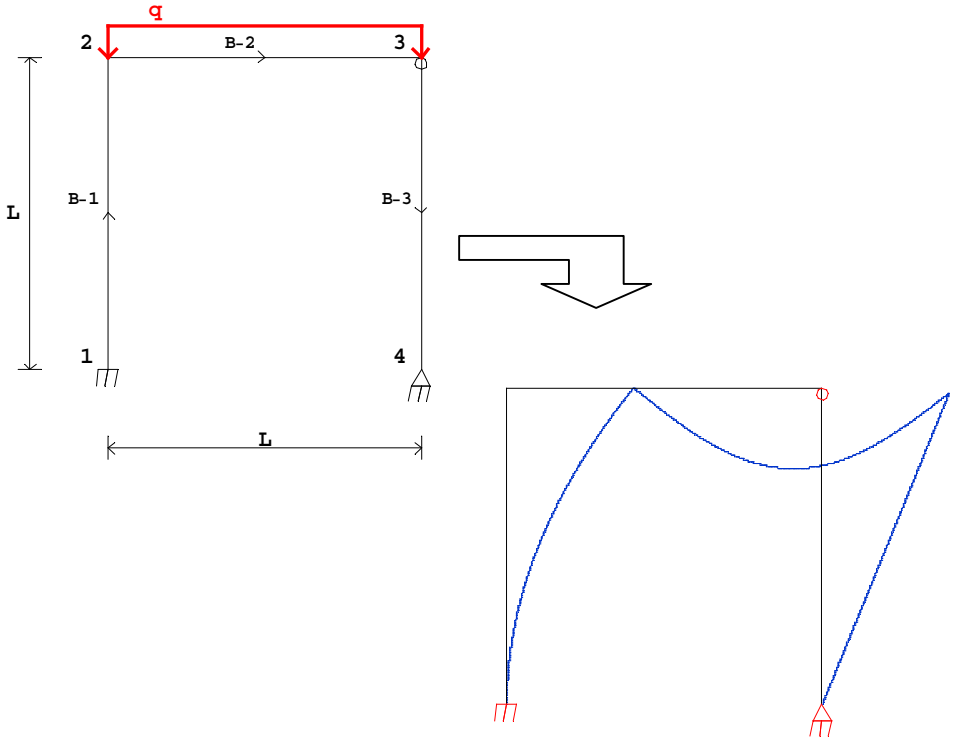
GRUPO DE ELASTICIDAD Y RESISTENCIA DE MATERIALES

PROYECTO FIN DE CARRERA

UN METODO ANALÍTICO DOCENTE PARA LA RESOLUCIÓN DE ESTRUCTURAS. (M.A.D.R.E)



TOMO I

Francisco de Asís Salguero Alvarez

ÍNDICE

Agradecimientos especiales a
Rafael Picón Carrizosa por los
consejos ofrecidos y la brillantez de
las ideas aportadas.

Este proyecto está dedicado a mi padre que siempre luchó por dar a sus hijos las posibilidades que a él no le ofrecieron y a mi madre por resistir estóicamente los embates de esta vida.

1-.INTRODUCCIÓN

1.1. Objetivos.

El objeto principal del proyecto es dar a los alumnos interesados una herramienta informática que les permita comprobar sus conocimientos en el cálculo de las estructuras.

Los alumnos en la Escuela Superior de Ingenieros de Sevilla, cuando llegan a la asignatura de Resistencia de Materiales (RM), tienen la obligación de enfrentarse al cálculo de estructuras. Cuando lo hacen, muchos de ellos suelen encontrar problemas a la hora de resolver sus ejercicios y sobre todo de comprobar los resultados obtenidos, las aplicaciones desarrolladas permitirán al alumno realizar el cálculo de estructuras sencillas mediante el Método de las Fuerzas, un método idóneo para la resolución manual de estructuras y que es el que se utiliza en la mencionada asignatura.

Debido al carácter docente de la aplicación, se ha intentando seguir al pie de la letra la metodología usada por los alumnos en el cálculo de estructuras a la vez que se ha tenido en cuenta los problemas que ello genera a la hora de realizar la programación. Este motivo ha sido el de mayor peso a la hora de realizar un cálculo matricial, y se ha intentado en todo lo posible usar una formulación matricial del problema que nos simplificara su resolución computacional. Por otro lado, ha sido necesario usar un programa que nos permita usar matemática simbólica, tales como Matlab, Mathematica, Maple, etc. Se ha elegido Mathematica por proximidad, ya que era una herramienta que estaba en el departamento y también por ser conocido como uno de los mejores programas de cálculo matemático simbólico.

Esta aplicación está orientada a un uso docente, es decir, por parte de los profesores del Grupo de Elasticidad y Resistencia de Materiales y por el alumnado de RM, así que se ha restringido la tipología de las estructuras calculadas a pórticos planos y a emparrillados, que son las estructuras normalmente calculadas. Obviamente no tiene ningún sentido usar un método de cálculo manual de estructuras en estructuras complejas con un elevado número de incógnitas.

Por último solo queda comentar en esta introducción que el cálculo está dividido por un lado en una parte simbólica, donde se pretende que el usuario practique los conceptos asociados al cálculo de estructuras por el método de las fuerzas, es decir uso de las ecuaciones de equilibrio y compatibilidad – comportamiento, y una segunda parte de diseño de la estructura (elección de perfiles óptimos) que es completamente numérico y donde se trabaja con los algoritmos de búsqueda óptima de perfiles propios de este método.

1.2. Ecuaciones generales del cálculo de estructuras.

El proyecto consiste en la creación de una herramienta para el cálculo de estructuras bidimensionales, pórticos planos y emparrillados, mediante el Método de las Fuerzas, usando en la parte de diseño los algoritmos propios de esta metodología. Para ello resulta interesante realizar una breve introducción a las dos metodologías generales que existen para el cálculo de estructuras.

De la teoría de la Resistencia de Materiales sabemos que el problema del cálculo de estructuras es un problema cerrado donde tenemos una ecuaciones de equilibrio de fuerzas que relacionan los esfuerzos internos en las barras, que caracterizan la transmisión interna de fuerzas a través de las barras, con las fuerzas externas aplicadas. Unas ecuaciones de compatibilidad que nos relacionan las deformaciones, que representan medidas de la deformación del eje de la barra con los desplazamientos de dicho eje, y por último, unas ecuaciones de comportamiento que nos relacionan los esfuerzos y las deformaciones correspondientes. En la Fig. 1 podemos visualizar de forma sencilla esta relación:

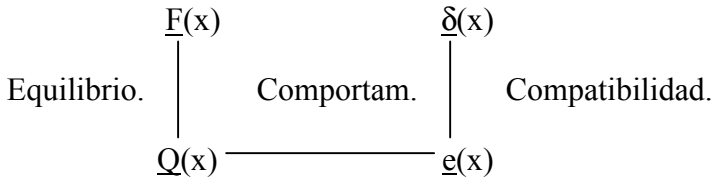


Fig. 1

Donde Q(x) son los esfuerzo internos, F(x) las cargas externas, e(x) son las deformaciones internas y δ(x) son los desplazamientos. Este planteamiento es un planteamiento muy general que en nuestro caso nosotros simplificamos usando las hipótesis de sólido homogéneo, isótropo y de comportamiento lineal y admitiendo que deformaciones y desplazamientos son muy pequeños, por lo que la geometría deformada se toma coincidente con la inicial.

Dependiendo de cómo combinemos los grupos de ecuaciones anteriores, existen principalmente dos métodos de cálculo de estructuras, el Método de los desplazamientos y el Método de las fuerzas.

El método de los desplazamientos se denomina así por ser los desplazamientos de los nudos las incógnitas finales del problema. Este método es el usado por la practica totalidad de los programas de ordenador para el cálculo de estructuras, debido a que todas las operaciones que lleva asociadas son muy fáciles de sistematizar y pueden llevarse a cabo de una forma computacionalmente muy eficiente.

El otro método es el Método de las Fuerzas, que es el que nosotros pretendemos implantar, y se denomina así por ser las fuerzas (realmente las incógnitas hiperestáticas) las incógnitas finales del problema. Las etapas a realizar en el análisis de una estructura por el MF se describen a continuación.

1. Determinar el grado de hiperestaticidad de una estructura y escoger un conjunto adecuado de incógnitas hiperestáticas.

2. Expresar los esfuerzos internos en todos los puntos de la estructura en función de las incógnitas hiperestáticas y de las cargas externas.

3. Expresar el valor de los desplazamientos asociados a las incógnitas hiperestáticas y las condiciones de contorno cinemáticas. Esto se puede hacer usando las ecuaciones de compatibilidad-comportamiento (Teoremas de Mohr, Fórmulas de Bresse o Principio de las Fuerzas Virtuales).

4. Calcular los esfuerzos internos en todos los puntos de la estructura. Una vez conocidas las incógnitas hiperestáticas, basta sustituir sus valores en las expresiones de los esfuerzos reales para tener definidos los esfuerzos internos en todos los puntos de la estructura. A partir de ese momento puede procederse al cálculo de tensiones en las secciones más desfavorables.

5. Calcular los desplazamientos deseados.

En estructuras simples, el grado de hiperestaticidad es bastante menor que el número de desplazamientos de los nudos, esta particularidad hace que el MF sea más adecuado para el uso docente y para el cálculo manual de pequeñas estructuras.

2.ANÁLISIS SIMBÓLICO DE LA ESTRUCTURA.

2.1. Pórticos planos.

2.1.1. Ecuaciones básicas a nivel diferencial.

En un caso general de cargas, es decir si tuviéramos una estructura tridimensional cargada, en cada barra tendríamos definidos 6 esfuerzos (N, V_y, V_z, M_y, M_z, T), tendríamos axil, cortante en la dirección y, cortante en la dirección z, flector en la dirección y, momento en la dirección z y momento en la dirección x (torsor), sin embargo, un pórtico plano es una estructura que se caracteriza por no tener cargas normales a su plano, es decir todas las cargas están dentro del plano de la estructura como puede verse en la Fig. 2.

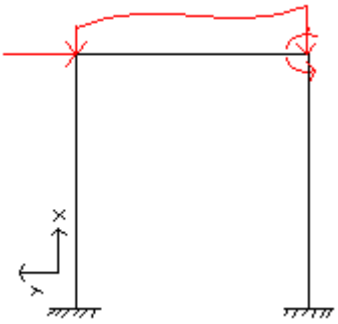


Fig.2

En estas condiciones aparece una flexión plana caracterizada por el siguiente conjunto de ecuaciones básicas. (Ver Fig. 3).

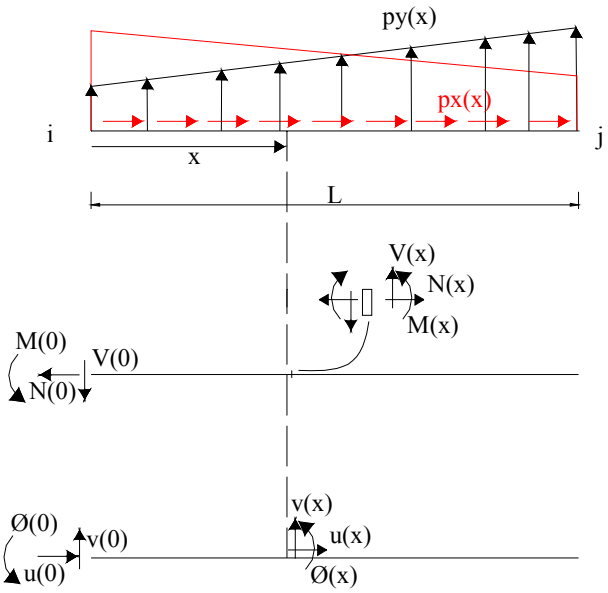


Fig. 3

Las ecuaciones diferenciales para la rebanada son:

Equilibrio:

$$\frac{dN}{dx}+p_x=0 \qquad \frac{dV}{dx}+p_y=0 \qquad \frac{dM}{dx}+V=0 \tag{1a}$$

Compatibilidad:

$$e_n=\frac{du}{dx} \qquad e_v=\frac{dv}{dx}-\phi \qquad e_m=\frac{d\phi}{dx} \tag{1b}$$

Comportamiento:

$$e_n=\frac{N}{EA}+\alpha T \qquad e_v=\frac{V}{GAc} \qquad e_m=\frac{M}{EI} \tag{1c}$$

en las que las variables introducidas son:

- p_x, p_y : fuerzas por unidad de longitud
- N,V,M: esfuerzos internos (axil, cortante y flector)
- e_n, e_v, e_m: deformaciones asociadas a los esfuerzos internos.
- u,v,ϕ: desplazamientos del eje de la barra y giro de la sección.
- α, T: coeficiente de dilatación lineal e incremento de temperatura
- E,G: Módulo de elasticidad y Módulo de cortadura
- A, A_c, I: Área de la sección transversal, área a cortante y momento de inercia.

En el caso de barras esbeltas, pueden despreciarse las deformaciones debidas al esfuerzo cortante, lo que matemáticamente equivale a hacer $A_c\to\infty$ lo que implica que $e_v=0$ si sustituimos en la ecuación correspondiente de comportamiento y $\phi=dv/dx$ de la ecuación de compatibilidad. Las ecuaciones básicas quedan si eliminamos las deformaciones:

Equilibrio:

$$\frac{dN}{dx}+p_x=0 \qquad \frac{dV}{dx}+p_y=0 \qquad \frac{dM}{dx}+V=0 \tag{2a}$$

Compatibilidad-Comportamiento:

$$\frac{du}{dx}=\frac{N}{EA}+\alpha T \qquad \frac{dv}{dx}=\phi \qquad \frac{d\phi}{dx}=\frac{M}{EI} \tag{2b}$$

Al realizar la integración de las ecuaciones (2a) y (2b) aparecen las integrales sucesivas de distintas funciones. Sea $f_g(x)$ una función cualquiera de x, es fácil demostrar integrando por partes que:

$$I_{f_g}^{k}(x)=\underbrace{\int_0^x\left[\int_0^x\left[\int_0^x .....\left[\int_0^x f_g(x)dx\right]dx\right]dx.....\right]dx}_{k-veces}=\frac{1}{(k-1)!}\int_0^x f_g(x')(x-x')dx'$$

Aplicando esto, la integración de las ecuaciones básica es trivial:

$$N(x) = N(0) - I^1_{p_x}(x) \tag{3a}$$

$$u(x) = u(0) + \alpha I^1_T(x) + \frac{1}{EA} I^1_N(x) = u(0) + \alpha I^1_T(x) + \frac{N(0)x}{EA} - \frac{1}{EA} I^2_{p_x}(x) \tag{3b}$$

$$V(x) = V(0) - I^1_{p_y}(x) \tag{3c}$$

$$M(x) = M(0) - I^1_V(x) = M(0) - V(0)x + I^2_{p_y}(x) \tag{3d}$$

$$\phi(x) = \phi(0) + \frac{1}{EI} I^1_M(x) = \phi(0) + \frac{M(0)x}{EI} - \frac{V(0)x^2}{2EI} + \frac{1}{EI} I^3_{p_y}(x) \tag{3e}$$

$$v(x) = v(0) + I^1_\phi(x) = v(0) + \phi(0)x + \frac{M(0)x^2}{2EI} - \frac{V(0)x^3}{6EI} + \frac{1}{EI} I^4_{p_y}(x) \tag{3f}$$

En este trabajo va a considerarse que la temperatura es constante en cada barra (es decir $I^1_T(x) = Tx$) y que las cargas $p_x(x)$ y $p_y(x)$ siguen una ley lineal definida por unos valores q_{1i}, q_{1j}, y q_{2i}, q_{2j} en los nudos extremos:

$$p_x(x) = q_{1i}\left(1 - \frac{x}{L}\right) + q_{1j}\frac{x}{L} \qquad p_y(x) = q_{2i}\left(1 - \frac{x}{L}\right) + q_{2j}\frac{x}{L}$$

en este caso, la integral k-veces de estas funciones quedan para $p_x(x)$:

$$I^k_{p_x}(x) = \frac{x^k\left[q_{1i}L(k+1) + (q_{1j} - q_{1i})x\right]}{Lk(k+1)(k-1)!}$$

siendo para $p_y(x)$ análoga.

Conocidos los esfuerzos internos en el origen de una barra, las ecuaciones anteriores (3a), (3b), (3c), (3d), (3e) y (3f) permiten calcular las leyes de esfuerzos y la deformada de la barra.

2.1.2. Ecuaciones básicas para la barra y la estructura.

Para el cálculo de la estructura por el método de las fuerzas, se han planteado las ecuaciones de forma matricial.

El método usado parte de una visión elemental, es decir a nivel de barra, para posteriormente extrapolarse a nivel de estructura, aunque los conceptos siguen siendo los mismos.

En una barra de un pórtico plano (Fig. 4) tenemos 6 incógnitas, que son los esfuerzos tanto al principio como al final de la barra, y 3 ecuaciones, las 3 ecuaciones de equilibrio globales de la estática; por tanto, a nivel elemental de barra, tenemos que el número de incógnitas desconocidas son 3, es decir, conociendo 3 valores de los esfuerzos de las barras, podemos determinar mediante las ecuaciones de equilibrio de

fuerzas las otras 3. Necesitamos por tanto determinar cuales van a ser nuestras incógnitas o variables, y hemos elegido por sencillez el axil, cortante y flector en el extremo de la barra.

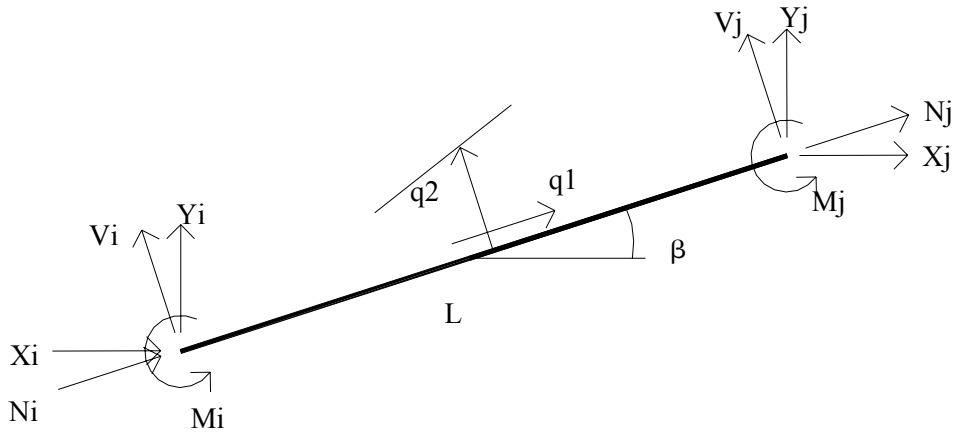


Fig. 4

Consideremos por tanto una barra elástica genérica de longitud L como se muestra en la Fig. 4 sometida a un conjunto de fuerzas N_i, V_i, M_i, N_j, V_j, M_j, en los extremos y a unas cargas distribuidas externas q_1, axiales y q_2 normales a la barra y que varían linealmente de q_{1i} a q_{1j} y de q_{2i} a q_{2j}. Si aplicamos las ecuaciones de la estática (suma de fuerzas y suma de momentos igual a cero) en coordenadas locales, tendremos 3 ecuaciones que nos relacionan 6 variables, por lo que éstas pueden ser expresadas en función de 3 de ellas (N_j, V_j y M_j). Transformando éstas relaciones a coordenadas globales y llamando s=senβ y c=cosβ al seno y coseno del ángulo que forma la barra con los ejes globales, tendremos las ecuaciones de equilibrio de la barra..

$$\begin{pmatrix} X_i \\ Y_i \\ X_j \\ Y_j \\ Z_i \\ Z_j \end{pmatrix} = \begin{pmatrix} -s & 0 & -c \\ -c & 0 & -s \\ -s & 0 & c \\ c & 0 & s \\ -L & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} V_j \\ M_j \\ N_j \end{pmatrix} - \begin{pmatrix} \frac{cL}{2} & \frac{cL}{2} & \frac{-sL}{2} & \frac{-sL}{2} \\ \frac{sL}{2} & \frac{sL}{2} & \frac{cL}{2} & \frac{cL}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{L^2}{6} & \frac{L^2}{3} \\ 0 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} q_{1i} \\ q_{1j} \\ q_{2i} \\ q_{2j} \end{pmatrix} \quad (4)$$

las cuales pueden escribirse de una manera más compacta de la forma:

$$\underline{F}_k=\underline{E}_k\underline{Q}_k+\underline{E}_{kq}\cdot\underline{F}_q \qquad (4b)$$

El vector $\underline{Q}_k$ juega el papel de magnitudes estáticas internas para la barra k y por lo tanto cada columna de la matriz $\underline{E}_k$ constituye un sistema de fuerzas autoequilibrado (de resultante y momento resultante nulos), que es la característica común a cualquier conjunto de magnitudes estáticas internas. En la Fig. 5 se muestra para el caso $\beta=0$ los conjuntos que juegan el papel de esfuerzos internos para la barra.

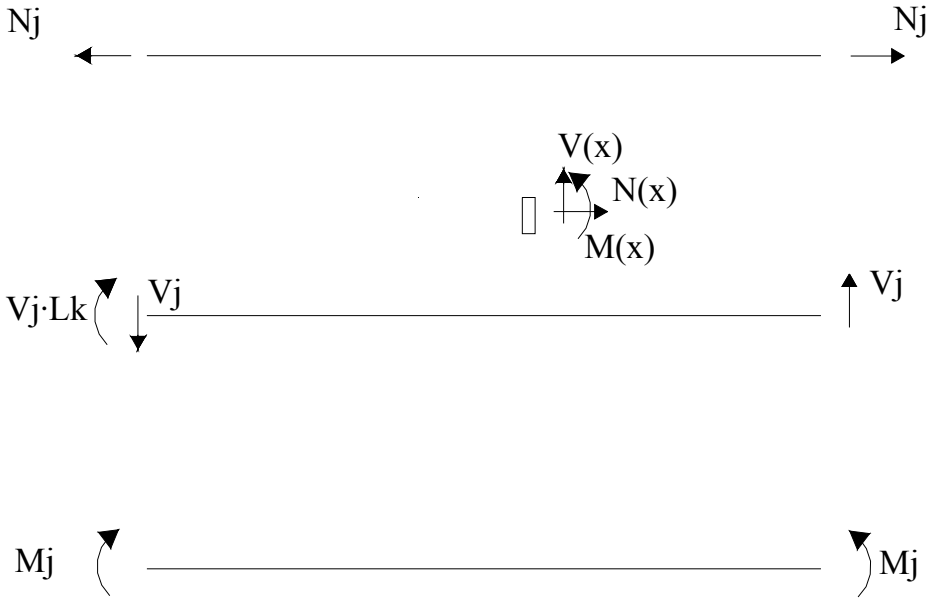


Fig. 5

Las ecuaciones de compatibilidad de la barra se obtienen definiendo las deformaciones asociadas a los esfuerzos internos mediante el Teorema de las Fuerzas Virtuales. De esta manera, definimos el vector $\underline{d}_k=(d_{Vj}\ d_{Mj}\ d_{Nj})^T$ que agrupa las deformaciones discretas de la barra, que hacen trabajo con V_j, M_j y N_j respectivamente. Así mismo, sea $\underline{u}_k=(u_i\ v_i\ u_j\ v_j\ \theta_i\ \theta_j\ )^T$ el vector de desplazamientos y giros de los nudos extremos de la barra k, que hacen trabajo con las acciones externas $\underline{F}_k$. Si aplicamos un conjunto arbitrario de fuerzas externas $\underline{F}^v_k$ y esfuerzos internos virtuales $\underline{Q}^v_k$ en equilibrio, y tomamos nulas las acciones virtuales distribuidas, las ecuaciones de equilibrio virtuales se expresan de la siguiente manera:

$$\underline{F}^v_k=\underline{E}_k\underline{Q}^v_k \qquad (5)$$

Imponiendo la igualdad de trabajos virtuales internos y externos:

$$[\underline{Q}^v_k]^T\underline{d}_k=[\underline{F}^v_k]^T\underline{u}_k \qquad (6)$$

Sustituyendo la ecuación (2) en (3), tenemos que:

$$[\underline{Q}^v_k]^T\underline{d}_k = [\underline{Q}^v_k]^T\underline{E}_k{}^T\underline{u}_k \quad \Rightarrow \quad [\underline{Q}^v_k]^T(\underline{d}_k-\underline{E}_k{}^T\underline{u}_k) = 0 \qquad (7)$$

y como el vector $\underline{Q}^v_k$ es arbitrario, el cumplimiento de la ecuación (7) exige que sea nulo el factor entre paréntesis, es decir que:

$$\underline{d}_k = \underline{E}_k^T \underline{u}_k \tag{8}$$

que son las ecuaciones de compatibilidad, que definen las deformaciones de la barra k en función de los desplazamientos y giros de sus nudos extremos. Desarrollando estas expresiones obtenemos:

$$\begin{pmatrix} d_{Vj} \\ d_{Mj} \\ d_{Nj} \end{pmatrix} = \begin{pmatrix} s & -c & -s & c & -L & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ -c & -s & c & s & 0 & 0 \end{pmatrix} . \begin{pmatrix} u_i \\ v_i \\ u_j \\ v_j \\ \alpha_i \\ \alpha_j \end{pmatrix} \tag{9}$$

El sentido geométrico de las deformaciones se deduce más fácilmente suponiendo que la barra es horizontal habida cuenta que los desplazamientos son invariantes ante giros como sólido rígido. En este caso la ecuación anterior queda:

$$\begin{pmatrix} d_{Vj} \\ d_{Mj} \\ d_{Nj} \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 & 1 & -L & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ -1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} . \begin{pmatrix} u_i \\ v_i \\ u_j \\ v_j \\ \alpha_i \\ \alpha_j \end{pmatrix} \tag{10}$$

donde puede verse que d_{Vj} y d_{Mj} representan el desplazamiento y el giro absolutos del extremo j menos los correspondientes valores debidos a los movimientos como sólido rígido de la barra, y d_{Nj} representa el alargamiento del eje de la barra. La Fig. 6 ilustra el significado geométrico de las deformaciones discretas asociadas a V_j y M_j. La forma más fácil de calcular estas magnitudes cinemáticas es considerar que la barra está empotrada en su extremo izquierdo, en coordenadas locales, y que está sometida a M_j, V_j y N_j en el extremo de la barra y a q_1 y q_2 a lo largo de su longitud. Entonces d_{Vj} y d_{Mj} son simplemente el desplazamiento y el giro del extremo derecho, que se calcula usando las fórmulas de Resistencia de Materiales:

$$\begin{pmatrix} d_{Vj} \\ d_{Mj} \\ d_{Nj} \end{pmatrix} = \begin{pmatrix} \frac{L^3}{3EI} & \frac{L^2}{2EI} & 0 \\ \frac{L^2}{2EI} & \frac{L}{EI} & 0 \\ 0 & 0 & \frac{L}{EA} \end{pmatrix} . \begin{pmatrix} V_j \\ M_j \\ N_j \end{pmatrix} + \begin{pmatrix} 0 & 0 & \frac{L^4}{30EI} & \frac{11L^3}{120EI} \\ 0 & 0 & \frac{L^3}{24EI} & \frac{L^3}{8EI} \\ \frac{L^2}{6EA} & \frac{L^2}{3EA} & 0 & 0 \end{pmatrix} . \begin{pmatrix} q_{1i} \\ q_{1j} \\ q_{2i} \\ q_{2j} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \alpha TL \end{pmatrix} \tag{11}$$

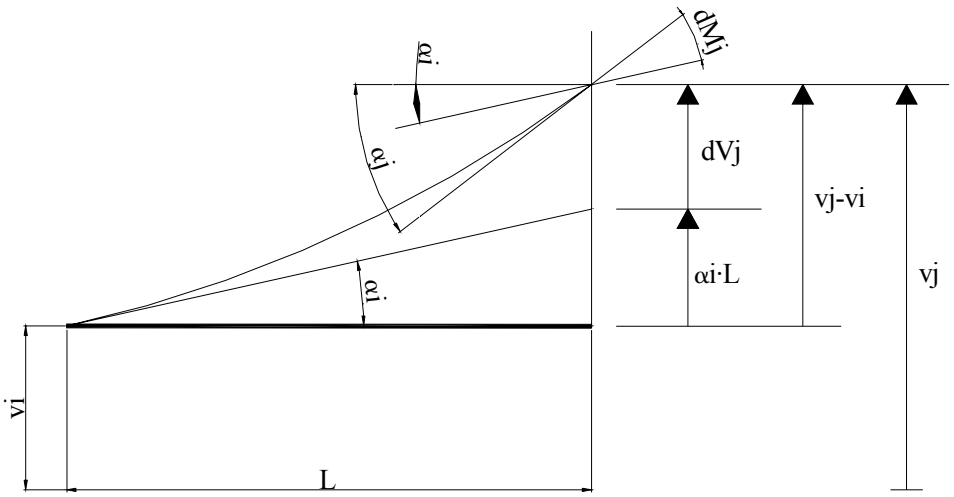


Fig. 6

donde EI y EA son las rigideces a axil y flector, y las deformaciones debidas al esfuerzo cortante no se han tenido en cuenta. Las ecuaciones (11) son las ecuaciones de comportamiento, que completan las ecuaciones básicas de una barra.

Para obtener las ecuaciones discretas de la estructura completa sólo tenemos que conectar las barras en los nudos estática y cinemáticamente. Para ello, supongamos que tenemos una estructura de b barras, r reacciones, l libertades de barras y g grados de libertad permitidos (GLP), y sea $\underline{F}_Q$ el vector de acciones externas en dichos grados de libertad (obtenido de la suma de las acciones concentradas actuantes en dichos grados de libertad más las acciones equivalentes a cargas repartidas, derivadas del vector $\underline{E}_{kq}\underline{q}$ de la ecuación (4b)). Expresando el equilibrio de los grados de libertad permitidos entre acciones en el nudo y acciones de extremo de barras sobre el nudo mediante la introducción de las submatrices de equilibrio elementales en las posiciones adecuadas, obtenemos la matriz de equilibrio global:

$$\underset{(gx3b)}{\underline{E}_Q} \cdot \underset{(3bx1)}{\underline{Q}} = \underset{(gx1)}{\underline{F}_Q} \qquad (12)$$

La aplicación del principio de contragradiencia a las ecuaciones de equilibrio, en una forma similar a como se hizo a nivel de barra, da las ecuaciones globales de compatibilidad:

$$\underset{(3bx1)}{\underline{d}} = \underset{(3bxg)}{\underline{E}_Q^T} \cdot \underset{(gx1)}{\underline{u}} \qquad (13)$$

Finalmente, las ecuaciones globales de comportamiento se obtienen uniendo todas las ecuaciones elementales:

$$\underset{(3bx1)}{\underline{d}} = \underset{(3bx3b)}{\underline{C}} \cdot \underset{(3bx1)}{\underline{Q}} + \underset{(3bx1)}{\underline{d}_q} \qquad (14)$$

Las ecuaciones (12),(13) y (14) constituyen un sistema lineal de g+6b ecuaciones y g+6b incógnitas que son los 3b esfuerzos, las 3b deformaciones asociadas y los g grados de libertad permitidos.

2.2. Emparrillados

Los emparrillados son estructuras como la mostrada en la Fig. 7. Este tipo de estructura plana está sometida únicamente a acciones externas que la deforman en dirección perpendicular a su plano (Fig. 7b). Admitimos que los perfiles utilizados son cerrados y que, consecuentemente, la torsión puede considerarse libre.

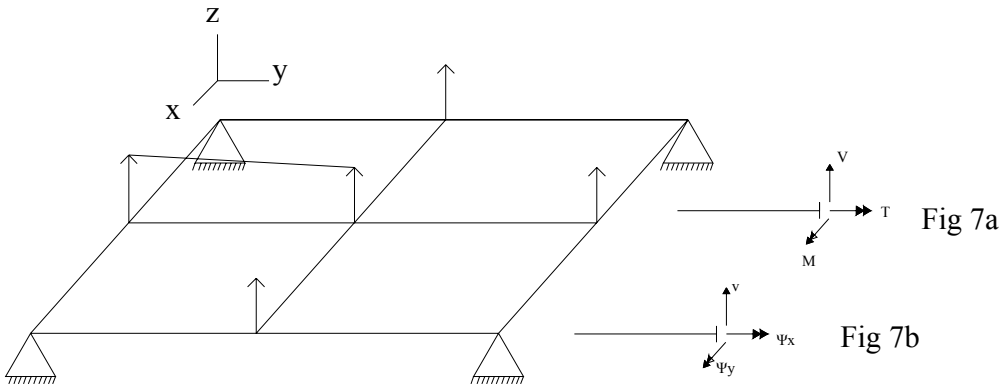


Fig 7

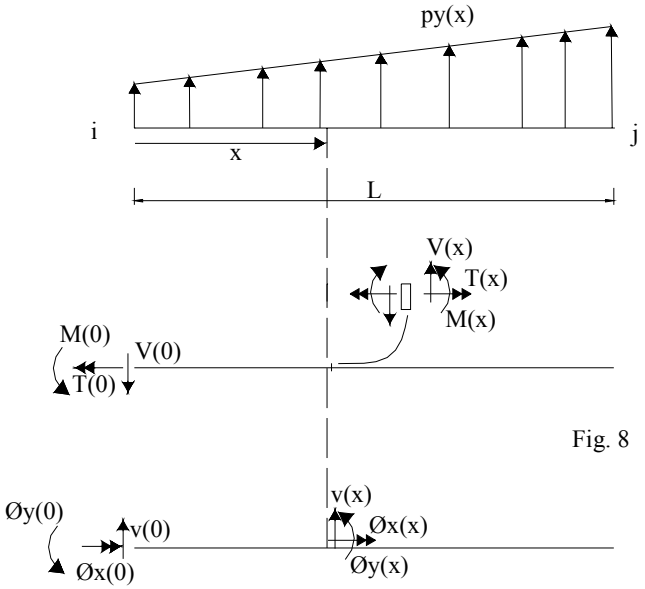


Fig. 8

En un emparrillado, los desplazamientos permitidos son los dos giros fuera del plano de la estructura y el desplazamiento normal al plano de la estructura y los esfuerzos internos de barras son torsor, cortante y flector (Fig. 8).

2.2.1. Ecuaciones básicas a nivel diferencial.

Las ecuaciones básicas relativas a la flexión son idénticas al caso de pórticos planos, y las relativas a la torsión son:

Equilibrio: $$\frac{dT}{dx} = 0 \qquad (15a)$$

Compatibilidad: $$e_T = \frac{d\phi_x}{dx} \qquad (15b)$$

Comportamiento: $$e_T = \frac{T}{GJ} \qquad (15c)$$

Siendo T el momento torsor, e_{mx} su deformación asociada, ϕ_x el ángulo de torsión y J la constante torsional, que para perfiles de pared delgada cerrados de espesor constante (e) viene dada por:

$$J = \frac{4\Omega^2 e}{S}$$

siendo Ω el área encerrada por la línea media de la sección y S su longitud. La constante torsional viene tabulada en las tablas de perfiles bajo la denominación I_t.

La integración de las ecuaciones (15a), (15b) y (15c) es ahora:

$$T(x)=T(0) \qquad (16a)$$

$$\phi_x(x) = \phi_x(0) + \frac{T(0)x}{GJ} \qquad (16b)$$

2.2.2. Ecuaciones básicas para la barra y la estructura.

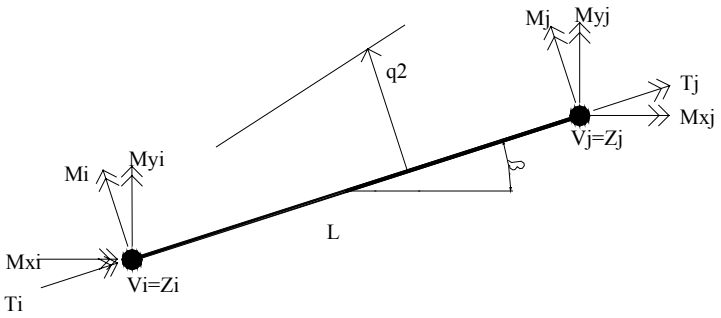


Fig. 9

Un desarrollo conceptualmente idéntico al realizado para pórticos planos permite obtener las ecuaciones básicas para la barra. Así, de los 6 esfuerzos en coordenadas locales (ver Fig. 9) se han elegido ahora T_j, M_j y V_j como esfuerzos internos. Las

ecuaciones asociadas al problema de flexión son idénticas a las de pórticos planos, en cuanto a la torsión, la deformación asociada a T_j (d_{Tj}) es de giro relativo del extremo j respecto al i. Las ecuaciones básicas son:

Equilibrio.

$$\begin{pmatrix} Mx_i \\ Mz_i \\ Y_i \\ Mx_j \\ Mz_j \\ Y_j \end{pmatrix} = \begin{pmatrix} -c & s & sL \\ -s & -c & -cL \\ 0 & 0 & -1 \\ c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} T_j \\ M_j \\ V_j \end{pmatrix} - \begin{pmatrix} \frac{-sL^2}{6} & \frac{-sL^2}{3} \\ \frac{cL^2}{6} & \frac{cL^2}{3} \\ \frac{L}{2} & \frac{L}{2} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} . \begin{pmatrix} q_i \\ q_j \end{pmatrix} \tag{17a}$$

las cuales pueden escribirse de una manera más compacta de la forma:

$$\underline{F_k} = \underline{E_k Q_k} + \underline{E_{qk} F_q} \tag{17b}$$

Nótese que sólo se han considerado fuerzas repartidas en dirección normal a al emparrillado, habida cuenta que es poco frecuente en la práctica la presencia de momentos torsores distribuidos.

Compatibilidad:

$$\begin{pmatrix} d_{Tj} \\ d_{Mj} \\ d_{Vj} \end{pmatrix} = \begin{pmatrix} -c & -s & 0 & c & s & 0 \\ s & -c & 0 & -s & c & 0 \\ sL & -cL & -1 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} \theta x_i \\ \theta z_i \\ v_i \\ \theta x_j \\ \theta z_j \\ v_j \end{pmatrix} \tag{18}$$

Comportamiento:

$$\begin{pmatrix} d_{Tj} \\ d_{Mj} \\ d_{Vj} \end{pmatrix} = \begin{pmatrix} \frac{L}{GJ} & 0 & 0 \\ 0 & \frac{L}{EI} & \frac{L^2}{2EI} \\ 0 & \frac{L^2}{2EI} & \frac{L^3}{3EI} \end{pmatrix} . \begin{pmatrix} T_j \\ M_j \\ V_j \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ \frac{L^3}{24EI} & \frac{L^3}{8EI} \\ \frac{L^4}{30EI} & \frac{11L^3}{120EI} \end{pmatrix} . \begin{pmatrix} q_i \\ q_j \end{pmatrix} \tag{19}$$

donde EI y GJ son las rigideces a flector y torsor, y las deformaciones debidas al esfuerzo cortante no se han tenido en cuenta.

Las matrices básicas para la estructura se establecen análogamente a como se comentó en pórticos planos, obteniéndose ecuaciones formalmente idénticas a las ecuaciones (12), (13) y (14).

2.3 Planteamiento analítico del cálculo de la estructura.

Ya hemos visto que el sistema de ecuaciones (12), (13) y (14) permiten obtener los esfuerzos internos en los extremos de las barras y los desplazamientos de los grados de libertad no coaccionados.

Sin embargo, el objeto de este trabajo es elaborar un programa de ayuda a la docencia, en el que se vayan reproduciendo los resultados analíticos parciales que se obtienen en el análisis manual de la estructura mediante el Método de las Fuerzas, siguiendo los pasos descritos en el apartado 1.2. Es interesante comentar que el álgebra matricial necesaria para la obtención de estos resultados analíticos parciales es más compleja que la involucrada en el planteamiento global de las ecuaciones (12), (13) y (14).

Las etapas a desarrollar en el cálculo manual de la estructura se detallan a continuación en los siguientes subapartados. En principio, el desarrollo se va a hacer para pórticos planos, aunque la metodología es directamente aplicable a emparrillados.

2.3.1. Elección de incógnitas hiperestáticas.

Manualmente, la fórmula más conveniente para calcular el grado de hiperestaticidad de un pórtico plano es:

$$h = (3c + r) - (3 + l) \qquad (20a)$$

siendo c el número de cortes necesarios para convertir la estructura en abierta (es decir, el número de bucles cerrados), r el número de reacciones y l el número de libertades internas de las barras. La fórmula (20a), basada en un planteamiento global del grado de hiperestaticidad, es la más conveniente para el cálculo manual porque involucra un número menor de incógnitas y ecuaciones que la fórmula alternativa:

$$h = 3b - (3n + l - r) = 3b - g \qquad (20b)$$

basada en un planteamiento local (barra a barra), donde n el número de nudos, y que sería la que se obtendría de las ecuaciones (12). La comparación de las ecuaciones (20a) y (20b) muestra que:

$$c = b - n + 1 \qquad (20c)$$

que da el número de bucles cerrados de una estructura plana.

