Lenguaje C **Curso 2010–2011, TRABAJO DE CURSO**

LIBRE CONFIGURACIÓN (Fecha de entrega: 20/01/2011)

Confeccionar una función en Lenguaje C conforme al siguiente prototipo:

```
int renum_(int *ne, int *nn, int *np, int *ln,
           int *lr, int *ll, int *nR1, int *nR2)
{
       ...
}
```

donde las variables tienen el siguiente significado:

`int *ne`	(DATO)	=	número de elementos de la malla,
`int nn[ ]`	(DATO)	=	número de nodos de cada elemento,
`int *np`	(DATO)	=	número de nodos de la malla,
`int ln[ ]`	(DATO)	=	conectividad de la malla,
`int lr[ ]`	(RESULTADO)	=	renumeración,
`int ll[ ]`	(RESULTADO)	=	renumeración inversa,
`int *nR1`	(RESULTADO)	=	tamaño del perfil antes de la renumeración,
`int *nR2`	(RESULTADO)	=	tamaño del perfil después de la renumeración.

DATOS

Las variables `*ne` y `*np` se pasan a la función por referencia y no por valor, para que la función pueda ser invocada directamente desde un programa Fortran si es necesario.

En esencia, la malla es el conjunto integrado por los `*ne` elementos.
El elemento `ie` (siendo $1 \leq$ `ie` $\leq$ `*ne`) está formado por `nn[ie - 1]` nodos.
A cada uno de los nodos `in` del elemento `ie` (siendo $1 \leq$ `in` $\leq$ `nn[ie - 1]`) le corresponde un número global `ip` (siendo $1 \leq$ `ip` $\leq$ `*np`). La relación entre la numeración local (nodo `in` del elemento `ie`) y la numeración global (nodo global `ip`) la establece el vector de conectividad `ln[ ]` en la forma

$$\texttt{ip = ln[(in - 1) + kie];} \qquad \text{donde} \quad \texttt{kie} = \sum_{\texttt{je<ie}} \texttt{nn[je - 1]}.$$

Para que la numeración se considere correcta, cada nodo global `ip` (siendo $1 \leq$ `ip` $\leq$ `*np`) debe aparecer al menos una vez en el vector de conectividad. Si un nodo global `ip` aparece más de una vez como parte del mismo elemento, éste se dice degenerado. Se admitirá la posibilidad de que haya elementos degenerados.

El vector de renumeración `lr[ ]` representa una posible modificación (permutación) de la numeración global, de forma que `ipr = lr[ip - 1]` indica cuál es el nuevo número `ipr` que se le ha asignado al nodo global `ip`. Este vector se inicializa de la siguiente forma

```
for (ip = 1; ip <= *np; ip++) {
       lr[ip - 1] = ip;
}
```

lo que indica que inicialmente los nodos no están renumerados.

El vector de renumeración inversa `ll[ ]` indica cuál es la permutación inversa que restaura la numeración inicial, de forma que `ip = ll[ipr - 1]` indica cuál es el nodo global `ip` al que se le ha asignado el nuevo número `ipr`. Este vector se puede obtener fácilmente a partir del vector `lr[ ]` mediante el siguiente procedimiento

```
for (ip = 1; ip <= *np; ip++) {
       ipr = lr[ip - 1];
       ll[ipr - 1] = ip;
}
```

Esta forma de codificar la información dará lugar (en una fase posterior de los cálculos) a una matriz cuadrada de orden `*np` que denominaremos matriz de rigidez

$$\underset{\sim}{K} = \left[k_{\texttt{ipr},\texttt{jpr}}\right] \qquad ;\ \texttt{ipr} = 1,\ldots,\texttt{*np},$$
$$;\ \texttt{jpr} = 1,\ldots,\texttt{*np}.$$

El procedimiento mediante el que se calculan los componentes de la matriz de rigidez se denomina **ENSAMBLAJE**. En esencia, el ensamblaje consiste en sumar la contribución de cada elemento `ie` al valor de cada coeficiente $k_{\texttt{ipr},\texttt{jpr}}$. Además, se sabe que el elemento `ie` sólo aportará algo al valor del coeficiente $k_{\texttt{ipr},\texttt{jpr}}$ cuando los nodos globales renumerados `ipr` y `jpr` formen parte del elemento `ie`.

