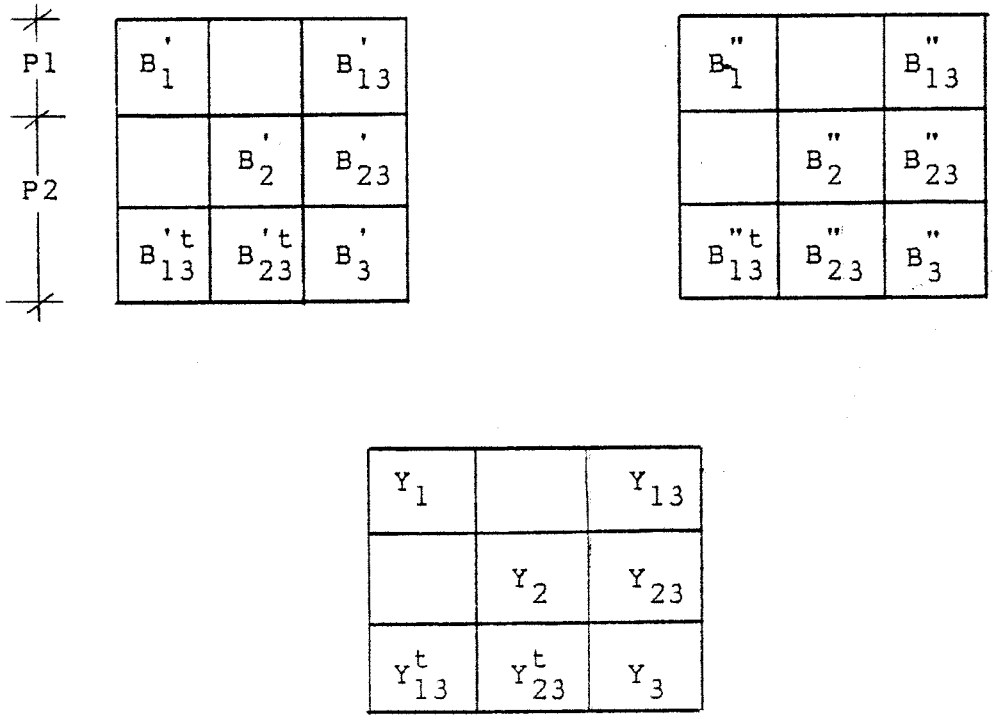


figura 6.14

El procesador P1 se encargará de la subdivisión superior, y el P2 de la inferior más la interconexión, como indica la figura 6.14.

Aprovechamos la simetría almacenando sólo las mitades superiores. Asímismo, las matrices B'_3, B''_3 e Y_3 se almacenan en la forma convencional, como un conjunto de dos dimensiones, puesto que son muy pequeñas y al final del proceso de triangularización quedarán llenas, probablemente en su totalidad (véanse las figuras 6.2 a 6.7).

El nudo de referencia se numerará el último en la matriz de admitancias, la única en que aparece dicho nudo.

Los vectores $\Delta P/V$, $\Delta Q/V$, ϑ, V, y numerosos vectores auxiliares, tambien quedarán particionados correspondientemente:

$$\begin{bmatrix} \Delta P_1/V \\ \Delta P_2/V \\ \Delta P_3/V \end{bmatrix} , \begin{bmatrix} \Delta Q_1/V \\ \Delta Q_2/V \\ \Delta Q_3/V \end{bmatrix} , \begin{bmatrix} \vartheta_1 \\ \vartheta_2 \\ \vartheta_3 \end{bmatrix} , \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix}$$

El proceso de ordenación es muy rápido (véase epígrafe 6.5), y lo realiza P2. Es factible introducir también aquí paralelismo, puesto que, una vez obtenida la interconexión, las dos submatrices resultantes se pueden ordenar independientemente por el esquema 2 de Tinney.

Se han probado las dos alternativas, y son equivalentes en sus resultados, debido a que el tiempo necesario para que P2 pase la información necesaria a P1, contrarresta lo que se pudiera ahorrar, siendo además más compleja la programación. Definitivamente se ha optado por la ordenación secuencial, aunque esta de-

cisión debería revisarse para sistemas más grandes que los utilizados en los tests, en los que los tiempos de ordenación pueden llegar a ser apreciables.

2) Formación de matrices. Esta etapa prácticamente no tiene ninguna particularidad, salvo la lógica simple.que decide a qué submatriz pertenece la línea que se está procesando en ese instante. Estas decisiones las toma P2, que es el que dispone de los datos del sistema.

Al igual que en el proceso de ordenación, P1 y P2 podrían colaborar en el de formación, pero al menos para los sistemas ensayados no ha dado resultados significativos, y sin embargo sí complica la programación, en lo que respecta a coordinación.

Estas etapas 1) y 2) son las únicas que no se realizan en paralelo, pero tampoco involucran excesivo cálculo, ni es preciso realizarlas por completo cada vez que haya un cambio, sino sólo parcialmente.

3) Factorización de matrices (fig. 6.15). Se describirá el proceso seguido para triangularizar B', y análogamente se haría con B". Para ello conviene tener en mente la estructura matricial dada en la figura 6.14.

Mientras P1 triangulariza B'_1, que sólo afecta a dicha matriz y a B'_{13}, P2 está haciendo simultáneamente lo propio con B'_2 y B'_{23}, sin interferirse.

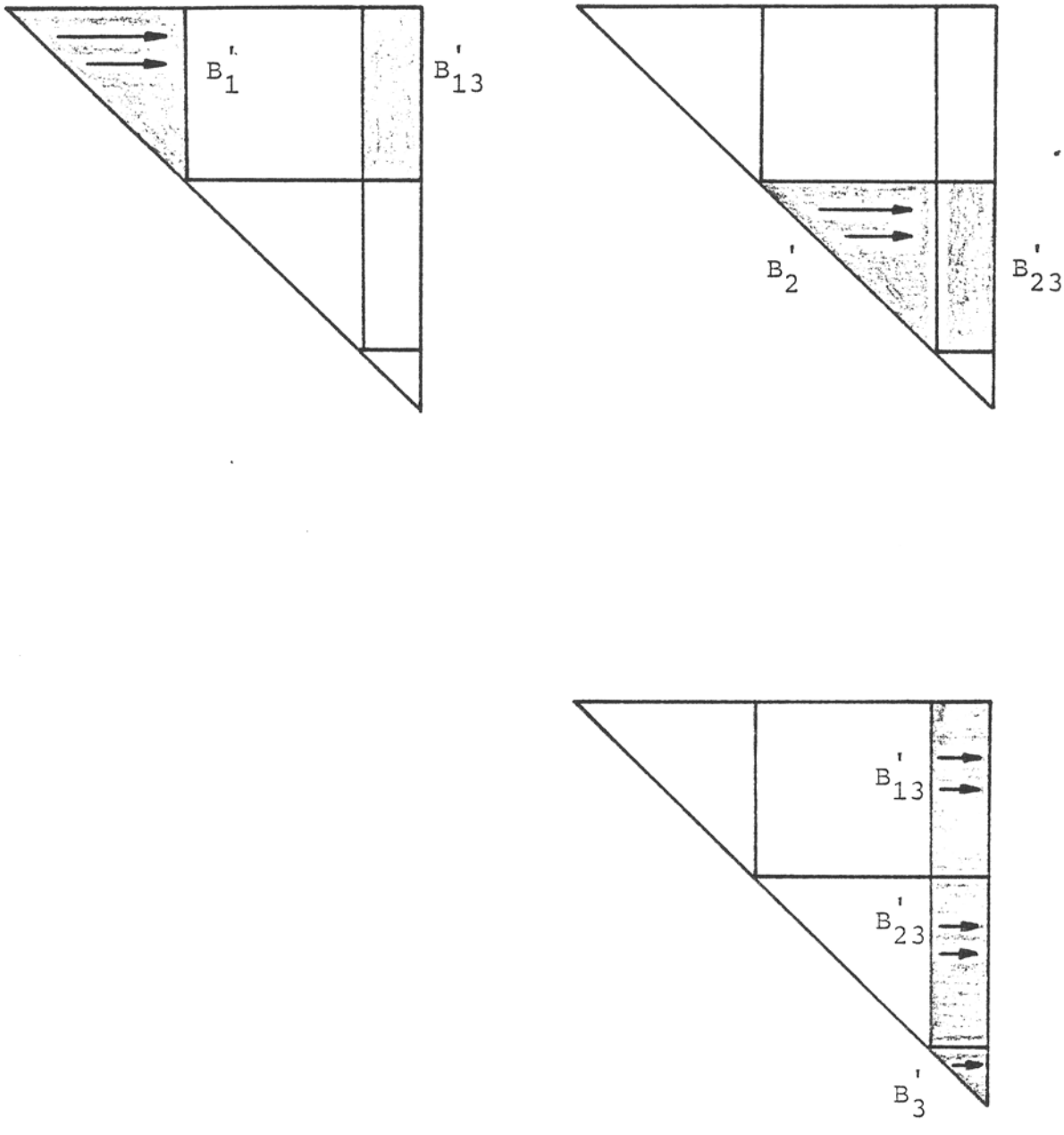


fig. 6.15

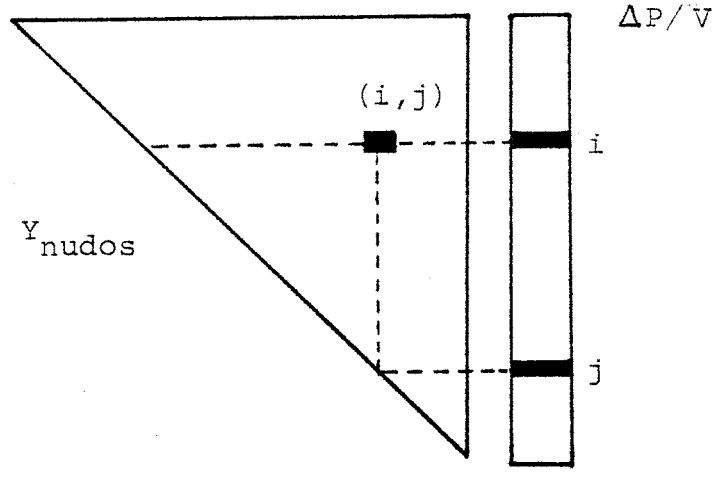
Cuando ambos procesadores han acabado esa fase, P2 hace el resto, es decir triangulariza B'_{13}, B'_{23} y B'_3, por este orden. Esto es relativamente rápido, puesto que estos bloques son muy dispersos, salvo B'_3 que es muy pequeño.

El algoritmo utilizado es el descrito en el Apéndice 4.

4) Valores iniciales. Esta etapa apenas tiene importancia, pero aún así P1 inicializa V_1 y ϑ_1, mientras que P2 inicializa el resto.

Se entra ya en el proceso iterativo, que es el más importante en el cómputo global (figura 5.6).

5) Cálculo de $\Delta P/V$ (fig. 6.16). El proceso que sigue es similar al del cálculo de $\Delta Q/V$, que no se repetirá. Debe recordarse que sólo almacenamos la parte superior de la matriz de admitancias, y que por tanto el elemento Y_{ij} contribuye a ΔP_i y ΔP_j:



Secuencia de P1:

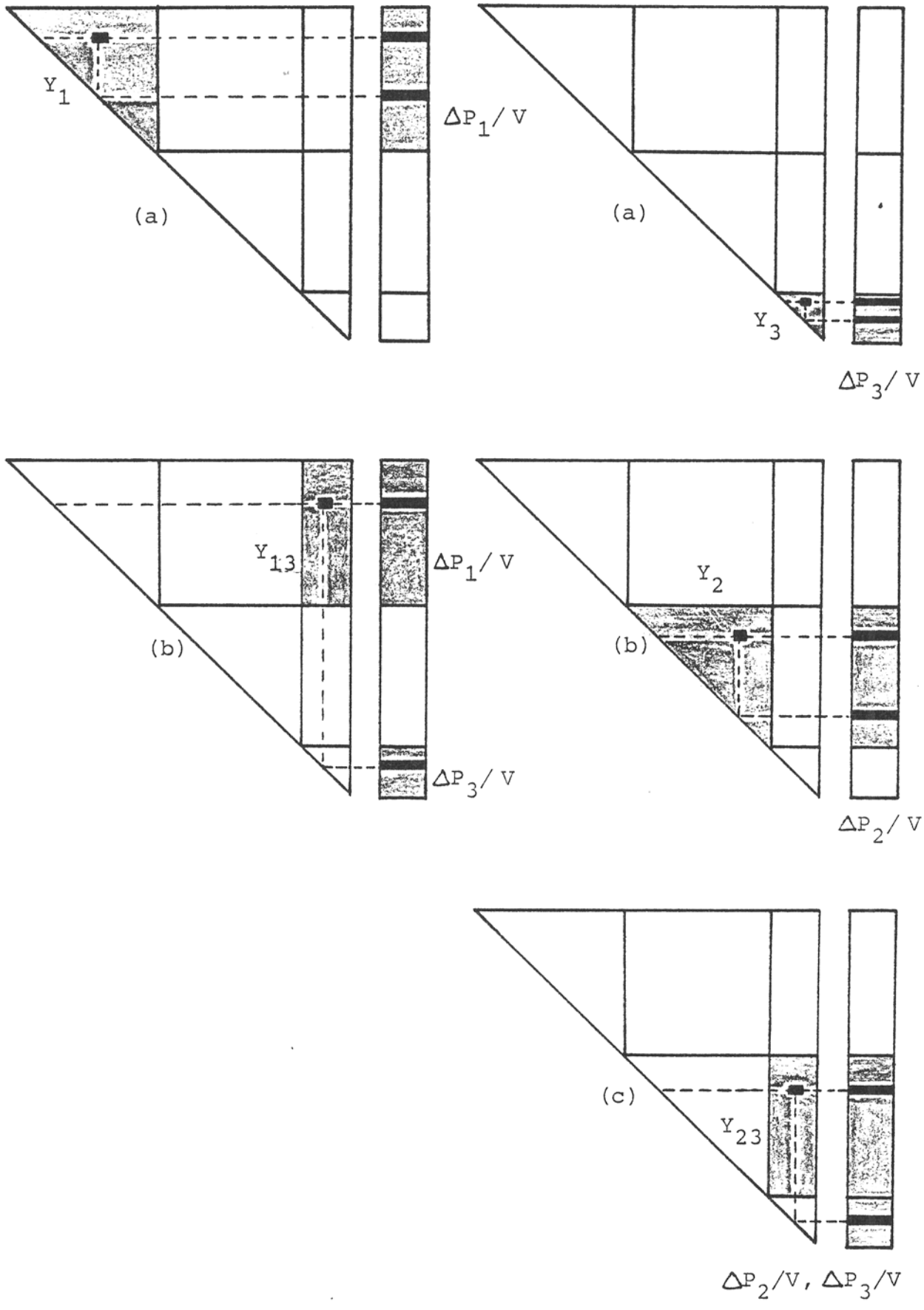

Y_1
$\Delta P_1 / V$
(a)
(a)
Y_3
$\Delta P_3 / V$
Y_{13}
$\Delta P_1 / V$
(b)
Y_2
(b)
$\Delta P_3 / V$
$\Delta P_2 / V$
(c)
Y_{23}
$\Delta P_2/V$, $\Delta P_3/V$
P1
P2

fig. 6.16

a) Contribución de Y_1 a $\Delta P_1/V$.

b) Si P2 ha realizado su correspondiente etapa a), que se describe más adelante, P1 calcula la contribución de Y_{13} a $\Delta P_1/V$ y $\Delta P_3/V$. El motivo de esperar a que P2 haga la secuencia a) es porque así $\Delta P_3/V$ estará correctamente inicializado.

c) Como $\Delta P_1/V$ está completamente calculado, P1 puede indagar si satisface el criterio de convergencia.

Secuencia de P2.

a) Contribución de Y_3 a $\Delta P_3/V$.

b) Contribución de Y_2 a $\Delta P_2/V$.

c) Si P1 ha realizado su etapa b), P2 calcula la contribución de Y_{23} a $\Delta P_2/V$ y $\Delta P_3/V$.

d) Al estar completamente calculadas $\Delta P_2/V$ y $\Delta P_3/V$, P2 averigua si se satisface el criterio de convergencia.

Si alguno de los dos procesadores, detecta que no se ha producido la convergencia, se pasa al siguiente punto.

6) Solución del sistema B'. $\Delta\theta = \Delta P/V$, con el conocimiento de la tabla de factores de B', calculada según 3). Consta de dos etapas diferentes:

I) Eliminación hacia adelante (fig. 6.17).

Secuencia de P1:

a) Aplicar la tabla de factores de B_1' sobre $\Delta P_1/V$

Secuencia de P2:

a) Aplicar B_2' sobre $\Delta P_2/V$.