La ecuación (20a) está asociada a una matriz de equilibrio reducida que relaciona 3c+r incógnitas (los 3 esfuerzos internos en cada uno de los c cortes, más el conjunto de las reacciones) mediante 3 ecuaciones globales de la estática, más l ecuaciones asociadas a las libertades cinemáticas de los extremos de barras, de las que sólo se ha considerado la libertad rotacional, es decir, la posibilidad de existencia de rótulas internas. Estas ecuaciones de equilibrio reducidas tienen por tanto la forma:

$$\underset{(3+l)x(r+3c)}{\underline{E}_{RQ_c}} \cdot \underset{(r+3c)x1}{\underline{RQ}_c} = \underset{(3+l)x1}{\underline{F}_{RQ_c}} \tag{21a}$$

en las que $\underline{RQ}_c$ es un vector que contiene las reacciones y los esfuerzos internos en los c cortes en los extremos de barras, que son escogidos por el usuario:

$$\underset{(r+3c)x1}{\underline{RQ}_c} = \left(\underset{(rx1)}{\underline{R}^T} \; \underset{(3cx1)}{\underline{Q}_c^T} \right)^T \tag{21b}$$

En todo el desarrollo analítico los vectores $\underline{R}$ y $\underline{Q}_c$ se manejan como vectores simbólicos, usando X[i], Y[i], Z[i] para las reacciones en el nudo i y N[k], V[k] y M[k] para los esfuerzos internos en el extremo de la barra k.

En el caso de estructuras abiertas (c=0) no existe $\underline{Q}_c$ y se tomará en general $\underline{RQ}_c=\underline{R}$, ya que todas las hiperestáticas pueden ser reacciones. Sin embargo, dado que en cualquier estructura todas las hiperestáticas pueden ser esfuerzos internos también se ha dado la posibilidad, en estructuras abiertas, de escoger hiperestáticas internas y externas; en este caso $\underline{RQ}_c$ será la unión de $\underline{R}$ y los esfuerzos internos elegidos como base para la elección de posibles hiperestáticas.

En cualquier caso, $\underline{RQ}_c$ será un subconjunto del conjunto $\underline{RQ}$, de r+3b componentes, y para obtener las ecuaciones de equilibrio reducidas (21a) vamos a ampliar las ecuaciones (12) para incluir en ellas el vector de reacciones. Para ello basta realizar el equilibrio de los grados de libertad coaccionados, obteniéndose:

$$\underset{(3+l)x(r+3c)}{\underline{E}_R} \cdot \underset{(r+3c)x1}{\underline{Q}} = \underset{rx1}{\underline{R}} + \underset{(3+l)x1}{\underline{F}_R} \tag{22}$$

siendo $\underline{R}$ el vector de reacciones y $\underline{F}_R$ el vector de fuerzas equivalentes debidas a cargas repartidas aplicadas en barras que concurren a nudos soportes. Pasando el vector $\underline{R}$ al primer término y uniendo las ecuaciones (22) a las ecuaciones (12) obtenemos las ecuaciones de equilibrio ampliadas:

$$\begin{pmatrix} \underline{E}_Q & \underline{0} \\ \underline{E}_R & -\underline{I} \end{pmatrix} \cdot \begin{pmatrix} \underline{Q} \\ \underline{R} \end{pmatrix} = \begin{pmatrix} \underline{F}_Q \\ \underline{F}_R \end{pmatrix} \tag{23a}$$

o bien en forma compacta:

$$\underset{(g+r)x(3b+r)}{\underline{E}_{QR}} \cdot \underset{(3b+r)x1}{\underline{QR}} = \underset{(g+r)x1}{\underline{F}_{QR}} \tag{23b}$$

Dividimos ahora las ecuaciones (23b) de la forma:

$$\underset{(g+r)x(r+3c)}{\underline{E}_{QR1}} \cdot \underset{(r+3c)x1}{\underline{RQ}_c} + \underset{(g+r)x(3b-3c)}{\underline{E}_{QR2}} \cdot \underset{(3b-3c)x1}{\overline{\underline{RQ}}_c} = \underset{(g+r)x1}{\underline{F}_{QR}} \tag{23c}$$

siendo $\overline{\underline{RQ}}_c$ las componentes de $\underline{RQ}$ no incluidas en $\underline{RQ}_c$. Observemos ahora que $\underline{E}_{RQ2}$ es una matriz "vertical", es decir una matriz con más filas que columnas puesto que :

$$g+r-(3b-3c)=3n+l-r+r-3b+3b-3n+3=3+l \tag{23d}$$

que es el número de ecuaciones de equilibrio reducidas [Ver ecuación (21a)]. Existirá por tanto una "matriz núcleo por la izquierda" [Referencia STRANG], definida como cualquier matriz de rango (3b-3c) que cumpla:

$$\underset{(3+l)x(g+r)}{\underline{E}^0_{QR2}} \cdot \underset{(g+r)x(3b-3c)}{\underline{E}_{QR2}} = \underset{(3+l)x(3b-3c)}{\underline{0}} \tag{23e}$$

La matrices núcleo se calculan en Mathematica con NullSpace.

Ahora, para eliminar $\overline{\underline{RQ}}_c$ de las ecuaciones (23c) basta premultiplicar por la matriz núcleo de $\underline{E}_{QR2}$:

$$\underset{(3+l)x(g+r)}{\underline{E}^0_{QR2}} \cdot \underset{(g+r)x(r+3c)}{\underline{E}_{QR1}} \cdot \underset{(r+3c)x1}{\underline{RQ}_c} = \underset{(3+l)x(g+r)}{\underline{E}^0_{QR2}} \cdot \underset{(g+r)x1}{\underline{F}_{QR}} \tag{23f}$$

y comparando con las ecuaciones (21a) vemos que:

$$\underline{E}_{RQ_c} = \underline{E}^0_{QR2} \cdot \underline{E}_{QR1} \tag{24a}$$

$$\underline{F}_{RQ_c} = \underline{E}^0_{QR2} \cdot \underline{F}_{QR} \tag{24b}$$

En general, este procedimiento puramente matricial no da las ecuaciones de equilibrio reducidas en la forma lógica en que dichas ecuaciones se escribirían en un cálculo manual (primero las 3 ecuaciones de la estática y luego las asociadas a libertades), sino como una combinación lineal de éstas. Por ello el programa suministra 2 versiones de estas ecuaciones: tal como se obtienen en el algoritmo anterior y triangulada canónicamente.

En este punto, el usuario, a la vista de la matriz de equilibrio, ha de escoger un posible conjunto de h incógnitas hiperestáticas, que será un subconjunto de $\underline{RQ}_c$ y por tanto un subconjunto de $\underline{RQ}$. A partir de la elección de del usuario, el conjunto de incógnitas hiperestáticas se define matricialmente mediante una matriz $\underline{H}$ (hx(3b-r)), en cada una de cuyas filas hay un 1 en la posición asociada a cada hiperestática, y el resto son ceros:

$$\underline{QR}_h = \underline{H} \cdot \underline{QR} \tag{25a}$$

Añadiendo estas ecuaciones a las ecuaciones (23b), obtenemos:

$$\underset{(3b+r)x(3b+r)}{\begin{pmatrix} \underline{H} \\ \underline{E}_{QR} \end{pmatrix}} \cdot \underline{QR} = \begin{pmatrix} \underline{QR}_h \\ \underline{F}_{QR} \end{pmatrix} \tag{25b}$$

Si el conjunto de incógnitas hiperestáticas es admisible, se pueden eliminar sus vínculos cinemáticos asociados y la estructura resultante debe ser una estructura isostática que no sea un mecanismo. Esto equivaldría a resolver el sistema (25b) suponiendo $\underline{QR}_h=\underline{0}$, y ello sólo es posible si el determinante de la matriz que multiplica a $\underline{QR}$ en la ec. (25b) es distinto de cero, lo cual indica que el conjunto elegido de hiperestáticas es admisible. En este caso, el programa proporciona una tercer versión de la matriz de equilibrio reducida $\underline{E}_{RQc}$, en la que se triangulan las columnas asociadas a variables no hiperestáticas.

Si las hiperestáticas son admisibles, la inversión del sistema (25b):

$$\begin{pmatrix}\underline{Q}\\ \underline{R}\end{pmatrix}=\begin{pmatrix}\underline{H}\\ \underline{E}_{QR}\end{pmatrix}^{-1}\begin{pmatrix}(\underline{QR})_h\\ \underline{F}_{QR}\end{pmatrix}=\begin{pmatrix}\underline{E}_Q^0 & \underline{E}_Q^{-1}\\ \underline{E}_R^0 & \underline{E}_R^{-1}\end{pmatrix}\begin{pmatrix}(\underline{QR})_h\\ \underline{F}_{QR}\end{pmatrix} \qquad (25c)$$

permite obtener las reacciones $\underline{R}$ y los esfuerzos internos $\underline{Q}$ en los extremos de las barras en función de las cargas externas y las incógnitas hiperestáticas.

2.3.2. Esfuerzos internos en las barras en función de las incógnitas hiperestáticas.

A partir de $\underline{Q}$, la integración de las ecuaciones de equilibrio tomando como condiciones iniciales el valor de los esfuerzos en los extremos de las barras, permite obtener las leyes de esfuerzos a lo largo de cada barra. En forma matricial:

$$\begin{pmatrix}N(x)\\ V(x)\\ M(x)\end{pmatrix}=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & L-x & 1\end{pmatrix}\cdot\begin{pmatrix}N_j\\ V_j\\ M_j\end{pmatrix}+$$
$$\begin{pmatrix}\frac{L}{2}-x+\frac{x^2}{2L} & \frac{L}{2}-\frac{x^2}{2L} & 0 & 0\\ 0 & 0 & \frac{L}{2}-x+\frac{x^2}{2L} & \frac{L}{2}-\frac{x^2}{2L}\\ 0 & 0 & \frac{L^2}{6}-\frac{L\cdot x}{2}+\frac{x^2}{2}-\frac{x^3}{6L} & \frac{L^2}{3}-\frac{L\cdot x}{2}+\frac{x^3}{6L}\end{pmatrix}\cdot\begin{pmatrix}q_{1i}\\ q_{1j}\\ q_{2i}\\ q_{2j}\end{pmatrix} \qquad (26)$$

en la que N_j, V_j y M_j se obtienen de las ecuaciones (25c).

Análogamente, en el caso de emparrillados se obtiene:

$$\begin{pmatrix} T(x) \\ V(x) \\ M(x) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & L-x & 1 \end{pmatrix} \cdot \begin{pmatrix} T_j \\ V_j \\ M_j \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ \frac{L}{2} - x + \frac{x^2}{2L} & \frac{L}{2} - \frac{x^2}{2L} \\ \frac{L^2}{6} - \frac{L \cdot x}{2} + \frac{x^2}{2} - \frac{x^3}{6L} & \frac{L^2}{3} - \frac{L \cdot x}{2} + \frac{x^3}{6L} \end{pmatrix} \cdot \begin{pmatrix} q_i \\ q_j \end{pmatrix} \quad (27)$$

2.3.3.Cálculo de incógnitas hiperestáticas

El planteamiento matricial que hemos realizado permite calcular el valor de las hiperestáticas matricialmente también, pero este programa tiene un enfoque docente y el uso del método matricial para el cálculo de hiperestáticas no proporciona las ecuaciones normalmente usadas por los alumnos, que habitualmente usan el PFV. Por este motivo se ha usado el PFV en el cálculo de hiperestáticas, aunque a continuación se explica de forma breve la metodología que se debería haber usado para el cálculo matricial de hiperestáticas.

2.3.3.1 Método Matricial.

El cálculo de las hiperestáticas se puede realizar matricialmente, usando las ecuaciones de compatibilidad-comportamiento (9) y (11). Manipulando estas ecuaciones podemos calcular las incógnitas hiperestáticas.

Tenemos de antes que:

$$\underline{d} = \underline{E}^T_Q \, \underline{u} \quad (28)$$

$$\underline{d} = \underline{C} \, \underline{Q} + \underline{d}_q \quad (29)$$

Puede demostrarse [Referencia 1] que la condición necesaria y suficiente para que el sistema sobredeterminado (28) tenga una solución única de desplazamientos es:

$$\left(\underline{E}^T_Q\right)^0 \cdot \underline{d} = \underline{0} \quad (30)$$

siendo $\left(\underline{E}^T_Q\right)^0$ la matriz núcleo de $\underline{E}_Q{}^T$. Sustituyendo el valor de (29) y (26) en (30) y operando, tenemos que:

$$\left(\underline{E}^T_Q\right)^0 \cdot \left(\underline{C} \cdot \underline{Q} + \underline{d}_q\right) = \left(\underline{E}^T_Q\right)^0 \cdot \underline{C} \cdot \left(\underline{E}^0_Q \cdot \underline{QR}_h + \underline{E}^{-1}_Q \cdot \underline{F}_Q\right) + \left(\underline{E}^T_Q\right)^0 \cdot \underline{d}_q = 0 \quad (31a)$$

$$\left(\left(\underline{E}_Q^T\right)^0\cdot\underline{C}\cdot\underline{E}_Q^0\right)\cdot\underline{QR}_h=-\left(\underline{E}_Q^T\right)^0\cdot\underline{C}\cdot\underline{E}_Q^{-1}\cdot\underline{F}_Q-\left(\underline{E}_Q^T\right)^0\cdot\underline{d}_q \qquad (31b)$$

$$\underline{QR}_h=\left(\left(\underline{E}_Q^T\right)^0\cdot\underline{C}\cdot\underline{E}_Q^0\right)^{-1}\cdot\left(-\left(\underline{E}_Q^T\right)^0\cdot\underline{C}\cdot\underline{E}_Q^{-1}\cdot\underline{F}_Q-\left(\underline{E}_Q^T\right)^0\cdot\underline{d}_q\right) \qquad (31c)$$

Todas esta matrices son conocidas de antes por lo tanto el cálculo de las hiperestáticas no tiene más dificultad que realizar las operaciones matriciales que aparecen en la ecuación (31c).

2.3.3.2. Uso del PFV.

El método usado en el programa para calcular las hiperestáticas es el PFV. El PFV es una forma energética de integrar las ecuaciones de compatibilidad-comportamiento. En pórticos planos, el resultado de la integración de estas ecuaciones despreciando la deformación debida al cortante es:

$$F_i^v\cdot\delta_i+R_i^v\cdot\delta_r=\sum_{k=1}^{b}\left(\int_0^{L_k}N_i^v(x)\cdot\left(\frac{N(x)}{EA_k}+\alpha\cdot T_k\right)dx+\int_0^{L_k}M_i^v(x)\cdot\frac{M(x)}{EI_k}dx\right),\quad i=1,h. \qquad (32)$$

donde F_i^v es la acción virtual asociada a la hiperestática i, δ_i^v su desplazamiento asociado; R_i^v el conjunto de reacciones virtuales y δ_r es el desplazamiento (si los hubiera) de los apoyos; E es el módulo de elasticidad del material, A_k, I_k y L_k son el área, inercia y longitud correspondientes a la barra de la cual se están integrando las leyes de esfuerzos; N(x) y M(x) son las leyes de esfuerzos de axiles y momentos respectivamente reales en equilibrio; $N_i^v(x)$, $M_i^v(x)$ los esfuerzos virtuales en equilibrio; α es el coeficiente de dilatación térmica del material y T_k es el incremento sobre la temperatura ambiente a la que está sometida la barra (solo se considerará en el programa el caso de un incremento constante de la temperatura de la barra). Las ecuaciones (32) representan h ecuaciones con h incógnitas $\underline{QR}_h$, de las que pueden despejarse éstas analíticamente, en función de las acciones externas y de las características geométricas de las barras (A_k, I_k; k=1, b).

Cada estado virtual en equilibrio se obtiene a partir del estado real haciendo 1 la incógnita hiperestática i, cero todas las demás, y anulando las cargas externas. Por tanto de las ecuaciones (25c), (26) y (27) se obtienen de forma inmediata los estados virtuales necesarios para aplicar la ecuación (32) y calcular las incógnitas hiperestáticas. Las integraciones analíticas se realizan mediante la función “Integrate” del programa Mathematica.

En el caso de emparrillados, despreciamos nuevamente las deformaciones debidas al cortante, las ecuaciones de compatibilidad se escriben:

$$F_i^v\cdot\delta_i+R_i^v\cdot\delta_r=\sum_{k=1}^{b}\left(\int_0^{L_k}T_i^v(x)\cdot\frac{T(x)}{GJ_k}dx+\int_0^{L_k}M_i^v(x)\cdot\frac{M(x)}{EI_k}dx\right),\ i=1,h \qquad (33)$$

siendo ahora J_k la constante torsional de la barra k.

2.3.4.Cálculo de desplazamientos.

En general, los desplazamientos de ciertos puntos de la estructura están acotados por motivos funcionales (criterio de rigidez).

Para el diseño de la estructura por métodos manuales (que se comentarán en el apartado 3) es más útil disponer de las expresiones de los desplazamientos en función de las acciones externas y de las incógnitas hiperestáticas, en vez de su expresión final una vez sustituidos los valores de éstas últimas. Estas expresiones podrían obtenerse usando nuevamente el PFV, pero el cálculo de los estados virtuales no es tan inmediato, por lo que es más cómodo y rápido (computacionalmente hablando), realizar un cálculo matricial.

La aplicación permite tener en cuenta temperaturas (para pórticos planos) y movimiento de los apoyos, por lo tanto estos dos efectos deberán ser introducidos en el cálculo de desplazamientos. Rescribimos las ecuaciones de equilibrio ampliadas (23a):

$$\underline{E}_Q \cdot \underline{Q} = \underline{F}_Q = \underline{F}_{Qc} + \underline{F}_{Qq} \tag{34a}$$

$$\underline{E}_R \cdot \underline{Q} = \underline{F}_{RR} = \underline{F}_{Rc} + \underline{F}_{Rq} = \underline{R} + \underline{F}_R \tag{34b}$$

Estas ecuaciones son iguales que las (23a), simplemente se ha querido resaltar el carácter puntual o distribuido de las cargas. $\underline{F}_Q$ es el vector de cargas externas aplicadas en los nudos con grados de libertad permitidos, y se dividen como se comentó en el apartado 2.1.2 en $\underline{F}_{Qc}$ de cargas concentradas y $\underline{F}_{Qq}$ de acciones equivalentes a cargas repartidas; $\underline{F}_{RR}$ es el vector de cargas externas aplicadas en los grados de libertad coaccionados donde $\underline{F}_{Rc}$ son las reacciones y cargas externas puntuales y $\underline{F}_{Rq}$ las equivalentes a cargas repartidas. Como vemos en la ecuación (34b), la suma de $\underline{F}_{Rc}$ y $\underline{F}_{Rq}$ nos dan el conjunto de reacciones $\underline{R}$ y el vector de cargas externas (concentradas y equivalentes distribuidas) $\underline{F}_R$.

Tengamos también las ecuaciones de comportamiento globales de la estructura (ecuaciones (14)) introduciendo el término $\underline{d}_T$ donde tendremos las deformaciones correspondientes a la temperatura:

$$\underline{d} = \underline{C}\,\underline{Q} + \underline{d}_q + \underline{d}_T \tag{35}$$

Si aplicamos el principio de contragradiencia a la estructura global y con las cargas externas que aparecen en las ecuaciones (34a) y (34b) $\underline{F}_Q$ y $\underline{F}_{RR}$ tendremos que se tiene que cumplir que:

$$[\underline{Q}^v]^T \underline{d} = [\underline{F}_Q^v]^T \underline{u} + [\underline{F}_{RR}^v]^T \underline{u}_r \tag{36a}$$

donde el superíndice v indica el estado virtual en equilibrio, $\underline{u}$ son los desplazamientos de los grados de libertad de la estructura, es decir tiene de dimensiones gx1, y $\underline{u}_r$ son los desplazamientos de los apoyos de dimensiones rx1. Operando con (34a), (34b) y (36a) tenemos:

$$[\underline{Q}^v]^T \underline{d} = [\underline{Q}^v]^T E_Q^T \underline{u} + [\underline{Q}^v]^T \underline{E}_R^T \underline{u}_r \tag{36b}$$

como esto tiene que cumplirse para cualquier campo virtual en equilibrio, se cumple que:

$$[\underline{Q}^{v}]^{T}(\underline{d}-E_{Q}^{T}\underline{u}-\underline{E}_{R}^{T}\underline{u}_{r})=0\Rightarrow\underline{d}=E_{Q}^{T}\underline{u}+\underline{E}_{R}^{T}\underline{u}_{r} \qquad (36c)$$

usando las ecuaciones (35) y (36c), tendremos que los desplazamientos se calculan como:

$$\underset{(gx1)}{\underline{u}}=\underset{(gx3b)}{[\underline{E}_{Q}^{T}]^{-1}}[\underset{(3bx3b)}{\underline{C}}\underset{(3bx1)}{\underline{Q}}+\underset{(3bx1)}{\underline{d}_{q}}+\underset{(3bx1)}{\underline{d}_{T}}-\underset{(3bxr)}{\underline{E}_{R}^{T}}\underset{(rx1)}{\underline{u}_{r}}] \qquad (36d)$$

donde la matriz $[\underline{E}_{Q}^{T}]^{-1}$ es una submatriz transpuesta de $\underline{E}_{Q}^{-1}$ que aparece en la ecuación (25c) y que tendrá dimensiones gx3b y $\underline{Q}$ se determina según las ecuaciones (25c)

La ecuación (36d) nos permite obtener un subconjunto de los desplazamientos de la estructura $\underline{u}$ en función de las incógnitas hiperestáticas, las cargas externas y los movimientos de los apoyos.

3. DISEÑO DE LA ESTRUCTURA.

3.1. Introducción.

El objetivo final del diseño es determinar las propiedades geométricas de las barras que vamos a necesitar. Es decir, en un caso general, tendremos un sistema de barras el cual se pretende que cumpla unos requisitos de resistencia última y de servicio, las cuales vendrán impuestas por las normas de construcción así como por las necesidades propias por las cuales se construye dicha estructura. Estos criterios son el criterio de resistencia, donde imponemos una cota máxima en tensiones para evitar que la barra llegue a su límite elástico y el criterio de rigidez, donde imponemos por motivos funcionales una cota máxima a los desplazamientos.

Una vez terminada el análisis simbólico del problema, todas las incógnitas del problema dependen de las características geométricas de las barras. Para poder determinar cuales son los perfiles más adecuados, tendremos que imponer un objetivo y una serie de restricciones ya que el proceso de cálculo de una estructura es un proceso de optimización.

Objetivo: Búsqueda del perfil de menor área que cumpla las restricciones.

Restricciones: Las restricciones pueden ser de 3 tipos:

1. Restricción por esbeltez de la barra: Las barras sometidas a compresión pueden presentar una situación de equilibrio inestable conocido como pandeo. Para el cálculo de las piezas sometidas a compresión se ha usado el método de las ω's que aunque se ha quedado un poco desfasado según la nueva normativa para el cálculo de estructuras (Eurocódigo nº 3), para el fin docente de la aplicación, es mucho más interesante. La condición de esbeltez se ha impuesto como una cota máxima de esta misma propiedad de las barras (λ_{max}) y tiene una carácter meramente práctico.

2. Restricción por resistencia: El criterio impuesto es que la sección de la barra más solicitada no plastifique según el criterio de Von-Mises, es decir $\sqrt{\sigma^2 + 3\tau^2} \leq \sigma_E$, donde σ y τ se calcularán con las fórmulas correspondientes según estemos en el caso de pórticos planos o emparrillados.

3. Restricción por rigidez: Las estructuras, por motivos funcionales a veces no deben pasar de unos valores máximos en desplazamientos cuyo valor se recoge en las normas pertinentes.

3.2. Algoritmo general de diseño.

El algoritmo general de diseño para estructuras hiperestáticas es un proceso iterativo que nos permite determinar cuales son los perfiles más adecuados para el problema que tenemos. Este algoritmo como puede verse en la Fig. 10 tiene una serie de pasos.

La elección inicial de las características geométricas de las barras será un paso necesario en el caso de estructuras hiperestáticas. La elección de las propiedades de las barras para comenzar a iterar no es un proceso único, existiendo diversas posibilidades. Sin embargo, la práctica demuestra que el proceso de diseño no suele alargarse sensiblemente si se parte de un conjunto arbitrario de características. Normalmente suele ser suficiente con coger los valores de un perfil intermedio.

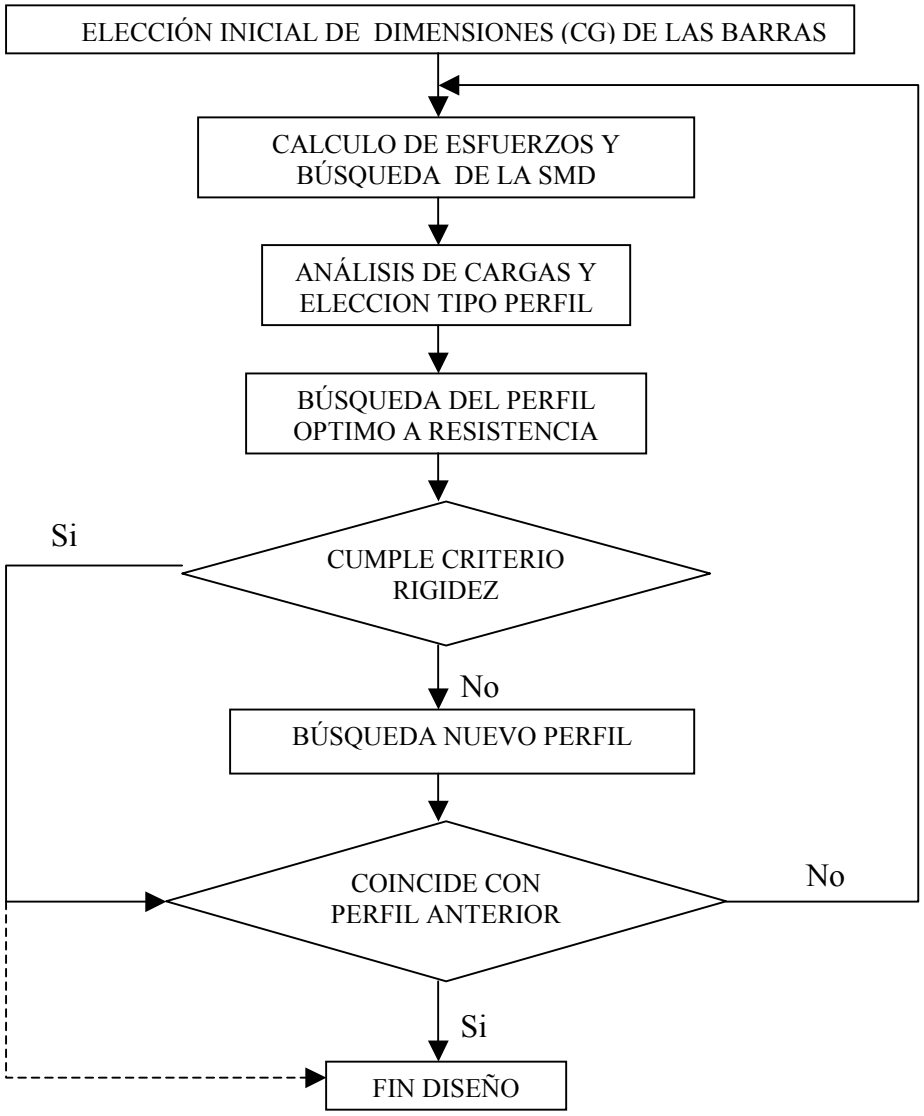


Fig. 10

La elección de los valores iniciales de las características geométricas de las barras no será un paso necesario cuando estemos trabajando con estructuras isostáticas y pseudohiperestáticas. El motivo de ello es que en las estructuras isostáticas, las leyes de esfuerzos no dependen de ninguna incógnita hiperestática, y en las estructuras pseudohiperestáticas, se da la particularidad de que las incógnitas hiperestáticas no dependen de las características geométricas de las barras y por lo tanto, las leyes de esfuerzos tampoco.

3.3. Análisis de esfuerzos y elección del tipo de perfil.

El motivo de usar perfiles conformados es usar de la manera más eficiente posible el acero. Hay perfiles que trabajan mejor bajo un tipo de esfuerzos que otros, y este es el motivo de realizar un análisis de las esfuerzos, con objeto de elegir qué geometría de perfil es más adecuada.

Como puede verse en las Fig. 11 y 12, sólo existen una serie de perfiles que podemos elegir, luego puede parecer en un principio que los diagramas de flujo dibujados no son necesarios o que son redundantes en el sentido que simplemente imponiendo las condiciones necesarias para cada perfil podríamos solucionar el problema de la elección de perfiles. Esto es completamente cierto; sin embargo, como se comentará posteriormente, estos diagramas de flujos no sólo están orientados a la determinación del tipo de perfil, sino a la determinación de los esfuerzos actuantes sobre la barra. Este doble objetivo es buscado en un principio como se ha dicho, para determinar el tipo de perfil adecuado y para ofrecer una información de carácter docente que sólo puede ser dada realizando un análisis de los esfuerzos.

El programa permite realizar el proceso de diseño automáticamente. Cuando se decida usar esta opción, los algoritmos que aparecen en las Fig. 11 y 12 serán los algoritmos directores de la elección de los perfiles más adecuados.

Como las cargas asociadas a pórticos planos y a emparrillados son diferentes, los algoritmos de análisis de esfuerzos, aunque serán parecidos, son diferentes.

3.3.1 Análisis de esfuerzos para pórticos planos.

Sabemos que los pórticos planos están sometidos a axil, cortante y flector (N, V, M), y cada uno de estos esfuerzos crea una distribución de tensiones que condicionará la elección del perfil más adecuado.

El algoritmo que aparece en la Fig. 11 busca cuál es el perfil más idóneo según los esfuerzos a los que esté sometida la barra. En el caso en que tenga que resolver la estructura automáticamente, esa será la elección usada, y en el caso en que el proceso sea manual, orientará al usuario sobre cual es el perfil que se considera más apropiado.

Si tenemos un caso general con axil, cortante y flector, normalmente el programa realizará una estimación del perfil por flector, mientras que al usuario se le propone la

oportunidad de hacer él mismo la estimación de los órdenes de magnitud de las tensiones normales debidas al axil y de las tensiones normales debidas al flector.

El tipo de perfil que debemos escoger está relacionado con el esfuerzo interno que, a priori, se suponga que va a crear la mayor tensión (La tensión tangencial debida al cortate es despreciable frente a la tensión normal provocada por el flector). Algunos casos son muy sencillos de determinar. La explicación de cada uno de ellos aparece a continuación

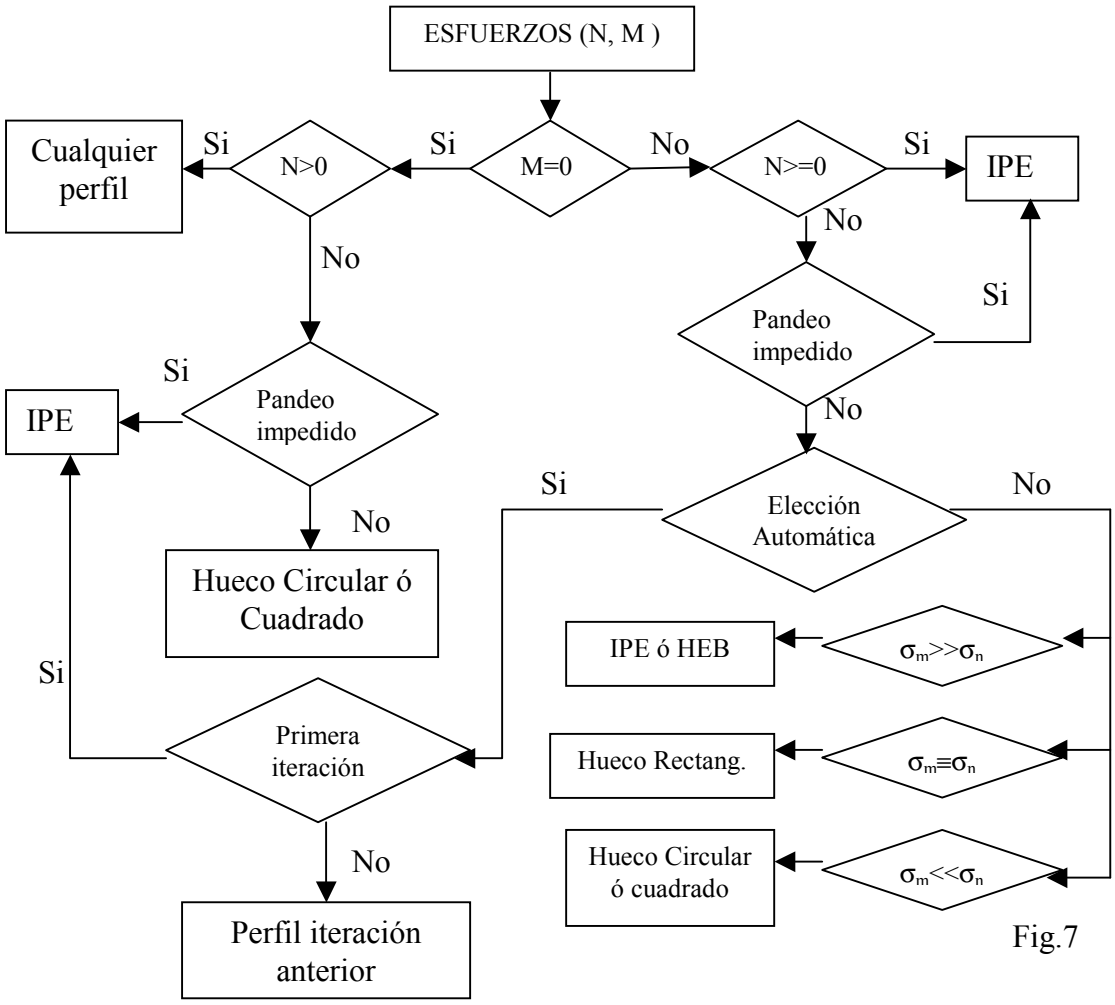


Fig.7

a) Cuando solo tenemos axil a tracción, sabemos que la única característica geométrica que necesitamos es área por lo tanto no hay ningún perfil que sea mejor que otro

b) Cuando tenemos sólo axil a compresión, tenemos que dividir según tengamos el pandeo fuera del plano de la estructura impedido o no. Si estamos en el caso en que el pandeo está impedido, el tipo de perfil que mejor nos vendría sería un perfil IPE, porque las tensiones normales debidas a axil están mayoradas por un factor ω que depende de la esbeltez de la barra λ, que depende inversamente del radio de giro de la sección; por lo tanto, mientras más grande sea el radio de giro más pequeña será la esbeltez de la barra y por lo tanto ω. Como $i = \sqrt{I/A}$, tenemos que mientras más alejada esté la masa del centro de gravedad de la

barra, mejor será el comportamiento de la barra frente al pandeo y por lo tanto mejor trabajará el perfil. Puede verse que cuando el pandeo está impedido podemos disminuir todo lo posible el radio de giro más pequeño de la sección y por lo tanto los perfiles más adecuados son los IPE, que son los que tienen relaciones I/A más altas y por lo tanto radios de giro más altos. Si por el contrario el perfil no tuviera el pandeo fuera del plano impedido, deberíamos buscar un perfil que tuviera un radio de giro grande y muy parecido para todos sus ejes; por lo tanto deberemos ir a los perfiles cerrados, es decir huecos cuadrados ó circulares, siendo éstos los óptimos debido a que la relación I/A es igual en todos sus ejes.

c) Si tenemos la barra solamente sometida a flexión (y cortante), el perfil que mejor se adapta a nuestras necesidades es un IPE, debido que el diagrama de tensiones normales provocadas por el flector es triangular con los máximos en los extremos superior e inferior del perfil. Por lo tanto lo que nos interesa es tener la mayor cantidad de área lejos el centro de gravedad, es decir la mayor inercia posible. Normalmente, las tensiones tangenciales debidas al cortante son mucho más pequeñas y por lo tanto poco relevantes a la hora de la elección del perfil.

d) Si tuviéramos una barra sometida a axil, cortante y flector y admitimos que las tensiones tangenciales generadas por el cortante son mucho más pequeñas que las tensiones normales, tenemos que diferenciar si el axil es a tracción o a compresión:

   1. Si el axil de la sección es de tracción, el perfil más adecuado es un IPE porque el axil a tracción sólo necesita área, por lo tanto puestos a elegir, el perfil más adecuado es el IPE que es el que mejor trabaja a flexión.

   2. Si el axil es de compresión hay que dividir el estudio también en dos: si el pandeo fuera del plano de la estructura está impedido, podemos disminuir todo lo posible uno de los radios de giro y buscar relaciones I/A más altas. Para estos casos el mejor perfil es el IPE que tiene relaciones I/A altas como se comentó en compresión y además módulos resistentes altos como se comentó en flexión, por lo tanto este perfil es el más adecuado; si el pandeo fuera del plano no está impedido, tenemos que hacer un análisis de los órdenes de magnitud de las tensiones normales debidas al flector y de las debidas al axil, σ_m y σ_n. Las tensiones normales debidas al flector tienen una distribución triangular a lo largo de la sección por lo tanto los máximos se dan en las caras superior e inferior de las secciones. Esto obliga a intentar poner la mayor cantidad de área en zonas alejadas del centro de masas, o lo que es lo mismo a perfiles con grandes inercias. Por otro lado, las tensiones normales debidas a axil de compresión, como dijimos antes, dependen en gran medida de la uniformidad en la distribución de área alrededor del centro de masas, por lo tanto lo que pretenderemos será intentar usar un perfil que se adapte mejor a las tensiones que sean predominantes. Por este motivo, en el análisis de esfuerzos, cuando nos encontramos con este caso lo que hacemos es elegir un IPE en el caso automático, para hacer

la primera aproximación o el perfil que salió en la anterior iteración. Este proceso se autocorrige en caso de que la elección del perfil no sea la adecuada (Véase 3.5.4), mientras que en el caso manual se le pide al usuario que haga una estimación de los órdenes de magnitud de σ_m y σ_n y que por lo tanto elija el perfil que mejor se adapte a la estimación de tensiones realizada.

3.3.2 Análisis de esfuerzos para emparrillados.

Los esfuerzos en una barra de un emparrillado sabemos que son torsor, cortante y flector (T, V, M). La distribución de tensiones que origina el torsor varía mucho de perfiles abiertos a perfiles cerrados. Con carácter general se puede decir que una sección de un perfil sometido a torsión sufre un giro alrededor del eje de la barra y a un alabeo de sus puntos. El comportamiento de una barra a torsión, depende crucialmente de que tal alabeo esté impedido o no. Por lo tanto podemos distinguir entre torsión libre o uniforme que se produce cuando el momento torsor es constante a lo largo de la barra y no existe ninguna restricción al alabeo de las secciones, y torsión restringida.

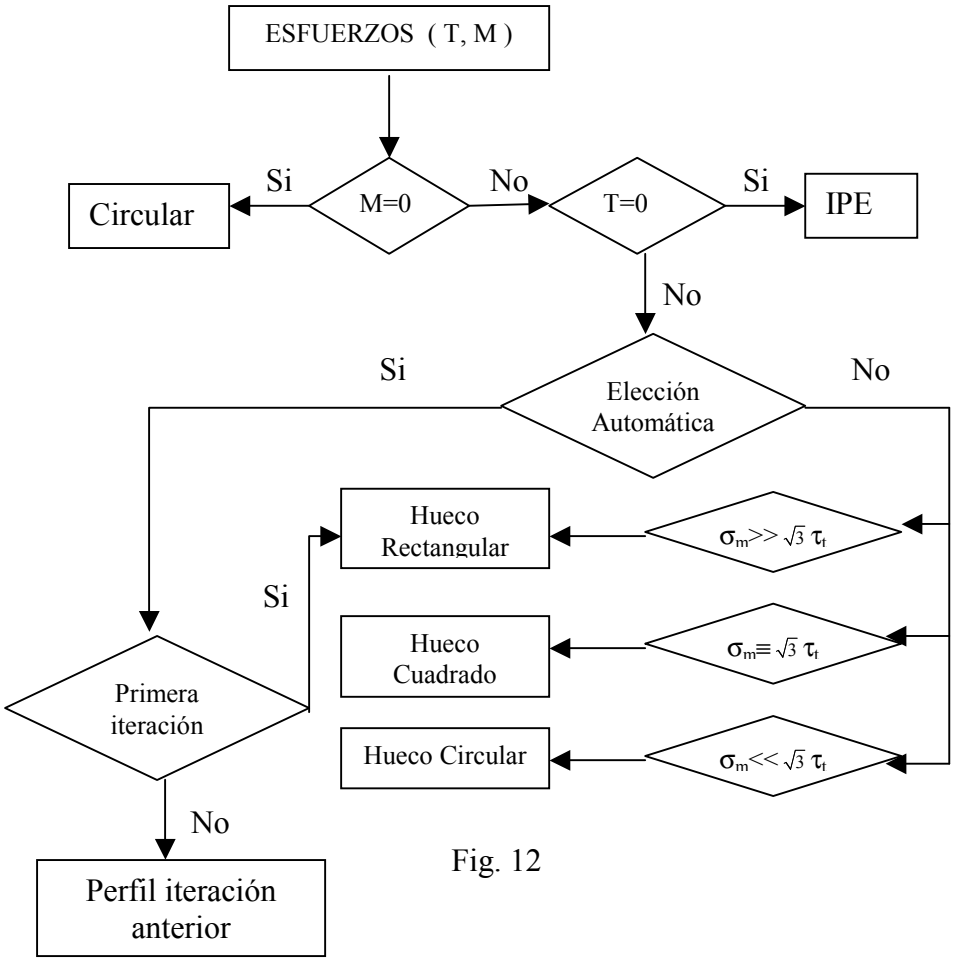


Fig. 12

El alabeo de los perfiles de pared delgada abiertos es mucho mayor que en los cerrados llegando a ser nulo en los huecos circulares y en secciones formadas por

rectángulos que se cortan en un punto, tales como secciones en L, T y X. Por lo tanto en el caso de impedir el alabeo, estos perfiles tenderán a girar y no a alabear y por lo tanto no aparecerán las tensiones normales que aparecerían para evitar que las secciones alabearan.

