UNIVERSITAT DE BARCELONA

Escola Tècnica Superior d'Enginyers de Camins, Canals i Ports de Barcelona

UNIVERSITAT POLITÈCNICA DE CATALUNYA

TRABAJO FINAL DE CARRERA

Título

Estudio de la evolución de las frecuencias naturales en estructuras sometidas a efectos del daño y la plasticidad.

Autor

JORGE BAÑOS LLOREDO

Tutores

Sergio Horacio Oller Martinez
Horia Alejandro Barbat Barbat
Jairo-Andrés Paredes López

Departamento

Resistencia de Materiales y Estructuras en la Ingeniería (RMEE)

Intensificación

Obras y conocimiento del terreno

Fecha

Julio, 2011

Agradecimientos

A mis directores, Dr. Sergio Oller, Dr. Alex Barbat y Jairo-Andrés Paredes, por proporcionarme una tesina que me ha permitido adquirir nuevos conocimientos tanto en el campo de las estructuras como en el del lenguaje FORTRAN entre otros. Sobre todo a Jairo-Andrés por su asesoramiento, ayuda y guía tanto en el uso y las modificaciones llevadas a cabo en el PLCD, como en la creación del CFYFP. Así como su ayuda en la redacción de esta tesina.

A Xavi Martínez, por proporcionarme las rutinas encargadas de resolver el problema de autovalores y por resolver mis dudas con respecto a esas rutinas aun estando de baja.

A mis profesores de la Universidad de Glasgow Nenad Bicanic y Chris Pearce, por despertar un tardío interés por las estructuras e iniciarme tanto en el mundo de los elementos finitos como en el análisis de las frecuencias naturales de las estructuras.

A mi familia y amigos, por aguantarme durante el desarrollo de este trabajo y por animarme a continuar cuando ha hecho falta. Por último, a mi perro por su compañía y sus consejos.

Resumen

Este trabajo contiene la memoria del proyecto de final de carrera “Estudio de la evolución de las frecuencias naturales en estructuras sometidas a efectos del daño y la plasticidad”. Se presenta el estudio y la implementación de las rutinas necesarias para calcular las frecuencias naturales de una estructura cuando ésta sufre deterioro utilizando dos programas de ordenador, un programa de elementos finitos existente llamada PLCD y un programa encargado de calcular las frecuencias y formas propias de las estructuras, CFYFP (siglas que corresponden a Cálculo de Frecuencias Y Formas Propias). Se incluyen las modificaciones de las rutinas del programa de elementos finitos existente y la implementación de las nuevas rutinas del programa creado para calcular las frecuencias. Todas las rutinas de los dos programas se encuentran en lenguaje FORTRAN.

En el presente documento se describe de forma detallada los pasos seguidos para la implementación del método de cálculo, poniendo especial énfasis en las rutinas del nuevo programa creado llamado CFYFP. Además se incluyen varios ejemplos llevados a cabo con los programas nuevos para estudiar, precisamente, la evolución de las frecuencias naturales de estructuras sometidas a efectos de daño.

En base a los casos estudiados, respecto a la evolución del daño debido a las acciones mecánicas impuestas sobre la estructura se ha podido establecer una relación entre la evolución de las frecuencias y el daño global de la estructura. Finalmente, se extraerán una serie conclusiones de los resultados obtenidos del análisis de las estructuras calculadas a modos de ejemplo y se propondrán unas mejoras en las rutinas utilizadas y futuras líneas de investigación.

Abstract

The following pages contain the work developed for the final degree project: “Estudio de la evolución de las frecuencias naturales en estructuras sometidas a efectos del daño y la plasticidad”, in English: study of the evolution of natural frequencies in structures subjected to the effects of damage and plasticity. After discussing some theoretical aspects, there will be introduced the routines and computer programs needed to compute the natural frequencies of a structure when it is damaged. Two computer programs will be used for this purpouse, the first one called PLCD, an existing finite element software, and a new one named CFYFP, which stands for “Cálculo de Frecuencias Y Formas Propias”, in English: Solving Eigenfrequencies and Eigenvectors. All the modifications done to the finite element program and the implementation of the new routines for the program created for solving the Eigenvalue program are included, as well as the justification for these decisions. All the routines used are written using FORTRAN language.

In this document are described in detail all the steps followed to implement the calculation method, focusing in the new routines for the CFYFP. In addition, several examples solved with the new programs for the study of the evolution of the natural frequencies of structures when have been damaged are also included.

Based on these cases, it has been possible to establish a correlation between the change in the natural frequencies of a structure and the level of global damage. Finally, a series of considerations and conclusions will be extracted from the studied cases, a number of improvements will be proposed and future research approaches too.

Índice

Índice figuras y tablas

Glosario

A lo largo de las siguientes secciones y capítulos se utilizarán una serie de símbolos, algunos de ellos serán particulares del punto en concreto en el que aparezcan y en tal caso serán definidos en el momento oportuno. Pero existen otros que de forma recurrente surgirán durante todo el texto, para ahorrar repeticiones y facilitar el seguimiento, se definirán a continuación.

En las páginas dedicadas a la explicación del daño estructural los símbolos más utilizados son:

w → Continuidad

d → Daño

σ → Tensión

E → Módulo de Young

Ψ → Energía libre de Helmholtz

ε → Deformación

μ → Parámetro de consistencia de daño

En el capítulo de problemas de valores y vectores propios y en el del método de iteración en el subespacio los símbolos más utilizados son:

$\mathbf{K}$ → Matriz de rigidez

$\mathbf{M}$ → Matriz de masa

Φ → Matriz de vectores propios

$\Lambda \rightarrow$ Matriz de valores propios

I $\rightarrow$ Matriz identidad

$\lambda \rightarrow$ Valor propio

$\mu \rightarrow$ Valor del shift

X $\rightarrow$ Matriz de vectores de la iteración donde se almacenan los vectores propios

L $\rightarrow$ Matriz triangular resultante de la descomposición LDL^T

$\mathbf{L^T} \rightarrow$ Matriz traspuesta de L.

D $\rightarrow$ Matriz diagonal de la descomposición LDL^T

p $\rightarrow$ Número de valores propios

Las variables representadas con mayúscula se refieren a una matriz. Si la misma letra es utilizada en minúscula, designa un elemento concreto de dicha matriz.

$k_{ij} \rightarrow$ Se refiere al elemento (i,j) de la matriz de rigidez **K**.

El superíndice "T" sirve para indicar que se trata de una matriz o vector traspuesto, el "-1" se utilizará para señalar que se trata de la matriz inversa.

1. Introducción

1.1 Motivación

El objetivo principal de esta tesina es el estudio de la variación de frecuencias de las estructuras debido a efectos no lineales, como lo son el daño y la plasticidad. En concreto se pretende estudiar la evolución que sufre la frecuencia de una estructura a medida que ésta se va dañando y ver si se puede establecer alguna relación entre el daño y la evolución de las frecuencias naturales. De la misma manera se intentará observar si dicha relación se muestra en los modos de vibración de las citadas frecuencias. El motivo final, aunque no forma parte del trabajo desarrollado en esta tesina, es poder medir en una estructura real las frecuencias naturales, y de este modo obtener el nivel de daño, gracias a la correlación realizada con los programas y estudios desarrollados en el presente trabajo.

1.2 Objetivos

El objetivo final es sentar las bases para poder establecer en una estructura real una correlación entre el nivel de daño y las frecuencias naturales, de este modo se podrá determinar el daño de una estructura, difícil de cuantificar en la práctica, a partir del valor de las frecuencias, que si se pueden obtener de forma sencilla. Por ello en esta tesina se hará el desarrollo teórico y se crearán las rutinas necesarias, además de establecer si es posible relacionar las frecuencias naturales con el nivel de deterioro de la estructura. En futuros trabajos se tendrá que comprobar que los resultados obtenidos son los mismos que los observados en la realidad.

1.3 Metodología aplicada

Se pueden distinguir dos fases en la realización de esta tesina, en la primera se han desarrollado una serie de rutinas en lenguaje FORTRAN para modificar un programa de elementos finitos ya existente, PLCD, y para crear un nuevo programa llamado CFYFP, que calcula las frecuencias y formas propias de las estructuras. El PLCD estudia la respuesta no lineal de los problemas y las modificaciones se han hecho para obtener los datos necesarios para que funcione correctamente el CFYFP. En éste se han creado y reutilizado una serie de rutinas para calcular los valores propios de las estructuras al dañarse a partir de los datos previamente calculados en el primer programa.

En una segunda fase y utilizando estos programas se ha realizado la validación de éstos y el estudio de varios casos: una columna empotrada, la misma columna pero con armadura y una viga biapoyada.

1.4 Contenido de la memoria

El presente trabajo se puede dividir en cinco partes principales: en una primera parte se hace un repaso por los trabajos, estudios e investigaciones previas realizadas en el campo del estudio de la variación de las frecuencias naturales de las estructuras cuando se encuentran dañadas.

En la segunda se hace una introducción a los fundamentos teóricos generales que se ven abordados en este proyecto, como son el análisis no lineal de estructuras, el daño, los problemas de valores y vectores propios y el método de iteración en el subespacio. Esta parte más teórica se encuentra en los primeros capítulos, que van desde el tercer hasta el sexto.

Seguidamente, en el séptimo y octavo capítulo, se muestran todo lo relacionado con los programas utilizados, en este apartado se incluyen lo relacionado con la

compatibilidad entre datos, las rutinas utilizadas y los cambios realizados para el PLCD. Además se explican las nuevas rutinas creadas y sus funciones, así como el trabajo realizado para el nuevo programa que se ha tenido que desarrollar llamado CFYFP. Finalmente se incluyen los pasos y ejemplos realizados para validar el correcto funcionamiento de las nuevas subrutinas y programas.

La tercera parte del trabajo contiene el desarrollo de los casos de estudio, que en este caso es una columna, con y sin armadura y una viga biapoyada que se pueden subdividir a su vez en dos bloques. Por una parte, se procederá a una estimación manual de los resultados esperados para los valores de la primera frecuencia natural utilizando el método energético de Rayleigh y, por otro lado, en el noveno capítulo, se realizará el análisis completo de las estructuras usando los programas descritos en las secciones anteriores.

Finalmente, se procederá a sacar las conclusiones de los resultados obtenidos en los capítulos noveno y décimo. Para terminar, se comentarán posibles mejoras y trabajos futuros que se pueden introducir en el proceso realizado y cambios que deberían ejecutarse, principalmente en los programas para mejorar el funcionamiento. Además de futuras líneas de investigación en el campo de estudio.

2. Trabajos y estudios previos

El hecho de intentar relacionar el daño con las frecuencias naturales y los modos de vibración de las estructura no es algo totalmente nuevo, sin embargo es un campo en el que no se ha profundizado mucho y la mayoría de estudios llevados a cabo tienen un carácter mucho más práctico, realizando en casi todos ensayos de laboratorio para obtener los resultados y sacar conclusiones. A diferencia de lo previamente enunciado, el presente trabajo obtendrá los resultados utilizando un programa de elementos finitos ya existente y un programa creado para calcular las frecuencias propias. A continuación se citarán varios trabajos relevantes llevados a cabo en este campo:

- En 1990 Mannan y Richardson [1] detectaron que los parámetros modales pueden ser útiles para detectar daño en una estructura, a pesar de no ser aptos para la localización del mismo, a menos que eso parámetros se encuentren asociados a frecuencias altas y permitan detectar cambios de rigidez en una zona reducida de la estructura. A pesar de demostrar que con modos altos era posible detectar una disminución de rigidez en un modelo muy simple de masa y resortes, el método demostró no ser muy fiable.
- Un año más tarde, en 1991, Chowdhury [2] intentó sin mucho éxito observar los cambios en las matrices de masa y rigidez obtenidas a partir de información modal experimentalmente, con una placa de acero similar a la usada por Mannan y Richardson.
- En 1992 Fox [3] comparó varias técnicas que utilizan las frecuencias naturales y los modos de vibración como parámetros que permiten localizar daños en las estructuras. Para ello utilizó un programa de elementos finitos y se construyó dos vigas con ambos extremos libres, una de ellas sin defectos y la otra con un defecto ubicado a un 20% de la luz. Se obtuvieron utilizando los modelos experimentales los seis primeros modos de vibración, mientras que con el programa de elementos finitos se encontraron los diez primeros. Fox concluyó que tanto las frecuencias naturales como los modos de vibración son buenos

indicadores de la presencia de defectos o daños, pero que sin embargo no son suficientes para detectar la ubicación de los mismos.

- También en el mismo año, Kam y Lee [4] utilizaron un modelo reducido de matrices de rigidez para determinar y ubicar el daño en una estructura. El método utilizado consistía en discretizar la estructura con un grupo de elementos y suponer la ubicación de la grieta en uno de ellos. Posteriormente se obtenían las frecuencias naturales y los modos de vibración usando pruebas experimentales y se realizaba un análisis estadístico de cada elemento con deterioro supuesto y, usando varios parámetros estadísticos, se ubicaba el elemento que tenía el defecto. Además también consiguieron determinar el tamaño del mismo. Para ello utilizaron una ecuación de equilibrio de energía de deformación. Al igual que los anteriores el modelo experimental utilizado era de acero.
- Pandey y Biswas [5] en 1994 propusieron detectar el daño basándose en los cambios observados en las matrices de flexibilidad obtenidas a partir de los modos de vibración. Los parámetros modales utilizados se obtuvieron tanto de manera analítica como experimental con vigas de acero estructural. De todas maneras, Padey y Biswas señalaron que los cambios en la flexibilidad dependen en gran manera de las condiciones de apoyo de la viga y el método propuesto obtuvo mejores resultados cuando el defecto coincidía con una zona en la que los momentos flectores eran mayores.
- Otro estudio llevado a cabo por Narkis [6] en 1994 propuso un método que permitía calcular las frecuencias naturales de vigas simplemente soportadas con grietas. Para ello modeló la grieta mediante un resorte lineal elástico. Desarrolló la ecuación de movimiento para las vigas usando un modelo de dos vigas unidas por un resorte torsional que modela la grieta. Concluyó que con solamente los dos primeros modos de vibración y las frecuencias naturales asociadas era suficiente para obtener tanto la ubicación de la grieta como su profundidad. Estos datos fueron comparados con un programa de elementos finitos obteniendo resultados aceptables.

- En el año 2004 Vázquez, Suárez y López [7] basándose y continuando un estudio previo llevado a cabo por Pérez y Suárez [8] en 1994 y realizando tanto ensayos con vigas de hormigón como utilizando elementos finitos, determinaron que la comparación entre los modos y frecuencias naturales no eran buenos indicadores de la presencia de daños. Por el contrario que las comparaciones entre matrices de rigidez y de flexibilidad permitían detectar y situar las grietas inducidas en las vigas.
- Por último, Oyarzo-Vera y Chouw [9] en el año 2010, determinaron que para los modos de vibración más bajos es posible observar claramente una variación de la frecuencia debida al daño. En algunos casos particulares consiguieron asociar la variación de la frecuencia con una localización específica del daño, pero en este aspecto los resultados obtenidos no eran concluyentes. Para realizar el estudio utilizaron un modelo físico de una casa de albañilería no reforzada a escala real.

Como se ha podido observar las conclusiones obtenidas de los diferentes estudios previos son diferentes y en algunos casos hasta contradictorias. El presente trabajo no pretende continuar la investigación de ninguno de los casos anteriores sino utilizar un enfoque basado en los elementos finitos para poder obtener una correlación entre el estado de deterioro de una estructura y los cambios en su contenido de frecuencias naturales. Se obtendrán una serie de conclusiones independientes que confirmarán algunas de las observaciones citadas anteriormente.

3. Análisis no lineal de estructuras

La no linealidad en el comportamiento de las estructuras surge cuando no se cumple una relación lineal entre causas y efectos, es decir cuando no se verifica que el material es elástico lineal o que los desplazamientos de la estructura son pequeños. En el caso que nos ocupa en este trabajo, la causa de la no linealidad es el comportamiento del material (no linealidad física). El desarrollo de los análisis no lineales de estructuras ha ido intensamente ligado al desarrollo de los ordenadores y han ido evolucionando a la par. Desde los comienzos con modelos muy sencillos que reproducían el material distinguiendo dos rangos de comportamientos, el elástico y el plástico, hasta la actualidad donde los modelos incorporan la degradación de la rigidez, el deterioro de la resistencia o el estrangulamiento de los lazos de histéresis.

Uno de los procedimientos de análisis más populares es el llamado análisis incremental de fuerzas, debido al bajo tiempo de ejecución y la relativamente fácil interpretación de los resultados. Básicamente consiste en imponer un control de desplazamientos para evitar problemas, como pueden ser los grandes desplazamientos una vez se alcanza la plasticidad. Para hacerlo, se imponen unos desplazamientos incrementales en los nodos de la estructura y se observa su comportamiento. Otra alternativa muy utilizada es la de aplicar una carga variable a la estructura, con la que se obtienen buenos resultados para los desplazamientos últimos, pero que tiende a subestimar la capacidad última de las estructuras.

El PLCD permite establecer ambos controles pero en los ejemplos utilizados para desarrollar esta tesina se ha utilizado el control de los desplazamientos. El proceso de para realizar un análisis no lineal es aproximadamente el siguiente:

- Seleccionar incremento de carga.
- Solución del campo de desplazamientos.
- Actualización de las coordenadas y cálculo de los desplazamientos.
- Cálculo de la deformación.
- Ecuación constitutiva.
- Cálculo fuerza residual.

- Mirar convergencia. Si no se ha logrado, se realiza una nueva iteración para el mismo incremento. Si la solución ha convergido, se procede con el siguiente incremento de carga.
- Una vez haya convergido el último, se acaba.

Un ejemplo del esquema de la resolución de un problema no lineal puede observarse en la figura 3.1 [10]

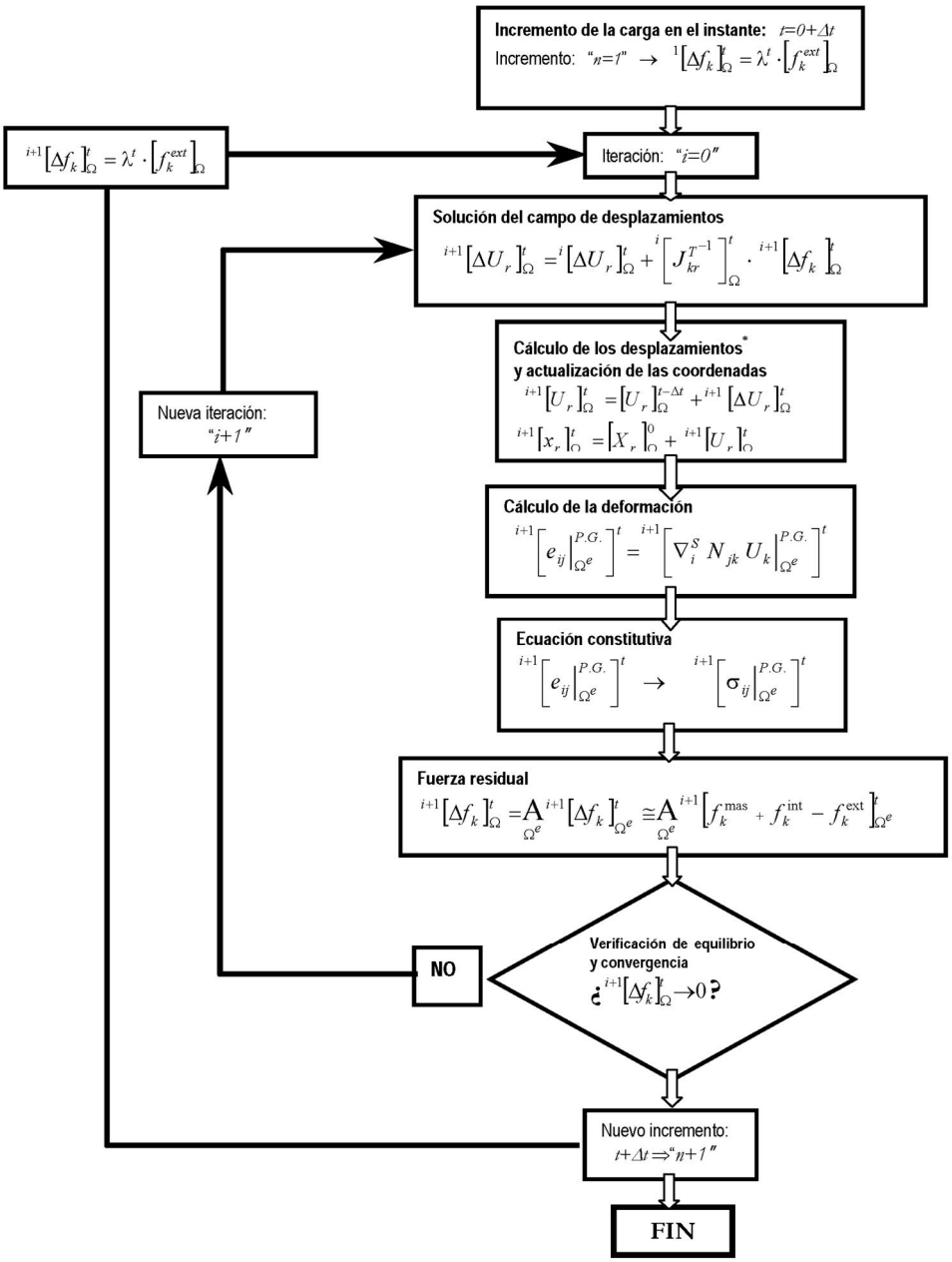


Fig. 1.4.1 Representación esquemática del procedimiento seguido para la resolución de un problema no lineal, en este caso usando Newton-Raphson. *El campo de velocidades y aceleraciones deben calcularse según el método de aproximación de la aceleración, por ejemplo Newmark.

4. Daño estructural

El principal objetivo de esta tesina es poder establecer una relación entre las frecuencias, las formas propias y el daño. Para establecer la relación requerida se usará el resultado del daño dado por el PLCD según [11] con la medida de fuerza. Pero antes de mostrar cómo se calculará se comentarán algunos aspectos generales del daño estructural.

4.1 Daño local

El daño local se expresa con un índice que mide la degradación de rigidez que sufre un punto material de un sólido que se encuentra bajo la acción de fuerzas externas. Este índice deriva del concepto de disipación, se trata de una medida objetiva y normalizada de la disipación total producida por la plasticidad y por la degradación de la rigidez. En nuestro caso el PLCD ya contiene las rutinas encargadas de calcular este parámetro para cada punto de Gauss de la estructura, sus valores pueden consultarse en el fichero de salida post.res para cada paso de carga realizado. Pero no se utilizarán para sacar conclusiones ya que para el objetivo de la tesina será más pertinente realizarlas con un valor del daño global de la estructura.

4.1.1 Introducción – Modelo constitutivo de daño

El hecho de que las estructuras se dañan es algo intuitivo y lógico. Al someter a una estructura a unas determinadas condiciones de carga pueden empezar a aparecer pequeñas fracturas o cavidades en las partes que soportan mayores esfuerzos. Este deterioro del material, lo debilita y disminuye su capacidad de soportar una carga, lo daña. Estos “defectos” son discretos pero tratarlos como tales al realizar un análisis

resultaría en un esfuerzo ingente e inabordable, al menos en la actualidad. De hecho se mantuvo como un problema inasumible hasta el año 1958 en el que Kachanov [12] acuñó el término continuidad para describir el efecto colectivo de estas discontinuidades en un sólido. Teniendo en cuenta la sección de un sólido cualquiera, Kachanov definió inicialmente ωcomo:

$$\omega(\vec{n}) = \frac{s}{s_o} \qquad 0 \leq \omega \leq 1 \qquad \text{Ec. 4.1.1}$$

Siendo S_o la sección inicial sin daño y S la sección perdida por la aparición de fisuras y cavidades al dañarse. Por tanto, ω puede tomar los siguientes valores dependiendo de las circunstancias:

- $\omega(\vec{n}) = 0$, el material no ha sido dañado
- $\omega(\vec{n}) = 1$, el material está completamente dañado
- $0 \leq \omega(\vec{n}) \leq 1$, ω representa el daño sufrido por el material.

Si se considera que el daño tiene un comportamiento isótropico y consecuentemente no importa la dirección, $\omega(\vec{n})$ no depende de la normal, por lo que el estado de daño se puede caracterizar con un valor escalar d, $\omega(\vec{n}) = d$.

4.1.2 Daño isótropo

Para todos los ejemplos y casos presentados en este trabajo se ha decidido que el modelo de daño considerado será el isótropo, esto significa que las fracturas y cavidades que se forman en la estructura se distribuyen de manera uniforme en todas direcciones, es decir el daño es adireccional en cada punto del sólido cuando se forma una fractura (figura 4.1.2.1). A pesar de que este concepto es opuesto a la realidad manifestada en datos experimentales, que demuestran que esto no es cierto y que durante la carga, las microfisuras crecen principalmente en la dirección perpendicular a la máxima tensión (Krajcinovic y Fonseka [13], 1981). Si se decide aproximar el comportamiento del material mediante una formulación continua, se admite como hipótesis que el daño macroscópico direccional (fracturas) proviene de un comportamiento microscópico adireccional de los puntos situados en la zona de daño.

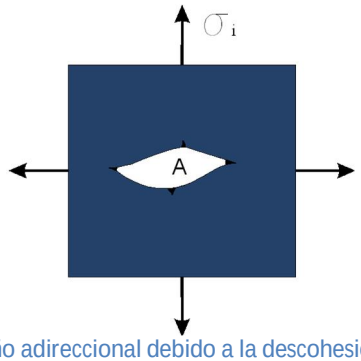


Fig. 4.1.2.1 Daño adireccional debido a la descohesión de un punto

Un concepto estrechamente relacionado con el daño es de la tensión efectiva, tensión calculada sobre la sección que resiste las fuerzas. El uso de la tensión efectiva para caracterizar el daño se debe a la gran dificultad existente en estimarlo usando el área perdida a causa de las fisuras y cavidades, debido al desconocimiento existente sobre su verdadera geometría. Si consideramos un caso uniaxial de una fuerza F aplicada en una determinada sección, la tensión será $\sigma = F/S$. Si la sección resultara dañada (d), el área efectiva ($\bar{S}$) que resistiría puede escribirse como $\bar{S} = S - S_D = S\,(1-d)$, en cuyo caso la nueva tensión efectiva ($\bar{\sigma}$) será $\bar{\sigma} = \frac{\sigma}{(1-d)}$. Este concepto de tensión efectiva en conexión con la hipótesis de equivalencia de deformaciones fue propuesta por primera vez en el año 1978 por Lemaitre-Chaboche [14] de la siguiente manera: "… la deformación asociada a un estado dañado bajo una tensión aplicada σ es equivalente a la deformación asociada con el estado no dañado sometido a una tensión efectiva σ_o". En la figura 4.1.2.2 se muestra una representación esquemática de esta cita.

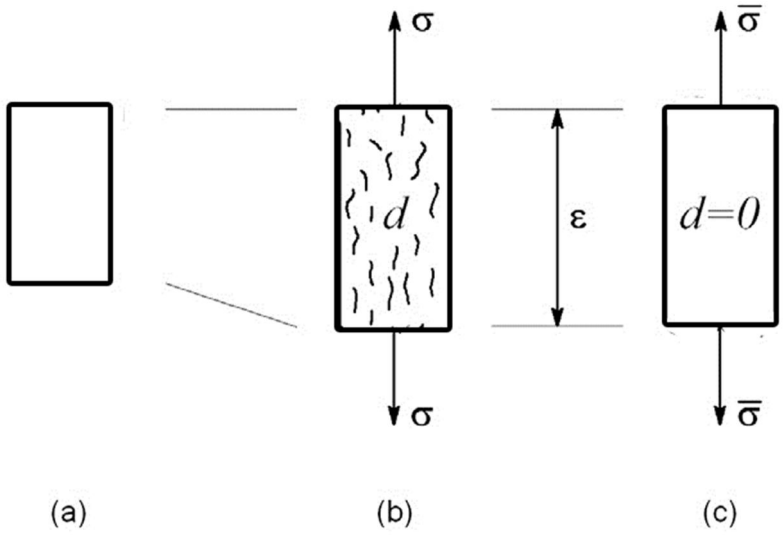


Fig. 4.1.2.2 Representación esquemática de la hipótesis de tensión efectiva

Se asume que cualquier deformación de un material dañado se representa con la ley constitutiva del material no dañado, pero sustituyendo la tensión normal por la tensión efectiva. Para un caso uniaxial, esto puede escribirse: $\sigma = (1-d)\bar{\sigma} = (1-d)\,E\,\varepsilon$. Donde E representa el módulo de Young. Como se verá con más detalle un poco más adelante el daño aparecerá cuando se supera un determinado umbral inicial de daño,

$$d = 0 \begin{cases} \sigma < \sigma_0 \\ \varepsilon < \varepsilon_0 \end{cases}$$

En caso de que se produzca una descarga, $\varepsilon < 0$ y el daño será nulo, por tanto, la descarga se producirá hasta el origen para un determinado daño. La recarga sucesiva seguirá el mismo camino que la anterior descarga hasta que se llegue al umbral de daño otra vez. Todo ello puede verse en la figura 4.1.2.3

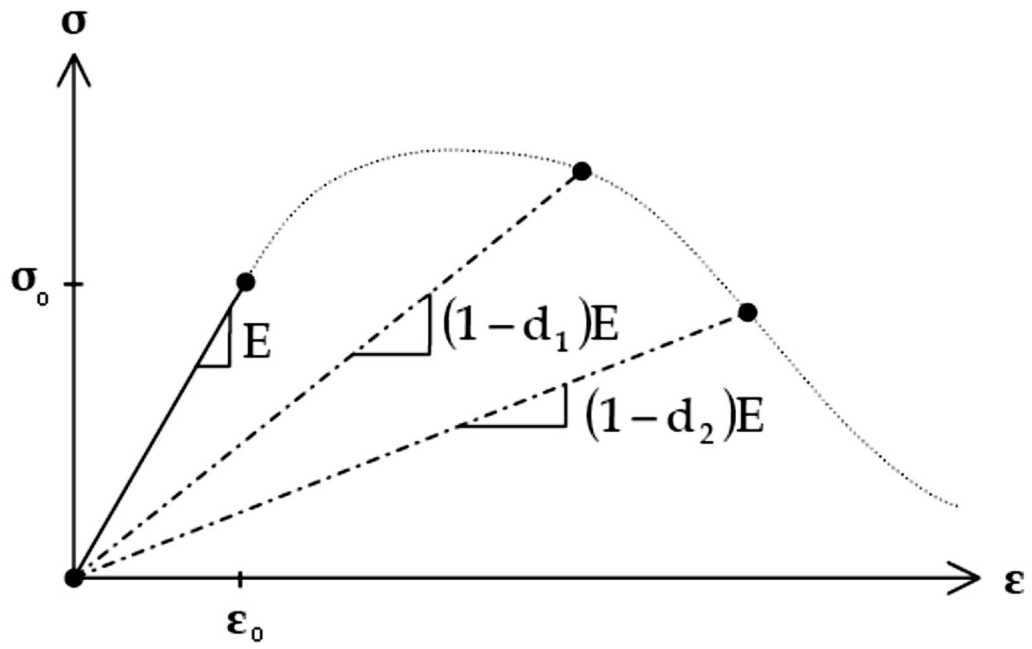


Fig. 4.1.2.3 Representación del umbral de daño y de varias descargas y recargas.

Los efectos del daño solamente afectan a las propiedades elásticas del material, mientras la plasticidad se desarrolla como consecuencia de un crecimiento irrecuperable en la deformación plástica. Estos dos fenómenos pueden actuar a la vez y en los materiales se observan ambos, pérdida de elasticidad por daño y aumento de la deformación inelástica por plasticidad. La principal diferencia entre el daño y un

modelo constitutivo plástico es que en el primero no se producen deformaciones plásticas irreversibles, toda deformación se recupera en caso de descarga, aunque las trayectorias de carga-descarga, a diferencia de lo que sucede en plasticidad, no son paralelas (figura 4.1.2.4).

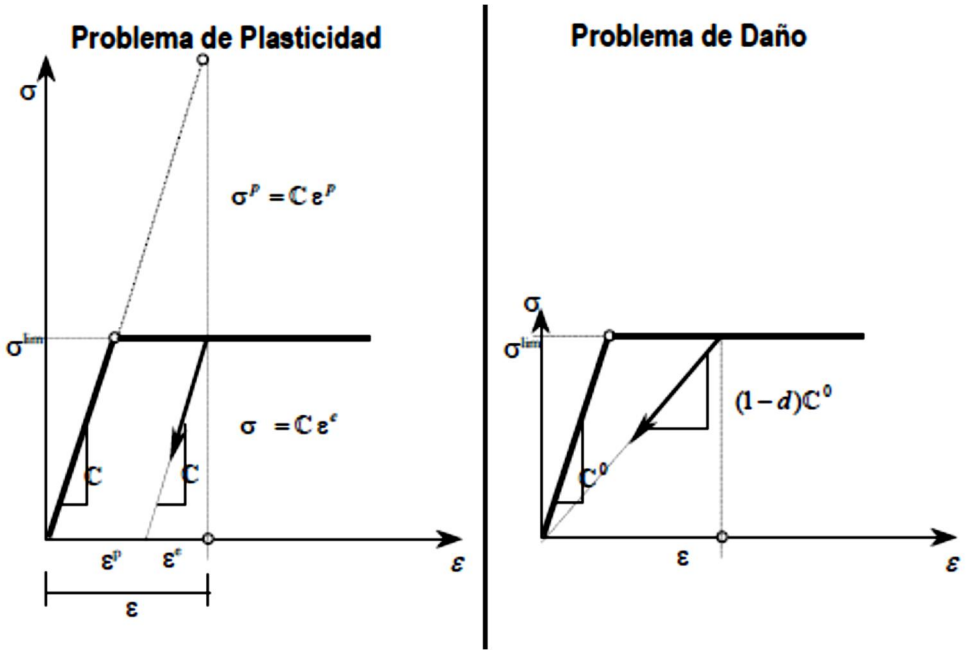


Fig. 4.1.2.4 Diferencia entre daño y plasticidad [15]

4.1.3 Energía libre de Helmhotz

Las rutinas del PLCD calculan el daño de dos formas distintas, medida en fuerzas y en energía. La formulación detallada a continuación se basa en la propuesta por Simó y Ju [16] en 1987, que proporciona un modelo constitutivo que a pesar de su simplicidad, es capaz de reproducir los comportamientos no lineales como son la pérdida de rigidez, endurecimiento y reblandecimiento, que es lo que usa el PLCD. La energía libre de Helmholtz por unidad de volumen para el caso de un modelo de daño isótropo como el que nos ocupa, a temperatura constante es:

$$\Psi = \Psi(\varepsilon; d) = (1-d)\Psi_o(\varepsilon) \qquad \text{Ec. 4.1.3.1}$$

donde $\Psi_o(\varepsilon)$ es la energía libre de Helmholtz elástica inicial del material no dañado. En el caso de tratar pequeñas deformaciones, como será nuestro caso, se puede definir esta energía usando el tensor constitutivo elástico del material no dañado ($\mathbb{C}_o$) como:

$$\Psi_o(\varepsilon) = \frac{1}{2}\varepsilon : \mathbb{C}_o : \varepsilon \qquad \text{Ec. 4.1.3.2}$$

Para el caso de problemas estables térmicamente, es válida la siguiente forma de la desigualdad de Clausius-Plank,

$$\Xi = \left(\sigma - \frac{\partial \Psi}{\partial \varepsilon}\right) : \dot{\varepsilon} - \frac{\partial \Psi}{\partial d}\dot{d} \geq 0 \qquad \text{Ec. 4.1.3.3}$$

Esta expresión de la potencia disipativa permite realizar las siguientes afirmaciones:

- La ecuación 4.1.3.3 se cumple para cualquier variación en el tiempo de ε.. Esta condición proporciona la siguiente ley constitutiva hiperelástica para el problema del daño escala:

$$\sigma = \frac{\partial \Psi}{\partial \varepsilon} \quad , \quad \frac{\partial \Psi}{\partial d} = -\Psi_o \leq 0 \;\rightarrow\; -\Psi_o \text{ conjugada de } d \qquad \text{Ec. 4.1.3.4}$$

- Teniendo en consideración esta ley constitutiva, el valor de la disipación del modelo de degradación es:

$$\Xi = \Psi_o \dot{d} \geq 0 \qquad \text{Ec. 4.1.3.5}$$

Finalmente, teniendo en consideración la ecuación 4.1.3.4 la ecuación constitutiva puede escribirse de la siguiente manera

$$\sigma = \frac{\partial \Psi}{\partial \varepsilon} = (1-d)\frac{\partial \Psi_o}{\partial \varepsilon} = (1-d)\mathbb{C}_o : \mathcal{E} \qquad \text{Ec. 4.1.3.6}$$

Las características de la ecuación constitutiva presentada en la ecuación 4.1.3.6 son las siguientes:

- Es un modelo isótropo, ya que las propiedades mecánicas del material sólo están afectadas por un escalar.
- La integración de la ecuación constitutiva es explícita.
- La ecuación 4.1.3.6 puede descomponerse en una forma aditiva de tensiones elásticas e inelásticas, tal y como se muestra en las siguiente figura (fig.4.1.3.1)

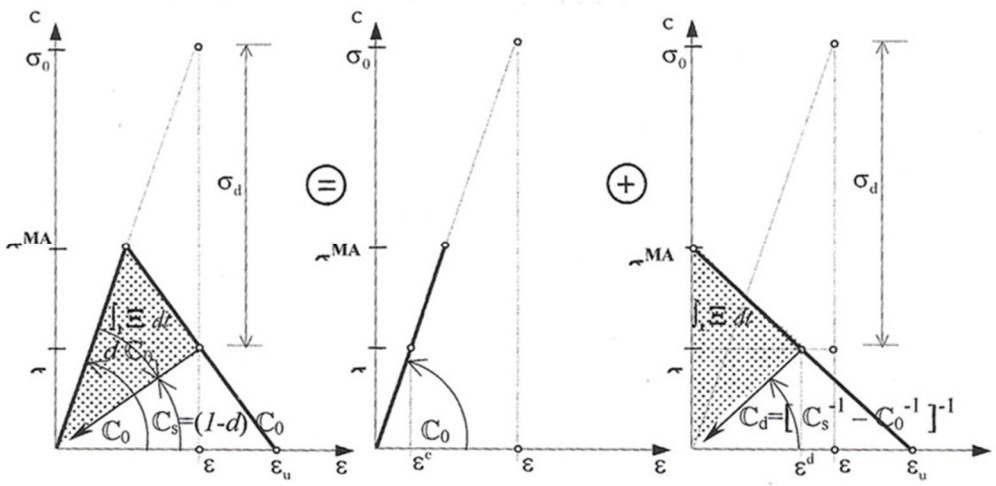


Fig. 4.1.3.1 Representación esquemática del modelo de daño uniaxial [10].

4.1.4 Umbral de daño

Como se ha visto anteriormente el daño aparece cuando se supera cierto umbral, éste es el que nos permite distinguir entre un estado de comportamiento dentro del régimen elástico y otro en el que se produce un proceso de degradación de las propiedades físicas del material. La función de umbral de daño depende del tipo de material y se define de la misma manera que en problemas de plasticidad,

$$F(\sigma_o) = f(\sigma_o) - c(d) \leq 0 \qquad \text{Ec. 4.1.4.1}$$

Donde $f(\sigma_o)$ es una función del tensor de tensiones y $c(d)$ es la función que define la posición del umbral de daño. Esta función permite, además de establecer el inicio del comportamiento dañado, definir los estados de carga, descarga y recarga. Se trata de una función escalar, positiva y para un estado no deformado debe ser nula. El valor inicial del umbral de daño $c(d^0) = c^{max} = \sigma^{max}$ es una propiedad del material y está relacionado con su resistencia a compresión según la función de umbral elegida. El daño ocurre cuando el valor de $f(\sigma_o)$ es igual o mayor que $c^{max} = \sigma^{max}$ por primera vez, la ecuación 4.1.4.1 puede expresarse también de la forma:

$$F(\sigma_o) = G[f(\sigma_o)] - G[c(d)] \leq 0 \qquad \text{Ec. 4.1.4.2}$$

Donde G[·] es una función escalar, invertible, positiva y de derivada positiva y monótona creciente.

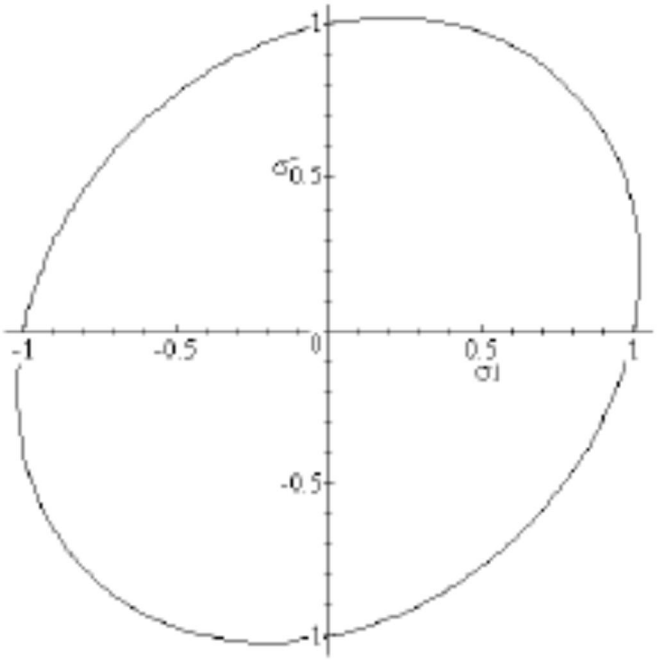


Fig. 4.1.4.1 Umbral de daño en el espacio de tensiones principales, con σ_3=0, según Simó y Ju [16].

4.1.5 Evolución de la variable interna de daño

Cuando intervienen variable internas es necesario definir su ley de evolución, en el caso del problema de daño, ésta viene dada por:

$$\dot{d} = \dot{\mu}\frac{\partial \mathrm{F}(\sigma_o;q)}{\partial[f(\sigma_o)]} = \dot{\mu}\frac{\partial G[f(\sigma_o)]}{\partial[f(\sigma_o)]} \qquad \text{Ec. 4.1.5.1}$$

Donde μ es un escalar no negativo llamado parámetro de consistencia de daño, cuyo equivalente en plasticidad es el factor de consistencia plástico λ, y se usa para definir las condiciones de carga, descarga y recarga a través de las condiciones de Kuhn-Tucker, que para problemas con restricciones unilaterales serán:

$$\dot{\mu} \geq 0\,; \qquad \mathrm{F}(\sigma_o;q) \leq 0\,; \qquad \mu \cdot \mathrm{F}(\dot{\sigma_o};q) = 0 \qquad \text{Ec.4.1.5.2}$$

En el caso de que el valor de $\mathrm{F}(\sigma_o;q) < 0$ el criterio de daño no se verifica y para que se cumplan las condiciones de Kuhn-Tucker necesariamente debe ocurrir que $\dot{\mu} = 0$. Esto nos lleva a deducir que la variación temporal del daño debe ser nula $\dot{d} = 0$ y, por tanto, el material no se dañará y se mantendrá en régimen elástico.

Al igual que en la teoría de plasticidad, la magnitud del factor de consistencia surge de imponer la condición de consistencia de daño, de ésta y de las propiedades de la función $\mathsf{G}[\cdot]$ se tiene que,

$$\mathrm{F}(\sigma_o; q) = 0 \rightarrow G[f(\sigma_o)] = G[c(d)] \rightarrow f(\sigma_o) = c(d) \rightarrow \frac{\partial G[f(\sigma_o)]}{\partial [f(\sigma_o)]} = \frac{\partial G[c(d)]}{\partial [c(d)]} \quad \text{Ec.4.1.5.3}$$

De la condición de permanencia sobre la superficie umbral de daño se deduce que,

$$\dot{F}(\sigma_o; q) = 0 \rightarrow \frac{\partial G[f(\sigma_o)]}{\partial [f(\sigma_o)]} \dot{f}(\sigma_o) - \frac{\partial G[c(d)]}{\partial [c(d)]} \dot{c}(d) = 0 \rightarrow \dot{f}(\sigma_o) = \dot{c}(d) \qquad \text{Ec.4.1.5.4}$$

Observando la variación temporal de $\dot{G}[f(\sigma_o)]$ y haciendo una analogía de la ley de evolución de la variable interna $\dot{d}$ (ec. 4.1.5.1) se deduce el parámetro de consistencia de daño como:

$$\dot{d} \equiv \dot{G}[f(\sigma_o)] \rightarrow \dot{\mu} = \dot{f}(\sigma_o) \qquad \text{Ec. 4.1.5.5}$$

Si se desarrolla más el parámetro de consistencia, puede llegar a escribirse como,

$$\dot{\mu} = \dot{f}(\sigma_o) = \dot{c}(d) = \frac{\partial f(\sigma_o)}{\partial \sigma_o} : \dot{\sigma_o} = \frac{\partial f(\sigma_o)}{\partial \sigma_o} : \mathbb{C}_0 : \dot{\varepsilon} \qquad \text{Ec. 4.1.5.6}$$

Integrando en el tiempo la variación temporal de la variable de daño, se obtiene la siguiente expresión explícita para representar el daño en cualquier punto del sólido,

$$d = \int_t \dot{d}\, dt = \int_t \dot{G}[f(\sigma_o)]\, dt = G[f(\sigma_o)] \qquad \text{Ec. 4.1.5.7}$$

Si se sustituye en la ecuación de la disipación de la energía (ec.4.3.5), se puede deducir que la expresión que describe la evolución temporal de la disipación es,

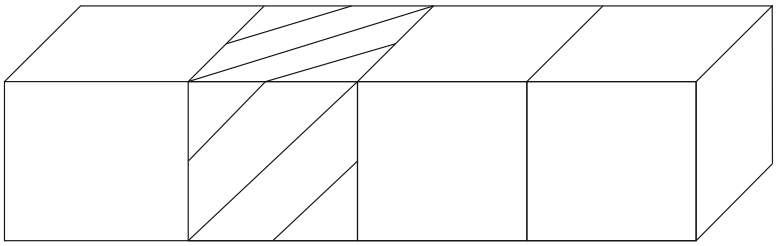
$$\Xi = \Psi_o \cdot \dot{d} = \Psi_o \cdot \dot{G}[f(\sigma_o)] = \frac{\partial G[f(\sigma_o)]}{\partial [f(\sigma_o)]} \frac{\partial f(\sigma_o)}{\partial \sigma_o} : \mathbb{C}_0 : \dot{\varepsilon} \qquad \text{Ec. 4.1.5.8}$$

Finalmente, de las definiciones anteriores se obtiene que el umbral de daño *c* en un tiempo s=*t* es,

$$c = \max\{c^{max}, \max\{f(\sigma_o)|_s\}\}, \forall\, 0 \le s \le t \qquad \text{Ec. 4.1.5.9}$$

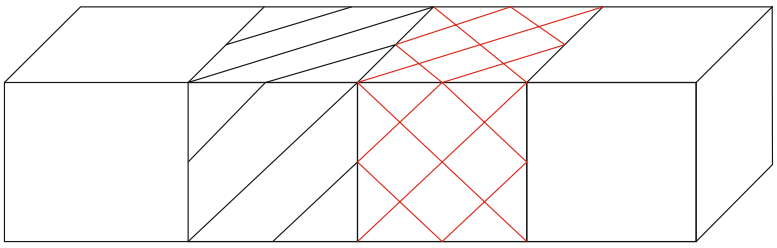
4.2 Daño global

A pesar de que como se ha comentado anteriormente el PLCD proporciona dos índices de daño global, uno medido en energía y otro en fuerzas, se ha decidido utilizar otro diferente [11] que se procederá a introducir a continuación. La razón de esta decisión se debe a que los índices usados con anterioridad no mostraban exactamente el daño global, sino que representaban el daño medio de la zona dañada, pero no proporcionaba un valor adecuado del daño global de la estructura. Podrían mejorarse si indicarán que proporción de estructura está dañada. Este hecho, la falta de un índice más global, provocaba que la variable de daño pudiera subir o bajar, cuando se ha visto que el daño global tiene que crecer o mantenerse pero no puede disminuir. Para explicarlo mejor se ha realizado el siguiente esquema (figura 4.2.1) se muestra la forma de establecer el daño que tenía originalmente el PLCD implementada.



En un determinado momento del análisis, tenemos un elemento de la estructura dañado. Con un daño de 0.40, por poner un ejemplo. Este valor se ha obtenido de la siguiente manera:

$$D_{global} = \frac{dvol \cdot \sum_{i=1}^{n} d_i}{n \cdot dvol} = \frac{\sum_{i=1}^{n} d_i}{n}$$



En otro momento tenemos que el primer elemento mantiene el daño, pero el de al lado se ha dañado ligeramente, el daño del primero seguiría siendo 0.40 mientras que el segundo tendría un daño del 0.20. Al calcular el daño global de la forma anterior tendríamos que el daño global ha bajado, ya que el volumen afectado es el doble pero el daño sumado no será el doble ni superior.

Fig. 4.2.1 Esquema del cálculo del índice de daño global mediante medida energética, utilizada por el PLCD, en él se muestra que el año podría no ser siempre creciente.

Se han modificado varias rutinas para cambiar la manera de calcular el daño en medida en fuerza, el nuevo parámetro de daño parte de la base que al imponer un determinado desplazamiento a una estructura se producirá una cierta reacción F_o. Esta reacción elástica lineal es el valor teórico que tendría la reacción de no producirse daño en la estructura. Pero como llegado un determinado umbral éste se producirá, la reacción real será F_i, con $F_i < F_o$ para cualquier daño producido. Todo ello se muestra en la figura 4.2.2 y en la ecuación 4.2.1

$$D_{global} = 1 - \frac{F_i}{F_o} = 1 - \frac{K_i}{K_o} \qquad \text{Ec. 4.2.1}$$

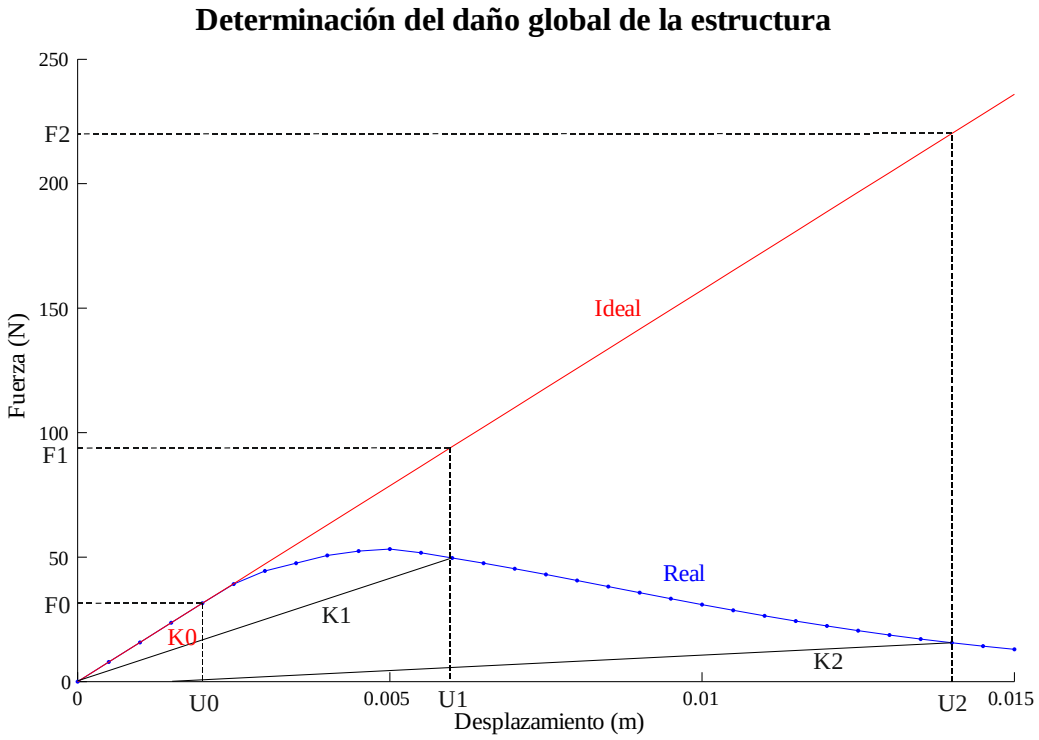


Fig. 4.2.2 Representación del cálculo del daño global de una cubo de hormigón.

El proceso de cálculo no lineal consiste en evaluar las deformaciones generalizadas seccionales correspondientes a los desplazamientos, tal que con éstas se pueden calcular las deformaciones en cada punto del sólido mediantes la siguiente ecuación:

$$\boldsymbol{e} = \begin{Bmatrix} e_x \\ \gamma_{xy} \\ \gamma_{xz} \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x} \\ \dfrac{\partial u}{\partial z} + \dfrac{\partial w}{\partial x} \end{Bmatrix}$$

y las tensiones $\boldsymbol{\sigma}$ mediante el modelo constitutivo definido en cada caso. Conocida la tensión verdadera en cada punto se recompone la tensión en las coordenadas generalizadas. De esta manera se puede exigir que se cumpla la siguiente expresión: $\hat{\mathcal{R}} = 0 = \hat{M} \cdot \hat{\ddot{U}} + \hat{f}_{int} - \hat{f}_{ext}$ y encontrar el daño en base a la relación de las fuerzas en el caso elástico y en el caso que se ha producido una degradación de la rigidez. Para información más detallada sobre la forma de calcular el daño puede consultarse la referencia [11].

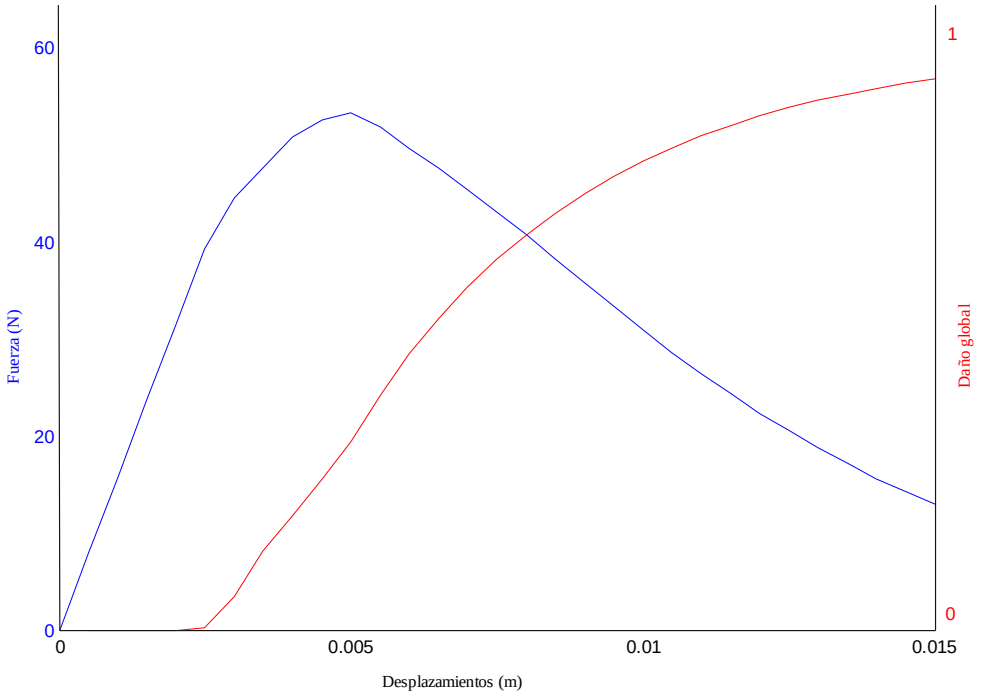


Fig. 4.2.3 Gráfico en el que se muestra la relación fuerza-desplazamientos y el daño global-desplazamientos del cubo de hormigón.

5. Introducción a los problemas de valores y vectores propios

Los problemas de valores y vectores propios son problemas dinámicos en los que no se tienen en cuenta las fuerzas externas y la vibración es libre. Por tanto, los resultados son función de la propia estructura, solamente se tienen en cuenta la masa y la rigidez. Antes de hablar de los algoritmos y los métodos de cálculo, es importante presentar algunas consideraciones básicas para la solución de este tipo de problemas.

5.1 Introducción

El problema más simple que podemos encontrar es el que corresponde a los valores propios estándar,

$$\mathbf{K}\boldsymbol{\phi} = \boldsymbol{\lambda}\boldsymbol{\phi} \qquad \text{Ec. 5.1.1}$$

Donde **K** es la matriz de rigidez de orden *n* de un elemento finito o del ensamblaje de varios elementos. Ha de ser semidefinida positiva o definida positiva. Existen *n* valores propios, con sus correspondientes vectores propios, que satisfarán la ecuación (la notación del i-ésimo par propio es λ_i, Φ_i). La solución para un número *p* de pares propios se escribe,

$$\boldsymbol{K\Phi} = \boldsymbol{\Phi\Lambda} \qquad \text{Ec. 5.1.2}$$

Donde Φ es una matriz de orden *n x p* con las columnas iguales a los *p* vectores propios y Λ es una matriz diagonal cuadrada de orden *p* que contiene los valores propios.

Pero en la realidad, no es la ecuación 5.1.1 la que se tiene que resolver en los problemas del cálculo de las frecuencias naturales de una estructura, sino un caso más general,

$$\boldsymbol{K\Phi} = \lambda \boldsymbol{M\Phi} \qquad \text{Ec. 5.1.3}$$

Donde **M** representa la matriz de masa que resulta de ensamblar las matrices de masa de los elementos y normalmente se trata de una matriz semidefinida positiva. De manera análoga a como se ha hecho en la ecuación 5.1.2, la solución para *p* pares propios se puede escribir,

$$\boldsymbol{K\Phi} = \boldsymbol{M\Phi\Lambda} \qquad \text{Ec. 5.1.4}$$

El problema generalizado se convierte en la versión reducida automáticamente cuando la matriz de masa es igual a la matriz identidad.

5.2 Propiedades de los vectores propios

Los métodos utilizados para solucionar esta clase de problemas se basan en las propiedades fundamentales de las matrices y las de los valores y vectores propios. A continuación se repasarán las principales propiedades según Bathe [17].

La solución de problema generalizado (Ec. 5.1.4) tiene *n* valores propios $\lambda_1,\ldots,\lambda_n$ y sus correspondientes vectores propios $\Phi_1,\ldots,\Phi_n$, cada par (λ_i,Φ_i) de los cuales satisface la ecuación 5.1.3. Para un par cualquiera tenemos,

$$\boldsymbol{K\Phi_i} = \lambda_i \boldsymbol{M\Phi_i}\,; \quad i = 1, \ldots, n \qquad \text{Ec. 5.2.1}$$

De esta ecuación, se puede extraer que si el vector $\lambda_i\mathbf{M\Phi_i}$ se establece como un vector de carga **R** en la ecuación **KU = R**, entonces **U =** $\mathbf{\Phi_i}$. Esta idea sugiere el uso de algoritmos estáticos para el cálculo de los vectores propios, de hecho el algoritmo de descomposición $\mathbf{LDL^T}$ es una parte importante del proceso resolutivo. La ecuación 5.2.1 también muestra que un vector propio se define sólo con un múltiplo de sí mismo,

$$\boldsymbol{K}(\alpha\,\boldsymbol{\Phi_i}) = \lambda_i \boldsymbol{M}(\alpha\boldsymbol{\Phi_i})\;;\; \boldsymbol{\alpha} \neq 0 \qquad \text{Ec. 5.2.2}$$

Por tanto, si $\boldsymbol{\Phi_i}$ es un vector propio, $(\alpha\boldsymbol{\Phi_i})$ será también vector propio, y podemos decir que un vector propio estará definido por su dirección en el espacio *n*-dimensional considerado. Aunque no sea necesario, cuando se haga referencia a vectores propios, no solamente serán aquéllos que satisfagan la ecuación 5.1.1 sino los que también cumplan la siguiente relación: $\boldsymbol{\Phi_i^T M \Phi_i = 1}$. Un vector que cumple la última relación se llama vector M-ortonormalizado u ortonormalizado por la masa. Además, los vectores propios también cumplen una relación similar con la matriz de rigidez, son vectores K-ortogonales, es decir cumplen: $\boldsymbol{\Phi_i^T K \Phi_i = \Lambda}$.

5.3 Polinomio característico

Una propiedad fundamental de los valores propios del problema $\boldsymbol{K\Phi = \lambda M\Phi}$ es que son raíces del polinomio característico,

$$p(\lambda) = \det(\boldsymbol{K} - \lambda\boldsymbol{M}) \qquad \text{Ec. 5.3.1}$$

Esta propiedad es consecuencia de la relación básica mostrada en la ecuación 5.2.1, que puede ser reescrita como

$$(\boldsymbol{K} - \lambda_i\boldsymbol{M})\boldsymbol{\Phi_i} = 0 \qquad \text{Ec. 5.3.2}$$

Podemos observar que la ecuación 5.2.1, dado que la matriz $\mathbf{K} - \lambda_i\mathbf{M}$ es singular, solamente puede ser satisfecha cuando $\boldsymbol{\Phi_i}$ es no trivial, cuando el vector es diferente del vector nulo.

5.4 Cambio (Shifting)

El *shifting* es un importante procedimiento usado ampliamente en la solución de valores y vectores propios, su función es acelerar los cálculos del problema. Dado el

problema $\boldsymbol{K\Phi} = \lambda \boldsymbol{M\Phi}$ (Ec. 5.1.3), el *shifting* consiste en transformar **K** (*shift*ρ en **K**) haciendo el siguiente cambio,

$$\boldsymbol{K'} = \boldsymbol{K} - \rho \boldsymbol{M} \qquad \text{Ec. 5.4.1}$$

Ahora el problema a considerar se ha transformado en el siguiente

$$\boldsymbol{K'\psi} = \mu \boldsymbol{M\psi} \qquad \text{Ec. 5.4.2}$$

Que podemos reescribir (Ec. 5.4.3) para ver la relación existente entre los valores y vectores propios del problema original (Ec. 5.1.3) y el transformado (Ec. 5.4.2). Teniendo en cuenta que $\gamma = \rho + \mu$.

$$\boldsymbol{K\psi} = \gamma \boldsymbol{M\psi} \qquad \text{Ec. 5.4.3}$$

De hecho la ecuación 5.4.3 es la misma que el problema original (Ec. 5.1.3), y como la solución del problema es única, tenemos que $\lambda_i = \rho + \mu_i$ y que $\Phi_\mathbf{i} = \psi_\mathbf{i}$. Dicho de otra manera, los vectores propios de las ecuaciones 5.4.3 y de 5.1.3 son iguales, pero los valores propios han disminuido en un factor ρ. El *shift* se suele utilizar cuando el algoritmo utilizado para resolver el problema no está diseñado para calcular valores propios de valor cero.

Se ha empezado la sección diciendo que el *shift* permite acelerar los cálculos, es decir, mejorar el ratio de convergencia entre valores propios, sobre todo entre los valores propios más pequeños de un problema. Dados $\lambda_1 < \lambda_2$, el vector iterativo converge con un ratio λ_1 / λ_2 hacia el vector propio $\Phi_\mathbf{1}$. Dependiendo de la relación entre los valores propios, el ratio puede ser muy lento (valores cercanos a 1) o extremadamente rápido (relación prácticamente igual a 0). Por ejemplo, sean λ_1=2040 y λ_2 = 2100 los dos primeros valores propios del problema $\mathbf{K\Phi} = \lambda\mathbf{M\Phi}$. Si aplicamos un *shift* de valor 2000 y teniendo en cuenta lo explicado anteriormente en esta sección, tenemos que los dos primeros valores propios son λ'_1= 40 y λ'_2 = 100 y que los ratios de convergencia son:

$$\frac{\lambda_1}{\lambda_2} = \frac{2040}{2100} = 0.9714 \qquad\qquad \frac{\lambda'_1}{\lambda'_2} = \frac{40}{100} = 0.40$$

Otra aplicación del *shift* es que permite la posibilidad de buscar valores propios por encima de un valor determinado.

5.5 Secuencia *Sturm*

La propiedad Sturm se utiliza ampliamente durante el proceso de solución de problemas de valores y vectores propios. En determinadas condiciones el uso de solamente esta propiedad permite resolver los problemas, pero eso no es posible cuando tratamos con la forma generalizada. En estos últimos casos, la propiedad de la secuencia Sturm se utiliza como herramienta auxiliar.

Según Bathe [17]: "Si para un valor del shif=μ, la factorización Gaussiana del problema generalizado $\mathbf{K}-\mu\mathbf{M}$ en la forma $\mathbf{LDL}^{\mathbf{T}}$ se puede obtener. Entonces el número de elementos negativos en la diagonal D es igual al número de valores propios con un valor menor que μ". Esta propiedad se usa para realizar la comprobación de que efectivamente los *n* valores encontrados se corresponden con los *n* primeros valores, ya que el número de valores negativos de la diagonal debería coincidir con el número de valores buscados.

6. Método de iteración en el subespacio

El método de iteración en el subespacio también conocido como SIM (Subspace Iteration Method) se usa comúnmente en la ingeniería para la resolución de problemas de valores y formas propias. Este método fue desarrollado por Klaus-Jürgen Bathe [17] y se trata de unos de los métodos más efectivos, consiste en:

- Establecer q vectores iterativos iniciales, con la condición que $q>p$ (siendo p el número de valores y vectores propios a calcular).
- Usar la inversión simultánea iterativa en los q vectores y el análisis Ritz para extraer las mejores aproximaciones a los valores y formas propias de los vectores q.
- Una vez lograda la convergencia en las iteraciones, se usa la secuencia Sturm para verificar que los valores calculados son efectivamente los que se buscaban.

El método recibe este nombre porque equivale a iterar en un subespacio de q dimensiones. Una de las grandes ventajas de este método frente a otros es que con unas pocas iteraciones ya se observan buenas aproximaciones a los resultados esperados. A continuación, se describirán la teoría básica y los pasos iterativos del método, finalmente, se presentará el programa encargado de realizar los cálculos de los valores y formas propias, que se basa en la teoría presentada en este capítulo.

6.1 Consideraciones previas

El objetivo básico es encontrar los p menores valores propios y sus correspondientes vectores, que satisfacen

$$\boldsymbol{K\Phi = M\Phi\Lambda}$$ Ec. 5.1.4

Donde Λ= diag(λ_i) y Φ=[Φ_1,…,Φ_p]

La idea fundamental del método se basa en una propiedad de los vectores propios, éstos forman una base **M-ortonormal** de los subespacios menos dominantes de las matrices **K** y **M**, que se denomina E_∞. La solución de la iteración con *p* vectores linealmente independientes se puede considerar como una iteración en un subespacio. Los vectores iniciales representan un subespacio E_1, y las sucesivas iteraciones lo modifican hasta llegar a un subespacio final E_∞, dentro de una tolerancia fijada. El número de iteraciones final dependerá la similitud, o la diferencia, entre el primer subespacio (E_1) y el final (E_∞), y no en las similitudes de cada vector iterativo con el vector de valores propios. Es en este aspecto donde reside la ventaja de este método, ya que es más sencillo encontrar un subespacio *p*-dimensional cercano a E_∞ que encontrar *p* vectores, cada uno de los cuales ha de ser cercano al vector propio. Además la convergencia del subespacio es todo lo que se necesita para la convergencia total, y no la convergencia de cada vector de la iteración.

6.2 Vectores iterativos iniciales

El primer paso y uno de los más importantes, es determinar una serie de vectores iniciales para empezar a iterar ($\mathbf{X_1}$). Es importante escoger estos vectores de manera adecuada para reducir el número de iteraciones. Existen serie de casos especiales donde la convergencia a la solución exacta se produce en una sola iteración; por ejemplo: en el caso que se pueda realizar condensación estática o si las matrices de rigidez y masa son ambas diagonales.

Lamentablemente, en los casos prácticos, generalmente no se encuentran matrices con estas propiedades, pero la observación de estos casos especiales ha permitido desarrollar una estrategia para encarar los casos generales.