Esperamos que la mayor parte de los coeficientes de la matriz $\underset{\sim}{K}$ serán nulos, ya que el coeficiente $k_{\texttt{ipr},\texttt{jpr}}$ sólo será distinto de cero cuando los nodos globales renumerados `ipr` y `jpr` pertenezcan simultáneamente a algún elemento. En caso contrario, este coeficiente no recibirá ninguna contribución de ningún elemento, por lo que será nulo. Los coeficientes de la diagonal principal serán necesariamente distintos de cero.

La matriz $\underset{\sim}{K}$ también será simétrica, por lo que sólo se almacenará su parte triangular superior (incluida la diagonal).

La parte triangular superior de la matriz $\underset{\sim}{K}$ se almacenará mediante la técnica conocida como *sky-line*, *column-profile* o almacenamiento en perfil. Para ello se determinará para cada columna `jpr` cuál es el valor

$$\mathcal{H}(\texttt{jpr}) = \min_{1\leq \texttt{ipr} \leq \texttt{jpr}} \left\{ \texttt{ipr} \;\middle|\; k_{\texttt{ipr},\texttt{jpr}} \neq 0 \right\},$$

o, lo que es lo mismo, cuál es el mínimo valor de `ipr` ($1 \leq$ `ipr` $\leq$ `jpr`) para el que `jpr` e `ipr` pertenecen al mismo elemento. Una vez determinados estos valores se almacenará (por columnas) la parte triangular superior de la matriz situada entre el primer coeficiente no nulo de cada columna y el correspondiente coeficiente de la diagonal principal. Por tanto, para cada columna `jpr` sólo se almacenarán los coeficientes situados entre la fila $\mathcal{H}$(`jpr`) y la fila `jpr`.

La parte almacenada se denomina perfil. Dentro del perfil puede haber componentes nulos, que será necesario almacenar porque podrían dejar de ser nulos en una fase posterior del cálculo. El objetivo de la renumeración es reducir (al mínimo, si es posible) el tamaño del perfil, esto es, el número de componentes de la matriz de rigidez que es necesario almacenar.

Por ejemplo, sea la malla (ver Figura 1)

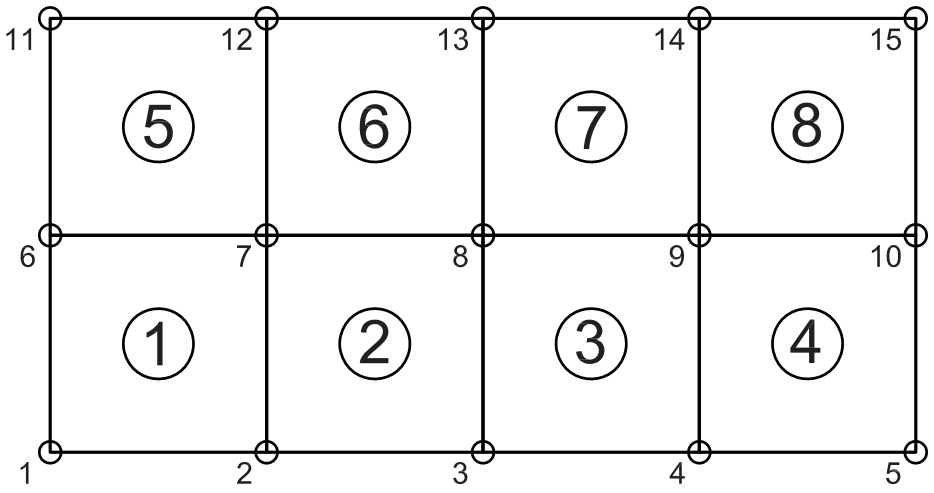

Fig 1. Ejemplo: malla formada por 8 elementos de 4 nodos (SIN RENUMERACIÓN)

para la que

$$\texttt{*ne} = 8, \qquad \texttt{nn[ ]} = \{4,4,4,4,4,4,4,4\}, \qquad \texttt{*np=15},$$
$$\texttt{ln[ ]} = \{1,2,7,6,2,3,8,7,3,4,9,8,4,5,10,9,6,7,12,11,7,8,13,12,8,9,14,13,9,10,15,14\},$$
$$\texttt{lr[ ]} = \{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15\} \quad \text{(sin renumeración)}.$$