Resumiendo, tenemos que la aplicación puede usar 5 tipos de perfiles, IPE, HEB, hueco circular, hueco rectangular y hueco cuadrado. Los alumnos solo estudian la torsión libre, y los perfiles IPE y HEB son perfiles susceptibles de alabear. Si como es habitual los nudos no permiten el alabeo (el torsor siempre será constante debido a que no se ha introducido el torsor distribuido a lo largo de la barra) no podremos usar perfiles IPE y HEB, mientras que sí podremos usar los perfiles huecos cuadrados, huecos rectangulares con una relación de lados inferior a 4, y huecos circulares, los cuales como se ha comentado anteriormente presentan poca tendencia al alabeo (pequeña para rectangulares y cuadrados y nula en el caso de los circulares).

La forma de elegir los perfiles de este algoritmo es muy similar a la anterior con la salvedad que ahora las tensiones de mayor importancia suelen ser las tensiones provocadas por el flector y por el torsor siendo normalmente las tensiones provocadas por el cortante de un orden inferior y por lo tanto despreciables durante este proceso de elección de perfiles.

a) Cuando la sección sólo está sometida a torsor, como se ha comentado anteriormente, sólo podemos elegir entre los perfiles cerrados, y el perfil que mejor trabaja a torsión es el perfil hueco circular.

b) Cuando la sección está sometida a flector (y cortante), la sección que mejor trabaja como se explicó anteriormente en pórticos planos es el IPE.

c) En el caso que tengamos tanto torsor como flector (y cortante), tendremos que hacer una estimación de los órdenes de magnitud de las tensiones. Las tensiones debidas al torsor son tangenciales y las debidas al flector normales. También hemos dicho que las tensiones debidas al cortante suelen ser despreciables por lo tanto del criterio de resistencia de Von Mises, podemos sacar que las tensiones que tenemos que comparar son la tensión debida al flector σ_m y la tensión tangencial producida por el torsor, $\sqrt{3}\,\tau_t$. También hemos dicho que cuando tengamos torsor sólo podemos usar perfiles cerrados por lo tanto nos quedamos con que cuando las tensiones debidas al flector sean mayores, el perfil más adecuado será el que tenga mayor módulo resistente y por lo tanto estos serán los huecos rectangulares. Cuando las tensiones debidas al torsor sean las predominantes, tendremos que usar huecos circulares y cuando las tensiones sean del mismo orden los cuadrados, ya que intentaremos buscar una solución de compromiso entre unos y otros.

3.4. Cálculo de esfuerzos y búsqueda de la sección más desfavorable

Como pudimos ver antes, las estructuras, pueden clasificarse en estructuras isostáticas y en estructuras hiperestáticas. Por definición, los esfuerzos en las estructuras isostáticas no dependen de las propiedades geométricas de las barras, simplemente dependen de las cargas externas y por lo tanto para determinar las leyes de esfuerzos en cada barra sólo tendremos que sustituir los valores numéricos de las fuerzas aplicadas y las longitudes de las barras. Esto es de gran importancia porque como podemos ver en la Fig. 10, hay una línea discontinua que va desde la comprobación de los criterios hasta FIN, donde se supone que hemos obtenido la solución a nuestro problema. Ello es debido a que para las estructuras isostáticas, al no depender los esfuerzos en las barras de las propiedades geométricas de las barras, con una sola iteración obtendremos el óptimo (sin tener que hacer una estimación de los valores iniciales para comenzar a iterar).

En las estructuras hiperestáticas, al tener que usar las ecuaciones de compatibilidad-comportamiento para determinar el valor de las incógnitas hiperestáticas y debido a que en estas ecuaciones aparecen las características geométricas de las barras, generalmente tendremos que las incógnitas hiperestáticas dependerán de las características de las barras y por consiguiente los esfuerzos en las barras también. Sin embargo hay un caso intermedio entre las estructuras hiperestáticas y las isostáticas que son las llamadas estructuras pseudohiperestáticas. Son estructuras hiperestáticas pero que tienen la peculiaridad de que al aplicarles las ecuaciones de compatibilidad-comportamiento, desaparecen de la formulación las características geométricas (CCGG) de las barras. Imaginemos por ejemplo, una estructura en la que todas las barras sean iguales (mismas características geométricas), los desplazamientos de los apoyos son nulos y en la que despreciamos la deformación debida al axil. Si le aplicamos el PFV a esta estructura tendremos que:

$$0 = \frac{1}{EI}\sum_{i=1}^{b}\left(\int_{0}^{L_{ki}} M^{v}(x)\cdot M(x)dx\right) \qquad (37)$$

obviamente podemos eliminar la dependencia con las CCGG de las barras de las incógnitas hiperestáticas y de los esfuerzos.

Esta dependencia en las estructuras hiperestáticas es lo que determina que el proceso de diseño sea iterativo, ya que al tener que hacer una estimación de los valores de las CCGG para hacer una estimación de los esfuerzos en las barras, los perfiles obtenidos son los perfiles óptimos para esta estimación. Si esos valores son próximos a los reales, con dos iteraciones habremos terminado. Esta misma dependencia es la que determinará también el fin del cálculo.

Una vez tenemos cuánto valen los esfuerzos en las barras, tendremos que determinar cual es la sección que más nos interesa para el dimensionado de cada grupo de diseño. Normalmente, cada grupo de diseño estará compuesto por una serie de barras y por lo tanto, tendremos que buscar cual de ellas es la que está sometida a los mayores esfuerzos.

Cuando el conjunto de barras está sometido solo a axil, como podría ser el caso de una celosía, o sólo a torsor en el caso de emparrillados, la determinación de la sección o secciones más solicitada es trivial. Lo mismo ocurre cuando la barra está sometida a flexión, si embargo cuando las barras están sometidas a axil ó torsor, cortante y flector, el asunto se complica y hay que buscar un criterio que nos sirva para determinar cual es la sección más desfavorable. Normalmente el criterio más usado es hacer la elección por flector debido a que las tensiones debidas al flector son mayores que las debidas al cortante.

3.5. Redimensionado por resistencia

El siguiente paso después de la elección de la tabla de perfiles adecuada, consiste en la búsqueda del perfil que mejor se adapte a nuestras necesidades; para ello se han usado una serie de algoritmos.

La idea básica para el redimensionado de las barras por resistencia y rigidez está basada en la dependencia débil que presentan los valores los esfuerzos internos respecto a las características geométricas de las barras en estructuras hiperestáticas. Esto se traducirá en fijar los esfuerzos internos en los valores conocidos del punto actual de diseño, dejando libres como variables a calcular las características geométricas.

Podemos extraer de la Fig. 11 y 12 que existe una casuística no solo relacionada con la elección de perfiles, sino además asociada a la forma de encontrar los perfiles dentro de las tablas. Los diferentes casos que nos podemos encontrar son:

- Axil a tracción
- Axil a compresión
- Flexión
- Axil, flector (y cortante)
- Torsión.
- Torsor, flector (y cortante)

3.5.1. Axil a tracción.

El caso en que tengamos una barra sometida solo a axil de tracción es el más sencillo que podemos encontrarnos, simplemente hay que determinar el área necesaria para cumplir el criterio de resistencia que sale de despejar

$$\sigma_E \leq N / A \Rightarrow A_{nec} \geq N / \sigma_E \qquad (38)$$

Obviamente la búsqueda en las tablas de perfiles es un proceso trivial, simplemente hay que mirar cual es el área del perfil que más se acerca, siempre por encima, al área mínima buscada.

3.5.2. Flexión

El caso en el que la barra esté sometida a flector (y cortante, en general) es un caso sencillo de resolver también.

En la Fig. 13, podemos ver cada uno de los pasos a seguir.

1. Primero hacemos una estimación del módulo resistente de la sección (despreciamos las tensiones tangenciales debidas al cortante), $\sigma_E \leq M / W_Z \Rightarrow W_{nec} \geq M / \sigma_E$.

2. Una vez hemos hecho la estimación del perfil, tendremos que buscar entre los perfiles que cumplan W_{nec} que tenga menor área. Esto puede resultar innecesario cuando trabajamos con IPE, en el sentido en que estos perfiles son de áreas crecientes; sin embargo esta tendencia no es tan clara en los perfiles cerrados, sobre todo en los perfiles rectangulares huecos.

3. Una vez hemos encontrado el perfil adecuado, tendremos que hacer una comprobación completa, incluyendo las tensiones tangenciales debidas al cortante. Si el perfil cumple, ya hemos llegado al óptimo, sin embargo, si no cumple tendremos que buscar entre los siguientes perfiles de mayor área, aquel que teniendo menor área tenga mayor W_z.

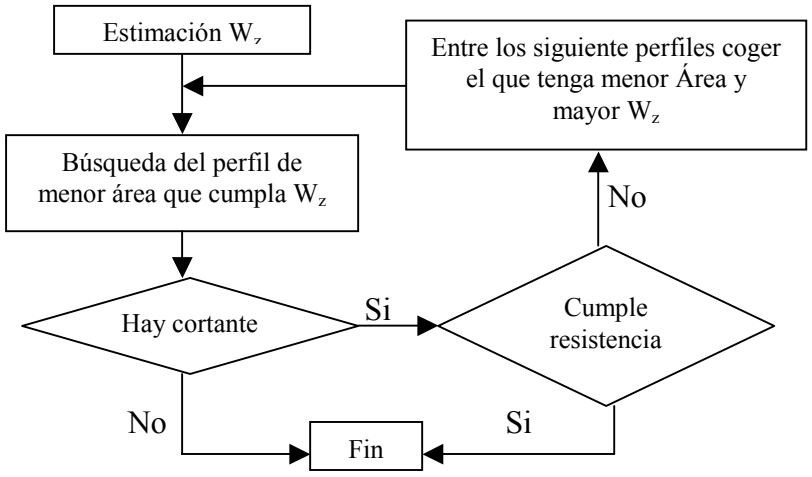


Fig. 13

Este algoritmo se utilizará en la estimación de perfiles para otros casos más generales y permite al usuario probar con los diferentes tipos de perfiles y comprobar cual de ellos es el más adecuado.

3.5.3. Axil a compresión

Este caso es más interesante y complicado que el anterior, sobre todo en perfiles cerrados. En Fig. 14 aparece el proceso de iteración usado para el caso de axil a

compresión en perfiles cerrados. Cómo puede verse, el proceso iterativo es más complicado.

Si echamos un vistazo a las tablas de perfiles cerrados, podemos ver claramente que éstos están asociados en series de 3 o 4 perfiles. Estas series tienen las siguientes características:

1. A medida que vamos aumentando el espesor, obviamente va aumentando el área de la sección
2. De la misma forma que va aumentado el área, aumentan el momento estático, la inercia y el módulo resistente de la sección.
3. El radio de giro de la sección va disminuyendo en el sentido creciente de las áreas, siendo el primer perfil de la serie el perfil de mayor radio de giro y el último el de menor radio de giro.

Este es el comportamiento de las características geométricas de las barras para perfiles de la misma serie, sin embargo, si cambiamos de serie, podemos ver que aunque la regla general indica que todas las características geométricas van aumentando, existen saltos de éstas. Por ejemplo, podemos comprobar que el área del ultimo perfil de una serie es mayor que la del primer perfil de la serie siguiente. Esto nos hace pensar que puede haber perfiles que se comporten mejor en unos casos que en otros y por lo tanto estaremos usando el acero de una forma más óptima.

El proceso iterativo lo podemos ver en la Fig. 14, como una primera cota, tenemos la esbeltez máxima de las barras; con esta esbeltez máxima, tendremos una cota mínima del radio de giro del perfil. Una vez explicada la evolución de las características geométricas de los perfiles cerrados, sólo tendremos que mirar los primeros perfiles de cada familia, ya que si el primer perfil no cumple la cota impuesta, no lo cumplirá ninguno dentro de esa serie.

Una vez hemos encontrado la primera serie que cumple, tendremos que comenzar a realizar las comprobaciones del criterio de resistencia. Sabemos que el criterio de resistencia es:

$$\sigma_E \leq \omega N / A \Rightarrow A_{nec} \geq \omega N / \sigma_E \qquad (39)$$

El problema radica en que ω depende de la esbeltez de la barra (λ), y esta depende del radio de giro de la sección por lo tanto el problema se complica, porque como anteriormente dijimos, el área del último perfil de una serie es mayor que el área del primer perfil de la serie siguiente, pero el radio de giro del último perfil de la serie es menor que el del primer perfil de la serie siguiente; por lo tanto aunque la sección del ultimo perfil de la serie sea mayor, debido a que el radio de giro es menor y por lo tanto su esbeltez mayor, ω será mayor, aumentado por tanto la necesidad de área como se desprende de la ecuación (39), mientras que el primer perfil de la serie al tener un radio de giro mayor y consecuentemente una esbeltez y ω menores, disminuye su necesidad de acero. Por este motivo es complicado decir cual es el óptimo, y lo que en un principio no puede hacerse debido al carácter docente de la aplicación, aunque computacionalmente no tenga ningún problema, es hacer la comprobación de todos los

perfiles. Ese es el motivo por el cual se usa un algoritmo de búsqueda de perfiles óptimos. Una vez realizada esta aclaración el proceso es:

1. Calculamos el A_{nec} con el último perfil de la serie que es más esbelto. Si el área de ese perfil es mayor que el área mínima ($A \geq A_{nec}$), ya tenemos una solución que satisface el criterio de resistencia.
2. Los perfiles con menor espesor de la serie tienen menor área que éste perfil y por lo tanto si alguno de ellos cumpliera que $A \geq A_{min}$, tendríamos que este perfil es mejor que el anterior y se convertiría una solución posible ya que al ir diminuyendo el espesor de los perfiles, aumenta su radio de giro y por lo tanto disminuiría la necesidad de área para ese perfil, con lo cual estamos del lado de la seguridad.
3. Una vez tenemos una solución válida, tendremos que buscar si en las siguientes series hay perfiles que tengan menor área que el que nosotros hemos elegido, en tal caso se vuelve a repetir el proceso hasta que lleguemos a un punto en el que no encontremos perfiles que tengan menor área.

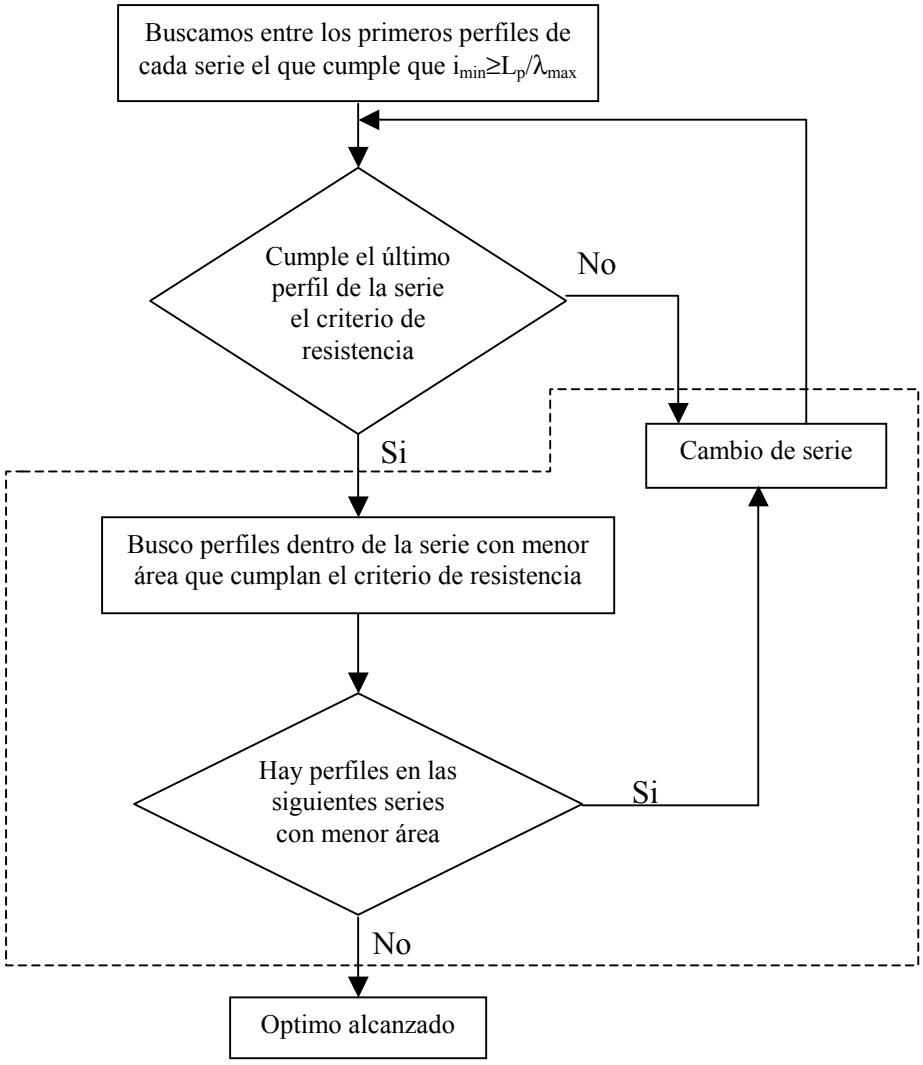


Fig. 14

Hay que hacer notar que el proceso tal y como está planteado y tal como es usado por los alumnos, no es un proceso que nos garantice siempre el perfil óptimo, aunque se acerca bastante; el motivo es que en el segundo punto imponemos, en la búsqueda de perfiles dentro de la serie, que el resto de perfiles que queremos comprobar cumplan el criterio anteriormente mencionado. Como anteriormente se indicó, este criterio es conservador y por lo tanto puede darse el caso en el que al ir aumentado el radio de giro, va disminuyendo la cantidad de material necesario, y a medida que va aumentando el radio de giro, también va diminuyendo el área de la sección por lo tanto dependerá de la pendiente de estas dos evoluciones negativas la que determine si en algún momento ese perfil cumple el criterio de resistencia. Este caso se ha contemplado en la aplicación y se ha introducido una mejora que permite al usuario comprobar cuando este algoritmo falla.

Para perfiles abiertos la única diferencia radica en que no existe esta variación de propiedades; en los perfiles abiertos, a medida que aumenta el área, aumentan el resto de propiedades geométricas de la sección y por lo tanto, toda la parte que se encuentra en el interior del recuadro de rayas discontinuas en la Fig. 14 no existe. Si el perfil no cumple, se comprueba el siguiente, hasta que uno cumpla.

3.5.4. Axil, flector (y cortante).

Digamos en primer lugar que la existencia del cortante no es demasiado relevante, debido a que las tensiones que éste produce suelen ser en la práctica bastante menores que las provocadas por el axil y el flector.

El algoritmo de búsqueda de perfiles óptimos a resistencia para este caso es mucho más complicado que en el resto de los casos debido a que las tensiones normales están provocadas por axiles y flectores como se desprende de $\sigma = \omega N / A + M / W$, donde el primer término corresponde a las tensiones normales provocadas por el axil (ω=1 para el caso de axil a tracción) y el segundo término corresponde a las tensiones provocadas por el flector.

Supongamos que tenemos una barra sometida a axil de tracción y flexión. Sabemos por lo discutido anteriormente que la sección que mejor se adapta por su geometría a esta solicitación es el IPE. Si tenemos dicha barra sometida a compresión pura y el pandeo fuera del plano no está impedido, el perfil más adecuado es un perfil hueco circular ó cuadrado. Si tenemos una barra sometida a axil de compresión y a flector sin tener el pandeo fuera del plano impedido y llamamos σ_n a la tensión normal provocada por el axil y σ_m la provocada por el flector tenemos que:

a) Si $\sigma_m >> \sigma_n$ deberemos coger perfiles que trabajen bien a flector es decir, perfiles IPE.

b) Si $\sigma_m << \sigma_n$ podemos diferenciar entre dos casos, si el pandeo fuera del plano está impedido, el perfil más adecuado es un perfil IPE, mientras que si el pandeo no está impedido, deberemos coger perfiles que sean apropiados para el trabajo a compresión, es decir los comentados anteriormente.

c) Si $\sigma_m \equiv \sigma_n$ (mismo orden de magnitud) también podemos distinguir entre dos casos: si el pandeo fuera del plano está impedido, tenemos que el perfil más adecuado es un perfil IPE, mientras que si el pandeo no está impedido, tendremos que buscar una solución de compromiso entre grandes inercias que permitan el aprovechamiento de la sección a flexión con uniformidad de masas alrededor del centro de masas (perfiles cerrados), por lo tanto el perfil más adecuado es el perfil hueco rectangular.

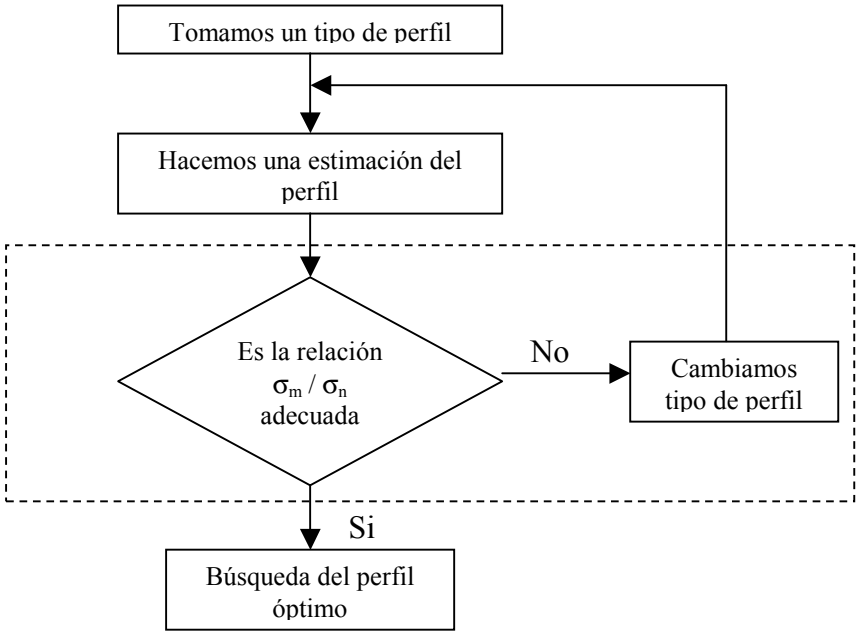


Fig. 15

Por otro lado, tenemos que echando un simple vistazo a los esfuerzos no podemos determinar qué tensión será predominante, por lo tanto tendremos en un principio que tomar un perfil y comparar los órdenes de magnitud de las tensiones provocadas y de esa manera, hacer una corrección si fuera necesaria del tipo de perfil elegido, es decir, en este proceso iterativo tendremos que incluir una redefinición del tipo de perfil elegido.

Como podemos ver en la Fig. 15, hay una zona recuadrada que corresponde al proceso de redefinición de perfiles. Esta parte no será necesario realizarla en el caso que tengamos que el axil sea de tracción o el pandeo fuera del plano esté impedido. El motivo por el cual no se realiza el proceso de redefinición para estos casos es que cuando el axil es de tracción, la única característica geométrica que podemos controlar para mejorar el proceso iterativo es el módulo resistente, y si el pandeo está impedido, las mejores secciones son las secciones en I, por lo tanto si el proceso es automático, esté habrá sido el perfil elegido como podemos ver en la Fig. 15, y si el proceso es manual el usuario habrá elegido el perfil que él cree que es más conveniente y por lo tanto si ha renunciado a la opción propuesta por el programa, sería redundante volverle a repetir esa misma propuesta.

El proceso mostrado en la Fig. 15 es el proceso teórico el cual tiene una serie de puntos que hay que matizar.

1. El perfil que normalmente se usa para la primera iteración dependerá de la estimación que realice el usuario; si supone que las tensiones debidas al flector son predominantes, el usuario deberá tomar un IPE o un HEB y hará la estimación usando el módulo resistente; si el usuario estima que la tensión normal del flector es del mismo orden de magnitud que la tensión normal debida al axil, el perfil que deberá elegir es un perfil hueco rectangular y se realizará una estimación del perfil por módulo resistente; por ultimo si el usuario estima que la tensión debida al axil es mucho mayor que la tensión debida al flector, el perfil que deberá elegir será un pefil hueco circular o hueco cuadrado y realizará una estimación por área.

2. La teoría nos dice que cuando $\sigma_m >> \sigma_n$ el perfil más adecuado es un IPE, y para el caso en que $\sigma_m << \sigma_n$ son mejores los huecos circulares ó cuadrados, quedando en medio los perfiles en cajón, sin embargo en la práctica, este criterio es poco útil debido a que no sabemos exactamente qué quiere decir "mucho mayor que", o "mucho menor que", o "del orden de", así que la solución propuesta es imponer unas cotas reales que nos permitan determinar en que rango de razón de tensiones estamos. Unos valores bastante acertados son $\sigma_m / \sigma_n < 0.5$ para perfiles hueco circulares, $\sigma_m / \sigma_n > 2$ para perfiles IPE, y la zona intermedia para los perfiles en cajón.

3. La búsqueda de un perfil óptimo es un proceso iterativo en el que se busca el siguiente perfil de menor área que cumpla el criterio de módulo resistente y radio de giro anteriormente impuestos (el de área va implícito) e ir realizando la comprobación correspondiente.

3.5.5. Torsión.

El proceso iterativo que aquí se sigue es un proceso iterativo muy parecido al de axil a compresión. El criterio de resistencia no dice que:

$$\sqrt{3\tau_t^2} \leq \sigma_E \tag{40}$$

para perfiles cerrados, tenemos que:

$$\tau_t = \frac{T}{2 \cdot \Omega \cdot e} \tag{41}$$

donde T es el momento torsor, e es el espesor del perfil y Ω es el área encerrada por la línea media.

A la vista de las tablas de perfiles cerrados, vemos que disponemos en cada serie de diferentes espesores, por lo tanto puede darse el caso que diferentes productos de Ωe nos permitan cumplir el criterio de resistencia por lo tanto habrá que realizar un proceso iterativo, que se muestra en la Fig. 16.

Si operamos con las ecuaciones (40) y (41), tenemos que:

$$\frac{T}{2\cdot\Omega\cdot e}\le\frac{\sigma_E}{\sqrt{3}}\Rightarrow e\cdot\Omega\ge\frac{\sqrt{3}T}{2\sigma_E} \tag{42}$$

Si aplicamos esto al hueco circular que es el mejor perfil para torsión tenemos que:

$$\Omega=\frac{\pi(D-e)^2}{4}\quad \text{si } D>>e\ \Rightarrow\quad \Omega=\frac{\pi D^2}{4}$$

de esta manera, nos queda que:

$$eD^2\ge\frac{2\sqrt{3}T}{\pi\sigma_E} \tag{43}$$

Con esta expresión podemos hacer la primera estimación del perfil que nos hace falta. Si miramos las tablas de perfiles, vemos que el espesor más grande son 8 mm, por lo tanto si sustituimos ese espesor en la ecuación (43) podemos tener una cota mínima del diámetro necesario, es decir de la serie mínima.

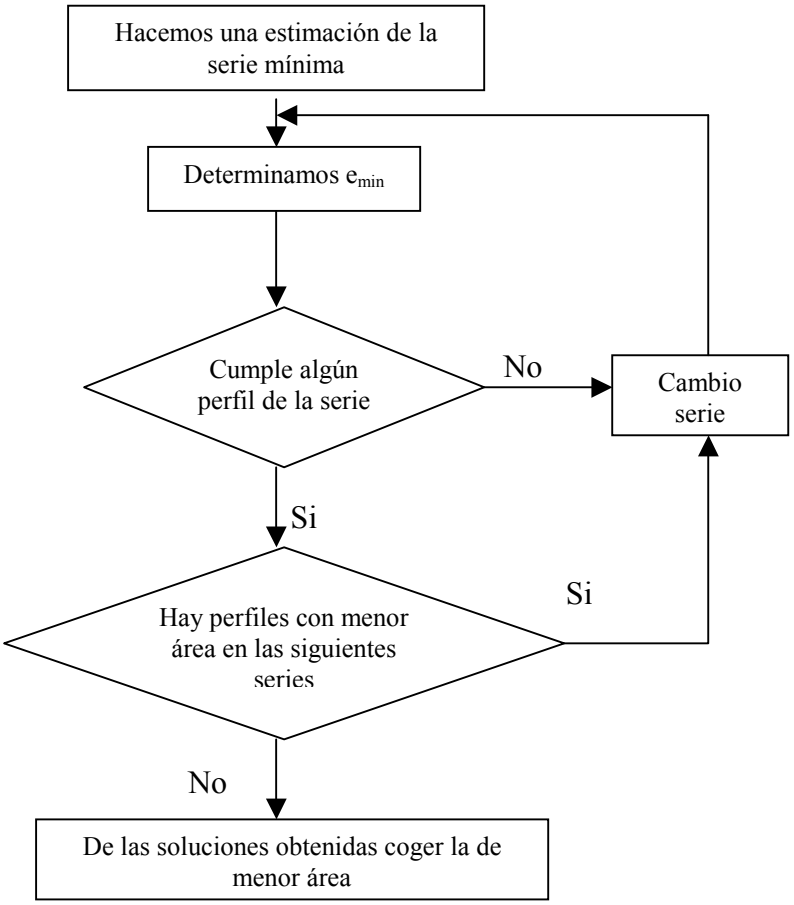


Fig. 16

El resto del proceso iterativo se pueden ver en la Fig. 16:

1. Cogemos el diámetro de la serie que tengamos y calculamos con la ecuación (43) el espesor mínimo que nos hace falta para ese diámetro y buscamos si alguno de los perfiles de la serie cumple. Si no cumple tendremos que cambiar de serie y volver a realizar este paso.

2. Si cumple, tenemos que comprobar si hay algún perfil en las siguientes series que tenga menor área que el que hemos conseguido. En tal caso, tendremos que realizar el paso 1 en dicha serie.

3. Una vez que no encontremos más perfiles que tengan menor área, cogemos el perfil que menor área tenga de todas las soluciones obtenidas.

Si el perfil que usamos no es un perfil hueco circular que sería el adecuado, el proceso se haría igual que hemos hecho aquí simplemente tendríamos que variar la fórmula del área encerrada por la línea media del perfil.

3.5.6. Torsor, flector (y cortante).

Este caso es parecido al que anteriormente tratamos donde teníamos axil, y flector, sólo que ahora tenemos torsor y flector, volviendo a ser poco importante la presencia de cortante.

Del criterio de resistencia de Von Mises, tenemos que

$$\sqrt{\sigma^2 + 3(\tau_c + \tau_t)^2} \leq \sigma_E \tag{44}$$

si despreciamos las tensiones debidas al cortante, la ecuación queda como

$$\sqrt{\sigma^2 + 3\tau_t^{\,2}} \leq \sigma_E \tag{45}$$

donde:

$$\sigma = \frac{M}{W_z} \text{ y } \tau_t = \frac{T}{2 \cdot \Omega \cdot e} \tag{46}$$

Como se desprende de las ecuaciones del problema, estamos en la misma situación que antes cuando teníamos axil y flector, no podemos determinar a priori qué tipo de perfil es más adecuado, así que haremos una estimación de las tensiones para determinar una cota del perfil que nos haga falta. En la Fig. 17 tenemos un esquema del proceso iterativo, que como vemos es idéntico al de antes.

Los criterios de elección de perfiles ya se han comentado anteriormente y aparecen a continuación de forma resumida.

a) Si $\sigma_m >> \sqrt{3}\,\tau_t$ deberemos coger perfiles cerrados que trabajen bien a flector es decir, perfiles huecos rectangulares.

b) Si $\sigma_m << \sqrt{3}\,\tau_t$ deberemos coger perfiles que sean apropiados para el trabajo a torsión, es decir los huecos circulares.

c) Si $\sigma_m \equiv \sqrt{3}\,\tau_t$ (mismo orden de magnitud) tendremos que buscar una solución de compromiso entre grandes inercias que permitan el aprovechamiento de la sección a flexión con productos de eΩ adecuados. Los perfiles cuadrados son ligeramente mejores que los huecos circulares a flexión porque dejan la mitad de su área aproximadamente trabajando de forma bastante adecuada a flexión, mientras que debido a su geometría no se penaliza demasiado el producto eΩ frente al área de la sección; por eso son adecuados en este caso.

Considerando sólo los esfuerzos no podemos determinar qué tensión será predominante, por lo tanto tendremos en un principio que elegir un perfil (ó hacer una estimación de los órdenes de magnitud de las tensiones) y comparar los órdenes de magnitud de las tensiones provocadas, y de esa manera hacer una corrección, si fuera necesaria del tipo de perfil elegido; es decir, en este proceso iterativo tendremos que incluir una redefinición del tipo de perfil elegido. En la Fig. 17 podemos ver un diagrama del proceso general de cálculo:

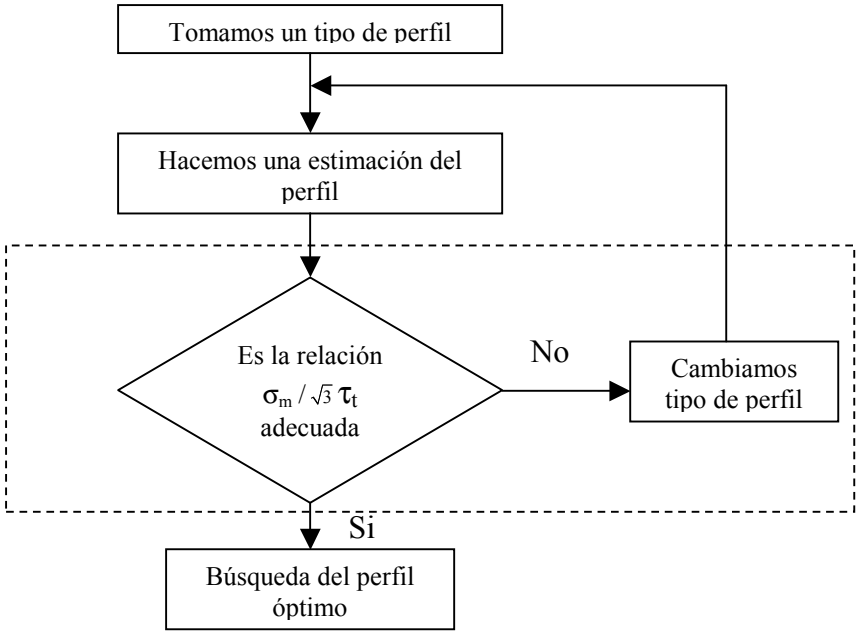


Fig.17

El diagrama muestra el proceso teórico de búsqueda de perfiles, sin embargo hay algunos puntos importantes que hay que aclarar y que en la práctica son los usados por la aplicación.

1. El perfil que usa en la primera iteración dependerá de la estimación que realice el usuario: si supone que las tensiones debidas al flector son predominantes, el usuario deberá tomar un hueco rectangular y hará la estimación usando el módulo resistente; si el usuario estima que la tensión normal del flector es del mismo orden de magnitud que la tensión tangencial debida al torsor, el perfil

que deberá elegir es un perfil hueco cuadrado y se realizará una estimación del perfil por módulo resistente; por ultimo, si el usuario estima que la tensión debida al torsor es mucho mayor que la tensión debida al flector, el perfil que deberá elegir será un perfil hueco circular y realizará una estimación por eΩ como se explicó en el proceso de iteración para torsión pura. En el caso automático como puede verse en la Fig. 12, es un hueco rectangular.

2. La teoría nos dice que cuando $\sigma_m >> \sqrt{3}\,\tau_t$ el perfil más adecuado es un hueco rectangular, y para el caso en que $\sigma_m << \sqrt{3}\,\tau_t$ son mejores los huecos circulares, quedando en medio los perfiles hueco cuadrados; sin embargo en la práctica, este criterio es poco útil debido a que no sabemos exactamente qué quiere decir "mucho mayor que", o "mucho menor que", o "del orden de", así que la solución propuesta es imponer unas cotas reales que nos permitan determinar en que rango de la razón de tensiones estamos. Unos valores bastante acertados son $\sigma_m/\sqrt{3}\,\tau_t$ <0.5 para perfiles huecos circulares, $\sigma_m/\sqrt{3}\,\tau_t$ >2 para perfiles en cajón, y la zona intermedia para los perfiles cuadrados.

3. Una vez hecha una estimación del perfil, buscamos el perfil óptimo. Existen dos formas diferentes de iterar, dependiendo del perfil que estemos usando.

   - Los perfiles circulares y en cajón están asociados a los casos extremos, es decir a tensiones debidas al torsor mucho mayores que las debidas al flector y tensiones debidas al flector mucho mayores que las debidas al torsor respectivamente. En estos casos, las estimaciones realizadas se suponen que son bastante buenas; por lo tanto, el proceso iterativo a seguir para conseguir el óptimo en el caso de que la estimación no cumpla el criterio general de resistencia es buscar el siguiente perfil de mayor área y hacer la comprobación general. Este proceso se continuará de forma cíclica hasta encontrar un perfil que cumpla.
     Cabe suponer que como al realizar la estimación hemos usado la tensión predominante, el número de iteraciones que tendremos que realizar será pequeño.

   - En el caso de los perfiles cuadrados, nos encontramos con que estamos en un caso intermedio; por lo tanto si hacemos una estimación por alguno de los extremos, posiblemente tengamos que iterar muchas más veces de la manera anteriormente comentada para llegar al óptimo. Para este tipo de perfiles, se ha implementado una metodología diferente de iteración:

     Despreciando la tensión tangencial producida por el cortante del criterio de resistencia tenemos la ecuación (45). Si definimos un factor $\alpha=\tau_t/\sigma_m$, la ecuación (45) queda de la forma:

$$\frac{M}{W_z}\sqrt{1+3\alpha^2} \le \sigma_E \qquad (46)$$

     Los perfiles cuadrados son autosimilares, y esta relación α permanece razonablemente constante (aprox. 0.6 para valores similares de torsor y flector); por lo tanto el proceso iterativo para cuadrados es muy sencillo:

1. Hacemos una primera estimación por módulo resistente como se dijo anteriormente.

2. Si el perfil no cumple calculamos α y con la ecuación (46) hallamos una cota de W_z, Con esta cota podemos determinar el perfil más adecuado. Volvemos a realizar la comprobación y si no cumple, buscamos el perfil como hemos comentado anteriormente con huecos rectangulares y circulares.

3.6. Comprobación del criterio de rigidez.

Para el caso en el que impongamos alguna restricción a los desplazamientos, tendremos que hacer la comprobación de dicho criterio. Hacer la comprobación es muy sencillo, ya que una vez hemos obtenido el valor de los desplazamientos que se han calculado con los métodos anteriormente descritos, lo único que tenemos que hacer es sustituir el valor de las características geométricas de las barras que hemos obtenido anteriormente.

3.7. Redimensionado por rigidez.

El proceso de rediseño por rigidez se produce cuando la solución obtenida del diseño por resistencia incumple la restricción de rigidez. Básicamente lo que tenemos que hacer el buscar un perfil o grupo de perfiles que nos permitan satisfacer esa restricción.

El problema al que nos enfrentamos no es un problema fácil de resolver como se verá posteriormente, ya que es un problema de programación no lineal, y no es la intención de este trabajo el desarrollar un algoritmo de optimización, simplemente se han desarrollado unos algoritmos que en casos sencillos sí nos permitirán encontrar el óptimo pero que en casos complicados simplemente se limitarán a buscar una solución válida.

El problema se ha planteado para pórticos planos y la explicación de cada uno de los puntos se ha realizado con este tipo de estructura siendo el proceso el mismo para emparrillados a excepción de las ecuaciones de compatibilidad-comportamiento.

3.7.1 Pórticos Planos.

El cálculo de los desplazamientos como ya se comentó (Apartado 2.3.4) se ha realizado matricialmente, auque también se podría haber utilizado el PFV habiéndose obtenido el mismo resultado pero con un mayor coste computacional.

Sea una estructura a la que queremos calcular un desplazamiento. Si usamos el PFV, para un caso en el que no tengamos ni cargas térmicas ni desplazamientos en los apoyos, la expresión quedaría como:

$$\delta = \sum_{i=1}^{b} \left(\int_0^{L_i} N^v(x) \cdot \frac{N(x)}{EA_i} dx + \int_0^{L_i} M^v(x) \cdot \frac{M(x)}{EI_i} dx \right) \quad (47)$$

Las leyes de esfuerzos reales dependen de las incógnitas hiperestáticas, mientras que las leyes virtuales $N^v(x)$ y $M^v(x)$ deber estar en equilibrio con la acción unidad asociada energéticamente al desplazamiento δ. Para calcular los esfuerzos virtuales se convierte la estructura en isostática liberando las ligaduras asociadas a las incógnitas hiperestáticas, de esta manera, tendremos una expresión que quedaría de la siguiente forma:

$$\delta = \sum_{i=1}^{b} \left(\frac{f_{1i}(H)}{EA_i} + \frac{f_{2i}(H)}{EI_i} \right) \quad (48)$$

donde tenemos un sumatorio extendido a todas las barras de la estructura y donde los numeradores de las fracciones son funciones del conjunto de incógnitas hiperestáticas $H=(H_1, H_2, ..., H_j)$. La forma final del criterio a resolver será:

$$\delta = \sum_{i=1}^{b} \left(\frac{f_{1i}(H)}{EA_i} + \frac{f_{2i}(H)}{EI_i} \right) \leq \delta_{max} \quad (49)$$

Una vez llegados a este punto tenemos dos opciones:

1. Sustituir el valor real de las hiperestáticas, es decir las hiperestáticas en función de las cargas externas y de las CCGG de las barras, con lo cual tendríamos una inecuación de difícil resolución.