Los vectores iniciales deben escogerse de tal manera que afecten en mayor medida a los grados de libertad asociados a valores de masa grandes y rigideces pequeñas. Un algoritmo que se ha demostrado adecuado para la elección de estos vectores es el siguiente [17]:

"La primera columna de $\boldsymbol{M \cdot X_1}$ *es simplemente la diagonal de* $\boldsymbol{M}$*. Las columnas restantes, excepto la última, serán vectores unitarios de valor +1 en el grado de libertad con el menor ratio* $\boldsymbol{k_{ii}/m_{ii}}$*. La última columna es un vector aleatorio."*

Este primer subespacio será solamente una aproximación al E_∞ requerido, pero cuanto más se parezca $\mathbf{M \cdot X_1}$ a la forma descrita por el algoritmo, menos iteraciones harán falta para llegar a la convergencia. En la práctica, el número de iteraciones dependerá de las matrices de rigidez y masa, del número de valores propios que se deseen calcular y de la precisión exigida.

Existen otras maneras de generar los vectores iniciales, como el procedimiento de Lanczos. Pero no se describen en detalle dado que el procedimiento utilizado en las rutinas implementadas es el propuesto por Bathe[5].

6.3 Iteración en el subespacio

El siguiente algoritmo, denominado *Iteración en el subespacio*, encuentra una base ortogonal de vectores en E_{k+1} y calcula los vectores propios en un paso cuando E_{k+1} converge a E_∞. Este algoritmo forma el cuerpo de las rutinas del mismo nombre que se utilizarán para resolver el problema de los valores y vectores propios y que se mostrará más adelante.

Para k = 1,2,…, iterar desde E_k hasta E_{k+1}:

$$K\bar{X}_{k+1} = MX_k \qquad \text{Ec. 6.1}$$

Buscar las proyecciones de las matrices K y M en E_{k+1}:

$$K_{k+1} = \bar{X}_{k+1}^T K \bar{X}_{k+1}$$

$$M_{k+1} = \bar{X}_{k+1}^T M \bar{X}_{k+1}$$

Resolver el problema de valores y vectores propios de las matrices proyectadas:

$$K_{k+1} Q_{k+1} = M_{k+1} Q_{k+1} \Lambda_{k+1}$$

Encontrar una aproximación mejor para los valores propios:

$$X_{k+1} = \bar{X}_{k+1} Q_{k+1}$$

Entonces, si los vectores X_1 no son ortogonales a uno de los valores propios buscados, tendremos

$$\Lambda_{k+1} \to \Lambda \quad y \quad X_{k+1} \to \Phi \quad con \ k \to \infty$$

En la iteración del subespacio, está implícito que los vectores iterativos están ordenados de una manera apropiada.

6.4 Convergencia

Como cualquier método es necesario medir la convergencia del método. Para hacerlo, asumimos que en la iteración *k* los valores propios aproximados calculados son λ_i, i =1,…,p. Alcanzaremos la convergencia cuando se cumpla

$\left|\lambda_i^{(k)} - \lambda_i^{(k-1)}\right| \leq tolerancia$ para todos los valores de i=1,…,p

El valor de la tolerancia es 10^{-2s} cuando se desea que los valores propios tengan una precisión de 2s dígitos. Por ejemplo si se itera hasta que los márgenes sean menores que 10^{-5}, vemos que los valores propios tienen una precisión de al menos cinco dígitos y en los valores más pequeños normalmente se observa una precisión mayor. A pesar de que la iteración se realiza con *q*vectores, la convergencia solamente se mide para las *p* menores aproximaciones obtenidas.

El último paso en el proceso iterativo en el subespacio e igualmente importante, es verificar que los valores y vectores propios pedidos han sido calculados, ya que la ecuación 5.1.4 se satisface para cualquier par de valor y vector propio.

La iteración converge si los vectores iniciales ($\mathbf{X_1}$) no son ortogonales a ninguno de los vectores propios pedidos. El método descrito anteriormente ha demostrado ser satisfactorio en la práctica, pero no existe ninguna demostración matemática que asegure que la convergencia está garantizada. Cuando la ecuación de la tolerancia ha sido satisfecha, se puede estar seguro que los valores y vectores propios han sido calculados. Para comprobar que son los deseados y no otros, se usa la *secuencia Sturm* del polinomio característico del problema $\mathbf{K}\Phi = \lambda M\Phi$ sobre un shift μ, donde μ es justamente un valor ligeramente superior a el último valor propio (λ_p) calculado. La propiedad de la secuencia *Sturm* asegura que en la factorización Gaussiana de $\mathbf{K}-\mu\mathbf{M}$ en $\mathbf{LDL}^{\mathbf{T}}$, el número de elementos negativos de **D** es igual al número de valores propios menores que μ. Para el caso de considerado de un valor ligeramente superior al último valr propio, la matriz **D** debería contener *p* valores negativos. Se ha de tener en cuenta, que para aplicar de manera correcta la secuencia *Sturm*, se ha de utilizar un valor de μ teniendo en cuenta el hecho que los valores propios calculados no son exactos sino aproximaciones.

La robustez del método permite asegurar que el algoritmo del subespacio converge de manera efectiva en la inmensa mayoría de los casos. Sin embargo, la velocidad de convergencia no está garantizada en ningún caso, por lo que es importante poner un límite al número de iteraciones que permitimos que haga el algoritmo.

7. Programas

Para la obtención de los datos básicos para la realización de la tesina se han utilizado dos programas, el PLCD y el CFYFP, tal y como se ha comentado con anterioridad, en algunos casos se ha tenido que incluir modificaciones en las rutinas del código existente y en otros se ha tenido que crear rutinas nuevas. La situación antes de empezar era la siguiente: se tenía un programa llamado PLCD desarrollado en el Departamento de Resistencia de Materiales y Estructuras en la Universidad Politécnica de Cataluña. Este código permite trabajar con problemas lineales, no lineales, con pequeñas o grandes deformaciones y análisis térmicos y termomecánicos acoplados. Las rutinas permiten utilizar materiales que presenten fenómenos de viscoelasticidad, daño y plasticidad generalizada no asociada con endurecimiento isótropo positivo, nulo o negativo. Se trata de un código de elementos finitos desarrollado íntegramente en FORTRAN. Por otro lado, se tenían un conjunto de rutinas proporcionadas por Xavier Martínez que contenían la implementación del SIM (Subspace Iteration Method) que resuelve el problema de autovalores por el método previamente explicado. Pero existían una serie de inconvenientes que había que resolver antes de poder realizar de forma satisfactoria el cálculo de los valores y formas propias de estructuras que presentaban un comportamiento no lineal. En las secciones que siguen a continuación se muestran instrucciones y códigos en lenguaje Fortran, para las dudas que pudieran surgir sobre las instrucciones utilizadas se puede consultar cualquier manual de Fortran [18].

7.1 Situación inicial

En principio estos programas presentaban dos dificultades claras y de diferente naturaleza. La primera es que las rutinas de solución del problema de autovalores no formaban un programa por si solas y se necesitaba crear un programa para poderlas utilizar. El segundo problema es que los dos eran independientes, cosa que nos

interesaba, y no se pasaban ninguna información entre ellos, cosa que nos hacía falta para el trabajo que se pretendía desarrollar.

Originalmente el PLCD utilizaba las matrices de masa y rigidez para sus cálculos pero no las guardaba ni se las comunicaba a ningún otro programa, datos fundamentales para el problema de autovalores. Por otra parte, las rutinas SIM formaban parte de otro programa mayor y aunque si realizaba la función que nosotros queríamos, no estaba preparado para leer la información del PLCD porque cuando se creó obtenía los datos necesarios de otro programa llamado COMET. Además existía una incompatibilidad entre la manera que uno y otro almacenaban los datos, el PLCD utiliza un almacenamiento tipo *skyline* para las matrices y el SIM requiere que las matrices proporcionadas esten en formato *Sparse*. Además existía otro inconveniente: el tamaño de los datos que se tenía que extraer, por ello, a pesar que el PLCD calcula como parte de su rutina la mayoría de los datos necesarios, éstos no se extraen directamente, sino que se saca la información básica y, a partir de ella, se vuelven a calcular los datos necesarios como parte de la rutina del nuevo programa. Esto es debido a que siempre va a ser mejor y más rápido calcular de nuevo que tener que almacenar un volumen extremadamente grande de datos y luego tener que leerlos, debido tanto al tamaño de los ficheros, como al tiempo invertido en escribir y leer.

Por ello se han realizado las siguientes tareas:

- Modificación del PLCD para extraer la información indispensable para poder calcular las matrices de masa y rigidez, tanto en estado elástico como en cada paso de carga deseado.
- Modificación del PLCD para obtener el daño global en cada paso de carga deseado.
- Creación de un programa (CFYFP) que incorpore las rutinas de solución del problema de autovalores por el método de iteración en el subespacio.
- Implementación de un nuevo sistema de ensamblaje de la matriz global de rigidez, para cada estado.
- Adición al CFYFP de las rutinas encargadas del cálculo de las matrices y su acoplamiento con las nuevas tareas.
- Creación de los ficheros de salida de los valores calculados para su postproceso.

7.2 Estructura general de los programas

Con el fin de proporcionar una imagen global del proceso que se realiza para solucionar el problema, que se procederá a pormenorizar en las secciones siguientes, y situar cada paso en un marco más general dentro del proyecto, a continuación se muestra un esquema (figura 7.2.1) con unas breves explicaciones para entender el proyecto en su conjunto.

En la figura se muestra el proceso de la manera más general, desde la entrada de datos en el PLCD para la realización del análisis no lineal de la estructura hasta la creación de los ficheros finales por parte del CFYFP que nos permitirán realizar el postprocesado (en nuestro caso se utilizará el GID para visualizar muchos de los resultados obtenidos y poder extraer conclusiones) y establecer las relaciones pertinentes.

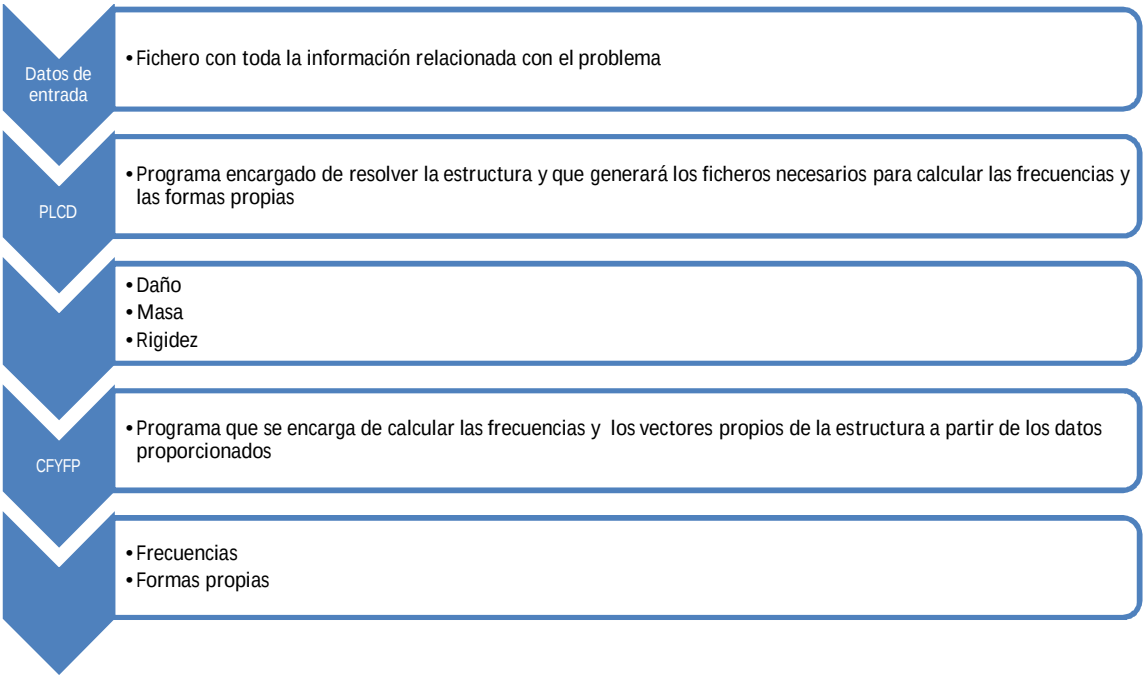


Fig. 7.2.1 Esquema general del proceso realizado

7.3 Compatibilidad entre datos

Como se ha comentado al principio de esta sección, existe un diferencia de formatos entre el almacenamiento de datos de los dos programas, esto hace que sea necesario una transformación previa para que el intercambio de información entre ellos se haga correctamente. Como se decidió calcular de nuevo las matrices de rigidez y de masa en vez de almacenarlas cuando se calculaban en el PLCD, el cambio se ha de realizar en las rutinas que se encargan de realizar el ensamblaje de la matriz global. La principal diferencia entre la nueva rutina de ensamblaje y la que utiliza el PLCD durante sus cálculos es que la antigua hace el ensamblaje y la almacena en banda en el mismo paso, en cambio la nueva hace el ensamblaje completo y posteriormente se llama a otra subrutina para que haga el guardado en formato *Sparse*. De esta forma se evita guardar la información en un formato poco útil (en banda) y tener que hacer la transformación entre formatos posteriormente, a cambio se tiene que reescribir el proceso de ensamblaje de la matriz global.

7.3.1 Almacenamiento Skyline

El programa PLCD almacena las matrices en banda por filas y para ello emplea dos vectores. El primero guarda las columnas de la parte triangular superior de la matriz comenzando cada fila desde el primer elemento no nulo hasta la diagonal. El segundo vector contiene las posiciones de los elementos diagonales dentro del primer vector. En el siguiente ejemplo para una matriz pequeña puede verse más claramente:

$$\boldsymbol{A} = \begin{pmatrix} 1 & 2 & 5 & 6 \\ 2 & 3 & 0 & 0 \\ 5 & 0 & 8 & 0 \\ 6 & 0 & 0 & 2 \end{pmatrix}$$

$$A_{Val} = (1, 2, 3, 5, 0, 8, 6, 0, 0, 2)$$

$$A_{Pos} = (1, 3, 6, 10)$$

7.3.2 Almacenamiento Sparse

El almacenamiento en coordenadas simétricas (SCS) utiliza tres vectores, en vez de dos como el caso anterior. Además existe otra diferencia respecto al almacenamiento utilizado por el PLCD, en este caso debido al formato que necesitan las subrutinas que implementan el método de iteración en el subespacio, se tiene que almacenar la matriz completa. Aunque si no tuviéramos esta limitación, dado que se trata de matrices simétricas solamente nos haría falta almacenar una parte de la matriz. En un primer vector se guardan todos los valores no nulos de la matriz. En el segundo se almacena la columna que contenga el valor no nulo, siempre siguiendo las filas. Finalmente, se archiva la posición del primer vector en la que comienza una nueva fila. Para verlo más fácilmente utilizaremos el mismo ejemplo que en el caso anterior:

$$\boldsymbol{A} = \begin{pmatrix} 1 & 2 & 5 & 6 \\ 2 & 3 & 0 & 0 \\ 5 & 0 & 8 & 0 \\ 6 & 0 & 0 & 2 \end{pmatrix}$$

$$A_{Val} = (1,2,5,6,2,3,5,8,6,2)$$

$$j = (1,2,3,4,1,2,1,3,1,4)$$

$$i = (1,5,7,9)$$

Teóricamente los vectores acabados de describir sirven para definir una matriz completa en formato Sparse, pero el programa necesita que se introduzcan unas informaciones adicionales para funcionar de forma correcta. Éstas incluyen dos datos que son fáciles de entender que sean necesarios, como son la dimensión de la matriz original (*n*), da igual el número de filas o de columnas porque será cuadrada, y el número de elementos no nulos de dicha matriz (*nnz*), necesario para conocer el tamaño de los vectores que se tendrán que utilizar para guardar la información. Toda la información presentada hasta el momento forma una sola variable que está subdividida en 5 apartados que almacenan esta información, tal y como se muestra a continuación para este ejemplo concreto:

Variable A

$$\begin{cases} A\%n = 4 \\ A\%nnz = 10 \\ A\%val = (1,2,5,6,2,3,5,8,6,2) \\ A\%j = (1,2,3,4,1,2,1,3,1,4) \\ A\%i = (1,5,7,9) \end{cases}$$

Si se lee el código que se mostrará en secciones posteriores y se compara con el ejemplo mostrado en esta sección, se puede observar que hay una cosa que no concuerda, la dimensión del vector *i*. Esto se debe a que el ejemplo de esta sección es un almacenamiento para matrices dispersas por llamarlo de una manera generalizado, y, en cambio, en el código se tienen en cuenta las operaciones que habrá que realizar con este vector y las singularidades del código implementado. Durante el transcurso de las operaciones que se realizan con las matrices, el código consulta en un mismo paso la posición *i* y la *i+1* del vector *i*. Si se realizará esta consulta con un vector como el mostrado anteriormente se produciría un error ya que la rutina intentaría acceder a una posición que no existe. Por ello en realidad la dimensión es un elemento más grande que la mostrada para este ejemplo y su valor es *nnz+1*, o lo que es lo mismo lo que tendría si hubiera una fila más en la matriz. No se producirá el problema de intentar acceder a una posición de *A%val* que no existe porque la rutina resta uno a la posición encontrada en *A%i*, para este último caso crítico se indicará el último valor de la matriz original no nulo. Para más información sobre el formato *sparse* consultar referencia [19].

7.4 Ficheros de entrada y salida utilizados

En las secciones que siguen a continuación se hablará de diferentes ficheros de entrada y de salida de datos, para tener una idea previa de cuales se han de proporcionar a los programas y cuales no han de ser proporcionados sino que son simplemente

ficheros de salida más o menos útiles, seguidamente se nombran y explican en que consiste cada uno de ellos. Un ejemplo de cada uno de los ficheros citados a continuación y que se usan en la realización de este trabajo se puede encontrar en el Anejo II: Archivos de entrada/salida. En la figura 7.4.1.1 se muestran de forma esquemática la creación y el uso de cada fichero de entrada y salida.

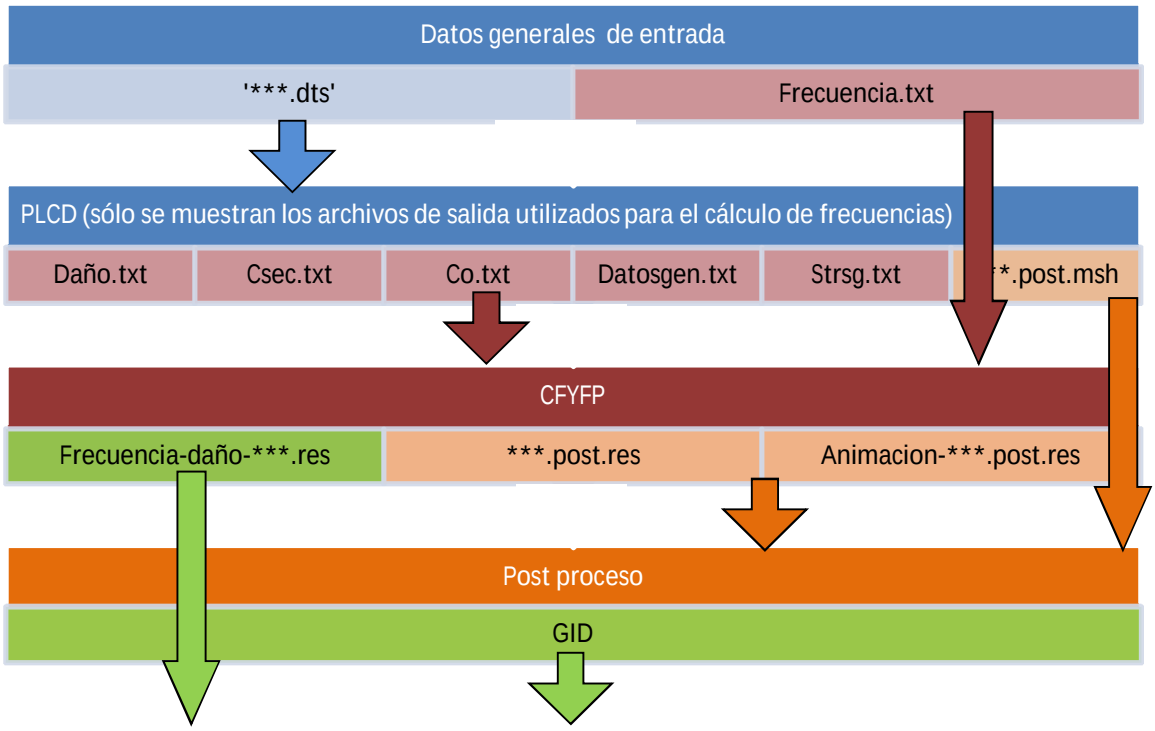


Fig. 7.4.1 Esquema de los diferentes ficheros utilizados durante el proceso

7.4.1 Ficheros de entrada y salida para el PLCD.

Los ficheros involucrados en el PLCD puede dividirse en dos grupos dependiendo de su uso: entrada y salida, o en función de su origen: ficheros existentes previamente y ficheros de nueva creación. Todos los datos de entrada necesarios para el cálculo de la estructura con un comportamiento no lineal son proporcionados en un mismo fichero:

Nombre del problema.dts Este fichero ha de estar ubicado en el directorio de trabajo del programa y su nombre nos será pedido por pantalla. En este archivo se carga la mayoría de los datos a excepción de las características de la carga dinámica, que no se utilizará en nuestro caso. Algunos de los datos que se incluyen son: tipo de problema a resolver, estrategia de solución, parámetros de resolución, número de elementos, de

nodos, de compuestos, de capas, de materiales, de fases de construcción y máximo incrementos de carga. Además del tipo de solver y el postproceso deseado, se definen los elementos (su conectividad y las coordenadas de sus nodos), los materiales compuestos (incluyendo la caracterización de los materiales simples). Finalmente se definen las restricciones de los nodos y las cargas aplicadas, asimismo se definen las fases de carga y los pasos de carga a realizar. Finalmente se definirán las curvas de salida deseadas. Una descripción completa del contenido y elaboración del fichero se puede consultar en el manual del PLCD [20].

Se ha realizado una modificación respecto al formato del fichero .dts original, se ha añadido una variable (*conf*) que servirá para indicar en que pasos concretos se quieren calcular las frecuencias y formas propias, si vale cero no se escribe información y, por tanto, no se calculará nada, si, por el contrario, vale uno, se escribirá la información. Se ha realizado está modificación para evitar tener información que no interesa y que además de ocupar espacio, necesitaría tiempo para ser escrita, leída y calculada. Por ejemplo: cuando los pasos a realizar han de ser por necesidad (lograr que converjan) muy pequeños, se realizarán muchos pasos, pero no se deseará calculas las frecuencias cada vez porque no aportaría nada sino, por ejemplo, cada 5 pasos. Esta nueva variable se puede localizar en el apartado correspondiente al grupo 18: Control Incremental de Cargas [20] (figura 7.4.1.1), concretamente entre los registros *LGRAD* (grado de libertad que se controla si *KARCL*=4 o 5) y los cuatro componentes de *IPOST3D* (sirven para indicar que post-procesos se desean). Para cualquier duda este fichero se puede consultar el manual del PLCD.

Fig. 7.4.1.1 Posición de la nueva variable *CONF* dentro del .dts, en la sección correspondiente al control incremental de cargas.

Dependiendo del .dts el número de los archivos de salida puede variar. A continuación se muestran los principales que se obtienen al resolver el tipo de problemas involucrados en este trabajo:

Nombre del problema.post.res (PLCD, original): contiene los resultados obtenidos para poderlos ver usando el software de postprocesado, en nuestro caso el GID. Los resultados incluyen los desplazamientos nodales por componentes (dx, dy, dz), las tensiones del compuesto por elementos (Sxx, Syy, Szz, Sxy, Sxz, Syz), las tensiones en las capas de materiales simples, las deformaciones de los elementos y de las capas de materiales simples, el daño local de cada punto de Gauss del compuesto y de los materiales simples. Todo ello se repite para cada uno de los pasos que se haya definido.

Nombre del problema.mec.sal: contiene un resumen de la información general del problema y de los resultados obtenidos en cada paso. Primero hay una sección en la que se muestran los datos leídos del .dts y que se utilizarán para realizar los cálculos y seguidamente para cada paso una síntesis de los resultados. Se incluye el tiempo de cálculo total y del paso, el número de incremento, la tolerancia, el número de iteraciones realizadas y el número máximo permitido, el algoritmo utilizado para obtener la solución, el residuo máximo, la potencia deformativa, la cinética, al disipada, la suma de éstas, la introducida y la energía introducida. Finalmente se muestra el daño global, medida energética y en fuerza.

Nombre del problema.post.msh: este archivo no es más que la información de la malla utilizada para la discretización del problema, incluye las coordenadas nodales, la conectividad de los elementos y el material compuesto asignado a cada uno. Este archivo nos sirve en el postproceso para dibujar la malla no deformada y poder a partir de esta dibujar los resultados obtenidos en el fichero .post.res.

fort.26 y **fort.31**: el primero contiene los residuos máximos y la convergencia de cada iteración y el resumen del tiempo de cálculo utilizado el segundo.

Nombre del problema.AA y **Nombre del problema.AB**: corresponde a la información necesaria para dibujar las curvas que se hayan pedido en el .dts, su número variará. Por ejemplo en el caso que encuentra en el Anejo III para el archivo.AA, son los datos de la fuerza y el desplazamiento para cada paso para un nodo y en una

dirección fijada. En el fichero .AB se encuentra la información para dibujar las curvas de daño y desplazamiento sufrido, contiene tres columnas con tantas filas como pasos realizados, en la primera encontramos los desplazamientos impuestos y, en las otras dos, los tipos de daño calculados por el PLCD, el medido en fuerzas y la medida energética. En caso de duda ver el manual del PLCD [20].

Los ficheros descritos hasta ahora son los que ya salían como parte de la rutina del programa antes de realizar cualquier modificación. Posteriormente y para solucionar el problema que nos ocupa, además de las nombradas hasta este punto, tenemos las siguientes salidas adicionales:

Datosgen.txt: este fichero no contiene ningún dato nuevo, respecto a los comentados anteriormente, pero tiene una razón muy sencilla para existir, facilita la lectura por parte de otras rutinas ya que los datos se encuentran sin títulos y todos seguidos sin saltos entre ellos. Como se verá más adelante este archivo es también un fichero de entrada, pero para el CFYFP. En el podemos encontrar según leemos: el nombre del problema, tipo de problema, el número de nodos, de elementos, de materiales compuestos, la conectividad nodal, las coordenadas nodales y las restricciones, entre otros datos. Para ver un ejemplo y la lista de toda la información extraída con el fichero Datosgen.txt se puede consultar el Anejo III.

Co.txt: la información que contiene este fichero de salida del PLCD y entrada del CFYFP es diferente de la descrita hasta el momento y es básica para poder calcular las frecuencias y formas propias de la estructura. Además es diferente de los anteriores en el formato en que se escribe, debido a que el tamaño del fichero puede llegar a ser bastante grande y para ahorrar espacio y ganar velocidad, se ha decidido que el fichero sea no formateado, en el anejo III se muestra el mismo archivo formateado y no formateado. El ahorro en tamaño varía, pero está entre un 40-50%. El fichero contiene tantas entradas de datos como número de materiales compuestos del problema, para cada uno de ellos tendremos: el número de material compuesto, el tensor anisótropo lineal (*Co*) y la densidad del material.

Csec.txt: al igual que el caso anterior, es un fichero de salida y entrada y está escrita sin formato para ahorrar espacio. Contiene los datos necesarios para poder crear las matrices de rigidez en los casos que la estructura ya ha sufrido daño. Para ello se

incluyen los siguiente datos: una variable auxiliar que indica el paso de actualización de la matriz de rigidez en daño, el elemento, el punto de Gauss de forma concatenada, el paso de carga, el paso de incremento para esa carga y, por último, el tensor secante del material compuesto asignado al respectivo punto de integración del elemento que ha entrado en no-linealidad.

Daño.txt: se trata, como el resto de ficheros .txt creados, de un archivo de salida de datos en el PLCD pero entrada de ellos en el CFYFP. Es un fichero formateado, con lo que puede abrirse y leerse en cualquier momento, que contiene el valor de daño global calculado con medida de fuerzas, ver ecuación 4.2.1, para aquellos pasos en los se quiere calcular los valores y formas propias de la estructura.

Strsg.txt: este fichero solamente se crea en el caso de que nos encontremos en grandes deformaciones, que no será el caso durante este estudio. No obstante se ha creado una rutina que escribe en él en caso de necesidad, ya que su información es necesaria para construir la matriz de rigidez de la estructura dañada cuando nos encontramos con el caso de tratar grandes deformaciones.

7.4.2 Ficheros de entrada y salida para el CFYFP

A diferencia de lo que pasaba en el PLCD, en este programa todos los ficheros son nuevos, ya que el programa es completamente nuevo. Los ficheros de entrada serán algunos de los de salida del PLCD: Datosgen.txt, Strsg.txt, Co.txt, Daño.txt y Csec.txt, además necesitará un nuevo .dts llamado Frecuencias. Como ficheros de salida tendremos de dos tipos: los que se escriben con un formato específico para ser usados en el GID y los que no.

Frecuencias.dts: se trata del único fichero de entrada que no es parte de la salida del PLCD y contiene los valores de los principales parámetros necesarios para el cálculo de los valores y las formas propias de la estructura. Estos son: el número de frecuencias que se quieren calcular (*NEIG* en el código del programa), la tolerancia (*TOLERSIM*), el número máximo de iteraciones permitidas (*MAXI*) y el número de formas propias de las que se quiere obtener una animación (*FANIM*).

Frecuencia-daño-Nombre del problema.res: es el fichero básico para poder observar la evolución de las frecuencias naturales de las estructuras a medida que se dañan. En él se puede observar un número de columnas igual al número de frecuencias buscadas +1, ya que en esta columna adicional se mostrará el daño global de la estructura. En la vertical veremos tantas entradas como pasos en daño se hayan realizado en el PLCD más un primer paso que corresponde al cálculo con la rigidez de la estructura "nueva" o no dañada.

Nombre del problemaF.post.msh (CFYFP): En este caso el principio del fichero es exactamente igual que el presentado anteriormente con el mismo nombre pero sin la F, pero los datos que contiene son sólo desplazamientos normalizados, que corresponden a las formas propias para cada frecuencia calculada.

Animación-Nombre del problema.post.msh: Fichero que al igual que al anterior está escrito en un formato que permita su procesamiento usando el GID. En el encontramos para la frecuencia deseada, la forma propia normalizada correspondiente escrita de tal modo que se pueda reproducir dando la sensación de movimiento. Servirá para poder ver una animación del modo de vibración de la estructura deseado.

7.5 Descripción de los cambios y nuevos desarrollos hechos en el PLCD. Programa de cálculo no lineal.

Como ya se ha comentado el PLCD es un código implícito de elementos finitos para la simulación del comportamiento de materiales compuestos desarrollado por la UPC (Universitat Politècnica de Catalunya). Se ha dicho al principio de este capítulo que la información que se deseaba extraer del programa eran las matrices de rigidez y masa, pero que debido al gran volumen de datos se había optado por extraer solamente la información mínima y recalcular las matrices cuando hicieran falta. Esta información mínima incluye los datos de los tensores del material en elasticidad y dañado y datos de carácter general del problema, como son el número de nodos y de elementos, sus coordenadas y conectividades, el material compuesto en cada caso, etc. Todo ello forma

parte de los ficheros que serán las entradas al programa de cálculo de las frecuencias naturales de la estructura, estos ficheros son: Datosgen.txt, co.txt, csec.txt y daño.txt.

Para obtener estos datos se han tenido que realizar modificaciones en la versión del PLCD original, se utilizó como base la versión actualizada de Marzo del 2011 y las modificaciones incluyen desde la creación de nuevas subrutinas para la extracción de datos hasta la modificación de las existentes para llamar a las nuevas. En los siguientes apartados se explicarán las nuevas subrutinas creadas y las modificaciones en las existentes.

7.5.1 Modificaciones en rutinas existentes

En general, la mayoría de modificaciones que se han llevado a cabo en rutinas ya existentes son consecuencia de la creación de nuevas rutinas, ya que se tienen que integrar en el funcionamiento global del programa, y no alteran las labores básicas del programa original. Por todo ello se comentarán los cambios de manera más general y superficial. Según su función se pueden distinguir dos tipos de cambios que se han realizado: simplemente llamadas a las nuevas subrutinas creadas y cambios que sirven para obtener información necesaria adicional para realizar los cálculos de las frecuencias.

Los del primer tipo han sido los siguientes:

- **Finish_V3D.f** --- Llamadas a las nuevas subrutinas calcsec_V3D.f, excsec.f y exgrandef.f están dentro de un bucle para evitar repeticiones de información innecesarias, este bucle tiene como condiciones de entrada que la respuesta haya convergido (*Base2(:,9) = 0*) y que se quieran calcular las frecuencias para ese paso (*conf = 1)*. También está dentro de este bucle la llamada para extraer los datos de las tensiones (exgrandef.f), en el caso de que el problema sea en grandes deformaciones. Las modificaciones se encuentran entre las líneas 40 y 51.
- **Calcin_V3D.f**--- Inclusión de la llamada a exco.f, inclusión de una condición para evitar repeticiones.

Otra información que hacía falta y no se ha sacado creando nuevas subrutinas es la perteneciente a las conectividades nodales de los elementos y las coordenadas nodales. De hecho, está información sí que la proporcionaba el programa pero no de una manera que nos resultara fácil luego aprovecharla. Este archivo original se sigue generando pero tiene los siguientes inconvenientes: no nos proporciona el número total de nodos, el número de elementos ni el los materiales compuestos, sin embargo este no es el mayor inconveniente sino la otra información que se guardaba. En el fichero '*Nombre del problema.post.msh*' se incluyen varios encabezados e información extra que aunque facilita su lectura para las personas, no hace sino entorpecerla en el caso de los ordenadores. Por ello se decidió crear un nuevo fichero en el que añadieran los datos de carácter general necesarios para el buen funcionamiento del nuevo programa. Como todos estos datos ya se producían, simplemente había que escribirlos en un nuevo documento llamado *Datosgen.txt*. Los cambios realizados relacionados con este fichero han sido los siguientes (se muestran en el orden se van accediendo durante el funcionamiento normal del programa):

- **Inicia.f** --- Modificaciones necesarias para la creación del nuevo fichero de salida (*Datosgen.txt*) en la línea 82, además en la línea siguiente se incluye la escritura del nombre del problema, que formará la primera línea del nuevo fichero.
- **InpuV3D.f** --- En esta rutina se escribe la mayoría de la información del fichero:
    - Línea 23: Tipo de problema.
    - Línea 24: Número total de nodos, número de elementos, número de materiales compuestos e indicador de grandes o pequeñas deformaciones.
    - Líneas 73-75: Sacar el número de nodos (*NNODE*) del tipo de elementos utilizados, como todos los elementos serán iguales, solamente se saca para el primero.
    - Líneas 78-86: Elemento, material compuesto, tipo de problema, fase, número de nodos del elemento, número de puntos de integración, cuadratura de integración a utilizar, conectividad del elemento y ángulos de Euler (necesarios cuando el sistema de coordenadas local y global no coinciden).
    - Líneas 203-206: Nodo y coordenadas nodales (i y i+1)

- Los formatos de escritura utilizados son básicamente los mismos que la rutina original pero adecuándolos en algún caso a nuestros datos de salida.

- **Inpu3_V3D.f** --- Línea 93, escribir información relativa a las restricciones de los nodos.
- **Resi2D_V3D.f** --- Se ha realizado un cambio en los parámetros que entran en la rutina, se ha añadido un valor extra (*AUX1*), debido a los cambios en la forma de calcular el daño global de la estructura. Además,
    - Línea 96: se realiza una llamada a *CALTEN_V3D* que antes no se realizaba, con el fin de calcular el vector de fuerzas elásticas.
- **Residu_V3D.f** --- Se han añadido varias salidas y entradas de datos en algunas rutinas debido al cambio en la manera de calcular el daño global:
    - Líneas 28-29: Adición de las nuevos elementos en la declaración de variables a utilizar, en este caso, para el tensor constitutivo elástico (*AUX1* y *VDMANX*).
    - Línea 127: Añadida una llamada a la base de datos 4 para obtener el tensor constitutivo en forma de vector.
    - Línea 135: Componer el tensor constitutivo a partir del vector leído en la base de datos 4.
    - Líneas 202-205: Modificación de la llamada a la rutina *Resi2D_V3D*, se añade el tensor constitutivo como valor de entrada (*AUX1*).
- **Damage_V3D.f** --- Se ha añadido una salida para los casos en que existe daño y se van a calcular las frecuencias para poder realizar una correlación. Por ello se crea un fichero nuevo (*Daño.txt*) y se escribe el daño global medido en fuerza que ha sido modificado, como se ha visto en el capítulo 4:
    - Líneas 51-53: modificación del cálculo del daño medido en fuerzas.
    - Líneas 81-85: adición de las instrucciones para generar un nuevo fichero de salida y escribir en él el valor del nuevo daño.

Además se ha modificado uno de los módulos utilizados del programa, en concreto el de *ENTEROS*, la razón de realizar este cambio es para poder definir dos variables globales nuevas, *JBL2* y *CONF*. La segunda ya ha sido explicada en el .dts, ya que era una nueva variable que servía para decidir en qué pasos de carga se quería

escribir los valores del tensor secante para calcular las frecuencias de la estructura dañada. La utilización de *JBL2* tiene relación también con el tensor secante ya que se utiliza para crear una variable auxiliar que ayude en el proceso de actualización de la matriz de rigidez, como se explicará más adelante en la siguiente sección en el apartado concerniente a la rutina *excsec.f*.

7.5.2 Nuevas rutinas creadas

Para obtener y guardar la información necesaria para poder calcular las frecuencias y las formas de vibración de las estructuras se han tenido que crear cinco nuevas rutinas: *calcsec_V3D.f, Clay-SP3Dsec.f90, exco.f, excsec.f, exgrandef*.f y *MODSEC_V3D.f*. Se puede dividir en dos grandes grupos según su función principal: las que buscan guardar la información y las que se encargan de calcular parte de esos valores.

Exco.f

Esta subrutina tiene como función extraer los datos del tensor anisótropo y la densidad de cada material compuesto, se puede consultar el código completo en el anejo número 1. La llamada a la subrutina tiene el formato habitual en lenguaje Fortran de `CALL exco(LPOP,VCMANX,DENSC)`. Los datos que necesitamos extraer se los damos como datos de entrada a la subrutina y lo único que tiene que hacer es escribir los datos en un fichero para poderse utilizar posteriormente. Esta subrutina sigue el siguiente esquema: llamado a los módulos necesarios seguido de la definición de las variables a utilizar. A continuación se crea un fichero nuevo llamado *Co.txt* para guardar los datos sin formato de escritura, que permitirá obtener un ahorro en el tamaño final de los archivos (fig. 7.5.2.1).

```
USE ENTEROS; USE MATRICES; USE PARAM3D;

IMPLICIT NONE;

REAL(KIND=8), DIMENSION(LONG2_V3D)    :: VDMANX,DENSC
INTEGER KTESC,INDMA,LONG2_VD3,MAT


  CREAMOS UN FICHERO NUEVO PARA GUARDAR LOS DATOS
  OPEN(UNIT=37,FILE = 'Co.txt',STATUS='UNKNOWN',
  POSITION='APPEND',FORM='UNFORMATTED')
```

Fig. 7.5.2.1 Uso de los módulos necesarios, definición de las variables y creación del nuevo fichero de salida

Como todos los datos necesarios ya han entrado en la rutina, ésta no tiene que realizar ninguna operación y puede escribir directamente en el fichero. Los datos siguen el siguiente orden: material, tensor anisótropo y densidad del material compuesto (fig. 7.5.2.2). Una muestra con un archivo de salida formateado se puede ver en el Anejo II: Archivos de salida.

```
WRITE(37,355)MAT,VDMANX(1:LONG2_V3D),DENSV(MATNO(1))
WRITE(37)MAT,VDMANX(1:LONG2_V3D),DENSV(MATNO(1))
```

Fig. 7.5.2.2 Escritura de los datos, la versión con formato está comentada pero se puede activar para poder visionar los datos del fichero y verificarlos.

Finalmente, lo único que nos queda es cerrar el fichero que hemos abierto y la subrutina, además definimos los formatos de escritura de datos en caso de que se quieran sacar los datos con formato.

Esta subrutina sólo ha de ser llamada una vez por cada material compuesto diferente porque representa el estado inicial en el que todavía estamos en linealidad. Existen varias posibles ubicaciones para la llamada a esta subrutina pero quizá la más lógica es al final de la subrutina *CALCIN_V3D*, que es la encargada de realizar precisamente el cálculo del tensor constitutivo elástico para el material compuesto. Para evitar que la llamada se repita más veces de las necesarias, incluimos un IF con la condición de que solamente se produzca la primera vez que pasa por esa instrucción. En la figura 7.5.2.3 se observa la subrutina *CALCIN_V3D* antes y después de la modificación realizada.

```
   DMANX=AUX1
   CVISC=AUX3

   CALL INVERT3D(MSTR1,DMANX,DMANXI,AUXL)

  DENSV(MATNO(1))=DENSc;

------------------ DESCOMPONE  TENSOR ------>  VECTOR ---------------------

   ICOM=2
   full=1  ! Complete Matrix
   CALL COMPON_V3D(VDMANX ,DMANX ,LONG2_V3D,full,ICOM)
   CALL COMPON_V3D(VDMANXI,DMANXI,LONG2_V3D,full,ICOM)
   CALL COMPON_V3D(VCVISC ,CVISC ,LONG2_V3D,full,ICOM)

-- ESCRITURA EN LA BASE DE DATOS DE LOS TENSORES CORRESPONDIENTES AL COMPUESTO --
                  ESCRITURA EN LA BASE DE DATOS - 4 -

   VB4_V3D((Ib4_4+1):(Ib4_4+LONG2_V3D))    = VDMANX(1:LONG2_V3D);
   VB4_V3D((Ib4_8+1):(Ib4_8+LONG2_V3D))    = VDMANXI(1:LONG2_V3D);
   VB4_V3D((Ib4_13+1):(Ib4_13+LONG2_V3D)) = VCVISC(1:LONG2_V3D);

-------- SE GUADA LOS TENSORES DEL COMPUESTO EN SUS COORDENADAS -------

   KTESC=1;
   INDMA=0;
   CALL BASE4_V3D(LPROP,INDMA,KTESC);

ENDDO

RETURN

END
```

```
   DMANX=AUX1
   CVISC=AUX3

   CALL INVERT3D(MSTR1,DMANX,DMANXI,AUXL)

  DENSV(MATNO(1))=DENSc;

------------------ DESCOMPONE  TENSOR ------>  VECTOR ---------------------

   ICOM=2
   full=1  ! Complete Matrix
   CALL COMPON_V3D(VDMANX ,DMANX ,LONG2_V3D,full,ICOM)
   CALL COMPON_V3D(VDMANXI,DMANXI,LONG2_V3D,full,ICOM)
   CALL COMPON_V3D(VCVISC ,CVISC ,LONG2_V3D,full,ICOM)

-- ESCRITURA EN LA BASE DE DATOS DE LOS TENSORES CORRESPONDIENTES AL COMPUESTO --
                  ESCRITURA EN LA BASE DE DATOS - 4 -

   VB4_V3D((Ib4_4+1):(Ib4_4+LONG2_V3D))    = VDMANX(1:LONG2_V3D);
   VB4_V3D((Ib4_8+1):(Ib4_8+LONG2_V3D))    = VDMANXI(1:LONG2_V3D);
   VB4_V3D((Ib4_13+1):(Ib4_13+LONG2_V3D)) = VCVISC(1:LONG2_V3D);

-------- SE GUADA LOS TENSORES DEL COMPUESTO EN SUS COORDENADAS -------

   KTESC=1;
   INDMA=0;
   CALL BASE4_V3D(LPROP,INDMA,KTESC);

   IF(ICARG.EQ.1.0) THEN
     CALL EXCO(LPROP,VDMANX,DENSC)
   ENDIF

ENDDO

RETURN

END
```

Fig. 7.5.2.3 Comparativa del código *CALCIN_V3D* antes (izqda.) y después (dcha.) de la modificación. Se muestra el código entre la línea 86 y el final en ambos casos.

Excsec.f

Al igual que la anterior subrutina ésta también tiene como finalidad extraer información del PLCD para utilizarla posteriormente al calcular los valores y vectores propios, por tanto la estructura del código y los comandos utilizados serán muy similares a los de la subrutina *exco*. El código completo se puede consultar en el anejo número 1. El llamado a esta subrutina se hará de la siguiente manera: `CALL exctan(i,j,k,l,DB2aus)`. Los datos que guardaremos en un nuevo fichero (*Csec.txt*) serán, en este caso, los pertenecientes al tensor secante (*CSEC*), además incluiremos el número de elemento examinado, la carga, el paso en el momento de tomar los datos, el punto de Gauss en el que se calcula y una variable auxiliar que nos servirá más adelante en la actualización de la matriz de rigidez dañada. El PLCD no calcula el tensor secante, porque en la solución de los problemas no lineales utiliza el tensor tangente. La estrategia empleada para obtener el tensor secante fue calcularlo una vez se obtenga la convergencia en el respectivo paso de carga. Para el cálculo se usa el valor de la variable de daño para calcular el *Csec=(1-d)Co* en el material simple y recomponer el *Csec* del compuesto. Para esto se tuvieron que crear las rutinas *CALSEC_V3D, CLAY-SP3D* y *MODCSEC_V3D*, por tanto su llamada tendrá que ser posterior al uso de estas tres nuevas rutinas.

En esta rutina, primero se llaman a los módulos necesarios, se inicializan las variables necesarias para este programa y se crea el fichero para escribir los datos. A continuación y para ahorrar espacio en el guardado se concatena la información de los tres puntos de Gauss en una sola variable (Fig. 7.5.2.4). Posteriormente, se verá que para usar la información de los puntos de Gauss se tiene que desconcatenar, pero no importa ya que el proceso es rápido y sencillo y para problemas grandes importa más ahorra memoria.

```
CONCATENAMOS EL PUNTO DE GAUSS PARA AHORRAR MEMORIA
PGAUS=0
PGAUS=(IGAUS*100+JGAUS*10+LGAUS)
```

Fig. 7.5.2.4 Concatenado de los puntos de Gauss para obtener un ahorro de espacio en el fichero

Seguidamente se crea un contador auxiliar (*AUX*) que tiene como función agrupar con un mismo dígito todos los puntos de Gauss que se encuentre en un mismo paso y carga, en este punto no tiene importancia, pero cuando se tenga que actualizar la matriz de rigidez elástica incorporando los valores dañados, servirá para tener

agrupados todos los valores que se tienen que cambiar juntos para obtener la matriz de rigidez modificada. Para ello usamos las instrucciones de la figura 7.5.2.5. Será en este punto cuando se usará la nueva variable creada en el módulo *ENTEROS*, *JBL2*, que se usará para que la primera vez que se usa la rutina tomen valor dos variables, que contienen el elemento (*INDXI*) y el punto de Gauss (*PGAUSI*), que sirven para comparar con el resto y decidir si se sigue en el mismo paso de actualización u otro. Una vez se llega un registro que requiere un nuevo paso de actualización, ya sea porque estamos en otro paso de carga o en otro incremento, se actualizarán todas las variables. Después de este pequeño cálculo para decidir el valor de *AUX*, la subrutina ya tiene toda la información que tiene que guardar y puede proceder a hacerlo (figura 7.5.2.6). Un ejemplo de la forma típica del archivo de salida puede consultarse en el Anejo II: Archivos de Entrada y Salida.

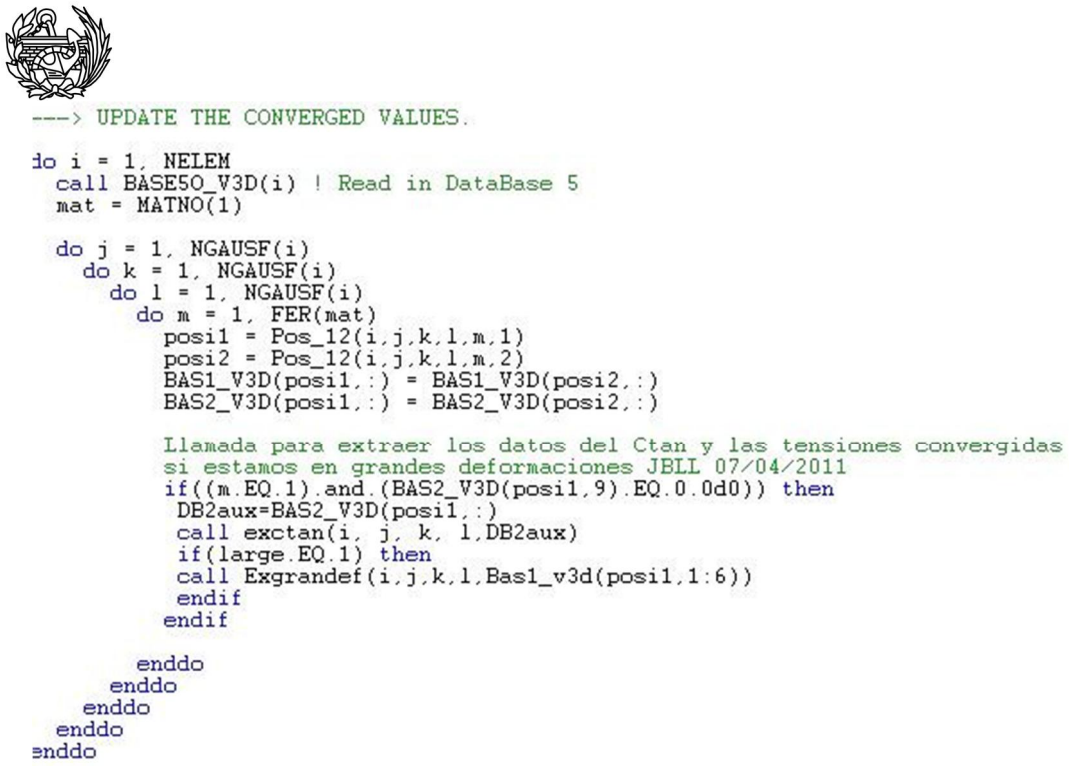
```
IF((ICARG.NE.ICARGV).OR.(IINCS.NE.IINCSV))THEN

      AUX=AUX+1
      IINCSV=IINCS
      ICARGV=ICARG

ENDIF
```

Fig. 7.5.2.5 Instrucciones para crear un contador que agrupe a los puntos de Gauss que tengan que actualizarse juntos

```
ESCRIBIMOS LA INFORMACIÓN NECESARIA EN EL NUEVO FICHERO CREADO
WRITE(36)AUX,INDX,PGAUS,ICARG,IINCS,DB2aux(17:Ib2_M)
```

Fig. 7.5.2.6 Escritura de la información en el fichero de salida Csec.txt

Como ya se ha dicho anteriormente, solamente nos interesará esta información cuando haya convergido, por tanto su ubicación tiene que estar fuera del bucle que busca la convergencia de la solución para evitar ser llamado inútilmente, se ha decidido hacer el llamado dentro de la subrutina *FINISH_V3D* que se encarga de actualizar los valores convergidos entre bases de datos. Dentro de esta subrutina existen una serie de bucles que de no incluir nosotros una nueva condición (Fig. 7.5.2.7), causaría que la información se repitiera en el archivo de salida. En la siguiente figura (Fig. 7.5.2.7) puede verse también la posición la nueva subrutina dentro de la rutina original *FINISH_V3D*, en ella se puede los diferentes cambios comentados y que se comentarán y que afectan a dicha rutina.



```
---> UPDATE THE CONVERGED VALUES.

do i = 1, NELEM
  call BASE5O_V3D(i) ! Read in DataBase 5
  mat = MATNO(1)

  do j = 1, NGAUSF(i)
    do k = 1, NGAUSF(i)
      do l = 1, NGAUSF(i)
        do m = 1, FER(mat)
          posi1 = Pos_12(i,j,k,l,m,1)
          posi2 = Pos_12(i,j,k,l,m,2)
          BAS1_V3D(posi1,:) = BAS1_V3D(posi2,:)
          BAS2_V3D(posi1,:) = BAS2_V3D(posi2,:)

          Llamada para extraer los datos del Ctan y las tensiones convergidas
          si estamos en grandes deformaciones JBLL 07/04/2011
          if((m.EQ.1).and.(BAS2_V3D(posi1,9).EQ.0.0d0)) then
           DB2aux=BAS2_V3D(posi1,:)
           call exctan(i, j, k, l,DB2aux)
           if(large.EQ.1) then
           call Exgrandef(i,j,k,l,Bas1_v3d(posi1,1:6))
           endif
          endif

        enddo
      enddo
    enddo
  enddo
enddo
```

Fig. 7.5.2.7 Condición para el llamado de la subrutina exsec y situación dentro del código de *FINISH_V3D.*Además se puede observar la llamada a la función Exgrandef, para el caso de tratar un problema con grandes deformaciones. Se muestra el código de la línea 25 a la 54

Exgrandef.f

Como se puede observar en la figura 7.5.2.7 dentro de la rutina *FINISH_V3D* existe una condición que en caso de tratar un problema de grandes deformaciones, se hace una llamada a una subrutina llamada *Exgrandef.f*. Su función es proporcionar la información del tensor de tensiones necesario para calcular la matriz de rigidez en este caso particular, el resto de las funciones suceden de forma normal y no se ven alteradas.

El código, que se puede consultar en el Anejo I, es muy similar al descrito en el caso de *excsec.f*, el único cambio es que en vez de escribir la información del *CSEC*, escribimos la de *STRSG* (el tensor de tensiones) y guardamos la información en un fichero llamado *Strsg.txt*. Todo lo demás es igual. Pero para el tipo de problemas descritos en este trabajo no es necesario.

Calcsec_V3D.f

Como se ha comentado anteriormente parte de la información necesaria que tenía que extraerse del PLCD era el tensor secante y éste no era calculado como parte de la rutina original. Por ello se han tenido que crear dos nuevas rutinas además de esta, como se verá más adelante. De todas maneras está rutina no es totalmente nueva, sino que se trata de una versión modificada del *CALCIN_V3D* que lo que hace es calcular el

tensor constitutivo secante del material compuesto para cada punto de integración que haya entrado en no linealidad.

A diferencia del original, a esta rutina tenemos que darle varios parámetros de entrada (*INDXB5, IGUAS, JGAUS, LGAUS*) y nos proporcionará uno de salida (*VDMANX*). Esto se hace porque su llamada se produce cada vez que un punto de Gauss ha convergido y ha sufrido daño, originalmente se hacía un bucle para cada material compuesto, pero ahora sólo para el punto concreto de la llamada. Lo que se hace en la rutina es lo siguiente: se ponen a cero las variables que se usarán, se hace un bucle sobre cada lámina del compuesto, dependiendo de la teoría de mezclas que se esté usando (paralelo, serie paralelo o nano tubos) se seleccionará una rutina según la teoría de mezclas usada para calcular el tensor. Una vez calculado se rotará el tensor desde las coordenadas locales a las globales usando los ángulos de Euler proporcionados en el .dts y se irán sumando las contribuciones de cada capa del compuesto teniendo en cuenta su participación volumétrica (figura 7.5.2.8). Una vez finalizado el bucle sobre las capas se transformará el tensor en vector para que sea más sencillo su almacenamiento, éste será devuelto a la rutina "madre" y escrito en el fichero *Csec.txt* usando la rutina *excsec*, explicada anteriormente.

```
      DO KLAY =1, NULAY(LPROP)        ! BUCLE SOBRE CADA LAMINA DEL COMPUESTO
          pvol = PVLAY(LPROP,KLAY)
          SELECT CASE (TYLAY(LPROP,KLAY))
          CASE ('P')
          CALL MODCsec_V3D(DMANX,CVISC,DENSL,INDXB5,IGAUS,JGAUS,LGAUS)
          CASE ('SP')
          CALL CLAY_SP3Dsec(DMANX,CVISC,DENSL,INDXB5,IGAUS,JGAUS,LGAUS)
          CASE ('NT')
          CALL CLAY_NT3D(DMANX,CVISC,DENSL)
          END SELECT
   -- ROTACIÓN DEL TENSOR CONSTITUTIVO LAYER A LAS COORDENADAS DEL COMPUESTO --------
          A1 = LAEULER(LPROP,KLAY,1)
          A2 = LAEULER(LPROP,KLAY,2)
          A3 = LAEULER(LPROP,KLAY,3)
          IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
          dir = 2  ! local LAMINA to global MATERIAL COMPUESTO
          call ROTM(DMANX,A1,A2,A3, mstr1, dir)
          call ROTM(CVISC,A1,A2,A3, mstr1, dir)
          END IF
   -- TENSOR CONSTITUTIVO DEL COMPUESTO EN SU COORDENADAS --------
          AUX1(:,:) = AUX1(:,:) + pvol*DMANX(:,:)
          AUX3(:,:) = AUX3(:,:) + pvol*CVISC(:,:)
          DENSc = DENSc + pvol*DENSL
      END DO
```

Fig. 7.5.2.8 Bucle de C*alsec_V3D* sobre las láminas del compuesto, cálculo según la teoría de mezclas apropiada, rotación usando los ángulos de Euler y cálculo del tensor teniendo en cuenta la participación volumétrica de cada capa.

Como se ha comentado, el cálculo del tensor secante se hace según la teoría de mezclas adecuada a la capa que nos ocupa, aunque se ha dicho que puede ser paralelo, serie paralelo o nano tubos, realmente solamente pueden utilizarse las dos primeras teorías ya que la tercera no ha sido adecuada para calcular el tensor secante, la teoría de nanotubos ha sido implementada por primera vez en el PLCD en su última versión y en el caso de los ejemplos utilizados en esta tesina no se ha utilizado en ningún caso. A continuación se explican las rutinas que calculan el tensor siguiendo teoría pertinente.

Modcsec_V3D.f

Esta rutina se encarga de calcular el tensor constitutivo secante para una capa con mezcla paralela y es una versión modificada de la rutina original del PLCD *MODCO_V3D*. De la misma manera que pasaba en la versión modificada de *CALCIN_V3D* a la rutina se le proporcionarán como datos de entrada el elemento y el punto de Gauss que ha entrado en daño y se hará un bucle sobre todas las capas que estén afectadas. Para calcular el tensor secante, se hace una llamada a la base de datos 4 para obtener los datos del tensor lineal del material simple que constituye la capa y una llamada a la base de datos 2 para obtener el daño de ese punto de Gauss de ese elemento. Con estos dos valores se calcula el tensor secante de la siguiente manera:

$$C_{aux} = (1 - d) \cdot C_o$$

Pero esta información si se miran las instrucciones no está en forma de tensor sino de vector, por ello se transforma en tensor usando la función *COMPON_V3D* y se continua rotándolo a las coordenadas de las capas y teniendo en cuenta su participación se irán añadiendo los efectos de las distintas capas. Una vez se finalice el bucle sobre todas, se devolverá el valor del tensor secante calculado a la rutina *CALCSEC_V3D* para que continúe su curso.

Clay-sp3dsec.f90

Esta rutina realiza la misma función que *MODCSEC_V3D* pero para los casos que la teoría de mezclas utilizada es serie-paralelo, al igual que en el caso de paralelo se le ha de suministrar como datos de entrada el elemento y el punto de Gauss. Es una versión modificada de *CLAY-SP3D*. El funcionamiento de la rutina es exactamente el mismo que la última pero en lugar de realizar n-veces las operaciones (llamada a las

bases de datos 4 y 2, multiplicación por el daño, composición del tensor, rotación y adición de las distintas contribuciones) se tiene que hacer una vez para la matriz y otra para la fibra. Esta pequeña diferencia hace que se utilicen algunas rutinas adicionales pero el proceso es el mismo.

7.6 Descripción de los cambios y aportes hechos la programa SIM.

Se trata de las rutinas principales, proporcionadas por Xavier Martínez, encargadas de resolver el problema de autovalores de la estructura, por tanto su correcto funcionamiento es una parte esencial del proceso de cara a obtener resultados correctos. Son una serie de subrutinas en código FORTRAN creadas alrededor de una principal llamada *Main-Sim* que va haciendo el llamado a las restantes y que mediante el método de iteración en el subespacio se encarga de calcular los valores y vectores propios. A continuación, en la siguiente página, se muestra un esquema con las principales acciones que realiza (Figura 7.6.1).

Tal y como se muestra en la figura 7.6.1 la primera tarea que se realiza es la factorización de Cholesky de la matriz de rigidez, el programa permite dos posibilidades aunque en nuestro caso siempre se hará lo mismo. Una de las posibilidades es dar el tamaño de la matriz descompuesta a la rutina, en cuyo caso se pasa directamente a realizar la factorización numérica de Cholesky. Si, por el contrario, no se le proporciona tamaño alguno, tal y como haremos nosotros siempre de manera automática, primero se realiza la llamada a la rutina encargada de calcularlo ese tamaño (*SIM_Symbol*) y, una vez conozca el tamaño adecuado para alocar las variables necesarias, proseguirá realizando la factorización numérica de Cholesky.

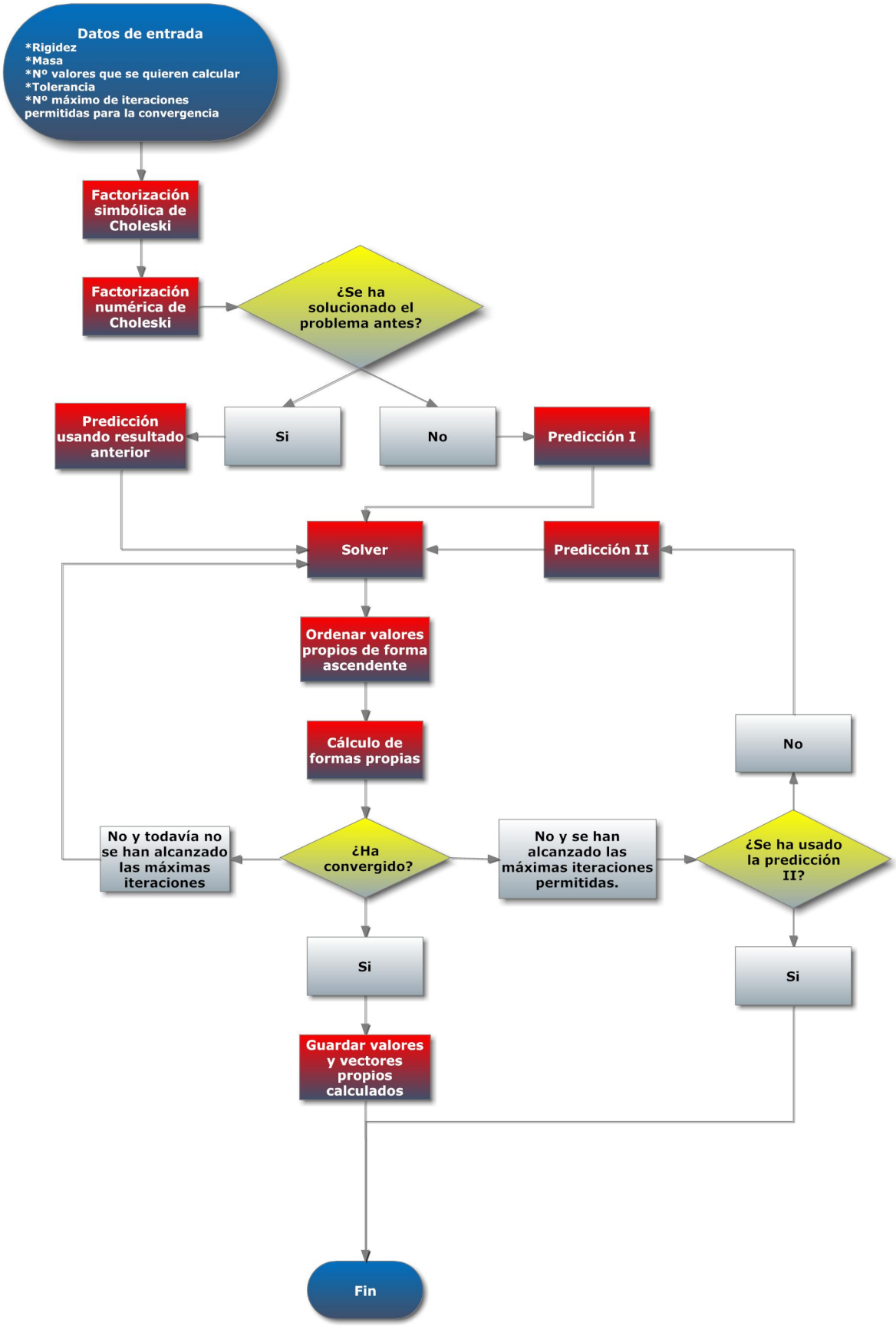


Fig. 7.6.1 Esquema de la rutina encarga del cálculo de los valores y vectores propios

7.6.1 Factorización de Cholesky

André-Louis Cholesky encontró que una matriz simétrica definida positiva se puede descomponer como el producto de una matriz triangular inferior y la traspuesta de ésta. En esto consiste la factorización que recibe su nombre, en encontrar esta matriz. Sin entrar en mucho detalle sobre la manera como se calculan los elementos, simplemente mostrar que la factorización, en caso de realizarla de la siguiente manera $\mathbf{A=U^T * U}$, sería igual:

$u_{ii}^2 = a_{ii} - \sum_{k=1}^{i-1} u_{ik}^2$ para los elementos de la diagonal principal

$u_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} u_{ik} u_{jk}}{u_{jj}}$ para el resto de elementos

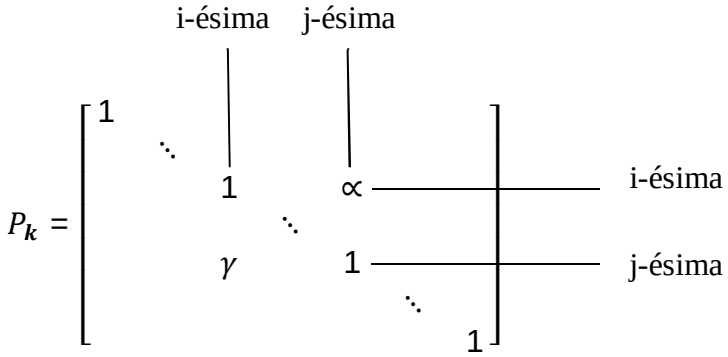
Una vez tenemos la matriz de rigidez descompuesta, se necesitan unos vectores iniciales para comenzar a iterar. En este caso, la rutina cuenta con dos formas de realizar estas predicciones la primera vez, la segundan solamente se utiliza en el caso de que no se alcance la convergencia en el número máximo de iteraciones permitidas. La primera rutina escoge los vectores de la manera explicada en la sección 3.2, la importancia de este paso radica en la disminución de las iteraciones necesarias para lograr la convergencia y, por tanto, del tiempo de cálculo. Con estos dos pasos previos ya se puede empezar el bucle principal del programa que tiene como condición de salida la convergencia, sólo se sale del bucle con una respuesta que haya convergido o sin respuesta, en caso de que se superen el número máximo de iteraciones permitidas. Para calcular la solución del problema de valores y vectores propios se utiliza la rutina llamada *SIM-Jacobi.f*, que se explica en el siguiente apartado.

7.6.2 Solver

En el esquema de la rutina utilizada (figura 7.6.1) para calcular los valores y vectores propios se puede observar que el paso fundamental es solucionar el problema de valores propios, para ello, en este caso concreto, se utiliza el método de Jacobi. Este método debe su nombre a su creador, Carl Gustav Jakob Jacobi, y fue desarrollado hace más de un siglo. En principio solamente servía para solucionar problema en los que la matriz de masa fuera la identidad (**M** = **I**) pero existe una solución generalizada para los problemas de valores propios $\mathbf{K}\Phi=\lambda\mathbf{M}\Phi$,$\mathbf{M}\neq\mathbf{I}$ (implementada en FORTRAN por

Bathe y Wilson en 1976). Este método soluciona a la vez todo el problema, cosa que hace que resulte poco eficiente cuando se está tratando con matrices de masa y rigidez de grandes dimensiones, sin embargo si resulta efectivo debido a su simplicidad para solucionar pequeños sistemas, como en el caso del método de los subespacios.