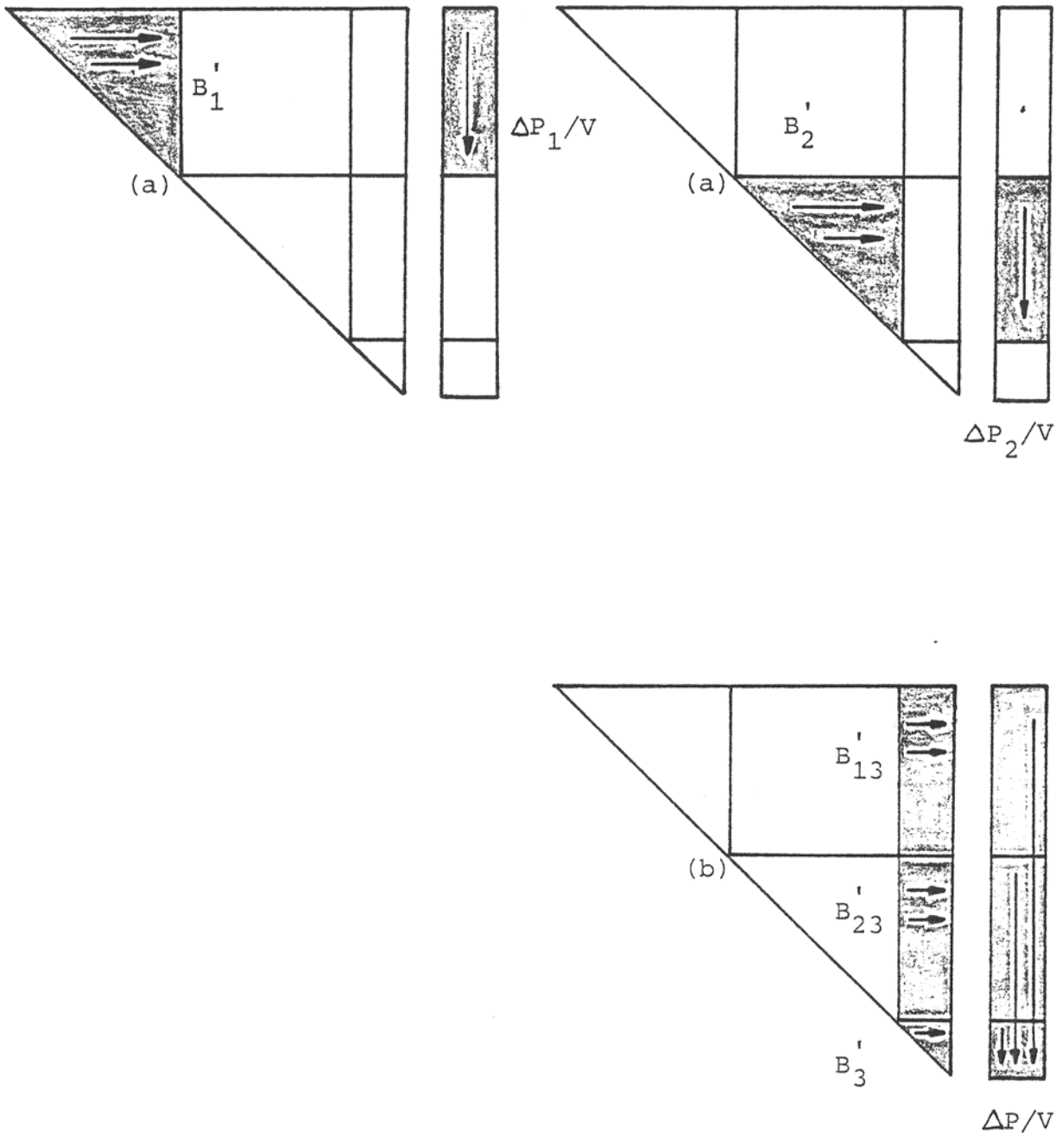
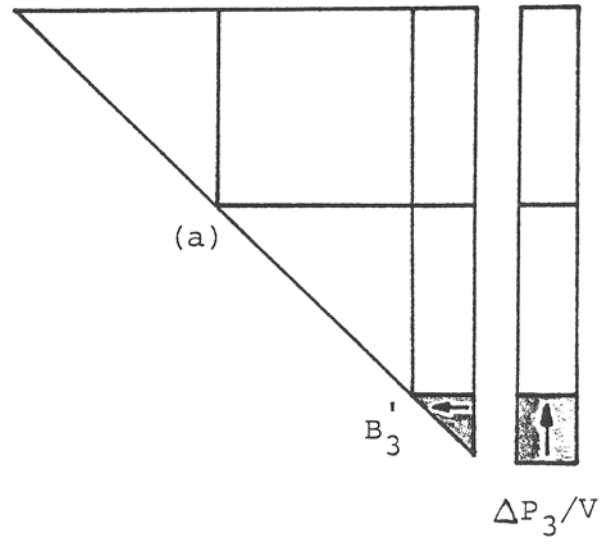
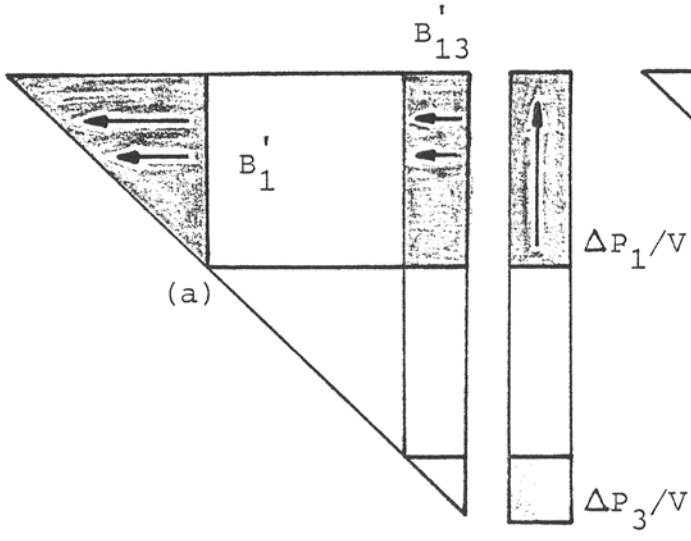
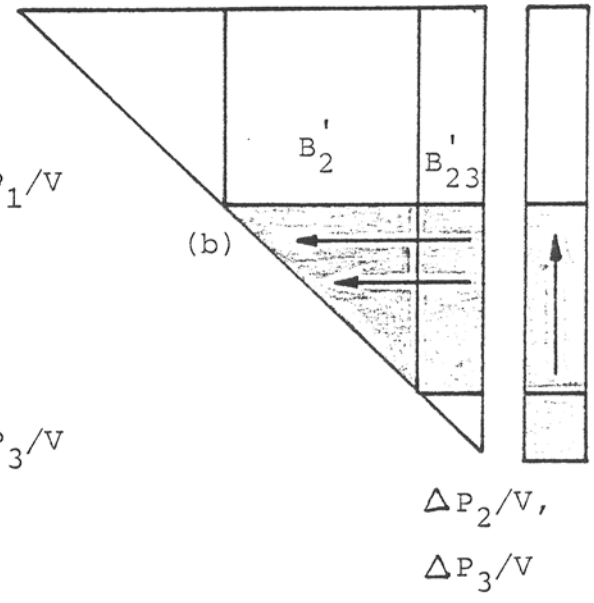


fig. 6.17







P1

P2

fig. 6.18

b) Si P_1 ha finalizado su fase a), aplicar B'_{13}, B'_{23} y B'_3 sobre $\Delta P_3/V$.

II) Substitución hacia atrás (fig. 6.18).

Secuencia de P2:

a) Aplicar B'_3 a $\Delta P_3/V$

b) Aplicar B'_2 y B'_{23} a $\Delta P_2/V$

Secuencia de P1:

a) Si P2 ha acabado el apartado a) anterior, aplicar B'_1 y B'_{13} a $\Delta P_1/V$.

Una vez realizadas estas operaciones, el vector $\Delta P/V$ se ha convertido automáticamente en el vector $\Delta\theta$. La actualización de θ se hace como en 4).

Obsérvese que el proceso en sí, dicta muchas veces el orden en que se deben realizar los diferente pasos. Cuando ésto no es así, y existe libertad para cambiar el orden de las operaciones, se ha procurado dar el mayor paralelismo posible, tal es el caso del cálculo de $\Delta P/V$, por ejemplo.

Asimismo, es preciso hacer notar que no todas las etapas son igual de costosas en tiempo. Las más rápidas, por el pequeño tamaño de las matrices en juego (interconexión), son precisamente las que no se pueden realizar en paralelo, en un esquema por bloques.

La coordinación entre las secuencias de los dos procesadores se realiza utilizando banderas. Un procesador informa al otro que ha llegado a un punto determinado, activando una bandera. El receptor de esta información se encarga de desactivarla, para un uso posterior. Se necesitan dos banderas diferentes para indicar si se ha optado por uno u otro camino en la toma de una decisión (tal es el caso del estudio de la convergencia).

El otro subproblema del algoritmo desacoplado rápido, la solución de B" . $\Delta V = \Delta Q/V$, se lleva a cabo de la misma manera, utilizando las matrices y vectores apropiados en las llamadas a las mismas subrutinas, desarrolladas para el proceso descrito con anterioridad.

6.4 ARQUITECTURA ADOPTADA.

Lo ideal hubiera sido disponer de dos PDP-11/34, puesto que ya se habían desarrollado numerosos programas para el caso secuencial, así como las rutinas de tratamiento de matrices vacias, que seguían siendo válidas.

El único inconveniente de esta arquitectura, hubiera sido el lograr una comunicación eficiente entre ambos miniordenadores, puesto que en cada iteración es preciso traspasar de uno a otro resultados parciales.

Con una sola máquina como las mencionadas, lo único que se podía hacer era simular el comportamiento del algoritmo. Para

ello, como el sistema operativo RSX-11M gestiona múltiples tareas, se programaron dos, de forma que cada una realizara las labores de un procesador. Las comunicaciones entre ambas tareas se hacían a través de una zona COMMON residente en memoria. La sincronización se realizaba con los propios "flags" globales que el sistema operativo pone a disposición de los usuarios.

Se pretendía con la simulación, como así fue, comprobar la validez de los algoritmos mencionados en los epígrafes anteriores.

Quedó en el aire, sin embargo, lo más importante, que era determinar la eficacia del algoritmo, dado que el sistema operativo no permite medir los tiempos que la CPU dedica a cada tarea, y además el COMMON residente no es una forma realista de simular las comunicaciones entre procesadores.

A pesar de que numerosas publicaciones se quedan en la etapa de simulación, ésto no parece satisfactorio, por lo que se decidió pasar a la ejecución con dos procesadores reales, que permitiría con certeza conocer la bondad del algoritmo.

Para ello se adoptó una solución económica, consistente en la utilización de dos microprocesadores de 16 bits, conectados en un bus estandard VME. La figura 6.19 muestra gráficamente esta arquitectura, que se pasa a describir.

Cada uno de los procesadores consta de los siguientes elementos:

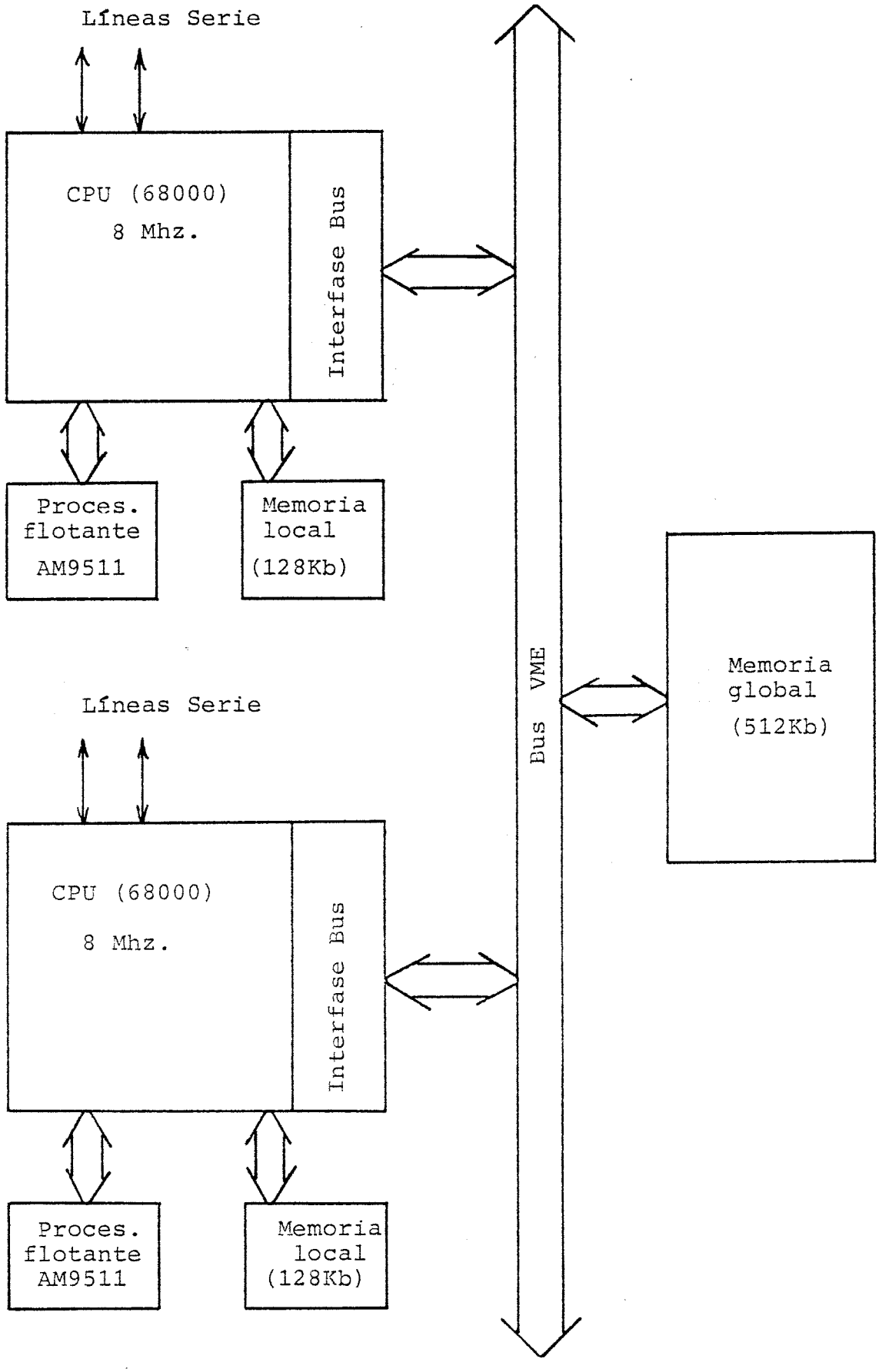

Líneas Serie
CPU (68000)
8 Mhz.
Interfase Bus
Proces. flotante AM9511
Memoria local (128Kb)
Líneas Serie
CPU (68000)
8 Mhz.
Interfase Bus
Proces. flotante AM9511
Memoria local (128Kb)
Bus VME
Memoria global (512Kb)

fig. 6.19

- Tarjeta de CPU basada en el MC68000, con un bus de datos de 16 bits, una capacidad de direccionamiento de 16 Mbytes, y una frecuencia de trabajo de 8 Mhz.
- Memoria local, en la propia tarjeta de hasta 128 Kbytes de RAM o ROM.
- Tres contadores programables, que nos permitirían medir exactamente tiempos de ejecución.
- Interfase con el bus estandard VME, de utilización cada vez más extendida, lo que la hace muy interesante para aplicaciones como la nuestra de multiprocesadores.
- Arbitro del bus, seleccionable opcionalmente.
- Dos interfases serie RS232C.
- Puertas programables de entrada/salida paralelo (no utilizadas).

Entre los elementos constitutivos mencionados se nota la ausencia de uno, fundamental para nuestra aplicación: Un procesador de punto flotante.

El propio fabricante de la CPU, ha puesto en el mercado el procesador flotante MC68881, que sería el ideal, al ser de 32 bits. Por desgracia, en estos momentos, sólo se han distribuido muestras, no estando totalmente comercializado, al menos en nuestro país.

Entre los procesadores disponibles más populares se encuentra el AM9511, por el que finalmente se ha optado. Su principal inconveniente es que externamente es sólo de 8 bits, aunque permite realizar prácticamente todas las funciones más conocidas, y

desde luego todas las necesarias.

El otro inconveniente es que el formato de los números reales es diferente al del PDP-11/34, con lo que ha sido precisa la conversión de los datos de las redes, de un formato a otro.

Al ser de 8 bits, el tiempo requerido para pasarle los datos (que son de 32 bits), las órdenes, y para recibir el resultado, supone una parte importante del tiempo total ("overhead").

Por poner un ejemplo, la multiplicación del número 1.3579 por sí mismo, tarda unos 135 μseg (a 2 Mhz), frente a 74 μseg que en teoría tarda el proceso multiplicativo estrictamente.

Existían dos opciones para el acoplamiento de este procesador flotante a la tarjeta primitiva: Conectarlo como los periféricos de 8 bits que ella misma trae, en forma síncrona, o bien asíncronamente, como se realiza el acceso a memoria (local o global). Se llevó a cabo la segunda posibilidad porque daba mayor rapidez, aunque su interfase resultaba más compleja.

El último elemento que aparece en la figura 6.19, es una tarjeta con 512 Kbytes de memoria RAM dinámica, accesible desde ambos procesadores, y que servirá básicamente como elemento comunicador entre ellos (sustituyendo al COMMON residente de la simulación).

Para poder utilizar el bus, y por tanto la memoria global, se ha habilitado el árbitro en una de las dos tarjetas de CPU.

Este árbitro gestiona, en forma transparente al usuario, el acceso al bus, de manera que no hay problema si ambos procesadores quieren acceder simultáneamente a la memoria externa. Lógicamente, el acceso a esta memoria es más lento que a la local, debido al tiempo que conlleva la realización del arbitraje, y a los retardos propios del bus.

Con esta arquitectura, se han vuelto a desarrollar todos los programas que sirvieron para la simulación del algoritmo, pero utilizando lenguaje máquina. Para ello se dispone de un ensamblador cruzado en el PDP-11/34. Una vez ensamblados los programas, se transmiten junto a los datos de la red en concreto, por una de las líneas serie, a la memoria del microprocesador.

De esta forma se simplifica el "software" necesario para este desarrollo, siendo preciso sólo un pequeño programa monitor que permita la recepción en memoria de los programas y su "debugging". Al mismo tiempo se utilizan los recursos en memoria de masas del PDP-11/34, lo que hace finalmente que la solución adoptada sea muy económica, si ya se dispone de un sistema de desarrollo mínimo.

Lo más engorroso de esta etapa ha sido sin duda la necesidad de realizar todos los programas en ensamblador.

Desacoplado Rápido (68.000).

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	.003	.005	.010	.013	.016	.021	.023	.050	.162
Form. Matr.	.027	.040	.063	.091	.086	.117	.096	.161	.405
Factorizac.	.008	.019	.025	.032	.056	.042	.069	.139	.181
Proc. Iter.	.458	.529	.940	1.960	1.099	2.731	2.050	2.701	5.187
Total	.496	.593	1.038	2.096	1.257	2.911	2.238	3.051	5.935

Iteraciones	5 1/2	4	4 1/2	7 1/2	3 1/2	8	6	4 1/2	4 1/2
Tiempo / Itera.	.0832	.1322	.2088	.2613	.3140	.3413	.3416	.6002	1.1526

fig 6.20

6.5 RESULTADOS COMPARATIVOS.

Para poder evaluar la bondad del algoritmo de resolución en paralelo, ha sido preciso primero programar la versión secuencial trabajando en uno solo de los procesadores.

Los resultados se muestran en la figura 6.20, donde los tiempos se expresan como siempre en segundos.

Antes de continuar, se pueden comparar por curiosidad los tiempos de esa figura, con los dados en la figura 5.7, que son los resultados del mismo algoritmo secuencial pero para el PDP-11/34.

Son evidentes los siguientes hechos:

1) Los tiempos de ordenación de nudos y de formación de las matrices, son mayores en el PDP, lo cual es lógico si se tiene en cuenta que ahí van incluidos los accesos al disco, que en el 68000 no existen. Además esos procesos requieren pocas operaciones aritméticas.

2) Los resultados en lo que respecta a la factorización no son significativos en el PDP-11/34, porque son muy pequeños, del mismo orden que la precisión con que se mide el tiempo, $\pm$.02 seg.

3) Los tiempos más importantes son sin duda los del proceso iterativo, en los que existe igualdad de condiciones en ambas máquinas.