2. Usar la idea básica del redimensionamiento, suponiendo que el valor de las hiperestáticas permanecerá razonablemente constante cuando varíen las propiedades geométricas de las barras; podemos obtener así una inecuación formada por un sumatorio de fracciones con valores numéricos en el numerador y A_i ó I_i en el denominador.

La solución escogida obviamente es la segunda. Esta solución en sí misma tampoco es única, porque si hemos ido siguiendo el desarrollo del algoritmo, podemos ver que hemos realizado un dimensionado por resistencia con unos valores de las hiperestáticas. Una vez hemos hecho el dimensionado por resistencia y por lo tanto conseguido unos perfiles que no tienen porqué coincidir con los originales, el valor de las hiperestáticas habrá variado, por lo tanto tenemos dos opciones, o elegir el valor antiguo o elegir el nuevo valor que obtendríamos de sustituir los nuevos valores de las CCGG de las barras.

El proceso iterativo en la etapa de dimensionado de perfiles se basa en la hipótesis de que el valor de las hiperestáticas variará poco con las CCGG de las barras, y en este caso haremos la misma suposición. No hay ninguna prueba que indique que sea mejor hacerlo de una de las maneras, siendo lo habitual que los alumnos sigan usando los valores antiguos de las hiperestáticas. El programa incluye una opción que permite al

usuario elegir, qué desea hacer, usar el nuevo valor de las hiperestáticas o usar el valor antiguo. El motivo de esta decisión es que algunas veces una mala condición inicial puede degenerar en valores muy malos (lejos de los reales) de los valores de las incógnitas hiperestáticas influyendo negativamente en las posibilidades de cumplimiento del criterio de rigidez.

Como se desprende de las ecuaciones anteriores, el problema aún así planteado no es un problema sencillo de resolver porque es un problema de programación no lineal discreta. Para solucionar este problema se ha desarrollado un pequeño algoritmo que nos permitirá hacer un redimensionado por rigidez para la cantidad de incógnitas usadas en los casos prácticos y para la cantidad de restricciones de rigidez normalmente impuestas (1 ó 2).

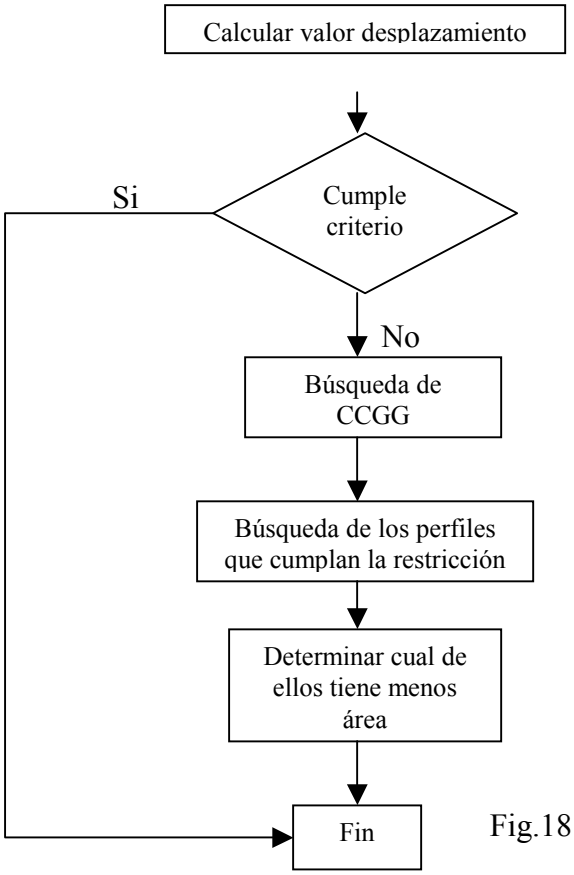


Fig.18

Este algoritmo no garantiza que se alcance el punto óptimo; sin embargo, la solución que da normalmente son muy aceptables, y de hecho garantiza dentro de los límites teóricos anteriormente impuestos el óptimo para el caso en el que solo tengamos una restricción y una característica geométrica involucrada directamente en el desplazamiento, que es el caso más habitual en los problemas planteados en Resistencia de Materiales.

La idea que se persigue en este algoritmo está en la búsqueda de las CCGG de las barras que están en el denominador de la ecuación (49), rescrita aquí debajo de forma más compacta:

$$\delta = \sum_{i=1}^{nvd} \left(\frac{K1_i}{A_i} + \frac{K2_i}{I_i} \right) \leq \delta_{max} \qquad (50)$$

donde nvd es el número de barras diferentes que tenemos (número de variables de diseño) y K1, K2 son las constantes asociadas a las características de las barras en cuestión, K1 para las áreas y K2 para las inercias.

Para un caso típico en el que tengamos por ejemplo 2 variables de diseño y en una de ellas podamos despreciar la deformación debida al axil, tendremos como mucho 3 términos (podemos tener menos de 3 términos). Las características obtenidas en el proceso de resistencia son necesarias, por lo tanto no pueden disminuir. Además, tenemos que al incumplir el criterio de rigidez, tendremos que aumentar dichas características para disminuir el desplazamiento, ya que EA es la rigidez a tracción y EI es la rigidez a flexión por lo tanto tendremos que aumentar el área o la inercia (no se tendrán en cuenta los factores negativos), y el procedimiento seguido es dejar el resto de propiedades constantes y variar una de ellas, o mejor dicho determinar cuanto tendría que valer esa propiedad para que cumpliera el criterio de rigidez. Esto lo hacemos con cada una de las CCGG que tengamos en la ecuación, y con los valores obtenidos, buscamos el perfil que nos permita cumplir la restricción. Una vez hemos encontrado todos los perfiles, comparamos sus áreas de manera que el perfil que menor área tenga será el perfil más adecuado.

Es fácil ver que este algoritmo tiene algunos problemas, como son:

- En el caso en que tengamos en la misma ecuación el área y la inercia del mismo grupo de barras, cuando hacemos la variación por separado de cada una de las características, el proceso es completamente irreal; si realmente fuéramos aumentando el tamaño de los perfiles, tendríamos que ambas propiedades van aumentando y por lo tanto se estaría teniendo en cuenta su influencia (la de las dos características geométricas) al mismo tiempo.

- Este algoritmo tampoco garantiza las soluciones intermedias entre barras, es decir si tenemos en la ecuación características geométricas de diferentes grupos de barras, no sabemos si una solución intermedia entre las soluciones de perfiles obtenidas nos daría el cumplimiento de la restricción usando perfiles más pequeños.

- Todos estos problemas antes comentados se agudizan cuando tenemos varias restricciones de rigidez.

La ventaja que presenta este algoritmo es que el proceso manual que se usa para imponer el criterio de rigidez es muy parecido, y por lo tanto, las dificultades son igualmente complicadas. Si atendemos al carácter docente de la aplicación, se entiende que la cantidad de variables de diseño usadas y la cantidad de CCGG que aparezcan en el problema será pequeña, así que el algoritmo de búsqueda de CCGG se puede considerar satisfactorio, ya que para el caso en que tengamos una sola CG en la ecuación

es capaz de darnos una cota real del perfil que necesitamos buscar.

Este proceso, como se ha comentado es adecuado para casos muy sencillos que serán los más habituales; sin embargo, no garantiza una solución. Esta afirmación puede entenderse fácilmente si miramos la ecuación (50) donde tenemos un sumatorio de fracciones. En el método propuesto anteriormente, se busca una cota del perfil necesario suponiendo que la aportación a la expresión del resto de barras permanece constante, en el caso en que esta aportación sea mayor que la cota impuesta, el término sale negativo, con lo que no se puede encontrar ningún perfil que cumpla dicha condición. Este caso que se puede dar cuando las restricciones impuestas al desplazamiento son muy restrictivas para las cargas dadas. Para solucionar este problema se ha implantado un algoritmo que intenta dar una solución a este problema. El algoritmo se explica en la Fig. 19

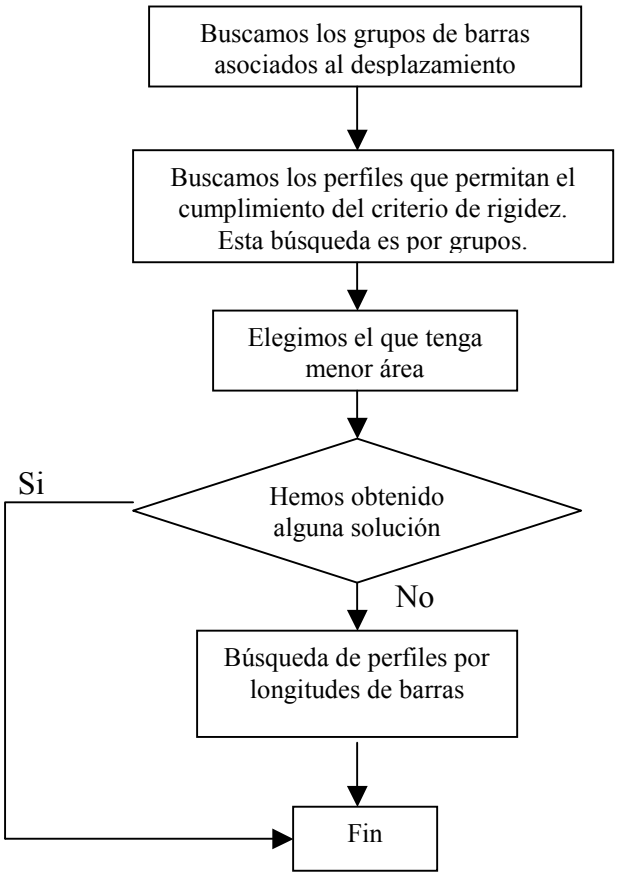


Fig. 19

Este algoritmo realiza dos búsquedas: la idea que se persigue en la primera búsqueda es evitar el primer inconveniente encontrado en el proceso anterior, es decir que estemos intentando imponer cotas a dos características geométricas que pertenezcan al mismo grupo de barras. Lo que hacemos es ir aumentando el perfil uno a uno hasta que encontramos una solución que satisfaga el criterio de rigidez. Esto lo hacemos con todos los grupos de barras que estén involucrados en el desplazamiento que tengamos restringido. Como podemos ver el proceso es muy parecido al anterior con la salvedad de que en vez de movernos entre características, lo que hacemos es movernos entre grupos de diseño. Este proceso no garantiza una solución porque como se comentó

anteriormente, puede darse el caso que los términos de la ecuación pertenezcan a grupos diferentes de diseño y donde solo halla una característica geométrica por barra.

Para describir el algoritmo de búsqueda por longitudes, rescribiremos el problema que queremos resolver:

$$\text{Min}(V) = \sum_{j=1}^{nvd} L_j^{vd} A_j \tag{51a}$$

$$\delta_s = \sum_{j=1}^{nvd} \frac{K_{sj}^A}{A_j} + \sum_{j=1}^{nvd} \frac{K_{sj}^I}{I_j} \le \delta_s^* \quad , \ s=1,S \tag{51b}$$

siendo L_j^{vd} la suma de las longitudes de las barras cuya variable de diseño es j, y δ_s^* las cotas conocidas en desplazamientos, con s=1,S y S el número de desplazamientos acotados. Admitimos que las deformaciones debidas al flector son mayores que las debidas al axil, como sucede en estructuras de poca altura. La idea básica del algoritmo es asignar las menores áreas posibles a los términos que más aporten al volumen.

Describimos ahora el algoritmo para el caso de nvd=3, por simplicidad:

1. Se ordenan las componentes de L_j^{vd} de mayor a menor. Sean m_1, m_2, m_3 los valores de j cuando ordenamos L_j^{vd}, es decir $L_{m1}^{vd} > L_{m2}^{vd} > L_{m3}^{vd}$.
2. Para $j=m_1$, cada término del sumatorio (51b) en que intervenga I_{m1} ha de ser menor que δ_s^*, por lo que de cada restricción en la que K_{sj}^I sea positivo podemos obtener una cota inferior de I_{m1} y tomar luego la máxima de todos.

$$\frac{K_{sm_1}^I}{I_{m_1}} \le \delta_s^*, \ s=1,S \ \Rightarrow \ I_{m_1}^l \le \text{Max}[\frac{K_{sm_1}^I}{\delta_s^*}, s=1,S]$$

3. Tomamos como valor de I_{m1} el valor discreto por exceso de I_{m1}^l ($I_{m_1} = \text{VDE}[I_{m_1}^l]$) y $A_{m_1} = A_{m_1}[I_{m_1}]$.
4. Sustituimos en (51b) y tenemos un sistema de desigualdades con una variable menos (I_{m1}).
5. Para $j=m_2$ se saca la cota de $I_{m_2}(I_{m_2}^l)$. Si $I_{m_2}^l$ no es admisible, porque $I_{m_2}^l > I_{max}$ (tablas), se aumenta I_{m1} al siguiente discreto, y se sigue en 4. Si $I_{m_2}^l$ es admisible, $I_{m_2} = \text{VDE}[I_{m_2}^l]$, $A_{m_2} = A_{m_2}[I_{m_2}]$.
6. Se sutituye en (51b) y tenemos un sistema de desigualdades con otra variable menos ($I_{m_2}^l$).

7. Para $j=m_3$, se saca la cota de $I_{m_3}(I_{m_3}^l)$. Si $I_{m_3}^l$ no es admisible, porque $I_{m_3}^l > I_{max}$ (tablas), se aumenta I_{m2} al siguiente discreto, y se sigue en 6. Si $I_{m_3}^l$ es admisible, $I_{m_3} = \text{VDE}[I_{m_3}^l]$, $A_{m_3} = A_{m_3}[I_{m_3}]$.

8. FIN.

Si en algún punto del algoritmo no hay ninguna solución admisible, admitimos que eso significa que no existen soluciones al conjunto de desigualdades. La Fig. 20 muestra el diagrama de flujo del algoritmo para cualquier valor de nvd.

Cuando la estructura es isostática el proceso seguido es el mismo, con la diferencia que como no tenemos incógnitas hiperestáticas el proceso de rediseño es único.

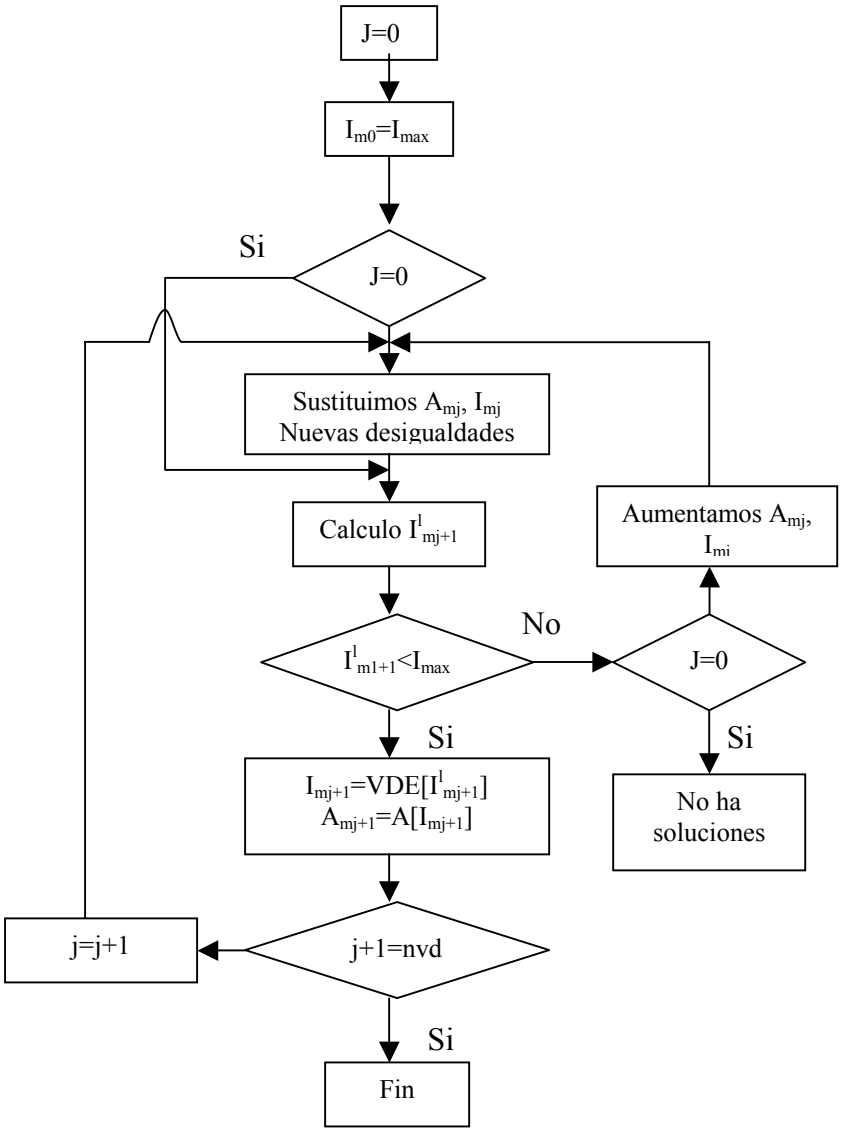


Fig. 20

3.7.2 Emparrillado.

Para el caso en el que trabajemos con emparrillados, el algoritmo de redefinición por rigidez funciona de forma muy parecida, lo único que hay que cambiar son las ecuaciones de compatibilidad-comportamiento:

$$\delta = \sum_{i=1}^{b} \left(\int_{0}^{L_{ki}} T^{v}(x) \cdot \frac{T(x)}{GJ_i} dx + \int_{0}^{L_{ki}} M^{v}(x) \cdot \frac{M(x)}{EI_i} dx \right) \quad (51)$$

donde T(x) y M(x) son las leyes de esfuerzos torsor y flector de cada barra, $T^v(x)$ y $M^v(x)$ son las leyes virtuales de esfuerzos, G, E son las constantes del material y J_i y I_i son las características geométricas de las barras.

En emparrillados, no podemos hacer la suposición realizada anteriormente acerca de las deformaciones debidas al axil y al flector, por lo tanto, no podemos usar el algoritmo anteriormente comentado (Fig. 20). Para poder garantizar al menos una solución aunque no sea la óptima, se ha usado un algoritmo de búsqueda parecido al mostrado en la Fig. 19, usando en vez del algoritmo de búsqueda por longitudes, un rastreo de posibles soluciones, aumentando poco a poco todos los grupos de barras asociados a cada desplazamiento.

3.8. Fin de iteración.

El criterio usado para el fin del proceso iterativo en el caso de estructuras hiperestáticas es la coincidencia de perfiles. Cuando el conjunto de perfiles coincida con el anterior, el proceso iterativo habrá terminado. Esto es lógico ya que al recalcular el valor de las hiperestáticas volveremos a obtener el mismo valor y por lo tanto volveremos a tener el mismo campo estático en equilibrio y en fin los mismos perfiles otra vez.

4-. PROGRAMACIÓN.

4.1. Introducción a Mathematica

Mathematica es un entorno de cálculo científico, compuesto por un lenguaje de programación de alto nivel, una colección extensa de funciones matemáticas y de propósito general, y un sistema para editar interactivamente documentos que contienen texto normal, órdenes sencillas, programas más complejos, gráficos, etc. Es un programa que intenta conjuntar un procesador de textos con un entorno de desarrollo matemático, donde a la vez que se va escribiendo información, se puede realizar un desarrollo matemático.

En todo lo que sigue nos referenciamos básicamente a la versión 3.0 de Mathematica. Las referencias básicas son [6],[7],[8],[9],[10],[11],[12],[13] que aparecen en el apartado de Referencias y la ayuda del programa Mathematica.

Mathematica puede ser usado como una simple calculadora numérica o simbólica, como plataforma para visualizar información o para ejecutar programas y aplicaciones. Se puede usar para crear documentos interactivos con animaciones, texto, sonido y fórmulas. Con Mathematica, además, tenemos la posibilidad de unir el uso de visualización matemática simbólica tal y como la conocemos y un uso por comandos mucho más apropiado para un usuario avanzado.

La construcción de los algoritmos en Mathematica se hace mediante un estilo de programación diferente a C. Aunque es posible usar construcciones de tipo for, while, if, etc., para controlar la asignación de valores a variables de acuerdo con la lógica del algoritmo (estilo "procedimental"), Mathematica está más orientado hacia un estilo de programación llamado "funcional" y, sobre todo, a la descripción de los algoritmos mediante lo que podemos llamar transformación estructural basada en "pattern matching" (reconocimiento de patrones sintácticos, o "esquemas de expresión").

Mathematica se divide en dos programas diferentes, uno llamado el FrontEnd, que es el encargado de formatear toda la información que entra y sale y el Kernel, que es el núcleo de programa. El Kernel tiene identidad propia y puede ser usado sin el FrontEnd debido a que es el que verdaderamente realiza todo el cálculo matemático. En la Fig. 21 podemos ver un diagrama del funcionamiento del programa:

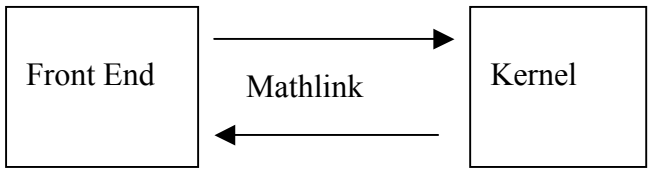


Fig. 21

Un usuario normal, cuando inicia una sesión de Mathematica, lo que hace es iniciar el FrontEnd que, aparte de ser un programa con entidad propia y lenguaje de programación propio que interpreta él mismo sin necesidad de llamar al Kernel, sirve de visualizador de la información, y es la interfaz de usuario del Kernel. Mientras no se

pretenda realizar ninguna operación que requiera el uso del Kernel, éste no será arrancado; por lo tanto podemos estar programando en el lenguaje del FrontEnd y no hacer ninguna llamada al Kernel. En el momento en que precisamos de alguna operación matemática, el FrontEnd hace una llamada al Kernel que es el programa encargado de realizar todas las. Como vemos en la Fig. 21, hay un paquete llamado Mathlink que acompaña a Mathematica desde la versión 2.2 y que hace de conexión entre el FrontEnd y el Kernel. Este mismo paquete puede ser usado para conectar el Kernel a otros programas como puede verse en uno de los ejemplos incluidos en Mathematica, donde tenemos una aplicación completamente programada en Visual Basic y que usa Mathlink para hacer llamadas al Kernel de Mathematica. En la versión 3 de Mathematica, usada en este proyecto, el lenguaje de programación del FrontEnd permite trabajar con objetos, botones y Notebooks (documentos), permitiendo una programación orientada a objetos.

Otra característica importante de Mathematica es que permite trabajar con contextos. Una forma sencilla de explicar lo que son los contextos es imaginar que son subdirectorios, es decir, podemos tener archivos con el mismo nombre mientras estén en directorios diferentes; de la misma manera, trabajando con contextos en Mathematica, podemos tener variables con el mismo nombre en diferentes contextos permitiendo que no haya interferencias entre ellas. Esta característica es muy importante asociada al uso de paquetes, ya que Mathematica, por un lado tiene lo que podemos llamar las habilidades innatas al programa, que son las operaciones básicas que podemos realizar (aunque esta idea es matizable, sin embargo esto es solo una breve introducción a Mathematica), y por otro lado tenemos una serie de aplicaciones o paquetes que no son de uso normal pero que permiten expandir el uso de Mathematica; éstas son aplicaciones más específicas programadas normalmente en lenguaje de Mathematica. Normalmente, éstas aplicaciones pertenecen o están en un contexto específico para evitar problemas de compatibilidad entre el trabajo realizado por el usuario y el paquete usado en cuestión.

Es importante entender los contextos y la forma que tienen de trabajar, sobre todo cuando entramos en el uso o programación de paquetes propios. Supongamos que tenemos un paquete llamado *Madre`grafico* donde tenemos una función que es por ejemplo *dibujarestructura[arg_]*. Si antes de cargar este paquete le damos ese mismo nombre *dibujarestructura[arg_]* a una variable, tendremos que cuando carguemos el paquete y queramos correr esa función, no estaremos cargando la función que pretendemos usar, sino la anterior, apareciendo un mensaje de Mathematica indicándonos que esa misma variable está en dos contextos, en el que estemos usando y en el contexto del paquete que hemos cargado. Para evitar estos problemas en los paquetes, podemos usar una palabra clave *`Private`* cuando estemos programando la parte de código correspondiente al paquete, evitándonos interacciones con el resto de variables que haya definidas en la sesión de Mathematica abierta en ese momento. Esta palabra indica a Mathematica que todo lo que está dentro del ámbito de ese comando tiene entidad propia y no tiene que interaccionar con el resto; sin embargo, básicamente, esto no es más que otro contexto.

4.2 Introducción a la aplicación.

La aplicación está formada por dos paquetes de cálculo de estructuras, uno para pórticos plano y otro para emparrillados, con su correspondiente interfaz. Los paquetes contienen toda la información referente al cálculo de dichas estructuras, todo lo que son las funciones de cálculo y las funciones gráficas, mientras que la interfaz interactúa con el usuario.

Las aplicaciones se han programado de forma que un usuario que no tenga conocimientos de Mathematica pueda usarlas; por lo tanto es de esperar que toda la gestión de las variables usadas la lleve la propia aplicación evitándonos el problema de nombrar variables iguales que puedan afectar al comportamiento de la aplicación (Véase 4.1). Sin embargo, también se ha querido dejar abierta la posibilidad de poder usar Mathematica independientemente, es decir que si durante el proceso de cálculo se desea realizar algún tipo de cálculo con Mathematica, se pueda hacer perfectamente dentro de la hoja (Notebook) de resultados, u otra cualquiera.

Ambas aplicaciones permiten un cálculo simbólico completo de ambos tipos de estructuras usando el método de las fuerzas y un cálculo numérico posterior donde se usan una serie de algoritmos de elección y búsqueda de perfiles. Este segundo proceso se puede realizar de forma manual o automática, siendo la manual asistida, de forma que el usuario podrá elegir entre una serie de perfiles (IPE, HEB, hueco circular, hueco cuadrado y hueco rectangular) mientras el programa le orienta sobre cuáles son los perfiles más adecuados y porqué. Así mismo, durante el proceso iterativo se le indicará cuáles son los criterios que hay que cumplir y se le explicará paso a paso el proceso iterativo seguido.

De forma resumida el programa permite realizar las siguientes operaciones:

- Resolución de estructuras por el Método de las Fuerzas. Estructuras isostáticas e hiperestáticas. (Pórticos planos y emparrillados).
- Introducción de datos vía consola.
- Calculo de estructuras con cargas en nudos, barras (permite usar distribuciones de carga lineales) y momentos en libertades en extremos de barras.
- Representación de la estructura y visualización de las cargas.
- Representación de diagramas de esfuerzos.
- Representación de la deformada.
- Resolución de problemas con cargas térmicas (sólo para pórticos planos) y desplazamiento en los apoyos.
- Resolución de la estructura de forma simbólica (Cálculo de hiperestáticas, reacciones en los apoyos, leyes de esfuerzos, desplazamiento de nudos y giro en libertades).
- Resolución de la estructura de forma numérica tanto manual como automática.
- Generación de informes (mediante impresión y mediante archivo)
- Utilización de 5 tipos de perfiles (IPE, HEB, hueco circular, hueco cuadrado, hueco rectangular).
- Redimensionado de estructuras por rigidez

- Modificación de los criterios de elección de perfiles.
- Uso independiente de los algoritmos de iteración.

Como se ha comentado, hay dos tipos diferentes de estructuras a calcular, pórticos planos y emparrillados. Estas estructuras tienen algunas similitudes pero también bastantes diferencias, así que desde el primer momento se intentó programar las aplicaciones de forma que se pudiera usar la mayor cantidad de código posible con ambas aplicaciones. Por este motivo, se ha usado un estilo de programación por funciones (ó funcional) en la que cada función recibe unos argumentos y devuelve unos resultados.

El código de cada una de las aplicaciones se ha generado con Mathematica y posteriormente se ha incluido en un paquete que puede ser cargado por el usuario mediante la interfaz de usuario o mediante una sesión normal de Mathematica.

No es la intención de esta parte de la memoria hacer una explicación exhaustiva de cada una de las funciones programadas en la aplicación ni hacer una análisis completo de la aplicación, para ello se puede consultar el listado completo al final del documento, donde aparecen algunas partes comentadas.

4.3 Esquema general de programación.

Para ambos tipos de aplicaciones, tenemos el mismo esquema general de funcionamiento que es el que aparece en la Fig. 22. Este esquema presenta una serie de ventajas y una serie de inconvenientes. Por un lado, veremos que tenemos 3 partes del programa que son visuales, es decir con las que el usuario trabajará. Tenemos una Interfaz que se explicará posteriormente, que permite al usuario realizar las operaciones que necesite. Tenemos un Input que nos permite la introducción o toma de datos en tiempo real de ejecución y un documento de salida que permite visualizar todos los resultados generados por la aplicación. Estas tres partes serán explicadas con más detalle posteriormente.

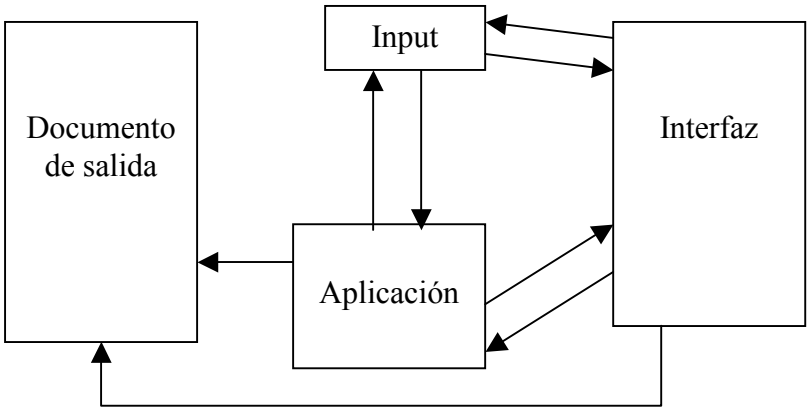


Fig. 22

La otra parte que podemos ver en la Fig. 22 es lo que llamo la aplicación. La aplicación realmente es un paquete que cargamos al inicio del programa. Como se comentó anteriormente, cada uno de los paquetes, tiene su propio contexto, que son *Estructuras`portico* y *Estructuras`emparrillado* para pórticos planos y emparrillados respectivamente.

Anteriormente se comentó que una forma de aislar el funcionamiento del paquete para evitar posibles interferencias es usar *`Private`*, sin embargo su uso en estos paquetes ha sido bastante reducida por los siguientes motivos:

a) La aplicación ha sido realizada con la idea de simplificar su uso a los posibles usuarios, sobre todo a los que han tenido poco contacto con Mathematica; intentado evitar por un lado el handicap que produce el desconocimiento de un programa, y por otro lado permitir la ínter actuación lo máximo posible y de la manera más sencilla posible. Esto permite al programador un mejor control de las variables usadas, y una necesidad menor de proteger las funciones programadas.

b) Como podemos ver en la Fig. 22, existen muchas conexiones entre los diferentes bloques del diagrama, y esto permite un peor control de las variables y contextos. Como poede verse, tanto el paquete como la interfaz tienen acceso al documento de salida; por lo tanto, en el caso que toda la aplicación permaneciera bajo el control de *`Private`,* las variables asociadas hay que especificarlas de forma más concreta, es decir, usando el nombre completo de la variable *Estructuras`portico`Private`nombrevariable* . Como puede entenderse ese es un trabajo que no merece la pena cuando la interfaz es la que está gestionando las variables.

Se puede decir básicamente que se ha buscado que la aplicación sea más funcional que robusta. Se buscaba una aplicación que permitiera un uso amigable y que permitiera la interacción y se ha sacrificado un poco la robustez de la aplicación. Esto no es tan crítico, en el sentido que las funciones programadas están protegidas (usando *Protect*), y aquellas que se han podido introducir dentro del control de *`Private`* se han introducido; sin embargo no se ha buscado una programación orientada a esa forma de programar. Algo que como hemos visto en el apartado 4.1 no es tan relevante y puede en un momento dado producir interacciones con otras variables.

4.4 ¿Por qué usar Mathematica?

El objetivo del proyecto es realizar una aplicación que permita el cálculo analítico paso a paso de estructuras por el método de las fuerzas. Con esta aplicación se pretende ofrecer un apoyo informático a aquellas personas que deseen tener una herramienta para comprobar por sí mismos sus resultados en el cálculo de las estructuras.

Como también se comentó anteriormente, el problema docente tiene dos partes completamente diferenciadas: por un lado un cálculo simbólico detallado y por otro lado una parte numérica de diseño.

Mathematica es capaz de manejar una programación "orientada a objetos", como se comentó anteriormente, permitiendo el uso de botones y documentos y su tratamiento como objetos con propiedades, y permitiendo además usar el concepto de herencia entre los objetos. Esto permitió crear una interfaz de usuario más adecuada, que permitiera el uso de la aplicación incluso a personas con poca experiencia con Mathematica.

4.5 Funciones.

Como se ha comentado anteriormente, la aplicación se ha programado como una serie de funciones, las cuales han sido posteriormente conjuntadas en la interfaz de usuario. Las funciones pueden dividirse según la actividad realizada en 3 tipos importantes:

- Funciones gráficas: Estas son las funciones que explícitamente desarrollan los dibujos, es decir dibujo de la estructura, diagramas de esfuerzos etc.

- Funciones de cálculo: Entre estas funciones se encuentran aquellas que por ejemplo montan las matrices de equilibrio, calculan el valor de las hiperestáticas, realizan los procesos iterativos, etc.

- Funciones de introducción de datos: Estas funciones se han introducido por la necesidad de tener una forma de introducir datos más eficiente que la proporcionada por Mathematica

4.5.1 Funciones gráficas.

Los motivos por los que se han programado estas funciones gráficas son dos: por un lado para saber con qué se está trabajando, y por otro lado como comprobación de los datos introducidos, ya que un dibujo nos permite comprobar rápidamente si ha habido errores en la introducción de datos geométricos y de cargas.

En este tipo de funciones es donde más se diferencian las aplicaciones programadas. El motivo es muy claro, los pórticos planos son estructuras bidimensionales que están cargadas dentro de su plano, lo que permite realizar una representación completamente bidimensional tanto de la estructura como de sus cargas; sin embargo, los emparrillados son estructuras bidimensionales cargadas fuera de su plano, y por lo tanto la representación total tanto de la estructura como de sus cargas es una representación tridimensional.

Una representación tridimensional en Mathematica puede llegar a ser una representación poco clara, y por este motivo se ha decidido realizar una representación bidimensional en vez de una tridimensional, usando una serie de símbolos que nos indicarán en el caso que sea necesario, la dirección de las cargas perpendiculares al plano de la estructura.

El motivo de realizar esta representación bidimensional de los emparrillados es doble: por un lado por simplicidad a la hora de visualizar la estructura, ya que en esta representación bidimensional podemos ver claramente la colocación de las barras y los ángulos entre ellas; por otro lado ello a permitido volver a usar la mayor cantidad posible de las funciones programadas para pórticos planos.

Básicamente, el programa tiene 5 funciones gráficas:

- Una primera rutina gráfica que se encarga de dibujar la estructura, es decir sus barras nudos
- Una segunda función que nos permite enumerar los nudos y barras, indicar el sentido en que se han definido esta últimas y acotar el dibujo.
- Una rutina gráfica que genera los dibujos de las fuerzas aplicadas en nudos, barras y libertades de barras.
- Una rutina que genera los diagramas de esfuerzos de las barras
- Una rutina gráfica que genera la deformada de la estructura.

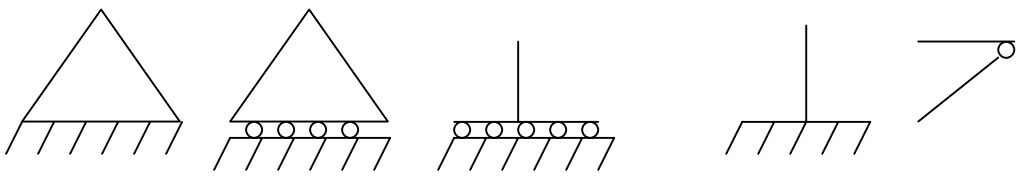
4.5.1.1 Representación de la estructura.

Esta función toma los valores introducidos, es decir las coordenadas de los nudos de la estructura, y genera la cantidad de líneas necesarias que unan dichos puntos mediante la conectividad de barras que es un dato introducido anteriormente.

En el caso que tengamos alguna barra con carga térmica, ésta es dibujada con un grosor un poco superior al del resto de las barras y en color rojo.

4.5.1.1.1 Apoyos y libertades internas en pórticos planos.

Lo siguiente que tenemos que dibujar son los apoyos y libertades de barras. Esto se ha hecho generando unos iconos básicos que son los mostrados en la Fig. 22:



a) b) c) d) e)

Fig. 23

Cada uno de estos símbolos se caracteriza porque impone una serie de restricciones al desplazamiento o permite una serie de movimientos. Debido a esta

casuística, algunos de estos símbolos necesitan ser girados cuando el desplazamiento permitido así lo necesite. Por ejemplo, a) impide los desplazamientos vertical y horizontal, permitiendo el giro del extremo de la barra, este símbolo no necesita ser girado (excepto por motivos estéticos) debido a que la dirección del movimiento permitido es perpendicular al plano de la estructura. En el caso b), tenemos que el desplazamiento impedido es el vertical, por lo tanto será uno de los símbolos que puede girar según sea el desplazamiento impedido vertical u horizontal.

La forma de operar ha sido generar unas funciones que generaran estos dibujos, y que según el caso que tuviéramos girara el símbolo o no. Solo hay un caso que no está contemplado y es el caso en que la barra esté inclinada una ángulo diferente de 0º ó 90º y tenga el desplazamiento permitido en la dirección normal a la barra mientras tiene el axial a la barra restringido (independientemente del giro), véase b) y c) girados un ángulo intermedio entre 0º y un múltiplo de 90º. El motivo nace de la propia introducción de datos debido a que nosotros introducimos los desplazamientos impedidos en coordenadas globales, por lo tanto un apoyo de esta forma en globales tiene "permitidos" tanto el desplazamiento vertical como el horizontal y por lo tanto no correspondería con ninguno de los casos anteriormente propuesto, en todo caso, corresponderían con un nudo libre y un nudo con el giro impedido respectivamente. Otro motivo por el que no se ha introducido este caso es que complica de una manera innecesaria el proceso de cálculo sin aportar nada nuevo.

En el caso de las libertades asociadas a las barras (caso e), se ha desplazado ligeramente el dibujo de la libertad (círculo) evitando dibujarla exactamente en el centro geométrico del nudo. Este pequeño desplazamiento permite al usuario determinar a qué barra pertenece la libertad.

Por último, comentar que en todas las rutinas gráficas ha sido necesario escalar los dibujos, con lo que estas rutinas sirven para generar la estructura tanto en el caso adimensional como en el caso dimensional.

4.5.1.1.2 Apoyos y libertades internas en emparrillados.

Como se comentó anteriormente, los emparrillados son estructuras planas cargadas fuera de su plano, por lo tanto para realizar la representación de las libertades en extremos de barras y los apoyos, la filosofía usada es diferente a la de pórticos planos.

Los apoyos en un emparrillado, teóricamente pueden llegar a ser de muchas clases, si entendemos como clases cada una de las representaciones más o menos reales del tipo de restricción que estamos imponiendo. Esto, unido a la representación bidimensional que vamos a realizar complica bastante la representación de los apoyos en este tipo de estructuras (aunque no mejora demasiado en el caso de una representación tridimensional). Por poner un ejemplo, imaginemos que tenemos un apoyo que permite el giro a flexión de la barra e impide el giro a torsión y el desplazamiento fuera del plano de la estructura. Esto sería como una bisagra horizontal empotrada en una chapa. Si el desplazamiento permitido es el giro a torsión, la

representación empieza a convertirse en algo más complicado porque sería algo así como una bisagra empotrada en una chapa que gire alrededor del eje de la barra.