El método generalizado pretende reducir a cero de manera simultánea los elementos (i,j) en la matrices de rigidez (**K**) y masa (**M**) realizando las siguientes multiplicaciones: $\boldsymbol{P}_{\mathrm{k}}^{T}\boldsymbol{K}_{k}\boldsymbol{P}_{k}$ y$\boldsymbol{P}_{\mathrm{k}}^{T}\boldsymbol{M}_{k}\boldsymbol{P}_{k}$, siendo $\mathbf{P}_{\mathrm{k}}$ la siguiente matriz:



dónde el subíndice k hace referencia la iteración (k-ésima). En general, los dos parámetros (α y γ) utilizados se definen de la siguiente manera:

$$\gamma = -\frac{\bar{k}_{ii}^{(k)}}{x} \propto = \frac{\bar{k}_{jj}^{(k)}}{x} \qquad \text{(Ec. 7.6.1.1)}$$

siendo x el valor determinado usando:

$$x = \frac{\bar{k}^{(k)}}{2} + sign(\bar{k}^{(k)})\sqrt{\left(\frac{\bar{k}^{(k)}}{2}\right)^{2} + \bar{k}_{ii}^{(k)}\bar{k}_{jj}^{(k)}} \qquad \text{(Ec. 7.6.1.2)}$$

Estas relaciones mostradas en las ecuaciones 7.6.1.1 y 7.6.1.2 fueron desarrolladas para matrices **M** llenas o bandadas, existen otras que se ajustan de mejor manera a otros casos.

El proceso de reducción de los elementos no diagonales se va repitiendo para cada elemento de manera que al acabar de recorrer la matriz de dimensión *n* tendremos: $\boldsymbol{P}_1^T\boldsymbol{P}_2^T\boldsymbol{P}_3^T\boldsymbol{P}_4^T \dots \boldsymbol{P}_n^T\boldsymbol{K}\boldsymbol{P}_1\boldsymbol{P}_2\boldsymbol{P}_3\boldsymbol{P}_4 \dots \boldsymbol{P}_n$ y $\boldsymbol{P}_1^T\boldsymbol{P}_2^T\boldsymbol{P}_3^T\boldsymbol{P}_4^T \dots \boldsymbol{P}_n^T\boldsymbol{M}\boldsymbol{P}_1\boldsymbol{P}_2\boldsymbol{P}_3\boldsymbol{P}_4 \dots \boldsymbol{P}_n$. Si la primera no es una matriz diagonal deberemos seguir iterando hasta que lo sea, cada bucle sobre la matriz es una iteración, si lo es ya tendremos, respectivamente, los valores y formas propias en cada una de las matrices obtenidas.

Por último, se ha de comentar que normalmente para evitar tener que transformar en cero valores que ya sean muy cercanos, se establece un umbral que fija a partir de qué valor se hace el cálculo. Esto se debe a que si el número ya es originalmente muy cercano a cero, el resultado final no variará significativamente y se puede ahorrar el cálculo. Además como normalmente se trata con matrices simétricas, solamente hay que recorrer la matriz triangular inferior o superior.

Una vez finaliza la rutina se obtienen unas soluciones al problema, primero se ordenan en orden ascendente para facilitar su visualización, seguidamente se ha de comprobar si la respuesta obtenida es buena o no, para ello se usa el parámetro de Rayleigh, además como se ha visto se ha modificado la forma de calcular la convergencia, por tanto si ha convergido usando Rayleigh se pasará a la nueva comprobación implementada y ya explicada en la sección de las modificaciones.

7.6.3 Parámetro de Rayleigh

El cociente de Rayleigh se calcula de la siguiente manera:

$$R = \frac{V^T \cdot \boldsymbol{K} \cdot V}{V^T \cdot \boldsymbol{M} \cdot V}$$

Siendo **K** y **M** las matrices de rigidez y masa respectivamente y V los vectores propios obtenidos al solucionar el problema.

Con este valor obtenido (R) se calcula un error comparándolo con el valor propio encontrado, que si supera la tolerancia nos envía otra vez a realizar otra iteración para encontrar una respuesta más precisa. Si se ha alcanzado la convergencia se devolverán los valores a la rutina principal. El problema que presenta está forma de comprobar la convergencia sin ninguna instrucción adicional es que el programa calcula los valores propios y, en base a ellos, los vectores asociados. Al utilizar estos vectores para calcular los valores propios se está haciendo el mismo paso pero al revés, de tal forma que el programa logra siempre la convergencia en la primera iteración.

7.6.4 Rutinas modificadas del SIM

Respecto a las rutinas originalmente proporcionada por Xavier Martínez se han realizado unas pequeñas modificaciones para lograr una compilación de las rutinas de forma correcta, seguramente la mayoría de los problemas encontrados se deban a la utilización de distintas versiones del compilador, ya que este problema se observó en otros casos durante la realización de esta tesina, o también puede ser porque las matrices que se usan como datos tienen propiedades distintas a las que eran utilizadas en su día por Xavier Martínez. Los cambios realizados han sido:

- Se ha cambiado la llamada a la rutina principal añadiendo nuevas variables, en consecuencia, en *Main-SIM.f90* también se ha tenido que realizar estos cambios, pasando a ser el encabezamiento de la subrutina el siguiente: `Eigen_SIM (A, B, new_size, sval, svec, neig, toler, maxi, iout, iiter, StVer, sturm1, sturm2, pred, spred, R)`. Las nuevas entradas añadidas han sido: *pred*, *spred* y *R*. La primera sirve para indicar si se ha solucionado antes el problema de autovalores de la estructura, en tal caso se utilizarán los valores obtenidos la última vez como predicción para empezar a iterar. *Spred* contiene esos valores que han formado la solución en la anterior resolución del problema. Finalmente, *R* sirve para dar dimensión a la matriz *Spred.*
- En la rutina *Main-SIM.f90* se ha creado la variable *new_n* que toma el valor de *new_size* debido esta versión del programa no permitía el cambio de un valor de una variable que era la encargada de controlar el bucle (*new_size*). Por ello la nueva variable toma el valor de la original y se utiliza en lugar de ésta.
- En la misma rutina que el anterior se ha desactivado la parte del código encargada de realizar el Sturm check. La razón ha sido que con las matrices que tenemos que calcular se produce un error ya que en formato Sparse el número de valores no nulos de las matrices de masa y rigidez no tiene que coincidir, es más en general no coincidirán y el que lo hagan es una probabilidad muy pequeña.

- Todavía en *Main-SIM.f90*, entre las líneas 77 y 80 se ha añadido un *if*, la razón es determinar si ya se han calculado antes las frecuencias y formas propias, en tal caso se utilizará esta respuesta como predicción inicial y nos saltaremos el cálculo de los vectores predictores (`go to 50`). En la línea 90 se ha tenido que añadir la bandera de salto anterior (50).
- Entre las líneas 149 y 155 de la rutina *Main-SIM.f90* se ha añadido una nueva condición de convergencia, ya que la anterior por si sola no la garantizaba. La nueva convergencia se realiza solamente en el valor propio más alto calculado, ya que es el que más problemas presenta normalmente. Se supone que si se cumple para el valor más alto los demás también lo harán. Para ello se compara el valor de la iteración anterior con el actual de la siguiente manera: `abs(svalv(neig)-Vk(neig))/Vk(neig)).lt.toler`, donde *svalv* contiene el valor calculado en la iteración anterior. Si se cumple la condición se abandona el bucle de cálculo, en caso contrario se vuelve otra vez al bucle y se cambia el indicador de convergencia a no convergido.
- El último cambio de esta rutina principal se ha realizado en la línea 175, se ha añadido una instrucción para almacenar la respuesta convergida y utilizarla como predicción inicial la siguiente vez que se calculen los resultados.
- En *SIM-Predict2.f* en las líneas 38 y 39 se ha añadido la siguiente instrucción `abs`, se ha realizado este cambio porque con el tipo de problemas que se van a resolver pueden surgir problemas si no se utiliza el valor absoluto. Antes siempre se trataba de un valor positivo y por eso se miraba si era menor o que el *rl*, pero ahora se ha de tener en cuenta que podría ser negativo en algún caso. Este cambio, al igual que el de la rutina siguiente fueron propuesto por Xavier Martínez.
- En *SIM-Predict.f90* entre las líneas 39-40 y 44-45 se ha realizado el mismo cambio que en el caso anterior, se ha añadido la instrucción `abs` para que use el valor absoluto de `w(i)` y evitar posibles problemas que pudieran surgir durante la ejecución y que se han detecta al realizar algún ejemplo.

7.7 Descripción de los desarrollos hechos en el CFYFP.

A diferencia de las subrutinas comentadas hasta este momento que formaban parte del PLCD, el CPYFP es un programa totalmente independiente creado a partir de cero y que nos permite calcular las frecuencias y las formas propias de las estructuras al evolucionar. En este caso se está utilizando para resolver estructuras que se van dañando pero no tiene esa limitación. Al programa no le importa qué tipo de problema estamos resolviendo, en este caso es daño pero puede resolver problemas de viscoplasticidad o de plasticidad, que resuelva este tipo de problemas depende de las salidas que proporcione el PLCD. El CFYFP no le importa la manera en que se han tenido que obtener los datos, mientras los tenga, calculará las frecuencias. De manera muy esquemática y general se puede decir que el programa tiene principalmente las siguientes funciones: transformar la información obtenida del PLCD en matrices de rigidez y masa, almacenarlas de manera que éstas puedan ser procesadas por la rutina encargada del cálculo de los valores y vectores propios, resolver el problema de autovalores y escribir los resultados. Este proceso se realizará para el caso base no dañado y para todos los pasos dañados que se hayan obtenido en el PLCD. El código completo puede verse en el Anejo I.

Las rutinas utilizadas en este programa pueden dividirse en tres según su origen: rutinas ya existentes en el PLCD y que se reutilizarán, las suministradas por Xavier Martínez (encargadas de realizar el cálculo de los valores y vectores propios) y, finalmente, las creadas a partir de cero para complementar a los dos primeros grupos y darle consistencia a todo el programa. A continuación se muestra una lista de todas las rutinas utilizadas clasificadas según su origen:

Subrutinas utilizadas por el PLCD y que se reutilizan:

ALOKA_V3D.F	BMAT3D.F	BMATRIS.F
BASE6I_V3D.F	BMATRIL.F	BMATSOL_V3D.F
BASE6O_V3D.F	BMATRIL3D.F	CALRME.FOR

CARTEL.F	JACOB3D.FOR	PROMA1.F
COMPON_V3D.F	KMATRI_V3D.F	PROMAT.F
DETERM3D.F	MAKINA.F	ROT-SP.F90
ENTEROS.F	MASMATV3D.F	ROT.F
FUNCIONES.F	MATRICES.F	RUNEND_V3D.F
GRADEF_V3D.F	PARAM.F	SFR3D.F
INGAUS_V3D.F	PARAM3D.F	SUMMAR.F
INICIAL.F	PRESENT.F	TIMUSE.F
INVERT3D.F	PROCOM_V.F	TYPEV3D.F

El conjunto de subrutinas suministradas por Xavier Martínez y que tienen la tarea de calcular los valores y vectores propios utilizando el método de iteración en el subespacio:

ALLOCATE_MATRIX.F90	SIM_JACOBI.F90	SIM_RAYLEIGH.F90
DEALLOCATE_MATRIX(LA).F90	SIM_MATMAT.F90	SIM_SOLVER.F90
MAIN_SIM.F90	SIM_MATVEC.F90	SIM_STURM.F90
SIM_CHOL.F90	SIM_ORDER.F90	SIM_STURM_LDL.F90
SIM_SYMBOL.F90	SIM_PREDICTION.F90	
SIM_ERROR.F90	SIM_PREDICTION2.F90	

Por último, las subrutinas nuevas que no existían y que han tenido que crearse, para algunas de ellas se han utilizado de base otras ya existentes y en el resto son completamente nuevas:

ALMACEN.F CFYFP.F VECTPROP.F

ANIMACION.F ESCRIBIR.F

7.7.1 Estructura del programa CFYFP

Desde el punto de vista de su funcionamiento, este programa se puede dividir en varios bloques principales que básicamente se repiten salvo pequeñas diferencias para el caso no dañado y para los dañados. En la figura 7.6.1.1, que se encuentra en la siguiente página, se puede ver de manera gráfica el funcionamiento esquemático del programa completo y sus pasos más importantes.

En la figura 7.6.1.1 se ha descrito el siguiente proceso:

- Cálculo de las matrices elementales de rigidez, pasando por todos sus pasos intemerdios.
- Ensamblaje de la matriz de rigidez global.
- Reducción de la matriz en base a las restricciones nodales.
- Almacenamiento de esa matriz en formato SPARSE
- Llamada a la rutina encarga de calcular las frecuencias y los vectores propios.
- Escritura de los resultados obtenidos en los ficheros de salida que correspondan.

En el caso del cálculo cuando la estructura está dañada existe alguna pequeña diferencia que hace el proceso un poco más complejo debido a que se tiene que realizar un proceso adicional: para aquellos puntos de Gauss que hayan entrado en daño en un determinado paso se procederá a descontar la contribución que tenía ese punto en la matriz global **K** original y se le adicionará la nueva contribución “dañada”. Para ello, se ha de mirar que puntos de Gauss entraron en plasticidad y usar el *Csec* en esos casos, en los otros seguimos con el *Co*. De esta manera se obtendrán tantas matrices como pasos con variable *CONF* igual a uno se hayan demandado en el .dts del PLCD.

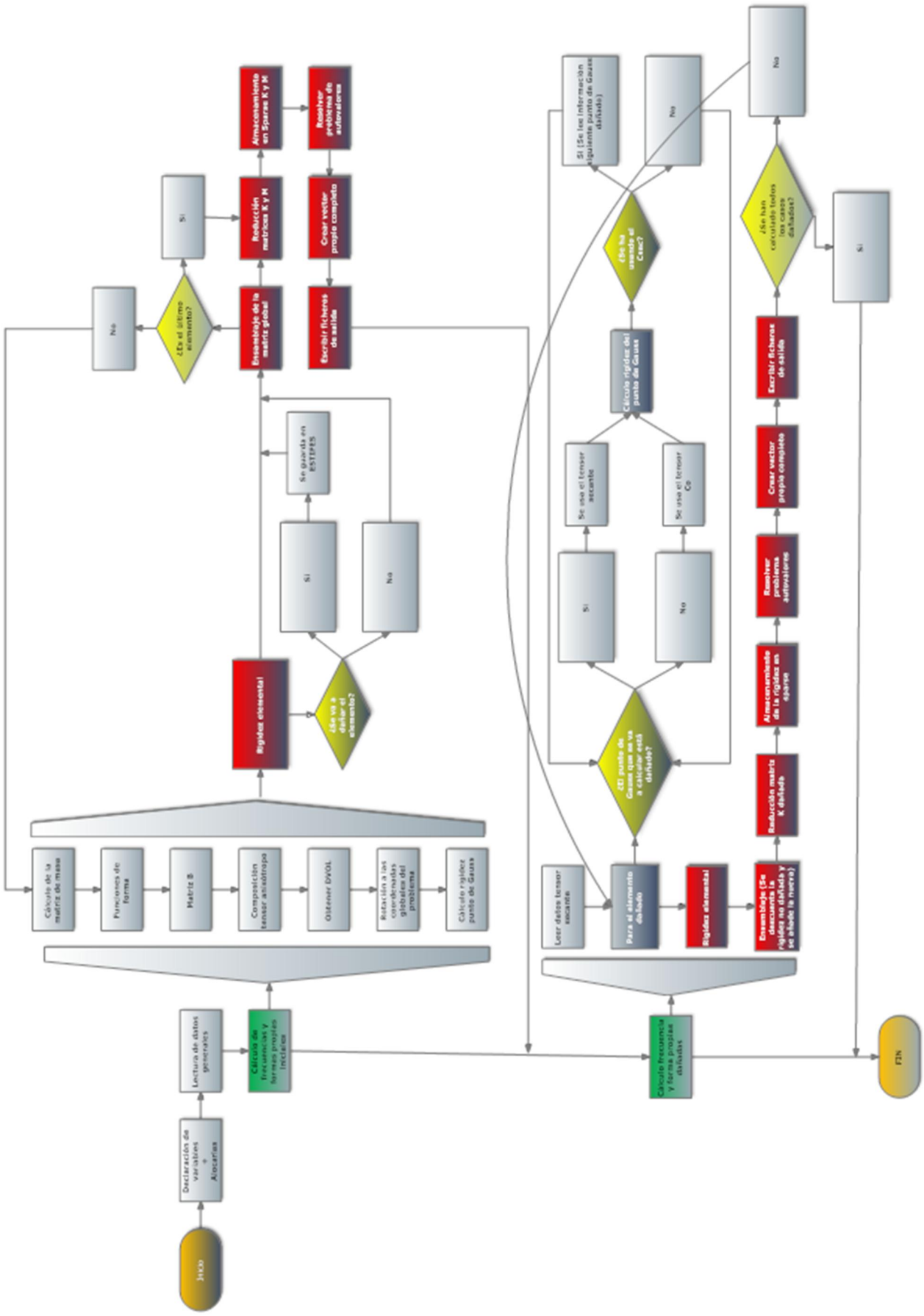


Fig. 7.7.1.1 Representación esquemática de la estructura de CFYFP

Para tener más claro el funcionamiento del nuevo programa, primero se detallará la rutina *CFYFP.f*, ya que es la base de todo el funcionamiento y la rutina principal. A continuación se explicarán las instrucciones de las otras rutinas nuevas. Para terminar se comentarán los cambios realizados en las ya existentes.

7.7.2 Rutina principal programa CFYFP (*CFYFP.F*)

Esta rutina es la más importante porque como se ha comentado es la columna vertebral del programa en la que se realizan la mayoría de llamadas a otras rutinas y la que permite seguir todo el proceso que se realiza. Es la encargada de ir relacionando los resultados obtenidos en una subrutina con la siguiente y la que integra las rutinas reutilizadas ya existentes en el PLCD, con las proporcionadas por Xavier Martínez y con las nuevas que se han tenido que crear para que el programa desarrolle las funciones que se pretendían.

Como en todas las rutinas lo primero que hay que hacer es la declaración de variables a utilizar, en este caso tenemos que declarar cuatro tipos de elementos: valores enteros y reales, matrices y vectores de dimensión conocida y desconocida, por el momento, ya que posteriormente se procederá a darlos la pertienente dimensión. Al darse la circunstancia que muchas variables utilizadas eran las mismas que el PLCD, se ha decidido utilizar los módulos `ENTEROS`, `MATRICES` y `PARAM3D`, se han modificado ya que incorporaban multitud de variables que en este caso no hacían falta y se han quitado, asimismo también se usa un módulo para las matrices en formato SPARSE (`Sparse_SCSR`) que se utilizaba en la rutinas de Xavi Martínez. Consecuentemente, se ha usado también la rutina de alocar variables `ALLOCA_V3D` que ya se usaba en el PLCD. Después de definir las variables y abrir los archivos de entrada de datos, se leen unos datos generales para todo el programa como son: el nombre del problema, el tipo de problema, el número de nodos, el número de elementos, el número de materiales compuestos utilizados y un indicador de si el problema es en pequeñas o grandes deformaciones (*CPROB, TYPEP, NPOIN, NELEM, NMATC, LARGE*). Con estos valores ya podemos dar dimensión a la mayor parte de las variables que se habían dejado sin una conocida al principio del programa.

Seguidamente se prosigue realizando tareas que servirán a lo largo de toda la rutina: guardar las conectividades nodales de los elementos (*CON*), el material asociado a cada elemento (*MAT*) y los ángulo de Euler (*MATEULER*), almacenar las coordenadas de todos los nodos (*COOR*), leer las restricciones impuestas a dichos nodos (*RESTR*), calcular los datos de la base de datos 6 (grados de libertad de cada nodo y los acumulados), establecer el valor de las constantes de integración de Gauss (*POSGP* y *WEIGP*), por último se realiza un bucle sobre los materiales compuestos para tener la densidad y el tensor elástico (C_o) de cada uno de ellos.

Finalmente hay que mirar en que elementos se va a necesitar actualizar en la fase de daño para guardar sus matrices elementales de rigidez no dañadas. Para ello, primero hay que recorrer *Csec.txt* para ver cuantos registros contiene, como la dimensión de este fichero es desconocida utilizamos la instrucción `DO WHILE (.NOT.EOF(42))`que nos permite que mientras no se llegue al final de la unidad de lectura se realizarán las instrucciones dentro del bucle, en este caso, se irán contando el número de registros que contiene el archivo, en este punto se puede observar que también estamos realizando un proceso de lectura porque tiene que realizarse alguna acción para que el bucle avance, pero el dato que lee se asigna a una variable que contiene el anterior, así que no se van guardando los valores. Una vez su dimensión es conocida y sabemos el número de veces que se actualizará la matriz de rigidez (o cuantos pasos no lineales tenemos y, por tanto, se sabe cuantas matrices de rigidez diferentes se calcularán) se hace lo siguiente: se buscan los elementos que hayan sido dañados en el último paso registrado para el análisis de frecuencias, se cuentan y se guardan en un vector cuales son esos pasos. Se hace en el último paso porque el daño es acumulativo y todos los elementos que se dañan en algún momento aparecen como dañados en el último paso pero no quiere decir que todos los pasos tengan todos esos elementos dañados. Con ello ya se puede alocar la matriz (*ESTIFES*) que almacenará temporalmente las matrices de rigidez elementales no dañadas de los elementos que en algún momento se dañarán, más adelante se hablará más en profundidad de este proceso, tanto del guardado de la matriz como de su posterior uso, ya que se trata de un aspecto fundamental para el buen funcionamiento del programa.

Cálculo de las frecuencias y valores propios iniciales

Una vez realizados estos trámites iniciales pasamos al primer eslabón del proceso que vamos a desarrollar: el cálculo de la rigidez elemental inicial, es decir la rigidez que tiene en elasticidad un elemento, como no podemos garantizar que todos los elementos sean exactamente iguales, habrá que repetir el proceso para todos los elementos. Para ello seguimos los siguientes pasos: hacemos un bucle sobre cada elemento y pasamos a la variable *ELCOD* las coordenadas nodales del elemento en cuestión, almacenadas previamente en *COOR*. Seguidamente se calcula la matriz de masa, que solamente tendrá que hacerse una vez en todo el problema porque se supone constante, para ello se utiliza una rutina ya existente en el PLCD, `CALL MASMATV3D (ELEM, MAT, COOR, DENS)`. Esta rutina permite calcular la matriz de masa como matriz consistente o de manera diagonal, en una sección posterior de esta tesina se comentará las diferencias existentes entre estas dos posibilidades, pero se hará de manera diagonal. Realizamos un bucle sobre los puntos de Gauss (8 si el tipo de elementos usados son hexaedros o 1 si usamos tetraedros) y dentro de este bucle se realiza el cálculo de la rigidez.

Primero de todo se han de calcular las funciones de forma (`CALL SFR3D(EXISP,ETASP,EZETA)`) y seguidamente las coordenadas de los puntos de integración, la matriz Jacobiana, su determinante y la inversa y las derivadas cartesianas (todo ello se realiza con la llamada `CALL JACOB3D(XJACM, DJACB,GPCOD)`). Se hace una llamada a otra subrutina para obtener la matriz cinemática **B** (`CALL BMATSOL_V3D(BMATX,GPCOD,FGRAD)`) y se compone el tensor anisótropo (*Co*) a partir de vector que habíamos almacenado previamente en el PLCD y se había guardado en el fichero *Co.txt*. Lo único que falta para poder calcular la rigidez es obtener el *DVOL* que lo hacemos seguidamente. Si estamos en grandes deformaciones nos hará falta calcular el valor de las tensiones en el paso anterior en este punto del bucle y para ello se tendrá que utilizar la información que se ha guardado previamente en el PLCD en el fichero de salida *Strsg.txt*. Antes de poder calcular la rigidez tenemos que rotar todo a los ejes globales del problema, finalmente ya se puede calcular la matriz de rigidez del punto de Gauss del elemento que se esté examinando (`CALL KMATRI_V3D (BMATX, DVOL, CO, STRSG)`).Una vez finaliza el bucle sobre los puntos de Gauss, se tendrá la rigidez de un elemento. Si el elemento calculado se va a dañar durante el proceso, se guardará su

valor junto con el número del elemento en la matriz *ESTIFES*. El cálculo de la rigidez de cada elemento puede observarse en la figura 7.7.2.1. Para almacenar la matriz del elemento (*ESTIF*), la transformamos en un vector y la copiamos en otra matriz mayor (*ESTIFES*), que al acabar el bucle sobre todos los elementos contendrá en cada fila el número de elemento y la matriz descompuesta de cada elemento dañado en el último paso.

```
        DO IGAUS=1,NGAUX
            DO JGAUS=1,NGAUY
                DO LGAUS=1,NGAUZ

    EXISP=POSGP(IGAUS)
    ETASP=POSGP(JGAUS)
    EZETA=POSGP(LGAUS)

    CALL SFR3D(EXISP,ETASP,EZETA)

    KGASP=KGASP+1


    CALL JACOB3D(XJACM,DJACB,GPCOD)

    CALL BMATSOL_V3D(BMATX,GPCOD,FGRAD)
COMPONEMOS EL TENSOR Co, A PARTIR DEL VCO
    CALL COMPON_V3D(VCO(MAT(ELEM),:),CO,36,1,1)
CÁLCULO DEL DVOL
    PESO=WEIGP(JGAUS)
    PESO1=WEIGP(LGAUS)
    DVOL=DJACB*WEIGP(IGAUS)*PESO*PESO1

SI ESTAMOS EN GRANDES DEFORMACIONES ACTUALIZAMOS EL VALOR DE LAS TENSIONES (STRSG)
    IF(LARGE.EQ.1)THEN
    READ(57)INDX,PGAUS,ICARG,IINCS,STRSG
    ENDIF


    A1 = MATEULER(ELEM,1)
    A2 = MATEULER(ELEM,2)
    A3 = MATEULER(ELEM,3)
------ ESTABLECE LA MATRIZ CONSTITUTIVA DEL COMPUESTO DE CADA ELEMENTO EN S. GLOBAL
        IF(A1**2+A2**2+A3**2.GT.1.e-9) THEN
            dir = 2  ! local Elemento to global Estructura
            call ROTM(CO,A1,A2,A3, mstr1, dir)
        END IF

    CALL KMATRI_V3D(BMATX,DVOL,CO,STRSG)

                ENDDO
            ENDDO
        ENDDO
```

Fig. 7.7.2.1 Cálculo de la rigidez elemental elástica. Código mostrado entre las líneas 238 y 288.

El siguiente paso debe ser el ensamblaje de la matriz de rigidez global de la estructura, para ello necesitaremos la información perteneciente a las conectividades entre los diferentes nodos. Para hacerlo, creamos un bucle sobre todos los elementos de la matriz elemental (matriz cuadrada de dimensión igual a los tres grados de libertad por el número de nodos) para formar la matriz de rigidez global. Inicializamos dos vectores

que nos harán falta más adelante (*C* y *F*), que se usarán para calcular la posición de cada elemento en la matriz global. Utilizando una serie de variables auxiliares (*AUX1*, *AUX2*, *AUX3*, *AUX4*) fijamos un valor que nos servirá para decidir la posición en la matriz global, las dos primeras hacen lo mismo (una para la posición de la fila y la otra para la de la columna), dividimos la posición del elemento entre tres y le sumamos 0.9 (no hay ninguna razón para que sea 0.9, puede ser cualquier valor entre un tercio y uno, sin ser ninguno de los extremos) para hacer que los tres grados de libertad den un mismo valor, el valor del nodo en el que se encuentran. Para ello nos quedamos solamente con la parte entera (*AUX3* y *AUX4*). La parte del código encargada del ensamblaje se puede ver en la figura 7.7.2.3. Además de ensamblar la matriz de rigidez también se tiene que hacer lo mismo con la matriz de masa elemental que se ha calculado con anteioridad.

```
C     ENSAMBLAJE DE LA MATRIZ DE RIGUIDEZ GLOBAL INICIAL [KO]G

            F(1:8)=1.0
            DO ELEMF=1,3*NNODE
                C(1:8)=1.0
                DO ELEMC=1,3*NNODE

                    AUX1=(ELEMF*0.333)+0.9
                    AUX2=(ELEMC*0.333)+0.9
                    AUX3=AUX1
                    AUX4=AUX2

                    GLOBF=((CON(ELEM,AUX3)-1)*3)+F(AUX3)
                    GLOBC=((CON(ELEM,AUX4)-1)*3)+C(AUX4)

                    C(AUX4)=C(AUX4)+1

                    STIFFGO(GLOBF,GLOBC)=STIFFGO(GLOBF,GLOBC)
                    +ESTIF(ELEMF,ELEMC)


                   MAS(GLOBF,GLOBC)=MAS(GLOBF,GLOBC)+EMASA(ELEMF,ELEMC)

                ENDDO
                F(AUX3)=F(AUX3)+1
            ENDDO
```

Fig.7.7.2.3 Sección del código encargada de realizar el ensamblaje de la matriz de rigidez y masa globales.

A continuación se muestra un ejemplo de lo que hace esta parte del código del ensamblaje para que quede más claro:

Ejemplo

Ejemplo para los tres primeros elementos de una matriz elemental (posiciones: (1,1), (1,2) y (1,3)) y para los elementos (3,4), (3,5) y (3,6):

	Elemento					
	1,1	***1,2***	***1,3***	***3,4***	***3,5***	***3,6***
AUX1	*1.233*	*1.233*	*1.233*	*1.899*	*1.899*	*1.899*
AUX2	*1.233*	*1.566*	*1.899*	*2.232*	*2.565*	*2.898*
AUX3	*1*	*1*	*1*	*1*	*1*	*1*
AUX4	*1*	*1*	*1*	*2*	*2*	*2*

Con esto sabemos que nodo dentro del elemento se está tratando, pero no sabemos si ese primer nodo del elemento es el nodo 1 o el 50 del global. Para ello tenemos que mirar la información de las conectividades y para saber el grado de libertad en el que nos encontramos utilizaremos los dos vectores creados anteriormente (*C* y *F*). Para saber la posición global haremos: mirar el valor del vector de conectividades (sabremos en que nodo global nos encontramos), le restamos uno y lo multiplicamos por tres, seguidamente le sumamos un valor del vector auxiliar adecuado (dependiendo si estamos mirando la posición de las filas→F o de las columnas→ C). Finalmente hay que actualizar el valor de estos vectores auxiliares, sumando uno a la posición consultada en el caso de las columnas y cuando acabemos la fila, en ésta. Con la finalidad de clarificar la explicación se continuará con el ejemplo anterior. Para realizarlo nos hace falta la información de las conectividades para ello supondremos: *CON=[1 2 3 4 5 6 7 8]*

	Elemento					
	1,1	***1,2***	***1,3***	***3,4***	***3,5***	***3,6***
GLOBF	*1*	*1*	*1*	*3*	*3*	*3*
GLOBC	*1*	*2*	*3*	*4*	*5*	*6*
C	*[1 1 1 1 1 1 1 1]*	*[2 1 1 1 1 1 1 1]*	*[3 1 1 1 1 1 1 1]*	*[4 1 1 1 1 1 1 1]*	*[4 2 1 1 1 1 1 1]*	*[4 3 1 1 1 1 1 1]*
F	*[1 1 1 1 1 1 1 1]*			*[3 1 1 1 1 1 1 1]*		

Una vez conocida la posición en la matriz global, lo único que queda es actualizar el valor de esa posición, por ello añadimos el valor nuevo al valor anterior. Al acabar el bucle tendremos la matriz de rigidez global completa inicial o elástica, que utilizaremos más tarde para calcular las rigideces globales dañadas, ya que de esta matriz de la estructura nueva es la que se modificará ligeramente en cada paso, para no tener que realizar el cálculo de la matriz completa en cada paso.

Las matrices que tenemos en este momento son matrices completas (**K** y **M**), no se han tenido en cuenta todavía que hay una serie de grados de libertad restringidos y que se podrán utilizar en su lugar matrices más pequeñas, matrices reducidas. Para hacerlo se crea un vector que tiene tantos registros como grados de libertad el problema, cada posición contiene un uno (restringido) o un cero (libre) según la propiedad del nodo, en los nodos que se ha aplica un desplazamiento aunque puedan parecer fijos en un principio, tienen completa libertad (por ello existen los tres *if*, para realizar está corrección). Para ello, se leen las restricciones y se hacen dos bucles, uno sobre las filas de **K** y **M**, y otro sobre las columnas de **K** y de **M**. Durante este recorrido sobre ellas se eliminan aquellas filas y columnas que están restringidas, quedando finalmente sólo la información de los nodos libres. El proceso de la reducción de las matrices puede verse en la figura 7.7.2.4

```
REDUCIMOS LA MATRIZ
      VRESTR=0.0
      DO I=1,NPOIN
           RESTX=(RESTR(I,1))/100
           RESTY=(RESTR(I,1)-(RESTX*100))*0.1
           RESTZ=RESTR(I,1)-RESTX*100-RESTY*10
           VRESTR(3*I-2)=RESTX
           VRESTR(3*I-1)=RESTY
           VRESTR(3*I)=RESTZ
           IF (RESTR(I,2).NE.0)THEN
           VRESTR(3*I-2)=0
           ENDIF
           IF (RESTR(I,3).NE.0)THEN
           VRESTR(3*I-1)=0
           ENDIF
           IF (RESTR(I,4).NE.0)THEN
           VRESTR(3*I)=0
           ENDIF
      ENDDO

      STIFAUX(:,:)=0.0
      STIFRED(:,:)=0.0
      MASAUX(:,:)=0.0
      MASRED(:,:)=0.0

      DO I=1,NTOTV
           IF(VRESTR(I).EQ.0)THEN
           STIFAUX(CRED+1,:)=STIFFGO(I,:)
           MASAUX(CRED+1,:)=MAS(I,:)
           CRED=CRED+1
       ENDIF
      ENDDO



      DO I=1,NTOTV
           IF(VRESTR(I).EQ.0) THEN
           STIFRED(:,AUXRED+1)=STIFAUX(:,I)
           MASRED(:,AUXRED+1)=MASAUX(:,I)
           AUXRED=AUXRED+1
           ENDIF
      ENDDO
```

Fig. 7.7.2.4 Proceso de reducción de las matrices teniendo en cuenta los grados de libertad restringidos

Posteriormente se procede al almacenamiento en formato SPARSE para ello se realiza la siguiente llamada: `CALL ALMACEN (STIFRED2, CRED, SPARK)`. Los detalles de la subrutina *Almacen.f* se encuentran en la sección 7.7.3.

Finalmente llegamos al punto en el que se calculan los valores y vectores propios para el caso elástico. Primero de todo, se ha de decidir el número de valores propios que se quieren calcular, la tolerancia aceptada, el número máximo de iteraciones permitidas para alcanzarla y si se va a querer realizar una animación de alguna de las formas propias se pedirá en este punto (*FANIM*), todos estos datos se leerán del archivo de entrada *frecuencias.dts*. Después damos dimensión a las variables de salida (*SVAL*, *SVEC*, *EIGENFREQ*,, *SPRED*) en las que se almacenarán los valores que se desean calcular y se pasa a realizar el llamado a la rutina encargada de realizar los cálculos utilizando el método de iteración en el subespacio (`CALL Eigen_SIM(SPARK,SPARM,new_size,sval,svec,neig,tolersim,maxi,iout,iitersim,StVer,sturm1,sturm2,pred,spred,R)`). La explicación concerniente a esta rutina se encuentra en la sección 7.6, por ahora digamos que funciona correctamente y que se resuelve el problema de autovalores, la rutina nos devolverá: los valores propios, los correspondientes vectores y un vector con los valores convergidos para usar como vector predictor en el siguiente paso y con el cual no se tiene que realizar ninguna acción. Para concluir, nos queda escribir los datos en un fichero de salida que nos permita realizar el postproceso. Pero antes no hay que olvidar que se ha realizado la reducción de la matriz ensamblada completa debido a que algunos grados de libertad estaban restringidos, para no tener problemas más adelante hay que reescribir los vectores propios que hemos calculado pero teniendo en cuenta está reducción, para ello hacemos una llamada a una de las nuevas rutinas creadas: `CALL VectProp (svec,NEIG,CRED,NTOTV,SVECC,VRESTR)` y que se describe más adelante en la sección 7.7.3. Como los valores calculados no son las frecuencias de la estructura, sino que corresponden al valor propio, las calculamos para poder guardarlas. Para acabar solamente nos quedará escribir los resultados: la relación daño-frecuencia en una tabla (*Daño-frecuencia-nombre_del_problema.res*) y las correspondientes formas propias en un fichero (*nombre_del_problemaF.post.res*) para su posterior visualización en un programa de postprocesado (en nuestro caso se usará el GID). Para ello hacemos un llamado a la nueva rutina `ESCRIBIR(eigenfreq,NEIG,NTOTV,SVECC,0,0,1,CPROB)`, programa descrito en las páginas siguientes, en la sección 7.6.3. De manera opcional, en

este punto se puede realizar un llamado a la subrutina *ANIMACION* (`CALL ANIMACION (SVECC(:,FANIM), FANIM, NTOTV,CPROB)`) que se encargará de crear los datos necesarios para poder visualizar una animación de los *fanim*-primeros modos de vibración de la estructura, en la sección 7.7.3 se describen sus instrucciones y su funcionamiento.

Cálculo de las frecuencias y valores propios en no linealidad

La estructura de la sección de la rutina encargada de realizar el cálculo en plasticidad no difiere mucho de la mostrada hasta el momento para el caso base (no aparecen rutinas o instrucciones nuevas), sin embargo contiene ciertos puntos que por fuerza han de ser diferentes y que seguidamente se explican.

Al principio de la rutina se ha calculado la dimensión del fichero *csec.txt* (*CTRL2*), usaremos este valor para dar dimensión a tres vectores que aún no la tenían y que usaremos en el proceso de actualizar la matriz de rigidez original (VELEM, VAUX, VPGAUS) y que se corresponden con las columnas descritas con el mismo nombre en el apartado en el que se han explicado los distintos ficheros utilizados. Una vez tengan dimensión, procederemos a llenarlos. Para hacerlo recorremos otra vez el fichero *Csec.txt* pero esta vez guardamos la información que necesitamos (Figura 7.7.2.5).

```
CÁLCULO DE LA RIGUIDEZ ELEMENTAL EN PLASTICIDAD [KP]E Y ACTUALIZAR LA MATRIZ GLOBAL

    CTRL6=1 !Para que sólo entre en el primer paso
    STIFFGP=0.0

ALOCAMOS DIMENSIONES DE LOS VECTORES QUE USAREMOS A CONTINUACIÓN

    ALLOCATE(VELEM(CTRL2+1)) ! +1 Para que no de error en el último paso de
    ALLOCATE(VAUX(CTRL2+1))  ! actualización.
    ALLOCATE(VPGAUS(CTRL2))

VOLVEMOS AL PRINCIPIO DEL ARCHIVO Y LLENAMOS LOS VECTORES QUE ACABAMOS DE CREAR
    REWIND(42)
    DO I=1,CTRL2

     READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
     VAUX(I)=AUX
     VELEM(I)=ELEM
     VPGAUS(I)=PGAUS

    ENDDO
```

Fig. 7.7.2.5 Comienzo de la parte no lineal y lectura de información de *Csec.txt*

Hasta este punto la rutina no lineal solamente se ejecuta una vez, pero a partir de aquí se repetirá para realizar las sucesivas actualizaciones de la matriz de rigidez, por ello situamos una bandera en esta posición. A partir de aquí se puede decir que realmente empieza el proceso de cálculo de la matriz de rigidez dañada. Primero se

cuentan cuantos puntos de Gauss han entrado en plasticidad en un mismo paso, sin tener en cuenta si son o no de diferentes elementos, y posteriormente se cuenta de cuantos elementos son esos puntos de Gauss. Todo ello se muestra en la figura 7.7.2.6

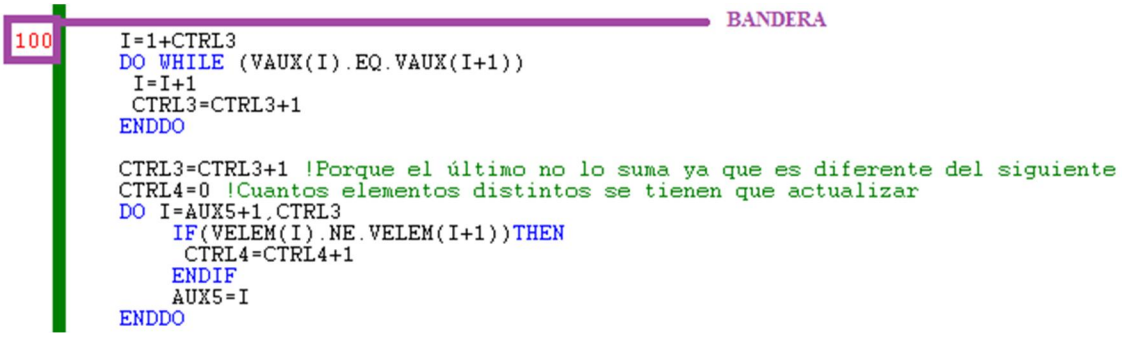


Fig. 7.7.2.6 Bandera que marca el punto a partir del cual algunas instrucciones se repetirán y la creación de dos contadores que se usarán para la actualización.

A continuación se leerán los datos del punto de Gauss y el Csec para la actualización, pero sólo se hará en esta posición la primera vez que se pase. Obviamente esta información tiene que ser leída más de una vez, pero se hará al final del bucle que se encarga de reemplazar la matriz de rigidez y, por tanto, no tiene que leerse en este punto del código más veces.

Seguidamente empezará un proceso que será similar al descrito para el caso base o elástico pero con una diferencia muy importante, no se tendrá que crear la matriz de rigidez elemental para cada elemento sino que sólo para aquellos que tengan algún punto de Gauss dañado. Por esta razón creamos un bucle para repetir el proceso tantas veces como elementos a actualizar en el paso de carga estudiado tengamos (el contador `CTRL4` que se ha visto previamente tiene está función). Posteriormente se leen las coordenadas y se hace un bucle sobre todos los puntos de Gauss del elemento en cuestión. Lo primero que hay que hacer es ver si el punto de Gauss que se va a examinar es uno de los que ha entrado en daño (figura 7.7.2.7), como se han escrito en orden en el fichero *Csec.txt* sólo se ha de comprobar el que hemos leído. Si coinciden los dos puntos de Gauss utilizaremos el tensor secante que hemos leído del fichero, en caso contrario se usará el Co como en el caso elástico. A partir de este punto el proceso continúa como en el caso base, siguiendo los mismos pasos hasta calcular la matriz de rigidez. Después de calcularla si se ha utilizado ya la información leída de *Csec.txt* (`CONTROL=1`) se leerá el punto siguiente, sino se seguirá con el que ya se tiene en la memoria. Cuando finalice el elemento, se ensamblará la nueva rigidez en la posición

conveniente, el proceso de ensamblaje es ligeramente distinto al caso base debido a que se ha de descontar la contribución elástica de ese elemento y sumarle la dañada (fig. 7.7.2.8). Se continuará reduciendo la matriz, almacenándola en formato SPARSE y finalmente calculando las nuevas frecuencias y formas propias. Antes de volver a la bandera se guardarán los resultados para su posterior uso, se hará de igual manera que en el caso base, utilizando la rutina *ESCRIBIR*. Finalmente se vuelve a la bandera y se procede con el siguiente caso de actualización.

```
IF ((IGAUSL.EQ.IGAUS).AND.(JGAUSL.EQ.JGAUS).AND.
    (LGAUSL.EQ.LGAUS).AND.(ELEMV.EQ.ELEM))THEN

CALL COMPON_V3D(VCTAN,CTAN,36,1,1) !Es el tensor secante aunque
                                   !ponga tangente
CONTROL=1 !Sirve para indicar si hemos usado el Csec y necesitamos
ELSE !leer el siguiente.
CALL COMPON_V3D(VCO(MAT(ELEMV),:),CTAN,36,1,1)
!Se llama Ctan pero en este caso sería Co, pero es para no hacer
!los mínimos cambios en el código
CONTROL=0
ENDIF
```

Fig. 7.7.2.7 Comparación del punto de Gauss leído y el que toca calcular, si coincide se utiliza la información leída.

```
STIFFG(GLOBF,GLOBC)=STIFFG(GLOBF,GLOBC)
-ESTIFO(ELEMF,ELEMC)+ESTIF(ELEMF,ELEMC)
```

Fig. 7.7.2.8 Actualización de la matriz de rigidez, descontamos la contribución de la parte elástica y le sumamos la plástica.

El proceso se repetirá tantas veces como sea necesario para procesar los datos obtenidos del PLCD, hasta que se alcance la condición de salida del bucle (Figura 7.6.2.9) y el programa finalice.

```
IF(AUX6.EQ.CTRL2)THEN !Para salir del bucle una vez haya llegado al final
    CONTINUE
ELSE
    GO TO 100
ENDIF
```

Fig. 7.7.2.9 Condición situada al final de la rutina y que decide si se vuelve a la bandera y se prosigue con otra actualización o si se ha finalizado el tratamiento de los datos proporcionados por el PLCD y hay que salir del programa.

A continuación se muestra un pequeño ejemplo del proceso que se realiza para actualizar y calcular las matrices de rigidez degradadas, para que quede más claro el funcionamiento de este punto que es clave en el funcionamiento del programa CFYFP.

Por poner ejemplo se supone que la siguiente de matriz es una matriz de rigidez global de una barra discretizada en dos elementos para un momento inicial sin daño alguno:

$$K_O = \begin{pmatrix} 5 & -5 & 0 \\ 5 & 9 & -4 \\ 0 & -4 & 4 \end{pmatrix}$$

En la que supondremos, que el primer elemento de la barra tiene una rigidez de cinco y el segundo de cuatro. Se va a realizar el proceso de actualización para dos hipotéticos pasos en los que se ha degradado la rigidez. Se sabe que en el segundo paso estarán dañados todos los elementos que estuvieran dañados en el primero más los que se hayan dañado en el segundo, en este caso se sabe que se habrán dañado los dos elementos al final. Por lo tanto se habrá guardado en *ESTIFES* las rigideces de estos elementos (primero y segundo) sin daño alguno. Que en este caso serán las siguientes matrices pero en forma vectorial.

$$k_1 = \begin{pmatrix} 5 & -5 \\ -5 & 5 \end{pmatrix} \qquad k_2 = \begin{pmatrix} 4 & -4 \\ -4 & 4 \end{pmatrix}$$

Primer paso

Se han dañado el primer elemento y sus rigidez es ahora 4. El proceso que sigue la rutina es el siguiente:

- Se ha identifica el primer elemento dañado, en este caso el primero.
- Se calcula la nueva matriz de rigidez elemental

$$k' = \begin{pmatrix} 4 & -4 \\ -4 & 4 \end{pmatrix}$$

- Se busca en qué posición dentro de la matriz de rigidez le corresponde a cada elemento de la matriz elemental.
- Se actualizan los valores correspondientes de la matriz de rigidez global:

k_{11} = 5 – 5 + 4 = 4

k_{12} = -5 – (-5) +(- 4) = - 4

k_{21} = -5 – (-5) +(- 4) = - 4

k_{22} = 9 – 5 + 4 = 8

- Una vez se han actualizado todos los valores del mismo paso se obtiene la nueva rigidez degradada:

$$K' = \begin{pmatrix} 4 & -4 & 0 \\ -4 & 8 & -4 \\ 0 & -4 & 4 \end{pmatrix}$$

- Con esta nueva rigidez se calcularán las frecuencias de nuevo.

Segundo paso

Los elementos dañados en el primer caso se han dañado más pero además aparece un nuevo elemento dañado, el segundo. La rigidez del primero pasa a valer 2 y la del segundo 3. El proceso que sigue la rutina es el siguiente:

- Se ha identifica el primer elemento dañado, en este caso el primero.
- Se calcula la nueva matriz de rigidez elemental.

$$k' = \begin{pmatrix} 2 & -2 \\ -2 & 2 \end{pmatrix}$$

- Se busca en qué posición dentro de la matriz de rigidez le corresponde, en este caso la primera posición.
- Se actualizan los valores correspondientes de la matriz de rigidez global:

k_{11} = 5 – 5 + 2 = 2

k_{12} = -5 – (-5) +(- 2) = - 2

k_{21} = -5 – (-5) +(- 2) = -2

k_{22} = 9 – 5 + 2 = 6

- Una vez se han actualizado todos los valores del primer elemento, se procede de igual manera pero para el segundo elemento:

k_{22} = 6 – 4 + 3 = 5

k_{23} = -4 – (-4) + (-3) = - 3

k_{32} = -4 – (-4) + (- 3) = -3

k_{33} = 4 – 4 + 3 = 3

- Una vez se han actualizado todos los valores del mismo paso se obtiene la nueva rigidez degradada:

$$K'' = \begin{pmatrix} 2 & -2 & 0 \\ -2 & 5 & -3 \\ 0 & -3 & 3 \end{pmatrix}$$

- Con esta nueva rigidez se calcularán las frecuencias de nuevo.

Los puntos importantes de este proceso son los siguientes: se parte siempre de la matriz no dañada, se identifican los elementos dañados, se les resta la contribución elástica y se le añade la contribución dañada calculada en las rutinas a partir del tensor secante. Una vez actualizados todos los elementos de un mismo paso se resuelve el problema de autovalores. Se repite el proceso tantas veces como pasos dañados se hayan obtenido en el PLCD.

7.7.3 Otra rutinas nuevas incluidas en el CFYFP

Como se ha visto, hay una serie de rutinas nuevas que se utilizan a lo largo del proceso, seguidamente se procederá a explarlas.

Almacen.f

Como ya se ha comentado anteriormente, *almacen.f* es una subrutina encargada de realizar el almacenamiento de las matrices en formato Sparse. Dada una matriz cuadrada simétrica y su dimensión devuelve una variable Sparza que contiene toda la información pero ocupando menos tamaño. El código completo se encuentra en el Anejo I pero a continuación se mostrarán las instrucciones principales.

Ante la imposibilidad de crear vectores o matrices sin una dimensión asignada, nos vemos obligados a realizar un primer pase sobre la matriz para conocer la

dimensión de las variables a utilizar y poder alocarlas (Figura 7.7.3.1). A pesar de tratarse de matrices simétricas, tanto la de rigidez como la de masa, debido a la configuración de las rutinas de cálculo nos vemos obligados a almacenar la matriz completa.

```
HAGO UN PRIMER RECORRIDO SOBRE LA MATRIZ PARA CONOCER LAS DIMENSIONES DE LA MATRIZ
AUX=0
DO FIL=1,NTOT
    DO COL=1,NTOT
        IF (STIF(FIL,COL).NE.0) THEN
          AUX=AUX+1
        ENDIF
    ENDDO
ENDDO

ALOCO LAS DIMENSIONES DE LAS VARIABLES A UTILIZAR
CALL allocate_SCSR (A, NTOT, AUX)
```

Fig. 7.7.3.1 Primer bucle sobre la matriz que se quiere almacenar y alocamiento de las variables

Seguidamente se recorre la matriz por segunda vez y se asignan valores a las variables creadas. Éstas son para: los valores no nulos de la matriz, columna en la que se encuentra el valor no nulo, posición en el vector de valores no nulos donde empieza una nueva fila de la matriz, dimensión de la matriz inicial y número total de valores no nulos (Figura 7.7.3.2). De la misma forma que la mostrada en la sección 7.3.2 que se comentó este tipo de almacenamiento.

```
RECORRO LA MATRIZ POR SEGUDA VEZ PARA ALMAMCENARLA EN FORMATO SPARSE
p=1
o=1
DO FIL=1,NTOT
    AUX=1
    DO COL=1,NTOT
     IF (STIF(FIL,COL).EQ.0) THEN
     ELSE
         A%Val(p)=STIF(FIL,COL)
         A%j(p)=COL
         p=p+1
         IF (AUX.EQ.1) THEN
            A%i(o)=p-1
            o=o+1
            AUX=2
         ENDIF
      ENDIF
    ENDDO
ENDDO

DAMOS VALOR AL ÚLTIMO ELEMENTO DEL VECTOR DE POSICIONAMIENTO A%i
PARA QUE FUNCIONEN LAS RUTINAS SUCESIVAS TIENE QUE VALER A%nnz+1

A%i(A%n+1)=A%nnz+1
```

Fig. 7.7.3.2 Segundo recorrido sobre la matriz y asignación de valores a las variables. Por último hay que asignar valor a la última posición del vector i.

Por último y antes de terminar se asigna un valor a la última posición del vector i (*A%i(n+1)*) (Figura 7.7.3.2) debido a la estructura del programa, más adelante puede verse una explicación detallada de las razones de esta asignación. Al finalizar la rutina, la variable presentará una estructura similar a la siguiente:

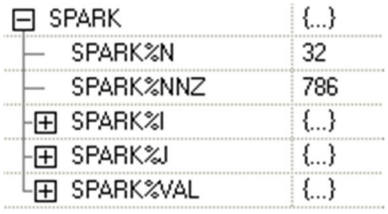


Fig. 7.7.3.3 Ejemplo variable almacenada en formato sparse, en este caso la variable se llama Spark

VectProp.f

La función de esta subrutina es obtener el vector propio completo, ya que el obtenido con el método de iteración en el subespacio al haberse calculado a partir de la matriz reducida y no incluye los grados de libertad restringidos. El código completo de la rutina se encuentra en el Anejo I. El código es muy sencillo, se definen las variables y a continuación se lee el vector de restricciones, si el grado de libertad está restringido se inserta un 0 en esa posición del vector propio completo, en caso de tratarse de un nodo libre se le da el valor calculado anteriormente (figura 7.7.3.4). Como se ha visto en la explicación del código de *CFYFP.f*, se distinguen entre los nodos restringidos (apoyos) y los nodos de carga y *VREST* llega a este punto con los nodos de carga como nodos libres. Aunque realmente no se definan como libres ya que tiene una carga o un desplazamiento impuesto, en el problema de frecuencias serán libres ya que podrán moverse en cualquier dirección según la forma de vibración.

```
DO AUX1=1,NEIG

 AUX3=1

 DO AUX2=1,NTOTV
    IF (VREST(AUX2).EQ.1) THEN
        SVECC(AUX2,AUX1)=0
    ELSE
        SVECC(AUX2,AUX1)=SVEC(AUX3,AUX1)
        AUX3=AUX3+1
    ENDIF

 ENDDO

ENDDO
```

Fig. 7.7.3.4 Formación del vector propio completo a partir del calculado previamente con las rutinas explicadas.

Escribir.f

La subrutina *Escribir.f* recoge los datos que resultan de la rutina *Eigen_SIM* (los vectores propios y las frecuencias, que se han calculado a partir de los valores propios computados) y los escribe en los ficheros de salida, dos archivos diferentes: uno que muestra la evolución de las frecuencias y del daño, y otro, con los vectores propios en

un formato que permita su utilización en el GID (*.post.res*), programa con el que se realizará el postprocesado de los resultados.

Frecuencias-daño

Este archivo es el más sencillo de los dos, en ella se presentan los resultados en forma de tabla, un ejemplo del formato de uno de estos archivos de salida puede consultarse en el Anejo II: Archivos de salida y entrada. Como el nombre del archivo indica se representa el daño que presenta la estructura en el paso y el valor de las frecuencias calculadas en Hercios. Cada fila representa un estado de daño diferente, que empieza con la estructura "nueva" (daño 0) y va aumentando hacia la parte inferior de la tabla hasta alcanzar el mayor nivel de daño calculado con el PLCD.

Para hacerlo primero se crea el encabezado de la tabla que solamente se escribirá la primera vez que se llame a la rutina, es decir en el caso elástico o básico, en el que la estructura todavía no ha sufrido daño alguno. A continuación se crea un bucle sobre el número de valores calculados y se escriben el daño seguido de los resultados de las frecuencias obtenidas.(Figura 7.7.3.6)

Una vez finalizada la creación, los datos pueden ser exportados a otro programa para su tratamiento y estudio.

Vectores propios

Para poder postprocesar fácilmente los resultados obtenidos en el GID el fichero de salida tiene que tener una determinada estructura con unos parámetros fijos y otros que si se pueden variar. La idea antes de crear este archivo era que todos los vectores propios que perteneciera a un misma frecuencia (primer caso (no dañado), segundo caso (empieza el daño), tercer caso (el daño va aumentando),..) se agruparan juntos bajo una misma etiqueta para observar mejor si variaban. Se ha decidido obviar que las primeras frecuencias no son constantes, ya que a medida que se va sufriendo daño la estructura van cambiando ligeramente. Esto se ha hecho ya que el modo de vibración se espera que sea similar en todos los casos.

Antes de escribir los resultados, se hace una normalización de los datos de las formas propias. La normalización se hará de tal manera que el valor máximo del vector sea uno (figura 7.7.3.5).

```
NORMALIZAR LAS FORMAS PROPIAS DE TAL FORMA QUE EL VALOR MÁXIMO SEA 1
DO B=1,NEIG
     A=MAXVAL(ABS(SVECC(:,B)))
     AUXFORM(:,B)=(SVECC(:,B))/A
ENDDO
SVECC=AUXFORM
```

Fig. 7.7.3.5 Normalización de las formas propias antes de su escritura.

La parte de la rutina encargada de escribir esto ficheros presenta una estructura muy similar a el caso anterior, primero hay que introducir un encabezamiento fijo para el archivo (solamente se hace una vez en todo el proceso) y luego para cada modo de vibración se realiza otro encabezamiento (se repite tantas veces como modos de vibración totales calculados, número de frecuencias buscadas por el número de pasos dañados más el caso original). Los datos del vector propio los tenemos en la variable *SVECC* (el vector de cada frecuencia ocupa una de las columnas de la matriz), tenemos que escribir en cada fila un nodo y los desplazamientos que ha sufrido en cada grado de libertad (DX, DY y DZ), para ello se utilizan las instrucciones mostradas en la figura 7.7.3.6, que hacen que se vaya escribiendo los datos del vector de tres en tres y formando filas.

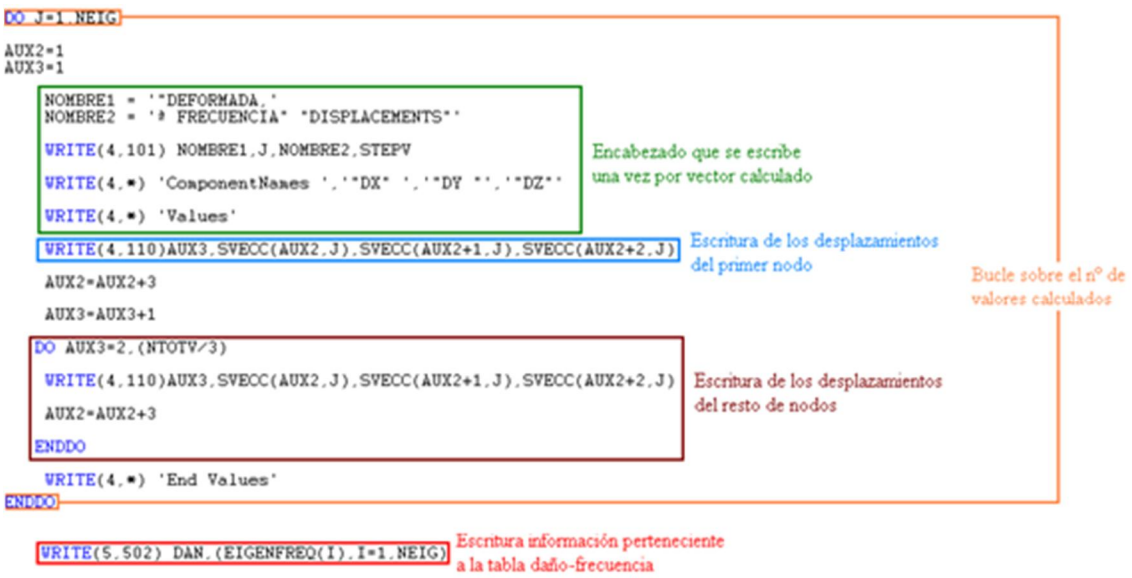


Fig. 7.7.3.8 Elementos principales de la rutina *Escribir.f*

Animacion.f

A diferencia de las otras subrutinas, ésta es totalmente opcional y su llamado viene condicionado por uno de los parámetros de entrada, la razón de esta rutina es poder crear los datos necesarios para poder visualizar los *n* primeros modos de vibración en forma de animación. De la misma forma que las demás rutinas comentadas hasta ahora, el código completo se puede consultar en el Anejo I.

Esta rutina recibe cuatro valores de entrada (`SUBROUTINE ANIMACION (VEC, F, NTOTV, CPROB)`) que representan, de izquierda a derecha: el vector propio correspondiente a la forma que se quiere animar, el número de frecuencia de dicho vector, la dimensión del vector y, por último, el nombre del problema. De todos ellos, es el segundo el que marcará si se realiza el llamado o no, si su valor es nulo la rutina no entrará a calcular la animación.

Sin embargo, la información fundamental es la proporcionada por el vector propio, éste contiene los movimientos en cada dirección que sufre cada nodo del problema, hay que recordar que estos movimientos no son movimientos propiamente dichos, sino que representan una forma, es decir un vector y otro multiplicado por un escalar son la misma forma. Otro concepto importante es que este vector representa las posiciones finales de los nodos, no tenemos pasos intermedios y éstos son precisamente los que vamos a crear. El primer paso es normalizar el vector, para ello buscamos el máximo valor y dividimos todo el vector por éste. En este punto tenemos la forma normalizada en un extremo, para encontrar el otro multiplicamos por menos uno. A continuación se han de crear los incrementos que nos llevarán desde uno hasta otro, en este caso se ha decidido que se realicen 20 pasos entre los dos extremos o, lo que es lo mismo, 10 pasos entre la posición de reposo y cada uno de los finales, pero la adopción del número de incrementos es totalmente arbitrario y dependerá de la fluidez que se pretenda dar al movimiento.

Seguidamente se crea el archivo de salida (*Animacion-Nombre del problema. post . res*) y su encabezado, que será exactamente igual que el descrito para la rutina *Escribir.f* ya que se realizará el postprocesado con el mismo software. Para escribir los resultados se crea un bucle del número de pasos fijados (20 en nuestro caso) y se partirá desde el extremo que se ha multiplicado por menos uno, una vez escrito, se sumará el

incremento del paso y se volverá escribir, al finalizar el bucle nos encontraremos en la otra posición extrema, pero con pasos intermedios.

Para conseguir un movimiento completo, de mínimo a máximo y de máximo a mínimo, se realiza seguidamente otro bucle, con el mismo número de pasos que partiendo de la posición máxima, irá restando el incremento de paso hasta llegar a la mínima, por supuesto se irá escribiendo cada paso intermedio. De tal forma que al abrir este fichero se pueda reproducir los pasos creados en bucle sin fin creando la sensación de que vemos una película de la forma de vibración deseada.

7.8 Programa auxiliar: Mallas-animación.bat

Como se puede ver en la terminación del archivo, se trata de un ejecutable también llamado fichero de procesamiento por lotes. No forma parte de ninguno de los dos programas descritos anteriormente, PLCD y CFYFP, y su utilización no forma parte del proceso del cálculo de las frecuencias y formas propias de una estructura. Su ejecución es totalmente opcional y se usa porque permite realizar el post-proceso de los resultados obtenidos de manera mucho más cómoda. Esto se debe a que nos sirve para:

- Generar una carpeta para cada una de las animaciones creadas.
- Mover cada animación creada a la carpeta correspondiente.
- Hacer una copia de la malla creada por el PLCD, renombrarla para que coincida con cada animación y ponerla en la carpeta adecuada, esto nos permitirá ver los resultados en el GID sin tener que ir renombrando la malla original continuamente.
- Hacer una copia de la malla y renombrarla para que coincida con el nombre del fichero de salida de todas las formas propias calculadas, servirá para no tener que hacer nosotros de forma manual una copia y renombramiento de la malla original para poder visualizar los resultados en el GID.

Se tiene que correr una vez se hayan utilizado el PLCD y el CFYFP y nos pedirá por pantalla el nombre del problema resuelto y el número de animaciones creadas. Estas funciones se podrían haber implementado como parte de las rutinas que forman el CFYFP, pero en ese caso simplemente causarían un gran retraso en el cálculo ya que implicarían como mínimo la lectura de la malla original de la estructura una vez y la

escritura de ésta *neig*+1 veces, siendo *neig* el número de frecuencias buscadas. Todo ello en medio del proceso de cálculo, es por ello que se hace totalmente aparte y de manera opcional, ya que no cambia ninguno de los resultados obtenidos previamente y, simplemente, facilita su tratamiento posterior. El código completo se encuentra en el Anejo I.

7.9 Matriz de masa

Para poder solucionar el problema necesitamos una matriz de masa de la estructura de las mismas dimensiones que la matriz de rigidez, para hacerlo se sigue el mismo procedimiento que para la rigidez, primero se crea la matriz de un elemento y después se ensambla en su posición dentro de una matriz global. Como se ha comentado anteriormente, la matriz de masa se calcula usando una rutina ya existente en el PLCD llamada *MASMATV3D* y que permite dos posibilidades de cálculo, se puede crear la matriz de masa consistente o la diagonal. A continuación se mostrará que realmente no importa cuál de las dos opciones se utilice, desde el punto de vista de los resultados obtenidos.

El método más sencillo para crear la matriz de masa es concentrar ésta en las coordenadas nodales que definen los desplazamientos de traslación, por eso es conocido también con el nombre de método de masas concentradas (lumped matrix). Lo más habitual es distribuir la masa de cada uno de los elementos en los nodos del elemento. Este método supone que el efecto inercial asociado a los grados de libertad angular es cero, aunque se pueden asociar valores finitos a estos grados de libertad. Existen varios métodos para distribuir la masa en los nodos, la rutina del PLCD utiliza las funciones de forma para calcular que masa asigna a cada punto e integra sobre todos los nodos del elemento, finalmente los valores que calcula solamente los asigna a la diagonal de la matriz de masa elemental.

El segundo método para definir la matriz de masa, llamado de masa consistente, se basa en calcular los coeficientes de la matriz de manera similar a la utilizada para

determinar los de la de rigidez. En este caso la rutina calcula los valores de igual manera usando las funciones de forma e integrando pero en vez de concentrar la masa en los elementos de la diagonal los asigna a la posición que corresponda.

Aunque las dos matrices son claramente diferentes el uso de una u otra no presenta grandes diferencias en los resultados finales para las primeras frecuencias propias, por eso y debido a que la matriz de masa diagonal presenta evidentes ventajas desde el punto de vista computacional, es la matriz concentrada la que se usa normalmente en la mayoría de programas comerciales y, en este caso, se usará también para resolver las estructuras.

8. Validaciones

A lo largo del desarrollo de esta tesina se han tenido que ir validando las nuevas partes de código creadas además del funcionamiento global de todas las rutinas. En esta sección se muestran las distintas validaciones realizadas durante el todo el proceso, tanto las parciales como la global.

8.1 Ensamblaje de las matrices

Para validar el sistema de ensamblaje se ha realizado la siguiente prueba: se han generado tres matrices de dimensión 24x24 de manera aleatoria (que simulan una matriz cualquiera de rigidez). Estas matrices representan tres elementos de una columna de 16 nodos y 3 grados de libertad por nodo. Se ha realizado el ensamblaje de forma manual y se ha obtenido una matriz de 48x48. A continuación se han introducido las tres matrices elementales en la rutina de cálculo y se ha guardado en un fichero el resultado del ensamblaje (una matriz de dimensión 48x48). Las dos matrices de "rigidez" se han comparado y se ha podido evidenciar que ambas son iguales, con lo que queda ratificado el proceso de ensamblaje anteriormente descrito en la sección 7.7.2 de este trabajo. Los datos utilizados para realizar la validación de este proceso de ensamblaje pueden verse en el Anejo III.

8.2 Almacenamiento en formato Sparse

Como se ha visto en secciones anteriores el formato de almacenamiento necesario era diferente del utilizado por el programa original y, en consecuencia, se ha

tenido que crear una rutina que se encargue de hacerlo. Para comprobar que efectivamente esa rutina funciona de forma adecuada y sin errores se ha probado con matrices de tamaño pequeño, de tal forma que fuera fácilmente verificable si el resultado final era el esperado o no. Como muestra del correcto funcionamiento del proceso veremos el resultado para una matriz dispersa como la del ejemplo de la sección 7.3.2 *Almacenamiento en Sparse*. El resultado obtenido al realizar el proceso que el citado ejemplo ha sido el mostrado a continuación (Figura 8.2.1).

Como se puede comprobar el resultado obtenido coincide con el esperado completamente, teniendo en cuenta la pequeña diferencia descrita en los párrafos que siguen al ejemplo citado y que se observan en *A%i*. Por lo tanto podemos concluir que esta rutina presenta un funcionamiento adecuado.