De acuerdo con lo expuesto anteriormente, la matriz de rigidez correspondiente a este caso tendrá el aspecto siguiente

$$\underset{\sim}{K} = \begin{bmatrix}
\star & \star & & & \star & \star & & & & & & & \\
 & \star & \star & & \star & \star & \star & & & & & & \\
 & & \star & \star & 0 & \star & \star & \star & & & & & \\
 & & & \star & 0 & 0 & \star & \star & \star & & & & \\
 & & & \star & 0 & 0 & 0 & \star & \star & & & & \\
 & & & & \star & \star & 0 & 0 & 0 & \star & \star & & \\
 & & & & & \star & \star & 0 & 0 & \star & \star & \star & \\
 & & & & & & \star & \star & 0 & 0 & \star & \star & \star \\
 & & & & & & & \star & \star & 0 & 0 & \star & \star \\
 & & & & & & & & \star & \star & 0 & 0 & 0 \\
 & & & & & & & & & \star & \star & 0 & 0 \\
 & & & & & & & & & & \star & \star & 0 \\
 & & & & & & & & & & & \star & \star \\
 & & & & & & & & & & & & \star
\end{bmatrix}, \qquad \text{SIMÉTRICA}$$

por lo que será necesario almacenar los $\texttt{*nR1}$=77 valores que se indican a continuación

$$\{k_{1,1}, k_{1,2}, k_{2,2}, k_{2,3}, k_{3,3}, k_{3,4}, k_{4,4}, k_{4,5}, k_{5,5},$$
$$k_{1,6}, \ldots, k_{6,6}, k_{1,7}, \ldots, k_{7,7}, k_{2,8}, \ldots, k_{8,8}, k_{3,9}, \ldots, k_{9,9}, k_{4,10}, \ldots, k_{10,10},$$
$$k_{6,11}, \ldots, k_{11,11}, k_{6,12}, \ldots, k_{12,12}, k_{7,13}, \ldots, k_{13,13}, k_{8,14}, \ldots, k_{14,14}, k_{9,15}, \ldots, k_{15,15}\},$$

de los cuales 53 son no nulos y 24 son nulos.

Sin embargo, la renumeración (ver Figura 2)

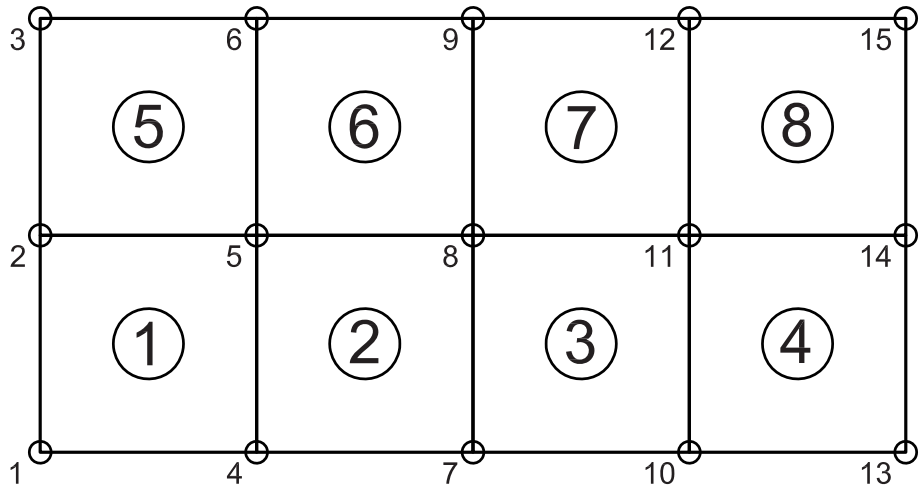

Fig 2. Ejemplo: malla formada por 8 elementos de 4 nodos (CON RENUMERACIÓN)

para la que

$$\texttt{*ne} = 8, \qquad \texttt{nn[ ]} = \{4,4,4,4,4,4,4,4\}, \qquad \texttt{*np=15},$$
$$\texttt{ln[ ]} = \{1,2,7,6,2,3,8,7,3,4,9,8,4,5,10,9,6,7,12,11,7,8,13,12,8,9,14,13,9,10,15,14\},$$
$$\texttt{lr[ ]} = \{1,4,7,10,13,2,5,8,11,14,3,6,9,12,15\} \quad \text{(con renumeración)},$$

dará lugar a la matriz

$$\underset{\sim}{K} = \begin{bmatrix}
\star & \star & & \star & \star & & & & & & & & & & \\
 & \star & \star & \star & \star & \star & & & & & & & & & \\
 & & \star & 0 & \star & \star & & & & & & & & & \\
 & & & \star & \star & 0 & \star & \star & & & & & & & \\
 & & & & \star & \star & \star & \star & \star & & & & & & \\
 & & & & & \star & 0 & \star & \star & & & & & & \\
 & & & & & & \star & \star & 0 & \star & \star & & & & \\
 & & & & & & & \star & \star & \star & \star & \star & & & \\
 & & & & & & & & \star & 0 & \star & \star & & & \\
 & & & & & & & & & \star & \star & 0 & \star & \star & \\
 & & & & & & & & & & \star & \star & \star & \star & \star \\
 & & & & & & & & & & & \star & 0 & \star & \star \\
 & & & & & & & & & & & & \star & \star & 0 \\
 & & & & & & & & & & & & & \star & \star \\
 & & & & & & & & & & & & & & \star
\end{bmatrix},$$