Trás un pequeño cálculo, se obtiene que los tiempos invertidos en una iteración por el microprocesador MC68000 son 4.45 $\pm$.14 veces lo que tarda el PDP-11/34.

Ello se explica si tenemos en cuenta que la multiplicación mencionada en el epígrafe anterior, en la que el procesador AM-9511 tardaba 135 μseg, se lleva a cabo en tan solo 26 μseg, en el procesador flotante del PDP, lo que supone una velocidad de cálculo unas 5 veces mayor. La pequeña diferencia entre 5 y 4.45 es debida a que en las instrucciones no aritméticas, el MC68000 aventaja sobradamente al PDP.

4) Destaca el hecho de que en la red de 39 nudos, se requiere media iteración más para obtener la convergencia. Como se comentó en 3.5, ésto obedece seguramente a que el procesador AM9511 tiene un formato con un bit menos de mantisa, y por tanto se van acumulando más errores de redondeo, que pueden ser importantes frente a .0001, que es el criterio de convergencia tomado siempre.

Los resultados del algoritmo en paralelo se muestran en la figura 6.21. Para poder compararlos con los del algoritmo convencional, en la figura 6.22 se muestran los cocientes T2/T1, siendo T2 los tiempos invertidos por los dos procesadores (figura 6.21) y T1 los de un solo procesador (figura 6.20).

Desacoplado Rápido. Solución Paralelo.

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	.004	.006	.011	.014	.017	.020	.024	.051	.124
Form. Matr.	.030	.044	.067	.096	.091	.123	.100	.169	.423
Factorizac.	.007	.015	.015	.027	.042	.028	.061	.127	.136
Proc. Iter.	.329	.340	.580	1.293	.733	1.569	1.501	1.889	3.092
Total	.370	.405	.673	1.430	.883	1.740	1.686	2.236	3.775

Iteraciones	5 1/2	4	4 1/2	7 1/2	3 1/2	8	6	4 1/2	4 1/2
Tiempo / Itera.	.0598	.085	.1288	.1724	.209	.196	.250	.419	.687

fig. 6.21

Tiempos algoritmo paralelo/ Tiempos algoritmo secuencial

Nudos / Proceso	13	14	26	29	30	38	39	57	118
Ordenación	1.333	1.2	1.1	1.076	1.062	.952	1.043	1.02	.765
Form. Matr.	1.11	1.1	1.063	1.054	1.058	1.051	1.041	1.049	1.044
Factorizac.	.875	.789	.6	.843	.75	.666	.884	.913	.751
Proc. Iter.	.718	.642	.616	.659	.665	.574	.731	.698	.596
Total	.745	.682	.648	.682	.705	.597	.753	.732	.636

Iteraciones	5 1/2	4	4 1/2	7 1/2	3 1/2	8	6	4 1/2	4 1/2
Tiempo / Itera.	.718	.642	.616	.659	.665	.574	.731	.698	.596

fig. 6.22

6.6 DISCUSION DE RESULTADOS.

La figura 6.22 es la más importante para evaluar la bondad del algoritmo de resolución en paralelo.

En primer lugar destaca, que los tiempos de ordenación de la matriz, empiezan siendo sensiblemente iguales que para el caso secuencial, pero luego disminuyen conforme aumenta el número de nudos. En la figura 6.23 se puede visualizar esta afirmación.

La ordenación en bloques se diferencia básicamente de la ordenación por el esquema 2 (o de mínimo grado) en dos puntos.

1) Requiere ordenar inicialmente la matriz en banda.
2) Una vez identificados los dos bloques, ordena por el esquema 2 cada bloque por separado, en lugar de hacerlo con la matriz completa.

Estas diferencias justifican los resultados de la figura 6.23. En efecto, el punto 1) haría que los tiempos fuesen mayores, aunque sólo ligeremente, puesto que esa etapa es muy rápida. Por el contrario, el punto 2), que pesa más en el cómputo global, ahorra tiempo frente a la ordenación convencional, puesto que se requiere más esfuerzo para ordenar una matriz, que dos matrices aproximadamente la mitad de grandes, a causa de que el nudo con mínimo grado hay que buscarlo entre un conjunto más pequeño (y el tiempo de esta búsqueda crece más que linealmente).

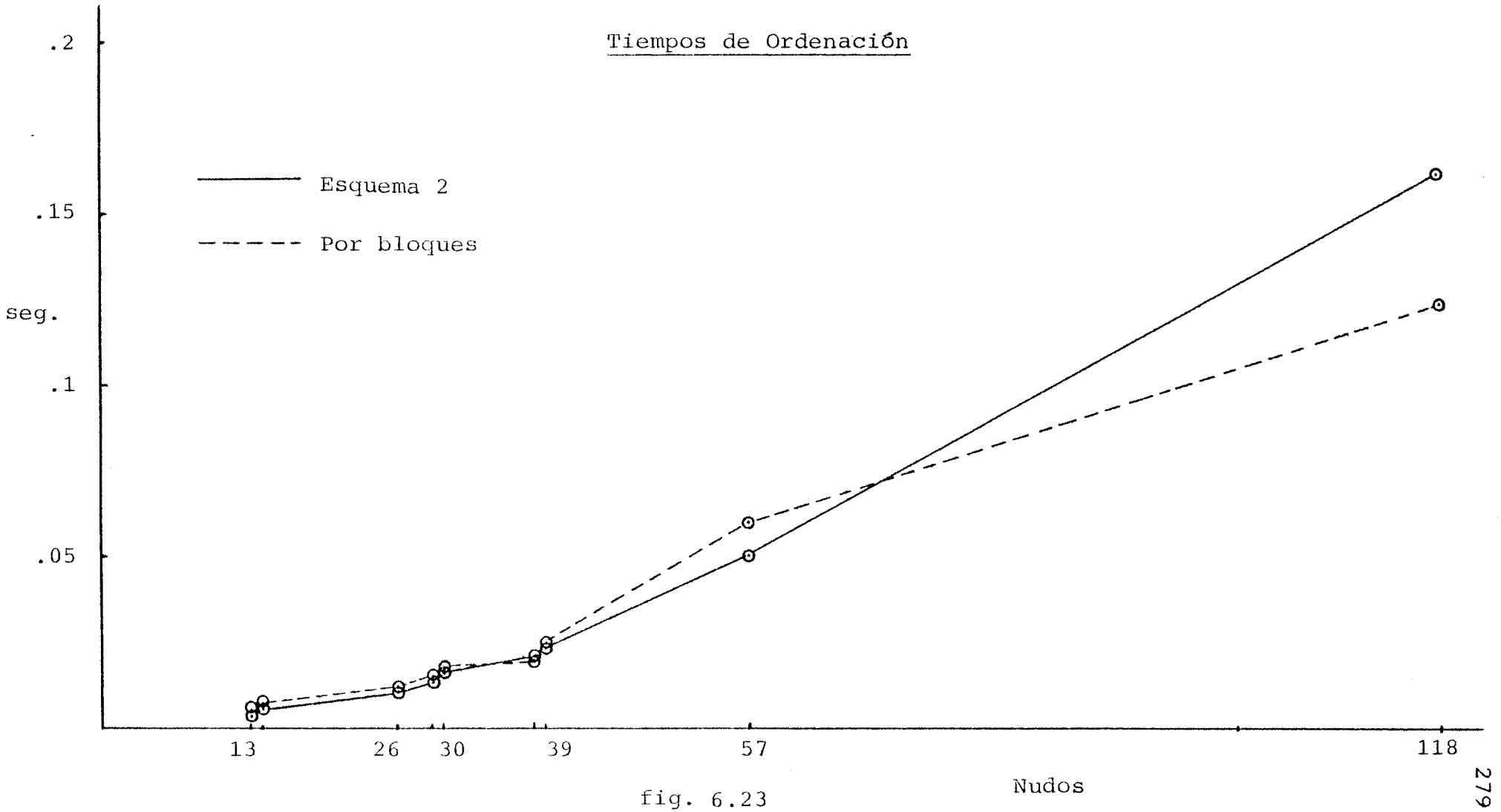

Tiempos de Ordenación
Esquema 2
Por bloques
seg.
.2
.15
.1
.05
13
26
30
39
57
118
Nudos

fig. 6.23

Para aclarar esto último piénsese en el siguiente ejemplo: La forma más corriente de ordenar de menor a mayor un vector de N números, requiere ejecutar N(N-1)/2 comparaciones. Por tanto, ordenar dos vectores de N/2 elementos necesitará N(N-2)/4, que aproximadamente supone la mitad, para N suficientemente grande.

El modelo de este ejemplo es válido exactamente para el esquema 1 de ordenación. El esquema 2 es más complejo, puesto que los nudos se van eliminando conforme se ordenan. Este proceso de eliminación, sin embargo es prácticamente el mismo para las dos submatrices, que para la matriz completa, y además crece sólo linealmente con el número de nudos, a juzgar por los resultados del Apéndice 4 .

Como resumen se puede decir, que ambas ordenaciones tienen un coste de tiempo fijo, que es mayor para el algoritmo propuesto, y un coste variable con el número de nudos, que crece menos acusadamente para dicho algoritmo. El punto en que este algoritmo empieza a ser más rápido, es para unos 60 nudos, según la figura 6.23.

Recuérdese que la ordenación de los nudos (y tambien la formación de matrices) no se hacía en paralelo, aunque es factible, porque no se encontró ninguna mejora en los tiempos. Como el tiempo necesario para que el procesador principal, pase al otro la estructura de la submatriz que le corresponde ordenar, es proporcional al número de nudos, debe haber un punto en que empiece tambien a ser rentable la ordenación en paralelo, aunque no se haya encontrado para los sistemas relativamente pequeños que he-

mos estudiado. A partir de ese punto, la ventaja del algoritmo propuesto es todavía mayor que la que deja entrever la figura 6.23.

Los tiempos de formación de matrices, son ligeramente mayores para los dos procesadores trabajando en paralelo que en el caso secuencial, aunque se tienden a igualar. Ello se debe a la lógica adicional que determina, cuando se leen los datos de una línea, cuáles de las 5 matrices deben modificarse, pero sobretodo a que algunas de estas matrices están en la memoria global, de acceso más lento.

Una buena parte de la factorización sí se realiza en paralelo, de ahí que los tiempos sean menores (figura 6.22). Los resultados demuestran que esta etapa del algoritmo no es muy eficiente. Además son irregulares, variando desde 0.6 para la red de 26 nudos, hasta 0.91 para la de 57. Este comportamiento depende de la distorsión que la ordenación por bloques, introduzca en la ordenación que la matriz tendría con el esquema 2 convencional. Es decir, el hecho de forzar a que ciertos nudos se numeren al final, y de que los demás no puedan salirse de su bloque respectivo, da como resultado un ligero aumento del número total de elementos en la matriz triangularizada. Este aumento es mayor cuanto más nudos tiene la interconexión, como se muestra en la figura 6.24. En concreto, la red de 26 nudos, cuya interconexión es de 1 nudo, sólo aumenta en 2 elementos, y el porcentaje de tiempos entre ambas factorizaciones es, como se ha dicho, del 60%. En el otro extremo está la red de 57 nudos, con una interconexión de 6 nudos, y un aumento de 26 elementos, el mayor de todos en forma

N° de Elementos de B' Triangularizada.

Nudos Totales	13	14	26	29	30	38	39	57	118
Nudos Interc.	2	2	1	2	3	1	3	6	6
Elem. Esq. 2	22	44	60	74	104	102	132	258	506
Eleme. Orden. en bloques	24	44	62	76	106	102	136	284	536

fig. 6.24

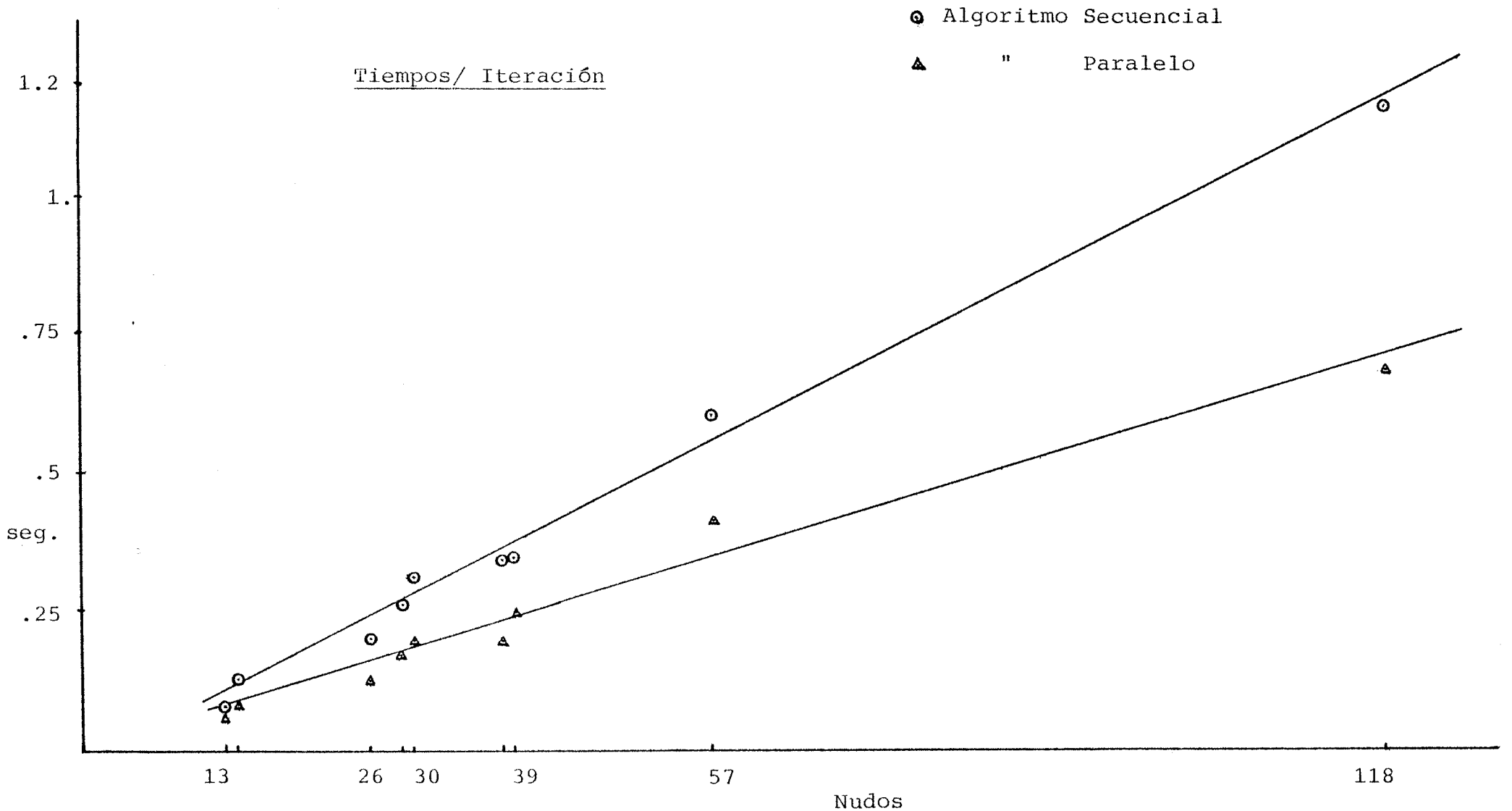

Algoritmo Secuencial
" Paralelo
Tiempos/ Iteración
1.2
1.
.75
.5
seg.
.25
13
26
30
39
57
118
Nudos

fig. 6.25

relativa, lo que hace que la factorización en paralelo propuesta, funcione peor (91%).

Centrémonos por último en los tiempos que se requiere en el proceso iterativo, que son los más importantes, y más en concreto en los tiempos por iteración, para obtener conclusiones independientemente del número de iteraciones.

En la figura 6.25 se representan los tiempos por iteración, para el algoritmo secuencial y para el algoritmo paralelo. Ambas nubes de puntos se han ajustado por mínimos cuadrados a sendas rectas, pués a juzgar por los resultados del Apéndice 4, y los comentarios hechos en 6.7, el esfuerzo de cálculo por iteración debe crecer linealmente. Verdaderamente el coeficiente de correlación es bastante alto (0.996 para el algoritmo secuencial, y 0.984 para la solución en paralelo) lo que confirma la bondad del ajuste.

Las ecuaciones de las rectas para uno y dos procesadores son:

$$T_1 = 0.010116\ N - 0.0275$$
$$T_2 = 0.0060344N + 0.001144$$

Es interesante representar los tiempos T_2/T_1, dados en la tabla de la figura 6.22. La nube de puntos se puede ajustar al cociente de las dos ecuaciones anteriores, como se ha hecho en la figura 6.26. Es interesante calcular el límite T_2/T_1, cuando N es

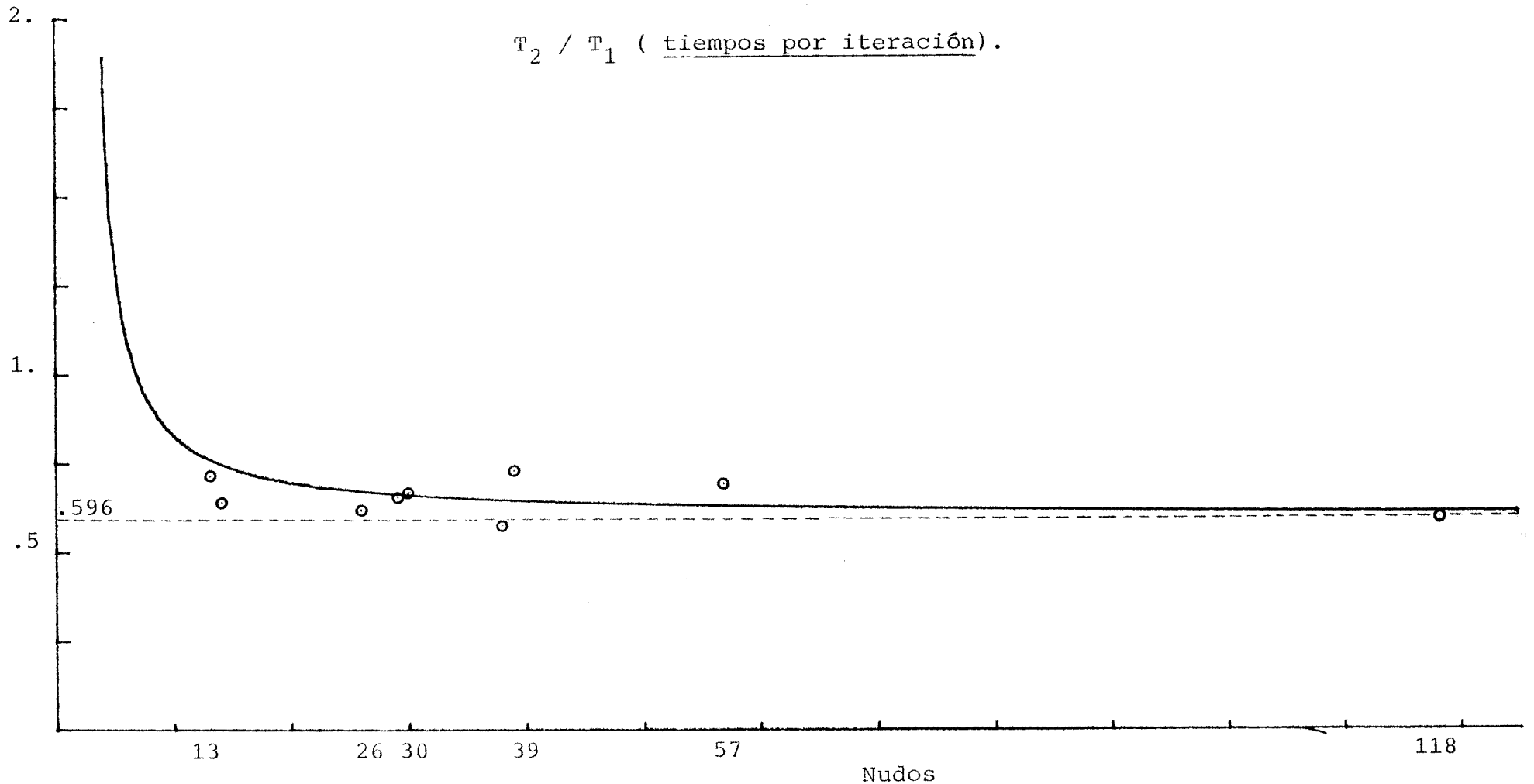


fig. 6.26

suficientemente grande. Ese valor lo da el cociente de las pendientes de las rectas, y es 0.596. Es decir, el algoritmo de resolución en paralelo, tardará el 59.6% del tiempo del algoritmo secuencial, en el proceso iterativo.