8.3 Método de iteración en subespacio

A continuación se mostrarán los resultados obtenidos mediante el método de iteración de subespacios para dos casos concretos y se compararán con las soluciones conocidas de esas estructuras para conocer el nivel de precisión del método.

8.3.1 Ejemplo 1

Como primer ejemplo para comprobar que las rutinas encargadas de calcular las frecuencias y los vectores propios funcionan correctamente se ha escogido uno previamente resuelto [21]. Se trata de una estructura construida de hormigón armado con cerramientos en muros de mampostería. Se ha simplificado la estructura utilizando como modelo un oscilador con una masa concentrada para cada piso. Las matrices de masa y rigidez utilizadas para realizar los cálculos son las que se muestran a continuación:

$$\boldsymbol{M} = 10^4 \begin{bmatrix} 20 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 10 \end{bmatrix} \quad \boldsymbol{K} = 10^7 \begin{bmatrix} 5 & -2 & 0 \\ -2 & 3 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

En este caso al tratarse de un caso tan elemental se ha decidido calcular las tres frecuencias propias, con sus correspondientes formas. Como las frecuencias y las formas están relacionadas, en este apartado se mostrarán solamente los resultados de las frecuencias (Tabla 8.3.1.1) pero en el Anejo IV: Validación SIM se muestra la comprobación completa.

Frecuencias		Error		
Teóricas	Calculadas	Absoluto	Relativo	%
0,942197	0,943542	-0,001344	-0,001427	-0,142670
2,019676	2,017316	0,002360	0,001169	0,116862
2,993704	2,995301	-0,001597	-0,000533	-0,053329

Tabla. 8.3.1.1 Resumen validación del primer ejemplo, teóricas se refiere a los valores proporcionados en el ejemplo de la referencia [21] y loas calculadas son los valores obtenidos al realizar el análisis con los programas explicados.

8.3.2 Ejemplo 2

El segundo caso [22] para comparar las soluciones obtenidas por las rutinas de cálculo es un poco más grande que el primer ejemplo, pero también se trata de hormigón armado con cerramientos en muros de mampostería (Figura 8.3.2.1). Para simplificar los cálculos, se ha considerado la estructura como un oscilador de masas concentradas en cada piso. Los datos (**K** y **M**) usados en este caso son:

$$M = \begin{pmatrix} 105750 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 99750 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 95138 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 93075 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 92325 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 90638 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 88950 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 86888 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 85575 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 63638 \end{pmatrix}$$

$$K = 1E9 \begin{pmatrix} 0.848 & -0.5964 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.5964 & 1.1928 & -0.5964 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.5964 & 0.884 & -0.2876 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.2876 & 0.5752 & -0.2876 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.2876 & 0.5752 & -0.2876 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.2876 & 0.4054 & -0.1178 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.1178 & 0.2356 & -0.1178 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.1178 & 0.1551 & -0.0373 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0373 & 0.0746 & -0.0373 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0373 & 0.0373 \end{pmatrix}$$

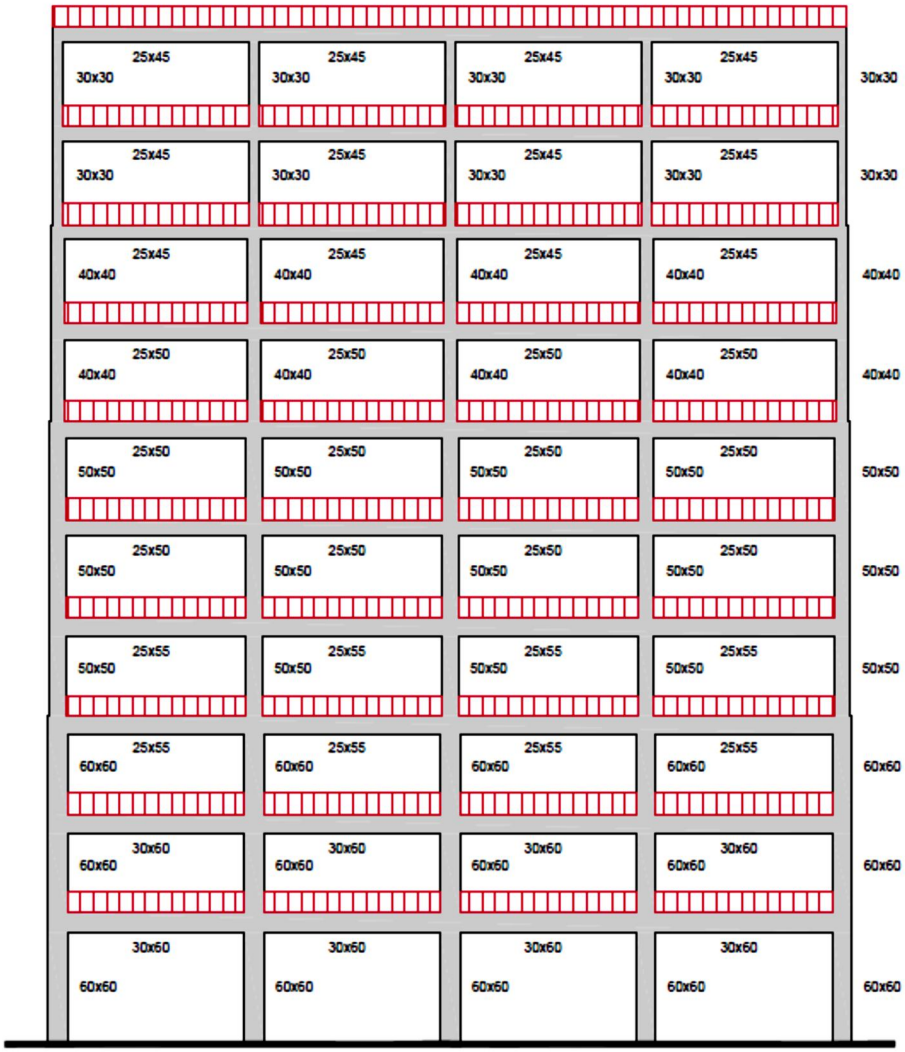


Fig. 8.3.2.1 Estructura del ejemplo 2, imagen extraída de [22].

Se ha decidido calcular las primeras cinco frecuencias y, al igual, que en el primer ejemplo a continuación solamente se muestran éstas (tabla 8.3.2.2) pero en el Anejo IV se pueden consultar todos los resultados obtenidos, incluyendo las formas propias.

Frecuencias		Error		
Teóricas	Calculadas	Absoluto	Relativo	%
1,302739355	1,299508131	-0,003231224	-0,002480330	-0,248033032
2,672665981	2,672236892	-0,000429089	-0,000160547	-0,016054702
4,433543687	4,434784709	0,001241023	0,000279917	0,027991669
5,731788010	5,734211926	0,002423916	0,000422890	0,042288998
6,941053437	6,941458765	0,000405328	0,000058396	0,005839573

Tabla 8.3.2.2 Resumen de los resultados obtenidos para el cálculo de las frecuencias naturales con las matrices del segundo ejemplo, los valores de la columna con el nombre de teóricas se refieren a los valores encontrados en la referencia [22] y utilizados para comprobar los valores calculados con las rutinas explicadas anteriormente.

8.4 Validación daño

Se ha realizado el cálculo del daño medido en fuerzas con las modificaciones en las rutinas y de forma manual para comparar los resultados. Para hacerlo, se ha escogido un problema sencillo, la carga uniaxial de un cubo de material homogéneo, en este caso hormigón..

8.4.1 Cálculo manual

El valor de las rigideces se obtendrá a partir de la información de las fuerzas y los desplazamientos obtenida en cada paso, proporcionada por la rutina del PLCD en uno de los ficheros de salida. Para ello el paso entre incrementos tiene que fijarse de tal forma que al menos el primer paso sea elástico, de otro modo el cálculo no sería correcto, ya que se menospreciaría la rigidez inicial y en consecuencia el daño global producido en cada paso. Todo ello se clarifica en la siguiente figura (fig. 8.4.1.1). Para obtener el daño global utilizaremos la siguiente fórmula:

$$D_{global} = 1 - \frac{F_i \cdot U}{F_e \cdot U} = 1 - \frac{K_I}{K_e} \qquad \text{Ec. 8.4.1.1}$$

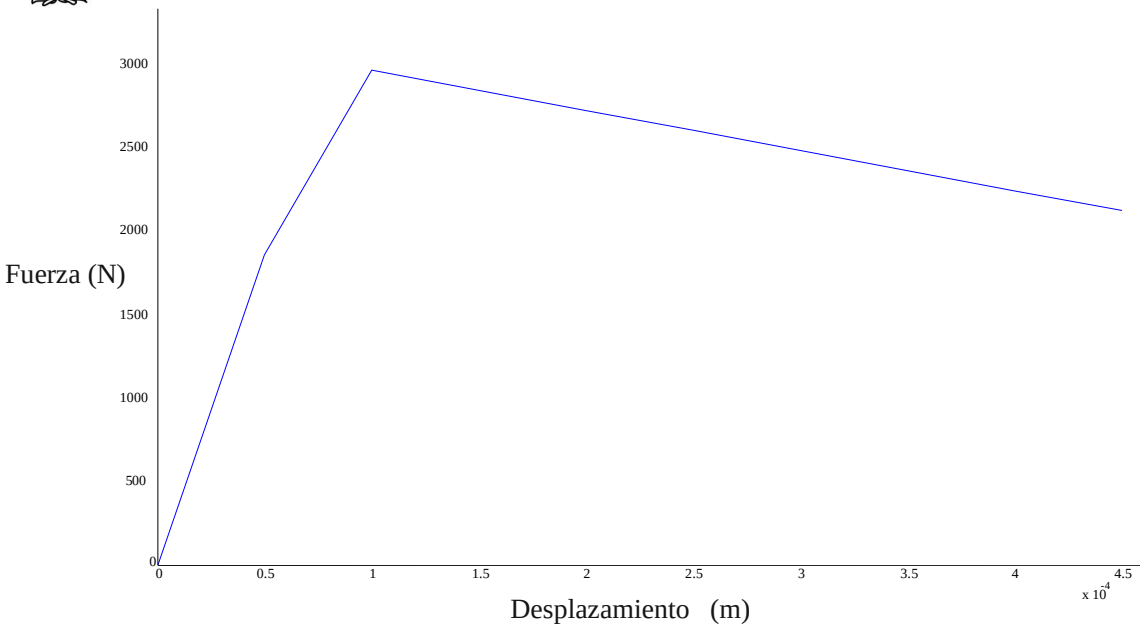


Fig. 8.4.1 Curva del comportamiento real de la estructura del cubo de hormigón utilizado a modo de ejemplo (F_i)

A continuación de muestran los resultados obtenidos para los pasos que han entrado en daño para los desplazamientos fijados:

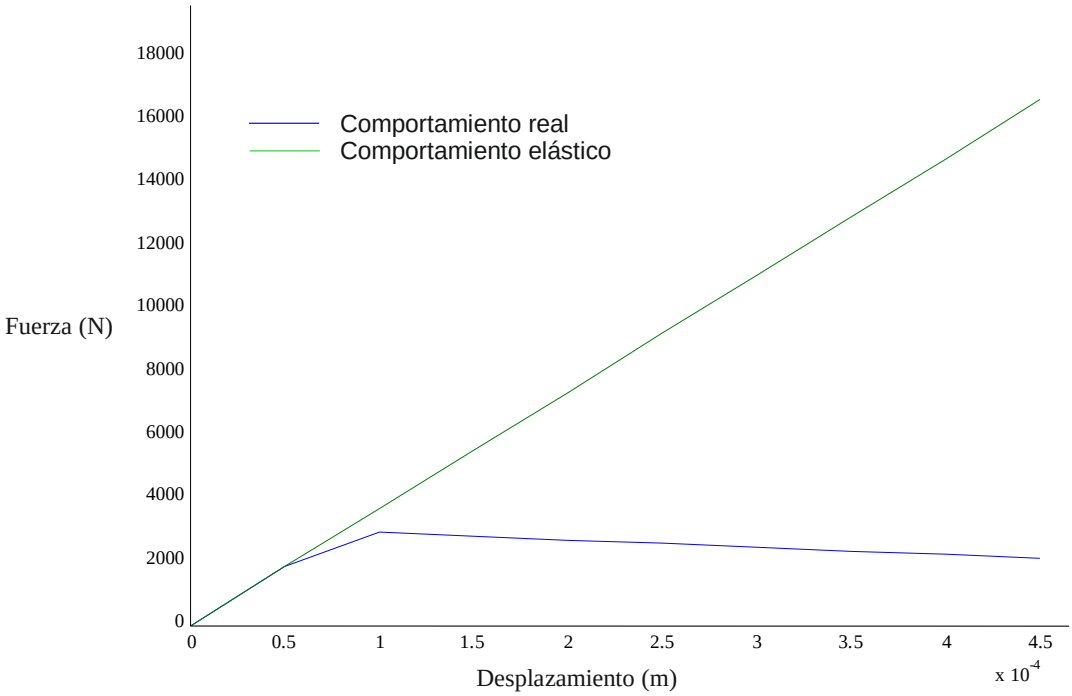


Fig. 8.4.1.2 Comportamiento real (F_i) y el teórico, si se siguiera siempre la teoría de la elasticidad (F_e).

La información presentada en la figura 8.4.1.2 puede expresarse también en forma de tabla (tabla 8.4.1.3). Además se pueden calcular el daño global utilizando la fórmula de la ecuación 8.4.1.1.

Desplazamientos	Fi	Fe	Daño
0,0000	0,0000	0,0000	0,0000
0,0001	1850,0000	1850,0000	0,0000
0,0001	2954,6763	3700,0000	0,2014
0,0002	2834,8921	5550,0000	0,4892
0,0002	2715,1079	7400,0000	0,6331
0,0003	2595,3238	9250,0000	0,7194
0,0003	2475,5396	11100,0000	0,7770
0,0004	2355,7554	12950,0000	0,8181
0,0004	2235,9712	14800,0000	0,8489
0,0005	2116,1870	16650,0000	0,8729

Tabla 8.4.1.3 Puntos calculados para representar las curvas y daño asociado a cada desplazamiento, calculado usando la fórmula de daño de la ecuación 8.4.1.1. Esta tabla se corresponde al ejemplo del cubo de hormigón. Fi representa la fuerza residual y Fe la fuerza elástica.

En el siguiente gráfico (figura 8.4.1.4) se muestra además de la información dada en la figura 8.4.1 la evolución del daño global con el desplazamiento impuesto, es decir, la cuarta columna de la figura 8.4.1.3.

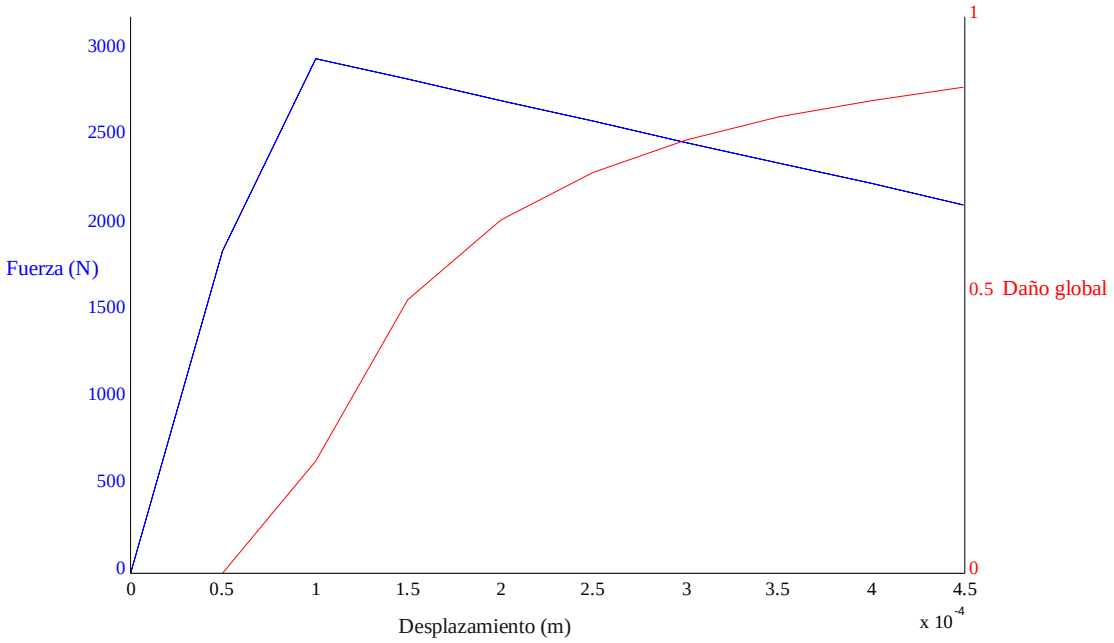


Fig. 8.4.1.4 Representación de la curva real del comportamiento del cubo de hormigón sometido a una fuerza de compresión y la evolución del daño global.

8.4.2 Daño calculado con el PLCD

A la vez que se ha obtenido la curva anterior (figura 8.4.1) que ha permitido calcular el daño global de manera manual, el PLCD también ha calculado el daño utilizando sus rutinas y ha obtenido los siguientes valores (tabla 8.4.2.1).

Desplazamientos	Daño PLCD
0,0001	0,0000
0,0001	0,2014
0,0002	0,4892
0,0002	0,6331
0,0003	0,7194
0,0003	0,7770
0,0004	0,8181
0,0004	0,8489
0,0005	0,8729

Tabla 8.4.2.1 Daño calculado con las nuevas rutinas del PLCD

8.4.3 Comparación

Como se puede observar en la figuras 8.4.1.3 y en la tabla 8.4.2.1 los resultados coinciden para los decimales mostrados, de tal manera que se puede concluir que las modificaciones llevadas a cabo para mejorar el índice de daño global han sido satisfactorias y tienen un funcionamiento correcto.

8.5 Validación completa

A fin de poder establecer si el conjunto formado por los dos programas, PLCD y CFYFP, funcionaba correctamente y realizaba lo que se pretendía, se ha realizado el análisis de una columna no dañada de tres maneras distintas: con las rutinas explicadas en este trabajo, con el software comercial Strand7 y manualmente usando la tabla presentada en el Anejo V.

8.5.1 Problema planteado

Para realizar el siguiente ejemplo se ha utilizado una estructura muy sencilla y, a la vez, con cierto parecido al aerogenerador que será el caso final de estudio. La estructura dimensionada se puede ver en la figura 8.5.1.1, el material utilizado será el hormigón con las características de la tabla 8.5.1.2. Los nodos de la base de la columna tendrán todo tipo de movimiento restringido y el resto son libres.

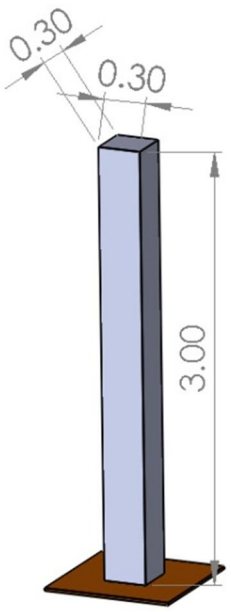


Fig. 8.5.1 Representación de la columna utilizada para realizar la validación, con las dimensiones expresadas en metros.

Criterio de fluencia utilizado	Mohr-Coulomb generalizado (Oller)
Módulo de Young	37000 Mpa
Coeficiente de Poisson	0,2
Densidad del material	2400 Kg/m^3
Tensión umbral	30 Mpa

Tabla 8.5.1.2 Características del hormigón utilizado

8.5.2 Cálculo usando las rutinas implementadas

Para realizar el análisis con las rutinas desarrolladas para este trabajo hay que ejecutar dos programas independientes, primero el PLCD y una vez obtenidos los archivos necesarios, el CFYFP. Para el cálculo se probaron distintas mallas para observar los resultados obtenidos y para determinar la más adecuada, buscando un balance entre un número de elementos excesivo que a pesar de permitir mucha precisión requiere mucho tiempo y un número demasiado bajo que permita un análisis rápido pero poco efectivo. Para determinar el tamaño más adecuado se realizó el siguiente procedimiento: se empezó con una mañana muy sencilla y se fue refinando hasta que entre una malla y su refinada no se observaron variaciones significativas de los resultados obtenidos para el cálculo de frecuencias. A continuación se muestra el proceso seguido (figura 8.5.2.1)

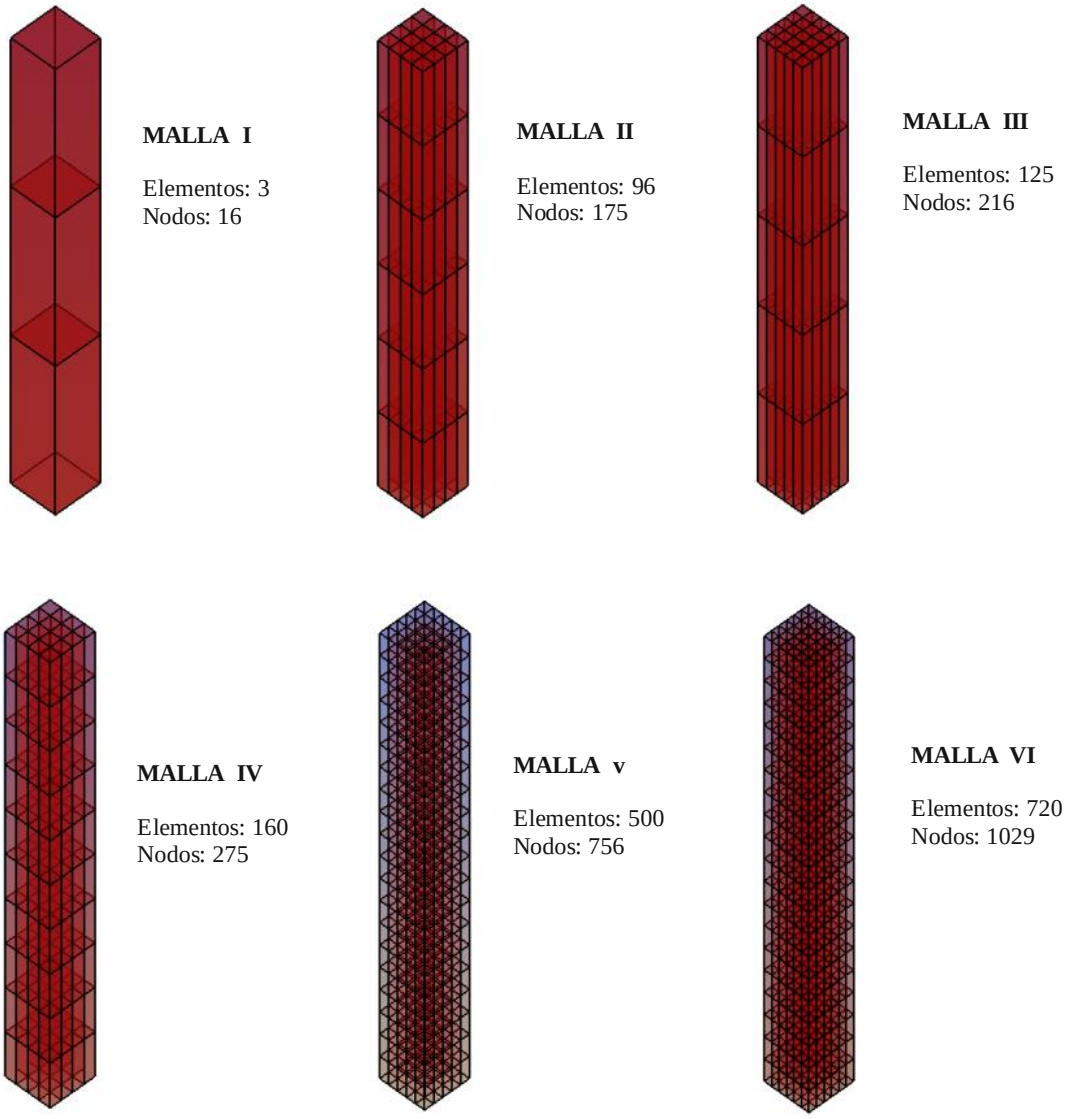


Fig. 8.5.2.1 Proceso de refinado de mallas seguido para encontrar las frecuencias naturales de la estructura.

Como se puede ver en las figuras anteriores, se produce una evolución de los valores (tabla 8.5.2.3) al incrementar el número de elementos hasta llegar a la quinta malla de 500 elementos. Al comparar los resultados entre esta malla y la siguiente utilizada (720 elementos) vemos que las frecuencias calculadas son prácticamente las mismas. Llegado este punto se ha considerado que estos valores que aparecen en cualquiera de las dos últimas mallas (figura 8.5.2.1) son correctos.

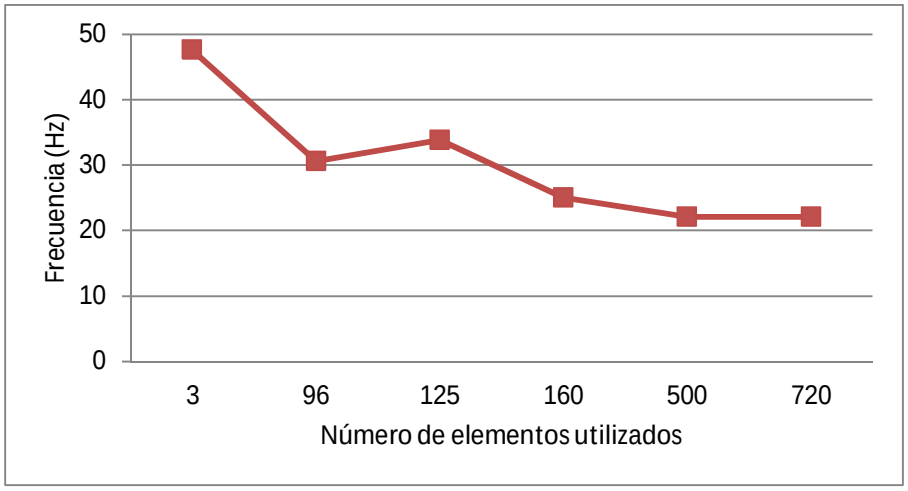


Fig. 8.5.2.2 Evolución de la primera frecuencia calculaba con el número de elementos utilizados

En la siguiente tabla (figura 8.5.2.3) se muestran todos los valores calculados, se puede observar que no existe una tendencia general para los valores, mientras la dos primeras frecuencias tienden a disminuir con las sucesivas mallas, si exceptuamos la malla de 125 elementos, otras como la sexta se mantienen bastante constantes a lo largo de todas los intentos realizados. La misma falta de tendencia clara puede verse en la figura 8.5.2.4 en la que se presentan los mismo valores pero dibujados.

Elementos	1º frecuencia	2º frecuencia	3º frecuencia	4º frecuencia	5º frecuencia	6º frecuencia	7º frecuencia
3	47,59223702	47,59224364	120,5523557	256,4904072	256,4924911	327,3608116	329,3551607
96	30,61233245	30,61233245	179,1815972	179,1815972	188,2642367	328,3869168	463,3116533
125	33,83239461	33,83239465	190,7119611	195,8626975	195,8627044	335,3512568	470,3616583
160	25,01157693	25,01161653	148,9212104	148,931596	188,1801152	328,1816343	390,5071055
500	22,12767656	22,1277096	132,7181103	132,7267516	190,2721118	327,924007	350,0962047
720	22,12056949	22,12058442	132,6800458	132,6837731	191,5439848	327,9087389	349,9825209

Tabla 8.5.2.3 Tabla resumen de los resultados obtenidos para el cálculo de frecuencias utilizando las rutinas del PLCD y del CFYFP.

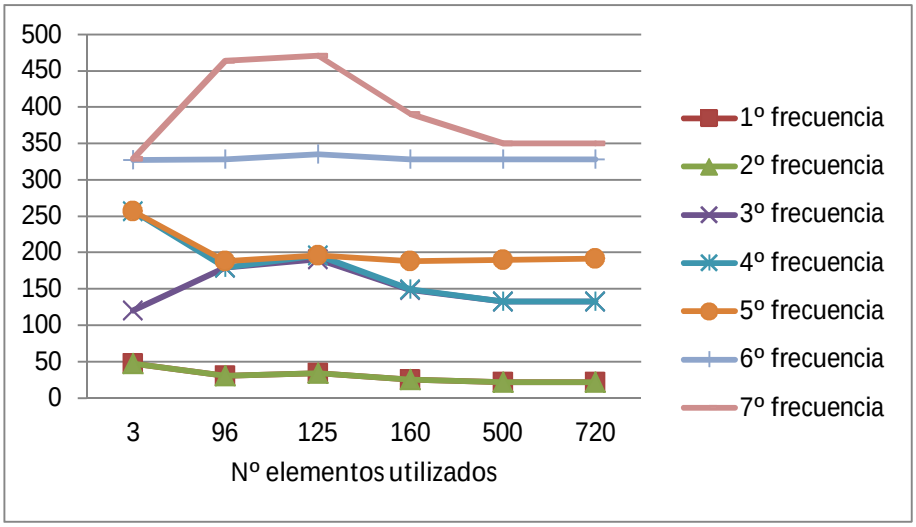


Fig. 8.5.2.4 Evolución de los valores calculados para diferentes mallas.

Para concluir los comentarios sobre estos resultados se muestran en la siguiente figura (8.5.2.5) las diferencias entre el valor calculado para cada malla y el valor "convergido", para el caso de la primera frecuencia.

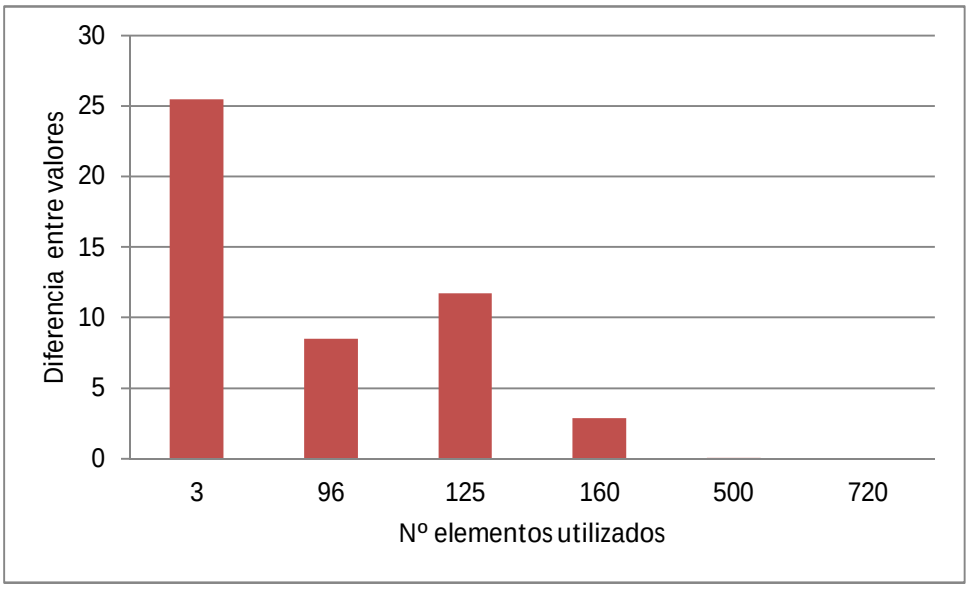


Figura 8.5.2.5 Diferencias observadas entre el valor calculado para la primera frecuencia en cada malla respecto al valor calculado en el último caso.

8.5.3 Cálculo de frecuencias usando *Strand7*

Se ha seguido un proceso similar al mostrado para el cálculo usando las rutinas creadas pero utilizando el programa *Strand7*, también conocido como *Straus7* en algunas zonas. Se ha comenzado con la malla más sencilla posible, de un solo elemento, y se ha ido refinando sucesivamente hasta que los resultados obtenidos para el cálculo de las frecuencias propias han dejado de variar entre dos mallas sucesivas. A continuación se muestran el resumen de los resultados obtenidos (tabla 8.5.3.1).

Elementos	1º Frecuencia	2º Frecuencia	3º Frecuencia	4º Frecuencia	5º Frecuencia	6º Frecuencia	7º Frecuencia
1	16,971646	16,971646	109,784965	298,265797	1794,5087	1917,65268	1917,65268
3	20,4733562	20,4733562	120,552258	124,715883	124,715883	326,877526	329,35489
81	21,0459979	21,0459979	126,471456	126,471456	183,239057	327,920135	336,064167
243	21,0391139	21,0391139	126,453485	126,453485	183,363936	327,691819	334,08607
1944	21,033906	21,033906	126,445777	126,445777	191,340124	327,63555	333,972775

Tabla 8.5.3.1 Resultados obtenidos para las siete primeras frecuencias naturales utilizando diferentes mallas en el *Strand7*.

Como se puede observar en la tabla anterior (fig. 8.5.3.1) las frecuencias más bajas presentan variaciones más pequeñas mientras que las más altas son mucho más grandes. Si se compara con las calculadas con nuestras rutinas (fig. 8.5.2.8) se observan

que el número de elementos tiene menos importancia y que la respuesta es buena con un malla más grande. Esto se puede observar claramente en las figuras 8.5.3.2 y 8.5.3.3, que representan lo mismo que la 8.5.2.9 y 8.5.2.10 y, por tanto, son comparables.

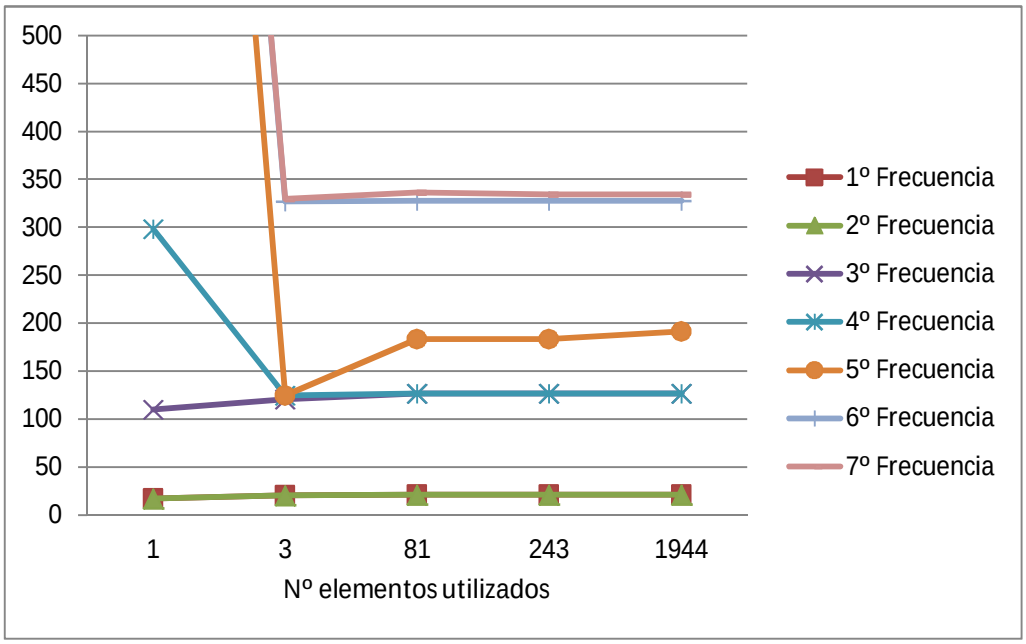


Fig. 8.5.3.2 Evolución de las frecuencias calculadas en función del número de elementos utilizados, se observa que a partir de la malla de 81 elementos las respuestas finales casi no varían.

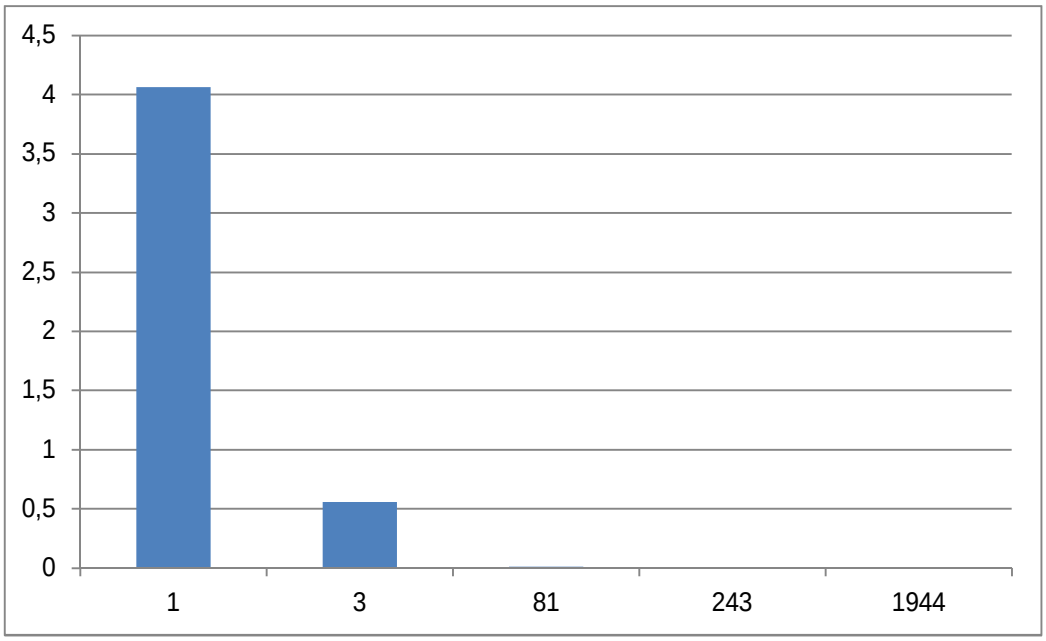


Fig. 8.5.3.3 Diferencias entre el valor de la primera frecuencia de la malla más densa y la malla correspondiente.

8.5.4 Cálculo teórico

A diferencia de los otros dos métodos, el cálculo manual no depende del tipo de malla utilizado. Para realizar los cálculos se ha utilizado la siguiente fórmula (ecuación 8.5.4.1):

$$f_n = \frac{A}{2\pi}\sqrt{\frac{EI}{\rho S l^4}} \qquad \text{Ec. 8.5.4.1}$$

Donde E es el módulo de Young, I es el momento de inercia, l es la longitud de la columna, ρ es la densidad del material, S es el área de la sección y A es un coeficiente que depende de las condiciones de contorno del problema y del modo de vibración buscado. Estas condiciones pueden consultarse en el Anejo V. Los valores que adquieren estas variables para nuestro problema concreto son:

$$E = 3.7 \cdot 10^{10}\,\frac{N}{m^2} \qquad \rho = 2400\,\frac{kg}{m^3}$$

$$S = 0.3 \cdot 0.3 = 0.09\,m^2 \qquad l = 3\,m$$

$$I = \frac{0.3^4}{12} = 6.75 \cdot 10^{-4}\,m^{-4}$$

$$A = 3.52\ para\ las\ dos\ primeras\ frecs.\ y\ 22.4\ para\ las\ dos\ siguientes$$

Sustituyendo estos valores en la fórmula presentada en la ecuación 8.5.4.1 se obtienen los siguientes resultados para las cuatro primeras frecuencias (tabla 8.5.4.1), utilizando esta fórmula no se pueden calcular las formas que involucran torsión.

1º Frecuencia	2º Frecuencia	3º Frecuencia	4º Frecuencia
21,166359	21,166359	134,695012	134,695012

Tabla 8.5.4.1 Resultados obtenidos para el cálculo manual.

8.5.5 Comparación de los resultados obtenidos

Las cuatro primeras frecuencias se han calculado de las tres formas distintas y los resultados obtenidos son comparables. En las siguientes figuras se presentan de forma gráfica éstos.

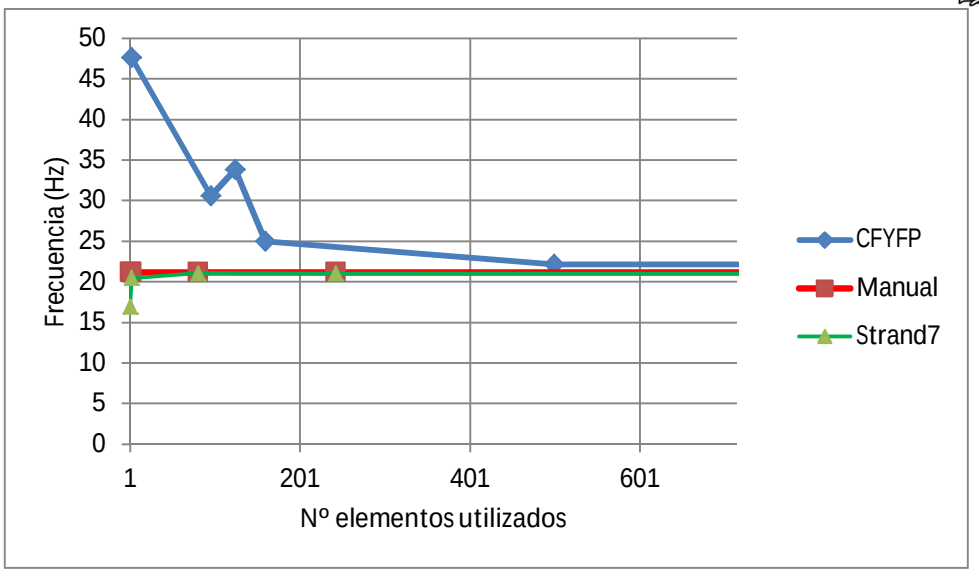


Fig. 8.5.5.1 Comparación resultados para la primera frecuencia

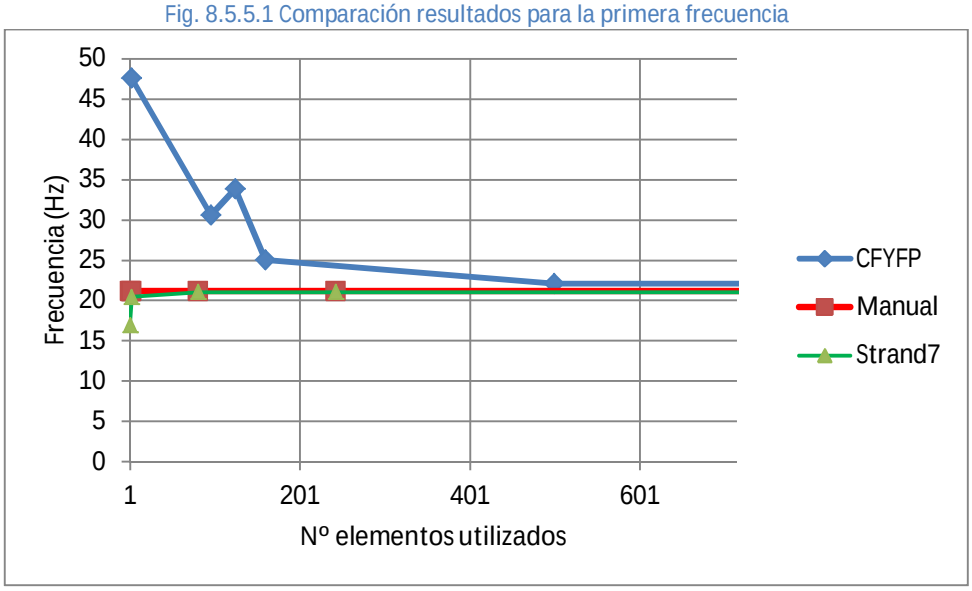


Fig. 8.5.5.2 Resultados para cada método para la segunda frecuencia

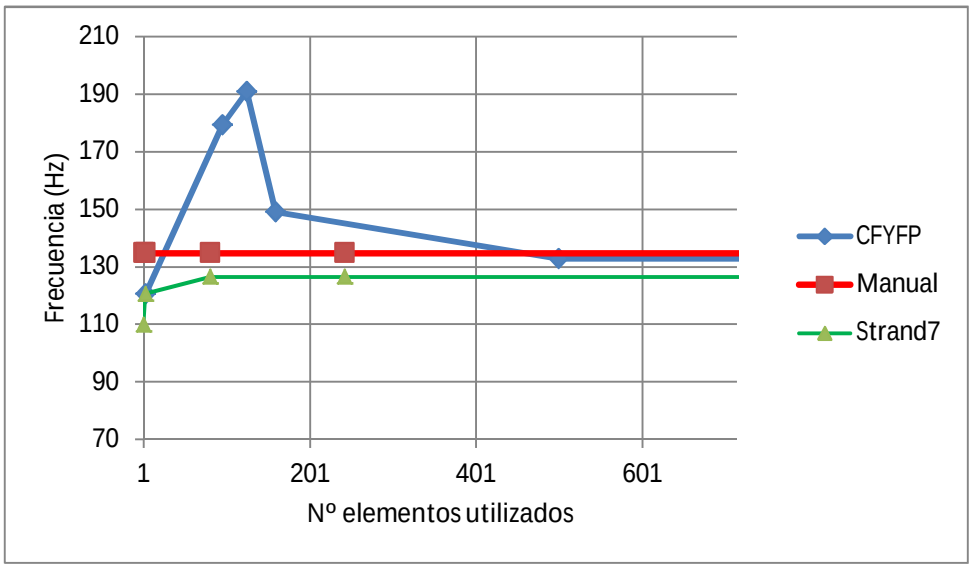


Fig. 8.5.5.3 Comparación de los diferentes métodos para el cálculo de la tercera frecuencia.

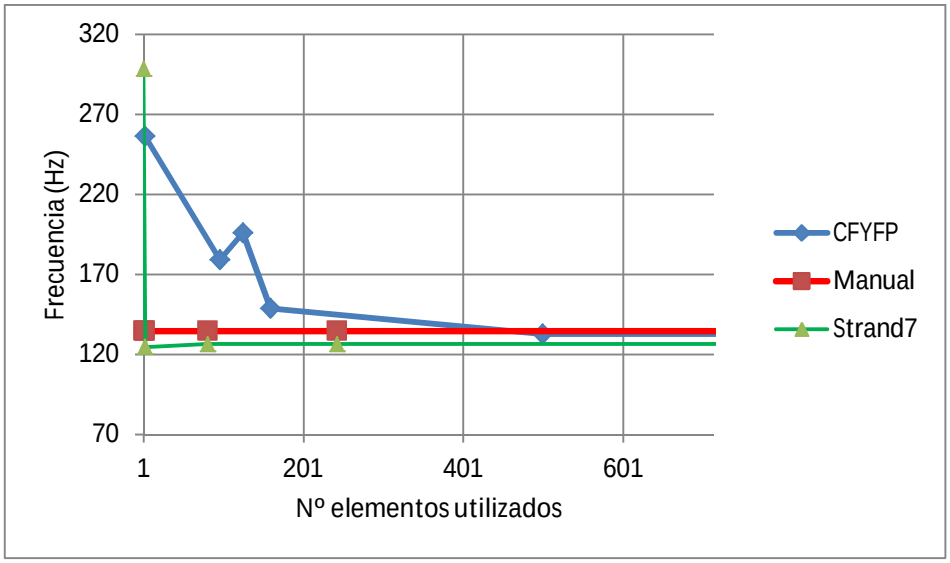


Fig. 8.5.5.4 Resultados para la cuarta frecuencia

Los resultados presentados en las gráficas de la 8.5.5.1 a la 8.5.5.4 se puede observar que a partir de los 500 elementos los resultados de los tres métodos tienen un valor parecido, un valor que puede darse por bueno. De esta forma se puede confirmar que las rutinas implementadas funcionan de forma correcta cuando la malla utilizada es correcta. Respecto a la malla utilizada se ha podido observar que el resultado puede darse por bueno cuando para una discretización y su refinada los valores calculados siguen siendo los mismos. Respecto a la proporción de los hexaedros utilizados, se ha observado que funcionan mejor cuando la relación entre dimensiones es 1:1:1 o cercana, por ejemplo 1:1:2, pero presenta problemas tanto de resultado como de convergencia de las rutinas cuando se aleja mucho 1:1:5 o 1:1:10.

9. Casos de estudio

Durante el desarrollo de las diferentes rutinas se ha trabajado con dos estructuras principalmente, una columna empotrada y una viga apoyada en sus extremos. Para poder obtener conclusiones sobre la evolución de las frecuencias y el daño global y ver si diferentes estructuras presentan comportamientos parecidos se ha decidido utilizar estos dos casos, además del principal caso de estudio, que es la torre del aerogenerador.

9.1 Columna empotrada

Este caso es exactamente el mismo presentado en el apartado 8.5 y utilizado para realizar la validación completa de las rutinas implementadas. Para saber los detalles de la columna de hormigón utilizada puede consultarse el citado apartado. Como se ha visto el aspecto de un elemento de la malla tiene que ser cercano a 1:1:1, para cada una de las direcciones, para obtener buenos resultados. Además en este caso ya se conoce una discretización que permite obtener buenos resultados, al haberse realizado el proceso de refinamiento de la malla con anterioridad.

La malla finalmente utilizada es la última vista en el apartado 8.5, la cual consta de 756 nodos, que forman 500 hexaedros repartidos en 20 niveles a lo largo de su longitud, con 25 elementos por nivel (figura 9.1.1).

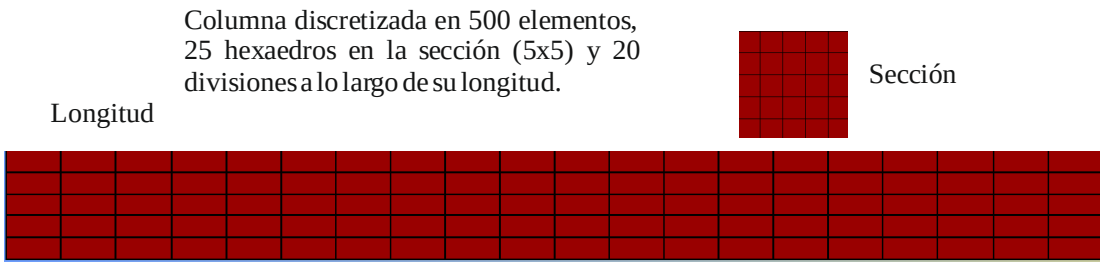


Fig. 9.1.1 Malla utilizada para resolver la columna empotrada.

El primer paso del análisis ha sido la creación del archivo general de entrada .dts. Para ello se han tenido que definir la posición de los nodos y la conectividad de los elementos. En este caso concreto todos los elementos definidos tenían el mismo material. Se han tenido que definir las restricciones nodales e imponer los desplazamientos en los nodos deseados. Por último, han sido definidos los incrementos de desplazamientos a realizar durante todo el proceso. El primer análisis de la estructura se ha realizado con el PLCD para estudiar del comportamiento no lineal de la columna y para obtener los datos y ficheros necesarios para poder utilizar el CFYFP. A continuación se ha creado el .dts necesario para el correcto funcionamiento del programa CFYFP y obtener de esta manera las frecuencias y formas propias deseadas. Se ha decidido calcular las diez primeras frecuencias naturales con sus correspondientes formas propias para los resultados obtenidos del PLCD, además se han fijado el número máximo de iteraciones y la tolerancia de los parámetros calculados. Los resultados del análisis no lineal incluyen 13 pasos dañados, lo que nos da un total de 14 estados de daño diferentes, los trece mencionados más un estado inicial en que la columna no ha sufrido daño alguno todavía. Los pasos dañados extraídos se han concentrado en valores de daño global inferiores al 60%, que ya es considerado un valor de daño global muy elevado, por lo que sacar muchos valores muy elevados tampoco iba a aportar información muy relevante en la práctica. Además como se verá más adelante los resultados del último paso se corresponden a un estado en que la columna ya ha fallado, aunque todavía se obtiene convergencia en el PLCD.

Daño global	1ª Frecuenc ia	2ª Frecuencia	3ª Frecuencia	4ª Frecuencia	5ª Frecuencia	6ª Frecuencia	7ª Frecuencia	8ª Frecuencia	9ª Frecuencia	10ª Frecuencia
0,0000	22,1277	22,1277	132,7121	132,7121	190,2611	327,8772	349,9544	349,9550	570,1968	636,9954
0,0035	22,1198	22,1244	132,6739	132,6968	190,2373	327,8508	349,8754	349,9213	570,1246	636,8872
0,0399	21,9463	22,0529	132,0405	132,4272	189,6290	327,1992	348,8104	349,3721	568,4625	635,4493
0,0914	21,6450	21,9231	131,1715	131,9977	188,4609	325,8841	347,2546	348,4317	565,7306	632,7286
0,1631	21,2247	21,7195	129,9746	131,3252	186,7931	323,7697	345,0235	346,9456	561,9645	628,9230
0,2326	20,7242	21,4644	128,5985	130,5131	184,7246	320,9950	342,4822	345,1218	557,2689	624,4943
0,2915	20,2204	21,2014	127,2630	129,7153	182,4002	318,1403	339,5729	343,2384	552,6337	619,8910
0,3648	19,7240	20,9261	125,9795	128,8573	180,1293	315,0822	336,8002	341,1522	547,7146	614,6108
0,4309	19,2042	20,6219	124,3625	127,7951	177,4598	311,6581	333,4326	338,6767	542,1852	608,9224
0,5137	18,6961	20,3130	122,7916	126,7206	174,8202	308,0871	330,0980	336,0804	536,5187	602,7784
0,5494	18,2205	20,0207	121,5172	125,7855	172,3233	304,6879	327,6436	333,9932	530,8380	598,0190
0,7767	16,3074	18,7687	115,0717	120,9103	161,7078	288,7655	316,8795	322,9524	505,1483	575,7098
0,9001	13,9481	17,1911	107,6433	114,3875	148,5579	266,4227	309,4298	309,7684	468,6465	554,6368
0,9384	0,0000	17,1911	107,6433	114,3875	148,5579	266,4227	309,4298	309,7684	468,6465	554,6368

Tabla 9.1.2a Resultados obtenidos para las diez primeras frecuencias naturales de la columna.

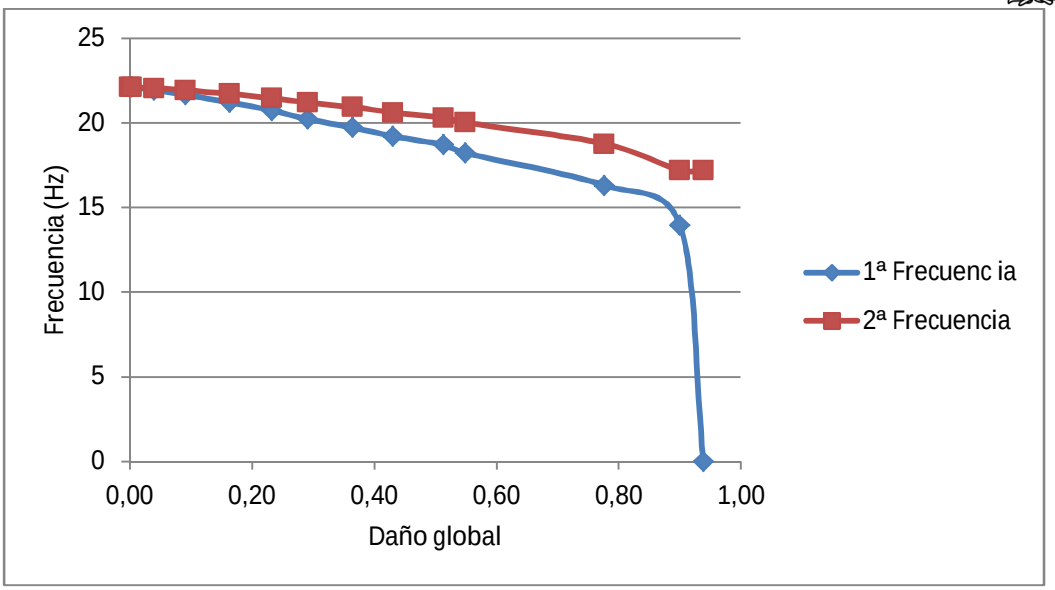


Fig. 9.1.2b Representación de la evolución de la primera y segunda frecuencias al dañarse la columna

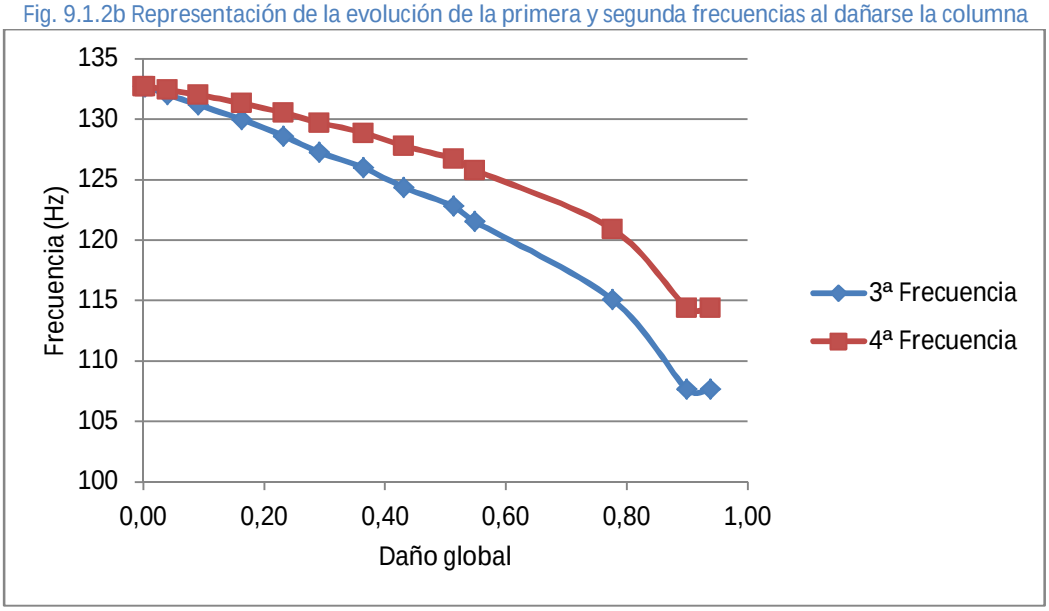


Fig. 9.1.2c Representación gráfica de la evolución de la tercera y cuarta frecuencia.

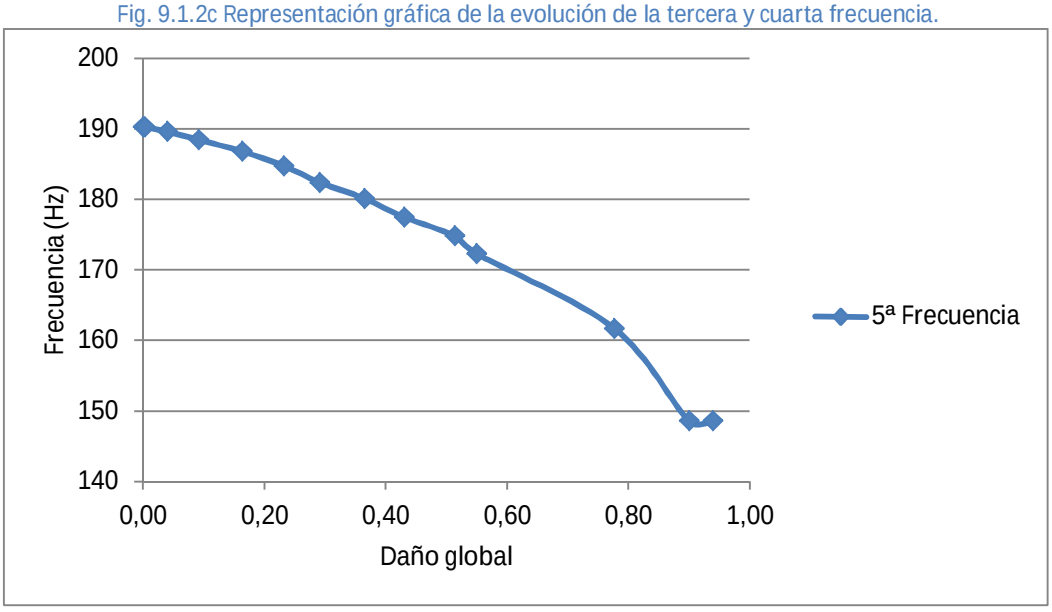


Fig. 9.1.2d Representación gráfica de la evolución de la quinta frecuencia al dañarse la columna.

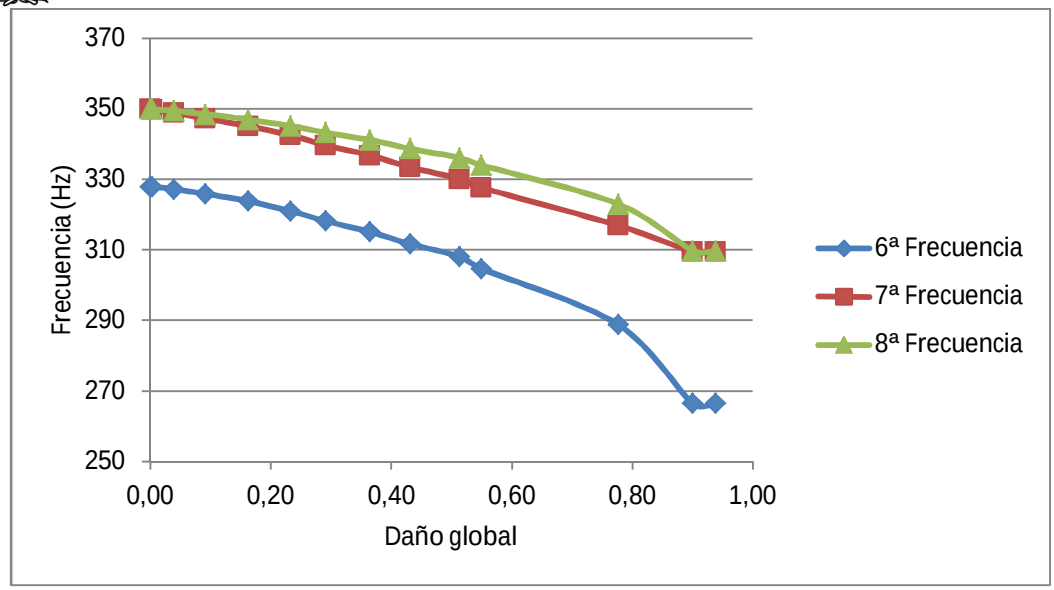


Fig. 9.1.2e Evolución de la sexta, séptima y octava frecuencia a medida que se va dañando la estructura.

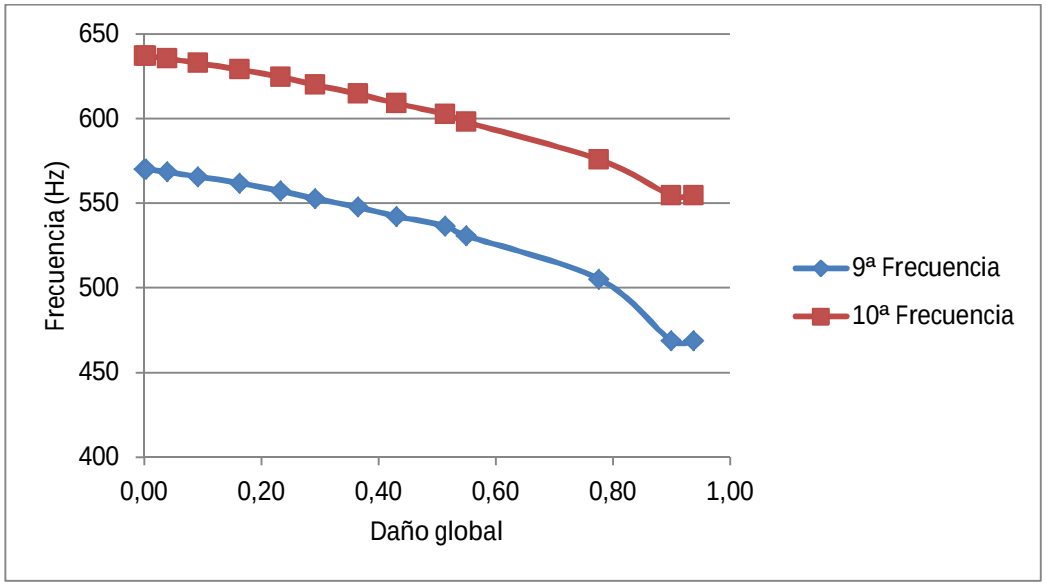


Fig. 9.1.2f Representación de la evolución a medida que se dañan de la novena y décima frecuencia.

En la tabla 9.1.2 y las figuras 9.1.2(b-f) se observa una tendencia clara en todas las frecuencias calculadas, el valor va disminuyendo a medida que aumenta el daño, pero no de forma lineal ni igual en todos ellos. Este hecho se puede ver claramente en aquellas frecuencias que inicialmente tienen un valor idéntico, como pueden ser la primera y la segunda o la tercera y la cuarta, como a medida que se daña la estructura evolucionan de manera distinta hasta llegar a valores claramente diferenciados. De todas maneras este aspecto no parece ser una norma general, ya que en el caso de la séptima y la octava frecuencia mantienen valores muy similares para los distintos pasos dañados calculados. La explicación está relacionada con el lugar en el que se produce el daño en

la estructura (figura 9.1.3). Para acabarlo de clarificar, esta última figura se complementa con la siguiente (figura 9.1.4) que muestra las formas propias de la primera y segunda frecuencia calculadas vistas desde el mismo punto de vista que la representación del daño. Como se puede observar los dos son un mismo movimiento pero cada uno en una dirección perpendicular al otro, esto provoca que cuando se daña la estructura de forma no homogénea, se vean afectadas de manera diferente. Al ser la primera forma en la dirección que daña la estructura se ve mucho más afectada que la segunda que el movimiento es prácticamente perpendicular a la zona con mayor daño. En el caso en el que la zona dañada afecta en menor medida a las formas deformadas, las frecuencias se ven menos afectadas. En el caso de la tercera y cuarta frecuencia, que son un mismo movimiento en direcciones perpendiculares, como la primera y la segunda, funcionan de manera más similar entre ellas a medida que se dañan debido a la forma propia que tienen, que se ve menos afectada por el daño producido en este caso en la columna. Un poco más adelante se discutirán los cambios sufridos por las formas propias durante el proceso de daño.