SIMÉTRICA

por lo que sólo será necesario almacenar los $\texttt{*nR2}$=61 valores siguientes

$$\{k_{1,1}, k_{1,2}, k_{2,2}, k_{2,3}, k_{3,3}, k_{1,4}, \ldots, k_{4,4}, k_{1,5}, \ldots, k_{5,5},$$
$$k_{2,6}, \ldots, k_{6,6}, k_{4,7}, \ldots, k_{7,7}, k_{4,8}, \ldots, k_{8,8}, k_{5,9}, \ldots, k_{9,9}, k_{7,10}, \ldots, k_{10,10},$$
$$k_{7,11}, \ldots, k_{11,11}, k_{8,12}, \ldots, k_{12,12}, k_{10,13}, \ldots, k_{13,13}, k_{10,14}, \ldots, k_{14,14}, k_{11,15}, \ldots, k_{15,15}\},$$

de los cuales 53 son no nulos (los mismos, obviamente que en el caso anterior) y 8 son nulos.

OBJETIVO

Por tanto, el objetivo de la función `int renum_()` será:

> Hallar los valores `jpr = lr[jp - 1]`, donde $1 \leq$ `jp` $\leq$ `*np`, para los que se minimiza el tamaño del perfil resultante
>
> $$\sum_{\texttt{jpr}=1,\texttt{*np}} \big(\texttt{jpr} - \mathcal{H}(\texttt{jpr}) + 1\big).$$

La renumeración tiene tanto más interés cuanto mayor es el tamaño del problema. Se entiende que estamos interesados en aplicar este tipo de técnicas a problemas en los que el tamaño del problema es relativamente grande (`*np` puede ser del orden de 10^4 hasta 10^8 o incluso superior).

RESULTADOS

La función `renum_()` deberá realizar dos tareas:

1) Verificar que la numeración es correcta, es decir que cada uno de los nodos globales aparece al menos una vez en el vector de conectividad. Se admitirá la posible existencia de elementos degenerados, por lo que un mismo nodo global podrá aparecer más de una vez como parte del mismo elemento. Si la numeración no es correcta, la función devolverá el valor 0 y no realizará la renumeración.

2) Renumerar la malla (sólo en caso de que la numeración sea correcta). El resultado principal será el vector `lr[ ]` que proporcionará una renumeración razonablemente buena y tanto mejor como sea posible (ya que en general no será posible obtener la renumeración óptima). Además, la función proporcionará el vector `ll[ ]` y los valores `*nR1` y `*nR2`. En este caso, la función devolverá el valor del tamaño del perfil que resulta de la renumeración (esto es, el valor `*nR2` $\neq 0$).

REALIZACIÓN DEL TRABAJO

Uno de los primeros algoritmos que demostró ser razonablemente eficaz para resolver este problema fue el conocido como **RCM** (siglas de *Reverse Cuthill–McKee*). La referencia inicial de este algoritmo[1] data de 1969, aunque con posterioridad se han propuesto diversas mejoras.

Pocos años más tarde se desarrolló el algoritmo conocido como **GPS** (siglas de *Gibbs–Poole–Stockmeyer*, que son los apellidos de sus autores). La referencia inicial de este algoritmo[2] data de 1976. En general, se acepta que este algoritmo puede proporcionar resultados comparables a los del **RCM**, con la ventaja de que el tiempo de computación puede ser inferior en la mayor parte de los casos. También para este algoritmo se han propuesto diversas mejoras desde entonces.

Una de estas posibles mejoras la constituye el algoritmo propuesto por Wang, Guo y Shi en 2009. El estudiante deberá implementar este algoritmo en los términos en que fue descrito originalmente por sus autores[3].

Al implementar el algoritmo se tendrá en cuenta que la función está destinada a tratar problemas de gran tamaño, por lo que se valorará la complejidad computacional de cada una de sus partes.