Este es un resultado importante, que muestra la eficacia del algoritmo. El límite teórico de tiempos es el 50%, pero dicho límite no se podrá alcanzar nunca por tres motivos, algunos de los cuales ya se han comentado:

1) Es preciso que algunos vectores y matrices, comunes a los dos microprocesadores, estén en la memoria global, de acceso considerablemente más lento.
2) La ordenación por bloques, como indica la figura 6.24, hace que el número total de elementos de las matrices B' y B" sea mayor que en el caso secuencial, y por tanto el número de operaciones por iteración es también mayor.
3) Existe una interconexión, que puede suponer entre un 5 y un 10% del total de nudos, que no se puede resolver en paralelo, y además las dos submatrices no son exactamente iguales. Esta es la causa más importante del aumento de tiempo sobre el 50% teórico.

Por lo tanto, bajar del 60%, se puede considerar satisfactorio para el algoritmo de solución en paralelo.

Se puede pensar que los resultados son incluso pesimistas, debido fundamentalmente a que la solución en paralelo tiene más

interés cuanto más grande es la red a resolver, y aquí se han mostrado resultados de sistemas excesivamente pequeños, al no haberse podido conseguir otros más grandes. Lo ideal hubiese sido trabajar con una decena de redes entre 100 y 500 nudos, con lo que seguramente los tiempos relativos obtenidos habrían sido menores.

El trabajar con sistemas grandes tiene además la ventaja de que los resultados deben ser más homogéneos.

En redes más pequeñas el algoritmo de ordenación se comporta más irregularmente. Así, la red de 57 nudos tiene una interconexión de 6 nudos, que comparativamente es muy alta (la misma que para 118 nudos). Como se observa claramente en la figura 6.25 y 6.26 esta red se aparta excesivamente de las previsiones, y contribuye a que la correlación (0.984) sea menor que en el algoritmo secuencial (0.996).

Si ignoramos por completo esta red, la correlación sube a 0.994, y el límite, para N grande, de T_2/T_1 pasa a ser del 57%.

No parece descabellado suponer que ese límite se acerque en realidad al 55%, aunque ésto necesita realizar trabajos adicionales, para afirmarlo rotundamente.

Para finalizar este capítulo, se harán algunas consideraciones sobre el algoritmo de partición de la matriz descrito en 6.2.

Dicho algoritmo ha demostrado ser muy rápido, y suficientemente eficiente, incluso aunque no existan de forma natural en la matriz dos submatrices débilmente conectadas.

Su principal punto débil radica en la elección del primer nudo para la ordenación en banda, que depende mucho de la ordenación natural de la matriz. Se toma un nudo con mínimo grado, pensando que es de la periferia. A veces no ocurre así, pudiendo estar ese nudo próximo al centro de gravedad de la red, lo que daría una interconexión con excesivos nudos.

Esto mismo ocurre en los otros algoritmos publicados [75,90], y aunque la solución del problema requiere una investigación adicional, se pueden apuntar las líneas a seguir:

1) Repetir el algoritmo de ordenación para todos los nudos de mínimo grado.
2) Utilizar un algoritmo, semejante al propuesto por Gibbs et al. [44], para encontrar un nudo pseudoperiférico, y realizar la ordenación en banda partiendo de ese nudo.

La segunda solución es sin duda la más fiable, puesto que un nudo periférico no tiene porqué ser forzosamente un nudo de mínimo grado.

6.7 CONSIDERACIONES FINALES.

Una pregunta muy habitual en estos casos es la siguiente: Cómo crecerá el esfuerzo de cálculo y la capacidad de memoria, conforme aumenta el tamaño de la red ?

Se ha especulado en numerosas ocasiones, que este crecimiento es sensiblemente lineal con el número de nudos, en base sobretodo a resultados experimentales.

Un estudio teórico exacto es imposible de realizar, si no es para un caso concreto, en que se conozca la estructura de cada fila de la matriz de admitancias de nudos.

Sin embargo, aquí haremos un análisis aproximado, basado fundamentalmente en los resultados del Apéndice 4, del que intentaremos sacar conclusiones.

La parte concerniente a la capacidad de memoria está suficientemente clara, una vez conocido el método de almacenamiento utilizado (véase 5.2). Como la relación entre el número de elementos de las matrices involucradas, y el número de nudos varía usualmente entre 1.5 y 2, siendo extraño que llegue a 2.5, resulta claro que las necesidades de memoria crecen linealmente con el número de nudos, dentro de una banda que tenga en cuenta el caracter variable y aleatorio de esa relación.

En lo que respecta al esfuerzo de cálculo, nos centraremos en las dos etapas relacionadas con la solución del sistema de

ecuaciones lineales, es decir, la factorización y el proceso iterativo. Además nos limitaremos a la solución secuencial del problema, cuyo comportamiento es más fácil de predecir. A través de los resultados del epígrafe anterior, tendremos una idea suficientemente aproximada de lo que costará llevar a cabo la solución en paralelo.

Con el procesador flotante utilizado (AM9511), una división y una multiplicación requieren aproximadamente el mismo tiempo. Las sumas y restas son bastante variables, en función de los operandos particulares, pero de media vienen a tardar lo que una multiplicación. Consideraremos como tiempo único para cualquier operación 140 μseg, lo que a veces puede resultar muy pesimista.

1) Factorización. El número de operaciones para realizar esta etapa en la matriz B', viene dado por la figura A4.2 (utilizamos el algoritmo de Doolittle), y son aproximadamente:

$$\text{Divisiones} = N$$

$$\text{Multiplicaciones} = \frac{B'}{2}\left(3+\frac{B'}{N}\right) \qquad (6.1)$$

$$\text{Restas} = \frac{B'}{2}\left(1+\frac{B'}{N}\right) - (B' - L)$$

donde N es el número de nudos, L el número de líneas y B' el número de elementos de la parte superior de la matriz del mismo nombre.

La diferencia entre el número de restas de (6.1) y el del Apéndice 4, proviene de que al crear un elemento nuevo, hay que restar de cero, y esta operación nos la ahorramos en B'-L elementos creados.

La factorización de la matriz B" es más difícil de predecir, a causa de que el número inicial de elementos, siempre menor que L, puede ser muy variable. Para un sistema suficientemente grande, con M nudos P-V, podemos suponer de forma razonable, que el número inicial de elementos de B" es:

$$\frac{L}{N} (N - M) \qquad (6.2)$$

Entonces, el número de operaciones se obtiene de:

$$\text{Divisiones} = N-M$$

$$\text{Multiplicaciones} = \frac{B"}{2} \left(3 + \frac{B"}{N\ M}\right) \qquad (6.3)$$

$$\text{Restas} = \frac{B"}{2} \left(1 + \frac{B"}{N-M} \right) - \left(B" - \frac{L}{N} (N-M) \right)$$

siendo B" el número de elementos de esa matriz.

Apliquemos estos resultados a la red de 57 nudos. Los datos de partida son:

$$N = 57$$

$$M = 7$$

$$L = 78$$

$$B' = 129$$

$$B'' = 94$$

Sustituyendo en (6.3) y (6.1), sumando el numero total de operaciones y multiplicando por 140 μseg, se obtiene un tiempo de 132 mseg, que se acerca bastante al dado en la figura 6.20, 139 mseg.

Por supuesto no se han contado los tiempos de las instrucciones no aritméticas, pero a 8 Mhz suponen una parte muy pequeña del total (salvo los errores debidos a las suposiciones hechas, deben ser la diferencia entre los dos tiempos).

2) Cálculos para una iteración. Comencemos con el cálculo del vector independiente $\triangle P/V$, que a tenor de su expresión, y teniendo en cuenta que las razones trigonométricas sólo necesitan 5 multiplicaciones y 3 sumas (véase 5.5), se deduce el siguiente número de operaciones:

$$\begin{aligned} &\text{Divisiones} = N-1 \\ &\text{Multiplicaciones} = N-1 + 9L \\ &\text{Sumas y restas} = N-1 + 8L \end{aligned} \qquad (6.4)$$

El cálculo de $\Delta Q/V$ será análogo, pero por cada extremo de línea que incida en un nudo P-V, nos ahorramos una multiplicación y dos sumas (o restas). Este ahorro es difícil de cuantificar, porque depende del reparto de líneas entre nudos, pero si este reparto es homogéneo, el número de operaciones se puede calcular de:

$$\begin{aligned} \text{Divisiones} &= N-M \\ \text{Multiplicaciones} &= N-M+9L-\frac{2L}{N}M \\ \text{Sumas y Restas} &= N-M+8L-\frac{4L}{N}M \end{aligned} \qquad (6.5)$$

La solución de los dos sistemas de ecuaciones lineales, para obtener $\Delta\theta$, ΔV, requiere un número de operaciones dado simplemente por la suma de las dos columnas de la derecha de la figura A4.2, para ambas matrices B' y B":

$$\begin{aligned} \text{Multiplicaciones} &= 4N+2B'+2B''-2 \\ \text{Sumas y Restas} &= 2B'+2B'' \end{aligned} \qquad (6.6)$$

Por último, se necesitan N-1 y N-M restas flotantes, para el estudio de la convergencia en ΔP y ΔQ respectivamente, así como N-1 y N-M sumas para la actualización del vector de tensión (V y θ).

Operaciones para Factorizar B' y B" (Aprox.)

Divisiones ..... $2N-M$

Multiplicaciones.... $\frac{B'}{2}(3+\frac{B'}{N})+\frac{B''}{2}(3+\frac{B''}{N-M})$

Sumas y Restas ..... $L(2-\frac{M}{N})+\frac{B'}{2}(\frac{B'}{N}-1)+\frac{B''}{2}(\frac{B''}{N-M}-1)$

Operaciones para realizar 1 iter. completa(Aprox.)

Divisiones ..... $2N-M-1$

Multiplicaciones ... $6N+L(18-\frac{2M}{N})-M+2B'+2B''-3$

Sumas y Restas ..... $6N+L(16-\frac{4M}{N})-3M+2B'+2B''-3$

fig. 6.27

En la figura 6.27, se resumen el total de operaciones para los dos procesos. Si utilizamos otra vez el ejemplo de 57 nudos, los cálculos nos dan 591 mseg para una iteración, mientras que el tiempo medido fue de 600 mseg (fig. 6.20).

Las ecuaciones de la figura 6.27, no se pueden utilizar a priori para calcular las operaciones que tenemos que realizar, ya que los elementos de B' y B" ya triangularizadas, no se conocen con exactitud hasta tanto no se ejecute ese proceso.

Dichas ecuaciones son suficientemente exactas, pero para responder a la pregunta formulada al principio, acerca del esfuerzo de cálculo que necesitamos para resolver sistemas grandes, no se requiere tanta exactitud.

Aprovechando que en las redes eléctricas, hay ciertas relaciones que se mueven en márgenes estrechos, se van a simplificar las expresiones anteriores.

La siguiente tabla muestra la relación L/N para todas las redes (salvo la de 13 nudos que no es representativa):

N	14	26	29	30	38	39	57	118
L/N	1.42	1.23	1.31	1.36	1.28	1.18	1.36	1.51

Es decir, la relación R entre el número de líneas y el número de nudos, se mueve aproximadamente entre 1.2 y 1.5, y ésto es válido prácticamente para todas las redes reales.

Veámos también cuánto vale la relación B'/L, es decir, el cociente entre el número de elementos de la matriz B' después y antes de la triangularización, que da una idea del llenado de la matriz B':

N	14	26	29	30	38	39	57	118
B'/L	1.22	1.15	1.23	1.33	1.18	1.46	1.74	1.46

Nuevamente el margen de variación es pequeño, lo que confirma la importancia de una adecuada ordenación.

A la vista de las tablas anteriores, no es descabellado hacer:

$$\frac{B'}{L} \simeq \frac{L}{N} = R \qquad (6.7)$$

que además tiene cierta lógica, pués a mayor número de líneas por nudo (L/N), mayor es también el llenado de la matriz (B'/L), siendo el coeficiente de proporcionalidad aproximadamente 1, como se desprende de dichas tablas.

Por tanto, la relación B'/N, que aparece en ciertas expresiones de la figura 6.27, valdrá

$$\frac{B'}{N} = \frac{B'}{L} \cdot \frac{L}{N} \simeq R^2 \qquad (6.8)$$

Por los mismos motivos, la expresión $B''/(N-M)$, se puede aproximar por:

$$\frac{B''}{N-M} \simeq \frac{B'}{N} \simeq R^2 \qquad (6.9)$$

Por último, y para obtener una ecuación más compacta, el número M (nudos P-V), lo podemos substituir por 0.1N.

Con las aproximaciones (6.7), (6.8) y (6.9), es factible obtener una ecuación que relacione el número de operaciones, con el número de nudos, en función de la relación R (L/N).

El total de operaciones, de cualquier tipo, para la factorización de B' y B" es entonces:

$$O_F = 1.9(1 + R + R^2 + R^4)N \qquad (6.10)$$

y para una iteración completa:

$$O_I = (13.5 + 33.4R + 7.6R^2)N \qquad (6.11)$$

Como ejemplo, para la red de 30 nudos

$N = 30$

$L = 41$

$R = 1.36$

$O_F = 439$

$O_I = 2199$

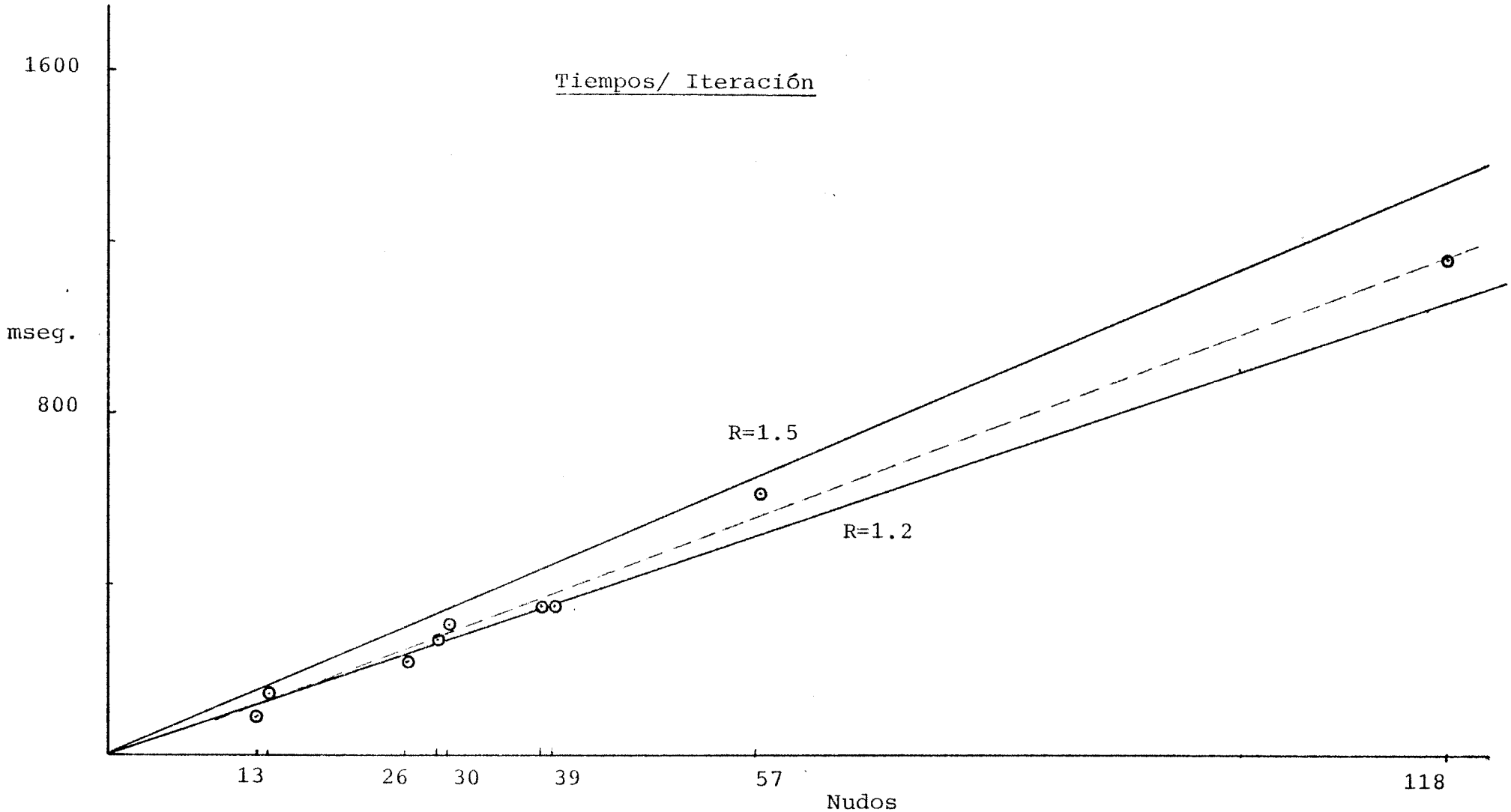

Tiempos/ Iteración
1600
mseg.
800
R=1.5
R=1.2
13
26
30
39
57
118
Nudos

fig. 6.28

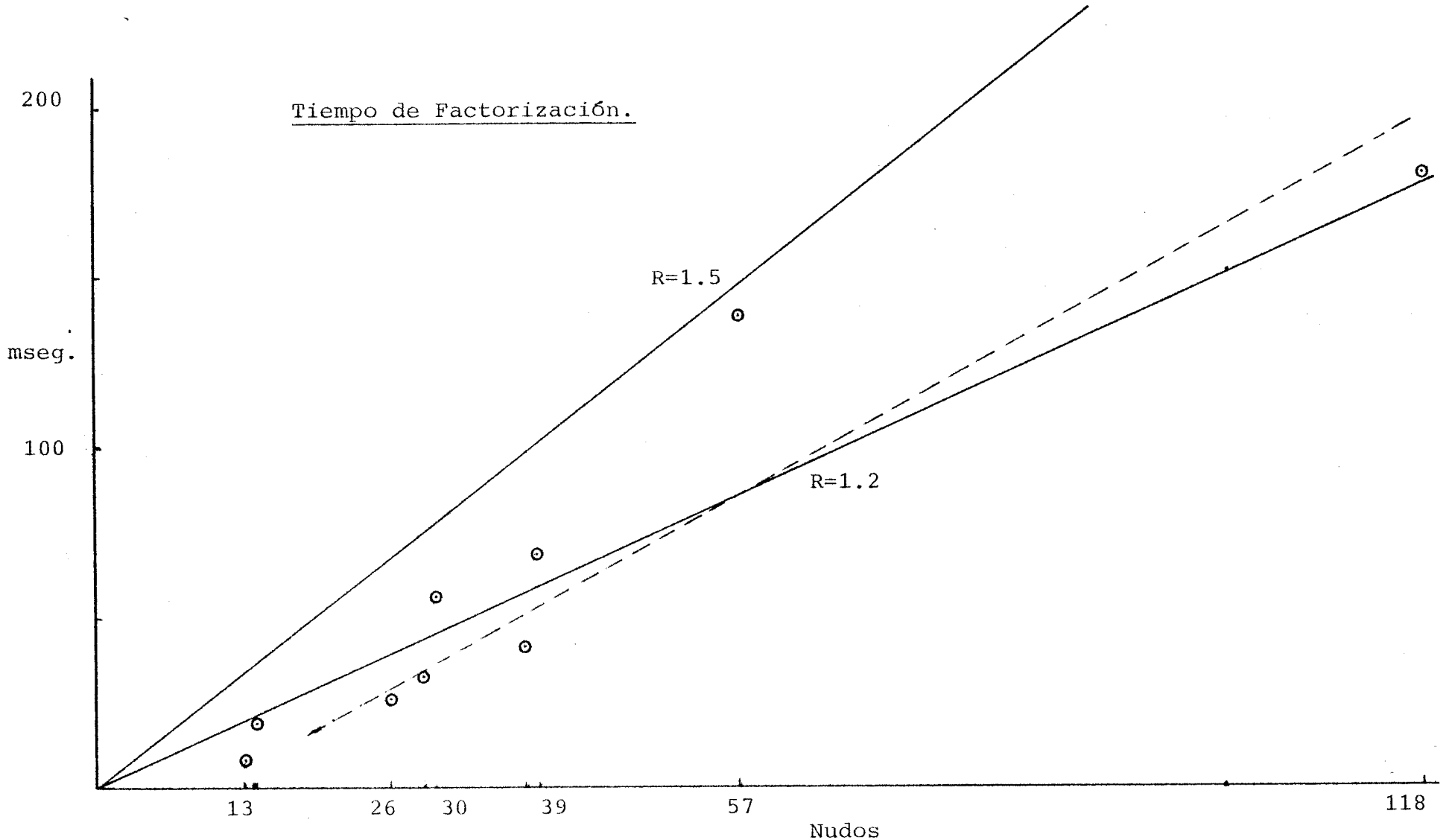

Tiempo de Factorización.
200
100
mseg.
R=1.5
R=1.2
13
26
30
39
57
118
Nudos

fig. 6.29

que a una media de 140 μseg por operación, da unos tiempos de 61 y 307 mseg. frente a 56 y 314 medidos (fig. 6.20).

Las ecuaciones (6.10) y (6.11), aún sin ser exactas, son muy importantes. Puesto que R se mantiene sensiblemente constante con N, indican que el esfuerzo de cálculo crece linealmente con el número de nudos.

En las figuras 6.28 y 6.29 se muestran las bandas de los tiempos de cálculo comprendidas entre R=1.2 y R=1.5, para la factorización y para 1 iteración, así como los tiempos medidos para las 9 redes. Las líneas de puntos son el ajuste lineal, y en ambos casos caen dentro de la banda, para N suficientemente grande.

Se pueden obtener algunas informaciones más de esas ecuaciones. La primera es que una iteración completa, requiere entre 4 y 6 veces el tiempo que se tarda en factorizar B' y B". Ello indica que puede ser interesante factorizar otra vez B' o B", si con ello ahorramos aunque sólo sea media iteración (en realidad, depende en gran medida de lo que se tarde en volver a formar la matriz, en lo que no hemos entrado).

La otra información interesante es cómo se distribuyen los tiempos dentro de una iteración.

Las operaciones O_I se descomponen en los siguientes items:

Cálculo $\Delta P/V$ y $\Delta Q/V$: $(5.7+33.4R)N$

Aplicación tabla factores B',B": $(4+7.6R^2)N$

Estudio convergencia, actualización V, θ: 3.8N

Del total de O_I, la parte más importante es el cálculo de los residuos de las potencias, que suponen entre un 69 y un 71% de ese total, lo que concuerda con los resultados experimentales de otros autores [55].

Por tanto, el método simplificado propuesto en 5.6, requiere menos tiempo que el convencional, aunque no ahorre iteraciones, puesto que evita calcular $\Delta P/V$ en la primera iteración.

7. CONCLUSIONES.

7.1 RESUMEN Y CONCLUSIONES.

En este trabajo se ha abordado en profundidad el problema del reparto de cargas, y se ha propuesto una clasificación de los algoritmos existentes para la solución del mismo. Se han investigado intensivamente las técnicas de tratamiento de matrices vacías, de las que se ha sacado el mayor partido posible.

Se han realizado numerosos programas de ordenador, y se han analizado sus resultados sobre diferentes sistemas utilizados como test.

Como aportaciones de esta tesis se pueden señalar:

- Una simplificación del método desacoplado rápido, que en numerosas ocasiones ahorra tiempo.
- Un algoritmo para la descomposición automática de una red (o de cualquier grafo) en dos partes sensiblemente iguales, con poca interconexión, pudiéndose extender recursivamente.
- La resolución en paralelo del sistema de ecuaciones resultante, utilizando para ello dos microprocesadores de 16 bits sobre un bus estandard, intercambiando sus datos mediante una memoria común.
- Una estimación teórico-práctica del esfuerzo de cálculo necesario, demostrándose que crece linealmente con el número de nudos de la red.

7.2 SUGERENCIAS PARA FUTUROS DESARROLLOS.

Dado lo clásico del problema que se está tratando, puede parecer que está agotado, y sin embargo no es así, como ha pretendido demostrar este trabajo.

Futuras investigaciones deben encaminarse hacia:

- Tratamiento más eficiente de los ajustes a la solución, que actualmente aumentan excesivamente el número de iteraciones.
- Una mayor atención al tratamiento de bases de datos que, como en este caso, se comparten para diferentes utilidades, y que a veces suponen una limitación más importante, respecto al tiempo total, que el propio algoritmo.
- Acercamiento entre los planteamientos teóricos y prácticos encaminado fundamentalmente a eliminar los problemas de convergencia que a veces se presentan, especialmente en la etapa de planificación, donde se resuelven sistemas que no representan situaciones prácticas.
- Un mayor aprovechamiento de las nuevas tecnologías, que ya ofrece la posibilidad de que cada usuario diseñe circuitos integrados a su medida. En esta línea es de esperar numerosas aportaciones en un futuro próximo.
- Por último, y en el ámbito concreto de los trabajos de esta tesis, queda por determinar qué mejoras se deben introducir en el algoritmo de descomposición de los

grafos, para que ofrezca mejores resultados en determinados casos (algunas ya se han apuntado).

BIBLIOGRAFIA.

[1] - Adibi M.M. et al., Solution methods for transient and dynamic stability. Procee. IEEE vol. 62 pg. 951-958, 1974.

[2] - Allan R.N., Arruda C., LTC transformers and MVAR violations in the fast decoupled load flow. PAS-101 pg 3328-3332, 1982.

[3] - Alvarado F.L., Enns M.K., Tinney W.F., Sparsity enhancement in mutually coupled networks. PAS-103 pg. 1502-1508, 1984.

[4] - Arapostathis A., Sastry S., Varaiya P., Analysis of power flow equation. Electrical Power & Energy Systems pg. 115-126, 1981.

[5] - Arapostathis A., Varaiya P., Behavior of three node power networks. Electrical Power & Energy Systems pg.22-30, 1980.

[6] - Babic B.S., Decoupled load flow with variables in rectangular form. IEE Procee. vol. C pg. 98-110, 1983.

[7] - Baillieul J.,Byrnes C.I., Washburn R.B., An algebraic--geometric and topological analysis of the solution to the load flow equations for a power system. 20th IEEE Confer. on Decision and Control pg. 1312-1320, 1981.

[8] - Baillieul J., Byrnes C.I., Geometric critical point analysis of lossless power systems models. CAS-29 pg. 724-737, 1982.

[9] - Baillieul J., Byrnes C.I., Remarks on the number of solutions to the load flow equations for a power system with electrical losses. Comunicación privada con los autores, 1982.

[10] - Baillieul J., Byrnes C.I., The load flow equations for a 3 node electrical power system. Systems & Control letters pg.321-329, North-Holland, 1983.

[11] - Baillieul J., Byrnes C.I., The singularity theory of the load flow equations for a 3 node electrical power system. Systems & Control letters pg. 330-340 North-Holland, 1983.

[12] - Baumann R., Some new aspects on load flow calculation. PAS-85, pg. 1164-1176, 1966.

[13] - Braess D., Grebe E., A numerical analysis of load flow calculation methods. PAS-100, pg. 3642-3647, 1981.

[14] - Brameller A., Denmead J.K., Some improved methods of digital network analysis. Procee. IEE vol. 109, pg. 109-116, 1962.

[15] - Brayton R.K. et al., Some results on sparse matrices. Math. Comput. vol. 24, pg. 937-954, 1970.

[16] - Britton J.P., Improved area interchange control for Newton's method load flows. PAS-88, pg. 1577-1581, 1969.

[17] - Britton J.P., Improved load flow performance through a more general equation form. PAS-90, pg. 109-116, 1971.

[18] - Brown H.E., Solution of large networks by matrix methods. John Wiley & Sons. New York 1975.

[19] - Brown H.E., Carter G.K., Happ H.H., Person C.E., Power flow solution by impedance matrix iterative method. PAS--82, pg. 1-10, 1963.

[20] - Brown R.J., Tinney W.F., Digital solutions for large power networks. PAS-76, pg. 347-355, 1957.

[21] - Bunch J.R., Rose D.J., Partitioning, tearing and modification of sparse linear systems. J. Math. Anal. App. vol. 48, pg. 575-593, 1974.

[22] - Cuthill E., Several strategies for reducing the bandwidth of matrices. En [84], pg. 157-166.

[23] - Cuthill E., McKee J., Reducing the bandwidth of sparse symmetric matrices. Proc. 24th National Conf. ACM. pg. 157-172, 1969.

[24] - Dembart B., Erisman A.M., Hybrid sparse matrix methods. CT-20, pg. 641-649, 1973.

[25] - Despotovic S.T., Babic B.S., Mastilovic V.P., A rapid and reliable method for solving load flow problems. PAS-90, pg. 123-130, 1971.

[26] - Dommel H.W., Tinney W.F., Optimal power flow solutions. PAS-87, pg. 1866-1874, 1968.

[27] - Duff I.S., Erisman A.M., Reid J.K., On George's nested dissection method. SIAM J.N. Anal. vol. 13, pg. 686-695, 1976.

[28] - Duff I.S., A survey of sparse matrix research. Procee. IEEE, pg. 500-535, 1977.

[29] - Dusonchet Y.P. et al., Load flows using a combination of point Jacobi and Newton's method. PAS-90, pg. 941-949, 1971.

[30] - Dy Liacco T.E., Real-Time computer control of power systems. Proc. IEEE vol. 62, pg. 884-891, 1974.

[31] - Dy Liacco T.E., System security: The computer's role. Spectrum IEEE, pg. 43-50, Junio 1978.

[32] - Dy Liacco T.E., Ramarao, Discusión de [122].

[33] - Elgerd O., Electric Energy Systems Theory. Mc Graw-Hill, 1983.

[34] - El-Hawary M.E., Wellon O.K., The alpha-modified quasi-second order Newton Raphson method for load flow solutions in rectangular form. PAS-101 pg. 854-866, 1982.

[35] - Freris L.L., Sasson A.M., Investigation of the load flow problem. Procee. IEE-115, pg. 1459-1470, 1968.

[36] - Galloway R.H. et al., New aproach to power system load flow analysis in a digital computer. Procee. IEE-117, pg. 165-169, 1970.

[37] - George J.A., Computer implementation of the finite element method. Ph. D. dissertation, Standford University, 1971.

[38] - George J.A., An efficient band oriented scheme for solving nxn grid problems. Proc. Fall Joint Comput. Conf., pg. 1317-1320, 1972.

[39] - George J.A., Nested disecction of a regular finite element mesh. SIAM J. Num. Anal. vol. 10, pg. 345-363, 1973.

[40] - George J.A., On block elimination for sparse linear systems. SIAM J. N. Anal. vol. 11, pg. 585-603, 1974.

[41] - George J.A., An automatic one-way disecction algorithm for irregular finite element problems. Lecture Notes in Math. No. 630, Springer-Verlag, pg. 78-89, 1978.

[42] - George J.A., Liu W.H., Algorithms for matrix partitioning and the numerical solution of finite element systems. SIAM J.N. Anal. vol. 15, pg. 297-327, 1978.

[43] - George J.A., Liu W.H., An automatic nested dissection algorithm for irregular finite element problems. SIAM J.N.

Anal. vol. 15, pg. 1053-1069, 1978.

[44] - Gibbs N.E., Poole W.G., Stockmeyer P.K., An algorithm for reducing the bandwidth and profile of a sparse matrix. SIAM J. N. Anal. vol. 13, pg. 236-250, 1976.

[45] - Glimn A.F., Stagg G.W., Automatic calculation of load flows. PAS-76, pg. 817-825, 1957.

[46] - Gupta P.P., Davies M.W.H., Digital computers in power system analysis. Procee. IEE-108, pg. 383-404, 1961.

[47] - Gustavson F.G., Some basic techniques for solving sparse systems of linear equations. En [84], pg. 41-52.

[48] - Hajj I.N., Sparsity considerations in network solution by tearing. CAS-27, pg. 357-366, 1980.

[49] - Hale H.W., Goodrich R., Digital computation of power flow - some new aspects. PAS-78, pg.919-924, 1959.

[50] - Haley P.H., Ayres M., Super decoupled load flow with distributed slack bus. PAS-104, pg. 104-113, 1985.

[51] - Happ H.H., Z Diakoptics: Torn divisions radially attached. PAS-86 pg. 751-769, 1967.

[52] - Happ H.H., Diakoptics and piecewise methods. PAS-89, pg. 1373-1382, 1970.

[53] - Happ H.H., Diakoptics: The solution of system problems by tearing. Procee. IEEE vol. 62, pg. 930-940, 1974.

[54] - Hobson E., Discusión de [103], PAS-93, pg. 868, 1974.

[55] - Hulskamp J.P., Chan S.M., Fazio J.F., Power flow outage studies using an array processor. PAS-101, pg. 254-261, 1982.

[56] - IEEE Commitee Report. IEEE reliability tests systems. PAS-98, pg. 2047-2054, 1979.

[57] - Iwamoto S., Tamura Y., A fast load flow method retaining nonlinearity. PAS-97, pg. 1586-1599, 1978.

[58] - Jennings A., Matrix computation for engineers and scientists. John Wiley & Sons, 1977.

[59] - Jordan R.H., Rapid converging digital load flow. PAS-76, pg. 1433-1438, 1957.

[60] - King I.P., An automatic reordering scheme for simultaneous equations derived from network systems. Int. J. Num. Meth. Eng. vol. 2, pg. 523-533, 1970.

[61] - Knight U.G. et al., Computers in power system planning. Procee. IEEE vol. 62, pg. 872-883, 1974.

[62] - Korsak A.J., On the question of uniqueness of stable load flow solutions. PAS-91, pg. 1093-1100, 1972.

[63] - Laha A.K., Bollinger K.E., Billington R., Dhar S.B., Modified form of Newton's method for faster load-flow solutions. Procee. IEE-121, pg. 849-853, 1974.

[64] - Laughton M.A., Davies M.W.H., Numerical techniques in solution of power systems load flow problems. Procee. IEE--111, pg. 1575-1588, 1964.

[65] - Levy R., Restructuring of the structural stiffness matrix to improve computational efficiency. Jet Propulsion Lab. Tech. Rev. vol. 1, pg. 61-70, 1971.

[66] - Lewis J.G., Poole W.G., Ordering algorithms applied to sparse matrices in electrical power problems. Procee. of SIAM on Electric power problems: The mathematical challenge. pg. 115-126, 1980.

[67] - Liu W.H., Sherman A.H., Comparative analysis of the Cuthill-McKee and the reverse Cuthill-McKee ordering algorithms for sparse matrices. SIAM J.N. Anal. vol. 13, pg.

198-213.

[68] - Mamandur K.R.C., Berg G.J., Automatic adjustment of generator voltages in Newton-Raphson method of power flow solutions. PAS-101, pg. 1400-1409, 1982.

[69] - Mamandur K.R.C., Berg G.J., Efficient simulation of line and transformer outages in power systems. PAS-101, pg. 3733-3741, 1982.

[70] - Markowitz H.M., The elimination form of the inverse and its application to linear programming. Manag. Sci. vol. 3, pg. 255-269, 1957.

[71] - Medicherla T.K.P., Sachdev M.S., A second order decoupled load flow technique. Summer meeting of the IEEE Power Eng. Society. Portland, 1976.