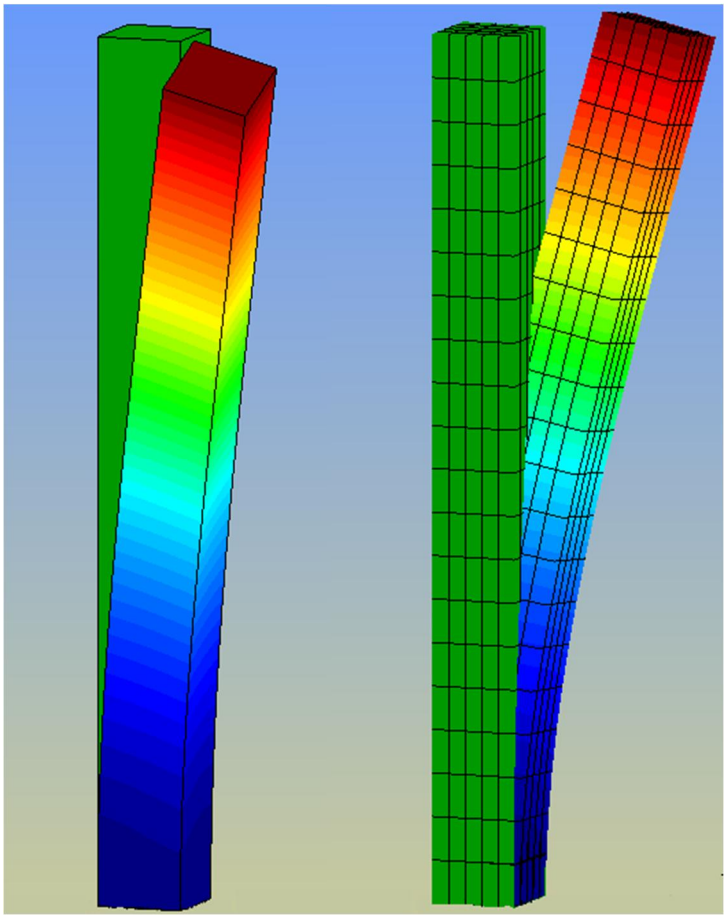


Fig. 9.1.4 Columna original y forma deformada para la primera (izquierda) y segunda frecuencia (derecha)

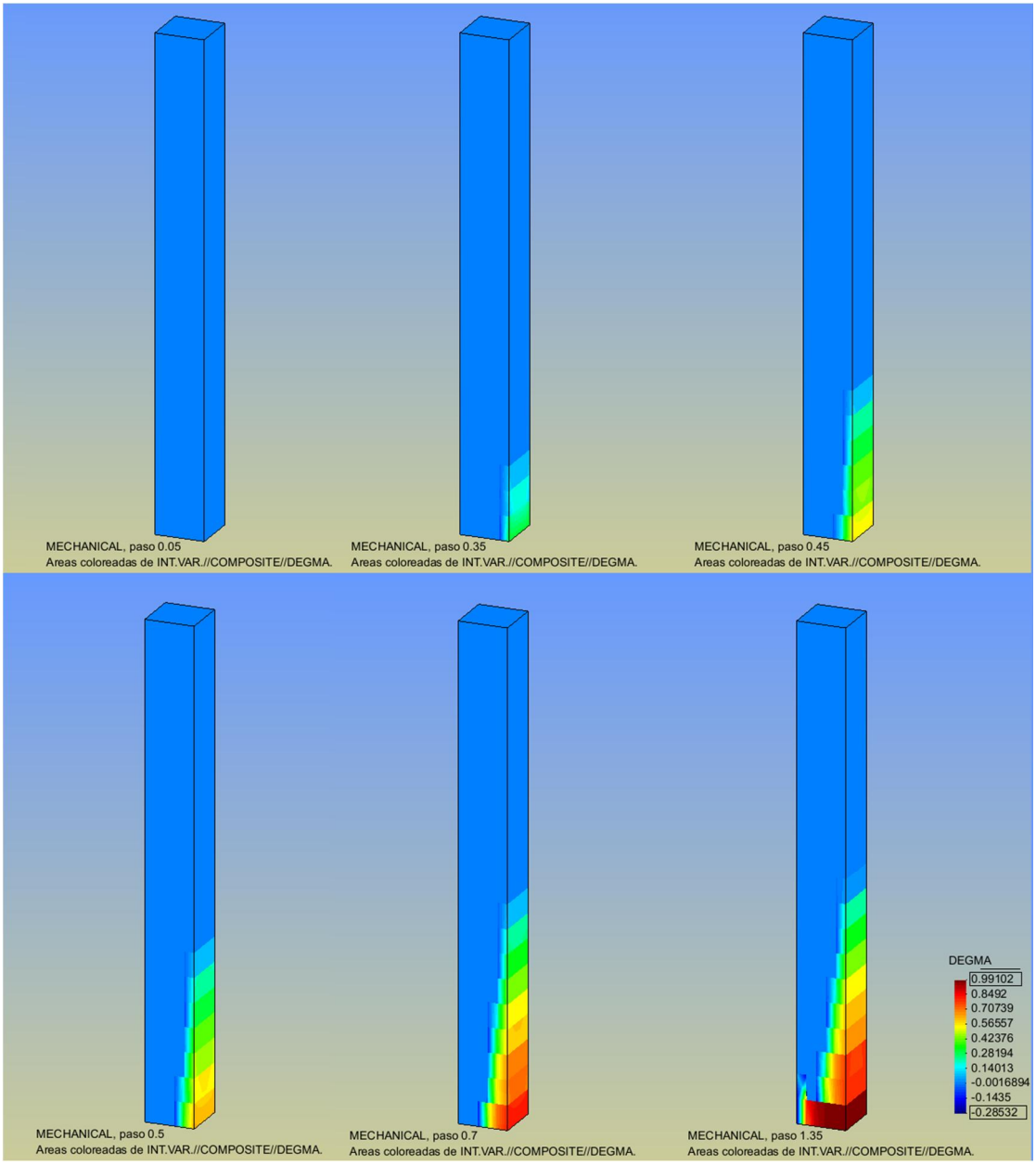


Fig. 9.1.3 Localización y evolución del daño local en la columna.

Tal y como se ha visto en la figura 9.1.2(a-f) el valor de las frecuencias disminuye y en términos de valores absolutos, el descenso es mayor cuanto mayor es la frecuencia. Sin embargo es interesante observar como son estos descensos en términos de valores relativos, esto es precisamente lo que se pretende mostrar a continuación en las figuras 9.1.5 y 9.1.6.

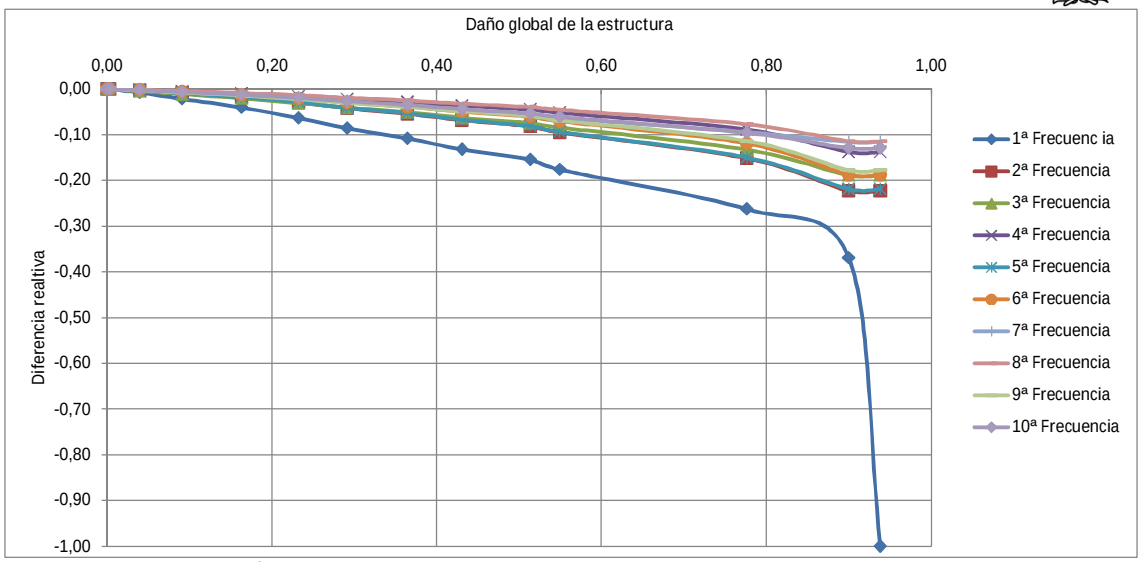


Fig. 9.1.5 Evolución de la diferencia relativa entre frecuencias a medida que la estructura se daña.

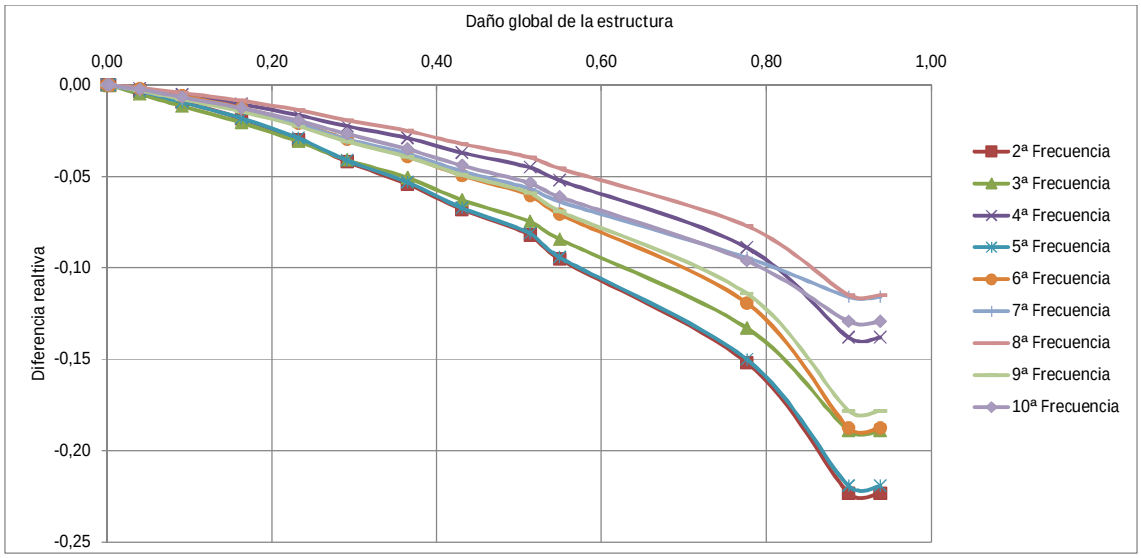


Fig. 9.1.6 Detalle de la figura 9.1.5, no se ha representado la primera frecuencia

Si exceptuamos el caso de la primera frecuencia que como se puede observar en la figura 9.1.5 es totalmente diferente al resto, se puede observar que en general todas las frecuencias presentan unos comportamientos bastante similares. En el caso de daño máximo las diferencias entre las diferencias relativas entre todas las frecuencias son inferiores a un 10%. Pero para un caso de un daño global cercano a 0.5 las frecuencias son aproximadamente un 6% menores que las originales (±2% para los valores extremos). Todo esto se puede observar en la figura 9.1.7, como se ha visto que la primera frecuencia tiene un comportamiento diferente del resto se incluyen los cálculos utilizando las diez frecuencias y también excluyendo la primera. Se puede ver que para valores de daño no muy elevados el promedio no se aleja mucho de los máximos y

mínimos de ese nivel, tal y como se ha visto gráficamente en la tabla 9.1.6 en la que las diferentes frecuencias estaban agrupadas.

	Todas	Sin la primera		
Daño global	Promedio	Promedio	Máximo	Mínimo
0,0000	0,0000	0,0000	0,0000	0,0000
0,0035	0,0173	0,0137	0,0288	0,0081
0,0399	0,3458	0,2638	0,5060	0,1666
0,0914	0,9019	0,6838	1,1608	0,4353
0,1631	1,7088	1,3008	2,0627	0,8599
0,2326	2,6852	2,0509	3,0996	1,3811
0,2915	3,6922	2,8303	4,1860	1,9193
0,3648	4,7229	3,6367	5,4302	2,5154
0,4309	5,8952	4,5740	6,8050	3,2228
0,5137	7,0767	5,5259	8,2006	3,9647
0,5494	8,1292	6,3635	9,5218	4,5611
0,7767	12,8800	10,2498	15,1800	7,7160
0,9001	18,6437	14,9471	22,3096	11,4834
0,9384	24,9471	14,9471	22,3096	11,4834

Tabla 9.1.7 La tabla anterior contiene en la primera columna el daño global, en la segunda el promedio de las diferencias relativas de todas las frecuencias respecto a la no dañada para ese nivel de daño. La tercera es igual que la segunda pero sin tener en cuenta la primera frecuencia. La cuarta y la quinta son, respectivamente, la diferencia porcentual máxima y mínima para ese nivel de daño.

Que las diferencias relativas entre una frecuencia dañada y la frecuencia inicial sean parecidas para todas las frecuencias, permite poder establecer una relación entre la frecuencia en un instante y el daño global de la estructura. Para obtener el daño global de la estructura en base a las diferencias relativas entre frecuencias, se propone lo siguiente: se tiene que realizar el promedio de: $\frac{f_i - f_o}{f_0} \cdot \frac{1}{0.12}$ para cada frecuencia dañada f_i para un mismo nivel de daño. En la figura 9.1.8b se puede ver el área cubierta con esta aproximación. En la tabla 9.1.8a se pueden ver los resultados obtenidos con este método y el error producido. Se puede observar que los errores son pequeños hasta que se llega al punto con daño 0.5494, a partir de este punto se van alejando y deja de poder ser útil. El cálculo se ha hecho con 0.12 que es valor intermedio pero el área rayada de la figura 9.1.8b es la que comprende entre 0.08 y 0.16.

Daño global	1ª Frec.	2ª Frec.	3ª Frec.	4ª Frec.	5ª Frec.	6ª Frec.	7ª Frec.	8ª Frec.	9ª Frec.	10ª Frec.	Calculado	Error absoluto
0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0035	0,0030	0,0012	0,0024	0,0010	0,0010	0,0007	0,0019	0,0008	0,0011	0,0014	0,0014	0,0020
0,0399	0,0683	0,0282	0,0422	0,0179	0,0277	0,0172	0,0272	0,0139	0,0253	0,0202	0,0288	0,0111
0,0914	0,1817	0,0770	0,0967	0,0449	0,0788	0,0507	0,0643	0,0363	0,0653	0,0558	0,0752	0,0162
0,1631	0,3401	0,1537	0,1719	0,0871	0,1519	0,1044	0,1174	0,0717	0,1203	0,1056	0,1424	0,0207
0,2326	0,5286	0,2498	0,2583	0,1381	0,2425	0,1749	0,1779	0,1151	0,1889	0,1635	0,2238	0,0088
0,2915	0,7183	0,3488	0,3422	0,1882	0,3443	0,2475	0,2472	0,1599	0,2567	0,2238	0,3077	-0,0162
0,3648	0,9052	0,4525	0,4228	0,2421	0,4438	0,3252	0,3132	0,2096	0,3286	0,2928	0,3936	-0,0287
0,4309	1,1010	0,5671	0,5243	0,3088	0,5607	0,4122	0,3934	0,2686	0,4094	0,3673	0,4913	-0,0603
0,5137	1,2923	0,6834	0,6229	0,3762	0,6763	0,5030	0,4728	0,3304	0,4922	0,4476	0,5897	-0,0760
0,5494	1,4714	0,7935	0,7030	0,4349	0,7857	0,5894	0,5313	0,3801	0,5752	0,5099	0,6774	-0,1280
0,7767	2,1919	1,2650	1,1077	0,7411	1,2506	0,9941	0,7876	0,6430	0,9507	0,8018	1,0733	-0,2967
0,9001	3,0805	1,8591	1,5741	1,1507	1,8266	1,5619	0,9650	0,9569	1,4841	1,0774	1,5536	-0,6535
0,9384	8,3333	1,8591	1,5741	1,1507	1,8266	1,5619	0,9650	0,9569	1,4841	1,0774	2,0789	-1,1405

Tabla 9.1.8a Daño global calculado usando las frecuencias y el error respecto al verdadero.

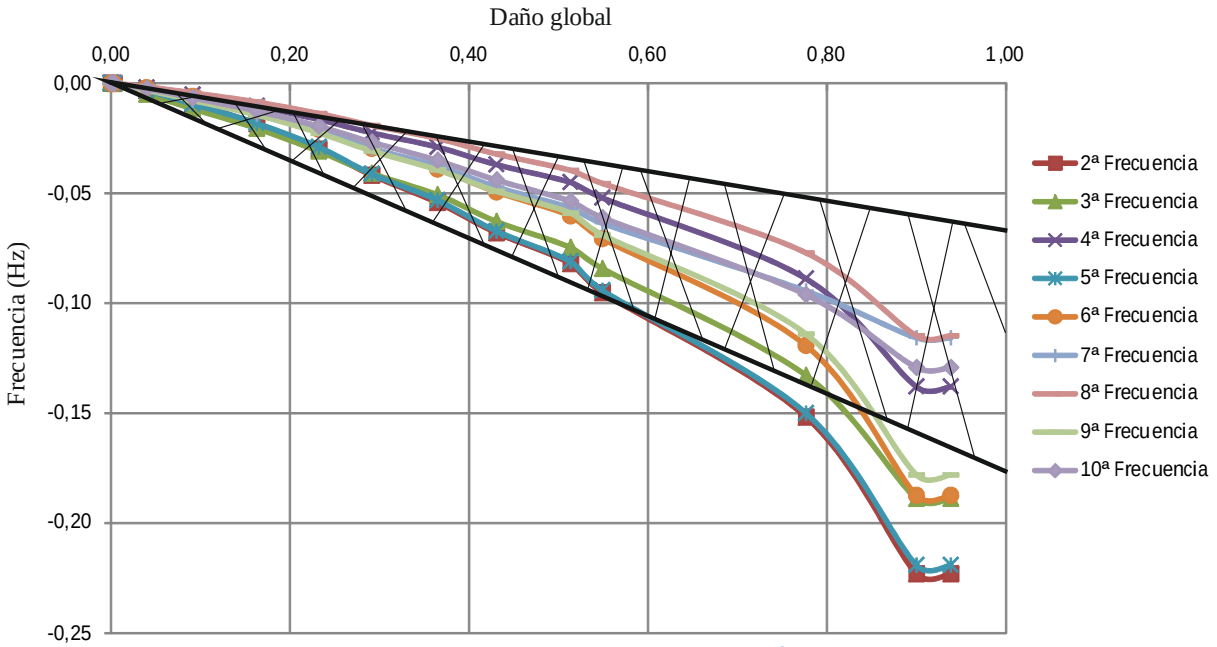


Fig. 9.1.8b La zona rayada representa los valores cubiertos con la aproximación explicada. La recta superior tiene valor de 0.08 y la inferior 0.16. Está aproximación es útil hasta valores de daño global cercanos al 60% pero luego deja de resultar interesante usarla.

En la figura 9.1.9 la curva del comportamiento de la estructura y la evolución del daño global a medida que se imponen unos desplazamientos. En el momento en que aparece el daño la curva deja de ser una recta y empieza a curvarse. En la curva del daño se observa en medio un comportamiento extraño debido a que no hay muchos puntos en esa zona, pero la parte que nos interesará será la parte inferior.

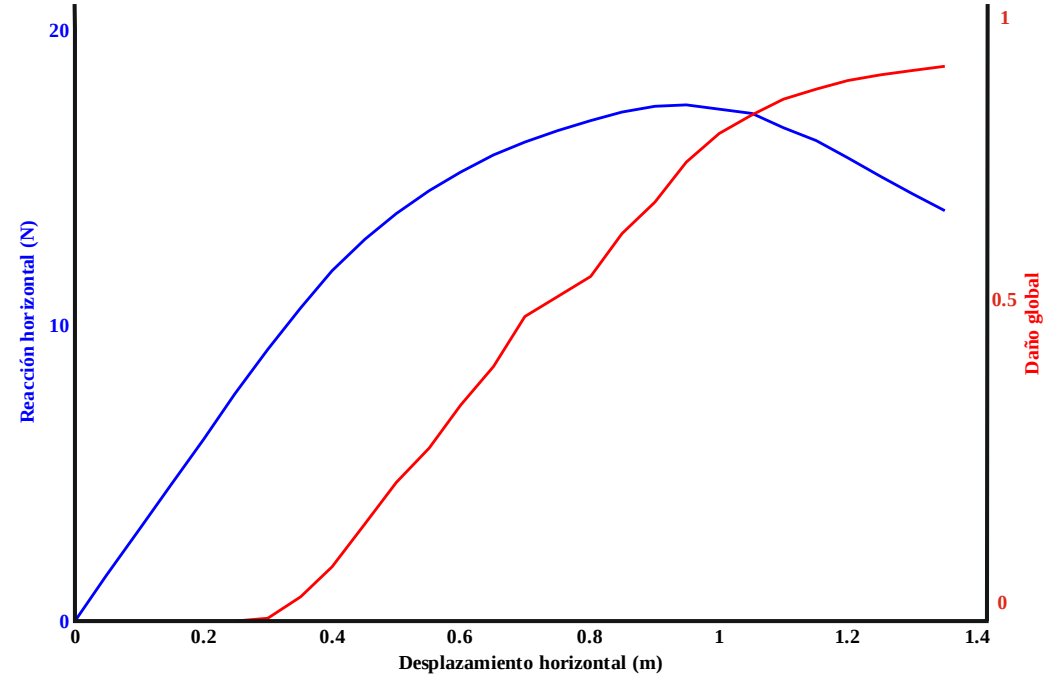


Fig. 9.1.9 Curva del comportamiento de la columna y daño global de la estructura.

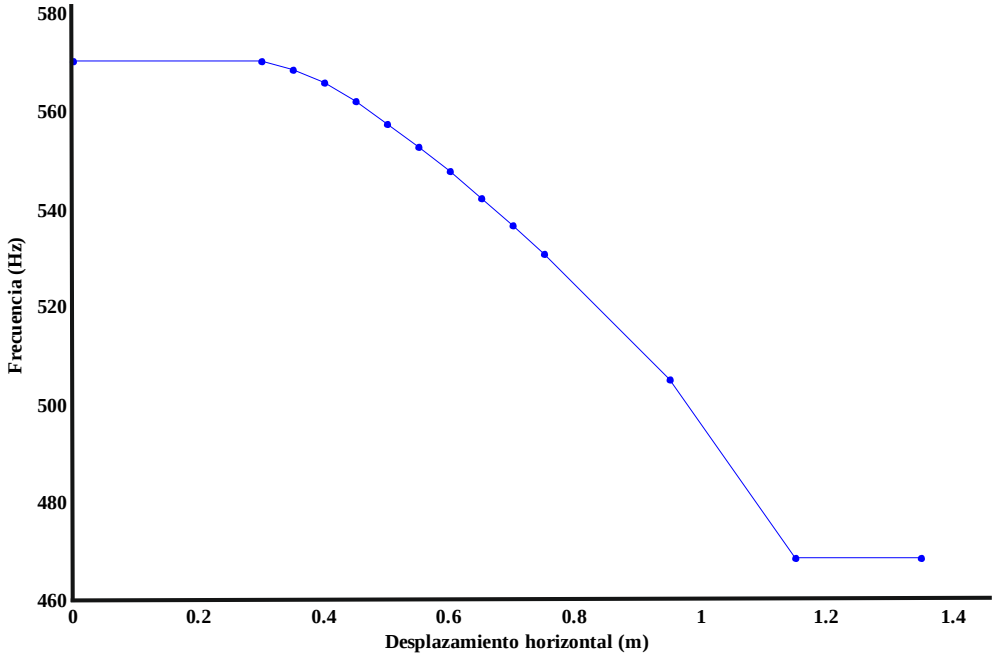


Fig. 9.1.10 Evolución del valor de una frecuencia al aumentar el desplazamiento horizontal de la columna. En este caso se ha representado la novena frecuencia pero todas presentan el mismo tipo de gráfica, a excepción de la primera porque al final se hace cero.

El comportamiento observado en la figura 9.1.10 se repite para todas las frecuencias calculadas con distintos valores. Como se puede ver la figura contiene más puntos calculados al inicio que al final y, por eso, la curva queda mejor definida en esta zona. Si se dibuja el daño en la misma gráfica (figura 9.1.11) se observa que para

valores de daño muy elevados el valor de las frecuencias tiende a estabilizarse y no continúan descendiendo hasta cero.

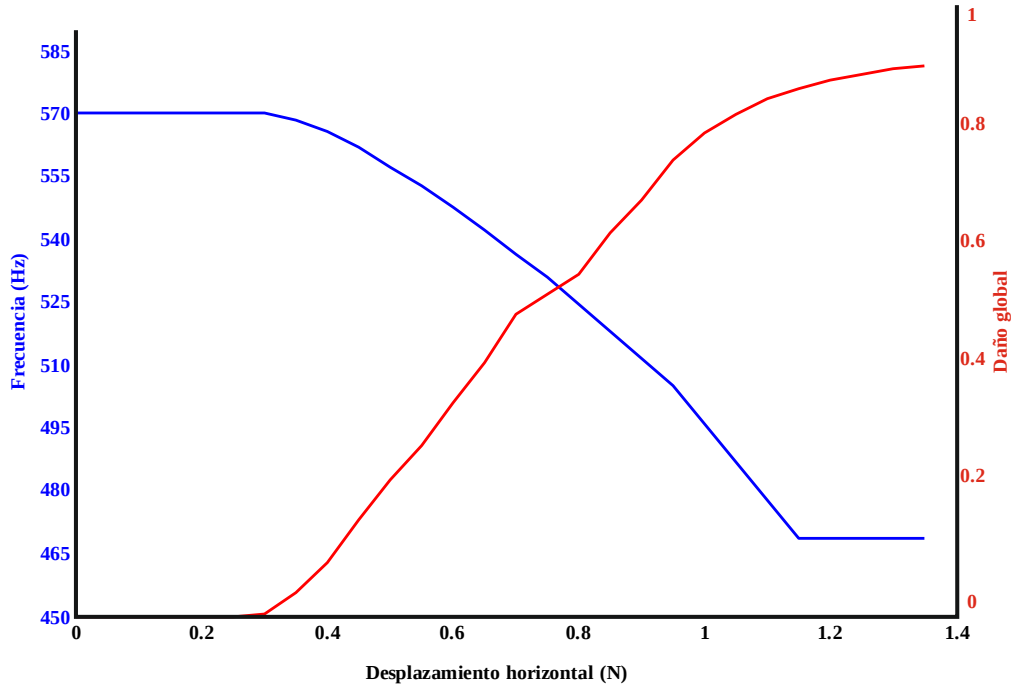


Fig. 9.1.11 Representación de la curva de daño y evolución de frecuencias a medida que se aumenta el desplazamiento horizontal.

En cuanto a las evoluciones de las formas propias no ha podido establecerse ninguna correlación clara o una tendencia que siguieran todas o la mayoría de ellas. En algunos casos las formas se mantienen en la forma casi invariables pero si cambian ligeramente la orientación del movimiento. En las frecuencias que involucran movimiento torsivos, se observa una disminución de la torsión al dañarse. En algún caso como el de la sexta frecuencia, se ha observado un cambio completo del tipo de movimiento producido por la frecuencia. Pasando de ser inicialmente un movimiento de subida y bajada a ser un movimiento ondulatorio similar a formas propias asociadas a otras frecuencias, como por ejemplo la quinta. Una posible explicación a las reorientaciones que sufren las formas de vibración tenga que ver con la localización del daño en la estructura y que una forma que inicialmente esté asociada a una frecuencia determinada pueda pasar con el daño a asociarse a otra frecuencia distinta. Este hecho se ha observado en aquellas frecuencias que tenían inicialmente el mismo valor (por ejemplo la primera y la segunda), cuyas formas propias han ido girando ligeramente a medida que se dañaba la columna hasta tener la primera frecuencia una orientación similar a la segunda y la segunda una similar a la primera. Todo ello se puede observar

en las figuras que siguen a continuación, en las que la columna verde representa la columna original sin daño ni movimiento alguno.

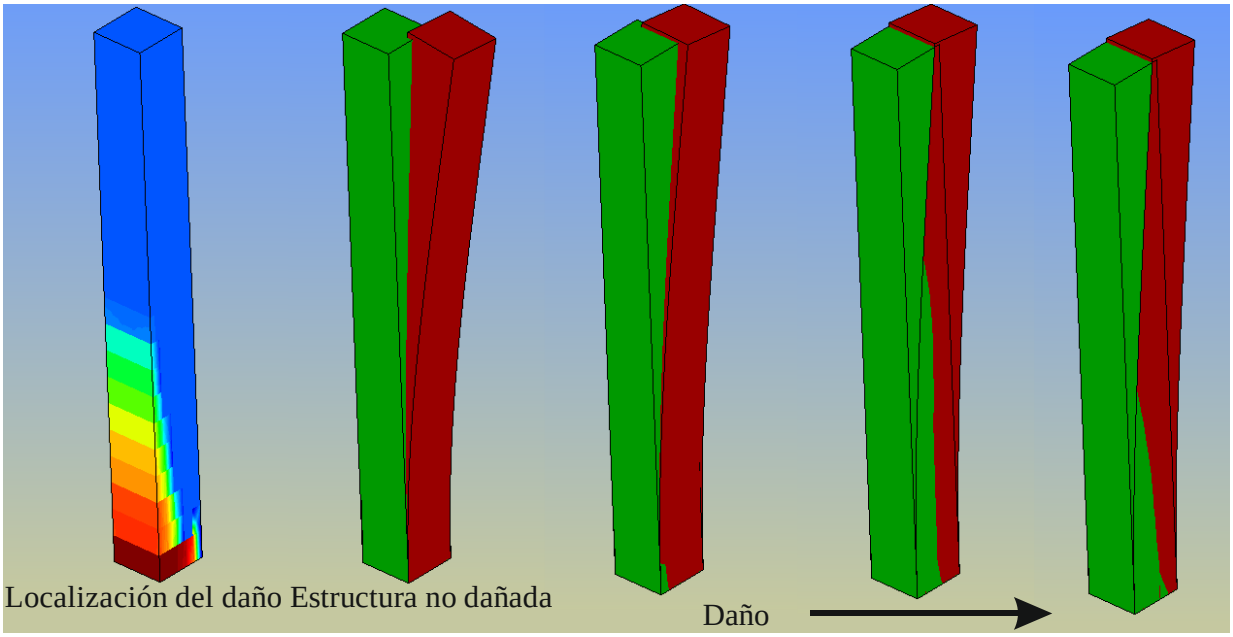


Fig. 9.1.12 Evolución de la primera forma propia, vemos como movimiento pasa de estar oblicuo a situarse paralelo a la zona dañada, y como a medida que se daña disminuye el desplazamiento máximo.

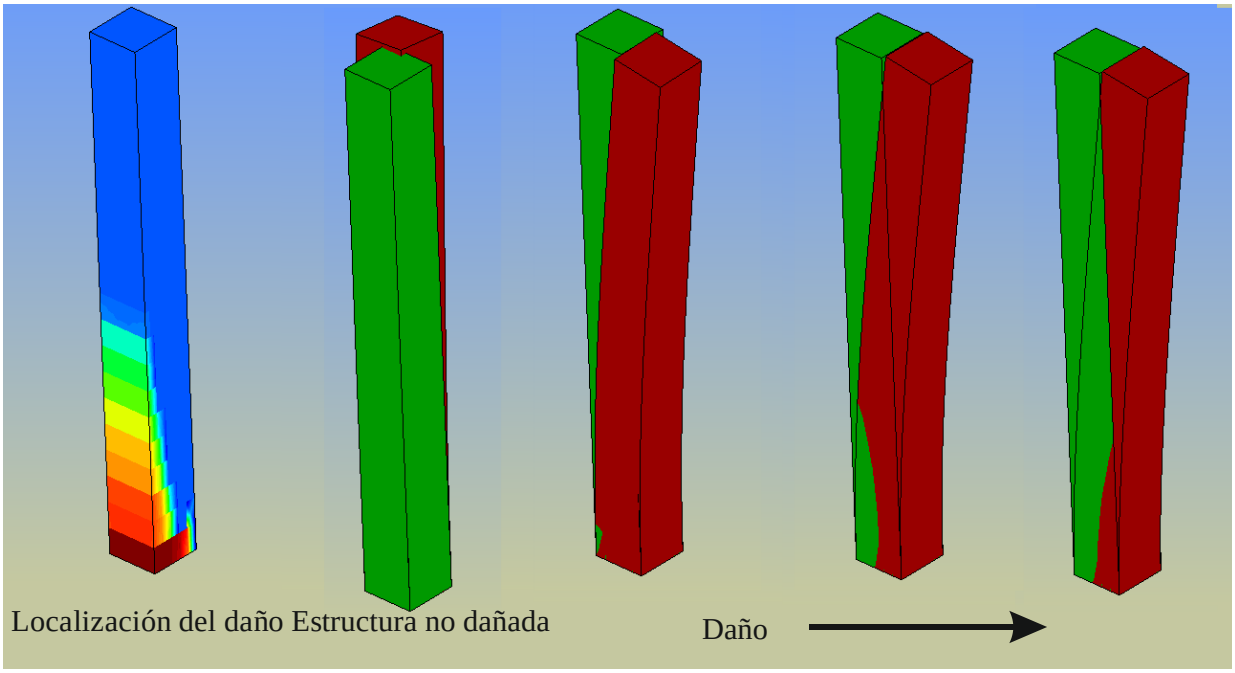


Fig. 9.1.13 Vemos que tiene un comportamiento similar a la primera frecuencia, se produce un giro, hasta se perpendicular al daño en este caso, y disminuye la amplitud del movimiento de forma ligera.

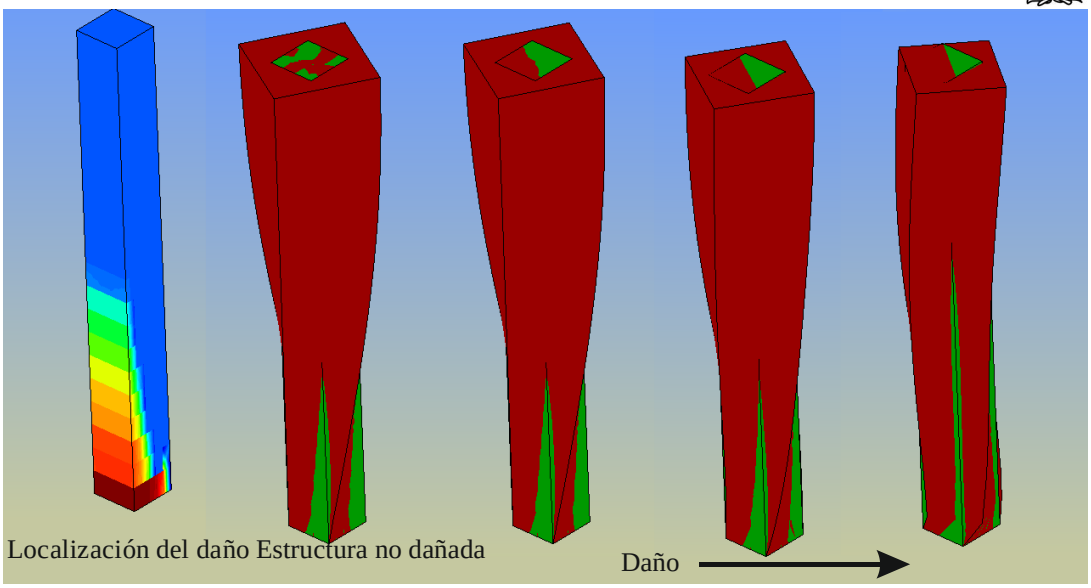


Fig. 9.1.14 Evolución de la quinta forma propia, se trata de un movimiento de torsión, se puede observar que es cada vez menor.

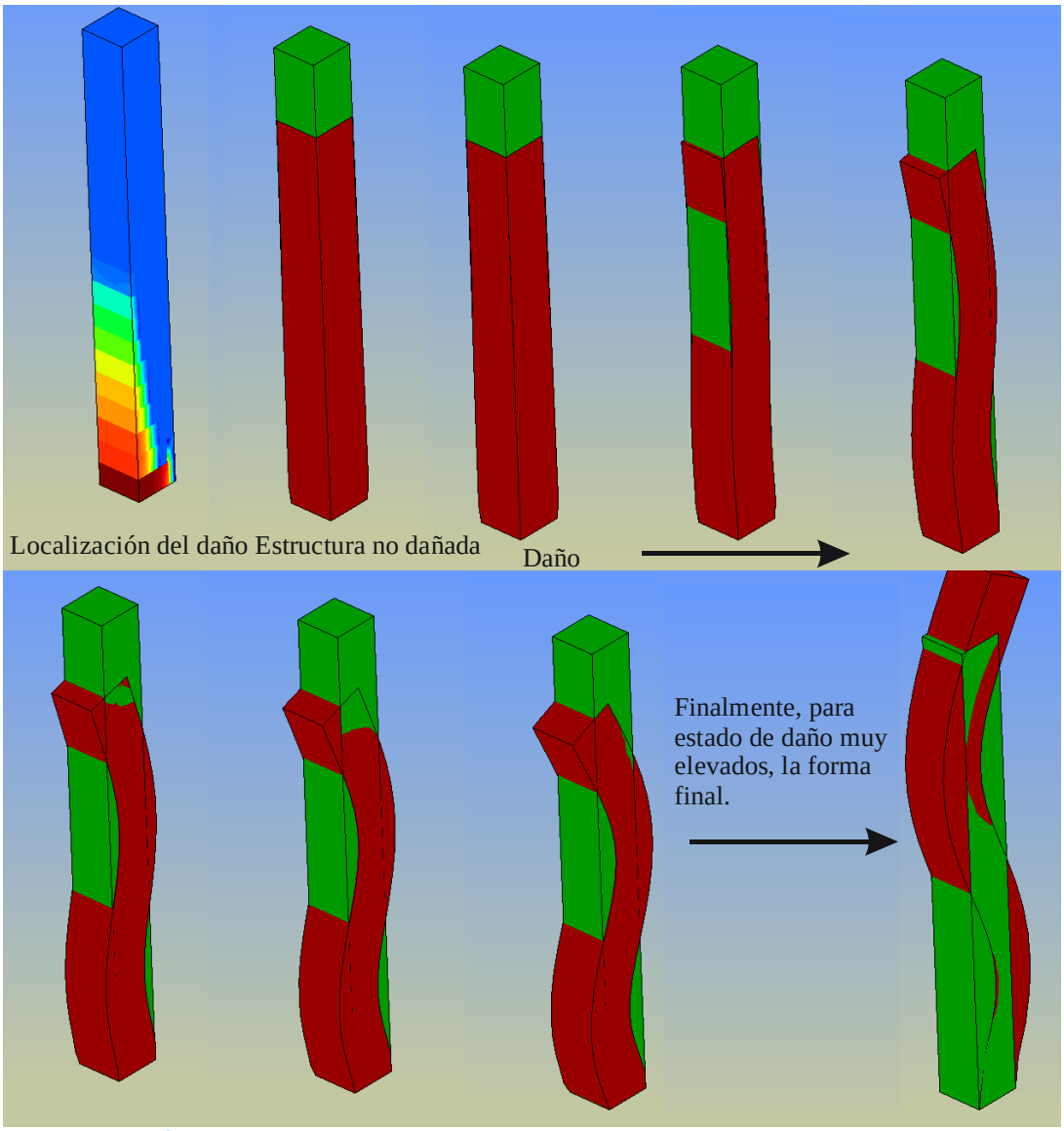


Fig. 9.1.15 Evolución de la sexta forma propia de la columna. A media que el daño aumenta deja de ser un movimiento ascendente y descendente y empieza a ondular, la ondulación aumenta con el daño.

9.2 Columna empotrada con armadura

Para poder apreciar el efecto que tiene la inclusión de una armadura en una estructura para el tipo de problema que nos ocupa, se ha decidido armar la columna de manera muy sencilla, manteniendo el resto de características inalterables, para realizar la comparación. En la figura 9.2.1 se puede observar la armadura utilizada en la columna.

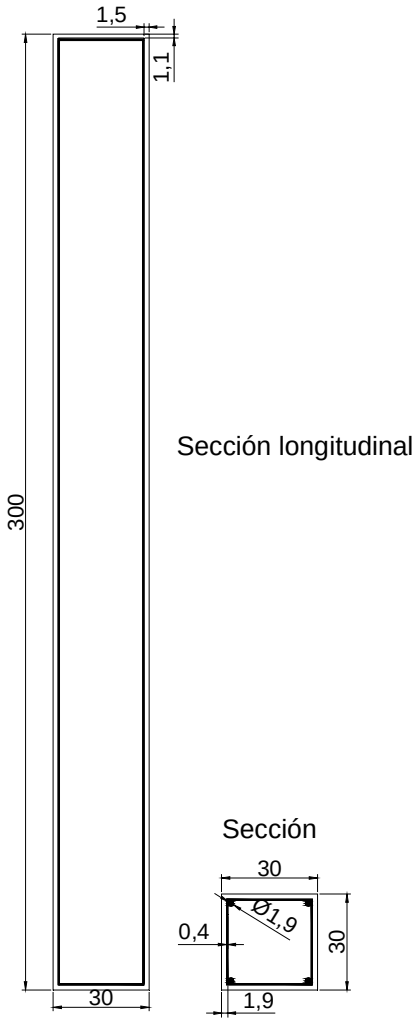


Fig. 9.2.1 Secciones de la columna descrita con la armadura, las dimensiones están en centímetros

Se ha utilizado la misma malla de 500 elementos que en el caso anterior para que la comparación sea realmente entre los columnas los más parecidas posibles, ya que en los elementos finitos la elección de la malla es un aspecto importante. Para realizar el

análisis se ha seguido el mismo procedimiento que el utilizado para la columna sin armadura pero se ha tenido que modificar el .dts original que recibe el PLCD, se ha tenido que definir el material de los elementos de la armadura y asignarlo a los elementos pertinentes. El .dts del CFYFP sigue siendo el mismo por lo que se han calculado las primeras diez frecuencias naturales con sus correspondientes formas. Los resultados obtenidos para las frecuencias en el caso de la columna armada han sido los siguientes:

Daño global	1ª Frecuenc i	2ª Frecuenci	3ª Frecuenci	4ª Frecuenci	5ª Frecuenci	6ª Frecuenci	7ª Frecuenci	8ª Frecuenci	9ª Frecuenci	10ª Frecuenc
0,0000	22,5886	22,5886	135,2288	135,2316	183,7287	328,9808	355,4639	355,5237	550,6112	644,4106
0,0115	22,5789	22,5817	135,1779	135,1972	183,6776	328,9609	355,3601	355,4249	550,4063	644,2495
0,0452	22,3863	22,4936	134,5079	134,8908	183,0794	328,2487	354,2888	354,8109	548,7629	642,9331
0,0993	22,0465	22,3308	133,5377	134,3860	181,9349	326,7112	352,5015	353,7034	546,0494	639,7094
0,1569	21,6195	22,1092	132,4016	133,7087	180,3674	324,6035	350,1837	352,1224	542,6271	635,8720
0,2162	21,1125	21,8360	131,0166	132,8610	178,3912	321,8212	347,2607	350,1267	538,4978	631,3216
0,2619	20,6299	21,5766	129,8454	132,1373	176,3649	319,1505	344,5630	348,3424	534,5089	626,8393
0,3143	20,1359	21,2990	128,4649	131,2495	174,1225	316,2142	341,2393	346,1043	530,3875	621,4350
0,3547	19,6699	21,0402	127,2606	130,4384	171,9846	313,4437	338,0526	344,0250	526,4291	616,6420
0,3994	19,2076	20,7630	125,8166	129,4634	169,6184	310,4482	334,7244	341,6646	522,0190	610,9808
0,4414	18,7553	20,4855	124,3457	128,4497	167,2437	307,3292	331,3364	339,2399	517,5428	605,3393
0,4989	17,9423	19,9925	121,9625	126,7789	162,8341	301,7173	325,2284	335,0994	509,4538	596,1590
0,5423	17,2042	19,5366	119,1725	124,8131	158,5664	296,1836	319,5395	331,1679	501,8108	584,9376
0,6009	16,2535	18,9304	115,9555	122,3684	152,7544	288,5914	311,4837	325,2168	491,4561	571,6378
0,6421	15,4122	18,3940	112,1140	119,5140	147,2687	281,2301	304,1480	319,4221	480,6082	556,0745
0,7024	14,2408	17,5298	107,2112	114,7874	139,6742	269,7141	292,8969	310,3728	465,1720	536,7700

Tabla 9.2.2a Tabla de los resultados obtenidos para las diez primeras frecuencias naturales y los distintos estados de daño.

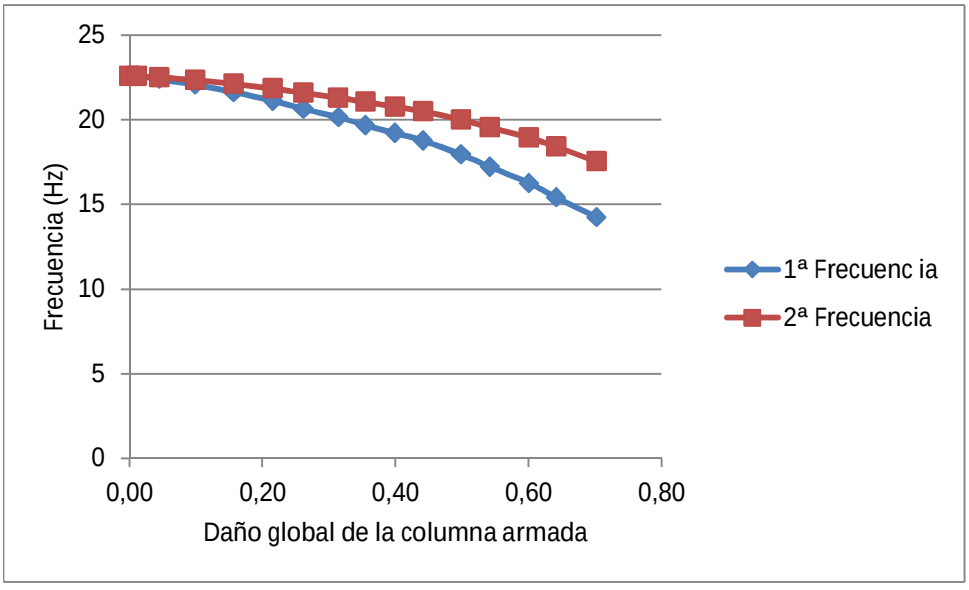


Fig.9.2.2b Evolución de las dos primeras frecuencias naturales para la columna con armadura.

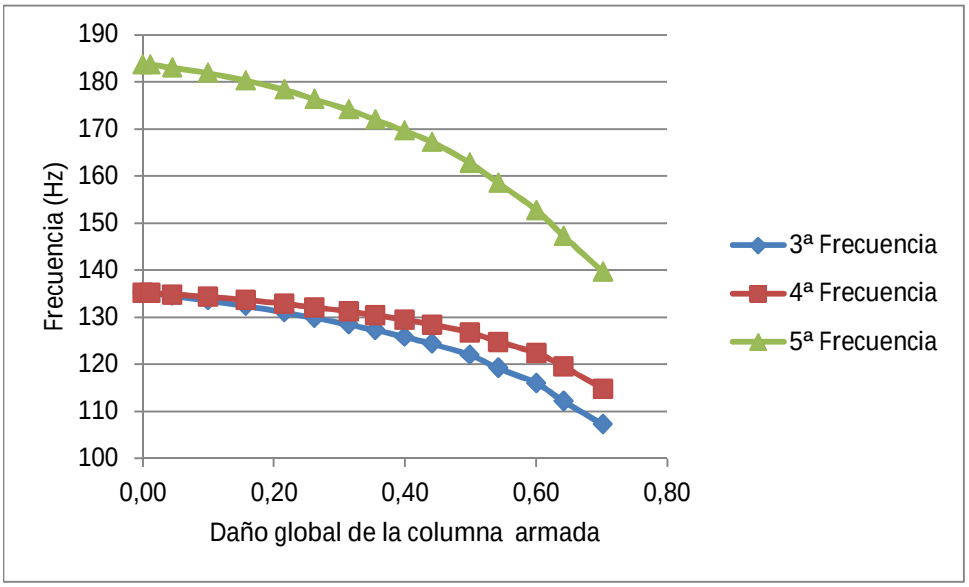


Fig. 9.2.2c Evolución al dañarse de la tercera, cuarta y quinta frecuencias naturales de la columna armada.

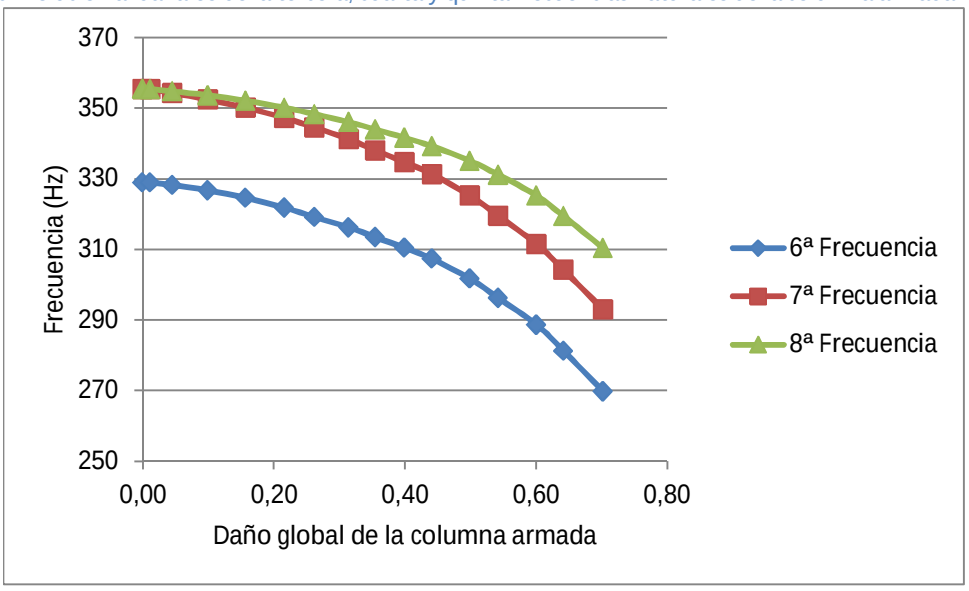


Fig. 9.2.2d Evolución de la sexta, séptima y octava frecuencias de la columna armada al dañarse

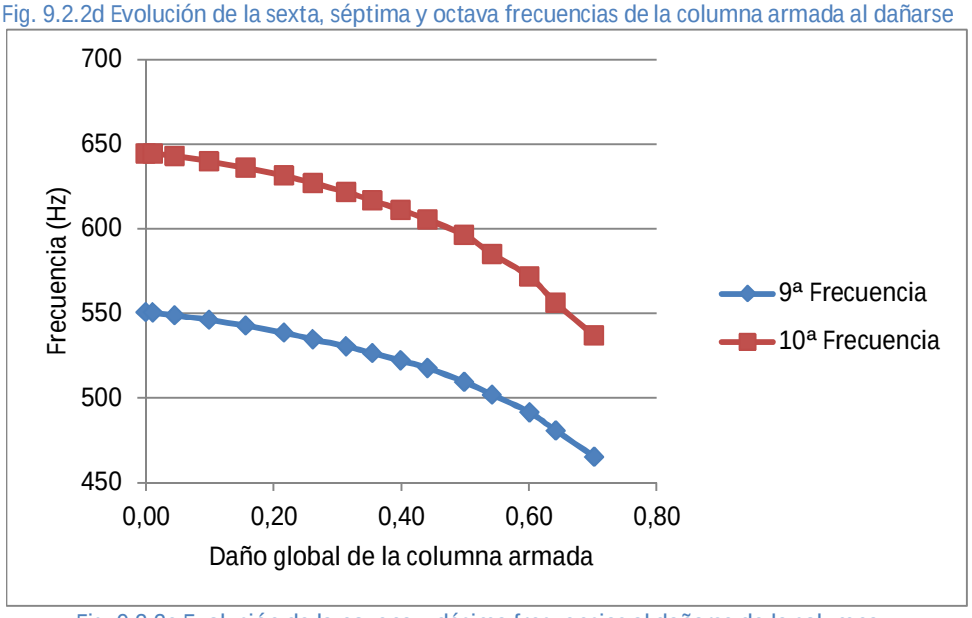


Fig. 9.2.2e Evolución de la novena y décima frecuencias al dañarse de la columna

En las figuras 9.2(a-e) se observa un descenso del valor de todas las frecuencias naturales a medida que la columna se va dañando, al igual que en la columna sin armadura se puede observar que el descenso no es lineal ni igual para todas las frecuencias. De la misma manera que la observada en el ejemplo anterior, frecuencias que inicialmente presentan un mismo valor evolucionan de manera distinta debido a la localización del daño, esto puede ser observado si se comparan la primera y la segunda, la tercera y la cuarta y la sexta y la séptima frecuencias. La evolución y localización del daño (figura 9.2.3) en este caso es un poco diferente de la observada en la primera columna estudiada.

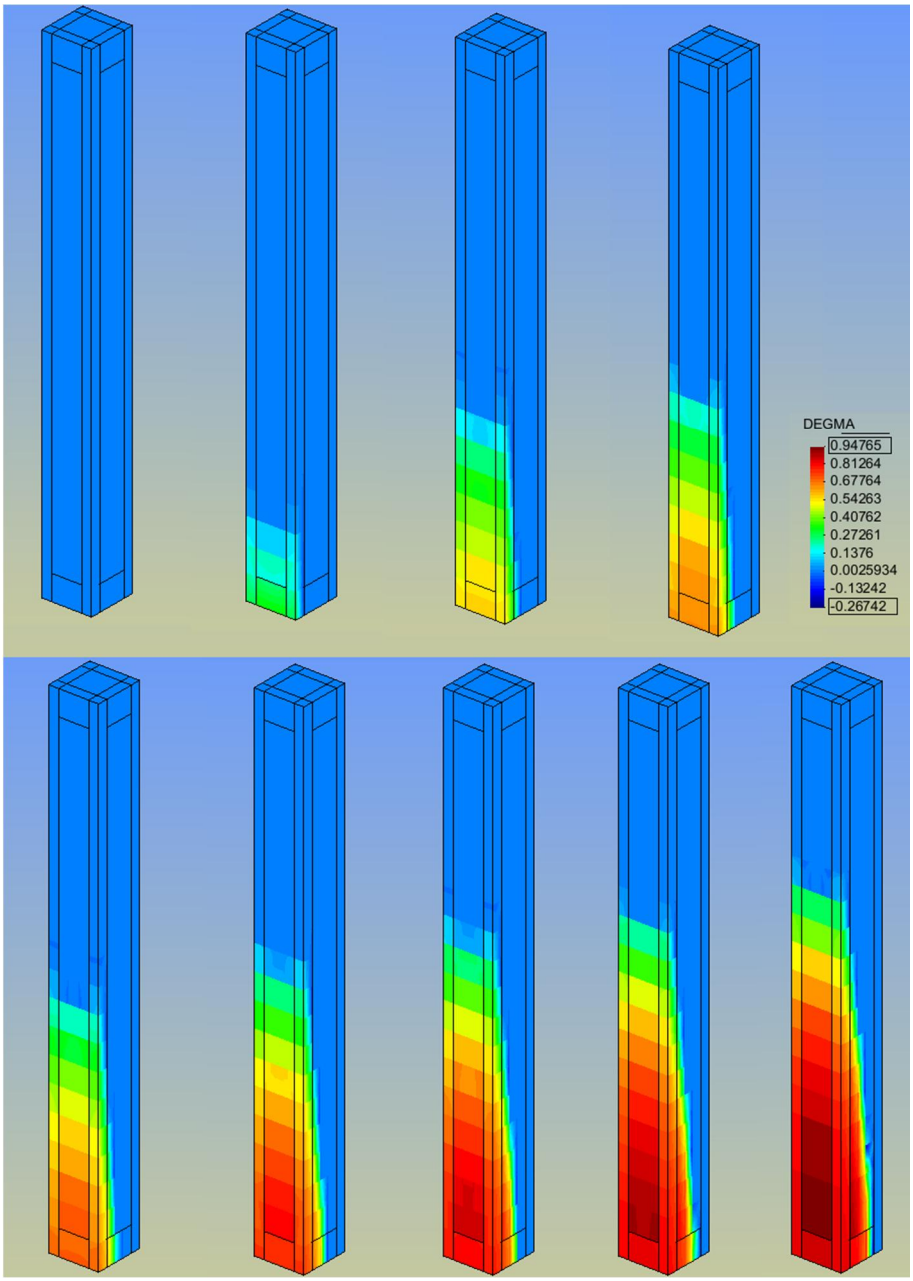


Fig. 9.2.3 Evolución del daño en la columna con armadura

El daño empieza en la parte inferior de la columna y se va subiendo, al principio la parte con armadura tiene mayor daño pero para estados más avanzados la parte sin armadura registra valores más altos. Si se compara con la columna sin armadura se puede observar que el daño afecta a más elementos en este caso.

Esta localización del daño hace que se observen los mismos efectos que en el primer ejemplo, la primera frecuencia se ve más afectada que la segunda por el daño y, por eso, el descenso es mayor en el primer caso que en el segundo. Este efecto también se observa aunque en menor medida en otros casos como: la tercera y la cuarta frecuencia y la sexta y la séptima. Todo depende de la forma de vibración y la localización del daño. Se debe a que la una de ellas es en la dirección del mayor daño y la otra es totalmente perpendicular, aunque las dos se ven afectadas una en mayor medida, debido a forma propia de la frecuencia. Las formas propias obtenidas son las siguientes:

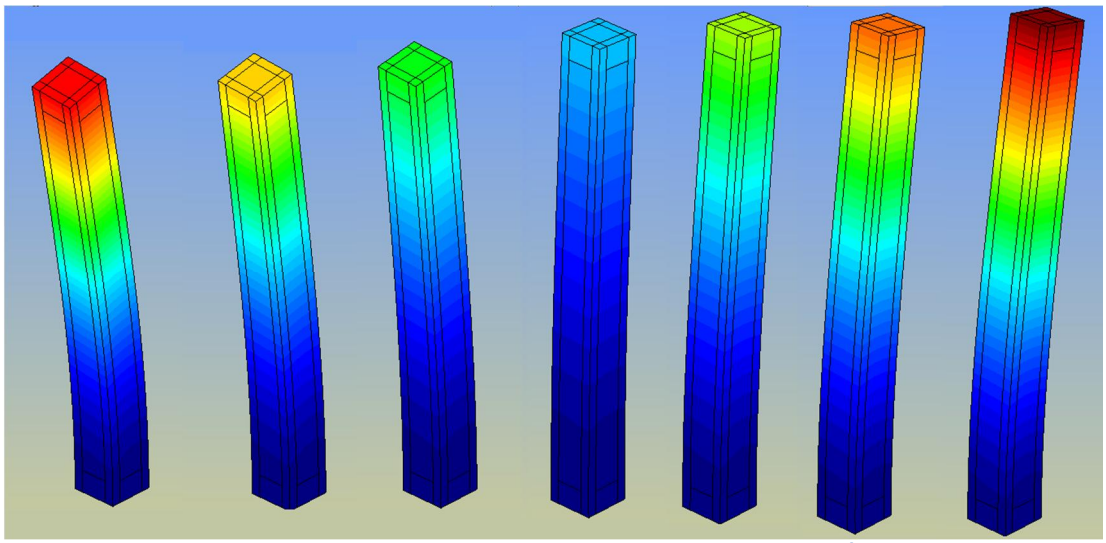


Fig 9.2.4 Forma propia de la primera frecuencia, la segunda es igual pero en dirección perpendicular a esta.

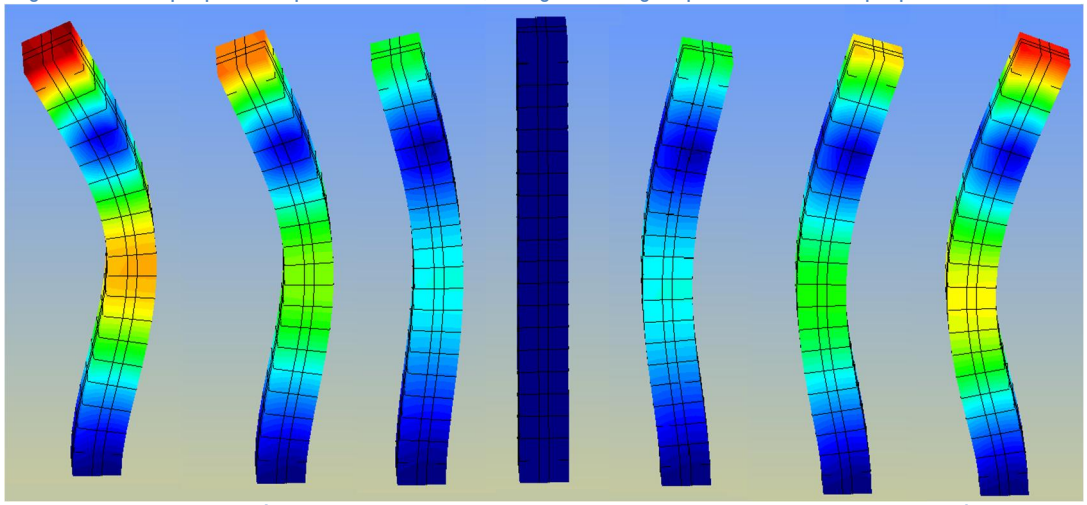


Fig. 9.2.5 Animación de la tercera forma propias, la cuarta es igual pero perpendicular a ésta.

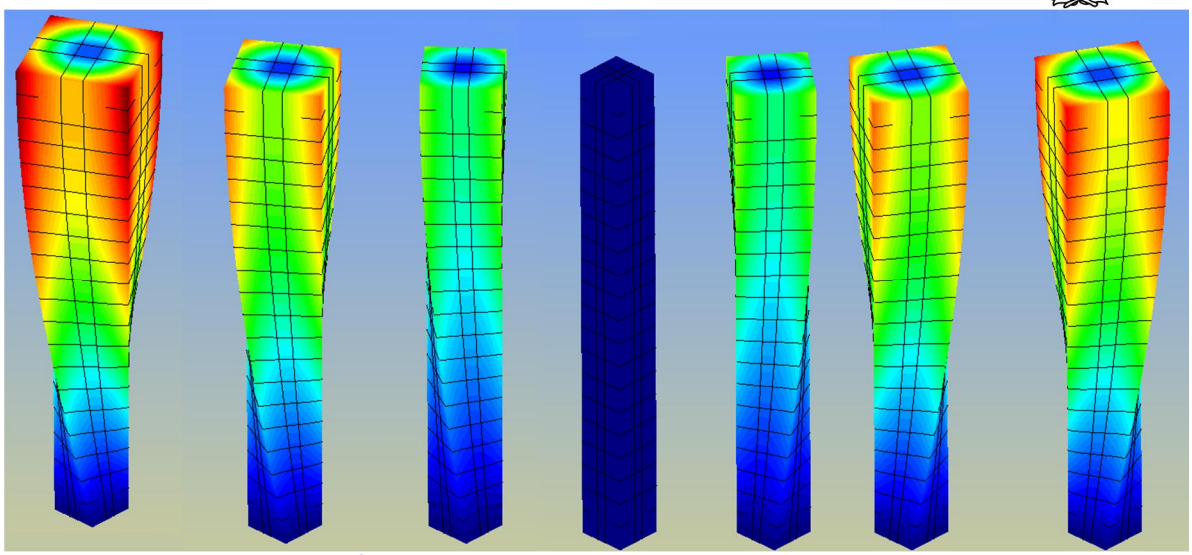


Fig. 9.2.6 Animación de la forma propia asociada a la quinta frecuencia natural.

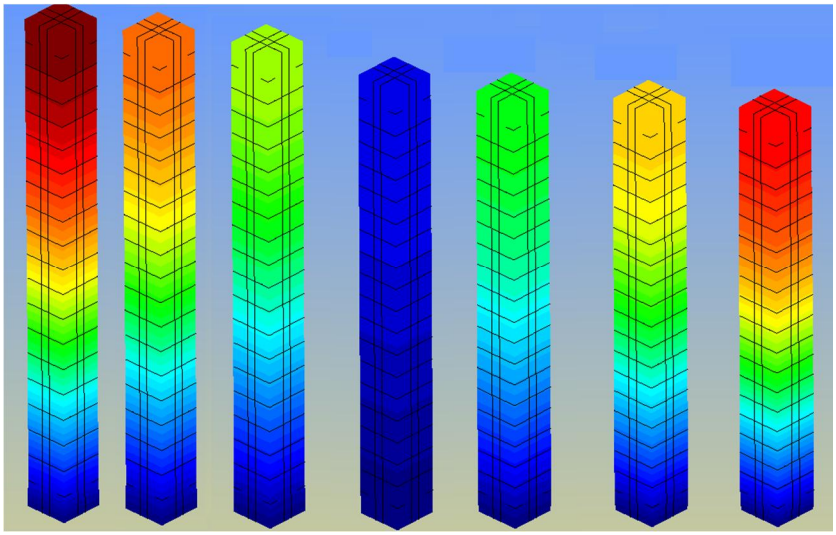


Fig. 9.2.7 Animación de la sexta forma propia.

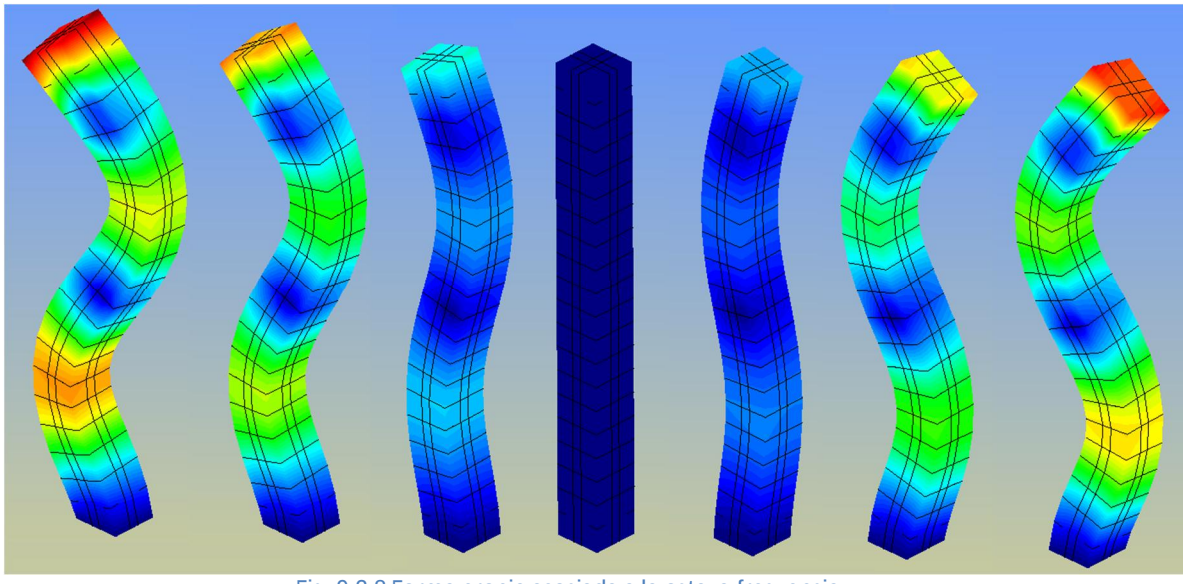


Fig. 9.2.8 Forma propia asociada a la octava frecuencia.

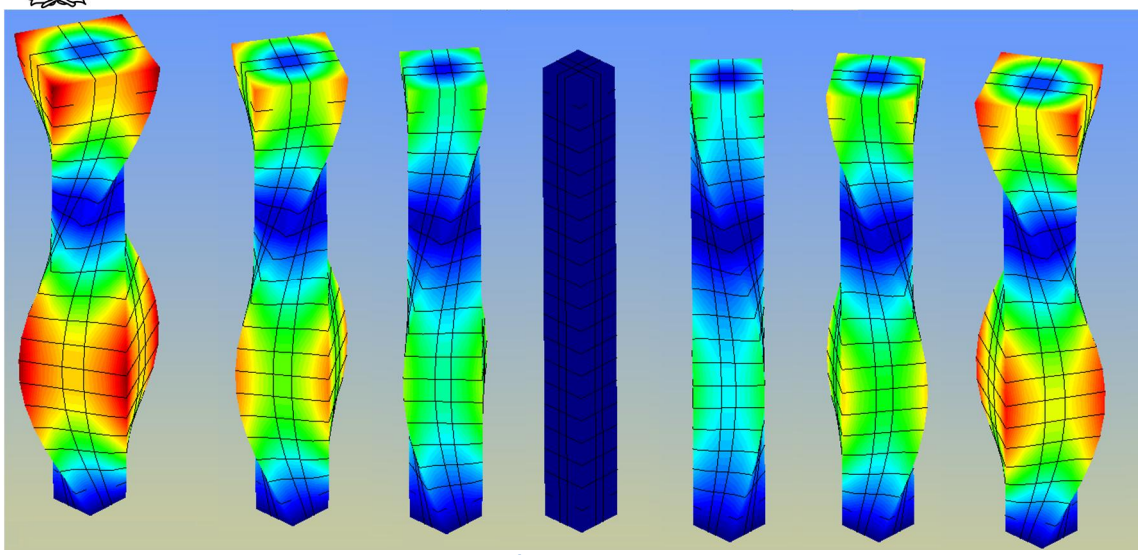


Fig. 9.2.9 Animación de la novena forma propia

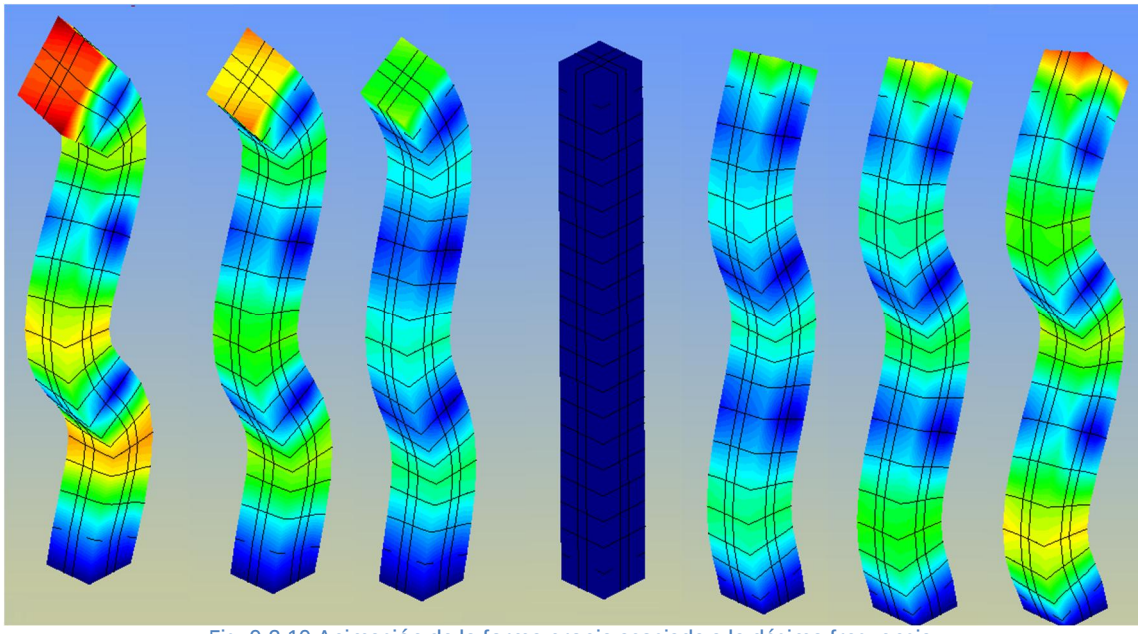


Fig. 9.2.10 Animación de la forma propia asociada a la décima frecuencia

Las figuras de las formas propias obtenidas para la columna con armadura no dañada son prácticamente las mismas que las obtenidas para la columna no armada. Se puede apreciar que el daño no afectará de igual manera a una forma muy simple como puede ser la primera frecuencia que a otra mucho más compleja como es la última calculada. Como se puede apreciar en la figura 9.2.2 en términos de valor absoluto, el descenso del valor de la frecuencia es mayor cuanto mayor es la frecuencia, pero es más interesante observar estos descensos en términos relativos, al igual que se hizo con la primera columna estudiada.