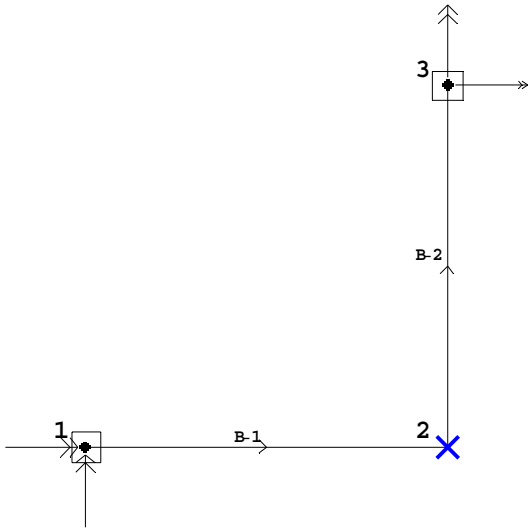


Fig. 24

La solución tomada consta de dos partes:

a) En cada nudo donde tengamos un apoyo colocaremos un cuadrado que indicará que ese nudo está apoyado (Véase nudos 1 y 3 de la Fig. 24).

b) Para indicar cuales son los grados de libertad coaccionados, representaremos la fuerza (reacción) a él asociada. Las posibles reacciones son 3, un momento en el eje "y" global de la estructura, representado como puede verse en el dibujo con un vector con doble flecha, un momento en la dirección x de los ejes globales de la estructura (representado también con doble flecha) y por último una reacción vertical que sale del plano del papel. Para representar las fuerzas que entran y salen del papel se ha tomado como criterio representar un punto (que indica la punta) cuando tiene sentido positivo (es decir sale del papel) y una cruz cuando tenga sentido negativo (entrando en el papel), como se ilustra en la Fig. 24.

   Una sección de una barra puede tener dos giros, uno alrededor del eje de la barra y otro alrededor del eje perpendicular tal como puede verse en la Fig. 7b. Por lo tanto las libertades rotacionales en los extremos de barras pueden ser de tres tipos, aquellas que permitan el giro alrededor del eje de la barra, aquellas que permitan el giro alrededor del eje perpendicular al eje de la barra (eje z local) y aquellas que permitan ambas, es decir rótulas.

   a) Para las libertades que permiten el giro alrededor del eje x local, se ha dibujado un cuadrado.

   b) Para las libertades que permiten el giro alrededor del eje z local, se ha dibujado un círculo.

   c) Para las libertades que permiten ambos giros se han dibujado los dos símbolos.(Véase Fig. 25)

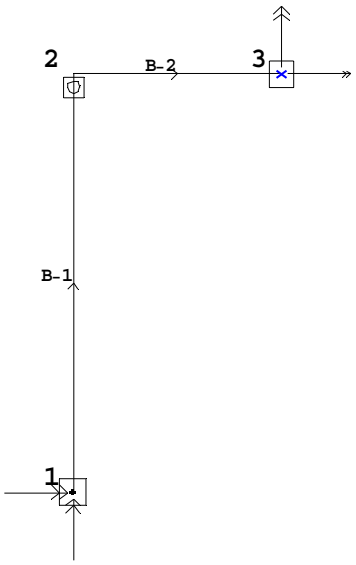


Fig. 25

4.5.1.2 Representación de fuerzas aplicadas.

Se ha desarrollado una función que genera los dibujos de las fuerzas aplicadas a la estructura, es decir, las aplicadas a los nudos de las barras, las aplicadas a las barras y los momentos aplicados en las libertades de barras.

Debido al tipo de cargas que presentan los diferentes tipos de estructuras, la forma de representarlas también será diferente.

4.5.1.2.1 Representación de cargas aplicadas en pórticos planos.

En pórticos planos pueden existir tanto fuerzas puntuales en los nudos (en coordenadas globales), como cargas distribuidas (longitudinales y transversales) sobre las barras (en coordenadas locales). Los momentos aplicados en libertades de barras se consideraron un caso intermedio entre los dos casos comentados, ya que un momento dibujado en un pórtico plano es fácil de dibujar (simplemente hay que dibujar un arco y una punta de flecha); sin embargo dibujar de cualquier forma ese arco puede dar lugar a confusión, ya que no nos indica si el momento está aplicado en el nudo o en la libertad, como puede verse en la Fig. 26, donde tenemos un momento aplicado a una libertad en la barra inclinada y un momento aplicado en un nudo. Si ambas se dibujaran igual, el usuario no podría saber si ha introducido bien los datos, o simplemente no sabría si lo que tiene es un momento aplicado en una libertad o un momento aplicado en un nudo.

Para la generación de las fuerzas se han desarrollado un conjunto de funciones básicas que dibujan una punta de flecha, una flecha y un momento. En un principio se genera cada uno de ellos en el origen de coordenadas, y según el caso que tengamos, montamos el símbolo. Por ejemplo, la función “punta de flecha” genera una punta de flecha según el tamaño indicado, la gira el ángulo que sea necesario para que indique la

dirección adecuada y la coloca en el sitio que haga falta. Esta función es muy útil porque si por ejemplo queremos dibujar una flecha para indicar una fuerza, solo tendremos que indicarle donde tiene que situarse e incluir la línea. Esto es lo que hace otra función, genera la flecha completa indicándole la longitud de la flecha y la posición del origen. Por último, hay una función que genera el dibujo de un momento donde se le indique y con una cantidad de ángulos determinada. Esta función vuelve a usar la función punta de flecha para indicar la dirección del momento. Básicamente lo que se hace es montar el símbolo que nos hace falta, luego lo gira el ángulo que hace falta y por último lo coloca en su sitio.

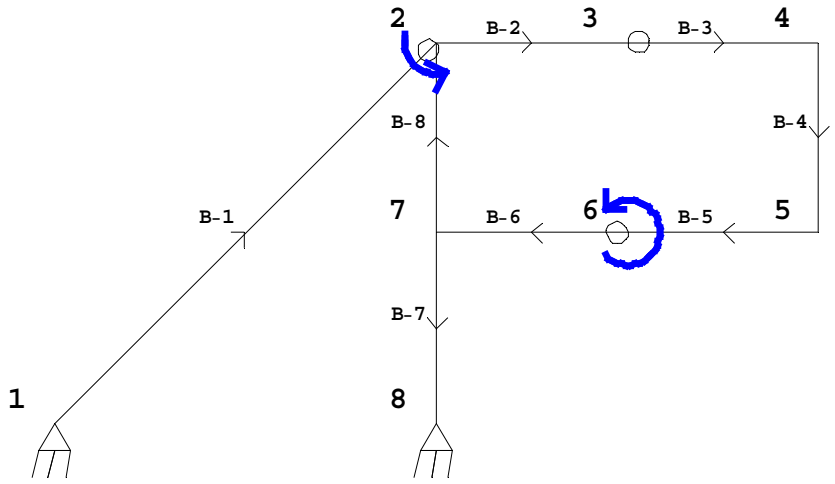


Fig. 26

Para dibujar las fuerzas en los nudos se intentó en todo momento evitar las interferencias de las líneas de las fuerzas con las líneas de la estructura. Esto ha sido complicado de resolver porque si por ejemplo tenemos una estructura como la de la Fig. 27, donde existen cargas horizontales, si queremos dibujar las cargas, normalmente se dibujarían como se muestra en la Fig. 27; por lo tanto no podemos usar un criterio de dibujo que sea dibujar el extremo en el nudo en cuestión porque en ese caso la fuerza aplicada en la parte izquierda estaría dibujada dentro de la estructura.

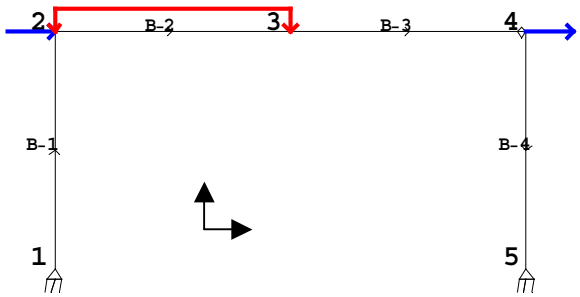


Fig. 27

Esto es un problema estético que no tiene mayor importancia, así que se ha intentado buscar una solución sencilla y que sea útil en la mayoría de los casos (es decir en estructuras de topología sencilla). Lo que se ha hecho ha sido determinar cual es la

coordenada x del punto más alejado, que es muy fácil de determinar; una vez tenemos esta coordenada, el criterio elegido es dibujar el extremo de la flecha en el nudo cuando el nudo tenga una coordenada x menor que esa coordenada. Lo mismo se ha hecho con las verticales como puede verse en la Fig. 28, y aunque el resultado no siempre es exacto, es una manera sencilla de mejorar la presentación.

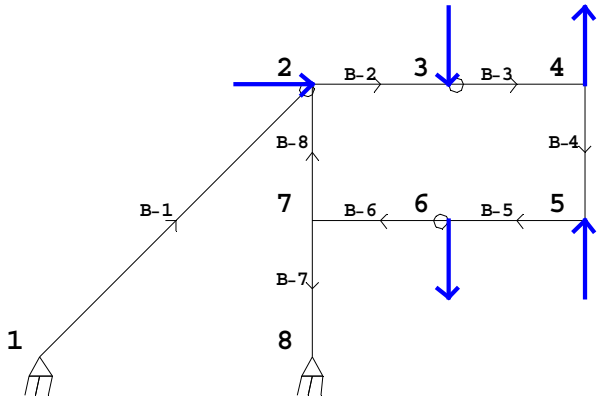


Fig. 28

Las barras de la estructura pueden estar sometidas a cargas distribuidas transversales o longitudinales, constantes o lineales. Estas cargas se representarán en la zona positiva del eje normal al de la barra (Fig. 29). El resto de detalles se ha realizado como antes, habiéndose dibujado las flechas con la rutina anteriormente comentada, simplemente realizando una discriminación de signos y diferenciando mediante colores (y flechas) las cargas transversales y longitudinales.

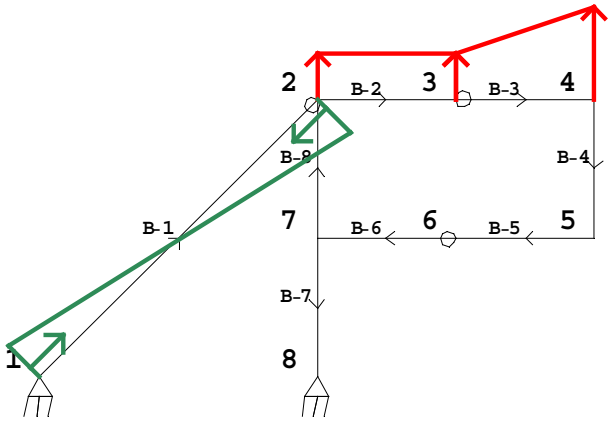


Fig. 29

4.5.1.2.2. Representación de cargas externa en emparrillados.

La metodología descrita para la representación de cargas aplicadas en pórticos planos es la misma usada aquí. La única diferencia con los pórticos planos, como se comentó anteriormente es la forma de representar las cargas propias de los emparrillados.

Al igual que en pórticos planos, tenemos que diferenciar entre cargas puntuales en los nudos y cargas distribuidas en las barras. Para los nudos, podemos tener dos

momentos aplicados y una carga vertical. Los momentos se representan con doble flecha y las cargas verticales con el criterio que anteriormente se comentó. El único problema que tiene este tipo de representación es que no nos da una idea cuantitativa del valor de la fuerza vertical; sin embargo, esta carencia no se considero que sea de gran relevancia. En la Fig. 30 podemos ver un ejemplo de cada una de estas cargas.

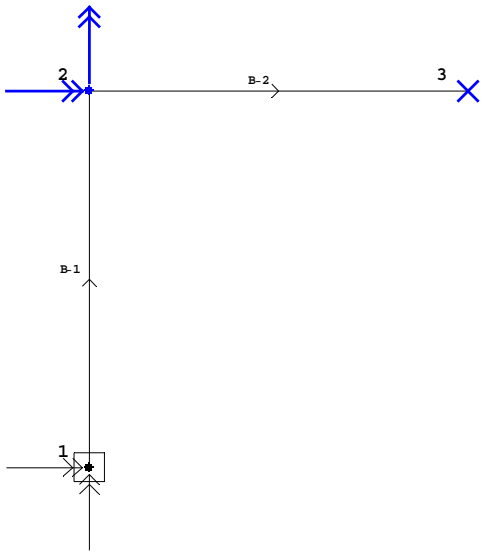


Fig.29

En emparrillados, sólo podemos cargar las barras con momentos distribuidos y con cargas transversales a las barras, sin embargo, el momento distribuido no se ha considerado en esta aplicación por ser un caso teórico y poco real. Las cargas distribuidas si se han dibujado abatidas sobre el plano de la estructura. En la Fig. 31 podemos ver la representación de estas cargas. Cuando las flechas tiran de la barra, la barra está carga en sentido positivo (hacia arriba), mientras que si las flechas inciden sobre la barra indicarán que el sentido de las flechas es negativo (hacia abajo).

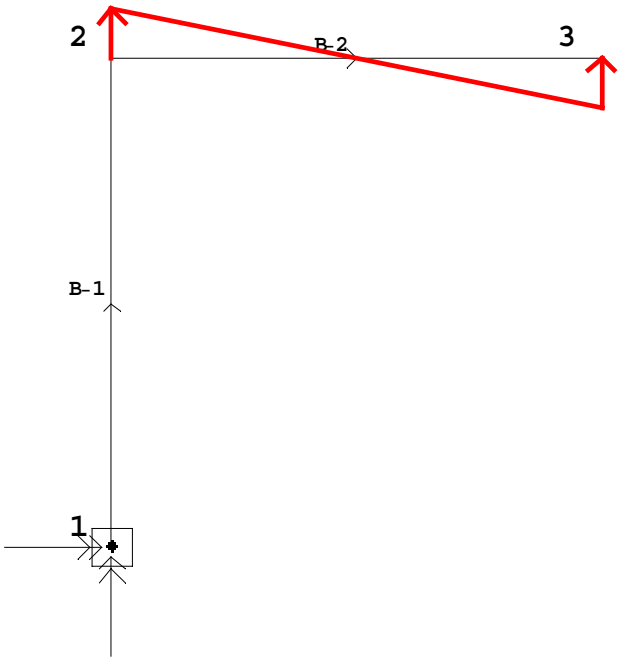


Fig. 31

4.5.1.3 Diagramas de esfuerzos.

Una de las representaciones más importantes en el cálculo de estructuras es la representación de los diagramas de esfuerzos. Para el caso en que tengamos un pórtico plano o un emparrillado, hay que representar 3 diagramas por barra, los diagramas de axiles, cortantes y flectores para pórticos y los de torsores, cortantes y flectores para los emparrillados.

Como era de esperar, existen diferencias entre la representación de los diagramas de esfuerzos para pórticos y emparrillados; sin embargo estas diferencias en este caso concreto son minúsculas como se podrá comprobar, debido a que lo único que ha habido que modificar son los símbolos que representan cada uno de los esfuerzos.

4.5.1.3.1.Representación de diagramas de esfuerzos para pórticos planos.

Los diagramas de esfuerzos para pórticos planos representan la evolución de los esfuerzos a lo largo de la barra. En este caso particular de pórticos planos sólo tendremos que tener en cuenta la representación del axil, cortante y flector.

Se han desarrollado unas funciones básicas similares a las anteriores que dibujan una punta de flecha, y una rebanada, y otras más complicadas que dibujan las flechas (rectas y de momentos) y los iconos correspondientes a axil, cortante y flector.

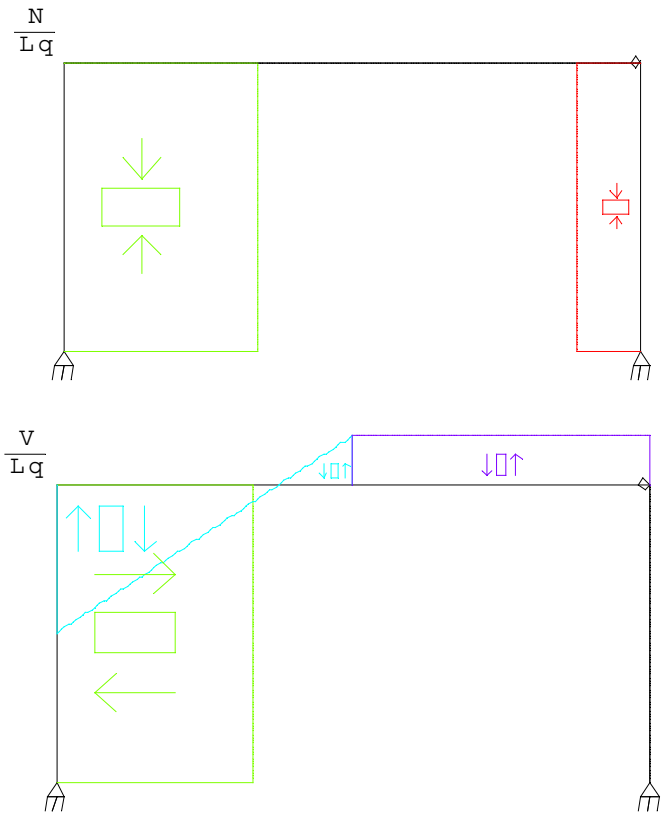
El siguiente paso a realizar es el dibujo de los diagramas. Al ser las cargas distribuidas de hasta primer grado (lineales), los diagramas de momentos serán como mucho un polinomio de tercer grado; por lo tanto tendremos que dibujar funciones desde constantes hasta polinomios de tercer grado.

Las leyes de esfuerzos pertenecen a cada barra, y se dibujarán sobre la estructura. En un principio, se pensó en dibujar simplemente las leyes de esfuerzos sobre cada una de las barras por separado, pero esto hubiera generado una gran cantidad de información (3 * b representaciones, siendo b el número de barras) y además sería poco ilustrativo, así que se decidió montar los diagramas de cada barra sobre la estructura. El problema que planteaba esto éra que se tendría que mover, rotar y girar cada una de las funciones. Por otro lado, se planteaba la necesidad de homogenizar los valores a representar; por un lado, teníamos la estructura y por otro lado los diagramas que algunas veces mostraban valores bastante dispares. Este problema era menos importante cuando se representaban exclusivamente cada diagrama en su barra; sin embargo aquí teníamos dos opciones, o escalar la representación de los diagramas a las barras o a la estructura. La primera opción era una opción poco adecuada debido a que en el caso de pórticos planos, una comprobación sencilla para ver si se han calculado los diagramas (leyes de esfuerzos) bien, es comprobar la continuidad (mismo valor) en los nudos de los momentos, y si se escalaba por barras estos valores no tendrían porqué coincidir, así que la solución adoptada fue la segunda. Se calcula para cada ley de esfuerzo, el máximo y el mínimo valor que tiene en el conjunto de la estructura y se toma como valor característico en el escalado de los diagramas la suma de ambos.

Por los motivos antes expuestos, pareció más lógico dibujar las leyes de esfuerzos de una manera discreta, aunque a primera vista pareciera más lógico hacer una transformación polar.

El último paso que queda es colocar los iconos representativos. En un principio no hubiera hecho falta dibujar los iconos debido a que, cuando se representan los diagramas, la parte que queda en el lado definido como positivo según la dirección de la barra, es la parte positiva del diagrama, sin embargo para simplificar la lectura de los resultados se decidió pintar los iconos. Los iconos se dibujan tomando como valor representativo de su tamaño el valor medio del intervalo, ya que hemos dicho que podemos tener hasta polinomios de tercer grado. Un polinomio de tercer grado puede tener hasta 3 raíces reales, por lo tanto podemos tener hasta 4 intervalos en una misma barra (caso poco probable), así que tenemos que determinar los intervalos de cambio de signo que estén dentro de nuestro dominio de trabajo. Otro problema está en la colocación horizontal de los iconos: la idea que se usó fue ponderar hacia el lado más grande del diagrama. Imaginemos que tenemos una ley de esfuerzos lineal que cambia de signo dentro del dominio de la barra; tendremos entonces que dibujar dos iconos. Si dibujamos el icono justo en la mitad, desperdiciaremos la mayor parte del espacio interior del dibujo o el icono aparecerá demasiado grande y se saldrá. La manera de solucionar esto es acercando más el icono hacia el valor más grande del diagrama, y esto se hizo ponderando los valores de los diagramas mediante la fórmula de carácter experimental:

$$\frac{valorinicial - valorfinal}{12 \cdot valorenmedio} longitud\,int ervalo$$



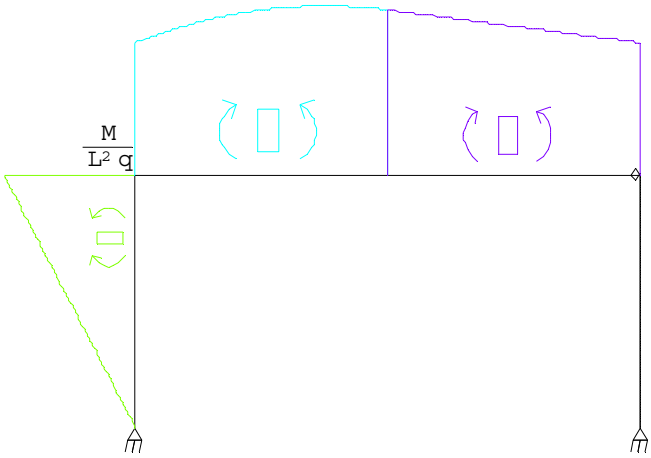


Fig.31

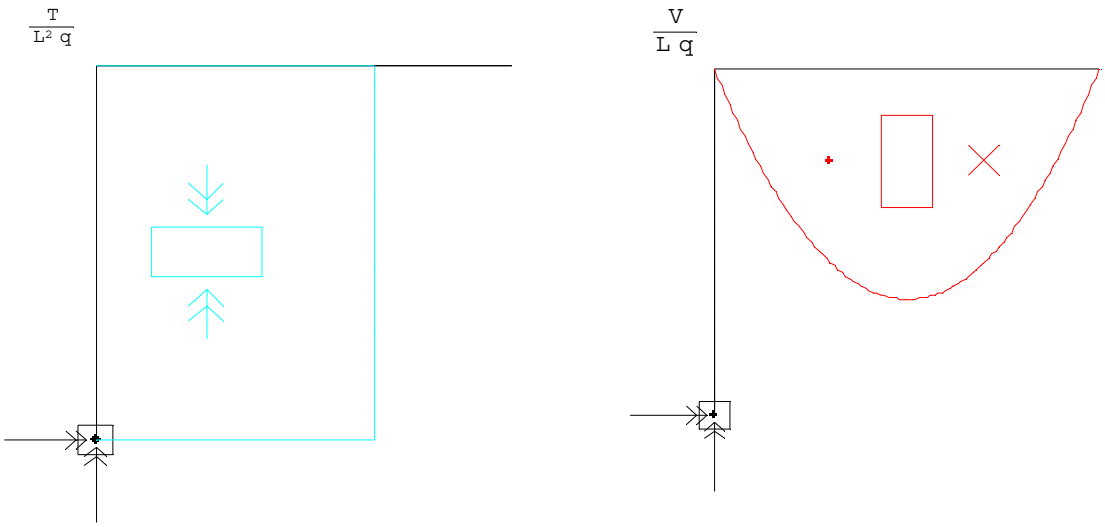
donde valorinicial, valorfinal y valorenmedio son valores en tres puntos del intervalo de la ley de esfuerzos y longitudintervalo la longitud del intervalo. El factor 12 es un factor experimental que suele ajustar bastante aceptablemente los tamaños.

Como puede verse en la Fig. 32, se han introducido unos identificadores en las esquinas superiores izquierdas, que aparecen cuando se representan los diagramas de esfuerzos de forma adimensional, es decir cuando tenemos que las leyes de esfuerzos son función de las cargas externas, y de las hiperestáticas. En estos casos se han "adimensionalizado" las leyes de esfuerzos para poder representar los diagramas, dibujando un diagrama por cada incógnita hiperestática, más uno para las cargas externas. En este caso en concreto la estructura es isostática y sólo aparecen estos dibujos.

Las leyes de esfuerzos también se representarán al final del cálculo, cuando tengamos todos los perfiles determinados.

4.5.1.3.2. Representación de los diagramas de esfuerzos para emparrillados.

Todo lo que se ha dicho anteriormente en pórticos planos es aplicable a los emparrillados. Puede observarse en la Fig. 33, que lo único que cambia es la



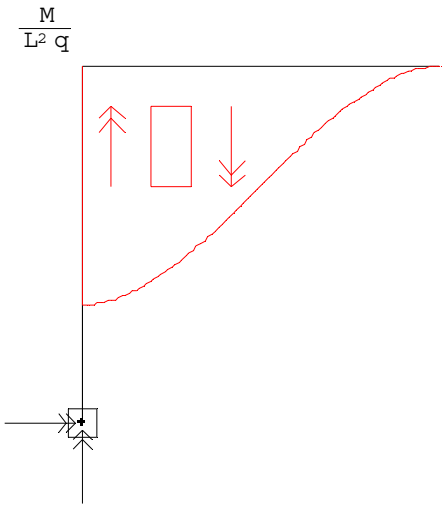


Fig. 33

representación de los iconos, que ahora son dobles flechas enfrentadas para indicar el torsor (33.1), punto y cruz para indicar el cortante (33.2) y doble flecha paralela (33.3) para indicar el sentido del momento flector.

4.5.1.4 Representación de la deformada.

La rutina para la representación de la deformada, trabaja de forma similar a la anterior. En este punto también existen diferencias a la hora de representar la deformada de un pórtico plano y de un emparrillado. Los desplazamientos de un pórtico plano son el desplazamiento longitudinal y transversal de la barra y un giro. Estos desplazamientos se producen dentro del plano de la estructura por lo tanto son fácilmente representables bidimensionalmente, mientras que en un emparrillado todos los desplazamientos son fuera del plano de la estructura, por lo que la representación debería hacerse de forma tridimensional.

4.5.1.4.1 Deformada de un pórtico plano.

Para representar la deformada de un pórtico plano, hemos calculado las leyes de desplazamientos de cada barra en función de su coordenada local x, obteniéndose de esta manera las leyes de desplazamientos verticales y horizontales. Tomando una cantidad de puntos de cada ley podemos determinar cuánto es el incremento en la posición original de cada punto y de esta manera obtenemos un conjunto de parejas (x, y) las cuales giramos y trasladamos, obteniendose la deformada, que se dibuja sobre la estructura (Fig. 34).

Los desplazamientos de una estructura son pequeños; para poder representar de forma cualitativa la deformada, los valores se han mayorado por un factor que es el 10% de la longitud L de la estructura.

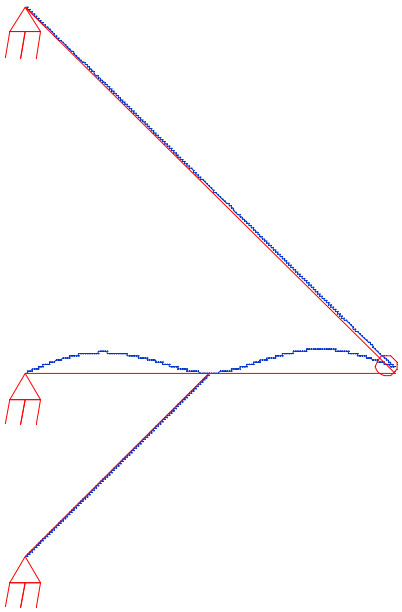


Fig. 34

4.5.1.4.2 Deformada de un emparrillado.

Sabemos que los emparrillados están cargados fuera de su plano y los desplazamientos que este tipo de estructura puede sufrir son dos giros y un desplazamiento normal a su plano como puede verse en la Fig. 34.

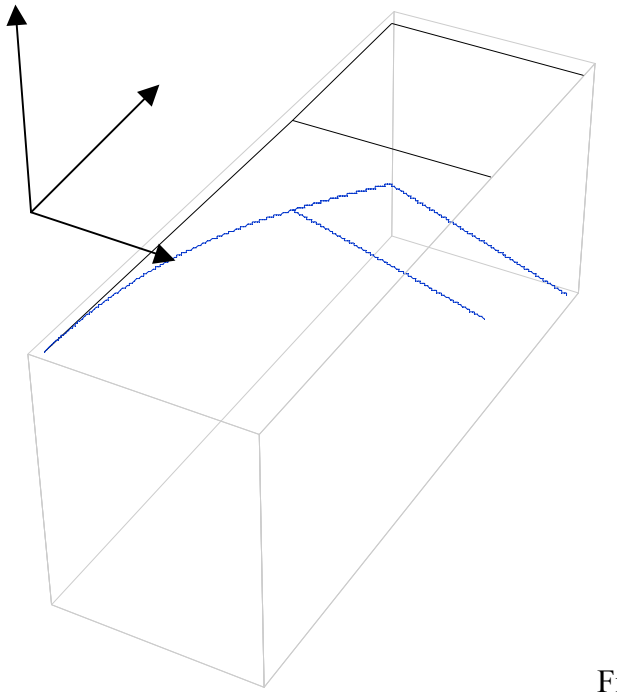


Fig. 35

Debido a este motivo se ha tenido que realizar una representación tridimensional de la estructura y de su deformada. La forma de dibujarla ha sido análoga a la anteriormente comentada en pórticos con la salvedad de tener que trabajar a nivel tridimensional.

En la Fig. 35 podemos ver que se ha dibujado los ejes de referencia, lo que da idea más clara de la tridimensionalidad al problema; para evitar que os ejes entorpezcan la visualización del dibujo, se les ha dado un color gris claro, mientras que la estructura original está en negro y su deformada en azul.

Al igual que hicimos en pórticos, los desplazamientos se han mayorado para mejorar su visualización; el factor de mayoración es ahora un 5% del valor genérico L de la longitud.

Para dibujar la deformada de la estructura se ha tenido en cuenta que en cada nudo tenemos dos giros, uno en la dirección "x" y otro en la dirección "y", esto nos obliga a determinar cuanto vale el giro de la barra asociado a su desplazamiento vertical.

4.5.2 Funciones de cálculo.

Las funciones de cálculo son una serie de funciones que se han programado para realizar el cálculo tanto simbólico como numérico; por lo tanto se pueden diferenciar entre lo que son las funciones de cálculo simbólico y las funciones dedicadas al diseño.

Entre las funciones de cálculo simbólico, están todas aquellas que participan en el montaje de matrices de equilibrio, cálculo de hiperestáticas, generación de matrices de transformación, etc. Estas funciones no tienen una gran relevancia en el sentido de que simplemente se limitan a realizar una serie de tareas que no tienen ningún valor añadido a lo anteriormente comentado en el apartado teórico.

Entre las funciones numéricas sí hay algunos aspectos que hay que aclarar, aparte de los ya explicados anteriormente en el apartado teórico donde se han expuestos los algoritmos de iteración usados.

4.5.2.1. Funciones de iteración.

Las funciones de iteración siguen en su mayor parte de código los algoritmos explicados anteriormente; algunos de estos algoritmos no son exactos como se explicó en su momento, por ejemplo en el caso de los perfiles cerrados sometidos a compresión. Para paliar posibles errores o simplemente para hacer las funciones de iteración más robustas se ha programado un pequeño algoritmo de búsqueda más computacional.

4.5.2.2. Búsqueda de la sección más desfavorable.

Cuando una barra está sometida solo a flector (y cortante) o sólo tiene axil, la búsqueda de la sección más desfavorable es sencilla; sin embargo cuando la barra o barras están sometidas a los tres esfuerzos, la determinación de la sección más desfavorable, como se indicó en el apartado 3.4, es más complicada.

Cuando tenemos un caso como el comentado anteriormente en el que tenemos axil y flector, la elección del perfil hemos dicho que se hace por flector, pero en una misma sección podemos tener (como se ve en la Fig. 36) un mismo flector y un axil con una discontinuidad. En estos casos tenemos un doble problema:

a) Por un lado, obviamente, tendremos que la sección más desfavorable será aquella en la que el axil sea mayor siempre y cuando tengan los dos el mismo sentido; sin embargo cuando tenemos de diferente sentido, la elección se complica un poco debido a que cuando el axil es de compresión las tensiones están mayoradas por un factor ω.

b) El otro problema es el numérico. Puede darse el caso que en la sección de mayor axil, numéricamente el flector salga algo menor (errores de redondeo); por lo tanto el ordenador elegiría una sección que no es la más desfavorable, es decir la de menor axil y mayor flector.

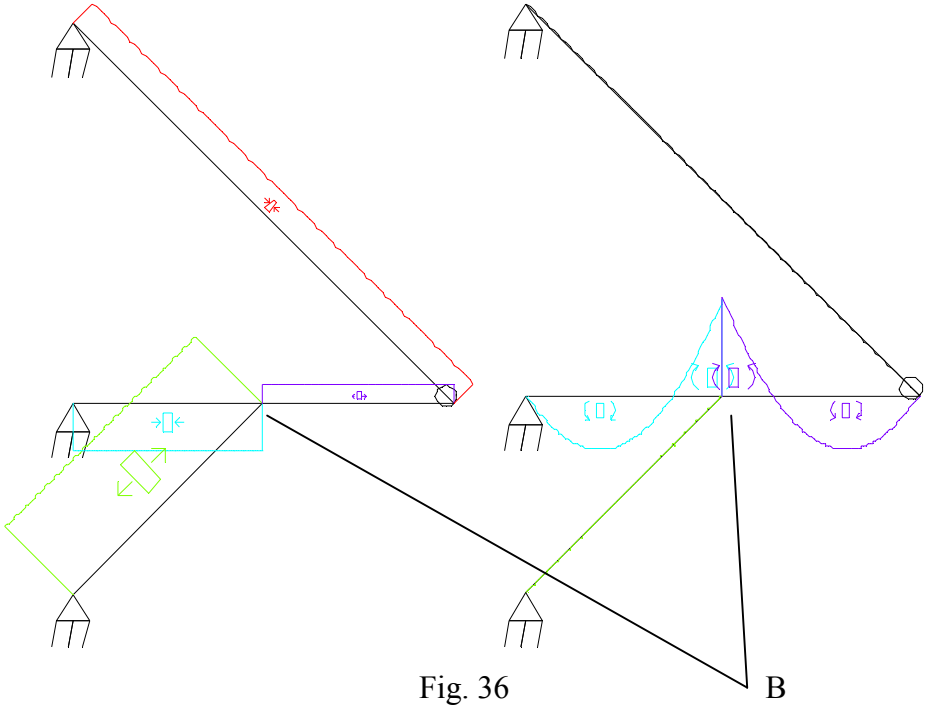


Fig. 36

Esta casuística se puede ver fácilmente en el nudo B de la Fig. 36, en el que por el lado izquierdo el axil es de compresión y por el lado derecho es de tracción. En este caso se vería claro que la sección más desfavorable es la sección B por la izquierda; sin embargo esto no sería tan trivial si ambos axiles fueran más o menos iguales y de sentido contrario la sección más desfavorable sería la de N<0. Por este motivo se ha tenido que usar una fórmula que nos sirva para paliar de forma sencilla estos problemas. La fórmula lo que hace es buscar que se cumpla una condición en valores absolutos. La condición para pórticos planos es:

$$N(Bi)>1.05N(Bd) \quad y \quad M(Bi)>0.99M(Bd)$$

donde N(Bi), N(Bd), M(Bi) y M(Bd) son los valores del axil y flector respectivamente a un lado y a otro de la sección.

El algoritmo de búsqueda de la sección más desfavorable consta de dos pasos:

1- Buscamos cual es la sección con mayor M y tomamos el valor de los tres esfuerzos en dicha sección.

2- Este paso depende de la iteración en la que estemos. Miramos las barras alrededor de ese nudo y si estamos en la primera itarción, se busca una que cumpla que al menos el valor del flector es el 99% del de la otra barra para corregir los posibles errores numéricos y que el valor del axil sea al menos en valor absoluto superior en un 5% al anterior. Si no estamos en la primera iteración, usando los perfiles calculados en la iteración anterior calculamos σ_{eq} y comparamos cual es la mayor.

Se ha tomado como margen de seguridad un 5% en el axil porque el axil suele ser menor que el flector, debido a que sus unidades de fuerza y el flector unidades.

En el caso de emparrillados, la condición impuesta es casi idéntica; la única diferencia radica en que al ser tanto el torsor como el flector del mismo orden se les ha dado el mismo margen de error:

$$T(Bi)>1.01T(Bd) \text{ y } M(Bi)>0.99M(Bd)$$

La comparación de σ_{eq} es más fiable, sin embargo, debido a los problemas derivados de la estimación de los perfiles en la primera iteración, se prefirió usar el primer método propuesto.

4.5.3 Introducción de datos.

El uso de esta aplicación requiere la introducción de una serie de datos de distinta índole, para ese propósito, Mathematica incorpora un comando, **Input[]**, pero este comando no trabaja adecuadamente por varias razones:

- La representación del requerimiento o pregunta es muy defectuosa, impidiendo el uso de símbolos y minúsculas, por ejemplo la letra “e” aparece como un punto negro como puede verse en Fig. 37.

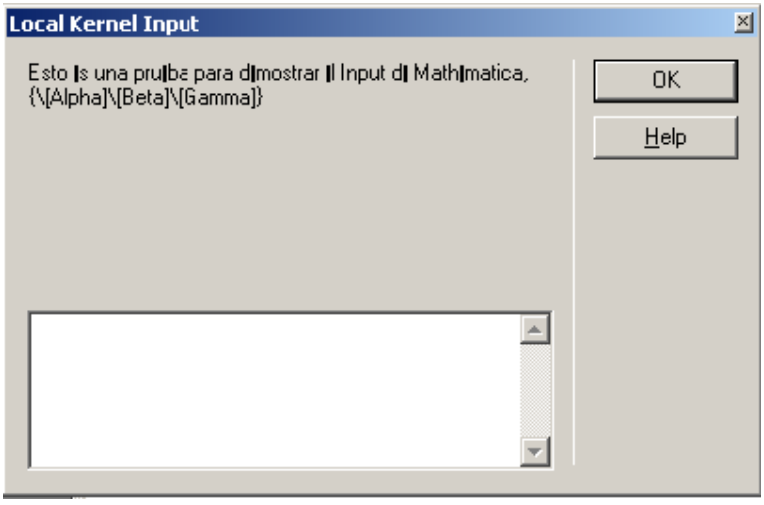


Fig.37

- No permite la manipulación de los datos de entrada. Debido a la forma de introducir los datos elegido, se ha considerado conveniente simplificar lo máximo posible la introducción de datos para evitar confusiones.

Por estos motivos se decidió programar dos comandos de entrada de datos que permitieran simplificar la reproducción y toma de datos, uno para información de carácter general y otro para el caso en que estemos trabajando con los perfiles (Fig. 38).

Ambos presentan características muy interesantes, el primero es un input general que tiene la particularidad de poder mostrar cualquier tipo de símbolo, ya que se ha usado una de las propiedades más importantes del FrontEnd, el uso de la gestión del espacio mediante "cajas". En el segundo, como podemos ver, además de aparecer botones como en el anterior, tenemos la posibilidad de ver qué son cada uno de ellos en la parte inferior de la pantalla simplemente colocando el puntero del ratón encima del botón y permite determinar el perfil elegido en la parte superior (donde pone visualiza opción elegida). Esto permite un uso muy sencillo por parte del usuario.

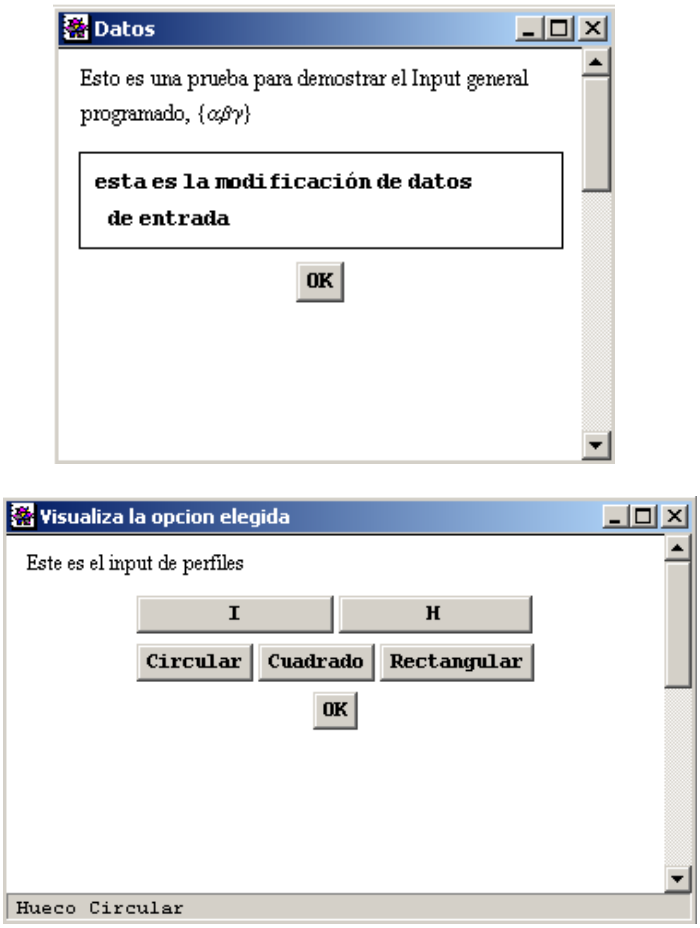


Fig. 38

4.6 Interfaz de usuario.

4.6.1 Introducción

Para conseguir simplificar el uso de la aplicación se ha programado una Interfaz de usuario. La idea que se persigue es muy similar a la comentada anteriormente cuando se hizo la introducción a Mathematica, donde se explicó que por un lado estaba el Kernel, y por otro lado el FrontEnd.

La interfaz incluye una serie de botones que permiten ejecutar las operaciones necesarias y a la vez incluye una serie de comentarios que ayudan al usuario a seguir la evolución natural de la resolución de un problema. Adicionalmente se genera un documento donde se desarrolla todo el problema, y en el que se escriben los resultados de los cálculos y se comentan algunos puntos de interés.