[72] - Nagendra P.S., Prakasa K.S., Nanda J., On the convergence characteristics and uniqueness of solution of a new load flow method. Procee. of the Symp. on Comp. applic. in large scale power systems. pg. 197-203, IFAC India, 1979.

[73] - Nagendra P.S., Prakasa K.S., Nanda J., An exact fast load flow method including second order terms in rectangular coordinates. PAS-101, pg. 3261-3268, 1982.

[74] - Nagendra P.S., Prakasa K.S., Nanda J., An empirical criterion for the convergence of the fast decoupled load flow method. PAS-103, pg. 974-981, 1984.

[75] - Ogbuobiri E.C., Tinney W.F., Walker J.W., Sparsity directed decomposition for Gaussian elimination on matrices. PAS-89, pg. 141, 1970.

[76] - Ortega J.M., Numerical Analysis. Academic Press, 1972.

[77] - Pai M.A., et al. A fast algorithm for on line load flow and contingency evaluation. Proc. of the Symp. on compu-

ters applic. in large scale power systems. IFAC, pg. 132-138, India 1979.

[78] - Pai M.A., Computer techniques in power system analysis. Mc Graw-hill, New Delhi, 1979.

[79] - Peterson N.M., Meyer W.S., Automatic adjustment of transformer and phase-shifter taps in the Newton power flow. PAS-90, pg. 103-108, 1971.

[80] - Peterson N.M., Tinney W.F., Bree D.W., Iterative linear ac power flow solution for fast approximate outage studies. PAS-91, pg. 2048-2056, 1972.

[81] - Podmore R. et al., An advanced dispatcher training simulator. PAS-101, pg. 17-25, 1982.

[82] - Pritchard R., Pottle C., High-speed power flows using attached scientific ("array") processors. PAS-101, pg. 249-253, 1982.

[83] - Reid J.K., A survey of sparse matrix computation. Proc. of SIAM on electric power problems: The mathematical challenge. pg. 41-68, 1980.

[84] - Rose D.J., Willoughby R.A., Sparse matrices and their applications. Plenum Press, 1972.

[85] - Roy L., Discusión de [57], PAS-97, pg. 1595-1596, 1978.

[86] - Roy L., Exact second order load flow. Procee. 6th. power system computational conference. West Germany, pg. 711-718, 1978.

[87] - Sachdev M.S., Ibrahim S.A., A fast approximate technique for outage studies in power system planning and operation. PAS-93, pg. 1133-1142, 1974.

[88] - Sachdev M.S., Ibrahim S.A., A modified Newton Raphson load flow technique and its use in simulating line and

transformers outages. Procee. of the Symp. on computer applic. in large scale power systems. IFAC, pg. 126-131, India 1979.

[89] - Sachdev M.S., Medicherla T.K.P., A second order load flow technique. PAS-96, pg. 189-197, 1977.

[90] - Sangiovanni-Vincentelli A. et al., An efficient heuristic cluster algorithm for tearing large scale networks. CAS--24, pg. 709-717, 1977.

[91] - Sasson A.M., Jaimes F.J., Digital methods applied to power flow studies. PAS-86, pg. 860-867, 1967.

[92] - Sasson A.M., Treviño C., Aboytes F., Improved Newton's load flow through a minimization technique. IEEE Pica Conf. Boston, 1971.

[93] - Sato N., Tinney W.F., Techniques for exploiting the sparsity of the network admittance matrix. PAS-82, pg. 944-950, 1963.

[94] - Sauer P.W., Explicit load flow series and functions. PAS--100, pg. 3754-3761, 1981.

[95] - Schweppe F.C., Handschin E.J., Static state estimation in electric power systems. Procee. IEEE vol. 62, pg. 972-982, 1974.

[96] - Shanblatt M.A., Leung Y.Y.J., The promise of VLSI load flow computation enhancement. PAS-103, pg. 1714-1719, 1984.

[97] - Sherman J., Morrison W.J., Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. Ann. Math. Stat. vol. 21, pg. 124, 1950.

[98] - Smith et al., Sparse solution using hash storage. PAS-91, pg. 1396-1404, 1972.

[99] - Stagg G.W., El Abiad A.H., Computer methods in power system analysis. Mc Graw- Hill, 1968.

[100] - Stott B., Solution of large power system networks by ordered elimination: a comparison of ordering schemes. Procee. IEEE, pg 125-134, 1971.

[101] - Stott B., Effective starting process for Newton Raphson load flows. Procee. IEE-118, pg. 983-987, 1971.

[102] - Stott B., Decoupled Newton load flow. PAS-91, pg. 1955-1959, 1972.

[103] - Stott B., Alsac O., Fast decoupled load flow. PAS-93, pg. 859, 1974.

[104] - Stott B., Review of load flow calculation methods. Procee. IEEE, vol. 62, pg. 916-929, 1974.

[105] - Stott B., Comments on "Review of load flow calculation methods". Procee. Letters pg. 712-715, Procee. IEEE 1975.

[106] - Subramanian D.K., Fast decoupled load flow solution by a higher order recursive algorithm. Proc. of the Symp. on Computer applica. in large scale power systems. IFAC, India pg. 193-196, 1979.

[107] - Távora C.J., Smith O.J.M., Equilibrium analysis of power systems. PAS-91, pg. 1131-1137, 1972.

[108] - Tinney W.F., Hart C.E., Power flow solution by Newton's method. PAS-86 pg. 1449-1460, 1967.

[109] - Tinney W.F., Meyer W.S., Solution of large sparse systems by ordered triangular factorization. AC-18, pg. 333-345, 1973.

[110] - Tinney W.F., Walker J.W., Direct solutions of sparse network equations by optimally ordered triangular factorization. Procee. IEEE vol. 55, pg. 1801-1809, 1967.

[111] - Tinney W.F., Compensation methods for networks solutions by optimally ordered triangular factorization. PAS-91, pg. 123-127, 1972.

[112] - Tripathy S.C. et al., Load-flow solutions for ill-conditioned power systems by a Newton-like method. PAS-101, pg 3648-3657, 1982.

[113] - Undrill J.M. et al. Interactive computation in power system analysis. Proc. IEEE, vol. 62, pg. 1009-1018, 1974.

[114] - Van Ness J.E., Iteration methods for digital load flow studies. PAS=78, pg. 583-588, 1959.

[115] - Van Ness J.E., Convergence of iterative load flow studies. PAS-78B, pg.1590-1595, 1960.

[116] - Van Ness J.E., Griffin J.H., Elimination methods for load flow studies. PAS-80, pg. 299-304, 1961.

[117] - Wallach Y. et al., Improved methods for load flow calculations. PAS-90 pg. 116-112, 1971.

[118] - Wallach Y., Konrad V., On block-parallel methods for solving linear equations. C-29, pg. 354-359, 1980.

[119] - Wamser R.J., Slutsker I.W., Power flow solution by the Newton Raphson method in transient stability studies. PAS-103, pg. 2299-2307, 1984.

[120] - Ward J.B., Hale H.W., Digital computer solution of power flow problems. PAS-75, pg. 398-404, 1956.

[121] - Weedy B.M., Electric Power Systems, John Wiley & Sons, New York, 1972.

[122] - Westlake J.R., Numerical matrix inversion and solution of linear equations John Wiley & Sons, 1968.

[123] - Wing O., Huang J.W., A computational model for parallel solution of linear equations. C-29, pg. 632-638, 1980.

[124] - Wu F.F., Theoretical study of the convergence of the fast decoupled load flow. PAS-96 pg. 268-275, 1977.

APENDICE 1. Modelo general para enlaces entre nudos.

En vista de los modelos que se han desarrollado por separado para líneas, transformadores, transformadores con tomas, y transformadores de desplazamiento de fase (estos últimos introducen asimetría en la matriz de admitancias), parece necesario desarrollar un modelo más general que sea capaz de describir, particularizándolo, cualquier enlace entre dos nudos.

Dicho modelo, como se indica en [17], podría ser una admitancia, en serie con un desfasador de fase perfecto y con un autotransformador perfecto, así

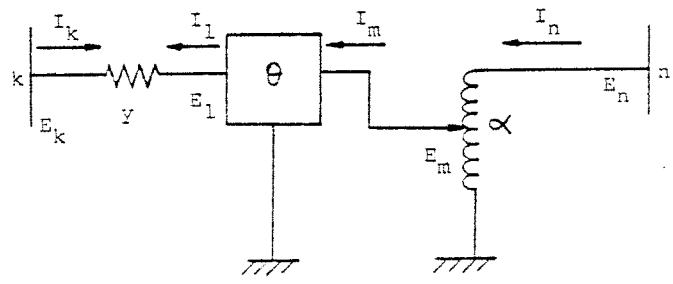


fig. A1.1

Se cumplen las siguientes relaciones:

$$\begin{aligned} I_l &= y(E_l - E_k) \\ E_l I_l^* &= E_m I_m^* \\ E_m I_m^* &= E_n I_n^* \end{aligned} \qquad \text{(A1.1)}$$

y también

$$\begin{aligned} E_l &= E_m \cdot 1\angle\theta \\ E_m &= E_n \end{aligned} \qquad \text{(A1.2)}$$

Sustituyendo en

$$I_k = y(E_k - E_1) \tag{A1.3}$$

las (A1.2) nos quedan:

$$I_k = yE_k - y(\alpha \angle\vartheta)E_n \tag{A1.4}$$

y de (A1.1) obtenemos operando

$$I_n^* = \frac{E_m}{E_n} \cdot \left[\frac{E_1}{E_m} (y\,E_1 - yE_k)^* \right] \tag{A1.5}$$

sustituyendo otra vez las (A1.2) en (A1.5)

$$I_n = \alpha \angle -\vartheta \cdot y(E_1 - E_k) \tag{A1.6}$$

y teniendo otra vez en cuenta las (A1.2) en (A1.6)

$$I_n = - y(\alpha \angle -\vartheta)E_k + \alpha^2\, y\, E_n \tag{A1.7}$$

Las ecuaciones (A1.4) y (A1.7), que son las que buscábamos, quedan, en forma matricial:

$$\begin{bmatrix} I_k \\ I_n \end{bmatrix} = \begin{bmatrix} y & -\alpha y \cdot (1 \angle\vartheta) \\ -\alpha\, y(1 \angle -\vartheta) & \alpha^2 y \end{bmatrix} \cdot \begin{bmatrix} E_k \\ E_n \end{bmatrix} \tag{A1.8}$$

donde

ϑ : desplazamiento de fase $\angle E_1 = \angle E_m + \vartheta$

α : relación de transformación $E_m = \alpha E_n$

y : admitancia serie total del enlace entre nudos.

En las ecuaciones de flujo de cargas, y en otras situaciones, nos interesará uno de los dos extremos aisladamente. La ecuación (A1.8) se ha obtenido considerando que el nudo K era el de las tomas, y no era la referencia para los desplazamientos de fase, que era el n. Llamemos en general, i y j a las dos barras unidas por el elemento considerado. Si introducimos los coeficientes $\mathcal{E}$ y $\mathcal{E}'$ con el siguiente significado:

$$\mathcal{E} = \begin{cases} +1 & : \text{si la barra i es el lado que no tiene tomas} \\ 0 & : \text{si la barra i es el lado de las tomas} \end{cases}$$

$$\mathcal{E}' = \begin{cases} 0 & : \text{si no hay desplazamiento de fase} \\ +1 & : \text{si el nudo i es la referencia para el desplaz..} \\ -1 & : \text{si el nudo j es la referencia para el desplaz.} \end{cases}$$

podemos considerar que la siguiente ecuación:

$$I_i = y \left[1 + \mathcal{E}(\alpha^2 - 1) \right] E_i - y(\alpha \underline{/-\mathcal{E}'\vartheta}) E_j \qquad \text{(A1.9)}$$

es totalmente general con los valores apropiados de α, ϑ, $\mathcal{E}$ y $\mathcal{E}'$. Compruébese que con $\mathcal{E}=0$, $\mathcal{E}'=-1$ obtenemos la primera de las ecuaciones de (A1.8), y con $\mathcal{E}=+1$, $\mathcal{E}'=+1$ obtenemos la segunda de ellas.

La ecuación (A1.9) vista desde el lado i equivaldría a

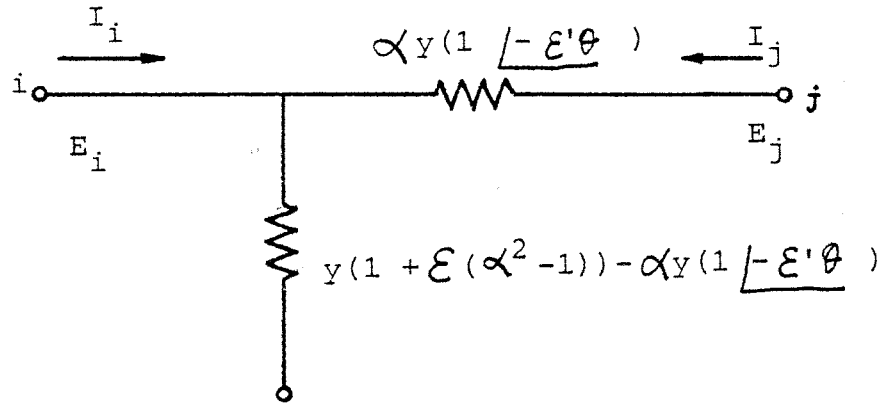


fig. A1.2

α y ϑ serán variables o constantes dependiendo del caso.

Con estas ideas, cabe generalizar las ecuaciones que imponen restricciones al calcular los flujos de carga, es decir, aquellas que relacionan las tensiones e intensidades en los nudos, con P y Q:

$$S_i = \sum_j k_{ij} \cdot E_i \cdot I_{ij}^* \qquad \text{(A1.10)}$$

Precisamente las constantes k_{ij}, generalizan las restricciones consideradas como normales clásicamente con $k_{ij}=1$, permitiendo la introducción de tipos de nudos diferentes a los P-Q, P-V (que son los más sencillos de tratar).

$k_{ij}, \varepsilon, \varepsilon', \alpha, \vartheta$ son los parámetros que dan generalidad a este enfoque.

Podemos descomponer (A1.10) en sus partes real e imaginaria

$$P_i = \sum_j k_{ij}\left[\left[1+\varepsilon_{ij}(\alpha_{ij}^2-1)\right] g_{ij}E_i^2 - Y_{ij}\cdot\alpha_{ij}\cdot E_i\cdot E_j\cdot \right.$$
$$\left.\cdot\cos(\delta_i-\delta_j-\beta_{ij}+\varepsilon'_{ij}\theta_{ij})\right] \qquad \text{(A1.11)}$$
$$Q_i = \sum_j k_{ij}\left[-(1+\varepsilon_{ij}(\alpha_{ij}^2-1))b_{ij}\cdot E_i^2 - Y_{ij}\cdot\alpha_{ij}\cdot E_i\cdot E_j\cdot \right.$$
$$\left.\cdot\operatorname{Sen}(\delta_i-\delta_j-\beta_{ij}+\varepsilon'_{ij}\cdot\theta_{ij})\right]$$

donde

$$E = E\angle\delta = e + jf$$
$$y = Y\angle\beta = g + jb$$

Con las ecuaciones (A1.11) se puede modelar una gran cantidad de problemas interesantes, de los que aparecen con frecuencia en los flujos de carga.

APENDICE 2. Método de Newton-Raphson y valores de los coeficientes para el algoritmo desacoplado rápido.

A2.1 Método de Newton Raphson.

Dada una ecuación no lineal en una sola variable

$$f(x)=0 \tag{A2.1}$$

para cuya solución disponemos de una estimación inicial x^o, se obtiene un mejor valor x^1 linealizando f(x) alrededor de x^o

$$f(x^o) + (x-x^o) \cdot f'(x^o)= 0 \tag{A2.2}$$

de donde

$$x^1=x^o-f(x^o)/f'(x^o) \tag{A2.3}$$

Generalizando la (A2.3) para la iteración m-ésima:

$$x^{m+1} = x^m - f(x^m)/f'(x^m) \tag{A2.4}$$

En el caso n-dimensional la (A2.4) se convierte en

$$X^{m+1} = X^m - \left[F'(X^m)\right]^{-1} \cdot f(x^m) \tag{A2.5}$$

donde F'(x) es el jacobiano de f(x), denominado simplemente J. La ecuación (A2.5) es equivalente a las dos siguientes

$$f(X^m) = - J^m . \Delta X^m$$
$$X^{m+1} = X^m + \Delta X^m \qquad \text{(A2.6)}$$

Las ecuaciones del flujo de cargas en coordenadas polares, puestas en la forma (A2.1) son:

$$P_i^{esp} - \sum_{j=1}^{n} V_i V_j (G_{ij} \text{Cos}\, \vartheta_{ij} + B_{ij}\, \text{Sen}\, \vartheta_{ij}) = 0 \qquad \text{(A2.7)}$$
$$i = 2 \ldots n$$

$$Q_i^{esp} - \sum_{j=1}^{n} V_i V_j (G_{ij} \text{Sen}\, \vartheta_{ij} - B_{ij}\, \text{Cos}\, \vartheta_{ij}) = 0 \qquad \text{(A2.8)}$$
$$i = m+1 \ldots n$$

habiéndose supuesto que el nudo 1 es el de referencia, del 2 al m los P–V y del m+1 al n los P–Q. En forma compacta:

$$\Delta P_i = 0 \qquad i = 2 \ldots n$$
$$\Delta Q_i = 0 \qquad i = m+1 \ldots n \qquad \text{(A2.9)}$$

El vector de incógnitas es:

$$X^t = (\vartheta_2, \vartheta_3 \ldots \vartheta_n, V_{m+1} \ldots V_n)$$

y, finalmente, las ecuaciones equivalentes a las (A2.6):

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix}^m = \begin{bmatrix} H & N \\ M & L \end{bmatrix}^m \cdot \begin{bmatrix} \Delta \vartheta \\ \Delta V \\ \hline V \end{bmatrix}^m \qquad \text{(A2.10)}$$

$$\begin{bmatrix} \theta \\ V \end{bmatrix}^{m+1} = \begin{bmatrix} \theta \\ V \end{bmatrix}^{m} + \begin{bmatrix} \Delta\theta \\ \Delta V \end{bmatrix}^{m} \qquad \text{(A2.11)}$$

El poner $\Delta V/V$ en lugar de ΔV simplifica los términos del jacobiano, que quedan:

Para $i \neq j$:

$$\begin{aligned} H_{ij} &= L_{ij} = V_i V_j (G_{ij} \text{Sen}\, \theta_{ij} - B_{ij} \text{Cos}\, \theta_{ij}) \\ N_{ij} &= -M_{ij} = V_i V_j (G_{ij} \text{Cos}\, \theta_{ij} + B_{ij} \text{Sen}\, \theta_{ij}) \end{aligned} \qquad \text{(A2.12)}$$

Para $i=j$:

$$\begin{aligned} H_{ii} &= -Q_i - B_{ii} V_i^2 \\ L_{ii} &= Q_i - B_{ii} V_i^2 \\ N_{ii} &= P_i + G_{ii} V_i^2 \\ M_{ii} &= P_i - G_{ii} V_i^2 \end{aligned} \qquad \text{(A.13)}$$

El proceso a seguir sería:

1) Se estima un valor inicial (véase capítulo 3).
2) Se calcula $\Delta P, \Delta Q$ para las tensiones supuestas.
3) Si $\Delta P, \Delta Q$ están dentro de una tolerancia, fin del proceso.
4) Se calcula el jacobiano de (A2.12),(A2.13).
5) Se resuelve $\Delta\theta$, $\Delta V/V$ de (A2.10).