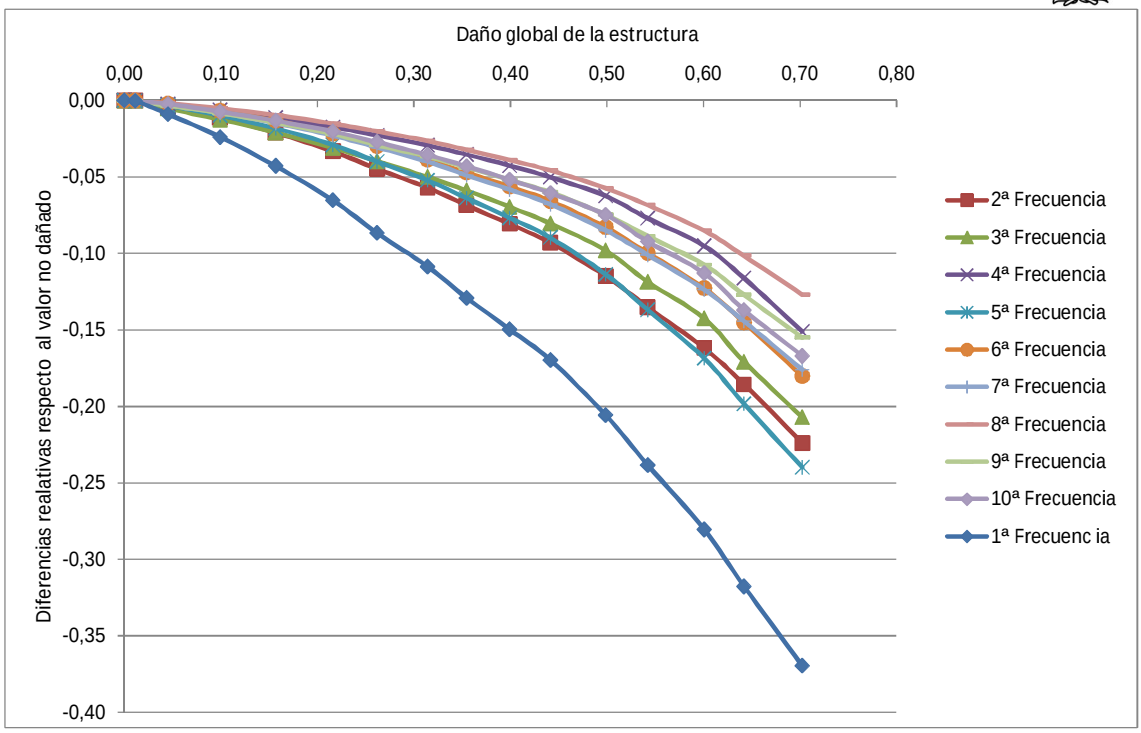


Fig. 9.2.11 Diferencias relativas en tanto por ciento entre las frecuencias dañadas y la inicial.

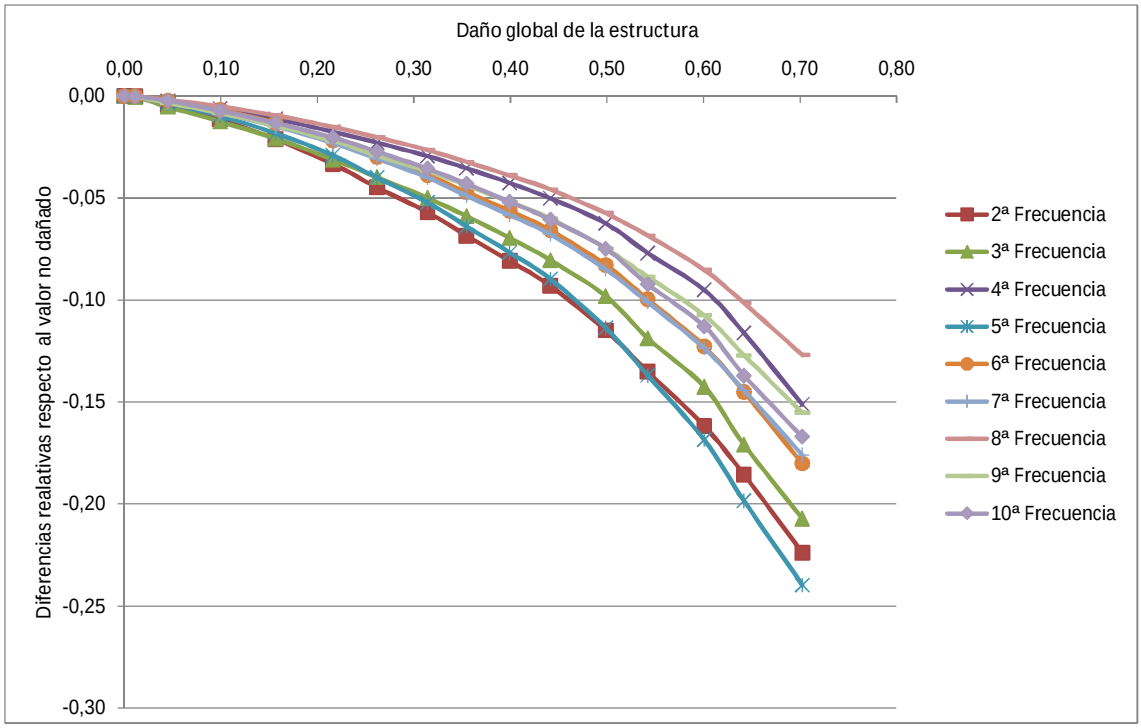


Fig. 9.2.12 Misma gráfica que la 9.2.11 pero sin tener en cuenta la primera frecuencia, ya que es sensiblemente diferente al resto.

Las figuras 9.2.11 y 9.2.12 representan los mismos datos, pero en la segunda no se incluye la información e la primera frecuencia, ya que se ve muy afectada por la localización del daño en este caso y distorsiona las observaciones. En la figura 9.2.12 se puede apreciar que todas la frecuencias presentan comportamientos muy similares entre ellos, similares a su vez en la forma a los observados para la columna sin armadura. No

sólo eso, sino que las frecuencias que se alejaban más del primer valor siguen siendo las mismas que eran en el caso de la primera columna. Lo que sí es diferente se compara con la figura 9.1.6 es el valor de las diferencias. En el caso de la columna sin armadura para un daño global de 0.60 se tenía unas diferencias que oscilaban entre un 5-12% respecto a la frecuencia original no dañada y, en este caso, tenemos que las diferencias están aproximadamente entre un 8 y un 17%.

	Todas	Sin la primera		
Daño global	Promedio	Promedio	Máximo	Mínimo
0,0000	0,0000	0,0000	0,0000	0,0000
0,0115	0,0290	0,0274	0,0377	0,0061
0,0452	0,3773	0,3198	0,5331	0,2005
0,0993	0,9986	0,8430	1,2505	0,5120
0,1569	1,8006	1,5240	2,1220	0,9567
0,2162	2,7873	2,3709	3,3318	1,5181
0,2619	3,7154	3,1648	4,4798	2,0199
0,3143	4,7512	4,0726	5,7087	2,6494
0,3547	5,7161	4,9156	6,8548	3,2343
0,3994	6,7701	5,8593	8,0818	3,8982
0,4414	7,8334	6,8182	9,3104	4,5803
0,4989	9,6996	8,4919	11,4929	5,7449
0,5423	11,5639	10,2003	13,6953	6,8507
0,6009	14,0074	12,4476	16,8587	8,5246
0,6421	16,4427	14,7397	19,8445	10,1545
0,7024	19,9703	18,0831	23,9780	12,6999

Tabla 9.2.13 Diferencias relativas promedios y máxima y mínima para cada estado de daño.

En el caso de la columna armada vemos que para los estados de daño estudiados la primera frecuencia no distorsiona tanto los resultados como ocurría en el caso de la primera columna no armada. Pero en este caso además de tener que los valores promedio son sensiblemente más altos que en la columna no armada, para valores aproximados de un 50% de daño global tenemos un 8.5% de promedio frente a un 5% que teníamos, las desviaciones respecto a éste también son mayores, ±3% frente a un ±2%. Como aparecen bastante agrupadas se puede intentar establecer una correlación entre estas diferencias relativas y el valor de daño global, pero está vez se intentará en vez de usar una correlación lineal una cuadrática, para poder comparar el error producido con la forma utilizada anteriormente.

Daño global	Diferencia promedio	Daño calculado	Error absoluto
0,0000	0,0000	0,0000	0,0000
0,0115	-0,0003	0,0000	0,0115
0,0452	-0,0032	0,0334	0,0118
0,0993	-0,0084	0,1180	-0,0187
0,1569	-0,0152	0,1803	-0,0234
0,2162	-0,0237	0,2376	-0,0214
0,2619	-0,0316	0,2816	-0,0197
0,3143	-0,0407	0,3251	-0,0109
0,3547	-0,0492	0,3611	-0,0064
0,3994	-0,0586	0,3977	0,0016
0,4414	-0,0682	0,4319	0,0095
0,4989	-0,0849	0,4861	0,0128
0,5423	-0,1020	0,5360	0,0064
0,6009	-0,1245	0,5955	0,0054
0,6421	-0,1474	0,6507	-0,0086
0,7024	-0,1808	0,7240	-0,0216

Tabla 9.2.14 Comparación entre el daño global de la columna dañada calculado usando las rutinas y usando las diferencias promedio entre las frecuencias dañadas y las originales (sin tener en cuenta la primera).

En la tabla 9.2.14 se puede observar el resultado final de este cálculo, no se ha tenido en cuenta la primera frecuencia porque se ha visto en la figura 9.2.11 que su comportamiento era sensiblemente diferente al resto. Primero se ha calculado la diferencia relativa entre cada frecuencia dañada y la original, posteriormente se ha hecho el promedio de estas diferencias para cada nivel de daño. Se propone utilizar la siguiente fórmula cuadrática:

$$\frac{f_i - f_o}{f_o} = -0.3223d^2 - 0.0131d - 0.0024 \qquad \text{Ec. 9.2.1}$$

Dónde f_i es la frecuencia obtenida para un determinado nivel de daño, f_o es la frecuencia original y d es el daño global de la columna con armadura. La mejora de la forma cuadrática respecto a la lineal es apreciable, cuanto mayor es el estado de daño mejor es la ganancia conseguida. Vemos que para niveles daño global de un 0.70 el error es de un 2% mientras que al usar la aproximación lineal el error era más de 10 veces superior. Tal y como se puede apreciar en la figura 9.2.15 no hay que irse a valores tan elevados de daño para discernir la sensible mejora que se produce respecto a la aproximación lineal al calcular el daño. Si bien es cierto es que para valores de daño bajos se puede usar una aproximación lineal ya que es más sencilla y los resultados son similares.

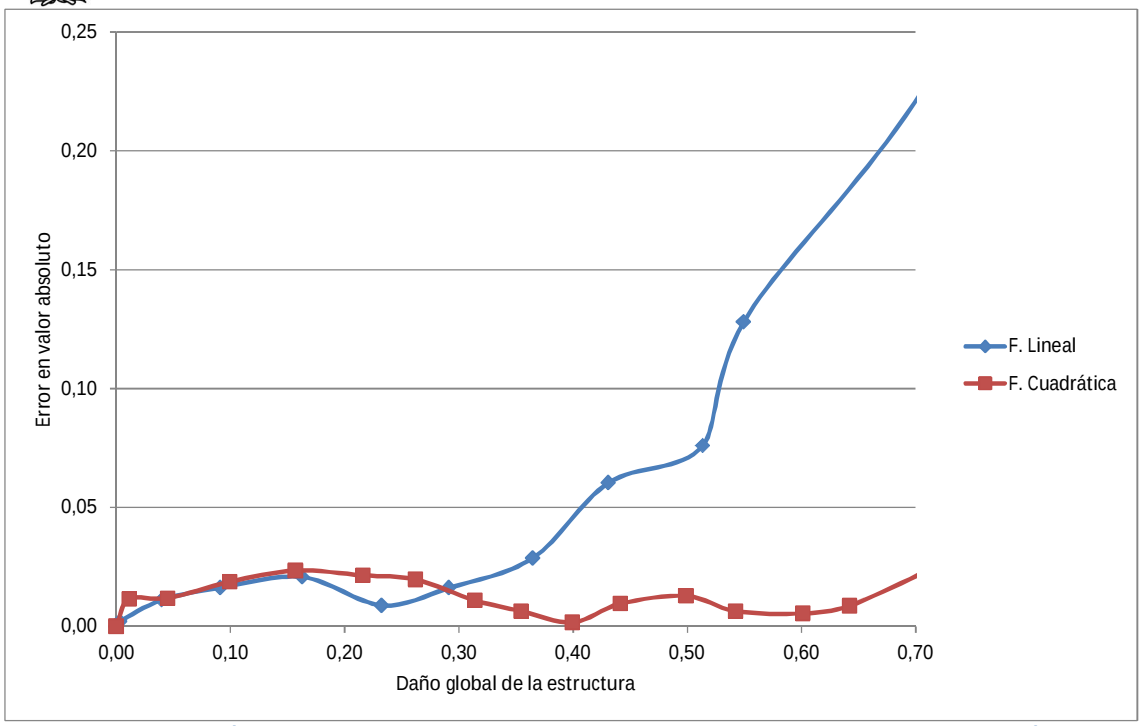


Fig. 9.2.15 Comparación entre los errores (en valor absoluto) cometidos al utilizar una aproximación lineal y cuadrática para calcular el daño global de la estructura a partir de las frecuencias obtenidas.

Si se comparan los niveles de daño sufridos por las dos columnas para niveles de desplazamientos impuestos de igual magnitud (figura 9.2.16) se puede observar que la armada tiene niveles de daño claramente inferiores a la primera, tal y como cabía esperar.

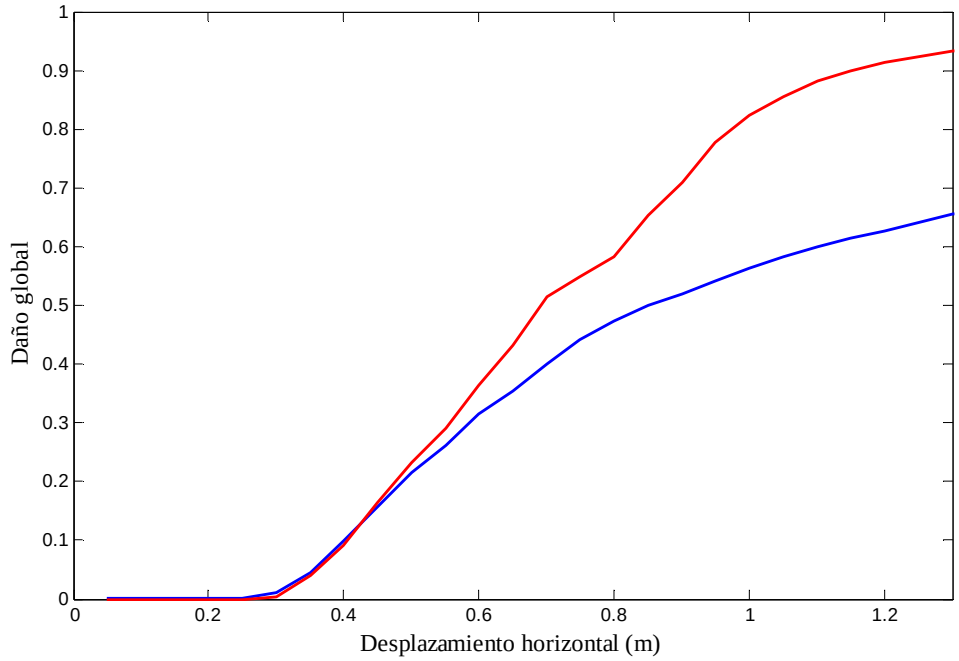


Fig. 9.2.16 Comparación de los niveles de daño global entre la columna armada (azul) y la columna sin armadura (rojo), al imponer unos desplazamientos de igual magnitud.

En lo que no se observa diferencias es en el comportamiento de las frecuencias a medida que aumentan los desplazamientos impuestos en la columna. La forma obtenida se puede apreciar en la figura 9.2.17 que tiene la misma forma que la vista en la figura 9.1.10 para la columna sin armadura.

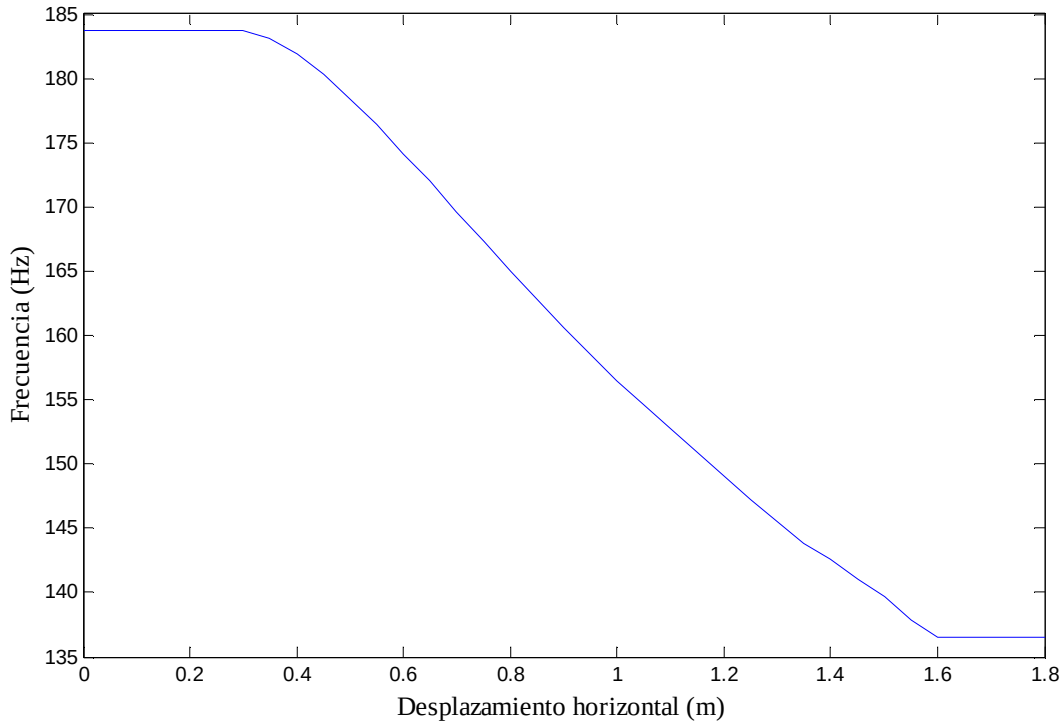


Fig. 9.2.17 Evolución de la quinta frecuencia a medida que aumenta el desplazamiento impuesto, es el mismo comportamiento observado en el resto de frecuencias.

En cuanto a la evolución de las formas propias es términos generales igual en la columna con armadura y la vista con anterioridad. Se producen pequeños giros y oscilaciones a medida que se dañan en la mayoría de ellas, tal y como se ha visto en la columna no armada. Las formas torsivas, como son las asociadas a la quinta y a la novena frecuencias, disminuyen la torsión a medida que se dañan (figura 9.2.18).

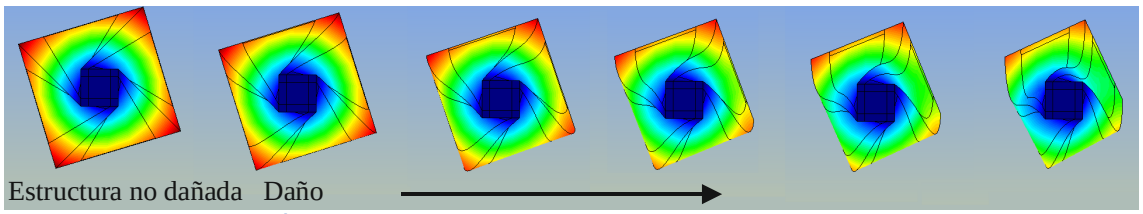


Fig. 9.2.18 Evolución de la forma asociada a la quinta frecuencia al dañarse, vista vertical mirando desde la parte de la columna fija (inferior)

Otro efecto ya observado con anterioridad, aunque no tan marcado es el que se presenta en la figura 9.2.19.

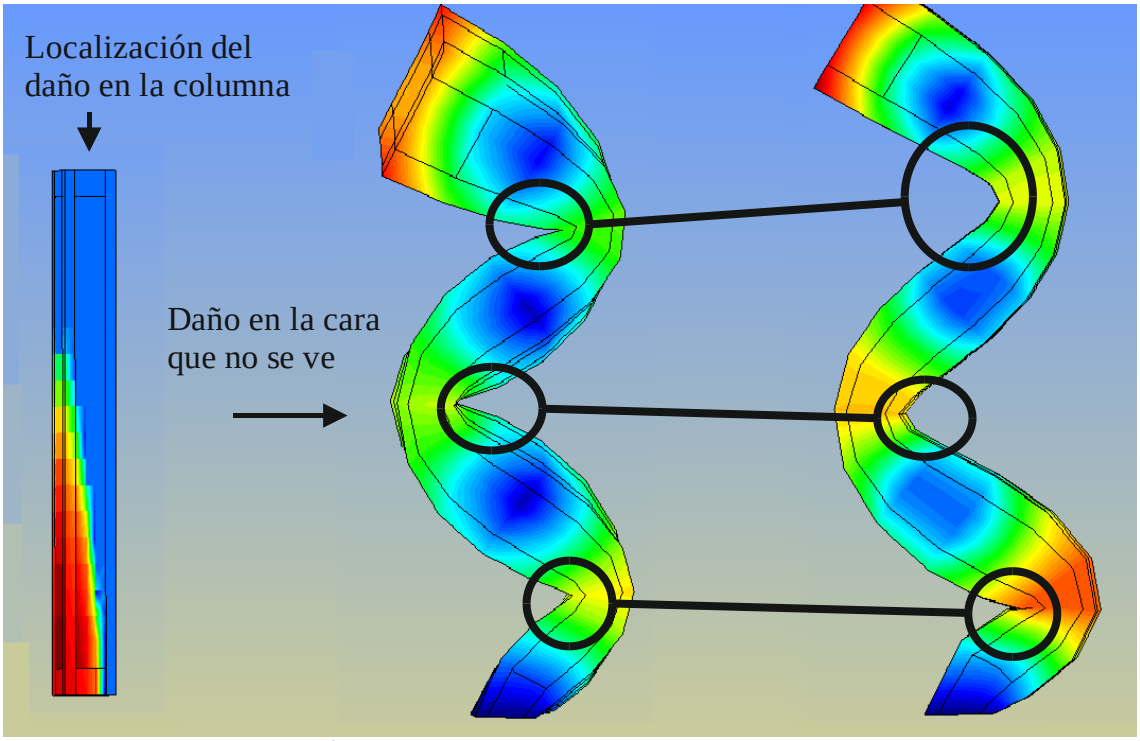


Fig. 9.2.19 Forma propia de la décima frecuencia no dañada (izquierda) y la correspondiente a un daño como el mostrado en la figura (derecha).

Además de unos pequeños giro en la orientación de la forma, lo que se puede apreciar en los círculos marcados es como algunos giros pasan a ser más cerrados (el más inferior) mientras que otros se tornan más suaves (los dos superiores). Asimismo se observa una especie de achatamiento en la parte inferior de la figura, es como si el daño hiciera que es parte fuera mayor, este hecho se a podido observar en otras formas propias, en las que la parte inferior de la columna parece abombarse y aumentar ligeramente de volumen.

9.3 Viga biapoyada

El tercer caso estudiado es una viga apoyada en sus extremos, uno de ellos está completamente fijo mientras que el otro permite desplazamientos en el sentido longitudinal de la viga. Para dañar la viga se ha impuesto un movimiento vertical descendente en los nodos superiores centrales de la viga. Debido al comportamiento del hormigón cuando se ve sometido a este tipo de esfuerzos los desplazamientos impuestos se han tenido que ir haciendo cada vez más pequeños para obtener la mayor parte de la

curva de comportamiento de la viga y de daño global, cosa muy difícil por la rotura frágil que sufre el material, ésta sucede cuando el hormigón alcanza su capacidad de deformación última, debido a eso los valores para el daño global alcanzado son mucho menores que los observados en otros ejemplos. La discretización elegida en este caso es de 756 nodos y 500 elementos, que es más densa que las anteriores utilizadas ya que la viga es tres veces más corta. Para evitar parte de los problemas que se tenían al intentar utilizar este ejemplo, se han tenido que modificar los materiales utilizados en dos tipos de elementos, en los que están situados en los apoyos de la pieza y en los que se imponen los desplazamientos, todo ello debido a las tensiones que se generan en esas zonas que no permiten que el análisis no lineal avance. Todo esto se puede observar en la figura 9.3.1

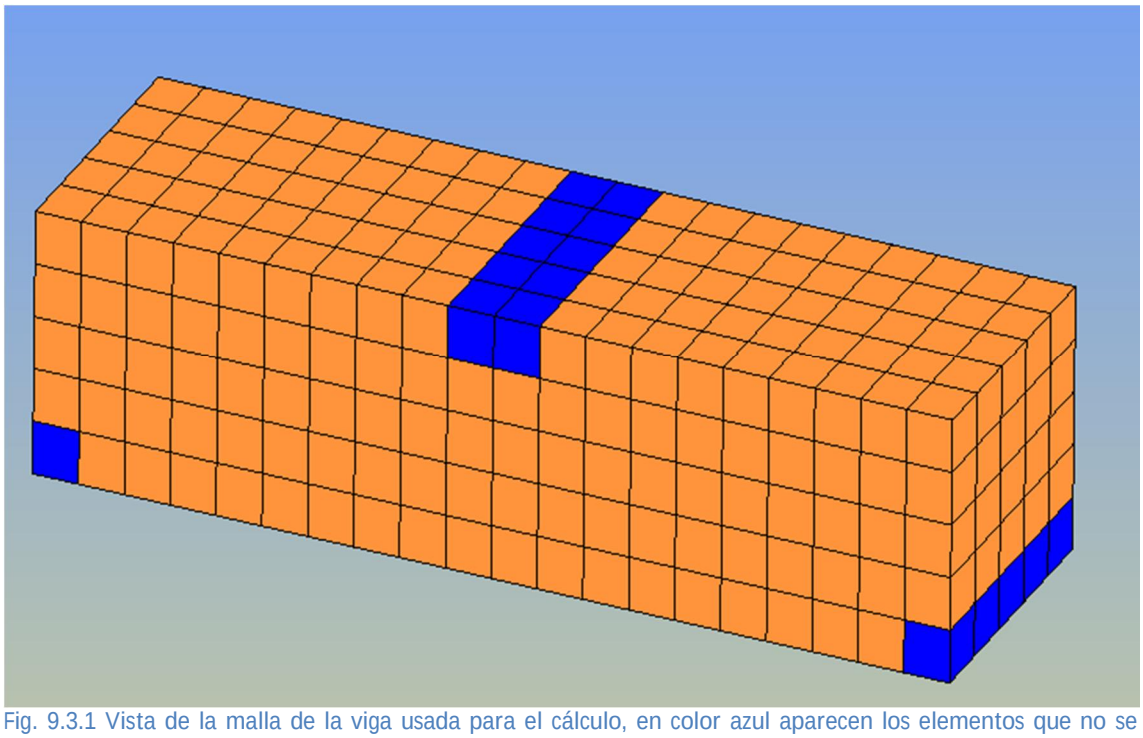


Fig. 9.3.1 Vista de la malla de la viga usada para el cálculo, en color azul aparecen los elementos que no se dañarán para poder llevar a cabo el análisis, que corresponden a los puntos de apoyo y al lugar en el que imponemos el desplazamiento. La malla consiste en 500 elementos repartidos en 25 hexahedros en cada uno de los veinte niveles que hay a lo largo de su longitud.

El proceso seguido para realizar el análisis ha sido el mismo que el comentado en las estructuras anteriores, pero poniendo especial atención en la definición de los incrementos de desplazamientos que se quería realizar, debido al tipo de rotura involucrada con esta estructura. En la figura 9.3.2 se puede ver tanto la curva de comportamiento como la del daño global, como se puede observar la curva ya ha llegado al máximo y se encuentra bajando, pero la solución no convergerá a pesar de

que el nivel de daño es bajo porque un pequeño desplazamiento vertical implica ya la rotura de la estructura.

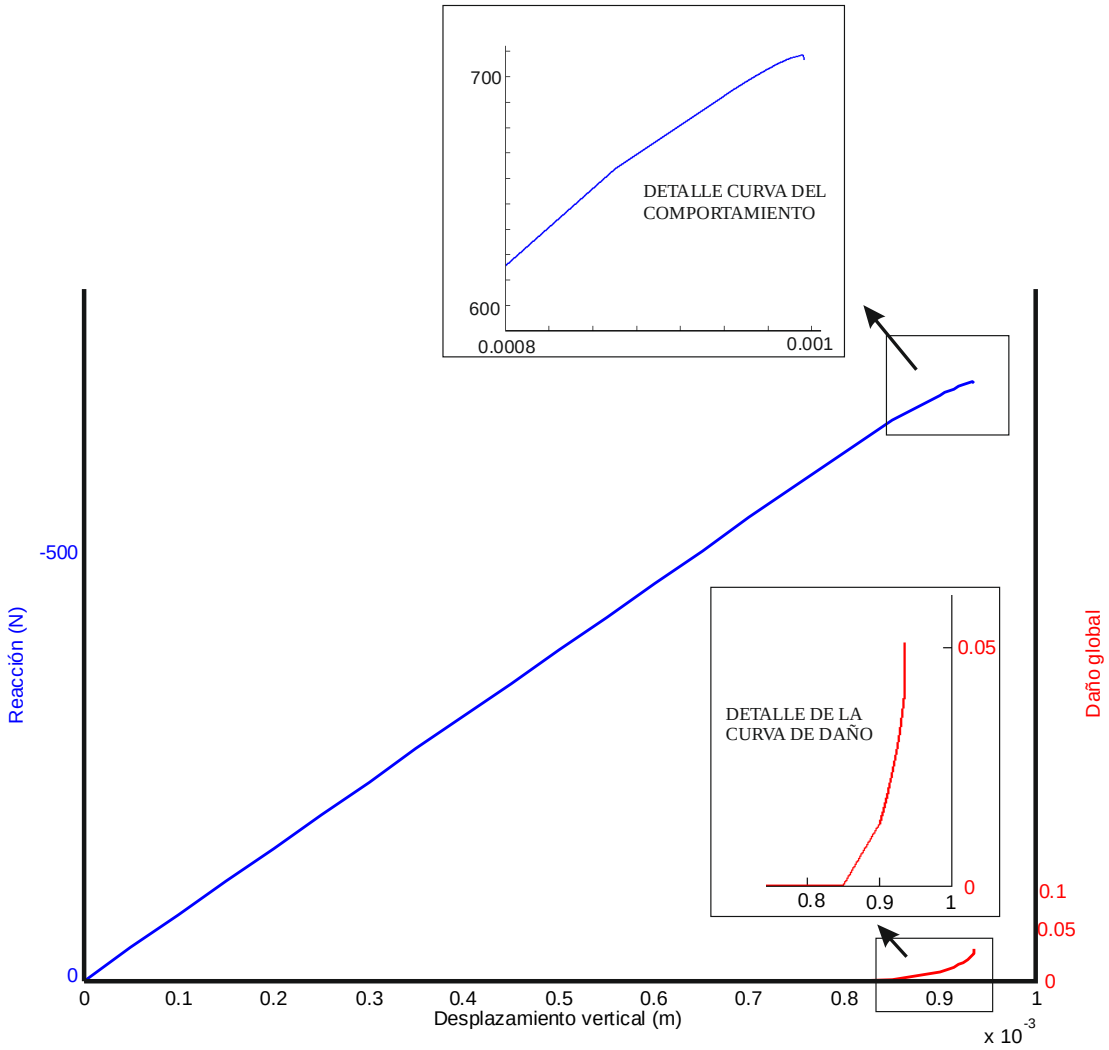


Fig. 9.3.2 Curvas del comportamiento de la viga y del daño global en relación con los desplazamientos, se incluye el detalle de las áreas de las curvas en las que el análisis deja de converger.

En los detalles de la figura 9.3.2 se puede observar tanto que el daño tiende a dispararse por pequeño que sea el incremento, ya que apunta casi en vertical, como que el comportamiento tiende a bajar casi verticalmente. Debido a estos dos hechos continuar con el análisis de igual manera resulta imposible, para hacerlo se tendría que variar la estrategia utilizada.

La evolución de las frecuencias para los bajos niveles de daño que se han podido estudiar, han sido los siguientes:

Daño	1ª Frec.	2ª Frec.	3ª Frec.	4ª Frec.	5ª Frec.	6ª Frec.	7ª Frec.	8ª Frec.	9ª Frec.	10ª Frec.
0,0000	333,2476	340,1944	633,5436	661,8032	796,8049	1144,0953	1504,3019	1669,7915	1714,2512	2097,1803
0,0130	333,0344	338,3361	633,3564	660,5656	796,2905	1143,9026	1501,7272	1660,3582	1706,9193	2090,2653
0,0214	333,0218	338,0789	633,2485	660,4982	796,1465	1143,8377	1501,5409	1658,5842	1705,7973	2089,0113
0,0333	332,9301	337,1285	632,8603	659,9918	795,6223	1143,6144	1500,3038	1653,5201	1701,7619	2084,8370
0,0401	332,9164	337,1810	632,6372	659,9207	795,3927	1143,5223	1499,9968	1654,9227	1701,9644	2084,6492
0,0502	332,8556	336,3463	632,5272	659,5832	795,1917	1143,4521	1499,3109	1649,4815	1698,4753	2081,5362

Tabla 9.3.3a Evolución de las diez primeras frecuencias naturales de la viga biapoyada estudiada.

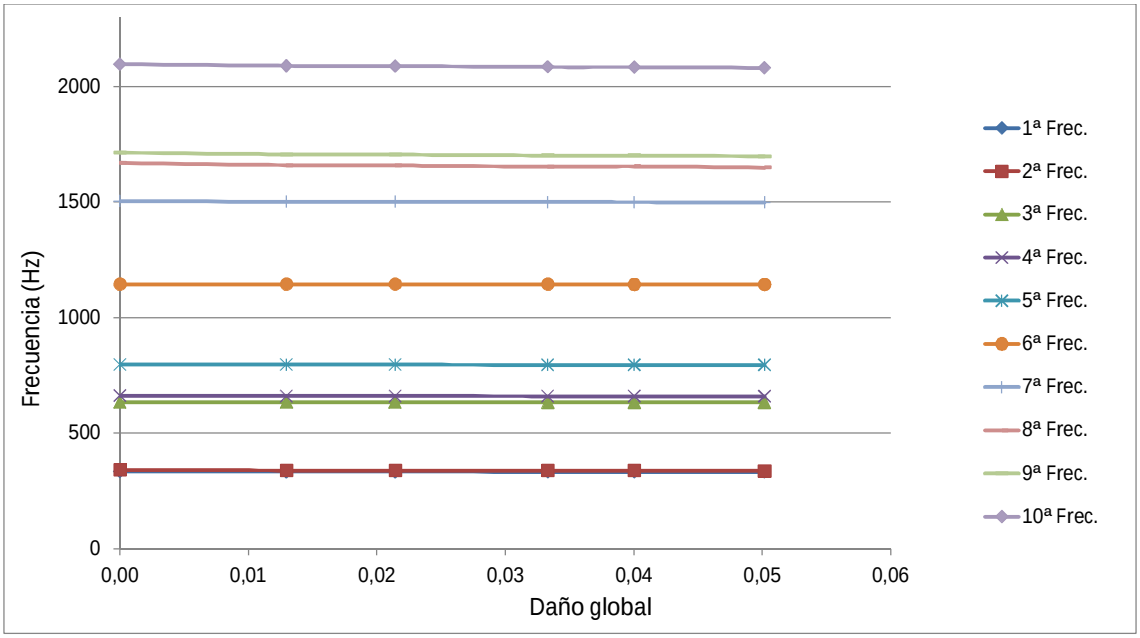


Fig. 9.3.3b Evolución gráfica de las frecuencias calculadas.

Como se puede ver de manera numérica en la tabla 9.3.3a y de manera gráfica en la 9.3.3b las frecuencias prácticamente se mantienen constantes para los estados de daño tan bajos calculados. Incluso cuando observamos las diferencias relativas entre una frecuencia dañada y su respectiva original en tanto por ciento (figura 9.3.4) las diferencias observadas son inferiores al 1% en todos los casos menos en uno, de forma que las diferencias son casi inapreciables.

Daño	1ª Frec.	2ª Frec.	3ª Frec.	4ª Frec.	5ª Frec.	6ª Frec.	7ª Frec.	8ª Frec.	9ª Frec.	10ª Frec.
0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
0,0130	-0,0640	-0,5463	-0,0295	-0,1870	-0,0646	-0,0169	-0,1712	-0,5649	-0,4277	-0,3297
0,0214	-0,0678	-0,6219	-0,0466	-0,1972	-0,0826	-0,0225	-0,1835	-0,6712	-0,4932	-0,3895
0,0333	-0,0953	-0,9012	-0,1078	-0,2737	-0,1484	-0,0420	-0,2658	-0,9745	-0,7286	-0,5886
0,0401	-0,0994	-0,8858	-0,1431	-0,2844	-0,1772	-0,0501	-0,2862	-0,8905	-0,7167	-0,5975
0,0502	-0,1176	-1,1311	-0,1604	-0,3354	-0,2025	-0,0562	-0,3318	-1,2163	-0,9203	-0,7460

Tabla 9.3.4 Diferencias relativas en % entre una frecuencia dañada y la respectiva sin daño alguno.

Si se observa la expresión gráfico del cuadro mostrado en la figura 9.3.4, se pueden apreciar que las trayectorias no son tan iguales entre frecuencias como eran en el caso de la columna, existen como dos grupos: la segunda y las tres últimas forman uno

y el resto otro. Con niveles de daño tan pequeños tampoco se pueden establecer conclusiones y harían falta análisis más extensos que nos permitan establecer alguna correlación clara como se ha hecho en los otros casos. Ya que aunque parece que las formas que tienen movimientos en la zona de localización del daño (parte inferior central de la viga) aparecen ser las más afectadas en algunos casos, no lo son en todos. Además al ser tan pequeños las diferencias entre las frecuencias calculadas y las originales carece de sentido intentar establecer una relación entre éstas y el daño ya que cualquier correlación obtendrá buenos resultados para valores tan pequeños.

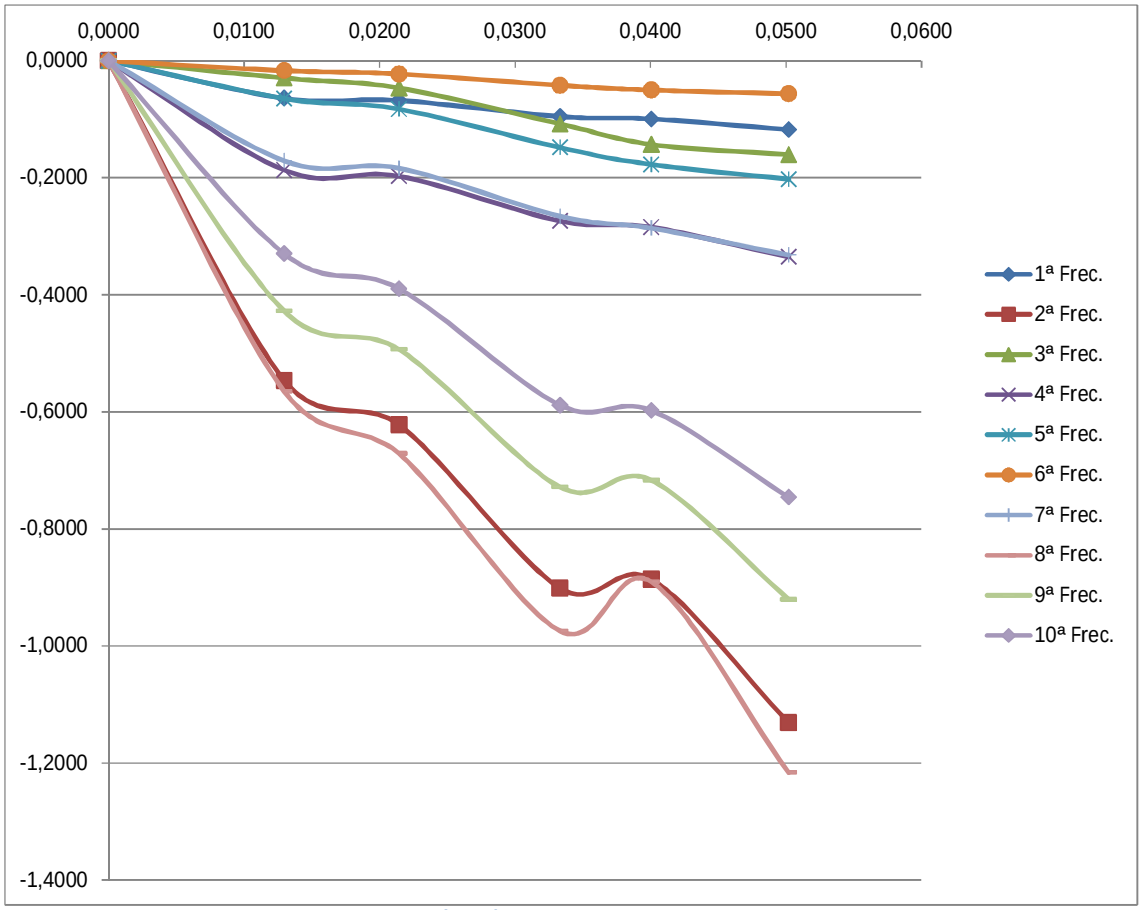


Fig. 9.3.5 Representación gráfica de las diferencias relativas observadas.

Como se acaba de observar en los datos obtenidos existe una limitación manifiesta para poder realizar un buen análisis y determinar la correlación entre las frecuencias naturales y el daño sufrido por la viga. La limitación no se debe a las rutinas del programa CFYFP sino a las existentes en el PLCD, debido a las limitaciones existentes en el tipo de modelo de daño utilizable para obtener el tensor secante en cada paso calculado.

El CFYFP ha calculado la variación de frecuencias y de formas propias para los valores proporcionados de manera correcta. Si se estudian las formas propias (figuras 9.3.6 a 9.3.15) y como varían se pueden observar dos cosas: para los niveles de daño estudiados, no se producen cambios apreciables en las formas y se intuye que las frecuencias más afectadas son las que sus formas sufren una mayor influencia de la zona dañada. Aunque para los niveles de daño global alcanzados no se puede establecer conclusiones.

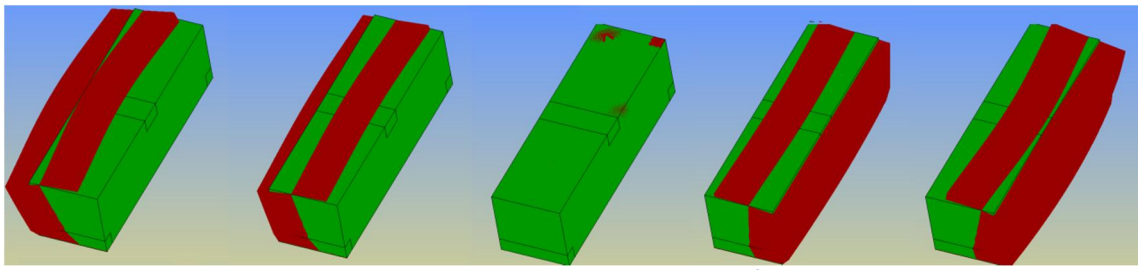


Fig. 9.3.6 Primera forma propia de la viga, en verde se ve la viga inmóvil y en granate el movimiento.

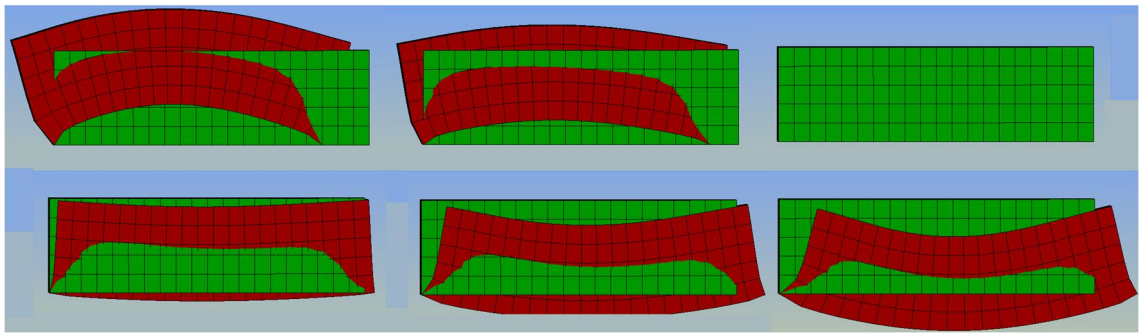


Fig. 9.3.7 Movimiento de la segunda forma propia, vista de la viga de forma lateral.

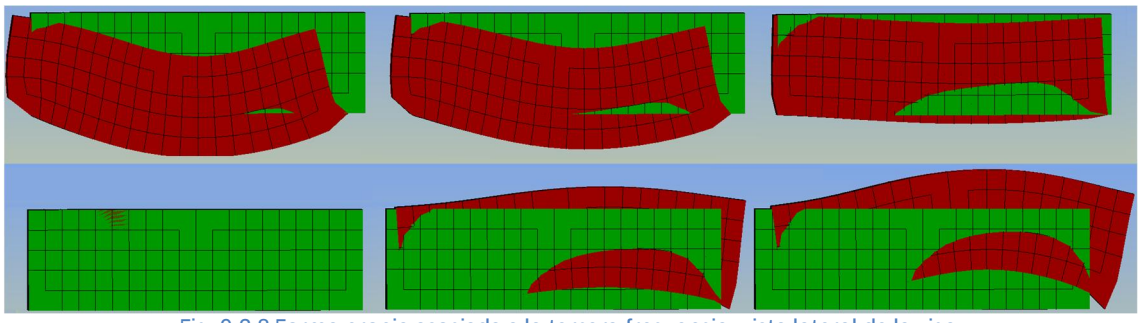


Fig. 9.3.8 Forma propia asociada a la tercera frecuencia, vista lateral de la viga.

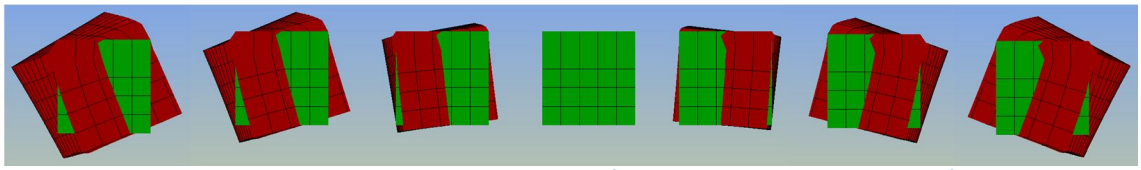


Fig. 9.3.9 Cuarta forma propia, vista en sección de la viga, se trata de una torsión.

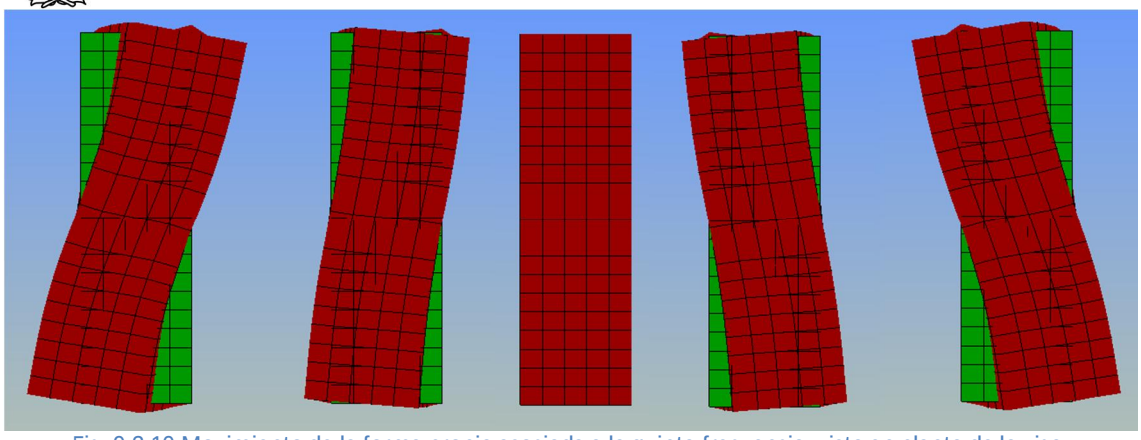


Fig. 9.3.10 Movimiento de la forma propia asociada a la quinta frecuencia, vista en planta de la viga.

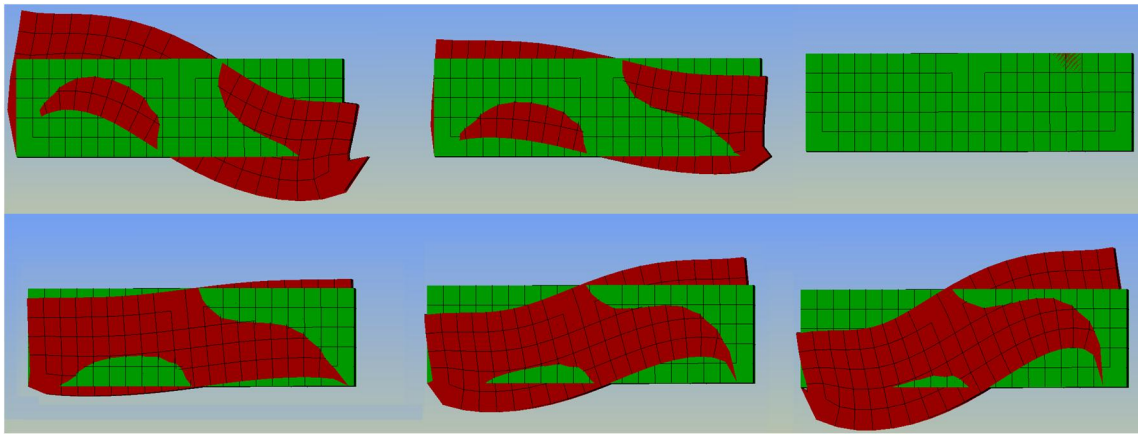


Fig. 9.3.11 Sexta forma propia de la viga biapoyada vista de forma lateral.

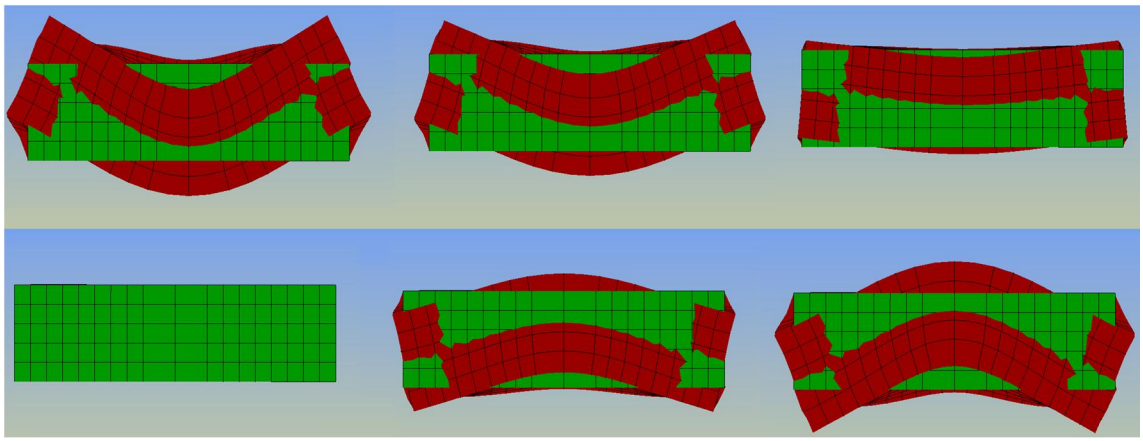


Fig.9.3.12 Vista en planta de la forma asociada a la séptima frecuencia.

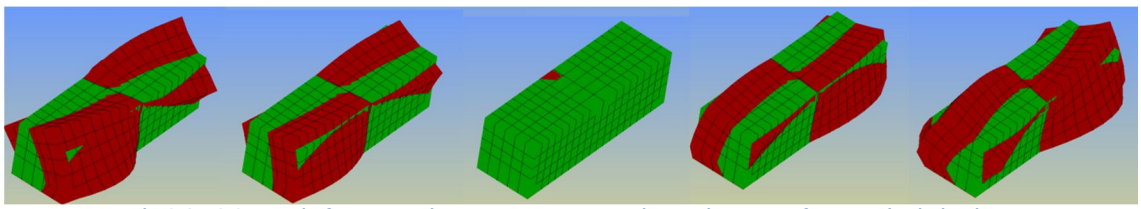


Fig.9.3.13 Segunda forma torsiva que se corresponde con la octava frecuencia de la viga.

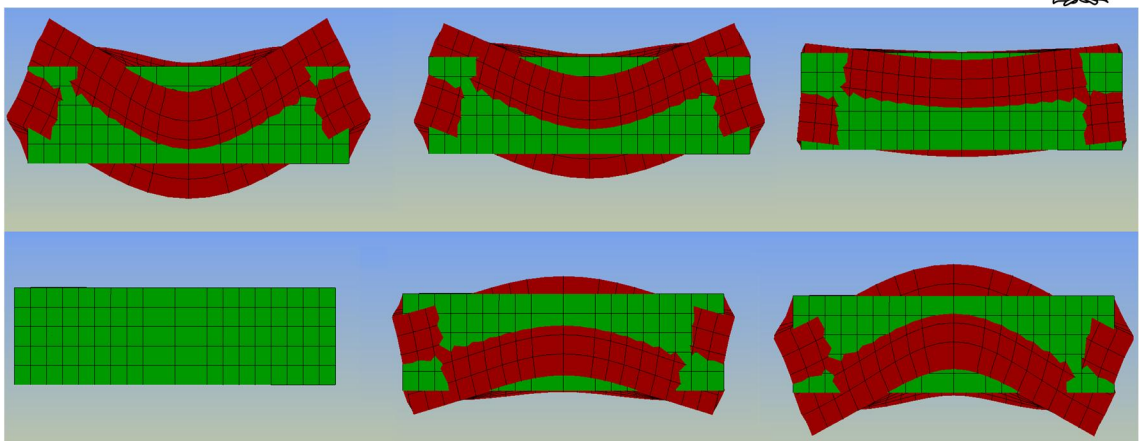


Fig. 9.3.14 Movimiento de la forma asociada a la novena frecuencia, con la viga vista de forma lateral, como se puede ver coincide con la séptima frecuencia pero en dirección perpendicular.

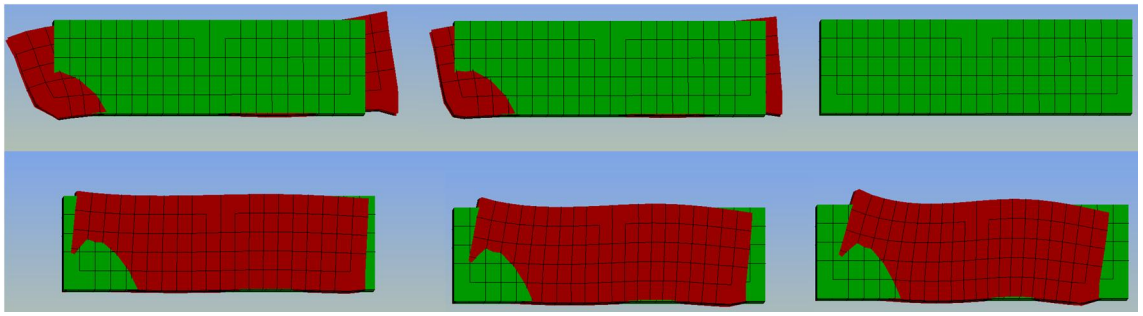


FIg. 9.3.15 Forma propia asociada a la décima frecuencia, viga vista en planta.

10. Análisis teórico de frecuencias

A fin de poder tener unos valores de referencia para poder comprobar las frecuencias y formas propias obtenidas en los casos de estudio, a continuación se realizará el análisis de forma manual utilizando el método energético de Rayleigh para el cálculo de la frecuencia no amortiguada de sistemas continuos.

Este método se basa en el principio de conservación de la energía, que se puede enunciar de la siguiente manera: si no actúa ninguna fuerza sobre el sistema, y no existe amortiguación, la energía total del sistema permanece constante durante el movimiento y, en consecuencia, su derivada respecto al tiempo es igual a cero. Dado que no se produce pérdida de energía, la energía cinética máxima ha de coincidir con la energía potencial máxima y la frecuencia natural puede obtenerse a partir de esta igualdad.

10.1 Columna empotrada

Para el caso de la columna empotrada que se ha estudiado, se puede de finir la energía potencial máxima de la siguiente manera:

$$V^{\max} = \frac{1}{2} K (u^{\max})^2 \quad ; \quad K = \frac{3EI}{L^3} \qquad \text{Ec. 10.1.1}$$

Donde E es el módulo de Young del material, I es la inercia de la sección transversal y L es la longitud de la columna. Por otro lado la energía cinética máxima para el caso de una columna empotrada puede expresarse como:

$$T_{max} = \frac{m}{2L} \int_0^L \left(\frac{3z^2 L - z^3}{2L^3} \omega u^{max} \right)^2 dz \qquad \text{Ec. 10.1.2}$$

Donde m es la masa, L es la longitud de la columna, z es la dimensión del eje longitudinal de la columna, ω es el valor propio y u es el desplazamiento de un punto de la columna.

Después de integrar la ecuación 10.1.2 e igualar el resultado con la expresión de la máxima energía potencial dada en la ecuación 10.1.1 resulta que la frecuencia natural en Hercios puede expresarse de la siguiente manera:

$$f = \frac{1}{2\pi}\sqrt{\frac{3EI}{L^3\left(\frac{1}{4}m\right)}} \qquad \text{Ec. 10.1.3}$$

Esta ecuación (10.1.3) solamente nos permite calcular la primera frecuencia natural para estructuras con un comportamiento lineal. No obstante, se puede aproximar el comportamiento dañado de la columna haciendo la siguiente suposición: se supondrá que la pérdida de rigidez de la estructura se deberá al cambio del módulo de elasticidad del hormigón y que la geometría de la estructura no cambiará al dañarse. De esta manera, el cambio en del módulo de Young al dañarse se expresará como:

$$E_d = E_o(1 - D) \qquad \text{Ec. 10.1.4}$$

Utilizando las ecuaciones presentadas previamente en esta sección, se puede calcular el valor de la primera frecuencia para los distintos estados de daño, entendiéndolo como un estado elástico con una elasticidad reducida, obteniendo los resultados presentados en la tabla 10.1.1. Hay que recordar que estos resultados no pretenden ser unos resultados exactos, sino una primera aproximación para poder valorar los resultados calculados con las rutinas implementadas. Además, debido a la suposición hecha para calcular las frecuencias dañadas, es de esperar que lo resultados se aparten más de la realidad cuanto mayor sea el daño sufrido por la columna.

Daño global	Ed	Frecuencia
0,0000	37000000000	21,4522
0,0035	36871151148	21,4148
0,0399	35524681569	21,0201
0,0914	33618734062	20,4485
0,1631	30965174326	19,6249
0,2326	28394067662	18,7925
0,2915	26213511578	18,0565
0,3648	23500874649	17,0967
0,4309	21055672025	16,1828
0,5137	17993440944	14,9599
0,5494	16672346705	14,4002
0,7767	8263641897	10,1381
0,9001	3696529108	6,7806
0,9384	2277705822	5,3225

Tabla 10.1.1 Cálculo aproximado de la frecuencia fundamental de la columna, para distintos estados de daño.

10.1.1 Comparación

Lo primero y más fácil de hacer es comparar directamente los valores obtenidos para la primera frecuencia utilizando el método de Rayleigh y los programas de elementos finitos (tabla 10.1.1.1).

Daño global	Rayleigh	Rutinas
0,0000	21,4522	22,1277
0,0035	21,4148	22,1198
0,0399	21,0201	21,9463
0,0914	20,4485	21,6450
0,1631	19,6249	21,2247
0,2326	18,7925	20,7242
0,2915	18,0565	20,2204
0,3648	17,0967	19,7240
0,4309	16,1828	19,2042
0,5137	14,9599	18,6961
0,5494	14,4002	18,2205
0,7767	10,1381	16,3074
0,9001	6,7806	13,9481
0,9384	5,3225	0,0000

Tabla 10.1.1.1 Comparación de los resultados obtenidos usando el método energético de Rayleigh y las rutinas implementadas.

Como se puede observar en la tabla 10.1.1.1 los valores obtenidos son más similares para valores de daño inferiores y dejan de ser del mismo orden de magnitud

para casos donde el daño global es muy elevado. Como ya se ha comentado que la utilidad del cálculo manual es confirmar los resultados obtenidos utilizando los nuevos programas, se puede concluir que en este caso los resultados obtenidos han sido correctos, teniendo en cuenta la aproximación realizada para calcular los valores dañados. A continuación se muestra la diferencia absoluta y relativa entre las frecuencias calculadas (tabla 10.1.1.2).

Daño global	Rayleigh	Rutinas	Dif. Abs	Dif. Rel. (%)
0,0000	21,4522	22,1277	0,6755	3,05
0,0035	21,4148	22,1198	0,7050	3,19
0,0399	21,0201	21,9463	0,9261	4,22
0,0914	20,4485	21,6450	1,1966	5,53
0,1631	19,6249	21,2247	1,5998	7,54
0,2326	18,7925	20,7242	1,9317	9,32
0,2915	18,0565	20,2204	2,1639	10,70
0,3648	17,0967	19,7240	2,6273	13,32
0,4309	16,1828	19,2042	3,0213	15,73
0,5137	14,9599	18,6961	3,7362	19,98
0,5494	14,4002	18,2205	3,8203	20,97
0,7767	10,1381	16,3074	6,1693	37,83
0,9001	6,7806	13,9481	7,1675	51,39
0,9384	5,3225	0,0000	-5,3225	100,00

Tabla 10.1.2 Diferencias entre la primera frecuencia calculada usando el método energético de Rayleigh y las rutinas implementadas.

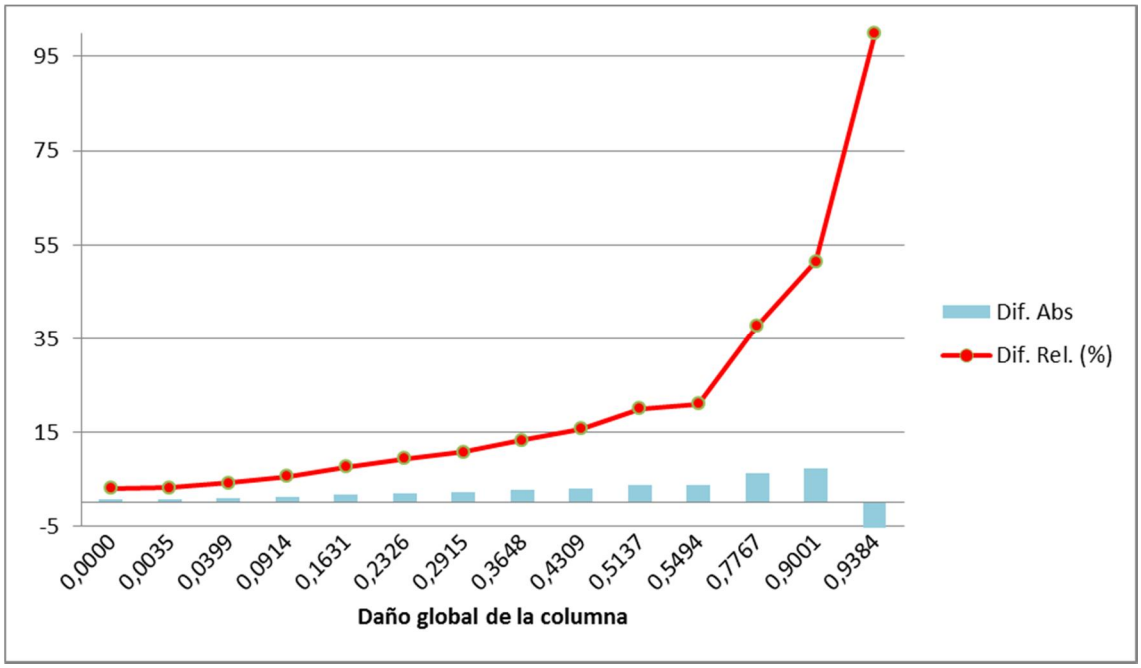


Fig. 10.1.1.3 Representación gráfica de las diferencias relativas y absolutas entre frecuencias calculadas de manera manual y con elementos finitos.

Como se puede observar en la figura 10.1.1.3 el error relativo se dispara cuando el valor del daño global supera el 50%, hecho esperado teniendo en cuenta las suposiciones realizadas para calcular las frecuencias usando el método de Rayleigh.

10.2 Columna con armadura

En el caso de la columna con armadura que se ha estudiado, las ecuaciones de energía potencial y de la energía cinemática son iguales, lo único que cambiaremos es el valor del módulo de Young del material, ya que en el caso anterior la estructura era de hormigón 100% y en este caso se ha de tener en cuenta el efecto de la armadura de acero. Para ello se ha calculado la proporción de acero y de hormigón en la estructura y, en base a ello, se ha determinado que el módulo de Young de la estructura será:

$$E_T = 0.976 \cdot E_{Hormigón} + 0.024 \cdot E_{Acero} \qquad \text{Ec. 10.2.1}$$

Utilizando las ecuaciones presentadas en la sección anterior 10.1 y la pequeña consideración de la ecuación 10.2.1, se puede calcular el valor de la primera frecuencia para los distintos estados de daño, obteniendo los resultados presentados en la tabla 10.2.1. Hay que tener en cuenta que si en el primer caso estos resultados no pretendían ser exactos, sino una primera aproximación para poder valorar los resultados calculados con las rutinas implementadas, en este caso se han de ver con mayor cautela debido a la forma de aproximar el módulo de Young empleado.

Daño global	Et	Frecuencia
0,0000	36116800000	21,1946
0,0115	35702245558	21,0726
0,0452	34483677696	20,7099
0,0993	32530876844	20,1149
0,1569	30450762571	19,4612
0,2162	28308733130	18,7642
0,2619	26656542182	18,2084
0,3143	24766771625	17,5511
0,3547	23305420641	17,0255
0,3994	21693251636	16,4260
0,4414	20174536631	15,8406
0,4989	18099915770	15,0041
0,5423	16530278487	14,3387
0,6009	14415005899	13,3899
0,6421	12925713160	12,6794
0,7024	10748527854	11,5623

Tabla 10.2.1 Valores calculados para la columna armada utilizando el método de Rayleigh.