En la Fig. 39 podemos ver la relación que existe entre cada uno de los documentos o partes de la aplicación. En esta misma figura se puede ver que se ha incluido el Input como si fuera una parte exterior a la aplicación. Esto no es estrictamente cierto porque se encuentra dentro de las funciones programadas dentro de la aplicación; sin embargo, se ha decidido poner aparte porque no forma parte de la interfaz por si mismo, y tampoco pertenece a la aplicación como tal, ya que es un documento (Notebook) generado y con el que el usuario puede interactuar.

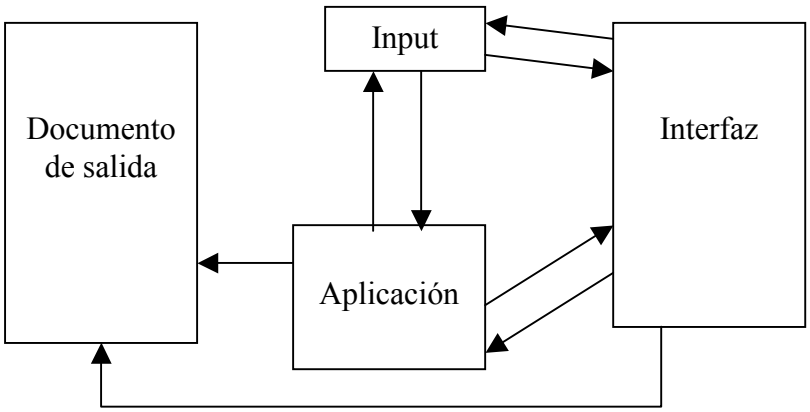
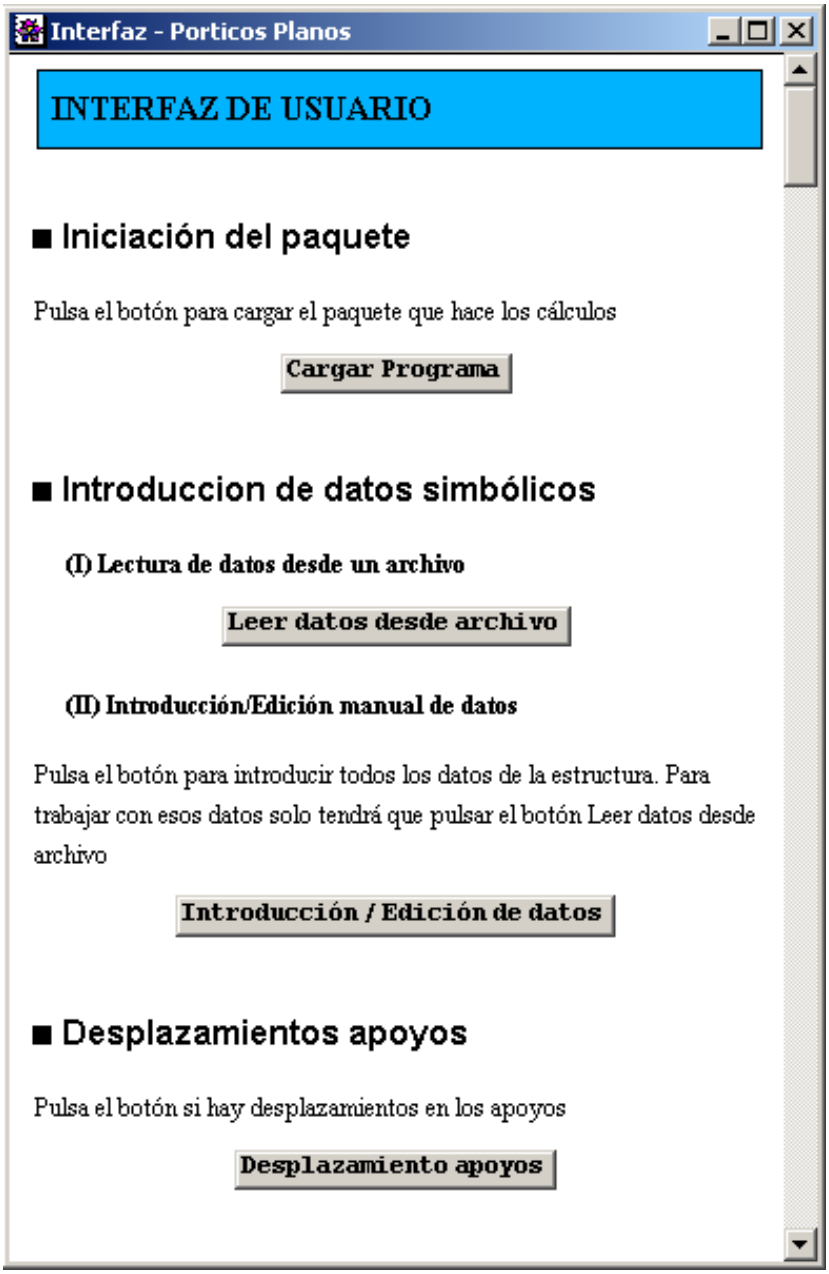


Fig. 39

4.6.2 Interfaz de usuario.

La interfaz de usuario es un documento de Mathematica que contiene información sobre la aplicación, así como acerca de los pasos necesarios para realizar un cálculo de una estructura por el Método de las Fuerzas. Este documento sirve de guía, no sólo ya para su uso, sino con fines didácticos, pues indica al usuario cada uno de los pasos que ha de seguir para resolver un problema. Así miso facilita la introducción de

los datos del problema de una manera gráfica y permite la generación de un documento compatible con Mathematica donde se escribe toda la información.



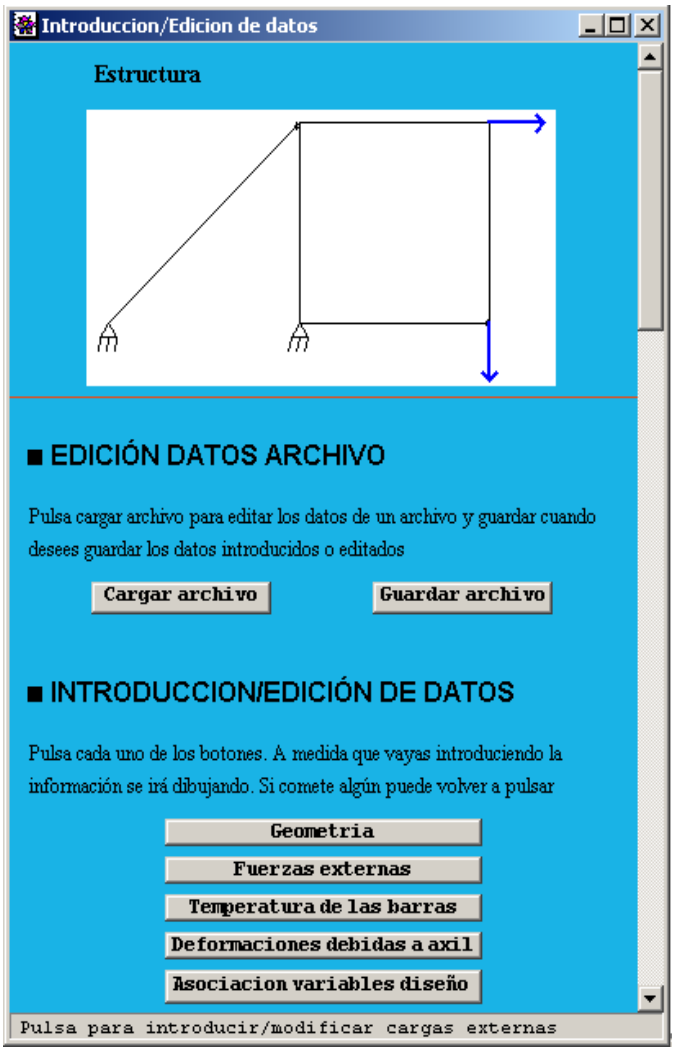
El siguiente dibujo muestra una parte de la interfaz de usuario. Como puede verse, es un documento normal de Mathematica con alguna información escrita y una serie de elementos activos, los botones. A medida que vamos realizando operaciones, lo único que tenemos que hacer es ir desplazándonos a lo largo del documento.

La interfaz está dividida en 4 partes perfectamente diferenciadas; esta primera parte que se muestra aquí está orientada a cargar el programa y a la introducción de datos. La introducción de datos se puede hace desde archivo ó manualmente (en cuyo caso lo que hacemos es generar el fichero que posteriormente leeremos).

Con respecto a la información, recodemos que el problema se divide en una parte simbólica de análisis y una parte numérica de diseño. La parte simbólica (y por tanto la introducción de sus datos) es obligatoria en todos los casos, no siendo así la numérica., cuyos datos son opcionales. Esta distinción obedece a que un caso simbólico da lugar a muchos estados diferentes de cargas, según los valores numéricos que introduzcamos. De cualquier manera, los datos numéricos pueden ser vueltos a editar o vueltos a introducir posteriormente.

Pulsando el botón de lectura de datos de archivo cargamos los datos del archivo y abrimos el documento de salida donde vamos a imprimir todos los resultados del problema y pulsando el botón “Introducción /Edición de datos”, abrimos una pantalla

donde podemos ir introduciendo poco a poco los datos del problema mientras visualizamos lo que estamos introduciendo / modificando.



La siguiente figura muestra una imagen del documento para introducir o modificar los datos. Es muy parecido a la interfaz y es generado de manera análoga a como se generan los inputs.

La segunda parte importante es la resolución analítica de la estructura. En esta parte están los botones que permiten la introducción de hiperestáticas (en caso de ser necesario), cálculo de reacciones, cálculo de leyes de esfuerzos, cálculo de hiperestáticas (en caso de ser necesario) y cálculo de desplazamientos.

La tercera parte de la interfaz es la correspondiente al diseño. En esta parte, está la introducción de valores numéricos, la introducción de los criterios de rigidez (en caso de ser necesario), las opciones para la modificación del criterio de elección de perfiles, y la posibilidad de realizar un cálculo automático donde el programa realizará todas las decisiones por sí mismo o un cálculo manual donde se irá pidiendo al usuario que elija él mismo los perfiles que estime adecuados.

La última parte del documento (interfaz), contiene la representación final de los resultados, es decir la representación de los diagramas de esfuerzos y de la deformada de la estructura, así como la posibilidad de conservar de forma impresa o en un archivo los resultados obtenidos en el proceso de cálculo.

4.6.3 Programación de la interfaz de usuario.

Para entender cómo se ha programado la interfaz, es importante aclarar algunos puntos sobre el funcionamiento de Mathematica:

Mathematica es un programa básicamente interpretativo, es decir que una vez introducimos un comando, Mathematica lo interpreta y lo ejecuta. Por este motivo, Mathematica guarda sus ficheros (Notebooks) como archivos de texto, que al ser cargados se vuelven a leer y a interpretar.

Otro concepto importante que comentamos anteriormente es la diferencia entre el FrontEnd y el Kernel de Mathematica. Anteriormente se comentó que cuando ejecutamos Mathematica, lo que hacemos es cargar el FrontEnd y que posteriormente, cuando intentamos evaluar una expresión, se establece una conexión vía Mathlink con el Kernel. También se comentó que al igual que el Kernel tiene sus comandos, el FrontEnd tiene los suyos propios, que interpreta como tales.

El último concepto importante que hay que comentar es cómo está estructurado un documento de Mathematica, aunque esto será conocido por la mayoría de usuarios de Mathematica. Los documentos de Mathematica están divididos en celdas o células (cell), cada una de las cuales tiene una serie de propiedades, y dentro de las que se guarda la información que estamos usando. Así, podemos establecer un bloque global donde tenemos un documento o Notebook, dentro de este documento tenemos células, dentro de estas células podemos tener otro grupo de células (agrupación de células), y dentro de cada célula una cantidad de información.

Una vez comentado todos estos aspectos, podemos decir que un documento ó Notebook es un comando para el FrontEnd de Mathematica que además de tener sus argumentos correspondientes, que en este caso serán el grupo de células y subcélulas que contenga, posee unas "opciones". Esta idea resulta muy conocida para los usuarios de Mathematica, donde muchas de las funciones usadas, tienen, aparte de los argumentos necesarios, una serie de opciones por defecto que serán más o menos útiles y que obviamente podemos cambiar en nuestro provecho. Esta misma idea será conocida para aquellas personas que trabajen con lenguajes de programación orientado a objetos como Visual Basic ó Visual C++, donde tenemos los Form y sus propiedades (tamaño, color del fondo, etc.). Juntando todo lo comentado podemos entender que hay dos maneras de crear un documento de Mathematica: usando el FrontEnd que nos facilita mucho determinadas tareas (como son por ejemplo el formateo de los objetos, la determinación de opciones por defecto de la mayoría de objetos, la presentación gráfica de una serie de comandos y en resumen simplificando la visualización de objetos mediante un entorno gráfico) o simplemente haciéndolo de forma directa con un editor de texto normal y corriente.

Para simplificar un poco la explicación, simplemente hay que acudir a la ayuda de Mathematica y buscar la palabra Notebook. La respuesta que obtendremos será algo parecido a:

Notebook[{cell1,cell2,....},opts], representa un Notebook que puede ser manipulado por el front end de Mathematica.

Si escribimos en un documento de Mathematica la frase "Esto es una célula de Mathematica" y posteriormente nuestra curiosidad nos hace abrir con un editor de textos normal (por ejemplo Notepad) el archivo que hemos generado con Mathematica, encontraremos, sorprendentemente, algo así: (ejemplo de un documento con una célula)

Notebook[{Cell[CellGroupData[{Cell[BoxData[\(Esto\ es\ una\ c\[EAcute]lula\ de\ Mathematica\)], "Input"],Cell[BoxData[\(c\[EAcute]lula\ de\ es\ Esto\ Mathematica\ una\)], "Output"]}, Open]]}

,FrontEndVersion->"Microsoft Windows 3.0",

ScreenRectangle->{{0, 1024}, {0, 712}},

WindowSize->{496, 604},

WindowMargins->{{50, Automatic}, {Automatic, 15}}

]

donde claramente podemos ver los comandos Notebook, CellGroupData, Cell, y BoxData, unos incluidos en otros; al final aparecen las opciones de cada uno. Conociendo estos comando y usando llaves (como todo en Mathematica), podemos construir con un editor de texto nuestro propio documento sin la necesidad de Mathematica.

La afirmación antes realizada es un poco extrema, ya que en estos documentos hay alguna información interna que usa el FrontEnd a la hora de interpretar los Notebooks, pero la explicación anteriormente dada nos da una idea no sólo de la organización interna de Mathematica, sino de su potencia.

Con todo lo hasta ahora comentado podemos plantear una de las formas a usar para crear nuestro interfaz: usando un editor de textos y conociendo los comandos necesarios podemos crear un documento donde podemos incluir objetos activos como son los botones, que sólo son una serie de objetos con unas propiedades y unos argumentos, entre los que están la posibilidad de ejecutar cierto código cada vez que son pinchados con el ratón.

Alternativamente, al igual que los lenguajes de programación orientada a objetos, podemos generar desde un documento en ejecución, es decir desde la sesión en la que estemos trabajando, otros objetos o documentos sin la necesidad de escribir en ellos directamente, lo que constituiría la segunda forma de generar la interfaz; mientras la primera sería una forma directa, esta segunda es una forma indirecta de generación de documentos.

Entre estas dos formas comentadas existen algunas diferencias: la forma directa es demasiado austera, requiere un conocimiento exhaustivo de los comandos y no permite corregir o detectar los posibles errores durante la programación; por contra la forma indirecta, tiene la ventaja de que podemos ir viendo como vamos generando poco a poco nuestro documento, podemos ir variando cosas que no nos gustan en ese momento y nos permite usar las facilidades (ya comentadas anteriormente) que ofrece el entorno de trabajo de Mathematica. Podemos además generar este documento tantas veces como queramos mientras conservemos el documento generador, que podría verse como el código fuente.

Por todos estos motivos, es fácil entender que tanto para la generación de la interfaz como para la generación del documento de impresión de resultados, se ha usado el método indirecto de generación de documentos.

A título informativo, es interesante decir que existe una tercera forma de generar la interfaz y es ir creando una por una las células, dándoles el color y forma deseados. En el caso de los botones, el Front End de Mathematica permite a los usuarios editar el

código a ejecutar desde el mismo Menú, permitiendo generar una serie de botones por defecto con un código determinado; sin embargo no se ha considerado que esta forma sea la más adecuada, sobre todo por los problemas asociados al control de los botones.

Un punto que considero importante aclarar es que Matemática permite la herencia entre objetos, lo que quiere decir que cuando dos objetos tengan una misma propiedad, por defecto el objeto de menor rango coge el valor del objeto de mayor rango en caso de no especificarse; sin embargo no siempre pasa eso, sobre todo con las propiedades derivadas del formateo de texto.

Por último sólo queda concluir que la programación de los Notebooks no es una tarea complicada, no así la de los botones, que pueden dar lugar a bastantes quebraderos de cabeza debido a las posibilidades que permiten. Así mismo, la información referente al lenguaje de programación del FrontEnd es bastante escasa y dispersa, complicando notablemente la programación de todos los documentos generados (Input's, interfaz de introducción de datos y de interacción con el usuario), así como la generación del documento de impresión de resultados [Véase referencias 8 y 9].

4.7. Listado de funciones programadas.

La metodología de programación, como se ha comentado anteriormente, ha sido funcional, así que con carácter general a continuación aparece una relación con las funciones programas, los argumentos que necesitan y los que devuelven, así como una breve explicación de lo que realizan. Estas funciones pueden ser llamadas por cualquier usuario de Mathematica siempre que cargue el paquete en cuestión.

4.7.1. Listado de funciones para pórticos planos.

Mientras no se indique lo contrario, las variables serán todas tipo lista, pudiendo ser estas listas tanto vectores simples como matrices. Los distintos tipos de variables usadas son los siguientes:

Integer: Entero.

Bool: Número booleano o binario (1 ó 0).

Real: Número real.

analisiscargas

Argumentos de entrada: analisiscargas[nombreperfilanterior, SMD, ivd_Integer, datosnumericos, it_Integer, diseñoautomatico_Bool]

Realiza un análisis de las cargas con las que estamos trabajando y realiza una decisión y una toma de datos si el proceso no fuera automático. Devuelve el

resultado de esta elección y un indicativo del estado de cargas al que está sometido

Argumentos de salida: {nombreperfiladecuado, caso}

calculodeformada

Argumentos de entrada: calculodeformada[numerobarras_Integer, numeronodos_Integer, unudos, libertadesbarras, conecbarras, vecAreas, vecInercias, datosnumericos, esfuerzos, coordenadanodos, angulobarra, clk]

Esta función genera los elementos geométricos necesarios para poder dibujar la deformada de la estructura.

Argumentos de salida: deformada

calculoesfuerzos:

Argumentos de entrada: calculoesfuerzos[clk, invtQ, EQ0, invtH, vechip, simbolof, EQm1, FQ, numerobarras_Integer, cfbarras, hip]

Calcula los esfuerzos en extremos de barras y a partir de ellos y de las cargas en barras calcula las leyes de esfuerzos en las barras. Devuelve las leyes de esfuerzos y el valor de los esfuerzos en los extremos de barras

Argumentos de salida: {valorQ, Q}"

calculoreacciones

Argumentos de entrada: calculoreacciones[EQR0, EQRm1, invtH, vechip, simbolof, FQR, numerobarras_Integer, nreacciones_Integer, nodos, hip]

Genera y devuelve el vector simbólico de las reacciones en los apoyos

Argumentos de salida: valorR

calculotensiones

Argumentos de entrada: calculotensiones[SMD, ivd_Integer, datosnumericos, clk, perfiloptimo, omega, numerocarasverticales_Integer, visible_Bool]

Realiza el cálculo de tensiones usando el perfil indicado. Devuelve una serie de resultados en forma de lista.

Argumentos de salida: {longitudpandeo, lambda, tnormalaxil, tnormalflector, tangencial, tequivalente}

cambioserie

Argumentos de entrada: cambioserie[puntero_Integer, tablaperfiles]

Devuelve la posición del primer y último perfil de la siguiente serie de la tabla donde nos encontremos

comprobacion2

Argumentos de entrada: comprobacion2[nvariablesdiseño_Integer, perfiles, valordesp, desp, cotas, hiperestaticas, perfilipe, perfilcuadrado, perfiltubular, perfilcajon, perfilheb, nombreperfil, id_Integer]

Hace el segundo proceso iterativo en el redimensionado por rigidez. Devuelve el perfil óptimo encontrado

comprobacion3

Argumentos de entrada: comprobacion3[nvariablesdiseño_Integer, perfiles, valordesp, desp, cotas, hiperestaticas, perfilipe, perfilcuadrado, perfiltubular, perfilcajon, perfilheb, nombreperfil, indicadorvardiseno, clk]

Hace el tercer proceso iterativo en el redimensionado por rigidez. Devuelve el perfil óptimo encontrado

comprobacionhiperestaticas

Argumentos de entrada: comprobacionhipertaticas[clavehipint, clavehipext, EQR, nodos, numerobarras, nreacciones_Integer, numhip_Integer]

Genera el vector simbólico asociado a las incógnitas hiperestáticas y genera la matriz HEQR y comprueba que el conjunto de hiperestáticas elegidas es adecuado. Devuelve una lista con la matriz HEQR, el vector de hiperestáticas y el índice de comprobación.

Argumentos de salida: {vechip, HEQR, indicecomprobación}

comprobacionrigidez

Argumentos de entrada: comprobacionrigidez[nvariablesdiseño_Integer, perfiles, valordesp, desp, cotas, hiperestaticas, perfilipe, perfilcuadrado, perfiltubular, perfilcajon, perfilheb, tipoperfil, indicadorvardiseno, clk]

Esta función hace la comprobación si es necesario de los criterios de rigidez y si no cumple comienza realiza el proceso de redimensionado. Devuelve el perfil óptimo obtenido

determinarSMD

Argumentos de entrada: determinarSMD[nvariablesdiseño_Integer, numerobarras_Integer, esfuerzos, indicadorvardiseño, datosnumericos, clk, conecbarras, perfiloptimoanterior, nombreperfilanterior]

Esta función determina cual es la sección más desfavorable para cada grupo de diseño. Devuelve una matriz con los valores del axil, cortante y flector de las sección más desfavorable y posición en la que se encuentra, barra y posición.

Argumentos de salida: {SMD, posicion}

diagramasfuerzas

Argumentos de entrada: diagramasfuerzas[simbolof, coordenadanodos, conecbarras, Q, valorQ, vechip, hip_Integer, clk, tamañounitario_Real, angulobarra]

Modifica los diagramas de esfuerzos anulando la parte relacionada con las hiperestáticas adimensionalizando por las cargas externas para poder representar los diagramas de esfuerzos adimensionales según cargas. Devuelve una lista con las entidades geométricas necesarias para poder dibujar los diagramas.

Argumentos de salida: {axil, terminacionesaxil, cortante, terminacionescortante, flector, terminaciones flector, iconoaxil, iconocortante, iconoflector}

diagramashiperestaticas

Argumentos de entrada: diagramashiperetaticas[simbolof, coordenadanodos, conecbarras, Q, valorQ, vechip, hip_Integer, clk, tamañounitario_Real, angulobarra, ihip_Integer]

Modifica los diagramas de esfuerzos anulando la parte relacionada con las cargas externas y adimensionalizando por cada hiperestática para poder representar los diagramas de esfuerzos adimensionales según las hiperestáticas. Devuelve una lista con las entidades geométricas necesarias para poder dibujar los diagramas.

Argumentos de salida: {axil, terminacionesaxil, cortante, terminacionescortante, flector, terminaciones flector, iconoaxil, iconocortante, iconoflector}

dibujardiagramas

Argumentos de entrada: dibujardiagramas[axil, terminacionesaxil, cortante, terminacionescortante, flector, terminacionesflector, dibujobarras, iconoaxil, iconocortante, iconoflector]

Esta función dibuja en un Notebook llamado nbsalida los diagramas de esfuerzos. Sus argumentos son las entidades geométricas necesarias para ello.

diseno

Argumentos de entrada: diseno[valorQ, valorH, vechip, vecAreas, vecInercias, datosnumericos, indicadorvardiseño, numerobarras_Integer, clk, temperaturas, vecTemperaturas, desp, valordesp, cotas, conecbarras, muchomayor_Real, muchomenor_Real, diseñoautomatico_Bool, valordespapoyos, vecdespapoyos, evaluahiperestaticas]

Esta función es la que controla todo el proceso de diseño. Devuelve una matriz con las leyes de esfuerzos finales y los perfiles óptimos conseguidos.

Argumentos de salida: {esfuerzos, Perfiloptimo}

dnodos

Argumentos de entrada: dnodos[numeronodos_Integer, Qdespl, clk, vecAreas, vecInercias, numerobarras_Integer, coordenadanodos, conecbarras, nodos, libertadesbarras, gt_Integer, gr_Integer, nreacciones_Integer, nlibertades_Integer, invtQ, EQm1, vecTemperaturas, datosnumericos, vecdespapoyos, EQR, indicadorvardiseño, simbolof, cfbarras]

Esta función busca cuanto valen los desplazamientos de todos los nudos y libertades y devuelve dos listas, una con los símbolos correspondientes a cada uno de los desplazamientos y otra con los valores correspondientes.

Argumentos de salida: {simbolicou, valoru}

equilibrioreducidas

Argumentos de entrada: equilibrioreducidas[barrascortadas, ERQc, FRQc, RQc, simbolom]

Genera las matrices de equilibrio reducidas y las imprime en un Notebook llamado nbsalida

equilibrioredudicassegunhip

Argumentos de entrada: equilibrioredudicassegunhip[ERQc, RQc, vechip, hip_Integer, simbolom]

Genera las matrices de equilibrio reducidas y las imprime en un Notebook llamado nbsalida

generacionEQR

Argumentos de entrada: generacionEQR[coordenadanodos, conecbarras, nodos, libertadesbarras, cfnudos, cfbarras, cflibertades, clk, cosenobarras, senobarras, hip_Integer, gt_Integer, gr_Integer, nreacciones_Integer, nlibertades_Integer]

Crea la matriz de equilibrio global de la estructura EQR.

Argumentos de salida: {EQR}

generacionFQR

Argumentos de entrada: generacionFQR[coordenadanodos, conecbarras, nodos, libertadesbarras, cfnudos, cfbarras, cflibertades, clk, cosenobarras, senobarras, hip_Integer, gt_Integer, gr_Integer, nreacciones_Integer, nlibertades_Integer]

Crea el vector de fuerzas aplicadas en los nudos de la estructura FQR

Argumentos de salida: {FQR}

generacionSENOCOSLONG

Argumentos de entrada: generacionSENOCOSLONG[coordenadanodos, conecbarras]

Genera los vectores de senos y cosenos de cada barra, la longitud de cada barra y el vector de ángulos de cada barra.

Argumentos de salida: {senobarras, cosenobarras, clk, angulobarra}

generadordibujodiagramas

Argumentos de entrada: generadordibujodiagramas[numerobarras_Integer, diagramas, factorescala_Real, coordenadanodos, conecbarras, angulobarra]

Esta función genera todos los elementos gráficos necesarios para representar los diagramas de esfuerzos y los devuelve en forma de listas.

Argumentos de salida: {axil, terminacionesaxil, cortante, terminacionescortante, flector, terminacionesflector, iconoaxil, iconocortante, iconoflector}

generaregla

Argumentos de entrada: generaregla[datosperfil]

Genera una regla de sustitución

gradohiperestaticidad

Argumentos de entrada: gradohiperestaticidad[conecbarras, nodos, libertadesbarras]

Calcula una serie de parámetros básicos y los devuelve en forma de lista.

Argumentos de salida: {número de hiperestáticas(hip), número de cortes(ncortes), número de grados de libertad translacionales(gt), número de grados de libertad rotacionales(gr), número de reacciones(nreacciones), número de libertades(nlibertades)}

grafico

Argumentos de entrada: grafico[coordenadanodos, conecbarras, nodos, libertadesbarras, temperaturas]

Genera el dibujo de la estructura y determina un factor de escala llamado tamañounitario. Devuelve una lista con las entidades geométricas y dicho valor.

Argumentos de salida: {dibujo, tamañounitario}

graficofuerzas

Argumentos de entrada: graficofuerzas[cfnudos, cfbarras, cflibertades, coordenadanodos, conecbarras, tamanounitario_Real, angulobarra]

Genera las entidades geométricas necesarias para dibujar las fuerzas en la estructura. Devuelve una lista con estas entidades geométricas.

Argumentos de salida: {dibujonudos,cargatransversal,cargalongitudinal}

Inputperfiles

Argumentos de entrada: Inputperfiles[prompt_String, texto]

Genera el input cuando es necesario pedir al usuario que elija un perfil. Devuelve el resultado en forma de cadena.

Inputpersonal

Argumentos de entrada: Inputpersonal[prompt_String, texto]

Abre una ventana en la que tenemos que la pregunta o texto de petición de datos es prompt y texto es el texto que aparece por defecto en el cajetín donde se introducen los datos. Devuelve lo que esté en el cajetín de texto

introducedatos

Argumentos de entrada: introducedatos[]

Esta función genera la interfaz necesaria para poder introducir editar todos los datos de un problema. No devuelve nada y lo guarda en un archivo.

iteracion

Argumentos de entrada: iteracion[nvariablesdiseño_Integer, numerobarras_Integer, esfuerzos, indicadorvardiseño, datosnumericos, clk, desp, valordesp, cotas, conecbarras, muchomayor_Real, muchomenor_Real, diseñoautomatico_Bool, it_Integer, nombrefilanterior, hiperestaticas, iteracionsimple_Bool, perfiloptimoanterior]

Esta función es la que controla todos los procesos necesarios para realizar una iteración completa. Devuelve la sección más desfavorable de esa iteración, los perfiles óptimos encontrados, las posiciones de las secciones más desfavorable y los nombres de los perfiles óptimos.

Argumentos de salida: {SMD, perfiloptimo, posiciones, nombreperfil}

iteracionaxilcompresionabtos

Argumentos de entrada: iteracionaxilcompresionabtos[SMD, clk, tablaperfiles, datosnumericos, ivd_Integer, nombreperfil_String]

Busca el perfil óptimo cuando el caso sea de un perfil abierto sometido a axil de compresión. Devuelve el perfil óptimo

iteracionaxilcompresioncerrados

Argumentos de entrada: "iteracionaxilcompresioncerrados[SMD, clk, tablaperfiles, datosnumericos, ivd_Integer, nombreperfil_String]

Busca el perfil óptimo cuando el caso sea de un perfil cerrado sometido a axil de compresión. Devuelve el perfil óptimo

iteracionaxiltraccion

Argumentos de entrada: iteracionaxiltraccion[tablaperfiles, SMD, datosnumericos, ivd_Integer]

Busca en el caso de axil a tracción el perfil más adecuado dentro de la tabla indicada. Devuelve el perfil óptimo

iteracionflector

Argumentos de entrada: iteracionflector[SMD, ivd_Integer, tablaperfiles, datosnumericos, nombreperfil]

Busca en el caso de flexión el perfil más adecuado dentro de la tabla indicada. Devuelve el perfil óptimo.

iteraciongeneral

Argumentos de entrada: iteraciongeneral[SMD, ivd_Integer, tablaperfiles, datosnumericos, clk, nombreperfil_String, diseñoautomatico_Bool, muchomayor_Real, muchomenor_Real, perfilipe, perfiltubular, perfilcuadrado, perfilcajon, perfilheb, vezllamada_Bool]

Busca el perfil óptimo para el caso general en el que tengamos como mínimo axil y flector. Devuelve el perfil óptimo y su nombre.

Argumentos de salida: {Perfiloptimo, nombreperfil}

iteracionunica

Argumentos de entrada: iteracionunica[muchomayor_Real, muchomenor_Real, datosnumericos]

Permite realizar una iteración usando los valor de axil, cortante y flector deseados.

leedatos

Argumentos de entrada: leedatos[nombrearchivo_String]

Lee los datos del archivo cuyo nombre indiquemos. Si el archivo es un archivo de datos adecuado devolverá una lista de variables.

Argumentos de salida: {coordenadanodos, conecbarras, nodos, libertadesbarras, cfnudos, cfbarras, cflibertades, indicadordefaxiles, indicadorvardiseño, temperaturas, datosnumericos}

matricesEQR0yEQRm1

Argumentos de entrada: matricesEQR0yEQRm1[HEQR, numerobarras_Integer, nreacciones_Integer, hip_Integer, gt_Integer,gr_Integer]

Genera y devuelve las matrices EQ0, EQR0, EQm1, EQRm1 y FQ necesarias para el cálculo de reacciones y esfuerzos en extremos de barras.

Argumentos de salida: {EQ0, EQR0, EQm1, EQRm1, FQ}

matrixERQc

Argumentos de entrada: matrixERQc[colRQc, EQR, FQR, numerobarras_Integer, nreacciones_Integer, nlibertades_Integer, numeronudos_Integer]

Genera y devuelve la matriz ERQc y el vector FRQc necesarios para las matrices de equilibrio reducidas.

nbperfiles

Argumentos de entrada: nbperfiles[prompt_String, texto_String]

Genera la pantalla de elección de perfiles

nbpersonal

Argumentos de entrada: nbpersonal[prompt_String, texto_String]

Genera la ventana usada por Inputpersonal

nombrecompletoperfil

Argumentos de entrada: nombrecompletoperfil[nombreperfil_String, perfil]

Esta función genera y devuelve una cadena de caracteres con el nombre de un perfil de la manera habitualmente usada

numeracionnudosbarras

Argumentos de entrada: numeracionnudosbarras[coordenadanodos, tamanounitario_Real, angulobarra, conecbarras]

Numera los nudos, barras y les da el sentido a las barras. Devuelve una lista con las entidades geométricas generadas.

permutacioncolumnas

Argumentos de entrada: permutacioncolumnas[matriz, col1_Integer, col2_Integer]

Esta matriz permuta las columnas indicadas en 1 y 2 de la matriz indicada. Devuelve la matriz permutada.

PFV

Argumentos de entrada: PFV[informacion, valorQ, clk, vecAreas, vecInercias, numerobarras_Integer, vecTemperaturas, valorR, vecdespapoyos,]

Esta rutina aplica el PFV para el cálculo de hiperestáticas. Devuelve las ecuaciones que salen de la aplicación de PFV.

simbolicoapoyos

Argumentos de entrada: simbolicoapoyos[despapoyos, nreacciones_Integer, nodos]

Genera y devuelve un vector con los símbolos asociados a los desplazamientos de los apoyos.

simbolicoinerciasareas

Argumentos de entrada: simbolicoinerciasareas[indicadordefaxiles, indicadorvardiseño, temperaturas]

Crea dos vectores simbólicos, uno con las inercias de cada barra y otro con las áreas de cada barra.

Argumentos de salida: {vecInercias,vecAreas}

simbolo

Argumentos de entrada: simbolo[cfbarras, numerobarras_Integer]

Crea los símbolos usados para indicar las fuerzas y momentos dependiendo de si hay barras cargadas o no y los devuelve n forma de lista.

Argumentos de salida: {simbolof, simbolom}

transformacionH

Argumentos de entrada: transformacionH[hip_Integer, clavehipext, calvehipint]

Genera y devuelve la inversa de la matriz de transformación asociada a las hiperestáticas para el cálculo de las reacciones y esfuerzos en extremos de barras

transformacionQ

Argumentos de entrada: transformacionQ[numerobarras_Integer, clk]

Genera y devuelve la inversa de la matriz de transformación asociada a los esfuerzos en extremos de barras para el cálculo de los esfuerzos en los extremos de barras.

vectorQc

Argumentos de entrada: vectorQc[ncortes_Integer, numerobarras_Integer, barrascortadas, indQcreducido_Bool]

Genera el vector simbólico donde tenemos las componenentes de los esfuerzos de las barra abiertas. Devuelve un vector con esas componentes y un vector con los número de las columnas de la matriz EQR asociadas a esos esfuerzos.

Argumentos de salida: {Qc, colQc}

vectorsimbolicoR

Argumentos de entrada: vectorsimbolicoR[nreacciones_Integer, nodos, numerobarras_Integer]

Genera y devuelve dos listas, una donde están los valores simbólicos de las reacciones y otra donde está el número de las columnas de las matriz EQR asociadas a las reacciones

Argumentos de salida: {R,colR}

desplazamientosmatricial

Argumentos de entrada: desplazamientosmatricial[numerobarras_Integer, clk, indicadorvardiseno, cfbarras, vecAreas, vecInercias, temperaturas, gt_Integer, gr_Integer, numeronodos_Inteer, nlibertades_Integer, nreacciones_Integer, vecdespapoyos, EQm1, EQR, invtQ, Qdespl, simbolof, largobarra, nodos, libertadesbarras, conecbarras]

Calcula los desplazamientos de todos los nudos de la estructura mediante un cálculo matricial. Devuelve un vector simbólico donde tenemos los símbolos de los desplazamientos y otro con los valores de éstos.

Argumentos de salida: {desplazamientos, valordesplazamientos}

matrizmCydq

Argumentos de entrada: matrizmCydq[numerobarras_Integer, clk, indicadorvardiseno, cfbarras, vecAreas, vecInercias, temperaturas]

Genera la matriz de compatibilidad mC y los vectores de deformaciones debidas a las cargas externas (fuerzas y temperatura).

Argumentos de salida: {mC,dq,dt}

4.7.2. Listado de funciones para emparrillados.

Las funciones usadas en emparrillados son básicamente iguales, las únicas diferencias con las mostradas para pórticos están en algunas funciones de iteración nuevas, y en la función de cálculo de tensiones que tiene diferentes argumentos de salida, debido obviamente a que las tensiones con las que estamos trabajando son diferentes. La siguiente relación muestra las funciones diferentes con sus respectivos argumentos de entrada y salida, así como una breve explicación de su funcionamiento.

comprobacion3

Argumentos de entrada: comprobacion3[nvariablesdiseño_Integer, perfiles, valordesp, desp, cotas, hiperestaticas, perfilipe, perfilcuadrado, perfiltubular, perfilcajon, perfilheb, nombreperfil, id_Integer]

Hace el tercer proceso iterativo en el redimensionado por rigidez. Devuelve el perfil óptimo encontrado.

grafico

Argumentos de entrada: grafico[coordenadanodos, conecbarras, nodos, libertadesbarras]

Genera el dibujo de la estructura y determina un factor de escala llamado tamañounitario. Devuelve una lista con las entidades geométricas y dicho valor.

Argumentos de salida: {dibujo, tamañounitario}

leedatos

Argumentos de entrada: leedatos[nombrearchivo]

Lee los datos del archivo cuyo nombre indiquemos. Si el archivo es un archivo de datos adecuado devolverá una lista de variables.

Argumentos de salida: {coordenadanodos, conecbarras, nodos, libertadesbarras, cfnudos, cfbarras, cflibertades, indicadorvardiseño, datosnumericos}

simbolicoinerciaJs

Argumentos de entrada: simbolicoinerciaJs[indicadorvardiseño]

Crea dos vectores simbólicos, uno con las inercias de cada barra y otro con las áreas de cada barra.

Argumentos de salida: {vecInercias,vecJs}

graficofuerzas

Argumentos de entrada: graficofuerzas[cfnudos, cfbarras, cflibertades, coordenadanodos, conecbarras, tamanounitario_Real, angulobarra]

Genera las entidades geométricas necesarias para dibujar las fuerzas en la estructura. Devuelve una lista con estas entidades geométricas.

Argumentos de salida: {dibujonudos,cargatransversal}

dnodos

Argumentos de entrada: dnodos[numeronodos_Integer, Qdespl, clk, vecJs, vecInercias, numerobarras_Integer, coordenadanodos, conecbarras, nodos, libertadesbarras, angulobarra, gt_Integer, gr_Integer, nreacciones_Integer, nlibertades_Integer, invtQ, EQm1, datosnumericos, vecdespapoyos, EQR, indicadorvardiseño, simbolof, cfbarras]

Esta función busca cuanto valen los desplazamientos de todos los nudos y libertades y devuelve dos listas, una con los símbolos correspondientes a cada uno de los desplazamientos y otra con los valores correspondientes.

Argumentos de salida: {simbolicou, valoru}

diseno

Argumentos de entrada: diseno[valorQ, valorH, vechip, vecJs, vecInercias, datosnumericos, indicadorvardiseño, numerobarras, clk, desp, valordesp, cotas, conecbarras, muchomayor_Real, muchomenor_Real, diseñoautomatico_Bool, valordespapoyos, vecdespapoyos, evaluahiperestaticas]

Esta función es la que controla todo el proceso de diseño. Devuelve una matriz con las leyes de esfuerzos finales y los perfiles óptimos conseguidos.

Argumentos de salida: {esfuerzos, Perfiloptimo}

calculotensiones

Argumentos de entrada: calculotensiones[SMD, ivd_Integer, datosnumericos, clk, perfiloptimo, omega, numerocarasverticales_Integer, visible_Bool]

Realiza el cálculo de tensiones usando el perfil indicado. Devuelve una serie de resultados en forma de lista.