6) Mediante (A2.11) actualizamos el vector de estado y vamos al paso 2) nuevamente.

Las funciones del flujo de cargas en coordenadas cartesianas son

$$(V_i^{esp})^2 - (e_i^2 + f_i^2) = 0 \qquad i = 2...n \tag{A2.14}$$

$$P_i^{esp} - \sum_{j=1}^{n} e_i(e_j G_{ij} - f_j B_{ij}) + f_i(f_j G_{ij} + e_j B_{ij}) = 0 \qquad i = 2...n \tag{A2.15}$$

$$Q_i^{esp} - \sum_{j=1}^{n} f_i(e_j G_{ij} - f_j B_{ij}) - e_i(f_j G_{ij} + e_j B_{ij}) = 0 \qquad i = m+1...n \tag{A2.16}$$

Mediante el siguiente esquema iterativo:

$$\begin{bmatrix} \Delta P \\ \Delta Q \\ \Delta V^2 \end{bmatrix}^m = \begin{bmatrix} S & T \\ U & W \\ C & D \end{bmatrix}^m \cdot \begin{bmatrix} \Delta e \\ \Delta f \end{bmatrix}^m \tag{A2.17}$$

$$\begin{bmatrix} e \\ f \end{bmatrix}^{m+1} = \begin{bmatrix} e \\ f \end{bmatrix}^m + \begin{bmatrix} \Delta e \\ \Delta f \end{bmatrix}^m \tag{A2.18}$$

se obtiene el vector de estado [e,f], donde los términos del jacobiano valen:

Para $i \neq j$:

$$
\begin{aligned}
S_{ij} &= -W_{ij} = G_{ij}e_i + B_{ij}f_i \\
T_{ij} &= \quad U_{ij} = -B_{ij}e_i + G_{ij}f_i \\
C_{ij} &= D_{ij} = 0
\end{aligned}
\qquad \text{(A2.19)}
$$

Para i=j:

$$
\begin{aligned}
S_{ii} &= a_i + G_{ii}e_i + B_{ii}f_i \\
W_{ii} &= a_i - G_{ii}e_i - B_{ii}f_i \\
T_{ii} &= b_i - B_{ii}e_i + G_{ii}f_i \\
U_{ii} &= -b_i - B_{ii}e_i + G_{ii}f_i \\
C_{ii} &= 2e_i \\
D_{ii} &= 2f_i
\end{aligned}
\qquad \text{(A2.20)}
$$

siendo a_i, b_i las componentes de la corriente entrante al nudo i, o sea:

$$
a_i + jb_i = \sum_{k=1}^{n} (G_{ik} + jB_{ik})(e_k + jf_k) \qquad \text{(A2.21)}
$$

A2.2 COEFICIENTES DEL JACOBIANO EN EL METODO DESACOPLADO RAPIDO.

Este algoritmo fue descrito en el capítulo 2 como:

$$
\frac{\Delta P}{V} = B' \Delta\vartheta \qquad \text{(A2.22)}
$$

$$\frac{\triangle Q}{V} = B'' \triangle V \qquad \text{(A2.23)}$$

Los coeficientes de B' y B" son :

$$B'_{ij} = -\frac{1}{X_{ij}} \qquad \text{(A.24)}$$

$$B'_{ii} = \sum_{j} \frac{1}{X_{ij}}$$

$$B''_{ij} = -B_{ij} \qquad \text{(A2.25)}$$
$$B''_{ii} = -B_{ii}$$

donde X_{ij} es la reactancia de la línea que une i con j, y B_{ij} es la parte imaginaria del elemento ij-ésimo de la matriz de admitancias de nudos.

APENDICE 3. Procedimiento simplificado para el cálculo del factor de sensitividad entre V y Q.

El subproblema reactivo en el algoritmo del flujo de cargas desacoplado rápido viene dado por:

$$\triangle Q/V = B'' \cdot \triangle V \qquad (A3.1)$$

Si un nudo PV supera uno de sus límites de reactiva en la cantidad $\triangle Q$, la corrección a realizar en su tensión se calcula de

$$\triangle V = S \ . \ \triangle Q/V \qquad (A3.2)$$

El factor de sensitividad S se obtiene como se indica a continuación, supuesta conocida la tabla de factores de B", y sin necesidad de rehacerla.

$$B'' = L \ . \ U \qquad (A3.3)$$

La ecuación que define a S es, según se explica en el Capítulo 3,

$$(B'')^{+} \cdot \begin{bmatrix} X \\ \hline S \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \\ \hline 1 \end{bmatrix} \qquad (A3.4)$$

donde $(B'')^{+}$ es la matriz B" ampliada en la fila y columna del nudo en cuestión:

$$(B'')^{+} = \left[\begin{array}{c|c} B'' & C \\ \hline C^{t} & D \end{array}\right] \qquad (A3.5)$$

y X es la parte de la inversa de $(B'')^{+}$ que no necesitamos (sólo necesitamos el elemento de la diagonal S).

De (A3.4) y (A3.5) obtenemos:

$$\begin{aligned} B'' \cdot X + C \cdot S &= 0 \\ C^{t} \cdot X + D \cdot S &= 1 \end{aligned} \qquad (A3.6)$$

y eliminando la parte que no nos interesa X:

$$S = (D - C_{t} \ (B'')^{-1} C)^{-1} \qquad (A3.7)$$

En (A3.7) no es preciso realizar explícitamente la inversión de B" tal como se indica, sino que se utiliza la tabla de factores de B" (A3.3) y las mismas rutinas del algoritmo principal (eliminación hacia adelante y sustitución hacia atrás).

Se debe aprovechar también el hecho de que el vector C e
muy vacío.

APENDICE 4. Triangularización de matrices Simétricas.

Para las matrices simétricas sólo almacenamos la mitad superior, realizándose las operaciones exclusivamente con esa zona.

La factorización de las matrices simétricas se hace casi siempre mediante el algoritmo de Cholesky, descrito en 3.6, cuyas ecuaciones repetimos aquí:

$$u_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} u_{ki}^2 \right)^{1/2} \qquad \text{(A4.1)}$$

$$u_{ij} = \frac{1}{u_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} u_{kj}\, u_{ki} \right) \qquad \begin{matrix} j>i \\ i=1\ldots N \end{matrix} \qquad \text{(A4.2)}$$

Alternativamente, podemos ahorrar el cálculo de N raices cuadradas, si utilizamos la versión simétrica del algoritmo de Doolittle:

$$u_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} u_{kk} \cdot u_{ki}^2 \right)^{-1} \qquad \text{(A4.3)}$$

$$u_{ij} = \left(a_{ij} - \sum_{k=1}^{i-1} u_{kk} \cdot u_{ki}\, u_{kj} \right) \qquad \begin{matrix} j>i \\ i=1\ldots N \end{matrix} \qquad \text{(A4.4)}$$

en el que además se utiliza el inverso de los elementos diagonales, lo que sustituye la mayoría de las divisiones por multiplicaciones, que normalmente consumen menos tiempo.

N°de Operaciones (Matrices Llenas).

CHOLESKY.

	Factorización	Elimin. adelante	Sustituc. atrás
Raices 2	N	-	-
Divisiones	$N^2/2-N/2$	N	N
Multiplic.	$N^3/6-N/6$	$N^2/2-N/2$	$N^2/2-N/2$
Sum. o Res.	$N^3/6-N/6$	$N^2/2-N/2$	$N^2/2-N/2$

DOOLITTLE SIMETRICO.

Divisiones	N	-	-
Multiplic.	$N^3/6+N^2/2-2N/3$	$N^2/2+N/2$	$N^2/2+N/2-1$
Sum. o Res.	$N^3/6-N/6$	$N^2/2-N/2$	$N^2/2-N/2$

DOOLITTLE SIMETRICO MODIFICADO.

Divisiones	N	-	-
Multiplic.	$N^3/6+N^2-7N/6$	$N^2/2+N/2$	$N^2/2-N/2$
Sum. o Res.	$N^3/6-N/6$	$N^2/2-N/2$	$N^2/2-N/2$

fig. A4.1

La figura A4.1, muestra el número de operaciones que hay que realizar para factorizar una matriz simétrica llena, así como las que se requieren en las dos etapas de la solución del sistema de ecuaciones (hacia adelante y hacia atrás). Recuérdese que estas etapas se realizan cada vez que se quiere resolver el sistema con un vector independiente diferente, mientras que la factorización sólo se realiza una vez.

Los resultados de esa figura indican que hemos ahorrado N raices cuadradas, a cambio de N divisiones. Las otras diferencias entre ambos algoritmos se pueden eliminar sustituyendo u_{ii} por $1/u_{ii}$ en (A4.1) y (A4.2), como se ha hecho en (A4.3), (A4.4).

Para obtener los resultados de la figura A4.1 se tendrá en cuenta que los productos de la forma $u_{kk} \cdot u_{ki}$ de las ecuaciones (A4.3) y (A4.4) sólo se realizan una vez, en lugar de N-i+1 veces como podría parecer, lo que resulta inmediato si los sumatorios se realizan en el orden adecuado.

Supongamos que en lugar de la ecuación (A4.4) utilizamos la siguiente:

$$u_{ij} = u_{ii} \left(a_{ij} - \sum_{k=1}^{i-1} u_{kk} \cdot u_{ki} \cdot u_{kj} \right)_{j>i} \qquad (A4.5)$$

$$i = 1...N$$

lo que equivale a multiplicar todos los elementos no diagonales por su diagonal respectiva.

Como indica la tabla inferior de la figura A4.1 ello supone incrementar en $N^2/2 - N/2$ el número de multiplicaciones en la factorización, y ahorrar N-1 multiplicaciones en la etapa de substitución hacia atrás.

El resultado neto no sería ventajoso, salvo que una misma factorización se utilizase N/2 veces en la substitución hacia atrás, lo que no es probable que ocurra con N>10.

Nuevamente se debe tener en cuenta que los resultados, en lo concerniente al número de operaciones, de la figura A4.1, se obtienen no sólo por las diferencias entre las ecuaciones (A4.4) y (A4.5), sino por el orden en que se ejecutan las operaciones de las etapas de eliminación hacia adelante y sustitución hacia atrás, en las que por brevedad no hemos entrado.

Sin embargo, cuando trabajamos con matrices vacías, las consideraciones hechas cambian por completo.

La figura A4.2 indica el número de operaciones a realizar en el caso de matrices vacías, para los tres algoritmos (su deducción es engorrosa y aquí no se incluye).

En dicha figura, B representa el número de elementos no diagonales en la mitad superior de la matriz, una vez triangularizada (un número que por supuesto no se conoce a priori). Asimismo B_i es el equivalente para la fila i. Obviamente.

CHOLESKY.

	Factorización	Elimin. adelante	Sustituc. atrás
Raices 2	N	-	-
Divisiones	B	N	N
Multiplic.	$\frac{B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(1 + \frac{B}{N})$	B	B
Sum. o Res.	$\frac{B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(1 + \frac{B}{N})$	B	B

DOOLITTLE SIMETRICO.

	Factorización	Elimin. adelante	Sustituc. atrás
Divisiones	N	-	-
Multiplic.	$\frac{3B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(3 + \frac{B}{N})$	B+N	B+N-1
Sum. o Res.	$\frac{B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(1 + \frac{B}{N})$	B	B

DOOLITTLE SIMETRICO MODIFICADO.

Divisiones	N	-	-
Multiplic.	$\frac{5B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(5 + \frac{B}{N})$	E+N	B
Sum. o Res.	$\frac{B}{2} + \frac{1}{2}\sum_{1}^{N} B_i^2 \simeq \frac{B}{2}(1 + \frac{B}{N})$	B	B

fig. A4.2

$$B = \sum_{i=1}^{N} B_i$$

La aproximación que aparece en dicha figura, consiste en sustituir B_i por el valor medio de elementos por fila B/N, con lo que

$$\sum_{i=1}^{N} B_i^2 = \frac{B^2}{N}$$

El error cometido no suele sobrepasar el 5%, obteniéndose siempre resultados optimistas. Como ejemplo considérese la red de 118 nudos. La matriz B' de esa red tiene por parámetros (ordenada por el esquema 2 de Tinney):

$$B = 253$$

$$N = 117$$

El número de multiplicaciones necesarias, para factorizar B' por el método de Doolittle sin modificar, es realmente 678, mientras que la ecuación aproximada:

$$\frac{B}{2}\left(3 + \frac{B}{N}\right) \qquad \text{(A4.6)}$$

da como resultado 653.

Volvamos a las comparaciones. Básicamente el método de Cholesky y el de Doolittle se diferencian en lo mismo (N raices cuadradas frente a N divisiones). Sin embargo, el algoritmo modificado que se ha propuesto, frente al método convencional de Doolittle, llega a ser ventajoso si una misma factorización se utiliza cierto número de veces.

En concreto, en la factorización se requieren B multiplicaciones extras, mientras que ahorramos N-1 en la substitución hacia atrás. Por tanto, empezamos a ganar tiempo, en cuanto una misma factorización se utilice un número de veces mayor a B/(N-1).

En el problema del flujo de cargas, casi siempre ocurre así, cuando se utiliza un jacobiano constante.

De nuevo utilizamos como ejemplo la red de 118 nudos. La misma B' se utiliza 5 veces en el proceso iterativo del método desacoplado rápido. Por tanto, frente a 253 multiplicaciones de más, ahorramos 5.(116)=580, lo que da un balance neto de 327 multiplicaciones ahorradas. A partir de 3 iteraciones (inclusive) ahorramos operaciones.

En resumen, deben darse dos circunstancias para que esta modificación sea ventajosa:

- Que la matriz sea suficientemente dispersa.
- Que se utilice la misma tabla de factores cierto número de veces.

Cuanto menos vacía sea la matriz, mayor número de veces, de utilización de la misma matriz, se requerirán para que la modificación sea rentable.

APENDICE 5. Datos de las redes.

Aquí sólo se mostrarán los datos de las redes de 29, 38 y 39 nudos. El resto se pueden encontrar en las referencias bibliográficas dadas en 5.1.

El orden de los datos es el siguiente:

- Número de nudos.
- Nudo de referencia.
- Número de líneas.
- Para cada línea: Nudos en los que incide, resistencia, reactancia (todo en p.u.), y relación de transformación si el enlace es un transformador con toma distinta de la nominal.
- Número de condensadores shunt, y para cada uno, nudo al que está conectado y susceptancia (p.u.).
- Número de nudos PV, y para cada uno, comenzando con el de referencia: Nudo, tensión (p.u.), potencia activa generada y consumida y límites de reactiva (mínimo, máximo).
- Seguidamente, sin solución de continuidad, y sin seguir un orden prefijado, el resto de nudos, dando para cada uno: Nudo, potencia activa generada y consumida y lo propio para la potencia reactiva.

Todas las potencias se expresan en MW (o MVAR).

A5.1 Red de 29 nudos.

```
 29  nudos
  5  nudo de referencia
 43  lineas
  1     1          2        .0850       .1307
  2     1          2        .0850       .1307
  3     1          3        .0546       .0863
  4     1          4        .0426       .0599
  5     1          4        .0426       .0599
  6     1          5        .0838       .1281
  7     1          5        .0838       .1281
  8     1          6        .0627       .1166
  9     1          6        .0627       .1166
 10     1          7        .0705       .1162
 11    28          9        .0367       .0537
 12    28          5        .0200       .0278
 13    10         11        .0069       .1405
 14    10         12        .0845       .1726
 15    13          5        .0821       .2167
 16    13         14        .1045       .2055
 17    15          9        .0448       .0558
 18    15         16        .0333       .0492
 19    17         16        .0253       .0441
 20    17          5        .0312       .0319
 21    18          2        .0859       .1387
 22    18         19        .0684       .1072
 23     2         19        .1446       .2364
 24    20         21        .0001       .0115
 25    22         21        .0675       .1012
 26    22         23        .1194       .1859
 27    11         23        .1128       .1848
 28    21         24        .0014       .0290
 29     9         25        .0358       .0514
 30     9          7        .0475       .0430
 31     3          4        .0138       .0216
 32    19         26        .0565       .0863
 33    27         16        .0012       .0342
 34    27         16        .0211       .0370
 35    16          5        .0516       .1663
 36    16         14        .1453       .2640
 37    24         29        .0099       .0743
 38     8         12        .0014       .0025
 39    12         29        .0365       .0676
 40    29          5        .0086       .1368
 41    29          6        .0863       .1761
```

```
42    5          25          .0121       .0212
43    5           7          .0418       .0427
 0  condendadores shunt
 5 nudos PV
 5   1.040
 1   1.040       121.2       20.8
 9   1.045        51.3
16   1.035        71.
24   1.050         8.6
 2                 21.5                    4.2
 3                 20.1                    5.1
 4                 29.                     8.6
 6                  8.2                    2.1
 7                 29.8                    7.6
28                 12.8                    2.2
10                 25.6                    5.3
11
12
13                 30.2                    2.9
14                 19.1                    8.3
15                 27.6                    9.1
17                 18.2                    4.2
18                 20.4                    1.6
19                 30.                     6.4
20
21
22                 12.3                    3.
23                  2.
25                 38.1                   10.
26                 19.4                    9.1
27                 20.2                    7.4
 8                  2.1                    .5
29
```

A5.2 Red de 38 nudos.

```
38   nudos
27   nudo de referencia (Santiponce)
55   lineas
 1    1          29          .1274       .1423
```

2	1	5	.0666	.0895
3	2	29	.1056	.1086
4	2	3	.0975	.1598
5	2	8	.0865	.1357
6	3	9	.1215	.2479
7	4	30	.1492	.1663
8	4	30	.1492	.1663
9	5	31	.0084	.0283
10	6	32	.1282	.2617
11	7	12	.1086	.2216
12	8	33	.1449	.2218
13	9	32	.1363	.2677
14	10	12	.1299	.149
15	11	12	.2011	.4105
16	11	30	.1632	.2646
17	12	30	.0623	.318
18	12	30	.1065	.1249
19	13	14	.0299	.0741
20	13	34	.006	.016
21	13	28	.0456	.1117
22	14	35	.0167	.056
23	14	35	.0214	.0524
24	14	28	.0755	.1858
25	15	5	.04	.108
26	15	27	.034	.1048
27	16	5	.032	.072
28	16	28	.0284	.056
29	17	5	.0108	.0148
30	17	28	.06	.1152
31	4	20	.068	.1495
32	4	21	.0776	.1592
33	4	35	.0656	.132
34	4	18	.0572	.1152
35	18	34	.006	.016
36	18	35	.0084	.0168
37	5	36	.0324	.0752
38	5	36	.0155	.0183
39	5	25	.044	.1192
40	5	27	.0904	.1476
41	5	27	.092	.1512
42	19	21	.041	.0985
43	19	28	.0409	.0885
44	20	21	.038	.0745
45	21	37	.0504	.084
46	21	24	.0364	.1104
47	21	24	.0916	.1028
48	21	38	.0332	.0788
49	22	37	.0136	.0368

```
50   22          28          .0324       .088
51   23          24          .3908       .3264
52   24          27          .0781       .2626
53   25          27          .0372       .1132
54   26          27          .1538       .2976
55   27          38          .0448       .1504
 0   condensadores shunt
 7   nudos pv (incluyendo slack)
27  1.047
 2  1.039       35.
 4  1.036       96.
 5  1.044       21.
12  1.030       6.
14  1.039       26.
28  1.037       18.
 1   5.         9.35        2.2         5.55
 3              5.35                    3.38
 6              3.57                    2.63
 7
 8              4.59                    3.2
 9              3.99                    2.73
10              1.7                     .66
11              13.26                   6.39
13
15              16.06                   5.83
16              31.02                   16.64
17              10.11                   5.17
18
19              24.39                   11.47
20              3.65                    2.07
21              8.84                    5.36
22              4.93                    .94
23              3.14                    1.5
24              13.17                   7.05
25              19.89                   7.24
26              11.3                    6.02
29   4.         4.59        3.          3.67
30              11.13                   5.92
31              8.5                     2.82
32              6.46                    2.82
33
34
35
36              14.96                   7.43
37              16.32                   5.92
38              17.17                   8.08
```

A5.3 Red de 39 nudos.