10.2.1 Comparación

Si comparamos los resultados con los obtenidos con las rutinas (tabla 10.2.1.1) se puede observar una serie de cosas: el cálculo teórico es inferior al de las rutinas en todos los casos, al contrario de lo que sucedía cuando no había armadura (tabla 10.1.1.1). La diferencia entre el valor obtenido usando el método energético de Rayleigh y el computado son a medida que se dañan más diferentes. Esto último ya había sido observado con anterioridad (tabla 10.1.1.1), pero en este caso la diferencia es bastante más grande desde el primer momento (6.58% en comparación con 3.05% sin armadura) y sigue creciendo, aunque para valores últimos las diferencias entre el valor teórico y el calculado en el caso sin armadura y el que la tiene son bastante similares.

Daño global	Rayleigh	Rutinas
0,0000	21,1946	22,5886
0,0115	21,0726	22,5789
0,0452	20,7099	22,3863
0,0993	20,1149	22,0465
0,1569	19,4612	21,6195
0,2162	18,7642	21,1125
0,2619	18,2084	20,6299
0,3143	17,5511	20,1359
0,3547	17,0255	19,6699
0,3994	16,4260	19,2077
0,4414	15,8406	18,7553
0,4989	15,0041	17,9423
0,5423	14,3387	17,2042
0,6009	13,3899	16,2535
0,6421	12,6794	15,4122
0,7024	11,5623	14,2408

Tabla 10.2.1.1 Comparación de los valores obtenidos teóricamente y con las rutinas.

Daño global	Rayleigh	Rutinas	Dif. Abs	Dif. Rel. (%)
0,0000	22,5886	21,1946	-1,3940	-6,58
0,0115	22,5789	21,0726	-1,5063	-7,15
0,0452	22,3863	20,7099	-1,6764	-8,09
0,0993	22,0465	20,1149	-1,9316	-9,60
0,1569	21,6195	19,4612	-2,1583	-11,09
0,2162	21,1125	18,7642	-2,3483	-12,51
0,2619	20,6299	18,2084	-2,4215	-13,30
0,3143	20,1359	17,5511	-2,5848	-14,73
0,3547	19,6699	17,0255	-2,6445	-15,53
0,3994	19,2077	16,4260	-2,7816	-16,93

Tabla 10.2.2a Comparación de los valores obtenidos. Sigue en la siguiente página.

Daño global	Rayleigh	Rutinas	Dif. Abs	Dif. Rel. (%)
0,4414	18,7553	15,8406	-2,9147	-18,40
0,4989	17,9423	15,0041	-2,9382	-19,58
0,5423	17,2042	14,3387	-2,8655	-19,98
0,6009	16,2535	13,3899	-2,8636	-21,39
0,6421	15,4122	12,6794	-2,7328	-21,55
0,7024	14,2408	11,5623	-2,6785	-23,17

Tabla 10.2.1.2b Continuación de la tabla 10.2.1a.

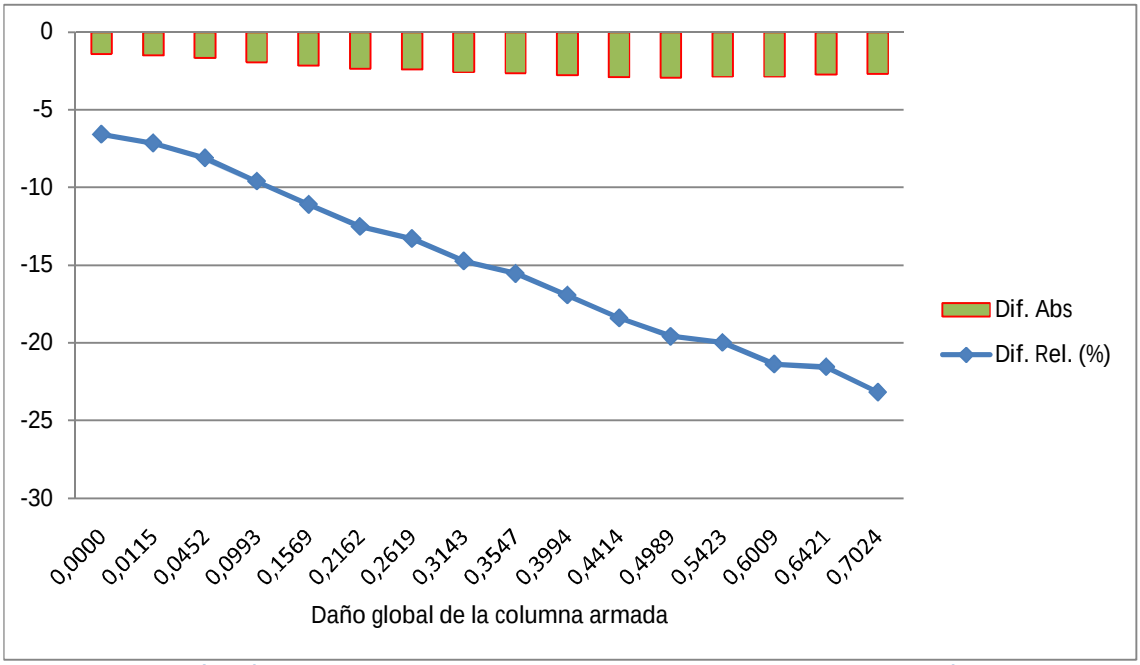


Fig. 10.2.3 Evolución gráfica de la diferencia de resultados para la primera frecuencia entre métodos utilizados.

En la figura 10.2.1.3 se puede observar que la diferencia relativa entre valores crece de forma casi constante y no tenemos un salto brusco como se observaba en la figura 10.1.1.3.

10.3 Viga apoyada

Para el caso estudiado de la viga apoyada, la energía potencial puede ser definida de la siguiente manera:

$$V^{max} = \frac{1}{2}\int_0^L EI\left(\frac{d^2y}{dx^2}\right)^2 dx \qquad \text{Ec. 10.3.1}$$

Donde E es el módulo de Young del material, I es la inercia de la sección transversal. Por otro lado la energía cinética máxima para el caso de una viga apoyada de longitud igual a la unidad puede expresarse como:

$$T_{max} = \frac{m \cdot \omega^2}{2L} \int_0^1 (u^{max}(z))^2 dz \qquad \text{Ec. 10.3.2}$$

Después de integrar la ecuación 10.3.2 e igualar el resultado con la expresión de la máxima energía potencial dada en la ecuación 10.3.1 resulta que la frecuencia natural en Hercios puede expresarse de la siguiente manera:

$$f = \sqrt{\frac{97.4 \cdot EI}{(\rho S^2 l^3)}} \qquad \text{Ec. 10.3.3}$$

Esta ecuación (10.3.3), al igual que lo que sucedía con la de la columna (ec. 10.1.3) nos permite calcular la primera frecuencia natural para estructuras con un comportamiento lineal pero no los casos dañados. Como se ha visto en el ejemplo anterior, se puede aproximar el comportamiento dañado de la columna haciendo la siguiente suposición: se supondrá que la pérdida de rigidez de la estructura se deberá al cambio del módulo de elasticidad del hormigón y que la geometría de la estructura no cambiará al dañarse. De esta manera, el cambio en del módulo de Young al dañarse se expresará de la manera mostrada en la ecuación 10.1.4.

Utilizando las ecuaciones presentadas previamente en esta sección, se puede calcular el valor de la primera frecuencia para los distintos estados de daño, obteniendo los resultados presentados en la tabla 10.3.1. Hay que recordar que estos resultados no pretenden ser unos resultados exactos, sino una primera aproximación para poder valorar los resultados calculados con las rutinas implementadas. Además, debido a la suposición hecha para calcular las frecuencias dañadas, es de esperar que lo resultados se aparten más de la realidad cuanto mayor sea el daño sufrido por la viga.

Daño global	Ed	Frecuencia
0,0000	37000000000	340,0368
0,0130	36993771524	340,0081
0,0214	36321810835	336,9060
0,0333	35937835957	335,1205
0,0401	35545338447	333,2854
0,0502	35150974395	331,4314

Tabla 10.3.1 Evolución de las frecuencias calculadas por el método de Rayleigh para distintos estados de daño.

10.3.1 Comparación

Si se comparan los resultados obtenidos con las rutinas con los calculados de forma manual para los pocos estados de daño que se han podido obtener con los programas (tabla 10.3.1.1), se observa que existe una diferencia inicial importante, o por lo menos, más grande que la observada en los anteriores ejemplos. Esto puede deberse al hecho de que en el modelo utilizado hay una serie de elementos que no se dañan como se ha visto anteriormente, porque al realizarse el análisis con todos los elementos de hormigón el valor obtenido para la primera frecuencia no dañada fue de 341 Hz, valor muy parecido al obtenido usando el método energético. Pero debido a que el daño no se localizaba en el lugar adecuado se ha tenido que utilizar la primera opción para el análisis. También puede verse que mientras que el cálculo manual va disminuyendo el valor de la frecuencia, en las rutinas lo hace de manera casi inapreciable.

Daño	Rayleigh	Rutinas
0,0000	340,0368	333,2476
0,0130	340,0081	333,0344
0,0214	336,9060	333,0218
0,0333	335,1205	332,9301
0,0401	333,2854	332,9164
0,0502	331,4314	332,8556

Tabla 10.3.2 Comparación de los dos métodos de cálculo utilizados.

A pesar de todo si se observa la figura 10.3.3 se verá que la diferencia relativa para los pocos valores comparados es como máximo de un 2% que no es un valor excesivamente elevado, teniendo en cuenta las diferencia entre los tipos de cálculo. Lo que es más, las diferencias relativas observadas son las menos de todos los ejemplos vistos hasta el momento.

Daño	Rayleigh	Rutinas	Dif. Abs.	Dif. Rel (%)
0,0000	340,0368	333,2476	6,7891	1,9966
0,0130	340,0081	333,0344	6,9738	2,0511
0,0214	336,9060	333,0218	3,8842	1,1529
0,0333	335,1205	332,9301	2,1904	0,6536
0,0401	333,2854	332,9164	0,3690	0,1107
0,0502	331,4314	332,8556	-1,4242	-0,4297

Tabla 10.3.3 Diferencias observadas entre métodos de cálculo.

11. Conclusiones

Del trabajo realizado y de los casos estudiados se pueden extraer varías conclusiones, las más importantes sobre el tema de la variación de las frecuencias naturales de las estructuras cuando éstas se dañan, también se extraen conclusiones sobre el trabajo realizado con las rutinas programadas y el nuevo programa desarrollado. En cuanto al tema principal de la tesina se han extraído las siguientes conclusiones:

- Al igual que lo observado en los trabajos previos realizados por otros investigadores (ver referencias [1] - [9]) se ha podido observar que todas las frecuencias naturales de las estructuras disminuyen cuando se incrementa el daño.
- En términos absolutos estos descensos han sido mayores cuanto más alta ha sido la frecuencia observada, sin embargo estudiando los descensos relativos ha podido observarse que los descensos son muy similares en todas las frecuencias.
- El hecho de que se encuentren en una horquilla bastante estrecha todas las diferencias relativas de las frecuencias, ha permitido determinar una relación entre las frecuencias y el daño.
- Esta relación será diferente para cada estructura pero puede determinarse con los programas de ordenador comentados y un postprocesado simple. Una vez realizada esta correlación, permitirá medir la frecuencia de una estructura real en un determinado momento y conociendo el valor que tenía en un estado “nuevo”, determinar el nivel de daño global que tiene en ese momento. Se ha visto que una relación cuadrática se ajusta de manera muy adecuada hasta para valores de daño global relativamente altos.
- El estudio de las formas propias de las frecuencias y de su variación al dañarse, permite si bien no localizar exactamente la zona dañada si hacerse una idea de su localización, auxiliando este diagnóstico con un sistema informático de análisis no lineal, como el presentado en este

trabajo. Se ha observado que las formas propias que tenían un movimiento importante en la zona dañada presentaban dos efectos a lo largo del proceso de daño: el descenso relativo de la frecuencia asociada era más importante que en otros casos y se producían variaciones importantes en la forma de vibración en la zona dañada.

- Se ha comprobado de que este método, que permite determinar el valor de daño global a partir de las variaciones de las frecuencias naturales, permite apreciar un importante cambio en las frecuencias naturales para valores aproximados de daño global del 3%.

Sobre el nuevo programa desarrollado para calcular las frecuencias y formas propias de las estructuras, CFYFP, también han podido concluir que el programa ha funcionado correctamente en 3D y sus limitaciones en este campo vienen impuestas por las propias del programa de elementos finitos PLCD y sus limitaciones para obtener solución en algunos problemas no lineales de daño o para problemas de plasticidad o viscoplasticidad.

12. Propuestas de mejora y futuras líneas de investigación

De los posibles cambios que se presentarán a continuación en esta sección hay de dos clases, los que sirven para mejorar el funcionamiento de las rutinas programadas (CFYFP) y los que indican futuros pasos que habría que seguir en la investigación de la evolución de las frecuencias al dañarse una estructura.

Se ha visto que el almacenamiento utilizado es el llamado *Sparse*, que conlleva un gran ahorro de espacio al no incluir los valores nulos de las matrices en el almacenamiento. Sin embargo, se está almacenando toda la matriz de rigidez, cosa que no es lo más eficaz ya que la matriz es simétrica y con almacenar la matriz triangular superior o inferior sería suficiente, este cambio comportaría un aumento significativo de memoria necesaria y de velocidad de ejecución.

Sería recomendable realizar una optimización completa de todas las rutinas utilizadas en el programa, para mejorar el uso de la memoria y los tiempos de ejecución.

Las pruebas y validaciones realizadas se han llevado a cabo siempre utilizando problemas en pequeñas deformaciones, por eso, y aunque las instrucciones para funcionar en grandes deformaciones han sido implementadas, deberían ser validadas con ejemplos concretos antes de empezar a resolver problemas que involucren grandes deformaciones.

Por el momento, el único modelo de daño utilizable es el isótropo de Kachanov, esta limitación viene impuesta por las rutinas del PLCD, no por las capacidades del CFYFP. Adecuar las rutinas del PLCD para habilitar el cálculo del tensor secante en todos los modelos de daño que tiene implementados permitiría utilizarlos al realizar el análisis de la evolución de frecuencias al deteriorarse la estructura. En el mismo sentido que esta última observación, debería estudiarse los cambios a realizar en las rutinas empleadas por el PLCD para poder estudiar problemas de viscoplasticidad y plasticidad.

Por último, sería muy recomendable realizar el estudio de campo de una estructura real que permita corroborar los datos y conclusiones en el presente estudio.

13. Bibliografía

13.1 DOCUMENTOS CITADOS EN LA TESINA

[1] M. A. Mannan and M. H. Richardson, "Detection and location of structural cracks using FRF measurements," presented at the Proceedings of the 9th IMAC, Florencia, Italia, 1991.

[2] M. R. Chowdhury, "Variation in the modal and system parameters of steel plates due to changes in physical properties," presented at the Proceedings of the 9th IMAC, Florencia, Italia, 1991.

[3] C. H. J. Fox, "The location of defects in structures: a comparison of the use of natural frequencies and mode shape data," presented at the Proceedings of the 10th IMAC, San Diego, California, 1992.

[4] T. Y. Kam and T. Y. Lee, "Detection of cracks in structures using modal test data," *Engineering Fracture Mechanics*, vol. 42, pp. 381-387, 1992.

[5] A. K. Pandey and M. Biswas, "Damage Detection in Structures Using Changes in Flexibility," *Journal of Sound and Vibration*, vol. 169, pp. 3-17, 1994.

[6] Y. Narkis, "Identification of Crack Location in Vibrating Simply Supported Beams," *Journal of Sound and Vibration*, vol. 172, pp. 549-558, 1994.

[7] D. E. V. Torres, L. E. Suárez, and R. R. López, "Identificación de daños en vigas de hormigón experimentales y analíticas utilizando metodologías modales," *Rev. Int. de Desastres Naturales, Accidentes e Infraestructura Civil*, vol. 4, pp. 183-200, 2004.

[8] J. M. Pérez, "Detection and location of damage using modal properties," PHD, Tesis de Maestría en Ciencias, Universidad de Puerto Rico, Mayagüez, 1994.

[9] C. Oyarzo-Vera and N. Chouw, "Ensayo modal experimental de una casa de alabañilería no reforzada," presented at the Congreso chileno de sismología e ingeniería antisísmica. X Jornadas, Santiago de Chile, 2010.

[10] S. Oller, *Dinámica no lineal*. Barcelona (España): CIMNE – Ediciones UPC, 2001.

[11] S. Oller, B. Luccioni, and A. H. Barbat, "Un método de evaluación del daño sísmico en estructuras de hormigón armado," *Revista internacional de métodos numéricos para cálculo y diseño en ingeniería*, vol. 12, pp. 215-238, 1996.

[12] L. M. Kachanov, "Time of the rupture process under creep conditions," *Izv. Akad. Nauk. SSSR, Otd. Tekhn. Nauk*, pp. 26-31, 1958.

[13] D. Krajcinovic and U. Fonseca, "The Continuous Damage Theory of Brittle Materials, Part I and II.," *Journal of Applied Mechanics*, pp. 809-824, 1981.

[14] J. Lemaitre and J. L. Chaboche, "Aspects phénoménologiques de la ruptura par endommagement," *Mècanique Appl.*, pp. 317-365, 1978.

[15] J. C. Vielma, A. H. Barbat, and S. Oller, "An objective seismic damage index to evaluate RC structures," *Elsevier*, 2008.

[16] J. C. Simo and J. W. Ju, "Strain- and stress-based continuum damage models--I. Formulation," *International Journal of Solids and Structures*, vol. 23, pp. 821-840, 1987.

[17] K.-J. Bathe, *Finite Element Procedures*. Nueva Jersey: Prentice Hall, 1996.

[18] M. Metcalf and J. Reid, *Fortran 90/95 Explained*: Oxford, 1996.

[19] C. A. Felippa, *Introduction to finite element methods*. Boulder: University of Colorado, 2004.

[20] S. Oller, B. Luccioni, A. Hanganu, E. Car, O. Salomon, L. Neamtu, F. Zalamea, P. Mata, X. Martínez, M. Molina, J. A. Paredes, and F. Otero, *Breve Manual de PLCD*, 2010.

[21] A. H. Barbat and S. Oller, *Conceptos de cálculo dinámico en las normativas de diseño sismorresistente. Aplicación a la normativa NCSE-02*: CIMNE, 1997.

[22] A. H. Barbat, S. Oller, and J. C. Vielma, *Cálculo y diseño sismorresistente de edificios. Aplicación de la norma NCSE-02*: CINME, 2006.

13.2 BIBLIOGRAFÍA COMPLEMETARIA

D. Baguley and D. R. Hose, How to interpret finite element results: NAFEMS, 1997.

E. Blanco, S. Oller, and L. Gil, "Análisis experimental de estructuras," CINME, 2007.

N. Bicanic, "Course notes of Structural Analysis 4," ed. Glasgow: University of Glasgow, 2010.

M. Cervera and E. Blanco, Mecánica de estructuras: Vol. 1 y Vol. 2. Barcelona: UPC, 2002.

C. A. Felippa, Introduction to finite element methods. Boulder: University of Colorado, 2004.

A. Miravete, Materiales compuestos, volumen I. Barcelona: Reverté, 2003.

J. Oliver, M. Cervera, S. Oller, and J. Lubliner, "Isotropoic damage models and smeared crack analysis of concrete.," presented at the Second International Conference on computer aided analysis and design of concrete structures, Austria, 1990.

S. Oller, "Un modelo de daño continúo para materiales friccionales," PHD, UPC, Barcelona, 1988.

S. Oller, B. Luccioni, and A. H. Barbat, "Un método de evaluación del daño sísmico en estructuras de hormigón armado," Revista internacional de métodos numéricos para cálculo y diseño en ingeniería, vol. 12, pp. 215-238, 1996.

M. Paz, Dinámica estructural, teoría y cálculo. Barcelona: Reverté, 1992.

C. J. Pearce, "Course notes of Applied engineering mechanics 4," ed. Glasgow: University of Glasgow, 2009.

L. Pelà, "Continuun damage model for nonlinear analysis of masonry structures" PHD, UPC-UNIFE, Barcelona, 2009.

S. Rajendran and G. Prathap, "Convergence of eigenvalues of a cantilever beam with 8- and 20- node hexahedral elements," Journal of Sound and Vibration, vol. 227, pp. 667-681, 1999.

J. C. Vielma, "Caracterización del comportamiento sísmico de edificios de hormigón armado mediante la respuesta no lienal," PHD, UPC, Barcelona, 2008.

O. C. Zienkiewicz and R. L. Taylor, El método de los elementos finitos, volumen 2: mecánica de sólidos y fluidos. Dinámica y no linealidad. Barcelona: CINME-McGraw-Hill, 1995.

Anejo I: Códigos

CÓDIGOS MODIFICADOS (PLCD) (Cambios señalados en negrita)

```
      SUBROUTINE INPU3_V3D(TITLE)

C-----------------------------------------------------------------------
C
C               THIS SUBROUTINE READ THE FIXED VALUES
C
C-----------------------------------------------------------------------

      USE ENTEROS; USE MATRICES;  USE PARAM3D;

          IMPLICIT NONE;

          INTEGER IVFIX,IFPRE,IDOFN,NLOCA,NGASH,IFDOF,NVOLD,IEVAB,INODE,
     .       IPOIN,NGRAD,IVINT,IFINT,ITOTV, IVOLD, OLD_NVF

      REAL(KIND=8) VAL1,VAL2

          REAL(kind=8) :: OLD_TRM(NPOIN,MDOFN),
     .               OLD_TRT(NPOIN,MDOFN),
     .                           OLD_NFX(NPOIN)

          CHARACTER TITLE*72,
     .         TEXT*20

!         ----------------------- INITIALIZE SOME VARIABLES ------------------
      OLD_TRM(:,:)    = 0.0d0;                        OLD_TRT(:,:)          = 0.0d0
          OLD_NFX(:)           = 0.0d0;                        OLD_NVF                       = NVFIX

C     ----------------------- READ THE FIRST DATA CARD -------------------
      READ(1,*) NVFIX,NVINT

!     ----------------------- SAVE PREVIOUS REACTION VALUES --------------
          if ( ICARG .gt. 1)  then
                          OLD_TRM(1:OLD_NVF,:)  =       TREACM(:,:)
                          OLD_TRT(1:OLD_NVF,:)  =       TREACT(:,:)
                          OLD_NFX(1:OLD_NVF)            = NOFIXM(:)
          endif

!         ALLOCA (Y DESALLOCA si es preciso) LOS VECTORES DE RESTRICCION DE NODOS Y REACCIONES

          IF (ICARG .GT. 1) THEN
            DEALLOCATE(NOFIXM,          NOFIXT,               NOINT,                IFFIXM,
     .          IFLIB,          FIXEMM,               TREACM,               TREACT )
          ENDIF

          ALLOCATE(  NOFIXM(NVFIX),                         NOFIXT(NVFIX),     NOINT(NVINT),
     .                                IFFIXM(NTOTV,2),                  IFLIB(NTOTV),FIXEMM(NTOTV),
     .                                TREACM(NVFIX,MDOFN),          TREACT(NVFIX,MDOFN) );

          IFFIXM(:,:) = 0;                  IFLIB(:)              = 0;                  FIXEMM(:) = 0;
          NOINT(:)    = 0;                  NOFIXM(:)             = 0;                  NOFIXT(:) = 0;
          TREACM(:,:) = 0.0;          TREACT(:,:)  = 0.0;

      IF(NVFIX.GT.NPOIN*MDOFN) THEN

         CPARA= '  >>>> DETENIDO EN LA RUTINA INPU2 '

         MENSA=
     .'*** DIAG. DE INPU2, NUMERO DE RESTRICCIONES > NPOIN*MDOFN'

         VAL1=DFLOAT(NVFIXM)

         VAL2=DFLOAT(NPOIN*MDOFN)

         CALL CHECK4_V3D(VAL1,VAL2)

      END IF

C     --- HACE POSITIVO LOS SIGNOS DEL VECTOR LNODS PARA EL PROXIMO PASO POR EL SOLVER  ---


      DO IELEM=1,NELEM

        CALL BASE5O_V3D(IELEM)

        LNODS(1:NNODE)=IABS(LNODS(1:NNOD E))
```

```
          CALL BASE5I_V3D(IELEM)

      ENDDO

C     ---------------------------- READ THE FIXED VALUES. ------------------------
      WRITE(2,907)

      DO IVFIX=1,NVFIX

        READ(1,*) NOFIXM(IVFIX),IFPRE,(PRESC(IDOFN),IDOFN=1,NDOFN)

        WRITE(2,908) NOFIXM(IVFIX),IFPRE,(PRESC(IDOFN),IDOFN=1,NDOFN)

           WRITE(61,908) NOFIXM(IVFIX),IFPRE,(PRESC(IDOFN),IDOFN=1,NDOFN)   !jbll 01/04/2011

        CALL BASE6O_V3D(LPNTN(NOFIXM(IVFIX)))

        NLOCA=NGLIBA-NGLIBL

        IFDOF=10**(NDOFN-1)

        DO IDOFN=1,NDOFN

          NGASH=NLOCA+IDOFN

          FIXEMM(NGASH)=PRESC(IDOFN)

          IF (IFPRE.LT.IFDOF) THEN

                    IFDOF=IFDOF/10

                ELSE

                      IFFIXM(NGASH,1)=1

            IFPRE=IFPRE-IFDOF

            IFDOF=IFDOF/10

                ENDIF

        ENDDO

          ENDDO

C       -------- GUARDA REACCIONES ANTIGUAS EN NUEVO VECTOR DE REACCIONES ----------

        IF ( ICARG .gt. 1) THEN
                    DO ivfix = 1, NVFIX

                          OLD_LOOP: DO ivold = 1, OLD_NVF
                                if (NOFIXM(ivfix) .eq. OLD_NFX(ivold)) then
                                      TREACM(ivfix,:) = OLD_TRM(ivold,:)
                                      TREACT(ivfix,:)       =       OLD_TRT(ivold,:)
                                      cycle OLD_LOOP
                                endif
                          ENDDO OLD_LOOP

                    ENDDO
          ENDIF

C     -------- INICIALIZA EL VECTOR DE FIJACION Y DE CARGAS IMPUESTAS CUANDO ---------
C           CAMBIA LA CONDICION DE IMPOSICION:  DESP. IMPUESTO --> LIBRE
C                                       CARG. IMPUESTA --> LIBRE
C                                       O LOS CASOS INVERSOS

      IF(ICARG.EQ.1)THEN

        IFFIXM(1:NTOTV,2)=IFFIXM(1:NTOTV,1)

        DO IELEM=1,NELEM

          CALL BASE5O_V3D(IELEM)

          DO IEVAB=1,NEVAB

            RLOAD(IEVAB,2)=RLOA  D(IEVAB,1)

          ENDDO
```

```
        CALL BASE5I_V3D(IELEM)
      ENDDO
    ELSE
      DO IELEM=1,NELEM
        CALL BASE5O_V3D(IELEM)
        DO INODE=1,NNODE
          IPOIN=IABS(LNODS(INODE))
          CALL BASE6O_V3D(IPOIN)
          DO IDOFN=1,NDOFN
            NGRAD=NGLIBA-NGLIBL+IDOFN
            IEVAB=(INODE-1)*NDOFN+IDOFN

C     ---------- LIBERACION DE DESPLAZAMIENTO ----------

            IF(KCARG.EQ.0)THEN
              IF(IFFIXM(NGRAD,1).NE.IFFIXM(NGRAD  ,2))THEN
                IF(IFFIXM(NGRAD,2).EQ.1) TLOAD(IEVAB)=0.0
              ENDIF
            ENDIF
            IFFIXM(NGRAD,2)=IFFIXM(NGRAD,1)

C     ---------- LIBERACION DE CARGA ----------

              IF(KCARG.EQ.0)THEN
                IF(RLOAD(IEVAB,1).NE.RLOAD(IEVAB,2))TLOAD(IEVAB)=0.0
              ENDIF
              RLOAD(IEVAB,2)=RLOAD(IEVAB,1)
          ENDDO
        ENDDO
        CALL BASE5I_V3 D(IELEM)
      ENDDO
    ENDIF
C     ---------- READ THE FREE DEGREE VALUES. ----------
    IF(NVINT.NE.0)THEN
        WRITE(2,912)
        DO IVINT=1,NVINT
      READ(1,*) NOINT(IVINT),IFINT
      WRITE(2,908) NOINT(IVINT),IFINT
      CALL BASE6O_V3D(LPNTN(NOINT(IVINT)))
      NLOCA=NGLIBA-NGLIBL
      IFDOF=10**(NDOFN-1)
      DO IDOFN=1,NDOFN
        NGASH=NLOCA+IDOFN
        IF (IFINT.LT.IFDOF) THEN
```

```
                        IFDOF=IFDOF/10
                      ELSE
              IFLIB(NGASH)=1
              IFINT=IFINT-IFDOF
              IFDOF=IFDOF/10
                      ENDIF
          ENDDO
         ENDDO
      ENDIF
C     ------------> ESCRITURA PARA POSTPROCESO.
      TEXT=TITLE
!         WRITE(20)(IFFIXM(ITOTV,1),ITOTV=1,NTOTV) !Eliminado por XMG Abril 2006

C     ------------> CONTROLA SI LOS VALORES FIJOS SON CORRECTOS.
      CALL CHECK3_V3D
          CALL INPU2T_V3D
      RETURN
C     ------------> LISTA DE FORM ATS
 907  FORMAT(//,10X,
     .   'ECHO OF FIXITY DATA FOLLOWS :',/,
     .10X,'============================',//,
     .   5H NODE,6X,4HCODE,8X,12HFIXED VALUES)
 908  FORMAT(1X,I4,5X,I5,5X,5F10.6)
 912  FORMAT(//5H NODE,6X,10HCOD. FREE )
      END
      SUBROUTINE INPUV3D
C     ------------------------------------------------------------------------
C
C     ESTA SUBRUTINA LEE LA ENTRADA DE SOLIDOS 3D.
C
C                   PMA, CIMNE, UPC, 2004.
C
C     ------------------------------------------------------------------------
      USE ENTEROS; USE MATRICES;  USE PARAM3D;
          IMPLICIT NONE;
      INTEGER IPOIN,IK,ITEMP,NRO,NMC,IMATS,MP,INODE,NNTER,NINTER,
     .        IP,IDIME,NUMAT,IPROP, ilay, rad, ampli, perio
          REAL(KIND=8) PINUM,EMAXI,VAL1,VA L2,P
          REAL(KIND=8), DIMENSION(16) :: AUXDATOS

          WRITE(61,*)TYPEP
          WRITE(61,*)NPOIN,NELEM,NMATC,LARGE
C     ---------------------------- FIN DECLARACIONES ---------------------------
          CALL OTABR01_V3D;
      CALL ALOKA_V3D;

      IF (TITLE1(1:6).NE.'START2') THEN ! MMH 19/05/2008 PARA CONTROL DE IMPRESION EN EL REINICIO
      WRITE(2,906); WRITE(2,902)
          ENDIF   ! RESTART
```

```
      PINUM=DACOS(-1.0D00)

  LPNTN(1:NPOIN)=[1:NPOIN]; LPNINV(1:NPOIN)=[1:NPOIN];

      DO IELEM=1,NELEM

        LNODS(:)=0

        READ(1,*)    NUMEL,
     .          (MATNO(IK),IK=1,(MINDE-1)), ! ACA MATNO(1)=ISEC,Nº DE SECCION.
     .          VAREL(1),
     .          NTYPE,
     .          EFASE,
     .          NNODE,
     .          NGAUS,
     .          NINTE,
     .          (LNODS(INODE),INODE=1,NNODE),
     .          (EAEULER(ik),ik=1,3)

        IF  (TITLE1(1:6).NE.'START2') THEN ! MMH 19/05/2008 PARA CONTROL DE IMPRESION EN EL REINICIO

        WRITE(2,903) NUMEL,
     .          MATNO(1),
     .          VAREL(1),
     .          NTYPE,
     .          EFASE,
     .          NNODE,
     .          NGAUS,
     .          NINTE,
     .          (LNODS(INODE),INODE=1,NNODE),
     .                      (EAEULER(ik),ik=1,3)

      !Sacar el número de nodos por elemento para distinguir entre hexaedros y tetraedros JBLL 03/06/11

            IF(IELEM.EQ.1)THEN
            WRITE(61,*)NNODE
            ENDIF

            WRITE(61,908) NUMEL,
     .          MATNO(1),
     .          NTYPE,
     .          EFASE,
     .          NNODE,
     .          NGAUS,
     .          NINTE,
     .          (LNODS(INODE),INODE=1,NNODE),
     .                  (EAEULER(ik),ik=1,3)

        ENDIF   ! RESTART

    IF (IELEM.EQ.1) THEN

          ITEMP=NTYPE

        ELSE

      IF(ITEMP.EQ.NTYPE) THEN

        IEFLAG=1 ! Elementos iguales.

      ELSE

        IEFLAG=0 ! En este caso los elementos son distintos.

      ENDIF

    ENDIF

    ITEMP=NTYPE ! Variable que verifica si el tipo de elemento cambia o no.


        VAREL(2)=DFLOAT(NTYPE) !     VAREL(2) = Tipo de prob.

        VAREL(3)=DFLOAT(NNODE) !     VAREL(3) = Nº de Nodos Elem.

        VAREL(4)=DFLOAT(NGAUS) !     VAREL(4) = Nº de Ptos. de Integ.

        VAREL(5)=DFLOAT(NINTE) !     VAREL(5) = Tipo de integracion.

        FELEM( NUMEL) = EFASE   ! Guarda (en FELEM) la fase del elemento

        CALL TYPEV3D

        CALL INTEGR3D(NGAUX)

    CALL BASE5I_V3D(NUMEL)
```

```
      CALL CHECK1_V3D

            NGAUSF(NUMEL) = NGAUS

            ELECOMT(MATNO(1)) = ELECOMT(MATNO(1)) + 1

            ELECOM(MATNO(1),ELECOMT(MATNO(1))) = IELEM

       ENDDO

C      CALCULO GRADO DE LIBERTAD MAXIMO POR NODO Y EL VALOR ACUMULADO

       NGLIBA=0

    DO IPOIN=1,NPOIN

      CALL NGLIB1_V3D(IPOIN)

      NGLIBA=NGLIBA+NGLIBL

      CALL BASE6I_V3D(IPOIN)

    ENDDO

       NTOTV=NGLIBA

       CALL ALOKA2_V3D

C      LECTURA DE LA RELACION ENTRE UN MATERIAL COMPUESTO Y SUS COMPONENTES
C      TODOS LOS MATERIAL  ES SON COMPUESTOS ( CON 1 o MAS COMPONENTES)


    DO NRO=1,NMATC

            READ(1,*) NMC, NULAY(NMC)

            DO ilay = 1, NULAY(NMC)

                  READ(1,*) TYLAY(NMC,ilay), PVLAY(NMC,ilay),
     .                  (LAEULER(NMC,ilay,ik), ik=1,3),NUCOMF(NMC,ilay),
     .                  (LCPRO(NMC,ilay,ik),PARTV(NMC,ilay,ik),
     .            ik=1,NUCOMF(NMC,ilay))

                  IF (TYLAY(NMC,ilay).EQ."NT") THEN
                        LCPRO(NMC,ilay,4) = LCPRO(NMC,ilay,2)
                        LCPRO(NMC,ilay,5) = LCPRO(NMC,ilay,3)
                        NUCOMF(NMC,ilay) = 5
                  END IF

            END DO

    END DO


!   Crea las bases de datos 1, 2 y  4

       CALL OTABR02_V3D;

C      ZERO ALL THE NODAL COORDINATES, PRIOR TO READING SOME OF THEM.

    COORD(:,:)=0.0D0;

    IF (TITLE1(1:6).NE.'START2') THEN ! MMH 19/05/2008 PARA CONTROL DE IMPRESION EN EL REINICIO

            WRITE(2,904);       WRITE(2,905)

       ENDIF  ! RESTART

    DO IP=1,NPOIN

        READ(1,*) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,3)

       ENDDO

    IF (TITLE1(1:6).NE.'START2') THEN ! MMH 19/05/2008 PARA CONTROL DE IMPRESION EN EL REINICIO

            DO IPOIN=1,NPOIN,2

      WRITE(2,907) IPOIN,
     .          (CO  ORD(IPOIN,IDIME),IDIME=1,3),
     .          IPOIN+1,
     .          (COORD(IPOIN+1,IDIME),IDIME=1,3)
```

```
          IF(IPOIN.EQ.NPOIN)THEN !EVITAR NODO EXTRA CUANDO SON IMPARES EL NÚMERO TOTAL DE NODOS
          WRITE(61,909)IPOIN,
     .          (COORD(IPOIN,IDIME),IDIME=1,3)
          ELSE
          WRITE(61,909)IPOIN,
     .          (COORD(IPOIN,IDIME),IDIME=1,3)
          WRITE(61,909)IPOIN+1,
     .          (COORD(IPOIN+1,IDIME),IDIME=1,3)
          ENDIF

        ENDDO

        ENDIF   ! RESTART
C     ---------- CALCULA LOS EJES LOCALES CON LA MANO DERECHA ----------

        CALL CONSTITUTIVAS;

C     ---------------------------------------------------------------------

  907 FORMAT(3X,I6,3F10.3,7X,I6,3F10.3)    !japl enero/23/09

  909 FORMAT(3X,I6,3F10.3)     !jbll 01/04/2011

  904 FORMAT(//,6X,'COORDINATES DEFINITION :',/,
     .       6X,'----------',//)
  905 FORMAT(3X,5H NODE,2X,7HX-COORD,2X,7HY-COORD,2X,7HZ-COORD,5X,
     .5H NODE,2X,7HX-COORD,2X,7HY-COORD,2X,7HZ-COORD)

  906 FORMAT(25X,30('='),/,36X,'MESH DATA',/,25X,30('='),///,
     .      6X,'ELEMENTS DEFINITION :',/,
     .      6X,'----------',//)

  903 FORMAT(I6,3X,I5,1X,1F10.3,1x,I5,1x,4I6,2X,8I8,3x,3F10.2)  !japl 25/11/2010

  908 FORMAT(I6,3X,I5,1X,I5,1x,4I6,2X,8I8,3x,3F10.2)  !jbll 01/04/2011

  902 FORMAT(//1X,8HELEMENT ,1X,6HMATNO ,3X,6HFRACL ,2X,
     .      6HNTYPE,1X,5HNFASE,1X,5HNNODE,1X,5HNGAUS,1X,
     .      6HNINTE,4X,15H NODE   NUMBERS,53X,7HÁNGULOS)

        END



    SUBROUTINE FINISH_V3D

C-----------------------------------------------------------------------
C
C       ESTA SUBROUTINA INTERCAMBIA LA BASE DE DATOS
C
C          NOCONVERGIDA ----------> CONVERGIDA
C
C-----------------------------------------------------------------------

        USE ENTEROS; USE MATRICES; USE PARAM3D;

        IMPLICIT NONE;
    Integer :: i, j, k, l, m, mat, posi1, posi2
        INTEGER :: IGAUS,JGAUS,LGAUS,IM,KTESC,ICONV,INDMA,IG
        Real(8) ::  DB2aux(36)

C     ---------- ---------------------------------------------------------

    KGAUS=0;

    KGSUM=0;

C     ---> UPDATE THE CONVERGED VALUES.

    do i = 1, NELEM
      call BASE5O_V3D(i) ! Read in DataBase 5
      mat = MATNO(1)
      LPROP=mat
      do j = 1, NGAUSF(i)
        do k = 1, NGAUSF(i)
          do l = 1, NGAUSF(i)
            do m = 1, FER(mat)
              posi1 = Pos_12(i,j,k,l,m,1)
                  posi2 = Pos_12(i,j,k,l,m,2)
```

```
c                  Llamada para extraer los datos del Csec y las tensiones convergidas,
c                  si estamos en grandes deformaciones exgrandef JBLL 07/04/2011
        if((m.EQ.1).and.(BAS2_V3D(posi1,9).EQ.0.0d0).and.(conf.eq.1))then

            CALL CALCsec_V3D(i,j,k,l,DB2aux)


                    call excsec(i, j, k, l,DB2aux)
                    if(large.EQ.1) then
                    call Exgrandef(i,j,k,l,Bas1_v3d(posi1,1:6))
                    endif
                   endif

          enddo
        enddo
      enddo
    enddo
  enddo

C     ---------- ACTUALIZA LA ENERGIA DISIPADA ----------

      DISIP=DISIP+DISID

      IF (TTVOL.NE.0.0D0) DAMAGM=DAMAG/TTVOL

      RETURN

      END



      SUBROUTINE CALCIN_V3D !*-*-*

C-----------------------------------------------------------------------
C
C     ESTA RUTINA CALCULA LOS TENSORES CONSTITUTIVOS Y LO ALMACENA
C
c         EN LA BASE DE DATOS Nro 4 EN FORMA DE VECTORES
C
C-----------------------------------------------------------------------

      USE ENTEROS; USE MATRICES; USE PARAM3D;

          IMPLICIT NONE;

          INTEGER IGAUS,JGAUS,INDXB5,IEQFU,INDMA,IM,INDXB4,ICOM,ISTR1,
     .       JSTR1,KTESC,I,J, full,DIR

      REAL(KIND=8) EXISP,ETASP,PARKC,DENSC,VAL1,VAL2,DMANT

          REAL(KIND=8), DIMENSION(LONG2_V3D)   :: VDMANX,VDMANXI,VCVISC

          REAL(KIND=8), DIMENSION(MSTR1,MSTR1) :: DMANX,DMANXI,CVISC, AUX1,
     .                                                AUX3, AUXL
          REAL (8) pvol, A1,A2,A3,DENSL

C     ----------> CALCULA LOS TENSOR ELASTICO DE LOS MATERIALES SIMPLES Y LOS GUARDA

          DO KPROP = 1, NMATS

                CALL MODPS_V3D                    !SE CÁLCULA Y SE GURADA EN LA BASE4

          END DO

C     ----------> CALCULA EL TENSOR ELASTICO ANISOTROPO PARA EL COMPUESTO

      DO LPROP=1,NMATC              ! BUCLE SOBRE CADA MAT. COMP.

           DMANX(1:MSTR1,1:MSTR1) =0.0D0;   ! TENSOR ANISOTROPO
           DMANXI(1:MSTR1,1:MSTR1)=0.0D0;   ! INV. DEL TENS. ANIST.
           CVISC(1:MSTR1,1:MSTR1) =0.0D0;   ! TENSOR VISCOSO.
        DENSc=0.0D0;                ! DENSIDAD DEL MATERIAL.
           AUX1(1:MSTR1,1:MSTR1) =0.0D0;
           AUX3(1:MSTR1,1:MSTR1) =0.0D0;

                DO KLAY =1, NULAY(LPROP)          ! BUCLE SOBRE CADA LAMINA DEL COMPUESTO

                     pvol = PVLAY(LPROP,KLAY)
```

```
                    SELECT CASE (TYLAY(LPROP,KLAY))

                    CASE ('P')

                    CALL MODCO_V3D(DMANX,CVISC,DENSL)

                    CASE ('SP')

                    CALL CLAY_SP3D(DMANX,CVISC,DENSL)

                    CASE ('NT')

                    CALL CLAY_NT3D(DMANX,CVISC,DENSL)

                    END SELECT

!       --- ROTACIÓN DEL TENSOR CONSTITUTIVO LAYER A LAS COORDENADAS DEL COMPUESTO --------

                    A1 = LAEULER(LPROP,KLAY,1)
                    A2 = LAEULER(LPROP,KLAY,2)
                    A3 = LAEULER(LPROP,KLAY,3)

                    IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
                    dir = 2  ! local LAMINA to global MATERIAL COMPUESTO
                call ROTM(DMANX,A1,A2,A3, mstr1, dir)
                call ROTM(CVISC,A1,A2,A3, mstr1, d ir)
                    END IF

!       --- TENSOR CONSTITUTIVO DEL COMPUESTO EN SU COORDENADAS --------

                    AUX1(:,:) = AUX1(:,:) + pvol*DMANX(:,:)
                AUX3(:,:) = AUX3(:,:) + pvol*CVISC(:,:)
                DENSc = DENSc + pvol*DENSL

            END DO

            DMANX=AUX1
            CVISC=AUX3

            CALL INVERT3D(MSTR1,DMANX,DMANXI,AUXL)

        DENSV(MATNO(1))=DENSc;


C   ------------------ DESCOMPONE  TENSOR ------>  VECTOR ----------------------

        ICOM=2
    full=1  ! Complete Matrix
    CALL COMPON_V3D(VDMANX ,DMANX ,LONG2_V3D,full,ICOM)
        CALL COMPON_V3D(VDMANXI,DMANXI,LONG2_V3D,full,ICOM)
        CALL COMPON_V3D(VCVISC ,CVISC ,LONG2_V3D,full,ICOM)

C   -- ESCRITURA EN LA BASE DE DATOS DE LOS TENSORES CORRESPONDIENTES AL COMPUESTO --
C               ESCRITURA EN LA BASE DE DATOS - 4 -

        VB4_V3D((lb4_4+1):(lb4_4+LONG2_V3D))   = VDMANX(1:LONG2_V3D);
    VB4_V3D((lb4_8+1):(lb4_8+LONG2_V3D))   = VDMANXI(1:LONG2_V3D);
    VB4_V3D((lb4_13+1):(lb4_13+LONG2_V3D)) = VCVISC(1:LONG2_V3D);

C       -------- SE GUADA LOS TENSORES DEL COMPUESTO EN SUS COORDENADAS -------

        KTESC=1;
        INDMA=0;
    CALL BASE4_V3D(LPROP,INDMA,KTESC);

        IF(ICARG.EQ.1.0) THEN
            CALL EXCO(LPROP,VDMANX,DENSC)
        ENDIF
       ENDDO

   RETURN

   END
```

```
      SUBROUTINE INICIA(DATOS1,DATOS4,CPROB1)

C     ------------------------------------------------------------------------
C
C     ESTA SUBRUTINA ABRE LOS ARCHIVOS DE DATOS DE LECTURA Y ESCRITURA
C
C                    PMA, CIMNE, UPC, 2004.
C
C                  INDI = 1. INICIA EL PROBLEMA.
C                         2. TERMINA.(CIERRA TODO).
C
C     ------------------------------------------------------------------------

      IMPLICIT NONE;

      INTEGER I,II,INDI,IK,IND ,IN1,IN2,IL1,IM1,J,IHA,IMI,ISG,NFORMATOS,
     .   NIND1,NIND2,NIND3,NPASO,NPUNT1,NPUNT2,NUNIT,NUNLOG

      REAL(KIND=8) TIME

    INTEGER, DIMENSION(14) :: DATOS1

      CHARACTER(LEN=64) CPSAL,CPROB1

      CHARACTER(LEN=64) CRA(20),CAP(20)

      CHARACTER(LEN=64), DIMENSION(21) :: DATOS4

      CHARACTER(LEN=64) CB

C     ------------------------- FIN DECLARACIONES ------------------------------


      NIND1=1; NIND2=1; NIND3=1; NPASO=0; NPUNT1=0; NPUNT2=0; ! PARAM INICIALES.

      NUNIT=2;          ! SALIDA DE RESULTADOS

      NUNLOG=8;         ! INFORME DE EJEC.

      DATOS1(1)=NIND1;

      DATOS1(2)=NIND2;

      DATOS1(3)=NIND3;

      DATOS1(4)=NPASO;

      DATOS1(5)=NPUNT1;

      DATOS1(6)=NPUNT2;

      DATOS1(7)=NUNIT;

      DATOS1(8)=NUNLOG;

      CRA(1)='.dts';        ! FORMATO DE ENTRADA.
      CRA(2)='.mec.sal';    ! FROMATO DE SALIDA.
      CRA(3)='.rst';        ! FORMATO DE SCRIPT DE CAD.
      CRA(4)='.mec.pos';    ! FORMATO DE SALIDA POSTPROCESO.
    CRA(5)='.din';        ! FORMATO DE DATOS DINAMICOS.
      CRA(6)='.arc';        ! FORMATO PARA "ARC-LENGTH".
      CRA(7)='.sec';        ! FORMATO DE DATOS DE SECCIONES, (SOLO 3D).
      CRA(8)='.mec.log'     ! FORMATO INFORME DE EJEC.
      CRA(9)='.mec.sec'     ! FORMATO INFORME SECCIONES.
      CRA(10)='.post.msh'   ! FORMATO DE MALLA PARA GID.
    CRA(11)='.post.res'   ! FORMATO DE RESULTADOS PARA GID.

      NFORMATOS=11;

    OPEN(UNIT=0,FILE='con',STATUS='UNKNOWN',
     .                      CARRIAGECONTROL='FORTRAN')

    CALL CARTEL;

      DO J=1,64

        IF (CPROB1(J:J).EQ.'.') II=(J-1);

    ENDDO

      OPEN(UNIT=61,FILE='Datosgen.txt',POSITION='APPEND') !jbll 01/04/2011 Para cálculo de frecuencias
      WRITE(61,*)CPROB1(1:II+1)
```

```
      DO IK=1,NFORMATOS

            CAP(IK)(1:(II+10))=CPROB1(1:II)//CRA(IK)(1:10);

      ENDDO

          DATOS4(1:20)=CAP(1:20);

          CB=CAP(2); !CA=CAP(1); CD=CAP(3); CE=CAP(4); CH=CAP(5);

C         CI=CAP(6); CJ=CAP(7); CK=CAP(8);

      DO IN1=1,64

        IF (CB(IN1:IN1).EQ.'.') GOTO 445

          ENDDO

  445 IN1=IN1-1

      IN2=64

      IF (IN1.LT.IN2) IN2=IN1;

          DATOS4(21)=CB;

          DATOS1(13)=IN1;

          DATOS1(14)=IN2;

C     ------------------ ENCUENTRA NUMERO DE CARACTERES UTILES --------------------------

       OPEN(UNIT=1,       FILE=CAP(1),STATUS='UNKNOWN')              ! ABRE ARCH. DE LECTURA
       OPEN(UNIT=3,       FILE=CAP(10),STATUS='UNKNOWN')             ! ABRE ARCHIVO DE GEOM.
          OPEN(UNIT=4,       FILE=CAP(11),ACCESS='APPEND',STATUS='UNKNOWN')  ! ABRE ARCH. DE RESULT.
       OPEN(UNIT=6,       FILE=CAP(6),STATUS='UNKNOWN')              ! ABRE ARCH. PAR DE PUNTO    S ARC. LENGHT

          OPEN(UNIT=DATOS1(7),FILE=CAP(2),ACCESS='APPEND',STATUS='UNKNOWN')   ! ABRE PPAL.DE SALIDA.
ARREGLO ACCESS
          OPEN(UNIT=DATOS1(8),FILE=CAP(8),ACCESS='APPEND',STATUS='UNKNOWN')   ! ABRE INFORME EJEC.


          CALL MAKINA(3);
       CALL MAKINA(4);
          CALL MAKINA(5);

          RETURN

          END



      SUBROUTINE RESI2D_V3D(ELDIS,VSTRAT,STRSG,STRAP,EPSTN,
     .                EFFST,STRAT,BMATX,IGAUS,JGAUS,LGAUS,
     .          EXISP,ETASP,EZETA,DVOLU,STRFAS,VSTRFAS,AUXI,   AUXI1

C-----------------------------------------------------------------------
C
C     ESTA RUTINA INICIA EL TRATAMIENTO CONSTITUTIVO PARA ELEMENTOS
C                    2 - D  y 3 - D
C
C-----------------------------------------------------------------------

      USE ENTEROS; USE MATRICES; USE PARAM3D;

      IMPLICIT NO NE;

          INTEGER INDV,JGAUS,IGAUS,LGAUS,KTESC,INDMA,ICOM,MGASH,INODE,
     .       IDOFN,ISTRE,IKND,I,OLD_LPROP, full,ICONV,J

          REAL(KIND=8) PINUM,ROOT3,TWOPI,ETASP,EXISP,DJACB,EZETA,THICK,PXY,
     .            PXZ,PYZ,PESO,DVOLU,EFFST,EPSTN,    ! japl ANGLOELE angulo del elemento
     .           LEFFST,LEPSTN,A1,A2,A3
      REAL(KIND=8), DIMENSION(MSTR1)  :: STRAN,STRAP,STRSP,
     .                             STRSG,STRAT,VSTRAN,
     .                            VSTRAT,STRFAS,VSTRFAS,
     .                             LSTRSG,LSTRSP,
     .                STRATcom,STRANcom, STRATlay,STRANlay,
     .                                      VSTRATcom,VSTRANcom,VSTRATlay,VSTRANLAY
          REAL(KIND=8), DIMENSION(MSTR1,MSTR1)       :: AUX1,AUXI
          REAL(KIND=8), DIMENSION(MSTR1,MDOFN*MNODE) :: BMATX
          REAL(KIND=8), DIMENSION(3,3)               :: FGRAD,RLAYX
      REAL(KIND=8), DIMENSION(LONG2_V3D)         :: VDMANX,VDMANXI

      REAL(KIND=8), DIMENSION(MDOFN,MNODE,2)     :: ELDIS
      REAL(KIND=8), DIMENSION(NDIME,36)          :: GPCOD
          REAL(KIND=8), DIMENSION(NDIME,NDIME)       :: XJACM

C     ---------------------------------------------------------------------
```

```
      CALL PROCOM_V(thick,3)
      CALL PROCOM_V(pxy,2)
          CALL PROCOM_V(pxz,38)
      CALL PROCOM_V(pyz,41)
      PESO=WEIGP(JGAUS)
      IF (NGAUY.EQ.1) PESO=1.0
          DVOLU=DJACB*WEIGP(IGAUS)*PESO
          IF (THICK.NE.0.0) DVOLU=DVOLU*THICK
          CALL BMATSOL_V3D(BMATX,GPCOD,FGRAD)
          IF (NTYPE.EQ.6) CALL DEFO3D(FGRAD,STRAN,STRAT,STRFAS)  ! MODIF V3D
C     ------ CALCULO DE LAS VELOCIDADES DE DEFORMACION - SOLO P/ PROBLEMA DINAMICO --------
      VSTRAN(:)=0.0D0;
          IF (IDINA.EQ.1) THEN
            INDV=2
            CALL GRADEF_V3D(XJACM,FGRAD,GPCOD,ELDIS,INDV)
            CALL DEFO3D(FGRAD,VSTRAN,VSTRAT,VSTRFAS)
          ENDIF
C     ---------------------- CALCULO DE LA FUERZA INTERNA ELASTICA -------------------------
      MGASH=0
      CALL CALTEN_V3D(AUX1,STRSG,STRAT)
          DO INODE=1,NNODE
            DO IDOFN=1,NDOFN
              MGASH=MGASH+1
              DO ISTRE=1,NSTRE
            ELOAD(MGASH,3)=ELOAD(MGASH,3)+
     .              BMATX(ISTRE,MGASH)*STRSG(ISTRE)*DVOLU
              ENDDO
            ENDDO
          ENDDO
C     ---- CASO GENERAL DE PROBLEMAS NO- LINEALES CON VARIOS MATERIALES DE VARIOS COMPONENTES --
      IF (ACTIV(FELEM(IELEM)) .EQ. 1) THEN ! Solo se calculan tensiones si elemento activo
          KTESC = 1              ! Write
          ICONV = 2              ! Not converged
```

```
          INDMA = 0               ! Composite material index in DataBase 2
          VB2_V3D(1:lb2_M) = 0.0d0;
          VB2_V3D(9)      = 1.0d0;   ! EUNLD = 1.0, Lo guardamos ya en DB2

          call BASE2_V3D(IELEM,IGAUS,JGAUS,LGAUS,INDMA,ICONV,KTESC)

           STRSG   = 0.0d0
           STRSP   = 0.0d0
           EFFST   = 0.0d0
                EPSTN   = 0.0d0
                STRATcom = STRAT
                STRANcom = STRAN
                VSTRATcom = VSTRAT
                VSTRANcom = VSTRAN

                A1 = EAEULER(1)
                A2 = EAEULER(2)
                A3 = EAEULER(3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then

                      CALL ROTV(STRATcom,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(STRANcom,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(VSTRATcom,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(VSTRANcom,A1,A2,A3,NSTRE,1,1)

                END IF

           DO KLAY = 1, NULAY(LPROP)

                LSTRSG = 0.0
                LSTRSP = 0.0
                STRATlay=STRATcom
                STRANlay=STRANcom
                VSTRATlay=VSTRATcom
                VSTRANlay=VSTRANcom

C               ROTACIÓN DE LAS DEFORMACIONES AL LOCAL DE LA LAYERS

                A1 = LAEULER(LPROP,KLAY,1)
                A2 = LAEULER(LPROP,KLAY,2)
                A3 = LAEULER(LPROP,KLAY,3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then

                      CALL ROTV(STRATlay,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(STRANlay,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(VSTRATlay,A1,A2,A3,NSTRE,1,1)
                      CALL ROTV(VSTRANlay,A1,A2,A3,NSTRE,1,1)

           END IF

                SELECT CASE (TYLAY(LPROP,Klay))

                 CASE ('P')

          CALL P_MNGER_V3D(LSTRSG,LSTRSP,STRATlay,STRANlay,LEFFST,LEPSTN,
         .       DVOLU,VSTRATlay,VSTRANlay,FGRAD,ETASP,EXISP,
         .       lGAUS,JGAUS,LGAUS)

                 CASE ('SP')

          CALL SP_MNGER_V3D(LSTRSG,LSTRSP,STRATlay,STRANlay,LEFFST,LEPSTN,
         .       DVOLU,VSTRATlay,VSTRANlay,FGRAD,ETASP,EXISP,
         .       EZETA,IGAUS,JGAUS,LGAUS)

                 CASE ('NT')

          CALL NT_MNGER_V3D(LSTRSG,LSTRSP,STRATlay,STRANlay,LEFFST,LEPSTN,
         .       DVOLU,VSTRATlay,VSTRANlay,FGRAD,ETASP,EXISP,
         .       EZETA,IGAUS,JGAUS,LGAUS)


                 END SELECT

C       ROTACIÓN DE LAS TENSIONES DE LOCAL LAYER TO COMPOSITE

                A1 = LAEULER(LPROP,KLAY,1)
                A2 = LAEULER(LPROP,KLAY,2)
                A3 = LAEULER(LPROP,KLAY,3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
```

```
                CALL ROTV(LSTRSG,A1,A2,A3,NSTRE,2,2)
            CALL ROTV(LSTRSP,A1,A2,A3,NSTRE,2,2)

          END IF

           STRSG(:) = STRSG(:) + PVLAY(LPROP,KLAY)*LSTRSG(:)
           STRSP(:) = STRSP(:) + PVLAY(LPROP,KLAY)*LSTRSP(:)
                EFFST   = EFFST   + PVLAY(LPROP,KLAY)*LEFFST
                EPSTN   = EPSTN   + PVLAY(LPROP,KLAY)*LEPSTN

              END DO


          DO I=1,NSTR1

                DO J=1,NSTR1

                STRAP(I)=STRAP(I)+AUXI(I,J)*STRSP(J);

                END DO

              END DO


        ELSE

         STRSG(:) = 0.0d0

        END IF

C       ROTACIÓN DE LAS TENSIONES DE LOCAL COMPOSITE TO GLOBAL ELEM   ENTS

        A1 = EAEULER(1)
        A2 = EAEULER(2)
        A3 = EAEULER(3)

        IF(A1**2+A2**2+A3**2.GT.1.e-9) Then

              CALL ROTV(STRSG,A1,A2,A3,NSTRE,2,2)
          CALL ROTV(STRAP,A1,A2,A3,NSTRE,2,1)

        END IF

C     ------ CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE ELEMENT NODES --------


        MGASH=0

        DO INODE = 1, NNODE

         DO IDOFN = 1, NDOFN

          MGASH = MGASH + 1

              DO ISTRE = 1, NSTRE

               DO IKND = 1, 2

         ELOAD(MGASH,IKND) = ELOAD(MGASH,IKND) +
     .                BMATX(I   STRE,MGASH)*STRSG(ISTRE)*DVOLU
        ENDDO

              ENDDO

         ENDDO

        ENDDO

      RETURN

      END
```

```
      SUBROUTINE DAMAGE_V3D

C-------------------------------------------------------------------------------
C
C    ESTA RUTINA EVALUA EL DANO GLOBAL EN FUNCION DE LA FUERZA REACTIVA
C
C             D = (1 - |Fr|/|Fo|)
C
C                    |Fr| = |STFOR|
C                    |Fo| = |TOFOR|
C
C-------------------------------------------------------------------------------

      USE ENTEROS; USE MATRICES; USE PARAM3D;

          IMPLICIT NONE;

          INTEGER KEVAB,INODE,LOCNO,IDOFN,NPOSI,ITOTV

          REAL(KIND=8) FR,FO

C     -- ---------------------------------------------------------------------

      Fr=0.0D0;

      Fo=0.0D0;

      STFOR(:)=0.0D0;

      TOFOR(:)=0.0D0;

      DO IELEM=1,NELEM

        KEVAB=0

            CALL BASE5O_V3D(IELEM)

            DO INODE=1,NNODE

              LOCNO=IABS(LNODS(INODE))

              CALL BASE6O_V3D(LOCNO)

              DO IDOFN=1,NDOFN

                KEVAB=KEVAB+1

                NPOSI=NGLIBA-NGLIBL+IDOFN

                STFOR(NPOSI)=STFOR(NPOSI)+ELOAD(KEVAB,2)

            TOFOR(NPOSI)=TOFOR(NPOSI)+   ELOAD(KEVAB,3)

          ENDDO

            ENDDO

          ENDDO

      DO ITOTV=1,NTOTV

            Fr=Fr+STFOR(ITOTV)*STFOR(ITOTV)

            Fo=Fo+TOFOR(ITOTV)*TOFOR(ITOTV)

          ENDDO

      Fr=SQRT(Fr)

      Fo=SQRT(Fo)

C     ------------------------------ INDICE DE DANO GLOBAL --------------------------

      DGFORC=1.0-(Fr/Fo)

C     -------------------------------- IMPRESION DEL DANO ----------------------------

      IF (NTYPE.LT.20) WRITE(2,900)DAMAGM,DGFORC

          IF((CONF.EQ.1).AND.(DGFORC.GT.0)) THEN  !JBLL 04/06/11
                OPEN(UNIT=75,FILE = 'Daño.txt',STATUS='UNKNOWN',
     .    POSITION='APPEND')
                WRITE(75,*)DGFORC
          ENDIF

  900 FORMAT('********  DANO GLOBAL -medida energetica--: ',1E15.6 /,
     .       '********  DANO GLOBAL -medida en fuerza-- : ',1E15.6 )

      RETURN

      END
```

```
      SUBROUTINE RESIDU_V3D

C-----------------------------------------------------------------------
C
C     THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND
C     EVALUATES THE EQUIVALENT NODAL FORCES
C
C-----------------------------------------------------------------------

          USE ENTEROS; USE MATRICES; USE PARAM3D;

!     IMPLICIT NONE;

          INTEGER INDXB5,INODE,IDIME,NPOSN,IDOFN,ID,LGAUS,IGAUS,JGAUS,JEQFU,
     .       KTESC,ICONV,INDMA,LNODE,IEQFU

          REAL(KIND=8) PINUM,ROOT3,TWOPI,EXISP,ETASP,EZE  TA,EPSTN,EFFST,
     .           DVOLU

      REAL(KIND=8), DIMENSION(MSTR1)      :: STRSG,STRAP,STRAT,VSTRAT,
     .                                STRSGt,STRATt,VSTRATt,
     .                               STRFAS,VSTRFAS

      REAL(KIND=8), DIMENSION(NDIME,NDIME) ::    XJACM
      REAL(KIND=8), DIMENSION(MDOFN,MNODE,2)    :: ELDIS
      REAL(KIND=8), DIMENSION(MSTR1,MDOFN*MNODE) :: BMATX
          REAL(KIND=8), DIMENSION(MSTR1,MSTR1)      :: AUXI,  AUX1
          REAL(KIND=8), DIMENSION(LONG2_V3D)        :: VDMANXI,V  DMANX

C     ------------------------------------------------------------------

          PINUM=DACOS(-1.0D00)

          ROOT3=DSQRT(3.0D00)

          TWOPI=2.0D00*PINUM

      DO IELEM=1,NELEM

        CALL BASE5O_V3D(IELEM);

        ELOAD(1:NEVAB,1:3)=0.0D0;      ! INICIALIZA LA MATRIZ DE CARGAS CON CERO.

        CALL BASE5I_V3D(IELEM);

      ENDDO

C     ---------- INICIA EL LOOP SOBRE TODOS LOS ELEMENTOS ---------

      DAMAG=0.0D0

      DISID=0.0D0

      TTVOL=0.0D0

      KGSUM=0

      KGAUS=0

      DO IELEM=1,NE LEM

            INDXB5=IELEM

            CALL BASE5O_V3D(IELEM)

        LPROP=MATNO(1)              ! TIPO DE MATERIAL

            NELIM=MATNO(2)              ! ELEMENTO NORMAL O JUNTA

C
C     ---- COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE ELEMENT NODAL POINTS ----
C

        ELDIS(:,:,:)=0.0D0;  !ELDIS(NDIME,MNODE,2)

        DO INODE=1,NNODE

              LNODE=IABS(LNODS(INODE))
```

```
          CALL BASE6O_V3D(LNODE)
          DO IDIME=1,NDIME
        ELCOD(IDIME,INODE)=COORD(LPNINV(LNODE),IDIME) ! COORDS DE NODOS.
      ENDDO
          NPOSN=NGLIBA-NGLIBL     ! POSICION GLOBAL DE LOS GDLS DEL ELEMENTO.
          DO IDOFN=1,NDOFN
            NPOSN=NPOSN+1
            ELDIS(IDOFN,INODE,1)=TDISP(NPOSN,1)
            ELDIS(IDOFN,INODE,2)=TDISP(NPOSN,2)
            IF (LARGE.NE.0) THEN
              IF (IDOFN.LE.NDIME) THEN     ! NO ACTUALIZA LAS RO   TACIONES !!!
                ID=IDOFN
                    ELDIS(IDOFN,INODE,1)=TDISP(NPOSN,1)+ELCOD(ID,INODE)
                ELDIS(IDOFN,INODE,2)=TDISP(NPOSN,2)
              ENDIF
                    ENDIF
      ENDDO
    ENDDO

C
C   ------------------- LECTURA INV. TENSOR ANISOTROPO PARA EL COMPUESTO --------------
C
              KTESC=2         ! LECTURA.
              INDMA=0         ! COMPUESTO.
              CALL BASE4_V3D(LPROP,INDMA,KTESC)
              VDMANX(1:LONG2_V3D)=VB4_V3D((lb4_4+1):(lb4_4+LONG2_V3D))
              VDMANXI(1:LONG2_V3D) = VB4_V3D((lb4_8+1):(lb4_8+LONG2_V3D))


              ICOM=1
              full = 1 ! Matriz entera
              CALL COMPON_V3D(VDMANX, AUX1,LONG2_V3D, full, ICOM)  ! Tensor Constitutivo
              CALL COMPON_V3D(VDMANXI,AUXI,LONG2_V3D, full, ICOM)

C
C   ------------------- BUCLE SOBRE LOS PUNTOS DE GAUSS DEL ELEMENTO --------------
C
        KGASP=0
    LGAUS=1  !CORREGIR PARA OTROS ELEMENTOS TRIDIMENSIONALES
        DO IGAUS=1,NGAUX
          EXISP=POSGP(IGAUS)
          DO JGAUS=1,NGAUY
            ETASP=POSGP(IEQFU(IGAUS,JGAUS))
            DO LGAUS=1,NGAUZ
              EZETA=POSGP(JEQFU(IGAUS,JGAUS,LGAUS))
              KGAUS=KGAUS+1
```

```
            KGASP=KGASP+1

C
C       LECTURA CONVERGIDA EN EL PASO ANTERIOR DE TENSIONES Y DEFORMACIONES PLASTICAS
C                       PARA EL COMPUESTO

          KTESC=2   ! LECTURA.

            ICONV=1   ! CONVERGIDO.

            INDMA=0   ! COMPUESTO

            CALL BASE1_V3D(IELEM,IGAUS,JGAUS,LGAUS,
     .                          INDMA,ICONV,KTESC);

            STRAT(1:NSTR1)  = VB1_V3D((lb1_1+1):(lb1_1+NSTR1))

            VSTRAT(1:NSTR1) = VB1_V3D((lb1_6+1):(lb1_6+NSTR1))

            STRSG(1:NSTR1)  = VB1_V3D(1:NSTR1)

            STRAP(1:NSTR1)  = VB1_V3D((lb1_2+1):(lb1_2+NSTR1))

            STRATt(1:NSTR1) = VB1_V3D((lb1_8+1):(lb1_8+NSTR1))

            VSTRATt(1:NSTR1)= VB1_V3D((lb1_9+1):(lb1_9+NSTR1))

                STRSGt(1:NSTR1) = VB1_V3D((lb1_7+1):(lb1_7+NSTR1))

                    STRFAS(1:NSTR1) = VB1_V3D((lb1_17+1):(lb1_17+NSTR1))

                    VSTRFAS(1:NSTR1)= VB1_V3D((lb1_18+1):(lb1_18+NSTR1))

          EPSTN=VB1_V3D(lb1_4)

            EFFST=VB1_V3D(lb1_5)


            SELECT CASE(NTYPE)

            CASE(6)       ! PARA ELEMENTOS FINITOS 2-D y 3 - D

              CALL RESI2D_V3D(ELDIS,VSTRAT,STRSG,STRAP,
     .                EPSTN,EFFST,STRAT,BMATX,IGAUS,JGAUS,
     .                LGAUS,EXISP,ETASP,EZETA,DVOLU,
     .                STRFAS,VSTRFAS,AUXI,AUX1)
            CASE DEFAULT

           WRITE(*,*) 'ERROR     : NTYPE BAD DEFINED'
              WRITE(*,*) 'SUBROUTINE : ----> RESIDU_V3D'
              STOP

            END SELECT

C    ---------------------- ESCRITURA EN LA BASE DE DATOS - 1 ------------------------

            VB1_V 3D(1:NSTR1)             = STRSG(1:NSTR1);

            VB1_V3D((1+lb1_1):(NSTR1+lb1_1))  = STRAT(1:NSTR1);

            VB1_V3D((1+lb1_2):(NSTR1+lb1_2))  = STRAP(1:NSTR1);

            VB1_V3D((1+lb1_6):(NSTR1+lb1_6))  = VSTRAT(1:NSTR1);

                    VB1_V3D((lb1_17+1):(lb1_17+NSTR1)) = STRFAS(1:NSTR1);

                    VB1_V3D((lb1_18+1):(lb1_18+NSTR1)) = VSTRFAS(1:NSTR1);

          VB1_V3D(lb1_4)=EPSTN;

            VB1_V3D(lb1_5)=EFFST;

            KTESC=1    ! ESCRITURA.

            ICONV=2    ! NO CONVERGIDO.

            INDMA=0    ! COMPUESTO
```

```
                CALL BASE1_V3D(IELEM,IGAUS,JGAUS,LGAUS,
     .                         INDMA,ICONV,KTESC);

              ENDDO

            ENDDO

          ENDDO

          CALL BASE5I_V3D(INDXB5)

        ENDDO

        RETURN

          END




          MODULE ENTEROS

C     ------------------------------------------------------------------------
C
C     ESTE MODULO TRASPASA AL RESTO DEL PROGRAMA LOS ENTEROS Y CARACTERES
C
C                 PMA, CIMNE, UPC, 2004
C
C     ------------------------------------------------------------------------

        SAVE

C     -------------------------- ENTEROS ---------------------------------------

          INTEGER NPASO,NIND1,NIND2,NIND3,NIND4,NIN,IN1,MDOFN

                    ! CPU

       REAL(KIND=8) CPUCO,CPUDE,CPUSL,CPUSU,CPUSO,CPUOU,CPURE,CPUUP,
     .            CPUAS,CPURS,CPUST,CPUSF,CPUDA,TIME1,TIME2,TOCPU,CPUID

                        ! ENTRADA DATOS

          INTEGER KINDP,LARGE,NCONST,NLAG,NALGO,ISIME,NPOIN,NELEM,NMATC,
     .        NMATS,NINCM,IRFLAG,ISOLVER,KRENU,IFIPO,ISMOO,ICOUP,MAXLAY,
     .       NTOTV,NKRYLO,NITER,IPRINT,NTYPE,NNODE,NGAUS,NINTE,MNODE,
     .       NORI
                        ! FASES DE CARGA

          INTEGER NFASE, EFASE, FASES

                         ! CARGAS

          INTEGER NCARG,KCARG,IPLOD,IGRAV,IEDGE,IDINA,IINTE,IPLODT,IGRAVT,
     .       IEDGET,NEDGE,KLECT,NGAUZ

                        ! VINCULACIONES

        INTEGER NVFIX,NVINT,NVFIXT,NVFIXC

                        ! AUXILIARES

        INTEGER INDRIG,IPROB,MFRON,NBANDA,NTBANDA,NBANDASRENU,NELIM,ICARG,
     .        NROIT,MITER,KARCL,IN2,NODIS,LGRAD,IPOST,NROPU,KGASP,INDFU,
     .       IITER,KRESL,MBUFA,MSTIF,MKRYL,MWORK,I_DOF,IFATIG,NCHEK,
     .       NARCH,MGA  U2,NFRON,KFRON,IINCS,MBANDA,MIWORK,NCURV,NINCS,
     .       ISISM,INSA,NTIME,IFMAS,IFATLOAD,IFRPO,KLODPT,NEVAB,NCODE,
     .       NUMEL,KGAUS,KGSUM,IELEM,NDFRO,NPG,IEFLAG,NGLIBA,NGLIBL,
     .       MXCON,NVFIXM,KUCOM,LPROP,KPROP,IUCOM,KLAY,ICAP,MEVAB,
     .       NSTRE,NDOFN,NSTR1,NGAUX,NGAUY,MTENS,NPUNT1,NPUNT2,IVERS,
     .      TFACTT,NOFIXC,WORDS,NWOPA,NNWOR,NNPAR,NUNLOG,NUNIT,  JBL2,conf
```

```
C     ------------------------------- SOLO PARA VIGAS 3D ----------------------

      INTEGER MASEC,          ! Nº DE SECCIONES TIPO.
     .        MAVERT,         ! MAX Nº DE VERTICES.
     .        MATRAP,         ! MAX Nº DE TRAPECIOS.
     .        NACAD,          ! :1 HACE ARCHIVO DE CAD, : 0 EN CASO CONTRARIO.
     .        LSECC,          ! INDICADOR DE SECCION
     .        ITRAP,          ! INDICADOR DEL Nº DE TRAPECIO.
     .        NSECC,          ! Nº DE SECCIONES    DONDE SE POSTPROCESA.
     .        ISECP

C     ------------------------------- BASES DE DATOS ----------------------------

                              ! BASE 1.

          INTEGER   lb1_1,lb1_2,lb1_3,lb1_4,lb1_5,lb1_6,lb1_7,lb1_8,lb1_9,
     .            lb1_10,lb1_11,lb1_12,lb1_13,lb1_14,lb1_15,lb1_16,lb1_17,
     .            lb1_18,lb1_19,lb1_M

                           ! BASE 2.

          INTEGER   lb2_1,lb2_2,lb2_M

                           ! BASE 3.

          INTEGER   lb3_1,lb3_2,lb3_3,lb3_4,lb3_5,lb3_6,lb3_M

                                                ! BASE 4.

      INTEGER   lb4_1, lb4_2, lb4_3, lb4_4, lb4_5, lb4_6, lb4_7, lb4_8,
     .          lb4_9, lb4_10,lb4_11,lb4_12,lb4_13,lb4_14,lb4_15,lb4_16,
     .          lb4_M, LONG, LONG2, LONG_V3D, LONG2_V3D

                           ! BASE 5.

      INTEGER   lb5_1,lb5_2,lb5_3,lb5_4,lb5_5,lb5_6,lb5_7,lb5_8,lb5_9,
     .          lb5_10,lb5_11,lb5_12,lb5_13,lb5_14,lb5_M

                           ! BASE 6.

          INTEGER   lb6_1,lb6_2,lb6_M

      INTEGER   MRb_4,MRb_5,MRb_6,MR_DUMP

          INTEGER   MBAS4,MBAS6,MBAS5,MAXIB2,AXIB6,MIB2

          INTEGER   MAXIB4,MAXIB5,MAXIB6,MIB3,MIB4,MIB5,MIB6

C     ----------------------- REALES (KIND=8) -----------------------


          REAL(KIND=8) DTIME,TTIME,DISIP,PDEFO,PCINE,EINTR,EDISI,PINTR,
     .           DAMAG,DISID,TTVOL,DAMAGM,DGFORC,HPAR,
     .          D1,ZETA,EEQI,GEQI,CPURN,TFACT,FAC    TO,TFACTM,TOLER,
     .          ARCLN,VNCICL,RATIO,REMAX,RAYLA,RAYLB,COEFM,COEFC,
     .          COEFD,DFGRAD,REVER,SAPL_MAX,RFATI,FREDUC,CONSTA,
     .          SMAX,SI_1,SI_2,BCOFAT,VALRA,VNEWB,VNEWG,PERIOD,
     .          VNCIC0,GMTOL,TVOLU,PDISI,SMIN,ALFEXP,GEQZ,PVALU,Npar

C     ------------------------- CARACTERES --------------------------

          CHARACTER*64 CPROB,TYPEP,TITLE1,TITLE2,CPARA,MENSA

          CHARACTER(LEN=64) CFIL1,CC,CA,CB,CD,CE,CH,CI,CJ,CK,CF1,CF2,CF3,
     .              CF4,CF5,CF6,CF7,CF8,CF9,CF10,CF18,CF19,CF20,
     .              CF21,CF22,CF23,CF24,CF25,CF26,CF27,CF28,CF29,
     .              CF30,CF31,CF32,CF34

      CHARACTER(LEN=23) TIMER;

      CHARACTER(LEN=90) CCARD;

      CHARACTER CCUR*5

                ! SOLO SE EMPLEAN EN PROB TERMICO ACOPLADO

C     NROPU(MROPU,2),IFFIXT(MTOTV,2),ICONVE(MVFIX,5),TLOADT(MEVAB),ELOADT(MEVAB,3),
C     RLOADT(MEVAB,3),AUXHI(MBAS2),XEPS1(MMATS,100),YSIG1(MMAT  S,100),XEPS2(MMATS,100),
C     YSIG2(MMATS,100),XEPS3(MMATS,100),YSIG3(MMATS,100),TDISPT(MPOIN,3),FIXEMT(MTOTV),
C     PROPOLD55(MMATS),PROPOLD56(MMATS),PROPOLD57(MMATS),CHARACTER*5 WORDS(MAXWP),NP
```

```
C     -------------------------------- VIGAS 3D ---------------------------------

      INTEGER LONG4          ! LONGITUD TENSORES EN BASES 4.

     INTEGER IBD1_1,        ! BASE DE DATOS 1.
    .       IBD1_2,
    .       IBD1_3,
    .       IBD1_4,
    .       IBD1_5,
    .       IBD1_6,
    .       IBD1_7,
    .       IBD1_8,
    .       IBD1_9,
    .       IBD1_10,
    .       IBD1_11,
    .       IBD1_12,
    .       IBD1_13,
    .       IBD1_14,
    .      IBD1_15,
    .       IBD1_16,
    .       IBD1_17,
    .       IBD1_M

      INTEGER IBD2_1,        ! BASE DE DATOS 2.
    .       IBD2_2,
    .       IBD2_M

      INTEGER IBD3_1,        ! BASE DE DATOS 3.
    .       IBD3_2,
    .       IBD3_3,
    .       IBD3_4,
    .       IBD3_5,
    .       IBD3_6,
    .       IBD3_M

     INTEGER IBD4_1,        ! BASE DE DATOS 4.
    .       IBD4_2,
    .       IBD4_3,
    .       IBD4_4,
    .       IBD4_5,
    .       IBD4_6,
    .       IBD4_7,
    .       IBD4_8,
    .       IBD4_9,
    .       IBD4_10,
    .       IBD4_11,
    .       IBD4_12,
    .       IBD4_13,
    .       IBD4_14,
    .       IBD4_15,
    .       IBD4_16,
    .       IBD4_17,
    .       IBD4_18,
    .       IBD4_M

     INTEGER IBD5_1,        ! BASE DE DATOS 5.
    .       IBD5_2,
    .       IBD5_3,
    .       IBD5_4,
    .       IBD5_5,
    .       IBD5_6,
    .       IBD5_7,
    .       IBD5_8,
    .       IBD5_9,
    .       IBD5_M

      INTEGER IBD6_1,   ! BASE 6.
    .       IBD6_2,
    .       IBD6_M

      END MODULE ENTEROS
```

```
      SUBROUTINE EXCO(MAT,VDMANX,DENSC)

C     ------------------------------------------------------------------------
C
C     ESTA SUBROUTINA EXTRAE LOS DATOS DEL TENSOR ANISOTROPO (Co)
C     Y LA DENIDAD (Densv) DE CADA MATERIAL COMPUESTO (Mat)
c
C                                          JBLL (02/04/2011)
C     ------------------------------------------------------------------------

      USE ENTEROS; USE MATRICES; USE PARAM3D;

      IMPLICIT NONE;

      REAL(KIND=8), DIMENSION(LONG2_V3D)   :: VDMANX,DENSC
      INTEGER KTESC,INDMA,LONG2_VD3,MAT


C       CREAMOS UN FICHERO NUEVO PARA GUARDAR LOS DATOS
        OPEN(UNIT=37,FILE = 'Co.txt',STATUS='UNKNOWN',
     . POSITION='APPEND',FORM='UNFORMATTED')
        OPEN(UNIT=38,FILE = 'Co2.txt',STATUS='UNKNOWN',
     . POSITION='APPEND')

C       ESCRIBIR DATOS DEL TENSOR EN UN FICHERO NUEVO

        WRITE(38,355)MAT,VDMANX(1:LONG2_V3D),DENSV(MATNO(1))
        WRITE(37)MAT,VDMANX(1:LONG2_V3D),DENSV(MATNO(1))


C       CERRAMOS EL FICHERO Y LA SUBRUTINA
        CLOSE(37)


 355  FORMAT(I2,36F13.3,1XF8.3)

      RETURN
      END
      SUBROUTINE EXGRANDEF(INDX, IGAUS, JGAUS, LGAUS,STRSG)


C     ------------------------------------------------------------------------
C
C
C                   ESTA SUBRUTINA EXTRAE LOS DATOS DEL TENSOR DE TENSIONES EN EL CASO DE GRANDES
DEFORMACIONES:
C
C
C                   INDX-----> Elemento
C                   PGAUS----> Punto de Gauss
C                   ICARG----> Carga
C                   IINCS----> Paso
C                   STRSG----> Tensiones
C
C                                          JBLL (07/04/2011)
C     ------------------------------------------------------------------------


        USE ENTEROS; USE MATRICES; USE PARAM3D

            Integer :: INDX, IGAUS, JGAUS, LGAUS,INDMAT, ICON  V,c, PGAUS,
     .                 AUX,ICARGV,IINCSV

            Real(8) ::  DB2aux(52)


c       CREAMOS UN FICHERO NUEVO PARA GUARDAR LOS DATOS QUE NECESITAMOS EXTRAER
        OPEN(UNIT=39,FILE = 'Strsg.txt',STATUS='UNKNOWN',
     .  POSITION='APPEND',FORM='UNFORMATTED')

C     CONCATENAMOS EL PUNTO DE GAUSS PARA AHORRAR MEMORIA
            PGAUS=0
            PGAUS=(IGAUS*100+JGAUS*10+LGAUS)


C           ESCRIBIMOS LA INFORMACIÓN NECESARIA EN EL NUEVO FICHERO CREADO
            WRITE(39)INDX,PGAUS,ICARG,IINCS,STRSG


 190  FORMAT(4I6,6F13.3)


      RETURN
      END
```

```
      SUBROUTINE EXCSEC(INDX, IGAUS, JGAUS, LGAUS,DB2aux)

C     ------------------------------------------------------------------
C
C
C                 ESTA SUBRUTINA EXTRAE LOS DATOS DE  CSEC     _A
C                 RUTINA DEL PLCD. SE EXTRAE LA SIGUIENTE INFOMACIÓN:
C
C                 AUX------> Variable auxiliar que se usa en la actualización de la matriz de Rigidez
C                 INDX-----> Elemento
C                 PGAUS----> Punto de Gauss
C                 ICARG----> Carga
C                 IINCS----> Paso
C                 DB2aux---> Csec
C
C                                                 JBLL (02/04/2011)
C     ------------------------------------------------------------------


        USE E NTEROS; USE MATRICES; USE PARAM3D

            Integer :: INDX, IGAUS, JGAUS, LGAUS,INDMAT, ICONV,c, PGAUS,
     .                 AUX,INCSI,ICARGI

            Real(8) ::  DB2aux(36)


c       CREAMOS UN FICHERO NUEVO PARA GUARDAR LOS DATOS QUE NECESITAMOS EXTRAER
        OPEN(UNIT=36,FILE = 'Csec.txt',STATUS='UNKNOWN',
     . POSITION='APPEND',FORM='UNFORMATTED')
            OPEN(UNIT=66,FILE = 'Csec2.txt',STATUS='UNKNOWN',
     . POSITION='APPEND')

C       CONCATENAMOS EL PUNTO DE GAUSS PARA AHORRAR MEMORIA
            PGAUS=0
            PGAUS=(IGAUS*100+JGAUS*10+LGAUS)

            IF(JBL2.EQ.0)THEN
            AUX=1
            IINCSI=IINCS
            ICARGI=ICARG
            JBL2=2

            ENDIF


C           CREAMOS UN CONTADOR AUX QUE SE UTILIZARÁ AL ACTUALIZAR LA MATRIZ DE
C           RIGIDEZ, LOS PUNTOS CON UNA MISMA CARGA Y EN EL MISMO PASO TIENEN EL
C           MISMO VALOR, QUE VA AUMENTANDO DESDE 1 HASTA DONDE HAGA FALTA
            IF((ICARG>ICARGI).OR.(IINCS>IINCSI))THEN
                AUX=AUX+1
                IF(ICARG>ICARGI)THEN
                    ICARGI=ICARG
                ENDIF
                IF(IINCS>IINCSI)THEN
                    IINCSI=IINCS
                ENDIF
            ENDIF



C           ESCRIBIMOS LA INFORMACIÓN NECESARIA EN EL NUEVO FICHERO CREADO
            WRITE(36)AUX,INDX,PGAUS,ICARG,IINCS,DB2aux
            WRITE(66,190)AUX,INDX,PGAUS,ICARG,IINCS,DB2aux


 190  FORMAT(5I6,36F13.3)


        RETURN
        END
```

```
      SUBROUTINE MODCSEC_V3D(AUX,AUX2,DENSc,INDXB5,IGAUS,JGAUS,LGAUS) ! *-*-*

C-----------------------------------------------------------------------
C
C     CALCULA EL TENSOR CONSTITUTIVO SECANTE PARA UNA LAYER CON MEZCLA P
C
C-----------------------------------------------------------------------

      USE ENTEROS; USE MATRICES; USE PARAM3D;

          IMPLICIT NONE;

          INTEGER IM,KTESC,INDMA,ICOM,FULL,DIR,INDXB5,IGAUS,JGAUS,LGAUS

          REAL(KIND=8) TAU,PARKC,DENSC, A1, A2, A3

      REAL(KIND=8), DIMENSION(MSTR 1,MSTR1) :: AUX,AUX2,DMANX,DMANXI,
     .                          AUX1I,AUXL
      REAL(KIND=8), DIMENSION(LONG2_V3D) :: VDMANX,VDMANXI

          AUX(1:MSTR1,1:MSTR1) =0.0D0;
          AUX2(1:MSTR1,1:MSTR1) =0.0D0;

C     ----------------------------------------------------------------

          DO IM=1,NUCOMF(LPROP,KLAY)

        KPROP=LCPRO(LPROP,KLAY,IM)
        PARKc=PARTV(LPROP,KLAY,IM)
        TAU=0.0D0


C     ---------------------- PROBLEMA VISCOELASTICO ----------------------

            IF(NTINT(KPROP).E Q.7) TAU=PROPS(KPROP,50) ! Tiempo de retardo.

                KTESC = 2
                INDMA = KPROP
            CALL BASE4_V3D(LPROP,INDMA,KTESC) !LECTURA DE LA PROPIEDAD DEL MATERIAL SIMPLE

                VDMANX(1:LONG2_V3D)=VB4_V3D((lb4_4+1):(lb4_4+LONG2_V3D));


            CALL BASE2_V3D(INDXB5,IGAUS,JGAUS,LGAUS,INDMA,2,KTESC)


          VDMANX=VDMANX*(1-VB2_V3D(6))




C         ------ OBTENSION TENSOR CONSTITUTIVO DEL MATERIAL SIMPLE ------

                ICOM = 1
        full = 1 !MATRIZ ENTERA
        CALL COMPON_V3D(VDMANX,DMANX,LONG2_V3D,full,ICOM)

C         --- ROTACIÓN DEL TE NSOR CONSTITUTIVO A LAS COORDENADAS DE LA LAYERS --------

                A1 = MAEULER(KPROP,1)
                A2 = MAEULER(KPROP,2)
                A3 = MAEULER(KPROP,3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
                        dir = 2  ! LOCAL to GLOBAL
                        call ROTM(DMANX,A1,A2,A3,MSTR1,dir)
                END IF

C         ------ OBTENSION DEL TENSOR CONSTITUTIVO DE LA LAYER---------

            AUX(:,:) = AUX(:,:) + PARKc*DMANX(:,:)
            AUX2(:,:) = AUX2(:,:) + TAU*PARKc*DMANX(:,:)
            DENSc = DENSc + PARKc*PROPS(KPROP,4)

          ENDDO

      RETURN
      END
```

```
      SUBROUTINE CALCsec_V3D(INDXB5,IGAUS,JGAUS,LGAUS,VDMANX) !*-*-*
C-------------------------------------------------------------------------
C
C     ESTA RUTINA CALCULA LOS TENSORES CONSTITUTIVOS secantes y los escribe en
C          un fichero Csec.txt para el calculo de frecuencias modales
C-------------------------------------------------------------------------
      USE ENTEROS; USE MATRICES; USE PARAM3D;

      IMPLICIT NONE;

      INTEGER IGAUS,JGAUS,LGAUS,INDXB5,IEQFU,INDMA,IM,INDXB4,ICOM,ISTR1,
     .        JSTR1,KTESC,I,J, full,DIR
      REAL(KIND=8) EXISP,ETASP,PARKC,DENSC,VAL1,VAL2,DMANT
      REAL(KIND=8), DIMENSION(LONG2_V3D)   :: VDMANX,VDMANXI,VCVISC
      REAL(KIND=8), DIMENSION(MSTR1,MSTR1) :: DMANX,DMANXI,CVISC, AUX1,
     .                                                          AUX3, AUXL
      REAL (8) pvol, A1,A2,A3,DENSL

C     ----------> CALCULA EL TENSOR secante para ANISOTROPO PARA EL COMPUESTO

c     DO LPROP=1,NMATC                ! BUCLE SOBRE CADA MAT. COMP.

          DMANX(1:MSTR1,1:MSTR1) =0.0D0;   ! TENSOR ANISOTROPO
          DMANXI(1:MSTR1,1:MSTR1)=0.0D0;   ! INV. DEL TENS. ANIST.
          CVISC(1:MSTR1,1:MSTR1) =0.0D0;   ! TENSOR VISCOSO.
          DENSc=0.0D0;                ! DENSIDAD DEL MATERIAL.
          AUX1(1:MSTR1,1:MSTR1) =0.0D0;
          AUX3(1:MSTR1,1:MSTR1) =0.0D0;

               DO KLAY =1, NULAY(LPROP)          ! BUCLE SOBRE CADA LAMINA DEL COMPUESTO

                    pvol = PVLAY(LPROP,KLAY)

                    SELECT CASE (TYLAY(LPROP,KLAY))

                    CASE ('P')
                    CALL MODCsec_V3D(DMANX,CVISC,DENSL,INDXB5,IGAUS,JGAUS,LGAUS)

                    CASE ('SP')
                    CALL CLAY_SP3Dsec(DMANX,CVISC,DENSL,INDXB5,IGAUS,JGAUS,LGAUS)

                    CASE ('NT')
                    CALL CLAY_NT3D(DMANX,CVISC,DENSL)

                    END SELECT

!         --- ROTACIÓN DEL TENSOR CONSTITUTIVO LAYER A LAS COORDENADAS DEL COMPUESTO --------

                    A1 = LAEULER(LPROP,KLAY,1)
                    A2 = LAEULER(LPROP,KLAY,2)
                    A3 = LAEULER(LPROP,KLAY,3)

                    IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
                    dir = 2  ! local LAMINA to global MATERIAL COMPUESTO
                call ROTM(DMANX,A1,A2,A3, mstr1, dir)
                call ROTM(CVISC,A1,A2,A3, mstr1, dir)
                    END IF

!         --- TENSOR CONSTITUTIVO DEL COMPUESTO EN SU COORDENADAS --------

                    AUX1(:,:) = AUX1(:,:) + pvol*DMANX(:,:)
                AUX3(:,:) = AUX3(:,:) + pvol*CVISC(:,:)
c               DENSc = DENSc + pvol*DENSL

               END DO

               DMANX=AUX1
               CVISC=AUX3

               CALL INVERT3D(MSTR1,DMANX,DMANXI,AUXL)

c         DENSV(MATNO(1))=DENSc;
C     ------------------ DESCOMPONE  TENSOR ------>  VECTOR ----------------------

          ICOM=2
       full=1  ! Complete Matrix
       CALL COMPON_V3D(VDMANX ,DMANX ,LONG2_V3D,full,ICOM)

      RETURN

      END
```

```
Subroutine CLAY_SP3Dsec(Ccomp, Cvisco, Dens,INDXB5,IGAUS,JGAUS,LGAUS)

! This subroutine obtains the Composite's Constitutive Tensor
! of composite COMP and layer LAY. It also obtains the Viscous
! tensor (if required)

  Use ENTEROS; Use MATRICES; Use PARAM3D;
        Implicit None

        Integer :: comp, visco, dir,KTESC,INDMA,ICOM,FULL,INDXB5,IGAUS,JGAUS,LGAUS
        Real(8) :: mK, fK, mTau, fTau, Dens, A1, A2, A3

        Real(8), Dimension(mstr1,mstr1) :: Cmat, Cfib, Ccomp, Cvisco, Aux
        REAL(KIND=8), DIMENSION(LONG2_V3D)        ::  VDMANX,VDMANXI

! Initialize some variables
        visco      = 0
    Cvisco(:,:) = 0.0d0
    Dens       = 0.0d0

! Obtain MATRIX Constitutive Tensor

    kprop = LCPRO(LPROP,KLAY,1)
    mK    = PARTV(LPROP,KLAY,1)
    Dens  = Dens + mK*props(kprop, 4)

        KTESC = 2
        INDMA = KPROP
    CALL BASE4_V3D(LPROP,INDMA,KTESC) !LECTURA DE LA PROPIEDAD DEL MATERIAL SIMPLE

        VDMANX(1:LONG2_V3D)=VB4_V3D((lb4_4+1):(lb4_4+LONG2_V3D));

     CALL BASE2_V3D(INDXB5,IGAUS,JGAUS,LGAUS,INDMA,2,KTESC)
        VDMANX=VDMANX*(1-VB2_V3D(6))



    ICOM = 1
    full = 1 !MATRIZ ENTERA
        CALL COMPON_V3D(VDMANX,Cmat,LONG2_V3D,full,ICOM)

!       --- ROTACIÓN DEL TENSOR CONSTITUTIVO MAT A LAS COORDENADAS DE LA LAYERS --------

                A1 = MAEULER(KPROP,1)
                A2 = MAEULER(KPROP,2)
                A3 = MAEULER(KPROP,3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
                        dir = 2  ! LOCAL to GLOBAL
                        call ROTM(Cmat,A1,A2,A3,MSTR1,dir)
                END IF

! Obtain viscous parameter if problem is viscoelastic

     if(ntint(kprop) .eq. 7) then
              mTau  = props(kprop,50)
              visco = 1
            else
              mTau  = 0.0d0
            endif

! Obtain FIBRE constitutive Tensor

    kprop = LCPRO(LPROP,KLAY,2)
    fK    = PARTV(LPROP,KLAY,2)
    Dens  = Dens + fK*props(kprop, 4)

        KTESC = 2
        INDMA = KPROP
    CALL BASE4_V3D(LPROP,INDMA,KTESC) !LECTURA DE LA PROPIEDAD DEL MATERIAL SIMPLE

        VDMANX(1:LONG2_V3D)=VB4_V3D((lb4_4+1):(lb4_4+LONG2_V3D));

        CALL BASE2_V3D(INDXB5,IGAUS,JGAUS,LGAUS,INDMA,2,KTESC)
        VDMANX=VDMANX*(1-VB2_V3D(6))

    ICOM = 1
    full = 1 !MATRIZ ENTERA
        CALL COMPON_V3D(VDMANX,Cfib,LONG2_V3D,full,ICOM)

!       --- ROTACIÓN DEL TENSOR CONSTI TUTIVO FIB A LAS COORDENADAS DE LA LAYERS --------+
```

```
+

                A1 = MAEULER(KPROP,1)
                A2 = MAEULER(KPROP,2)
                A3 = MAEULER(KPROP,3)

                IF(A1**2+A2**2+A3**2.GT.1.e-9) Then
                    dir = 2  ! LOCAL to GLOBAL
                    call ROTM(Cfib,A1,A2,A3,MSTR1,dir)
                END IF

! Obtain viscous parameter if problem is viscoelastic

    if(ntint(kprop) .eq. 7) then
          fTau  = props(kprop,50)
          visco = 1
        else
          fTau  = 0.0d0
        endif

! Obtain LAYER SP Constitutive Tensor IN LOCAL COORDENATE

   call Ccomp_SP(mK,Cmat,fK,Cfib,mstr1,Ccomp)
! In case of viscous problem

    if(visco .eq. 1) &
     call  Ccomp_SP(mk*mTau,Cmat,fK*fTau,Cfib,mstr1,Cvisco)

 Return
End
```