Argumentos de salida: {tnormalflector, tangencial, tangencialtorsor tequivalente}

iteraciontorsion

Argumentos de entrada: iteraciontorsion[SMD, clk, tablaperfiles, datosnumericos, ivd_Integer, nombreperfil_String]

Busca el perfil óptimo cuando el estado de cargas sea de torsión pura. Devuelve el perfil óptimo.

PFV

Argumentos de entrada: PFV[informacion, valorQ, clk, vecJs, vecInercias, numerobarras_Integer, valorR, vecdespapoyos]

Esta rutina aplica el PFV para el cálculo de hiperestáticas. Devuelve las ecuaciones que salen de la aplicación de PFV.

calculodeformada

Argumentos de entrada: calculodeformada[numerobarras_Integer, numeronodos_Integer, unudos, libertadesbarras, conecbarras, vecJs, vecInercias, datosnumericos, esfuerzos, coordenadanodos, angulobarra, clk]

Esta función genera los elementos geométricos necesarios para poder dibujar la deformada de la estructura.

Argumentos de salida: { deformada}

cuadrado

Argumentos de entrada: cuadrado[coordenadas, angulo_Real, tamañounitario_Real]

Esta función permite generar un elemento gráfico en las coordenadas indicadas girado el ángulo indicado y de tamaño el indicado por tamañounitario.

matrizdegiro

Argumentos de entrada: matrizdegiro[angulo_Real]

Esta función permite generar una matriz de giro en el plano x-y

punta

Argumentos de entrada: punta[origen, angulogiro_Real tamañounitariofuerza_Real]

Esta función permite generar un elemento gráfico en forma de punta según el lugar indicado, tamaño inidicado y giro indicado.

momentodoble

Argumentos de entrada: momentodoble[origen, longitud_Real, angulo_Real, tamañounitariofuerza_Real]

Esta función permite generar un elemento gráfico en forma de flecha con doble punta (momento) en el lugar indicado, formando un ángulo con la horizontal

indicado, con una longitud adecuada y con la propoción indicada en tamañounitariofuerza.

flecha

Argumetos de entrada: flecha[origen, longitud_Real, angulo_Real, tamañounitariofuerza_Real]

Esta función permite generar un elemento gráfico en forma de flecha simple en el lugar indicado, formando un ángulo con la horizontal indicado, con una longituda adecuada y con la proporción indicada en tamañounitariofuerza.

vertical

Argumentos de entrada: vertical[origen, signo_Real, tamañounitario_Real]

Esta función permite generar un elemento gráfico en forma de punto o de aspa (X) dependiendo del signo que tenga la fuerza en le lugar indicado y con el tamaño indicado.

comprobacionrigidez

Argumentos de entrada: comprobacionrigidez[nvariablesdiseño_Integer, perfiles, valordesp, desp, cotas, hiperestaticas, perfilipe, perfilcuadrado, perfiltubular, perfilcajon, perfilheb, tipoperfil]

Esta función hace la comprobación si es necesario de los criterios de rigidez y si no cumple comienza realiza el proceso de redimensionado. Devuelve el perfil óptimo obtenido.

graficofuerzas

Argumentos de entrada: graficofuerzas[cfnudos, cfbarras, cflibertades, coordenadanodos, conecbarras, tamanounitario_Real, angulobarra, esreaccion]

Genera las entidades geométricas necesarias para dibujar las fuerzas en la estructura. Devuelve una lista con estas entidades geométricas.

Argumentos de salida: {dibujonudos,cargatransversal}

desplazamientosmatricial

Argumentos de entrada: desplazamientosmatricial[numerobarras_Integer, clk, indicadorvardiseno, cfbarras, vecJs, vecInercias, temperaturas, gt_Integer, gr_Integer, numeronodos_Inteer, nlibertades_Integer, nreacciones_Integer, vecdespapoyos, EQm1, EQR, invtQ, Qdespl, simbolof, largobarra, nodos, libertadesbarras, conecbarras]

Calcula los desplazamientos de todos los nudos de la estructura mediante un cálculo matricial. Devuelve un vector simbólico donde tenemos los símbolos de los desplazamientos y otro con los valores de éstos.

Argumentos de salida: {desplazamientos, valordesplazamientos}

matrizmCydq

Argumentos de entrada: matrizmCydq[numerobarras_Integer, clk, indicadorvardiseno, cfbarras, vecJs, vecInercias, temperaturas]

Genera la matriz de compatibilidad mC y los vectores de deformaciones debidas a las cargas externas (fuerzas y temperatura).

Argumentos de salida: {mC,dq,dt}

5-. ENTRADA Y SALIDA DE DATOS. SISTEMA DE ARCHIVOS

Mathematica trabaja de forma muy eficiente con las listas, es decir conjuntos de elementos (caracteres alfanuméricos,otras listas (listas anidadas), etc.) entre llaves; por ese motivo se ha trabajado con listas en esta aplicación. También se comentó en el apartado 4.5.3 que se han programado unos inputs personales que nos permiten un mayor campo de movimiento o capacidad para poder tratar la información. En este apartado intentaremos dar una explicación a cada una de las pantallas usadas para introducir datos y cómo interactuar con ellas, así como explicar las diferentes formas de introducir datos.

Las aplicaciones, necesitan 4 tipos de datos:

- Datos de los perfiles
- Datos de la estructura (geométricos y cargas aplicadas)
- Datos referentes al tipo de material (Isótropo-Lineal siempre) y valores numéricos de la estructura.
- Datos de interacción. Se llaman datos de interacción aquellos que sólo son necesitados durante la ejecución del programa y que sólo se introducen durante la ejecución de este. Estos datos siempre se introducirán vía consola.

A excepción de la información necesaria durante la interacción del programa con el usuario, toda la información necesaria que necesita el programa la puede obtener de archivos. Entre los archivos podemos también diferenciar dos tipos:

1. Archivos que necesita el programa para funcionar. Entre estos archivos hay dos tipos que son los que tienen las características geométricas de los perfiles y un archivo donde se guardan los coeficientes de pandeo. Los nombres de los archivos son:
    - PerfilIPE.dat. Contiene las características geométricas de los perfiles IPE.
    - PerfilHEB.dat. Contiene las características geométricas de los perfiles HEB.
    - Perfiltubular.dat. Contiene las características geométricas de los perfiles hueco circulares.
    - Perfilcuadrado.dat. Contiene las características geométricas de los perfiles huecos cuadrados.
    - Perfilcajon.dat. Contiene las características geométricas de los perfiles huecos rectangulares.

- Coeficientepandeo.dat. Contiene la tabla que relaciona ω y λ para el tipo de acero elegido. La tabla que se usa por defecto es la del acero A-42.

La forma en que están introducidos los datos para las tablas de perfiles es común para todos y están en el siguiente orden:

Alto (h) , ancho (b), espesor1,espersor2, A, S_x, I_x, W_x, i_x, i_y, J

2. Archivos que genera o puede generar el propio usuario y que se clasifican también en dos tipos: aquellos que contienen los resultados del problema y aquellos que contienen información geométrica-dinámica y del material, es decir la información necesaria para poder resolver el problema. Estos archivos pueden ser generados tanto manualmente, es decir con un procesador de textos, como mediante consola presionando el botón **Introducción /Edición de datos** que aparece en la interfaz de usuario. Se recomienda que estos archivos no tengan ninguna extensión, ya que suelen dar problemas.

5.1-. Entrada de datos por archivo.

Como se ha comentado anteriormente, los datos necesarios pueden introducirse vía archivo o vía consola. En este último caso se genera un archivo que nos permite guardar estos datos y posteriormente lo volvemos a introducir en el programa leyendo del mismo.

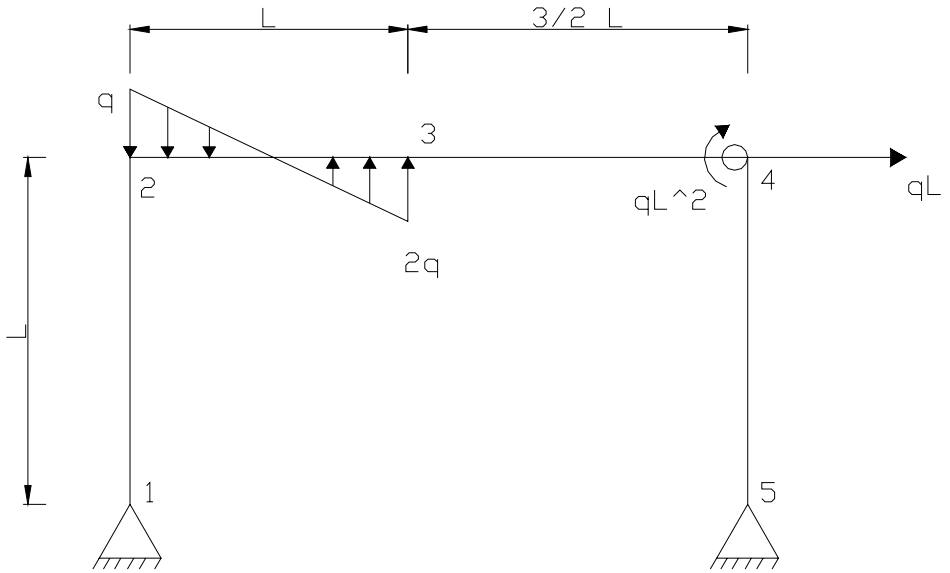


Fig. 40

Según el tipo de estructura que tengamos, los datos que tenemos que introducir son diferentes y por lo tanto los archivos diferentes. Los datos de entrada se dan mediante factores numéricos adimensionales que multiplican a determinadas funciones analíticas de las variables q(fuerza/longitud) ó F(fuerza), L(longitud) y T (ºC). La descripción de los datos de entrada y, como ilustración, su valor en el caso del pórtico plano de la Fig. 40, se detalla a continuación:

Parámetros generales:

b: Número de barras de la estructura.

n: Número de nudos de la estructura

- *Coordenadas de nudos (matriz n x 3): Constantes que multiplicadas por L dan las coordenadas de los nudos de la estructura.*

  {{0,0},{0,1},{1,1},{1,2.5},{1,0}}

- *Conectividad entre las barras (matriz b x 2): Nudos origen y destino de cada barra.*

  {{1,2},{2,3},{3,4},{4,5}}

- *Libertades en los nudos(matriz n x 3): Valores binarios (0 ó 1) que indican si el desplazamiento del nudo en cuestión en la dirección indicada, está permitido o no.*

  {{0,0,1},{1,1,1},{1,1,1},{1,1,1},{0,0,1}}

- *Libertades en los extremos de barras (matriz b x 2): Hacen referencia a la existencia de libertades en los extremos de las barras.*

  {{1,1},{1,1},{1,0},{1,1}}

- *Fuerzas aplicadas en los nudos (matriz n x 3): Constantes que multiplicadas por q, qL^2 ó F y FL dan el valor de las cargas aplicadas en los nudos de la estructura.*

  {{0,0,0},{0,0,0},{0,0,0},{1,0,0}}

- *Fuerzas aplicadas en las barras (matriz b x 4): Constantes que multiplicadas por q dan el valor inicial y final de las cargas distribuidas en cada barra.*

  {{0,0,0,0},{0,0,-1,2},{0,0,0,0},{0,0,0,0}}

- *Fuerzas aplicadas en las libertades de barras (matriz b x 2): Constantes que multiplicadas por FL ó qL^2 nos dan el valor de los momentos aplicados en las libertades de los extremos de barras.*

{{0,0},{0,0},{0,-1},{0,0}}

- *Temperaturas en las barras (lista b): Constantes que multiplicadas por T nos dan el valor de los incrementos térmicos de cada barra.*

  {0,0,0,0}

- *Indicativo de las deformaciones axiles (lista b): Conjunto binario (0 ó 1) que indica si hay que tener en cuenta las deformaciones axiles de las barras.*

  {0,0,0,1}

- *Indicativo de la variable de diseño asociada a la barra(lista b): Conjunto de valores que indican a qué grupo de diseño pertenece cada barra.*

  {1,1,1,2}

- *Datos numéricos del problema (lista con el siguiente orden {q ó F, longitud, σ_E,E,λ_{max}, λ_{max}, lista con los coeficientes β de cada barra, β, lista con el indicativo de pandeo impedido fuera del plano o no de cada grupo de barras, T, α}*)

  {25,500,2600,2.1*10^6,200,{1,1,1,1},{0,0},0,0}

Cuando tengamos cargas repartidas en la estructura (como la de la Fig. 40) el factor para las cargas distribuidas será q, para las fuerzas aplicadas será qL, y para los momentos aplicados qL^2 , mientras que si sólo tenemos aplicadas acciones concentradas el factor común será F para las fuerzas y FL para los momentos. Es fácil ver que existe una relación entre ellos muy sencilla.

Todas las unidades de longitud estarán adimensionalizadas o divididas por L, que es un valor genérico de longitud.

Cuando tengamos barras sometidas a un incremento de temperatura, que indicar adimensionalizando o dividiendo de forma análoga al anterior por un valor genérico de la temperatura T.

Las unidades para las que está preparada la aplicación son Kg, cm y ºC. Utilizar unidades diferentes significa cambiar los archivos que usa el programa para obtener los datos necesarios de los perfiles y puede falsear los resultados obtenidos; por este motivo no se recomienda en ningún caso variar dichos archivos o usar unidades diferentes a las recomendadas.

Es interesante explicar el contenido de algunas de estas líneas. Por ejemplo, la primera línea tenemos que guarda las componentes en coordenadas globales de los nudos de la estructura. La segunda línea guarda la conexión de las barras, es decir tenemos que la barra número uno va de 1 a 2, la segunda va de 2 a 3, etc. La tercera

línea indica en coordenadas globales si los nudos tienen grados de libertad coaccionados, cuando tenemos un 0, significa que ese grado de libertad está coaccionado y cuando tenemos un 1, significa que está libre. En la cuarta fila indicamos la presencia de libertades en las barras, un 0 indica que en el extremo de la barra que esté hay una libertad. En la quinta fila, tenemos las fuerzas aplicadas en los nudos de las barras en coordenadas globales. En la sexta tenemos las cargas sobre las barras. Cada barra podemos ver que tiene cuatro cantidades asignadas. En las dos primeras la información que tenemos es cuánto vale la ley de cargas longitudinales en el origen y en el extremo de la barra respectivamente y las ultimas dos componentes indican el valor de la ley de carga transversal en el origen y en el extremo de la barra.

El resto de filas son muy sencillas de entender y solo cabe mencionar especialmente que del análisis de esos datos, tenemos que no hay ninguna barra cargada térmicamente (fila 8), que podemos despreciar las deformaciones debidas al axil en todas las barras menos en la barra numero 4 (fila 9). Las barras 1,2,3 tienen las mismas características geométricas mientras que la 4 tiene características geométricas diferentes (fila 10) y que el pandeo no está impedido en ninguna barra (7º elemento de la fila 11 {0,0}). Este último dato es importante darse cuenta que si queremos diferenciar si parte de la estructura tiene el pandeo fuera del plano impedido o no, la tendremos que poner en un grupo de diseño diferente al del resto de barras.

Para el caso de emparrillados es necesaria menos información siendo la estructura general de un fichero de emparrillado la siguiente:

Coordenadas de nudos en forma de matriz (n x 3)

Conectividad entre las barras en forma de matriz (b x 2)

Libertades en los nudos en forma de matriz (n x 3)

Libertades en los extremos de barras en forma de matriz (b x 2 x 2)

Fuerzas aplicadas en los nudos en forma de matriz (n x 3)

Fuerzas aplicadas en las barras en forma de matriz (b x 2)

Fuerzas aplicadas en las libertades de barras (b x 2 x 2)

Indicativo de la variable d e diseño asociada a la barra en forma de lista (b)

Datos numéricos del problema en forma de lista con el siguiente orden {q ó F, longitud, σ_E, E, G}

Como podemos ver en el esquema, las tres primera líneas son comunes para pórticos y emparrillados, sin embargo la cuarta varía. El motivo es porque ahora, podemos tener 2 libertades en cada extremo de barra. La fila quinta es idéntica a pórticos, la sexta varía porque ahora sólo podemos cargar las barras transversalmente. La séptima cambia en

función de lo comentado para la fila 4. Desaparecen las filas 8 y 9 y la 10 disminuye la cantidad de información que necesita a 5 componentes donde la única novedad es la necesidad de incluir la constante del material G. El el archivo para la Fig. 41 sería:

{{0, 0}, {0, 1}, {1., 1.}}

{{1, 2}, {2, 3}}

{{0, 0, 0}, {1, 1, 1}, {0, 0, 0}}

{{{1, 1}, {1, 1}}, {{1, 1}, {1, 1}}}

{{0, 0, 0}, {0, 0, -1.}, {0, 0, 0}}

{{0, 0}, {0, 0}}

{{{0, 0}, {0, 0}}, {{0, 0}, {0, 0}}}

{1, 1}

{2000., 400., 2600., 2.1*^6, 810000.}

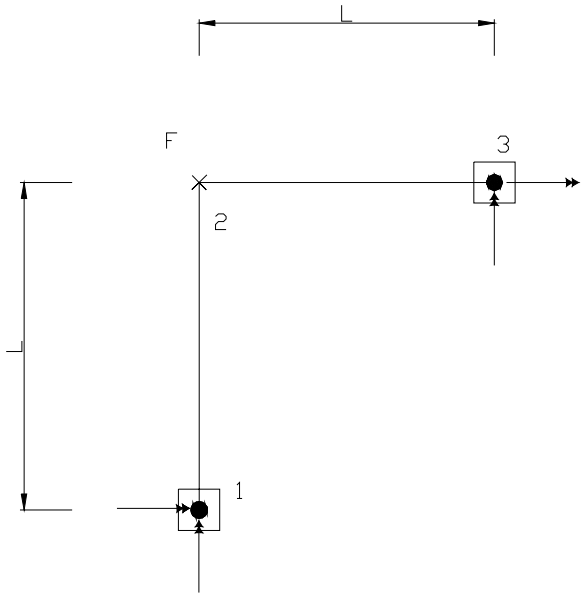


Fig. 41

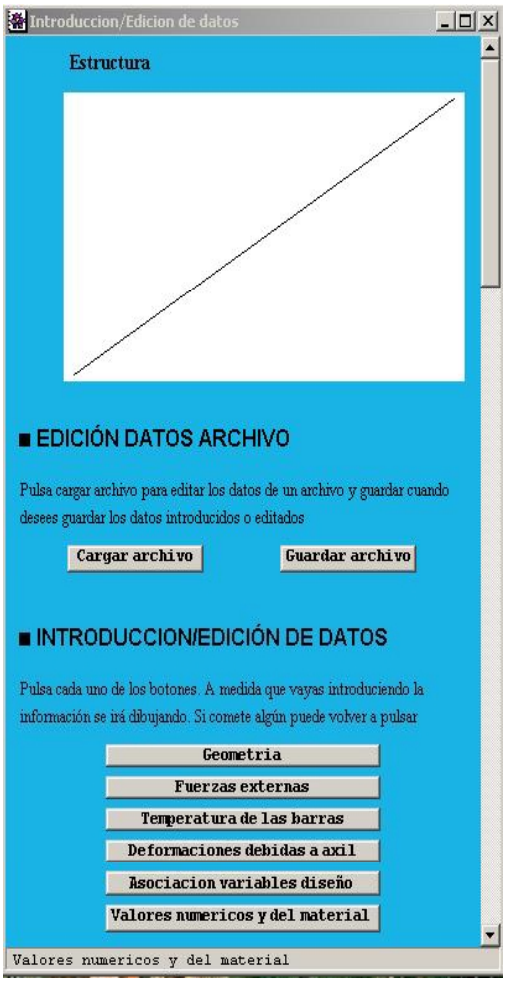
Con respecto a la forma de introducir los datos puede aplicarse todo lo anteriormente dicho para pórticos

5.2-. Entrada de datos por consola.

En la Fig. 42 podemos ver la parte de la Interfaz de usuario dedicada a la introducción de datos externa. En ella tenemos dos botones, uno de ellos comentado anteriormente, **Introducción / Edición de datos**, que es el botón que nos permite

introducir los datos vía consola y guardar la información en un archivo. Cuando presionamos este botón se abre una pantalla como la de la Fig. 43, donde se muestra una zona de dibujo donde se representarán los datos y una serie de botones.

Fig. 42



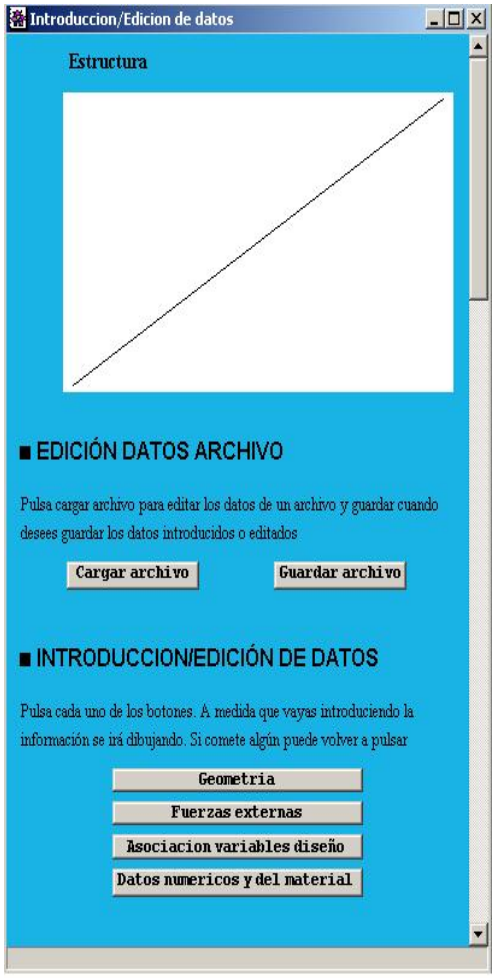
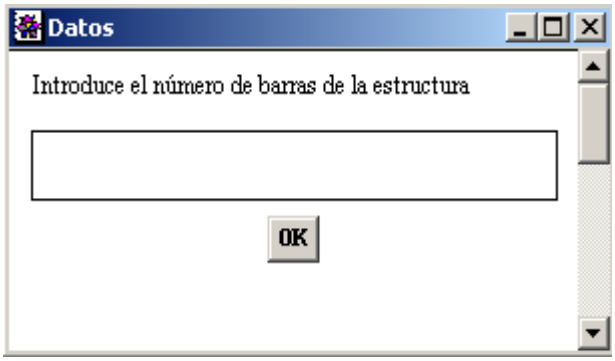


Fig 42 a) Fig 42 b)

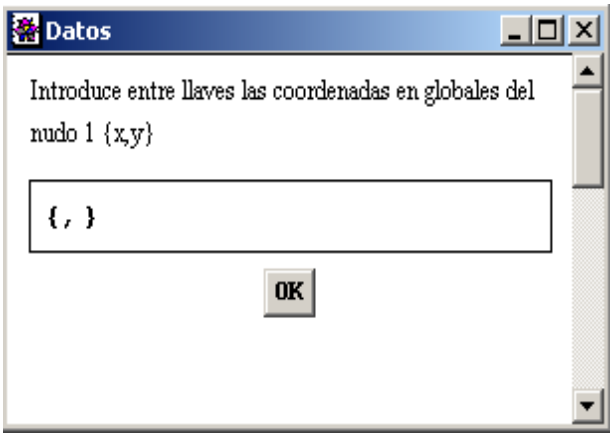
En la Fig. 43ª se muestra la ventana de interacción para pórticos y en la 42b la ventana de interacción para emparrillados. Se ha elegido el formato de listas para

introducir los datos necesarios. A continuación se incluye un breve repaso de cómo hay que introducir estos datos dependiendo de la información que se nos esté solicitando.

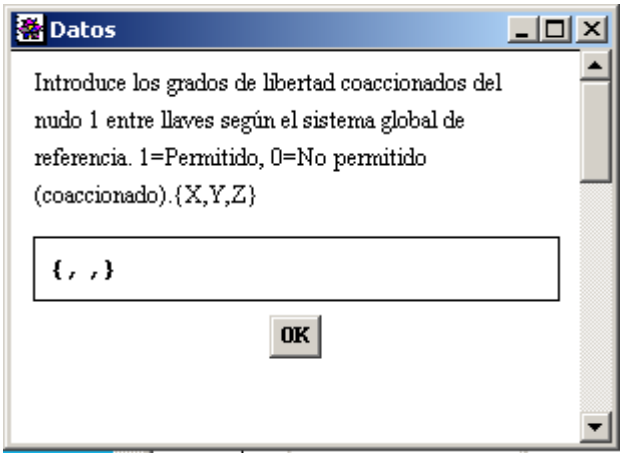
La primera información que se nos pide son el número de barras, número de nudos, número de libertades de la estructura y número de grupos de diseño diferentes usando pantallas como la de la.



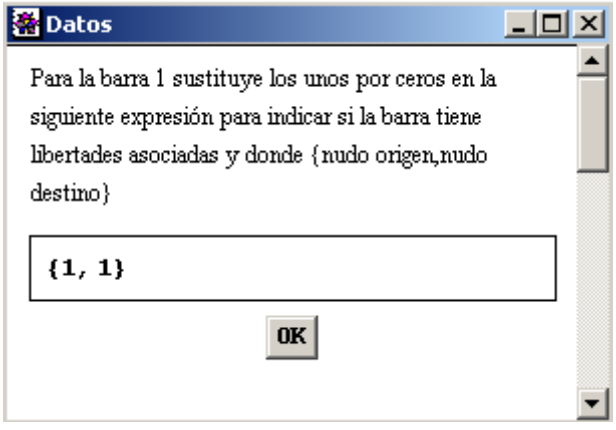
Cuando tengamos que introducir las coordenadas de los nudos, aparecerá una pantalla como la de la derecha donde tenemos un hueco que nos indica cómo introducir los datos. Esta forma de introducir datos es sencilla porque nos visualiza qué nudo es el que estamos introduciendo a la vez que nos permite desplazarnos rápidamente entre los huecos con los cursores, evitándonos tener que introducir las llaves. Con un formato parecido a este también se introduce la conectividad entre barras.



Para la introducción de los grados de libertad permitidos o coaccionados aparece una ventana como la de la derecha donde sólo tendremos que rellenar con 0 ó 1 dependiendo de si el grado de libertad que estamos introduciendo está o no coaccionado. De forma parecida se introducen los datos de fuerzas aplicadas en los nudos, dando ahora el valor correspondiente a la constante de fuerza aplicada en el nudo y dirección en cuestión.



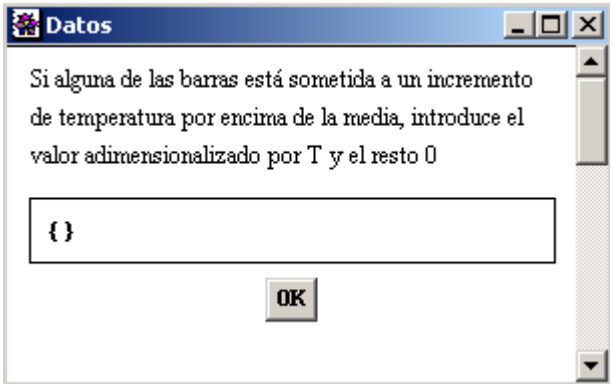
La introducción de libertades es ligeramente diferente al del resto, porque es menos normal tener que introducir una libertad; por lo tanto se da la información por defecto (sin libertades) y se le permite al usuario cambiar el 1 por 0 para introducir el lugar donde tengamos la libertad. Esta



forma de introducir los datos nos permite agilizar el proceso.

Cuando tenemos que introducir las fuerzas aplicadas a la estructura el proceso es similar, con valores nulos por defecto.

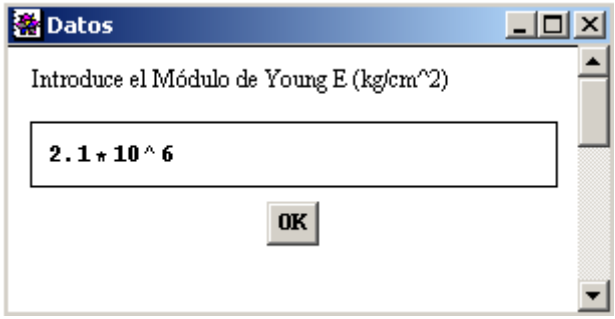
Para introducir la temperatura de las barras, la pantalla por defecto es ésta; en ella tendremos que introducir la constante de temperatura de cada una de las barras. Las pantallas para la introducción de las variables de diseño, indicación de las deformaciones axiles, indicación de los valores β e indicadores de pandeo impedido fuera del plano idénticas a la mostrada en la figura de la derecha.



En las Fig. 43 a) y b), podemos ver que el último botón del documento es **Valores numéricos y del material**. Este botón no es necesario usarlo cuando tengamos un problema simbólico en el que no tenemos que realizar la parte de diseño. Cuando tengamos que usarlo, se nos pedirá la información necesaria, dándonos cuando sea posible un valor numérico por defecto.

5.3. Introducción de datos durante el cálculo.

La interfaz es el documento que nos va orientado en el uso de la aplicación y en la forma natural de resolver el problema, como se muestra en la Fig. 44. En esta figura. aparecen una serie de botones que, pulsados secuencialmente nos permiten resolver la estructura simbólicamente.



Aunque la mayoría de la información que el programa necesita está bien especificada y se indica cómo introducir la información se va a hacer una breve reseña para eliminar posibles dudas.

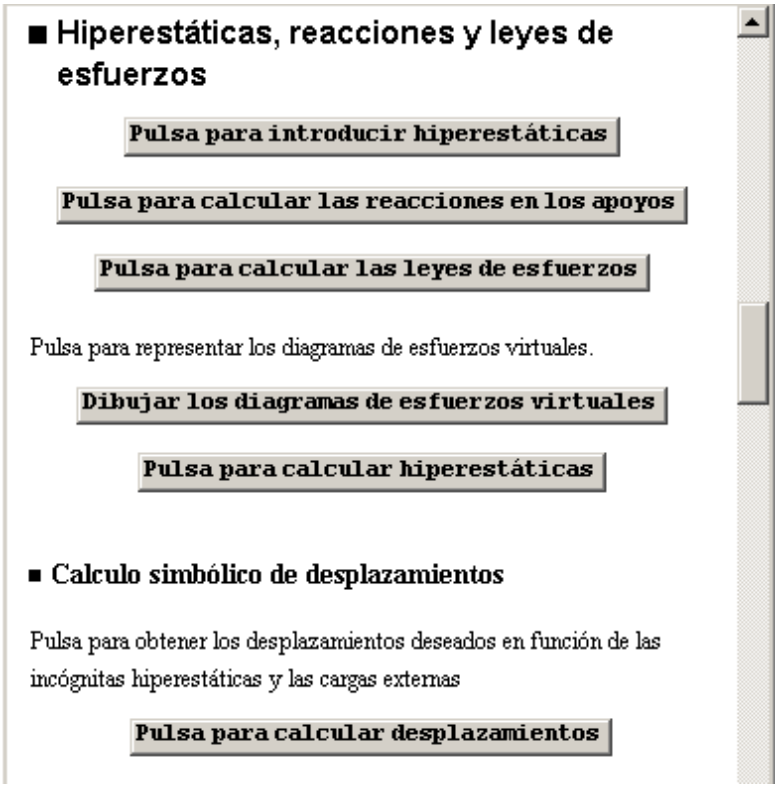
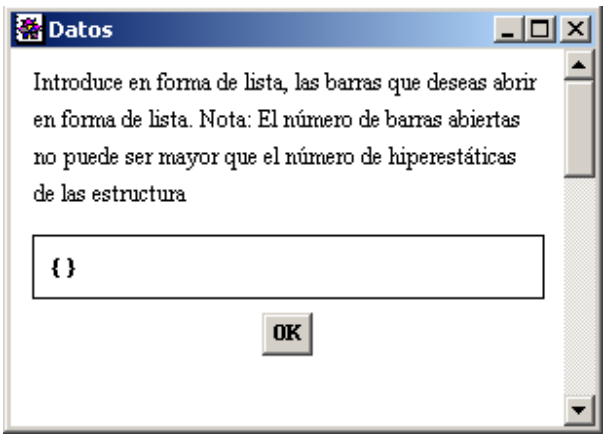


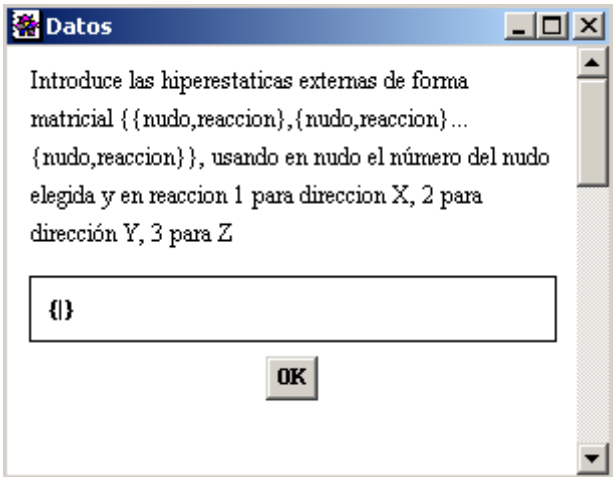
Fig. 44

Cuando queremos introducir las hiperestáticas, aparecen una serie de pantallas como las que aparecen a la derecha y en el orden en que estas aparecen.



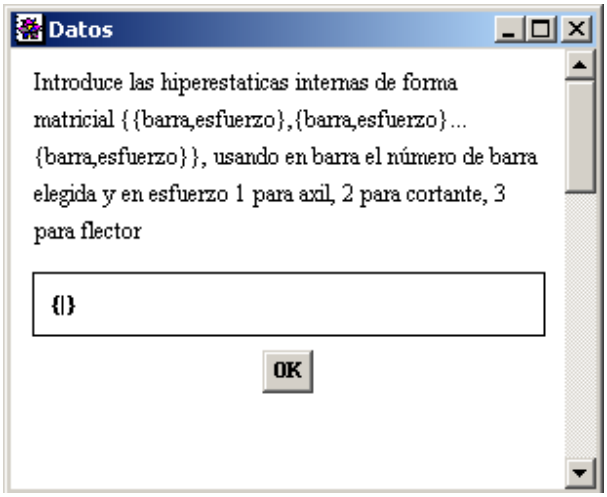
En esta primera pantalla deberemos introducir en forma de lista las barras que deseamos abrir, lo cual es necesario en estructuras cerradas y opcional en estructuras abiertas en las que se deseen escoger hiperestáticas internas. Una vez elegidas estas barras (se puede dejar como está, indicaría por defecto que no queremos abrir ninguna barra), el programa genera unas ecuaciones de equilibrio reducidas donde aparecen como incógnitas 3 esfuerzos por barra abierta, las reacciones de la estructura, y las cargas aplicadas. Estas matrices pueden verse en el apartado 5.3 y permiten al usuario, determinar cuales son las incógnitas que puede elegir como hiperestáticas (si no se eligen bien las incógnitas hiperestáticas, tenemos un mecanismo).

En la segunda y tercera pantallas que aparecen se nos pide que introduzcamos las incógnitas hiperestáticas que queremos usar. Esta pantalla pide las incógnitas hiperestáticas externas (reacciones) en la forma mostrada. Si queremos introducir la reacción en el eje global X de un nudo cuyo número es 1 lo que tendremos que introducir es {{1,1}}.



Para introducir las hiperestáticas internas el proceso es análogo. Por ejemplo si quisiéramos escoger el axil de la barra 3, tendríamos que introducir {{3,1}}.

Es importante aclarar que si una estructura tiene una sola hiperestática e introducimos una hiperestática externa, la pantalla para las hiperestáticas internas no aparecerá.



Si queremos introducir hiperestáticas internas y externas solamente tendremos que introducir la cantidad que queramos de cada una obviamente con su suma igual al grado de hiperestaticidad de la estructura.

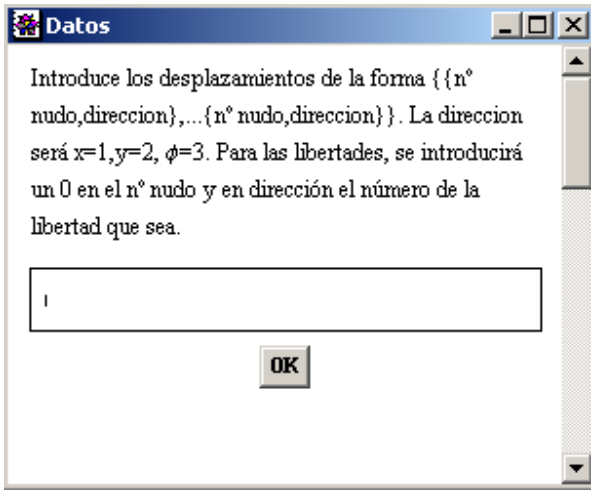
Si no queremos introducir hiperestáticas externas sólo tendremos que pulsar OK.

También es importante recordar que las hiperestáticas hay que introducirlas como si fueran matrices; si se introducen en forma de listas pueden dar errores, y ese es el motivo por el cual por defecto se escriban unas llaves, para que se introduzca en su interior las llaves correspondientes con el par de valores que se desee.

Volviendo a la Fig. 44, los siguiente cuatro botones que se encuentran por debajo de **Pulsa para introducir hiperestáticas** son botones ejecutores, cuyas acciones se han separado por motivos docentes, de forma que reproduzcan las etapas clásicas de la resolución manual de una estructura mediante el Método de las Fuerzas.

Uno de los botones que podemos ver en la Fig. 44 es **Dibujar los diagramas de esfuerzos virtuales**. Este botón no es necesario y su uso no es obligatorio desde el punto de vista del cálculo de la estructura, sin embargo estos estados virtuales (asociados a las incógnitas hiperestáticas) ayudan a comprender la liberación de ligaduras que permite obtener la estructura isostática ficticia. Además de los estados virtuales, se muestran los diagramas adimensionales de esfuerzo que originan en la estructura isostática ficticia las acciones externas.

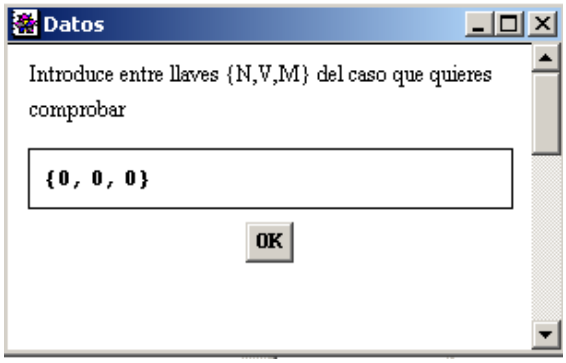
El último botón permite calcular los desplazamientos de la estructura en función de las acciones externas y las incógnitas hiperestáticas, que es la forma útil para el diseño de la estructura. El programa pide los desplazamientos que se desean calcular con una pantalla parecida a la que usamos para introducir las hiperestáticas y que podemos ver en la figura de la derecha. En esta pantalla no aparecen por defecto las llaves, ya que no es obligatorio calcular los desplazamientos, sino opcional, y deberá indicar en forma de matriz los desplazamientos que se desean calcular en coordenadas globales. La forma de indicar que queremos calcular el giro de una libertad es ligeramente diferente al del resto de desplazamientos. Para pedir el giro de una libertad se tendrá que indicar con un cero en la posición de nº de nudo (indica que es una libertad) y un número que indica el número de la libertad. Las libertades no son enumeradas, sin embargo, el programa identifica cada una de ellas con un número que indica en que lugar encuentra la libertad en función de la posición en la que la encuentra. Tengamos por ejemplo los datos de las libertades del ejemplo de la Fig. 40 "{{1,1},{1,1},{1,0},{1,1}}", tenemos solo una libertad que el nudo final de la barra 3, pues esa será la libertad 1, si por ejemplo tuviéramos que hay otra libertad en el nudo inicio de la barra 2 ({{1,1},{0,1},{1,0},{1,1}}), tendríamos que ésta sería la libertad 1, mientras que la de la barra 3 sería la libertad 2. Esta forma de identificar las libertades es igualmente aplicable a los emparrillados.



Aquí termina el cálculo simbólico de la estructura y comienza, si el usuario así lo decide, el diseño de la misma. En el proceso de diseño aparecen una serie de pantallas para introducir datos, siendo éstas de un uso más sencillo que las anteriores. A continuación se hace una relación de las más importantes y de las formas de interaccionar con ellas.

Iteración simple.

El programa permite, una vez introducidos los datos usar los algoritmos de la aplicación para resolver un caso concreto, en el caso de la figura de un pórtico con los valores dados de N (axil), V (cortante) y M (flector).



Elección de perfiles.

Para la elección de perfiles se programó como ya se indicó en la sección 4.5.3 un input diferente al hasta ahora visto. El motivo fue agilizar la elección de perfiles y disminuir los posibles errores a la hora de elegir el perfil que deseamos usar.

La manera de interactuar con el Input es muy sencilla. Se pulsa el botón correspondiente al perfil deseado y el nombre del perfil se escribe en la esquina superior izquierda de la pantalla. El usuario podrá cambiar de perfil tantas veces como lo desee mientras no pulse OK; en el momento que pulsa OK, el perfil que estuviera señalizado en la esquina superior izquierda es el perfil elegido.

En la Fig. 45 aparece un esquema donde se explican cada una de las partes de la pantalla.