La función se escribirá de forma que sea compatible con el estándar ANSI C. La totalidad del código fuente desarrollado por el estudiante (es decir la función `renum_()` y todas las funciones auxiliares que sean necesarias para su funcionamiento) estarán incluidas en un archivo que se denominará `subrenum.c`. Los prototipos de estas funciones estarán incluidas en un archivo que se denominará `renum.h`.

El estudiante deberá compilar y linkar su archivo `subrenum.c` con el contenido del archivo `prf.c` que se adjunta, utilizando el compilador `gcc` de GNU. Para ello se utilizará la instrucción

```
gcc prf.c subrenum.c -O2 -o prf.exe
```

El estudiante deberá comprobar que el compilado y linkado se realiza correctamente y que el programa ejecutable resultante `prf.exe` procesa adecuadamente los datos contenidos en los archivos `malla_1.txt` (malla de la figura 1), `malla_2.txt` y `malla_3.txt`.

La función `main()` del archivo `prf.c` ya incorpora el código necesario para medir el tiempo de CPU invertido en la renumeración e imprimirlo en pantalla. El estudiante deberá tomar nota de este dato para su uso posterior.

PRESENTACIÓN DEL TRABAJO

Los archivos `subrenum.c` y `renum.h` se enviarán por e-mail antes de las 15:00 horas del 20 de enero de 2011 a la dirección de correo `fermin.navarrina@udc.es` indicando el nombre completo del autor. No se enviará por correo electrónico ningún programa ejecutable. En el mismo plazo se entregará al profesor en formato papel una memoria. La memoria contendrá una explicación del problema que hay que resolver, una descripción del algoritmo propuesto (incluyendo sus características más importantes, su complejidad computacional y los detalles más relevantes de su implementación) y el listado del código fuente. Finalmente, se hará constar el tiempo de CPU invertido en renumerar los tres casos indicados en el apartado anterior, y se relacionará este dato con la complejidad computacional del algoritmo implementado.

El trabajo es personal. Por tanto, cada estudiante deberá realizar su trabajo personalmente. Obviamente, está permitido realizar consultas a otros profesores y/o a otros compañeros, pero debe quedar claro que realizar una consulta a otra(s) persona(s) no es lo mismo que presentar un trabajo realizado por otra(s) persona(s) en parte o en su totalidad. El trabajo de programación deberá ser realizado por el estudiante. Por tanto, el código no podrá consistir en una copia total o parcial de otras fuentes, ni en una traducción, transliteración u ofuscamiento de un programa preexistente realizado en C o en otro lenguaje.

La copia, traducción, transliteración u ofuscamiento de programas procedentes de otras fuentes o realizados por otras personas, o la realización del trabajo en grupo, supondrá el suspenso en dos convocatorias consecutivas de esta asignatura.

El día 27 de enero de 2011 los estudiantes deberán presentar en un tiempo máximo de 10 minutos el trabajo realizado. Para ello podrán utilizar los medios audiovisuales que consideren oportunos. La hora y el aula en la que realizará la exposición cada estudiante se publicarán el día 25 de enero en el tablón de anuncios y en la página web de la asignatura.

REFERENCIAS

1. E. Cuthill and J. McKee, "Reducing the bandwidth of sparse symmetric matrices". **Proceedings of the 1969 24th national conference**, pp. 157–172, New York, NY, USA, ACM Press (1969).
2. N.E. Gibbs, W.G. Poole and P.K. Stockmeyer, "An algorithm for reducing the bandwidth and profile of a sparse matrix", **SIAM Journal on Numerical Analysis**, Vol. 13, No. 2, pp. 236–250 (1976).
3. Q. Wang, Y.C. Guo and X.W. Shi, "An improved algorithm for matrix bandwidth and profile reduction in Finite Element Analysis", **Progress In Electromagnetics Research Letters**, Vol. 9, pp. 29–38 (2009).
http://www.jpier.org/PIERL/pierl09/04.09042305.pdf

NOTA: Alternativamente, los estudiantes que estén cursando o que hayan cursado otra asignatura en la que se utilice el Lenguaje C (o alguna de sus variantes) pueden realizar un trabajo relacionado con el temario de esa asignatura. Para ello, deben facilitarnos previamente el nombre del profesor correspondiente. Si éste está de acuerdo propondremos conjuntamente un trabajo. Lógicamente, este trabajo no puede ser uno de los que haya que realizar (en su caso) para aprobar la otra asignatura, aunque sí podría ser una extensión de éste.

Para resolver cualquier duda o problema relacionado con este enunciado el estudiante podrá ponerse en contacto con los profesores de la asignatura.

Fdo. Los Profesores de la Asignatura