39	NUDOS					
31	NUDO DE REFERENCIA					
46	LINEAS					
1	1	2	.0005	.0411	.6987	
2	1	39	.001	.025	.75	
3	2	3	.0013	.0151	.2572	
4	2	25	.007	.0086	.146	
5	2	30		.0181		1.025
6	3	4	.0013	.0213	.2214	
7	3	18	.0011	.0133	.2138	
8	4	5	.0008	.0128	.1342	
9	4	14	.0008	.0129	.1382	
10	5	6	.0002	.0026	.0434	
11	5	8	.0008	.0112	.1476	
12	6	7	.0006	.0092	.113	
13	6	11	.0007	.0082	.1389	
14	7	8	.0004	.0046	.078	
15	8	9	.0023	.0363	.3804	
16	9	39	.001	.025	1.2	
17	10	11	.0004	.0043	.0729	
18	10	13	.0004	.0043	.0729	
19	13	14	.0009	.0101	.1723	
20	14	15	.0018	.0217	.366	
21	15	16	.0009	.0094	.171	
22	16	17	.0007	.0089	.1342	
23	16	19	.0016	.0195	.304	
24	16	21	.0008	.0135	.2548	
25	16	24	.0003	.0059	.068	
26	17	18	.0007	.0082	.1319	
27	17	27	.0013	.0173	.3216	
28	21	22	.0008	.014	.2565	
29	22	23	.0006	.0096	.1846	
30	23	24	.0022	.035	.361	
31	25	26	.0032	.0323	.513	
32	25	37	.0006	.0232		1.025
33	26	27	.0014	.0147	.2396	
34	26	28	.0043	.0474	.7802	
35	26	29	.0057	.0625	1.029	
36	28	29	.0014	.0151	.249	
37	29	38	.0008	.0156		1.025
38	12	11	.0016	.0435		1.006
39	12	13	.0016	.0435		1.006
40	6	31		.025		1.07
41	10	32		.02		1.07

42	19	33	.0007	.0142	1.07
43	20	34	.0009	.018	1.00
44	22	35		.0143	1.02
45	23	36	.0005	.0272	
46	19	20	.0007	.0138	1.06
0	CONDENSADORES SHUNT				
10	NUDOS PV				
31	.982		9.2		
30	1.0475	250.			
32	.9831	650.			
33	.9972	632.			
34	1.0123	508.			
35	1.0493	650.			
36	1.0635	560.			
37	1.0278	540.			
38	1.0265	830.			
39	1.03	1000.	1104.		
1		.0		.0	
2					
3		322.		2.4	
4		500.		184.	
5					
6					
7		233.8		84.	
8		522.		176.6	
9					
10					
11					
12		8.5		88.	
13					
14					
15		320.		153.	
16		329.4		32.3	
17					
18		158.		30.	
19					
20		680.		103.	
21		274.		115.	
22					
23		247.5		84.6	
24		308.6		-92.2	
25		224.		47.2	
26		139.		17.	
27		281.		75.5	
28		206.		27.6	
29		283.5		26.9	

APENDICE 6. Resultados obtenidos.

Por limitaciones de espacio sólo se mostrarán los resultados obtenidos en el PDP-11/34, siguiendo el algoritmo descrito en 5.5. Las tablas de los capítulos 5 y 6 recogen los resultados, en cuanto a tiempos, del resto de algoritmos, pero lógicamente el valor de las tensiones sigue siendo el mismo que aquí se indica.

A6.1 Red de 13 nudos.

```
SEMI-ITERACIONES: 11
PREORDENACION:     0.02 SEG
FORMACION YBUS B B":     0.18 SEG
TRIANGULIZACION B B":     0.00 SEG
TOTAL:     0.33 SEG
```

NUDO	MODULO	FASE
2	0.965	0.38
5	1.000	-0.82
4	0.927	-0.82
3	0.953	-0.90
6	1.037	1.88
7	1.061	-0.36
8	1.100	-3.13
9	0.943	-0.74
10	1.100	-6.75
11	1.163	-3.45
12	1.094	-3.20
13	1.043	0.73
1	1.000	0.00

A6.2 Red de 14 nudos.

```
SEMI-ITERACIONES:  8
PREORDENACION:     0.04 SEG
FORMACION YBUS B B":     0.26 SEG
TRIANGULIZACION B B":     0.00 SEG
TOTAL:     0.42 SEG
NUDO      MODULO              FASE
  8       1.090             -13.37
  3       1.010             -12.72
  2       1.045              -4.98
  5       1.020              -8.78
  7       1.062             -13.37
  4       1.019             -10.32
 10       1.051             -15.10
 11       1.057             -14.79
  9       1.056             -14.95
 12       1.055             -15.08
  6       1.070             -14.22
 13       1.050             -15.16
 14       1.036             -16.04
  1       1.060               0.00
```

A6.3 Red de 26 nudos.

```
SEMI-ITERACIONES:  9
PREORDENACION:     0.06 SEG
FORMACION YBUS B B":     0.32 SEG
TRIANGULIZACION B B":     0.00 SEG
TOTAL:     0.60 SEG
NUDO      MODULO              FASE
  2       1.065              13.96
  7       1.000              -1.47
 25       1.038              -2.58
  8       1.000               2.61
  9       1.000              20.41
 12       1.030               3.01
 14       1.016              14.79
 21       0.967              -4.38
 22       1.040               0.87
 11       1.053               5.26
 19       1.025               4.00
```

18	0.961	-6.47
23	0.940	-6.58
3	1.049	5.24
15	1.033	3.04
16	1.006	0.96
4	1.003	13.86
6	0.887	-5.90
10	1.026	4.29
13	0.986	5.64
24	0.921	6.00
5	1.020	12.91
17	0.943	2.33
20	0.901	-6.43
26	0.930	1.54
1	1.011	0.00

A6.4 Red de 29 nudos.

SEMI-ITERACIONES: 14
PREORDENACION: 0.10 SEG
FORMACION YBUS B B": 0.40 SEG
TRIANGULIZACION B B": 0.02 SEG
TOTAL: 0.92 SEG

NUDO	MODULO	FASE
8	1.041	-0.19
13	1.017	-1.87
14	1.010	-1.87
17	1.035	-0.22
20	1.049	-1.43
25	1.040	-0.19
26	0.950	-4.40
27	1.034	-0.39
16	1.035	-0.33
15	1.036	-0.52
9	1.045	-0.37
7	1.040	-0.26
28	1.039	-1.35
3	1.035	-0.71
4	1.036	-0.67
1	1.040	-0.47

6	1.039	-0.70
10	1.028	-2.17
11	1.031	-1.98
12	1.039	-1.35
18	0.986	-2.96
19	0.971	-3.51
2	1.010	-1.81
21	1.049	-1.43
22	1.043	-1.68
23	1.037	-1.83
24	1.050	-1.33
29	1.044	-1.04
5	1.040	0.00

A6.5 Red de 30 nudos.

SEMI-ITERACIONES: 7
PREORDENACION: 0.12 SEG
FORMACION YBUS B B": 0.38 SEG
TRIANGULIZACION B B": 0.02 SEG
TOTAL: 0.76 SEG

NUDO	MODULO	FASE
3	1.021	-7.99
11	1.082	-14.42
13	1.071	-15.28
26	1.000	-16.79
5	1.010	-14.38
7	1.003	-13.13
2	1.045	-5.53
4	1.012	-9.64
8	1.010	-12.11
9	1.051	-14.42
14	1.043	-16.17
16	1.045	-15.86
17	1.040	-16.17
18	1.028	-16.86
19	1.026	-17.03
20	1.030	-16.84
21	1.033	-16.45
22	1.033	-16.44

23	1.027	-16.64
25	1.018	-16.38
28	1.007	-11.99
29	1.004	-17.08
30	0.992	-17.96
27	1.023	-15.85
6	1.011	-11.37
10	1.045	-16.01
12	1.057	-15.28
15	1.038	-16.26
24	1.022	-16.81
1	1.060	0.00

A6.6 Red de 38 nudos.

SEMI-ITERACIONES: 16
PREORDENACION: 0.16 SEG
FORMACION YBUS B B": 0.44 SEG
TRIANGULIZACION B B": 0.02 SEG
TOTAL: 1.22 SEG

NUDO	MODULO	FASE
26	1.012	-1.32
6	0.915	-1.95
7	1.030	2.05
10	1.027	1.96
15	1.040	-0.68
23	1.000	-0.26
24	1.017	-0.02
25	1.038	-0.81
31	1.043	-0.69
32	0.927	-1.54
9	0.958	-0.29
3	0.999	1.20
33	1.031	2.09
8	1.031	2.09
2	1.039	2.27
29	1.037	1.29
1	1.037	0.05
36	1.041	-0.65
38	1.022	-0.16

5	1.044	-0.57
11	1.003	1.37
12	1.030	2.05
30	1.022	2.38
16	1.030	-0.47
17	1.042	-0.50
19	1.019	-0.11
20	1.024	1.34
22	1.023	-0.06
34	1.038	2.29
37	1.019	-0.12
21	1.020	0.43
4	1.036	3.34
13	1.038	2.11
14	1.039	2.47
18	1.037	2.46
28	1.037	0.32
35	1.038	2.53
27	1.047	0.00

A6.7 Red de 39 nudos.

SEMI-ITERACIONES: 11
PREORDENACION: 0.14 SEG
FORMACION YBUS B B": 0.44 SEG
TRIANGULIZACION B B": 0.02 SEG
TOTAL: 1.04 SEG

NUDO	MODULO	FASE
30	1.048	-4.15
32	0.983	1.78
33	0.997	2.44
34	1.012	1.12
20	1.003	-4.09
19	1.067	-2.74
35	1.049	4.18
36	1.064	6.84
37	1.028	1.33
38	1.026	6.19
1	1.071	-8.96
7	1.027	-10.41

9	1.053	-10.69
10	1.042	-6.02
12	1.028	-6.86
11	1.039	-6.81
13	1.042	-6.71
15	1.052	-8.63
18	1.071	-9.03
21	1.060	-4.99
22	1.067	-0.70
23	1.063	-0.89
24	1.071	-7.17
16	1.066	-7.28
25	1.085	-5.28
27	1.090	-8.39
28	1.098	-3.27
29	1.086	-0.64
26	1.107	-6.56
39	1.030	-10.43
2	1.075	-6.51
3	1.066	-9.26
5	1.034	-8.99
6	1.035	-8.33
4	1.038	-10.07
8	1.026	-10.89
14	1.045	-8.29
17	1.075	-8.24
31	0.982	0.00

A6.8 Red de 57 nudos.

SEMI-ITERACIONES: 9
PREORDENACION: 0.28 SEG
FORMACION YBUS B B": 0.64 SEG
TRIANGULIZACION B B": 0.04 SEG
TOTAL: 1.56 SEG

NUDO	MODULO	FASE
2	1.010	-1.19
16	1.013	-8.86
17	1.017	-5.40
33	0.948	-18.55

3	0.985	-5.99
5	0.976	-8.55
18	1.001	-11.73
19	0.970	-13.23
20	0.964	-13.45
21	1.009	-12.93
23	1.009	-12.94
25	0.983	-18.17
26	0.959	-12.98
27	0.982	-11.51
28	0.997	-10.48
30	0.963	-18.72
31	0.936	-19.38
32	0.950	-18.51
34	0.959	-14.15
35	0.966	-13.91
39	0.983	-13.49
40	0.973	-13.66
42	0.967	-15.53
43	1.010	-11.35
41	0.996	-14.08
44	1.017	-11.86
45	1.036	-9.27
46	1.060	-11.12
47	1.033	-12.51
50	1.024	-13.41
51	1.052	-12.53
52	0.980	-11.50
53	0.971	-12.25
54	0.996	-11.71
55	1.031	-10.80
57	0.965	-16.58
4	0.981	-7.34
7	0.984	-7.60
8	1.005	-4.48
10	0.986	-11.45
11	0.974	-10.19
12	1.015	-10.47
14	0.970	-9.35
24	0.999	-13.29
37	0.985	-13.45
6	0.980	-8.67
29	1.010	-9.77
22	1.010	-12.88
36	0.976	-13.64
48	1.028	-12.61
49	1.036	-12.94
9	0.980	-9.58

13	0.979	-9.80
15	0.988	-7.19
38	1.013	-12.74
56	0.969	-16.07
1	1.040	0.00

A6.9 Red de 118 nudos.

SEMI-ITERACIONES: 9
PREORDENACION: 0.82 SEG
FORMACION YBUS B B": 1.28 SEG
TRIANGULIZACION B B": 0.08 SEG
TOTAL: 3.34 SEG

NUDO	MODULO	FASE
10	1.050	6.71
9	1.043	-0.87
73	0.991	-7.47
87	1.015	4.53
86	0.987	4.28
111	0.980	-5.83
112	0.975	-10.57
116	1.005	-2.09
117	0.974	-18.22
1	0.955	-18.19
2	0.971	-17.65
3	0.968	-17.31
4	0.998	-13.59
6	0.990	-15.87
7	0.989	-16.32
8	1.015	-8.13
13	0.968	-17.54
14	0.984	-17.39
16	0.984	-16.97
18	0.973	-17.38
20	0.957	-16.97
21	0.958	-15.39
22	0.969	-12.83
26	1.015	0.82
28	0.962	-15.14
29	0.963	-16.16

33	0.971	-18.34
35	0.980	-18.18
36	0.980	-18.18
39	0.970	-20.69
41	0.967	-22.25
40	0.970	-21.79
42	0.985	-20.68
43	0.977	-17.83
44	0.985	-15.44
47	1.017	-8.71
48	1.021	-9.41
46	1.005	-10.87
45	0.986	-13.64
50	1.001	-10.43
52	0.957	-13.98
53	0.946	-14.95
57	0.971	-12.94
58	0.959	-13.79
63	0.969	-6.49
67	1.020	-4.41
68	1.003	-1.66
71	0.987	-7.26
72	0.980	-8.25
24	0.992	-8.13
74	0.958	-7.66
70	0.984	-6.87
76	0.943	-7.09
78	1.003	-1.84
79	1.009	-1.46
81	0.997	-0.62
84	0.980	3.96
83	0.985	1.15
88	0.987	9.01
85	0.985	5.65
90	0.985	6.85
91	0.980	6.90
89	1.005	13.22
93	0.986	4.42
95	0.984	0.89
97	1.013	0.34
98	1.023	0.34
99	1.010	3.62
101	0.992	3.71
102	0.989	6.13
107	0.952	-8.03
106	0.961	-5.23
108	0.966	-6.17
109	0.967	-6.63

110	0.973	-7.47
113	0.993	-15.14
114	0.961	-14.17
115	0.960	-14.19
118	0.950	-7.16
75	0.968	-6.36
5	1.002	-13.15
11	0.985	-16.16
12	0.990	-16.68
25	1.050	-0.95
31	0.967	-16.05
32	0.963	-13.95
27	0.968	-13.39
34	0.984	-17.72
38	0.961	-12.12
51	0.967	-13.03
55	0.952	-14.31
54	0.955	-14.03
56	0.954	-14.13
60	0.993	-6.09
61	0.995	-5.20
59	0.985	-9.89
62	0.998	-5.82
64	0.984	-4.72
66	1.050	-1.78
49	1.025	-8.40
77	1.006	-1.59
92	0.990	7.50
94	0.991	2.21
96	0.995	0.34
82	0.990	-0.21
80	1.040	1.03
103	1.010	-1.30
104	0.971	-3.87
105	0.965	-4.97
100	1.017	2.44
15	0.970	-17.69
17	0.995	-15.15
19	0.962	-17.85
23	1.000	-7.91
30	0.985	-10.14
37	0.991	-17.27
65	1.005	-1.57
69	1.035	0.00

BIOGRAFIA DEL AUTOR.

Antonio Gómez Expósito nació en Andújar (Jaén) el 26 de Agosto de 1957.

En 1983 obtuvo el título de Ingeniero Industrial (Especialidad Eléctrica) en la Escuela Técnica Superior de Ingenieros Industriales de Sevilla, con la calificación de Notable, siendo el número uno de la Undécima Promoción.

Desde Octubre de 1982 viene impartiendo clases en las Cátedras de Electrotécnia y Electrónica de la E.T.S.I.I. de Sevilla. Sus áreas de investigación se centran en la aplicación de los computadores, y en especial de los microprocesadores, a diversas facetas del vasto campo que suponen los sistemas de potencia (simulación, protecciones, etc.).

Sevilla, Junio de 1985.