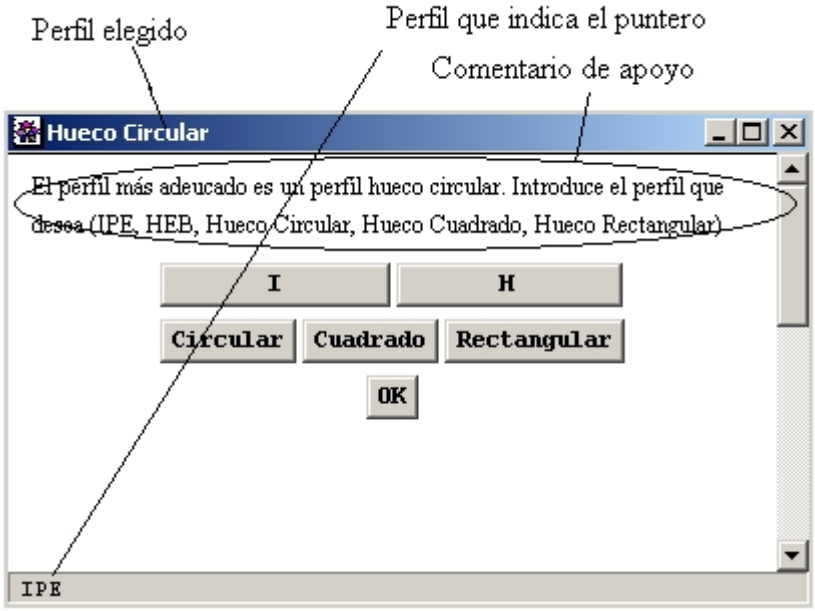


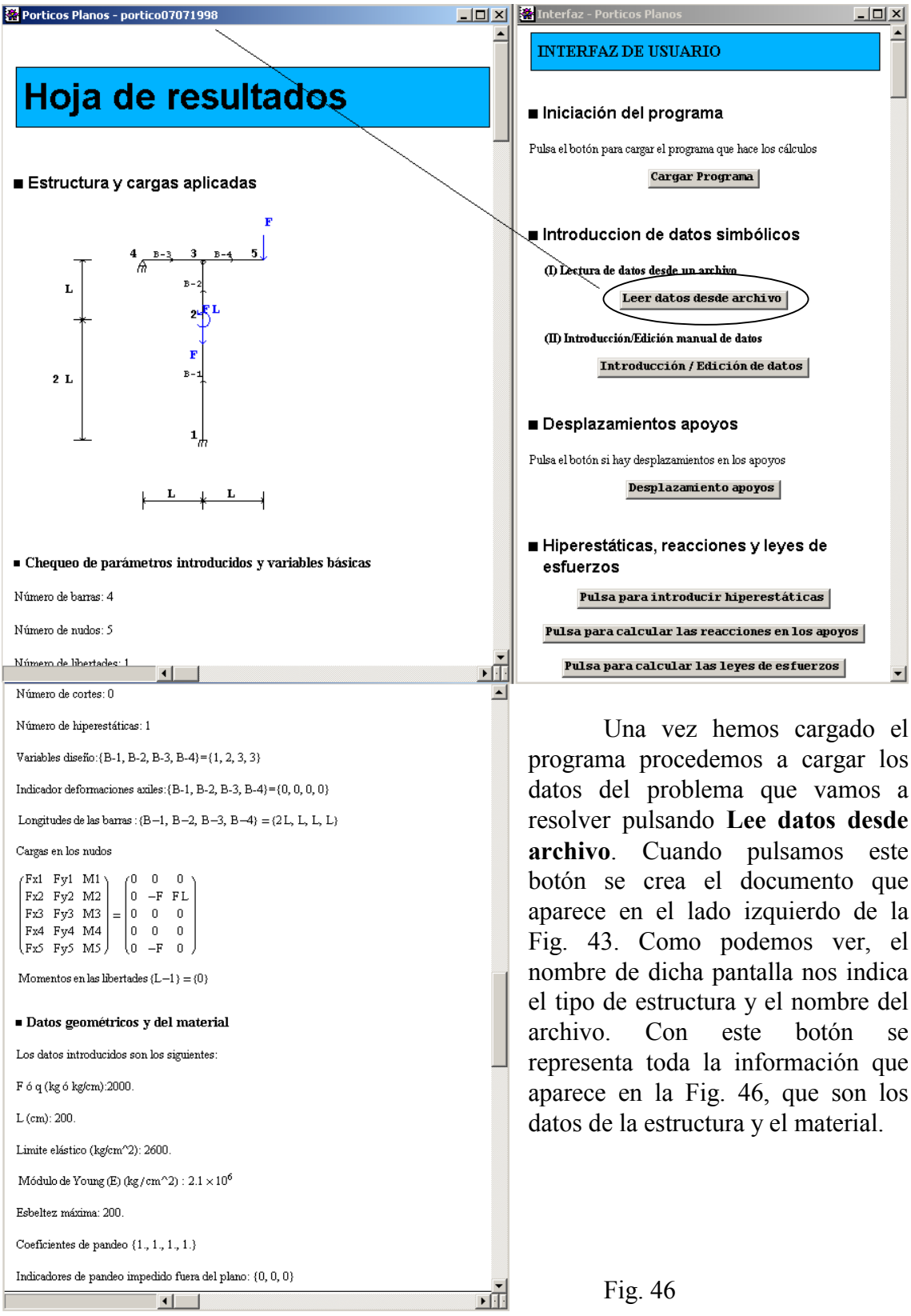
Fig. 45

5.4. Análisis de los datos de salida. Realización de un ejemplo.

En este apartado vamos a ilustrar la salida de resultados del programa en un ejemplo concreto, y se dará así mismo una breve explicación de las características más importantes de los datos de salida asociados a los botones que los generan.

El ejemplo que vamos a realizar es un problema que aparece en la colección de problemas de exámenes resueltos para los alumnos de Resistencia de Materiales del 3er Curso de Ingenieros Industriales, concretamente el segundo problema propuesto en el examen de junio de 1998 (7-7-1998).

Este problema es un pórtico plano, y por lo tanto seleccionamos la interfaz de pórticos planos **Interfaz-porticos.nb**. y cargamos el paquete pulsando el botón correspondiente.



Una vez hemos cargado el programa procedemos a cargar los datos del problema que vamos a resolver pulsando **Lee datos desde archivo**. Cuando pulsamos este botón se crea el documento que aparece en el lado izquierdo de la Fig. 43. Como podemos ver, el nombre de dicha pantalla nos indica el tipo de estructura y el nombre del archivo. Con este botón se representa toda la información que aparece en la Fig. 46, que son los datos de la estructura y el material.

Fig. 46

Como podemos ver en los resultados que salen, esta estructura es hiperestática de orden 1; por lo tanto tendremos que realizar la elección de hiperestáticas. Previamente se solicita la información relativa a las (secciones extremas de) barras por las que queremos abrir la estructura, lo que permite establecer las ecuaciones de equilibrio reducidas, de las que se dan dos versiones equivalentes (algorítmica y triangulada canónicamente, ver Fig. 47) para ayudar a comprobar una posible matriz establecida por el usuario. A partid de estas matrices pueden escogerse incógnitas hiperestáticas. Un conjunto de h variables estáticas es un conjunto admisible de hiperestáticas si al eliminar sus columnas asociadas en la matriz de equilibrio al matriz cuadrada resultante es no singular. Si el usuario escoge unas hiperestáticas admisibles se incluye una tercera versión de la matriz de equilibrio, en la que se triangulan las columnas asociadas a variables no hiperestáticas (Fig. 47).

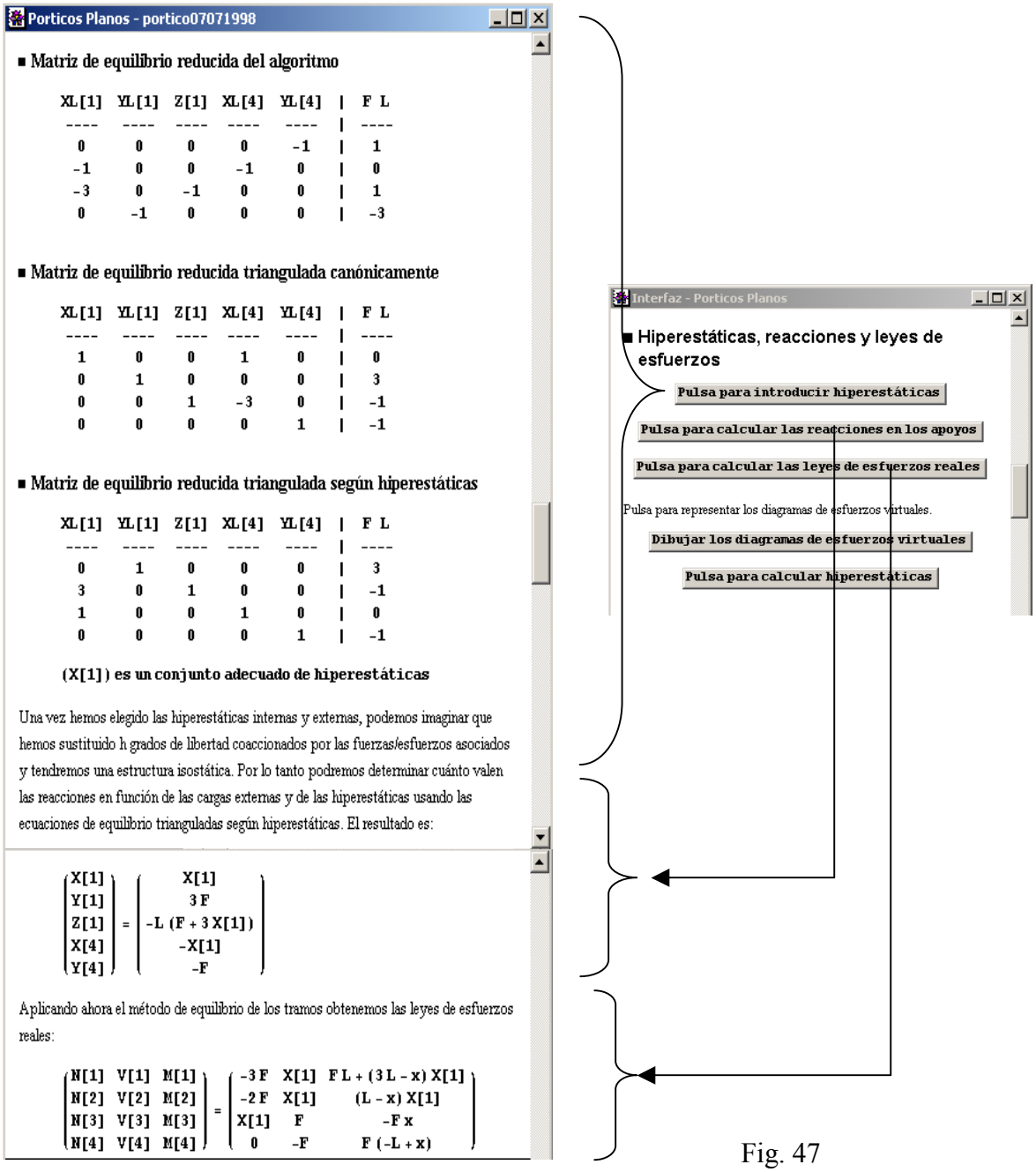


Fig. 47

También podemos ver el resultado del cálculo de las reacciones y los esfuerzos internos, expresados en función de las cargas externas y las incógnitas hiperestáticas.

En la Fig. 48 podemos ver el resultado de las ecuaciones de compatibilidad y la resolución de las incógnitas hiperestáticas (**Pulsa para calcular las hiperestáticas)**.

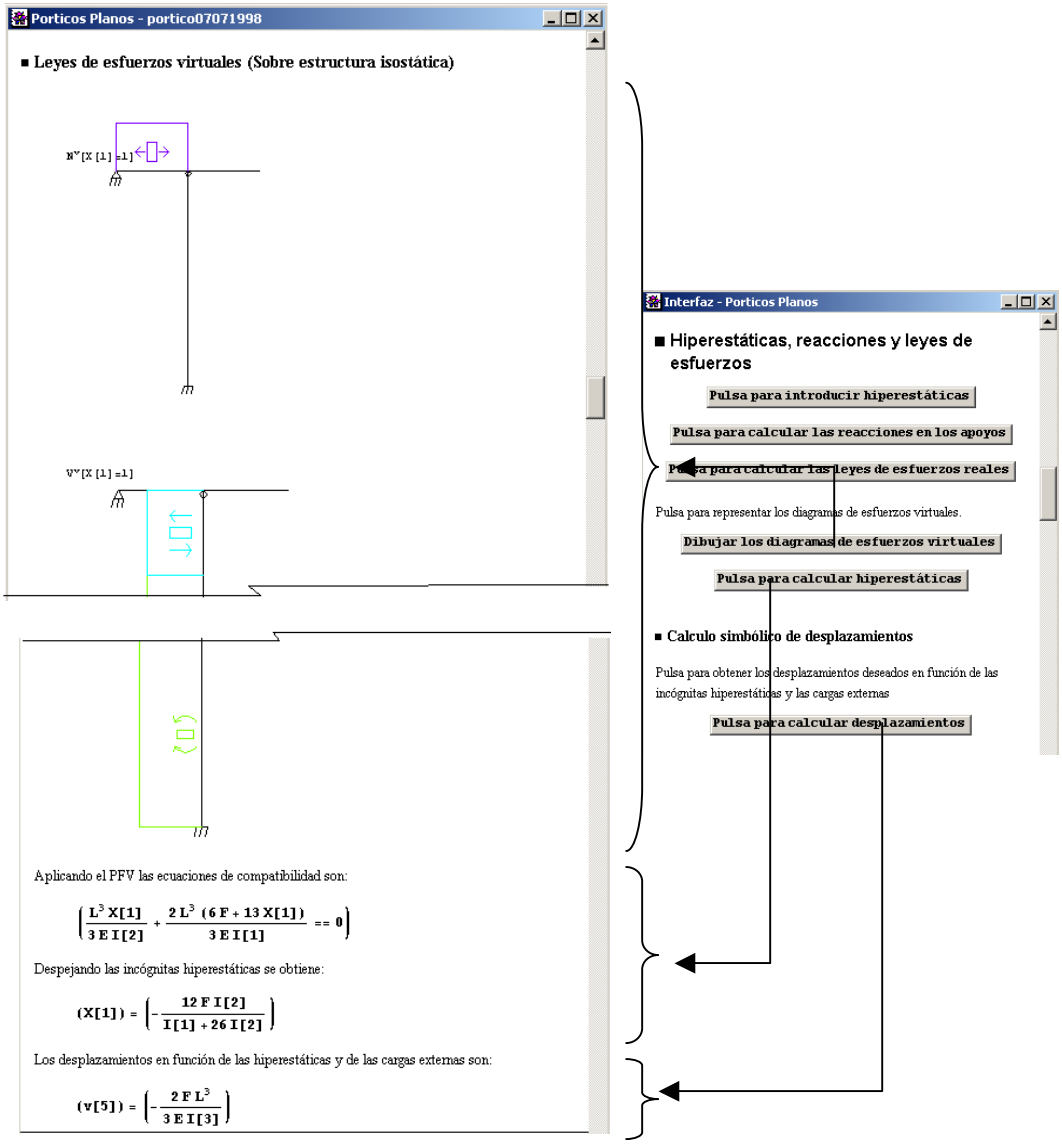


Fig. 48

Aquí acaba la resolución analítica de la estructura. El siguiente paso (opcional) es el diseño del misma. Para poder realizar el diseño necesitamos valores numéricos. Tendremos que introducir el valor de la longitud L, de q ó F, así como el valor de T que aparecen en las soluciones analíticas de la estructura. También necesitaremos los valores de las propiedades del material E, G, etc. Esto puede hacerse aquí si no se ha realizado anteriormente en el archivo, y la salida del programa es similar a lo que aparece en la Fig. 46. En este ejemplo se realizó la introducción de datos numéricos cuando cargamos el archivo.

En esta parte de diseño, como podemos ver en la Fig. 49 aparecen una serie de botones que no hemos usado en este ejemplo. Estos botones no son estrictamente necesarios, simplemente nos ayudan a determinar una serie de parámetros o a realizar una serie de operaciones independientes a las necesarias en la resolución de la estructura.

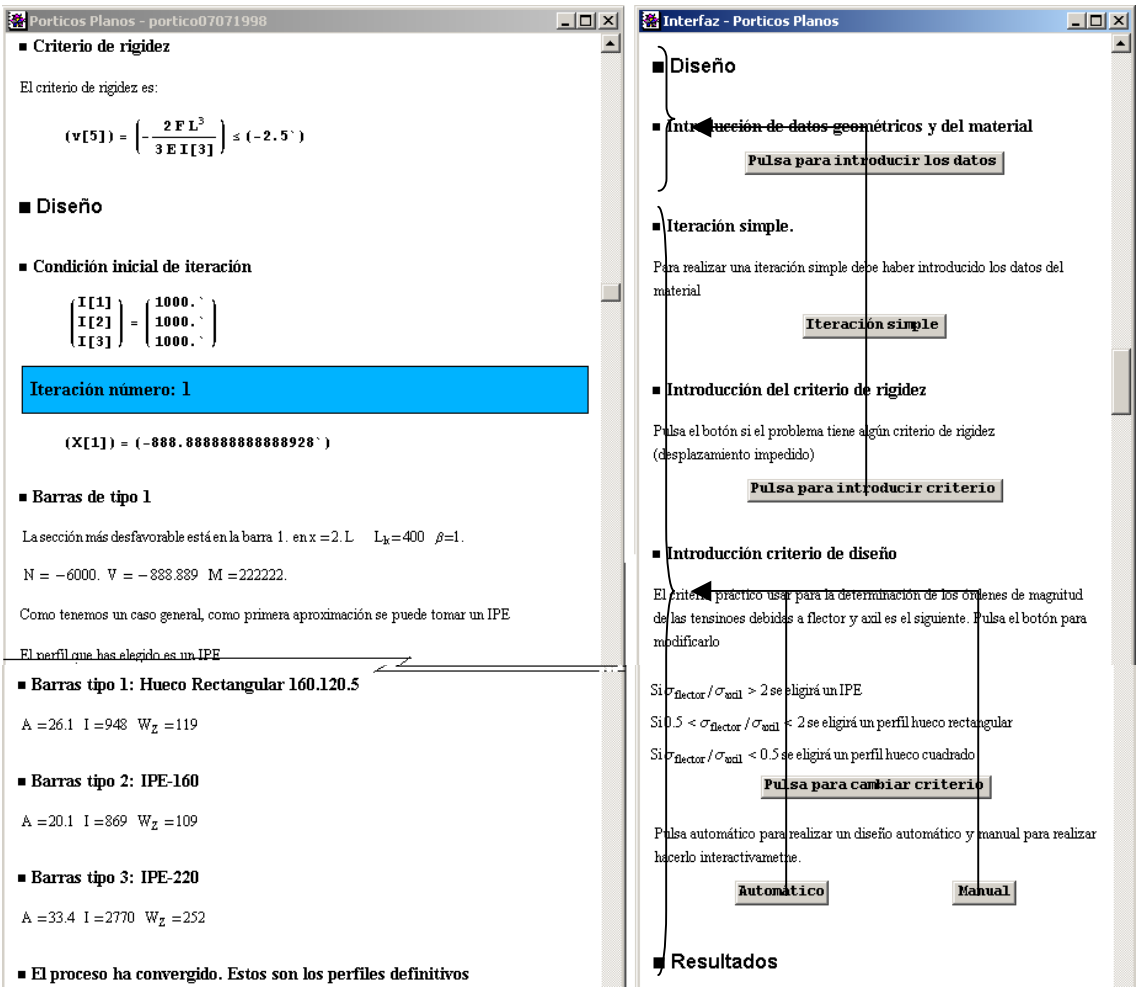


Fig. 49

La última parte que nos queda es la representación de datos, donde podemos representar los diagramas de esfuerzos finales y la deformada, así como la manipulación de archivos (imprimir el documento final o guardarlo en forma de archivo). Esto se ilustra en la Fig. 50 donde podemos ver la parte correspondiente a la representación de la deformada y parte de la representación de los diagramas de esfuerzos (la figura no muestra toda la información referente a los diagramas de esfuerzos por ser poco relevante y muy similar a lo que vimos en la Fig. 48.

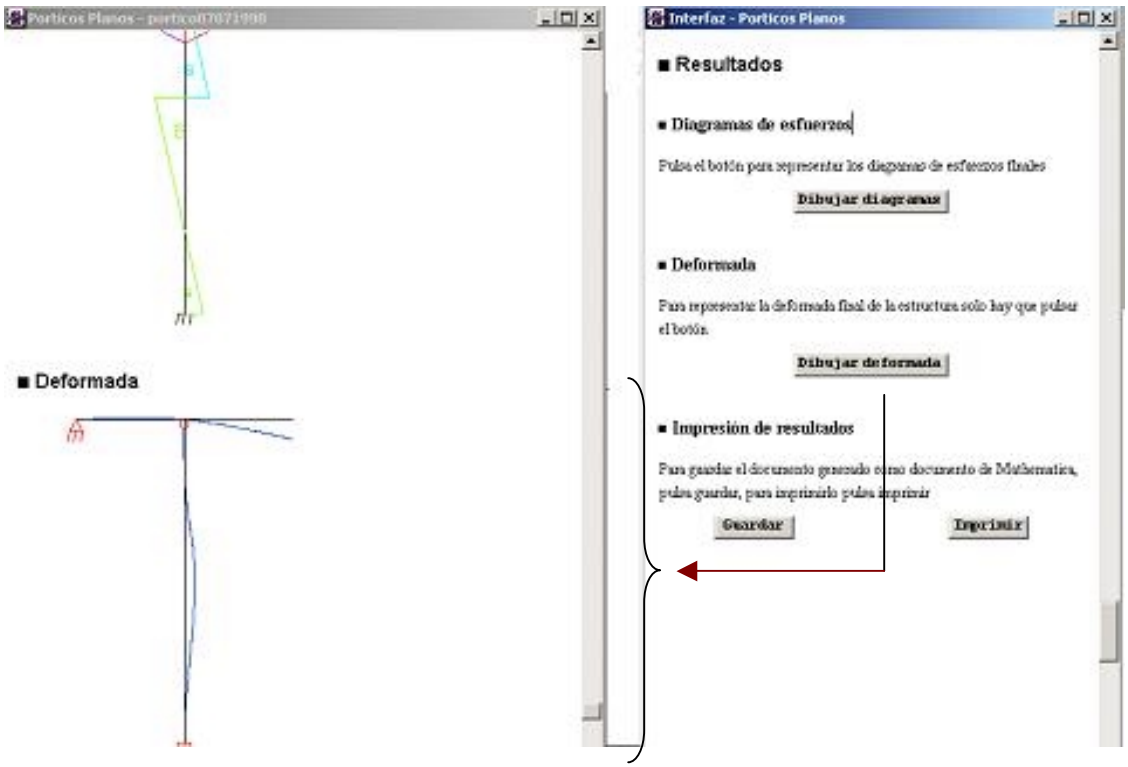


Fig. 50

5.5. Sistema de archivos. Directorios.

Anteriormente se comentó que Mathematica es un programa que funciona con módulos o paquetes. Esta característica lo hace muy potente y flexible. También se comentó anteriormente que Mathematica contiene paquetes de serie y para localizar estos paquetes durante una sesión de Mathematica no tenemos que indicarle a Mathematica donde se encuentran físicamente, Mathematica contiene una variable interna **$Path** con una lista de todos sus directorios relevantes; es decir, que si se le pide un archivo, Mathematica buscará en todos esos directorios.

Para evitar confusiones se ha optado por usar la estructura de archivos originales de Mathematica, donde Mathematica tiene para sus paquetes propios una carpeta llamada **StandardPackages** y para los externos otra carpeta llamada **ExtraPackages**. Se podía haber usado una estructuras de archivos diferente modificando oportunamente las variables internas de Mathematica; sin embargo para posteriores modificaciones y para mantener la misma estructura se ha decidido hacer de esta manera.

La aplicación está formada, como anteriormente se ha comentado, tanto por archivos de datos como por archivos de Mathematica. La organización de estos archivos depende de su función:

- Los paquetes se han guardado como un paquete extra para Mathematica (carpeta **ExtraPackages**). En la mayoría de los sistemas bajo Windows

que usen Mathematica 3.0, los paquetes estarán en la dirección: *D:\Archivos de programa\Wolfram\Research\Mathematica\3.0\AddOns\ExtraPackages\Madre* donde *Madre* es el nombre que le hemos dado al proyecto y el nombre de la carpeta que contiene estos archivos. El nombre los archivos de que contienen las funciones programadas son **portico.m** y **emparrillado.m**, los cuales pueden ser visualizados con un editor de textos .

- Los archivos de datos como pueden ser las tablas de perfiles y la tabla de los coeficientes de pandeo están en una subcarpeta de *Madre* llamada *Tablas*. Se ha decidido guardar esa información ahí porque no es información que debiera ser modificada por ningún usuario por formar una parte de uso interno del programa y porque su alteración podría provocar un erróneo funcionamiento de la aplicación.

- Los archivos creados por el usuario mientras utiliza la aplicación se guardarán en tres carpetas en el mismo subdirectorio donde tengamos la interfaz. Estos subdirectorios serán *Datos_Emparrillados* para los emparrillados, *Datos_Porticos* para los pórticos y *Resultados* para guardar todos los resultados.

Estos directorios y esta distribución de archivos son generados durante el proceso de instalación del programa y se recomienda su uso cuando se pretenda generar los archivos de datos manualmente.

5.6-. Requisitos mínimos de Hardware.

- Debe disponer de un PC con un procesador intel 80386, o superior.
- Disco duro necesario: Recomendado 63 MB. Mínimo 24MB (trabajando desde el CD de Mathematica).
- Sistema Operativo: Windows 95 o Windows NT 3.51
- Memoria Ram: Recomendado 16 MB. Mínimo 8MB.

.

5.7-. Errores conocidos.

- En la aplicación para pórticos planos se ha usado N para nombrar el axil de las barras. N es un comando de Mathematica que sirve para dar el valor numérico de una expresión, por este motivo, Mathematica da un warning relacionado con los contextos sin embargo, no afecta al funcionamiento de la aplicación. Durante la sesión de Mathematica correspondiente se ha renombrado esta función como NN.

- Si en la introducción del nombre de los archivos a cargar nos equivocamos en el nombre del archivo, se generará una serie de errores que no se han podido suprimir. La solución es volver a cargar el archivo con el nombre correcto. Si se

desea eliminar todos los warnings generados de la interfaz se puede volver a cargar esta sin la necesidad de volver a cargar el paquete en cuestión.

- Cuando se intenta grabar el documento generado durante la aplicación, éste se debería guardar en un directorio llamado Resueltos, sin embargo por motivos desconocidos, todos los esfuerzos realizados por lograr grabar dicho documento en el citado directorio han sido infructíferos a excepción de cuando se ha arrancado el programa desde dicho directorio. Este problema no se ha producido cuando se ha intentado guardar un archivo en un directorio diferente al usado cuando el proceso ha sido manual.

- El proceso automático de diseño depende de dos valores que determinan qué tipo de perfil es más adecuado. Una mala elección de estos valores puede producir que el proceso converja más lentamente.

6-. CONCLUSIONES.

Se ha desarrollado con el programa Mathematica un programa que permite reproducir las distintas etapas del cálculo manual analítico de una estructura mediante el Método de las Fuerzas; posteriormente se ha implementado una etapa de diseño numérico de la estructura. El usuario típico del programa sería un alumno de la asignatura de Resistencia de Materiales (3er curso de Ingenieros Industriales), y la finalidad básica de la aplicación es permitir al alumno el chequeo y aplicación de las ideas desarrolladas en la teoría de la mencionada asignatura.

Sin usar una herramienta que permitiera el cálculo simbólico por ordenador, el desarrollo de la aplicación hubiera sido imposible. El programa elegido ha sido Mathematica (versión 3.0). Este programa presenta una serie de ventajas, como ya se ha comentado a lo largo del documento, y también presenta una serie de inconvenientes, que se han intentado solventar de la manera más racional posible.

Uno de los problemas de Mathematica desde mi punto de vista es que no está completamente integrado dentro del entorno Windows, sobre todo con la serie de programas de gran difusión como Office, etc. Esta incompatibilidad se hace patente cuando se pretende usar el Kernel como núcleo de una aplicación programada en Visual Basic que hubiera mejorado mucho la presentación y las posibilidades de la aplicación (se intentó, pero llegados a un punto, la cantidad de problemas que daba no lo hacía útil y el servicio técnico de Mathematica respondió diciendo que este caso no está soportado por Mathematica (unsupported goodies)).

En este sentido, uno de los objetivos ha sido conseguir un entorno de trabajo lo más cómodo posible, sobre todo para aquellas personas que nunca hayan usado Mathematica. El entorno de trabajo de Mathematica es un entorno que al usuario novel puede resultarle austero, y este es el motivo de programar la interfaz de usuario. La interfaz permite tanto al usuario avanzado (en Mathematica) como al usuario novel, interactuar con la aplicación, y si lo desea, con Mathematica. Considero que dicha interfaz es un punto intermedio entre la austeridad de Mathematica y la simpleza y control de los entornos mas orientados hacia Windows, adecuada por tanto al entorno académico para el que se ha desarrollado.

La interfaz de usuario permite generar un documento claro, donde los usuarios pueden ver de forma idónea cada uno de los pasos realizados y en el orden en que se han realizado. La interfaz programada permite que el usuario siga el desarrollo natural de resolución de un problema, estableciendo un orden secuencial de operaciones que refuerza la finalidad docente del proyecto.

Uno de los inconvenientes que se planteaba en el cálculo analítico de las estructuras por el método de las fuerzas es la carga simbólica de todo el proceso, donde aparecen operaciones poco sistematizables de carácter simbólico. Mathematica permite realizar todas estas operaciones simbólicas (integración simbólica, resolución de sistemas de ecuaciones simbólicos, etc.), pero con unos tiempos de computación relativamente altos. Debido al carácter docente de la aplicación, los tiempos de computación no son una prioridad, y por el contrario, el método matricial desarrollado

tiene la ventaja de ser simple, rápido y robusto. Conceptualmente, se ha planteado el problema usando como operación más complicada la inversión de una matriz cuadrada no singular, es decir, sin integraciones ni operaciones complicadas siempre y cuando no haya prevalecido la finalidad docente de la aplicación.

Este proyecto permite también el diseño de estructuras, entendiendo éste como la búsqueda de los perfiles que permiten minimizar el peso de material requerido y satisfacer los criterios de diseño. Este proceso se puede realizar manual y automáticamente. El proceso manual permite al usuario ir comprobando sus conocimientos (e incluso aumentarlos) debido a que va guiado por un árbol de toma de decisiones tutelado. Alternativamente, el proceso de diseño automático ofrece la posibilidad de realizar el diseño automático completo de la estructura.

Resumiendo, el proyecto permite resolver el problema de la resolución analítica de una estructura mediante el método de las fuerzas de una manera efectiva, rápida y clara así como el diseño posterior de la misma. La interfaz de usuario programada controla todo el proceso de cálculo.

7-. DESARROLLO FUTURO

El programa desarrollado tiene algunos puntos que se podrían mejorar en cada una de las partes en las que se divide; tanto en la parte analítico-teórica como en parte de programación computacional. Las mejoras se pueden hacer en ambas partes:

- Tipología de las estructuras.

  Debido a la finalidad docente del proyecto, las estructuras que se pueden calcular son estructuras sencillas (pórticos planos y emparrillados), sin embargo la metodología desarrollada podría aplicarse con cambios menores a estructuras tridimensionales

- Criterio de Rigidez.

  En el apartado 3.7 se comentaron los problemas asociados al cumplimiento de las restricciones en desplazamientos (criterio de rigidez). A este problema se le ha dado una solución sencilla debido a que no era el fin del proyecto desarrollar un algoritmo de más complejidad, que podría ser objeto de un trabajo futuro.

- Iteración automática.

  El programa permite realizar automáticamente el proceso de diseño. Este proceso va controlado principalmente por dos variables que nos indican qué tipo de perfil es más adecuado para cada estado tensional (valores límite de la relación entre tensiones). Estas variables son fijas durante el proceso de diseño y modificables por el usuario. Hay casos excepcionales en los que el proceso no converge debido a que la solución oscila. Sería interesante poder desarrollar un mecanismo de autoajuste de manera que cuando se detecte que el proceso iterativo toma un patrón de repetición, se modifiquen estos valores para llegar a la convergencia.

- Interfaz de usuario.

  - Una de las partes esenciales del programa es la interfaz de usuario. La interfaz programada tiene la ventaja de permitir a los usuarios usar Mathematica a la vez que permite usar la aplicación; por contra, la interfaz es bastante rudimentaria, disponiendo simplemente de botones como elementos activos. Cuando Mathematica lo permita (en la versión 3 usada no funcionaba), se podría programar un entorno visual más conseguido, usando como lenguajes de programación Visual Basic, Visual C, etc. y usando el Kernel de Mathematica como núcleo del programa.

  - Se ha intentado que las rutinas de gráficos sean lo más generales posibles y que permitan dibujar cualquier tipo de estructura con la mayor claridad posible. Otro de los puntos que se podría mejorar es la representación de la información, mediante algún tipo de optimización interna de los

parámetros de dibujo, que permitieran mejorar la presentación de resultados.

- Otro aspecto interesante que se podría mejorar es la forma de introducir los datos, de forma que fuera lo más gráfica posible. Nuevamente esto requiere una variación de Mathematica que permita el manejo hacia un entorno más visual.

- Control de errores.

  Uno de los puntos más importantes de una aplicación informática son las rutinas de control de errores. En este proyecto se han programado algunas de estas rutinas, sobre todo aquellas que están relacionadas con el chequeo de los datos de entrada; sin embargo se generan algunos errores durante la ejecución que no se han podido eliminar. No son estrictamente errores, son advertencias (warnings) de Mathematica. Estos errores no han sido tratados por problemas innatos a Mathematica. Mathematica no es un lenguaje de programación, y por lo tanto carece de órdenes que permitan este tipo de controles, aunque cabe esperar que futuras versiones permitan un refinamiento en este sentido.

8-. REFERENCIAS.

1. Picón R. Analytical solutions of plane structures using Mathematica. *International Journal For Numerical Methods in Engineering 2001*; 50-969-992

2. Picón R. *Resistencia de Materiales.* Apuntes de Cátedra. Servicio de publicaciones de la E.T.S.I.I Sevilla, 2000.

3. *Elasticidad y Resistencia de Materiales. 2º Parcial. Colección de exámenes resueltos (1983-1996).* Apuntes de Cátedra. Servicio de publicaciones de la E.T.S.I.I Sevilla , 1997

4. *Resistencia de Materiales (3er Curso de Ingenieros industriales)*: *Problemas con solución; Problemas de exámenes resueltos (97-01).* Apuntes de Cátedra. Servicio de publicaciones de la E.T.S.I.I Sevilla, 2001

5. Strang G. *Linear Algebra* (2nd edn). Academic Press: New York, 1976.

6. Wolfram S. *Mathematica 2.2* (2nd edn). Adison Wesley: Redwood City, CA, 1988.

7. Maeder R. *Programming in Mathematica* (3rd edn). Adison Wesley: Reading, MA, 1997.

8. Hinton P.J. *How to Create Buttons* Conferencia dada el Viernes, 22 de Octubre de 1999. Mathematica Developer's Conference in Champaign, IL.

9. Novak John. M. *Front End Programming*. Conferencia. Mathematica Developer's Conference in Champaign, IL, 1999

10. Blachman Nancy. *Mathematica. Un enfoque práctio.* Ariel Informática. 1993

11. Hinton P.J. *Button Box HOW-TO: Making a graphics generator.* Mathematica Programming Group. Wolfram Rerearch Inc. April 25, 1998.

12. Galvey Todd. *Vertical Applications. Package Design.*

13. Wellin P. *Programming with Mathematica.* Ann Arbor, Michigan 11 Sep, 1993 and Boston, Masachusetts, 18 Sep, 1993.

9-. EJEMPLOS.

A continuación aparecen una serie de ejemplos resueltos [Referencias 3 y 4] que corresponden a ejercicios propuestos en exámenes para los alumnos de la Escuela Superior de Ingenieros de Sevilla.

Estos ejemplos permiten mostrar las posibilidades de los programas, así como mostrar los resultados que se obtienen. Esta colección de ejemplos ha sido elegida con la intención de representar el amplio abanico de posibilidades que ofrece el programa:

1. Ejemplo 1: portico 6

Este ejemplo corresponde al 2º ejercicio propuesto el 15 de Enero de 1993 en la convocatoria de diciembre. Es una estructura hiperestática con grado de hiperestaticidad 2 que presenta las siguiente particularidades:

- Las hiperestáticas son internas
- Las barras tienen tiene diferentes coeficiente β y tiene una restricción al desplazamiento.
- El pandeo de las barras fuera del plano de la estructura está impedido.

En la hoja de resultados del problema podemos ver la resolución analítica completa de la estructura, diseño de la estructura y representación de diagramas de esfuerzos y deformada. En la etapa de diseño cabe destacar que en la primera iteración, el perfil que obtenemos es el último de su tabla (IPE-600) y sin embargo no cumple, en este caso, la aplicación se queda con ese perfil como mejor solución disponible incluso cuando al imponer el criterio de rigidez se siga incumpliendo.

En este ejemplo, al estar impedido el pandeo, se hacen una serie de comentarios sobre cuales son los perfiles más adecuados y se dice cual es el perfil elegido. Estos comentarios son de gran ayuda para el usuario a la hora de entender cuales son los perfiles más adecuados para el estado de cargas que tenemos.

2. Ejemplo 2: portico 11

Este ejemplo corresponde al 4º ejercicio propuesto el 26 de Junio de 1990 en la convocatoria de junio. Las características de esta estructura son:

- Estructura isostática
- Tenemos cargas repartidas y puntuales en una libertad.
- La estructura tiene dos restricciones al desplazamiento.

En la hoja de resultados de esta estructura se puede ver la resolución analítica de la estructura, diseño de la estructura, representación de los diagramas adimensionales (en este caso sólo de fuerzas) y la deformada. En la etapa de diseño la particularidad más interesante de este problema es que tiene dos restricciones al desplazamiento. Este no suele ser un caso habitual, y como podemos ver en la resolución del

problema, aunque uno de los desplazamientos sólo depende de una de las características del material, el otro depende de dos. La solución que da el programa es la solución óptima.

3. Ejemplo 3: portico 14.

Este ejemplo corresponde al 1er problema propuesto el 8 de septiembre de 1995 en la convocatoria de septiembre. Las características de esta estructura son:

- Estructura hiperestática con grado de hiperestaticidad interna 1.
- La estructura tiene una barra sometida a un incremento de temperatura y sobre la estructura actúan cargas concentradas.

En la hoja de resultados podemos ver la resolución analítica de la estructura, el cálculo del desplazamiento horizontal de los puntos 3 y 4 y la representación adimensional (según hiperestáticas y cargas externas) de los diagramas de esfuerzos.

La principal característica de este ejemplo es poder observar que las barras sometidas a un incremento térmico son representadas en rojo, y la dependencia de la incógnita hiperestática, y en consecuencia de los desplazamientos, con la temperatura.

4. Ejemplo 4: portico13

Este ejemplo corresponde al 2º problema propuesto el 8 de Septiembre de 1995 en la convocatoria de septiembre.

Como en los ejemplos anteriores podemos ver la resolución analítica de la estructura, el proceso de diseño y la representación de la deformada.

En la resolución de esta estructura podemos ver que no se han seguido los pasos indicados por el "tutor" del programa. Este caso es curioso porque el óptimo que obtendríamos siguiendo las líneas de elección de perfiles normales es peor que el obtenido eligiendo para la barra de tipo 1 un perfil hueco cuadrado que uno hueco circular aunque la diferencia sea pequeña. Con este ejemplo se pretende ilustrar por un lado la dificultad de sistematizar el cálculo de las estructuras por este método y mostrar cómo funciona el mecanismo de redefinición de perfiles.

5. Ejemplo 5: emparrip98p2

Este ejemplo corresponde al 2º problema propuesto el 8 de junio de 1998 en el 2º parcial.

Representa una estructura isostática, un emparrillado. El motivo de este ejemplo es mostrar un ejemplo de cómo son los resultados que podemos obtener para emparrillados isostáticos.

En la solución podemos ver la resolución analítica de la estructura, la representación de los diagramas adimensionales y dimensionales (en este caso como era de esperar,

coinciden), el cálculo de desplazamientos (y el valor de estos durante el proceso de cálculo; esto se puede hacer cuando no imponemos ninguna restrición), y el proceso de diseño. En el proceso de diseño podemos ver cómo funcionan los algoritmos propios de este tipo de estructuras.

6. Ejemplo 6: emparrij01p1

Este ejemplo corresponde al 1er problema propuesto el 31 de Mayo del 2001 en la convocatoria de junio.

Es un emparrillado con grado de hiperestaticidad 1. Este ejemplo muestra la resolución analítica de emparrillados hiperestáticos. También se muestra la representación adimensional de los diagramas de esfuerzos.

Todos estos ejemplo han sido realizados con las aplicaciones realizadas, siendo sus resultados idénticos (se han usado para las estructuras hiperestáticas las mismas condiciones iniciales) a los obtenidos.