CÓDIGOS MODIFICADOS (CFYFP)

```
      Subroutine Eigen_SIM(A,B,new_size,sval,svec,neig,toler,maxi,iout, &
                  iiter, StVer,sturm1, sturm2,   pred,spred,R
!     ------------------------------------------------------------------
!           THIS SOUBROUTINE OBTAINS THE neig SMALLEST EIGENVALUES AND
!           EIGENVECTORS OF THE GENERALIZED EIGENVALUE PROBLEM:
!
!                                                       A·x = a·B·x
!
!           The eigenvalues and eigenvectors are obtained using the Subspace
!           Iteration Method (SIM).
!           This subroutine is an adaptation from the subroutine SSPACE that
!           is described in Bathe, K.J., "Finite Element Procedures" pages
!           966 to 977.
!
!           The data that has to be supplied to the subroutine is:
!
!                A --------> Stiffness matrix. Symmetric Sparse stored by rows.
!                B --------> Mass matrix. Symmetric Sparse stored by rows.
!                new_size -> New size of A matrix when it's factorized.
!                sval -----> Array with the nc eigenvalues used
!                svec -----> Matrix with the nc eigenvectors used eigvec(n,nc)
!                neig -----> Number of eigenvalues and eigenvectors sought
!                toler ----> Tolerance of the solution
!                maxi -----> Maximum number of iterations allowed
!    iiter ----> If there is a problem during calculation, iiter
!          is sent with a code to write the problem found
!          (values and problems are exposed in Eigen_Write.f90)
!
!     ------------------------------------------------------------------

   Use Sparse_SCSR
          Implicit NONE

!     Input and output variables
          Type(SCSR)    :: A, B, LA
          Integer                              :: neig, maxi, new_size,new_n, iiter, iout,  &
                  ipred, StVer, sturm1, sturm2,   pred,R
          Real(kind=8) :: sval(neig),svalv(neig), svec(A%n,neig), toler,&
                                        spred(A%n,R)

! Code variables
          Integer                              :: i, j, ij, ix, k, ichol
          Integer                              :: nc, nnc, iconv
          Real(kind=8) :: tol_jacob, ray, error

   Real(kind=8), allocatable :: yk1(:,:), xk1p(:,:), xk2p(:,:)
   Real(kind=8), allocatable :: Ar(:), Br(:), Qk(:,:), Vk(:)
```

```
!       Definition of main Variables to be used during the computation
              ipred  = 0
              iconv  = 0
              iiter  = 1

!       Define allocatable dimensions
              nc = min(A%n, 2*neig, neig+8) ! Number of eigenvalues sought with SIM method
              nnc = nc*(nc+1)/2          ! Subspace matrix size (symmetric)

!       Allocates main variables to be used
              Allocate(yk1(A%n,nc), xk1p(A%n,nc), xk2p(A%n,nc))
              Allocate(Ar(nnc), Br(nnc), Qk(nc,nc), Vk(nc))


! Cholesky decomposition of A matrix
   if (new_size .eq. 0) then
        new_n=new_size
        call SIM_Symbol(A, new_n)                                   ! Symbolic Factorization
        endif
              call SIM_Chol(A,LA,new_n,iout)        ! Numerical Factorization
              if (iout .eq. -1) Return

! Loop to permit consider different initial predictions if any of them does not
! manage to find the eigenvalues sought
   DO ipred = 1, 2

!       Obtains initial prediction of the eigenvectors
              yk1(:,:) = 0.0d0
              if(pred.eq.1) then
              yk1=spred
              go to 50
              endif

              if (ipred .eq. 1) then
              call SIM_Prediction(A, B, nc, yk1, svec, neig, iout)
                iout = 1
          else
              call SIM_Prediction2(A, B, nc, yk1, svec, neig)
              iout = 1
             endif

!       Start the Main Iteration Loop
50            DO WHILE (iconv .ne. 1)

                    if (iiter .gt. maxi) then
                          iout = -3
                          exit
                    endif

! Calculate projection: LA --> Ar
       do i = 1, nc
                    call SIM_Solver(LA, yk1(:,i), xk1p(:,i))
              enddo
              call SIM_Matmat(xk1p, yk1, Ar, LA%n, nc, nnc)

! Calculate projection: B --> Br
                    ichol = 0             ! ichol = 0 --> B matrix is NOT factorized
                    do i = 1, nc
                    call SIM_Matvec(B, xk1p(:,i), xk2p(:,i), ichol)
                    enddo
                    call SIM_Matmat(xk1p,xk2p,Br,B%n,nc,nnc)

! Solve for eigensystem of subspace operators
                    tol_jacob = toler**2
                    call SIM_Jacobi(Ar, Br, Qk, Vk, tol_jacob, nc, nnc, iout)
                    if (iout .lt. 0) then
                      exit
                    endif

! Arrange Eigenvalues in ascending order
                    call SIM_order(Qk, Vk, nc)

! Obtain aproximate eigenvectors yk1 = xk1p * Qk
                    yk1(:,:) = 0.0d0
                    do i = 1, LA%n
                          do j = 1, nc
                                do k = 1, nc
                                      yk1(i,j) = yk1(i,j) + xk1p(i,k)*Qk(k,j)
```

```
                                enddo
                            enddo
                        enddo

! Convergence Checking Using Rayleigh parameter
                        iconv = 1
                        do i = 1, neig
                        call SIM_Rayleigh(LA, B, Vk(i), yk1(:,i), ray)
                            error = abs( (Vk(i) - ray) / Vk(i) )
                            if (error .gt. toler) then
                                iconv = 0
                                exit
                            endif
                        enddo

100                     iiter = iiter + 1
                    ENDDO

! Check if convergence has been reached. If not, tries 2nd prediction if it has
! been tested with prediction 1 or leaves the subroutine without solution
                    if (iconv .eq. 1) then
                     !exit
                        if ((abs(svalv(neig)-Vk(neig))/Vk(neig)).lt.toler)then
                        exit
                        else
                        svalv(neig)=Vk(neig)
                        iconv=0
                        go to 100
                        endif
                    else
                    if (ipred .eq. 2) then
                            iout = -2
                                Return
                      endif
                     endif
                    ENDDO

! Loop for convergence has endded
   call Deallocate_Matrix(LA)

! Sturm Sequence Check --> Only does it if eigenvalues are computed
!           if ((iiter .gt. 0) .and. (StVer .ge. 1)) then
!     call SIM_Sturm(A,B,new_n,Vk(1:neig),neig,toler,StVer,sturm1,sturm2)
!   endif

!       Return neig eigenvalues and eigenvectors
                sval(:)         = Vk(1:neig)
                svec(:,:) = yk1(:,1:neig)
                spred(:,:)= yk1(:,:)
                Return
        End



        Subroutine SIM_Prediction(A, B, nc, ev, preev, pren,iout)
!*****************************************************************************
!   This subroutine makes a first prediction of the eigenvectors
!   These will be used to start the iteration method
!
!*****************************************************************************

   USE Sparse_SCSR
                Implicit NONE
                type(SCSR)    :: A, B
                Integer                                 :: nc, pren, iout
                Integer                                 :: i, j, l, ij, ix, k, nd, diag
                Real(kind=8) :: rt, pi, xx
                Real(kind=8) :: ev(A%n,nc), preev(A%n,pren), &
                        w(A%n), tt(A%n)

! Compute vector with B(diag)/A(diag) terms
   do i = 1, A%n
                w(i)       = B%Val(B%i(i+1)-1)/A%Val(A%i(i+1)-1)
          enddo
                nd = int(A%n/nc)

! Compute first pren prediction vectors
                if (iout .gt. 0) then                   ! If eigenvalues have been already
           ev(:,1:pren) = preev(:,:)                ! computed, use them as prediction
```

```
                else
    !   First vector = B diagonal
                do i = 1, A%n
                        ev(i,1) = B%Val(B%i(i+1)-1)
                 enddo

    ! Rest of the vectors = Put +1 o  n bigger Bii/Aii values
                 l = A%n - nd
                 do j = 2, nc-pren
                        rt = 0.0d0
                        do i = 1, l


                                if (abs(w(i)) .lt. rt) cycle
                                rt = abs(w(i))
                                ij = i
                        enddo
                        do i = l, A%n
                                if (abs(w(i)) .le. rt) cycle
                                rt = abs(w(i))
                                ij = i
                        enddo
                        tt(j)                   = float(ij)
                        w(ij)                   = 0.0d0
                        l                                       = l - nd
                        ev(ij, j) = 1.0d0
                enddo
  endif

! Compute nc-pren last prediction vectors
                l = (A%n - (pren*nd))
                        do j = nc-pren+1, nc
                        rt = 0.0d0
                 do i = 1, l
                                if (abs(w(i)) .lt. rt) cycle
                                rt = abs(w(i))
                                ij = i
                        enddo
                        do i = l, A%n
                                if (abs(w(i)) .le. rt) cycle
                                rt = abs(w(i))
                                ij = i
                        enddo
                        tt(j)                   = float(ij)
                        w(ij)                   = 0.0d0
                        l                                       = l - nd
                        ev(ij, j) = 1.0d0
                enddo

! Last Vector = Random vector
                pi = 2*dacos(0.0d0)
                xx = 0.5d0
                do k = 1, A%n
                        xx                              = (pi + xx)**5
                        ix                              = int(xx)
                        xx                              = xx - float(ix)
                        ev(k,nc)        = ev(k,nc) + xx
                enddo

                Return
        End
```

```
      Subroutine SIM_Prediction2(A, B, nc, ev, preev, pren)
!*************************************************************************
!  This subroutine makes a first prediction of the eigenvectors
!  These will be used to start the iteration method
!
!*************************************************************************
       USE Sparse_SCSR
            Implicit NONE
            type(SCSR)    :: A, B
            Integer                                  :: nc, pren, first, last
            Integer                                  :: i, j, l, ij, ix, k, nd, diag
            Real(kind=8) :: rt, pi, xx
            Real(kind=8) :: ev(A%n,nc),preev(A%n,pren),
   .           w(A%n), tt(A%n)

C       Compute vector with B(diag)/A(diag) terms
        do i = 1, A%n
             w(i)      = B%Val(B%i(i+1)-1)/A%Val(A%i(i+1)-1)
        enddo
             nd = int(A%n/nc)



C       Compute first pren prediction vectors
C       First vector = B diagonal
       do i = 1, A%n
        ev(i,1) = B%Val(B%i(i+1)-1)
        enddo

  ! Rest of the vectors = Put +1 on bigger Bii/Aii values
       l = A%n - nd
       do j = 2, nc-1
            rt = 0.0d0
            first = (j-1)*nd
            last  = min(j*nd,A%n)
           ! ij=j
            do i = first, last
                 if (abs(w(i)) .le. rt) cycle
                 rt = abs(w(i))
                 ij = i
            enddo
            tt(j)                = float(ij)
            w(ij)                = 0.0d0
            ev(ij, j) = 1.0d0
        enddo

C       Last Vector = Random vector
            pi = 2*dacos(0.0d0)
            xx = 0.5d0
            do k = 1, A%n
                 xx                          = (pi + xx)**5
                 ix                          = int(xx)
                 xx                          = xx - float(ix)
                 ev(k,nc)       = ev(k,nc) + xx
            enddo
            Return
       End
```

```
      SUBROUTINE ALMACEN(STIF,NTOT,A)

C     ------------------------------------------------------------------------------------
C
C
C                                 SUBRUTINA QUE DADA UNA MATRIZ LA ALMACENA EN FORMATO SPARSE
C
C                 Se realiza el almacenamiento en formato Sparse de una matriz completa por filas.
C
C                 EL SIGNIFICADO DE LAS VARIABLES UTILIZADAS ES LA SIGUIENTE:
C
C                       A%n ----> Número de filas/columnas de la matriz
C                       A%nnz --> Dimensión del vector que incluye todos los elementos no nulos de la matriz
C                       A%Val --> Valores de la matriz en formato Sparse por filas (matriz entera), dimensión =
A%Val(A%nnz)
C                       A%j ----> Posición columna de los valores de la matriz A, dimensión = A%j(A%nnz)
C                       A%i ----> Posición del vector A%Val en la que empieza una nueva fila de la matriz, contiene un
elemento
C                                 más de los estrictamente necesarios debido al funcionamiento de las rutinas
siguientes.
C
C     ------------------------------------------------------------------------------------

      Use Sparse_SCSR

      IMPLICIT NONE

      TYPE(SCSR) ::A

      INTEGER::COL,FIL ,NTOT,AUX,p,o
      REAL(KIND=8), DIMENSION(NTOT,NTOT)  ::STIF


C     HAGO UN PRIMER RECORRIDO SOBRE LA MATRIZ PARA CONOCER LAS DIMENSIONES DE LA MATRIZ
      AUX=0
      DO FIL=1,NTOT
            DO COL=1,NTOT
                  IF (STIF(FIL,COL).NE.0) THEN
                   AUX=AUX+1
                  ENDIF
            ENDDO
      ENDDO

C     ALOCO LAS DIMENSIONES DE LAS VARIABLES A UTILIZAR
      CALL allocate_SCSR (A, NTOT, AUX)

C     RECORRO LA MATRIZ POR SEGUDA VEZ PARA ALMAMCENARLA EN FORMATO SPARSE
      p=1
      o=1
      DO FIL=1,NTOT
            AUX=1
            DO COL=1,NTOT
             IF (STIF(FIL,COL).EQ.0) THEN
             ELSE
                  A%Val(p)=STIF(FIL,COL)
                  A%j(p)=COL
                  p=p+1
                  IF (AUX.EQ.1 ) THEN
                        A%i(o)=p-1
                        o=o+1
                        AUX=2
                  ENDIF
             ENDIF
            ENDDO
      ENDDO

C     DAMOS VALOR AL ÚLTIMO ELEMENTO DEL VECTOR DE POSICIONAMIENTO A%i
C     PARA QUE FUNCIONEN LAS RUTINAS SUCESIVAS TIENE QUE VALER A%nnz+1

      A%i(A%n+1)=A%nnz+1


      RETURN
      END
```

```
      SUBROUTINE ANIMACION(VEC,F,NTOTV,CPROB)

C           RUTINA QUE DADO UN VECTOR PROPIO CREA LOS DATOS NECESARIOS PARA
C
C                       OBTENER UNA ANIMACIÓN DE LA FORMA PROPIA
C

      IMPLICIT NONE


      INTEGER::AUX2,AUX3,F,I,II,III,J,NGPOIN,NTOTV
      REAL(KIND=8)::A
      REAL(KIND=8),DIMENSION(NTOTV)    ::VEC,VECNORM,VECMIN,VECPAS
      CHARACTER(LEN=33) CPROB
      CHARACTER(LEN=50) NOM
      CHARACTER(LEN=45) NOMBRE1,NOMBRE2
      CHARACTER(LEN=12) ELTYP,GAUSNAM,MESHNAM
      CHARACTER(LEN=10) TERM
      CHARACTER(LEN=2)  FCHAR


      IF(F.EQ.0)THEN
      return
      ENDIF

      A=MAXVAL(ABS(VEC))

      VECNORM=VEC/A
      VECMIN=VECNORM*(-1)
      VECPAS=VECNORM/10

C     BUSCAR EL TAMAÑO EXACTO DE LAS CADENAS PARA NO ESCRIBIR ESPACIOS
C     EN BLANCO EN EL NOMBRE DE LOS FICHEROS DE SALIDA

            DO J=1,33

         IF (CPROB(J:J).EQ.'.') II=(J-1);

            ENDDO


            TERM='.post.res'

            IF (F<10) THEN
            write(FCHAR,'(I1)')F
            III=1
            ELSE
            write(FCHAR,'(I2)')F
            III=2
            ENDIF

            NOM='Animacion-F'//FCHAR(1:III)//'-'//CPROB(1:II)//TERM


            open(unit=24,file=NOM,status='unknown')


C     INFORMACIÓN SOBRE EL TIPO DE ELEMENTOS

      IF(N.EQ.8) THEN

            ELTYP   ='HexaHedra'

            MESHNAM ='"HEXA-G8"'

            GAUSNAM ='"HEXA-G8"'

            NGPOIN  = 8

      ELSE

            ELTYP   ='TetraHedra'

            MESHNAM ='"TETRA-G1"'

            GAUSNAM ='"TETRA-G1"'

            NGPOIN  = 1

      ENDIF
```

```
C       ESCRITURA ENCABEZAMIENTO DEL FICHERO PARA GID
        WRITE(24,*) 'GiD Post Results File 1.0'
        WRITE(24,*)
        WRITE(24,310) 'GaussPoints',GAUSNAM,'Elemtype',ELTYP,MESHNAM
        WRITE(24,320) 'Number of Gauss Points: ',NGPOIN
        WRITE(24,*) 'Natural Coordinates   : Internal'
        WRITE(24,*) 'End GaussPoints'
        WRITE(24,*)

        DO I=1,20
              AUX2=1
              AUX3=1
              NOMBRE1 = «"ANIMACIÓN'
              NOMBRE2 = 'ª FRECUENCIA" “DISPLACEMENTS'"
              WRITE(24,101) NOMBRE1,F,NOMBRE2,I
              WRITE(24,*) 'ComponentNames ','"DX" ','"DY "',"DZ"'
              WRITE(24,*) 'Values'
        WRITE(24,110)AUX3,VECMIN(AUX2),VECMIN(AUX2+1),VECMIN(AUX2+2)
              AUX2=AUX2+3
              AUX3=AUX3+1
          DO AUX3=2,(NTOTV/3)
        WRITE(24,110)AUX3,VECMIN(AUX2),VECMIN(AUX2+1),VECMIN(AUX2+2)
              AUX2=AUX2+3
          ENDDO
              WRITE(24,*) 'End Values'
              VECMIN=VECMIN+VECPAS
        ENDDO

        DO I=21,40
              AUX2=1
              AUX3=1
              NOMBRE1 = «"ANIMACIÓN'
              NOMBRE2 = 'ª FRECUENCIA" “DISPLACEMENTS'"
              WRITE(24,101) NOMBRE1,F,NOMBRE2,I
              WRITE(24,*) 'ComponentNames ','"DX" ','"DY "',"DZ"'
              WRITE(24,*) 'Values'
        WRITE(24,110)AUX3,VECMIN(AUX2),VECMIN(AUX2+1),VECMIN(AUX2+2)
              AUX2=AUX2+3
              AUX3=AUX3+1
          DO AUX3=2,(NTOTV/3)
        WRITE(24,110)AUX3,VECMIN(AUX2),VECMIN(AUX2+1),VECMIN(AUX2+2)
              AUX2=AUX2+3
          ENDDO
              WRITE(24,*) 'End Values'
```

```
            VECMIN=VECMIN-VECPAS

        ENDDO


C       FORMATOS UTILIZADOS

 110 FORMAT(I5,1X,E25.15,1X,E25.15,1X,E25.15)

 101 FORMAT(6HResult,1X,A11,I2,A32,1X,I3,1X,
    .       6HVector,1X,7HOnnodes)

 310 FORMAT(1X,A11,2X,A12,5X,A8,3X,A12,3X,A12)

 320 FORMAT(1X,A24,I3)


        RETURN

        END


        SUBROUTINE ESCRIBIR(EIGENFREQ,NEIG,NTOTV,SVECC,DAN,I,AUX,CPROB,N)

C       ------------------------------------------------------------------------------------------
C
C                                       SUBRUTINA QUE SE ENCARGA DE ESCRIBIR LOS RESULTADOS
C                                       EN UN FICHERO PARA SU POSTERIOR POSTPROCESADO
C
C               NOTAS:
C               *Es una rutina pensada para hexaedros de 8 nodos o tetraedros de 4.
C               *I--> sirve para escribir el encabezado solamente la primera vez, nos indica si es el
C                caso elástico o si la estructura ya ha sido dañada.
C               *Aux--> indica el caso en elque nos encontramos, el 1 será el caso base y los demás
C                     irán sumando valores, cada valor diferente representa una actualización de la
C                           matriz de rigidez.
C               *N--> vale 8 si es un problema con hexaedros y 4 si son tetraedros.
C
C       ------------------------------------------------------------------------------------------

        IMPLICIT NONE

        INTEGER::AUX,AUX2,AUX3,B,I,II,J,NEIG,NGPOIN,NTOTV,C,N
        REAL(KIND=8)::A,STEPV,DAN
        REAL(KIND=8), DIMENSION(NTOTV,NEIG) ::SVECC,AUXFORM
        REAL(KIND=8), DIMENSION (NEIG)                  ::EIGENFREQ
        CHARACTER(LEN=12) ELTYP,GAUSNAM,MESHNAM
        CHARACTER(LEN=45) NOMBRE1,NOMBRE2
        CHARACTER(LEN=33) CPROB
        CHARACTER(LEN=68),DIMENSION(2)  ::NOM
        CHARACTER(LEN=10),DIMENSION(2)  ::TERM

C       NORMALIZAR LAS FORMAS PROPIAS DE TAL FORMA QUE EL VALOR MÁXIMO SEA 1

        DO B=1,NEIG

            A=MAXVAL(SVECC(:,B))

            C=MINVAL(SVECC(:,B))

            IF(ABS(A).GT.ABS(C))THEN

            AUXFORM(:,B)=(SVECC(:,B))/A

            SVECC(:,B)=AUXFORM(:,B)

            ELSE

            AUXFORM(:,B)=(SVECC(:,B))/C

            SVECC(:,B)=AUXFORM(:,B)

            ENDIF

        ENDDO
```

```
      IF(I.EQ.1)THEN
      GOTO 100
      ENDIF

C     CREACIÓN NOMBRE ARCHIVOS DE SALIDA

            DO J=1,33

         IF (CPROB(J:J).EQ.'.') II=(J-1);

            ENDDO

            TERM(1)='.post.res'
            TERM(2)='.res'

            NOM(1)=CPROB(1:II)//'F'//TERM(1)
            NOM(2)='Frecuencia-daño-'//CPROB(1:II)//TERM(2)

            open(unit=4,file=NOM(1),status='unknown')
            open(unit=5,file=NOM(2),status='unknown')

C     INFORMACIÓN SOBRE EL TIPO DE ELEMENTOS (HEXAEDROS O TETRAEDROS)

      IF(N.EQ.8) THEN

            ELTYP   ='HexaHedra'

            MESHNAM ='"HEXA-G8"'

            GAUSNAM ='"HEXA-G8"'

            NGPOIN  = 8

      ELSE

            ELTYP   ='TetraHedra'

            MESHNAM ='"TETRA-G1"'

            GAUSNAM ='"TETRA-G1"'

            NGPOIN  = 1

      ENDIF

C     ESCRITURA ENCABEZAMIENTO DEL FICHERO PARA GID

      WRITE(4,*) 'GiD Post Results File 1.0'

      WRITE(4,*)

      WRITE(4,310) 'GaussPoints',GAUSNAM,'Elemtype',ELTYP,MESHNAM

      WRITE(4,320) 'Number of Gauss Points: ',NGPOIN

      WRITE(4,*) 'Natural Coordinates   : Internal'

      WRITE(4,*) 'End GaussPoints'

      WRITE(4,*)

C     ESCRITURA ENCABEZAMIENTO DEL FICHERO DAÑO-FRECUENCIA

      WRITE(5,*)'Tabla de relación frecuencias VS. daño'

      WRITE(5,*)

      WRITE(5,*)

C     ESCRITURA RESULTADOS

100   STEPV = AUX

      DO J=1,NEIG

      AUX2=1
      AUX3=1
```

```
            NOMBRE1 = «"DEFORMADA,'
            NOMBRE2 = "ª FRECUENCIA" "DISPLACEMENTS"'

            WRITE(4,101) NOMBRE1,J,NOMBRE2,STEPV

            WRITE(4,*) 'ComponentNames ','"DX" ','"DY '","DZ"'

            WRITE(4,*) 'Values'

            WRITE(4,110)AUX3,SVECC(AUX2,J),SVECC(AUX2+1,J),SVECC(AUX2+2,J)

            AUX2=AUX2+3

            AUX3=AUX3+1

          DO AUX3=2,(NTOTV/3)

            WRITE(4,110)AUX3,SVECC(AUX2,J),SVECC(AUX2+1,J),SVECC(AUX2+2,J)

            AUX2=AUX2+3

          ENDDO

            WRITE(4,*) 'End Values'

        ENDDO

            WRITE(5,502) DAN,(EIGENFREQ(J),J=1,NEIG)

C       FORMATOS UTILIZADOS

 110 FORMAT(I5,1X,E25.15,1X ,E25.15,1X,E25.15)

 101 FORMAT(6HResult,1X,A11,I2,A32,1X,F12.6,1X,
    .       6HVector,1X,7HOnnodes)


 310 FORMAT(1X,A11,2X,A12,5X,A8,3X,A12,3X,A12)

 320 FORMAT(1X,A24,I3)

 501 FORMAT(4X,A5,5X,5(A14,2X))

 502 FORMAT(1X,F12.10,1X,10(F15.10,1X))


        RETURN

        END
        SUBROUTINE VectProp(svec,NEIG,CRED,NTOTV,SVECC,VREST)

C       ------------------------------------------------------------------------------
C
C                                       SUBRUTINA QUE DADO UN VECTOR PROPIO CON LOS VALORES DE LOS
C                                               NODOS NO RESTRINGIDOS, DEVUELVE EL VECTOR COMPLETO
C       ------------------------------------------------------------------------------

        IMPLICIT NONE

        INTEGER::NEIG,CRED,NTOTV,AUX1,AUX2,AUX3
        REAL(KIND=8), DIMENSION(CRED,NEIG)  ::SVEC
        REAL(KIND=8), DIMENSION(NTOTV,NEIG) ::SVECC
        REAL(KIND=8), DIMENSION (NTOTV)         ::VREST


        SVECC(:,:)=0.0

        DO AUX1=1,NEIG

         AUX3=1

         DO AUX2=1,NTOTV
                IF (VREST(AUX2).EQ.1) THEN
                        SVECC(AUX2,AUX1)=0
                ELSE
                        SVECC(AUX2,AUX1)=SVEC(AUX3,AUX1)
                        AUX3=AUX3+1
                ENDIF

         ENDDO

        ENDDO

        RETURN
        END
```

```
      PROGRAM CFYFP

C-------------------------------------------------------------------------
C
C                                   RUTINA PRINCIPAL DEL CFYFP
C
C
C                              Cálculo de Frecuencias Y Formas Propias
C
C
C
C     Esta rutina se encarga de procesar los datos extraídos del PLCD
C     y adecuarlos a la rutina implementada por Xavier Martínez para el
C     cálculo de valores y formas propias utilizando el SIM (Subspace
C     Iteration Method). Utilizando esas rutina calcula las frecuencias y
C     las formas propias de las estructuras. Finalmente se escribe los valores
C   obtenidos en varios ficheros de salida para su posterior postproceso.
C                                          JBLL 03/06/2011
C
C-------------------------------------------------------------------------

      Use Sparse_SCSR; USE ENTEROS; USE MATRICES; USE PARAM3D;
      IMPLICIT NONE;

C     DECLARACIÓN VARIABLES

      INTEGER::A,AUX,AUX3,AUX4,AUX5,AUX6,AUX7,AUXRED,AUXV,CARG,COMP,
     .      CONTROL,CRED,CTRL,CTRL2,CTRL3,CTRL4,CTRL6,DIR,ELAST,ELDIS,
     .      ELEMC,ELEMV,EMAT,FANIM,GLOBC,GLOBF,I,I2,IDIME,IGAUS,
     .      IGAUSL,ELEMF,ELEMO,iitersim,IKON,IMAT,INDX,INDXB  5,INODE,
     .      IOUTSIM,IPOIN,JGAUS,JGAUSL,JMAT,LGAUS,LGAUSL,MAXI,NC,NEIG,
     .      new_size,PASON,PASOV,PGAUS,RES,RESTX,RESTY,RESTZ,STURM1,
     .      STURM2,pred,npred,TMAT,AESTIF,IAUX


      REAL(8)::AUX1,AUX2,DESPX,DESPY,DESPZ,DJACB,DVOL,ETASP,EXISP,EZETA,
     .       PESO,PESO1,TOLERSIM,DAN,A1,A2,A3

      TYPE(SCSR) ::SPARK,SPARM

      REAL(KIND=8), DIMENSION(6)          :: STRSG
      REAL(KIND=8), DIMENSION(8)          :: F,C
      REAL(KIND=8), DIMENSION(36)         :: VCTAN
      REAL(KIND=8), DIMENSION(3,3)     :: FGRAD,XJACM
      REAL(KIND=8), DIMENSION(6,6)        :: CO,CTAN
      REAL(KIND=8), DIMENSION(3,36)    :: GPCOD
      REAL(KIND=8), DIMENSION(6,24)       :: BMATX
    REAL(KIND=8), DIMENSION(24,24)    :: ESTIFO


      REAL(KIND=8), ALLOCATABLE        ::COOR(:,:),DENS(:),
     .                                   EIGENFREQ(:),ESTIFES(:,:),
     .                                   MATEULER(:,:),MAS(:,:),
     .                                   MASAUX(:,:),MASRED(:,:),
     .                                   MASRED2(:,:),MAT(:),
     .                                   RESTR(:,:),spred(:,:),
     .                                   STIFAUX(:,:),STIFFGP(:,:),
     .                                   STIFFGO(:,:),STIFRED(:,:),
     .                                   STIFRED2(:,:),SVAL(:),
     .                                   SVEC(:,:),SVECC(:,:),
     .                                   VAUX(:),VCO(:,:),VECB(:),
     .                                   VELEM(:),VPGAUS(:),
     .                                   VRESTR(:),AUXELEM(:)



C     ABRIR ARCHIVOS DE ENTRADA

      OPEN(UNIT=41,FILE='Co.txt', STATUS='UNKNOWN',
    .POSITION='REWIND',FORM='UNFORMATTED')
      OPEN(UNIT=42,FILE='Csec.txt', STATUS='UNKNOWN',
    .POSITION='REWIND',FORM='UNFORMATTED')
      OPEN(UNIT=56,FILE='Datosgen.txt', STATUS='UNKNOWN',
    .POSITION='REWIND')
      OPEN(UNIT=57,FILE='Strsg.txt', STATUS='UNKNOWN',
    .POSITION='REWIND',FORM='UNFORMATTED')
      OPEN(UNIT=58,FILE='Daño.txt', STATUS='UNKNOWN',
    .POSITION='REWIND')
```

```
C       LEER LOS DATOS DE ENTRADA GENERALES
        READ(56,*)CPROB
        READ(56,*)TYPEP
        READ(56,*)NPOIN,NELEM,NMATC,LARGE
        READ(56,*)NNODE
        NTOTV=NPOIN*3;


C       ALOCAMOS ALGUNAS DE LAS VARIABLES TODAVÍA SIN DIMENSIÓN


        CALL ALOKA_V3D

        ALLOCATE(CON(NELEM,NNODE));
        ALLOCATE(COOR(NPOIN,3));
        ALLOCATE(DENS(NMATC));
        ALLOCATE(MAS(NTOTV,NTOTV));
        ALLOCATE(MASAUX(NTOTV,NTOTV));
        ALLOCATE(MASRED(NTOTV,NTOTV));
        ALLOCATE(MAT(NELEM));
        ALLOCATE(RESTR(NPOIN,4));
        ALLOCATE(STIFAUX(NTOTV,NTOTV));
        ALLOCATE(STIFFGP(NTOTV,NTOTV));
        ALLOCATE(STIFFGO(NTOTV,NTOTV));
        ALLOCATE(STIFRED(NTOTV,NTOTV));
        ALLOCATE(VBAS6(2,NPOIN));
        ALLOCATE(VB6(2));
        ALLOCATE(VCO(NMATC,36));
        ALLOCATE(VECB(9*NNODE*NNODE));
        ALLOCATE(VRESTR(NTOTV));
        ALLOCATE(MATEULER(NELEM,3));
        ALLOCATE(AUXELEM(NELEM));

C       LEER CONECTIVIDADES

                DO IELEM=1,NELEM
                        READ(56,*)ELEM,EMAT,NTYPE,EFASE,NNODE,NGAUS,NINTE,
     .                  CON(ELEM,:),MATEULER(ELEM,1),MATEULER(ELEM,2),
     .                  MATEULER(ELEM,3)
                        MAT(ELEM)=EMAT
                ENDDO

        CALL TYPEV3D


C       LEER COORDENADAS DE LOS NODOS

         DO I=1,NPOIN
          READ(56,*)A,COOR(I,:)
         ENDDO

C       LEER RESTRICCIONES
         RESTR=0.0
         DO WHILE(.NOT.EOF(56))
                READ(56,*)IPOIN,RES,DESPX,DESPY,DESPZ
                RESTR(IPOIN,1)=RES
                RESTR(IPOIN,2)=DESPX
                RESTR(IPOIN,3)=DESPY
                RESTR(IPOIN,4)=DESPZ
         ENDDO

                NGLIBA=0

      DO IPOIN=1,NPOIN

        NGLIBL=3

        NGLIBA=NGLIBA+NGLIBL

        CALL BASE6I_V3D(IPOIN)

      ENDDO
C       CONSTANTES DE INTEGRACIÓN

        IF(NNODE.EQ.8) THEN

                POSGP(1)=-1.0D0/DSQRT(3.0D0)

             POSGP(2)=1.0D0/DSQRT(3.0D0)
```

```
            WEIGP(1)=1.0D0

            WEIGP(2)=1.0D0
        ELSE
                POSGP(1)=1.0D0/4.D0

            WEIGP(1)=1.0D0/6.D0;

        ENDIF

C       LEER Y ALMACENAR LA INFORMACIÓN CONTENIDA EN Co.TXT

        DO COMP=1,NMATC !BUCLE SOBRE CADA MATERIAL COMPUESTO

                READ(41)EMAT,VCO(EMAT,:),DENS(EMAT)

        ENDDO

C       CÁLCULO DEL TAMAÑO DEL ARCHIVO Ctan.TXT PARA PODER ALOCAR LOS VECTORES Y USAR LOS DATOS

                DO WHILE (.not.eof(42))
                READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
                CTRL2=CTRL2+1
                ENDDO
C       MIRO DE QUE ELEMENTOS NECESITARÁN ACTUALIZARSE EN DAÑO PARA GUARDAR LA RI GUIDEZ
ELEMENTAL
                AUXV=AUX
                ELEMO=0
                TMAT=0
                REWIND(42)
                DO I=1,CTRL2
                READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
                IF((AUX.EQ.AUXV).AND.(ELEMO.NE.ELEM))THEN
                TMAT=TMAT+1
                ELEMO=ELEM
                AUXELEM(TMAT)=ELEM
                ENDIF
                ENDDO

C       ALOCO MATRIZ AUXILIAR PARA GUARDAR LAS RIGIDECES INICIALES DE LOS ELEMENTOS QUE SE DAÑARÁN

        ALLOCATE(ESTIFES(TMAT,9*NNODE*NNODE+1));

C       CÁLCULO DE LA RIGIDEZ ELEMENTAL INICIAL [KO]E PARA LOS DISTINTOS MATERIALES

        STIFFGO(:,:)=0.0
        MAS(:,:)=0.0
        AESTIF=1
        IAUX=1

        DO ELEM=1,NELEM

C       COORDENADAS DEL ELEMENTO

                        DO I=1,NNODE
                                        ELCOD(1,I)=COOR(CON(ELEM,I),1)
                                        ELCOD(2,I)=COOR(CON(ELEM,I),2)
                                        ELCOD(3,I)=COOR(CON(ELEM,I),3)
                        ENDDO

                        KGASP=0
                        STRSG(:)=0 !Para pequeñas deformaciones, si son grandes se actualiza el valor
                        FGRAD(:,:)=1 !Para el primer caso, no se usa (da igual el valor)
                        ESTIF(:,:)=0.0 !Ponemos la matriz de rigidez elemental a 0.

C       CÁLCULO DE LA MATRIZ DE MASA


         CALL MASMATV3D(ELEM,MAT,COOR,DENS)


C       INTEGRACIÓN SOBRE LOS PUNTOS DE GAUSS

                        DO IGAUS=1,NGAUX
                                DO JGAUS=1,NGAUY
                                        DO LGAUS=1,NGAUZ

                EXISP=POSGP(IGAUS)
                ETASP=POSGP(JGAUS)
                EZETA=POSGP(LGAUS)

                CALL SFR3D(EXISP,ETASP,EZETA)
```

```
            KGASP=KGASP+1


            CALL JACOB3D(XJACM,DJACB,GPCOD)

            CALL BMATSOL_V3D(BMATX,GPCOD,FGRAD)

C       COMPONEMOS EL TENSOR Co, A PARTIR DEL VCO

            CALL COMPON_V3D(VCO(MAT(ELEM),:),CO,36,1,1)

C       CÁLCULO DEL DVOL

            PESO=WEIGP(JGAUS)
         PESO1=WEIGP(LGAUS)
           DVOL=DJACB*WEIGP(IGAUS)*PESO*PESO1


C       SI ESTAMOS EN GRANDES DEFORMACIONES ACTUALIZAMOS EL VALOR DE LAS TENSIONES (STRSG)

            IF(LARGE.EQ.1)THEN
            READ(57)INDX,PGAUS,ICARG,IINCS,STRSG
            ENDIF


            A1 = MATEULER(ELEM,1)
            A2 = MATEULER(ELEM,2)
            A3 = MATEULER(ELEM,3)

C       ------ ESTABLECE LA MATRIZ CONSTITUTIVA DEL COMPUESTO DE CADA ELEMENTO EN S. GLOBAL

                  IF(A1**2+A2**2+A3**2.GT.1.e-9) THEN
                        dir = 2  ! local Elemento to global Estructura
                        call ROTM(CO,A1,A2,A3, mstr1, dir)
                  END IF

            CALL KMATRI_V3D(BMATX,DVOL,CO,STRSG)

                              ENDDO
                        ENDDO
                  ENDDO

C       DESCOMPONGO LA MATRIZ EN UN VECTOR PARA ALMACENARLA Y USARLA EN LA ACTUALIZACIÓN

            IKON=0
            DO IMAT=1,3*NNODE
                  DO JMAT=1,3*NNODE
                        IKON = IKON + 1
                        VECB(IKON) = ESTIF(IMAT,JMAT)
               ENDDO
               ENDDO

            IF(ELEM.EQ.AUXELEM(IAUX))THEN
             ESTIFES(AESTIF,1)=ELEM
             ESTIFES(AESTIF,2:) =VECB(:)
             AESTIF=AESTIF+1
             IAUX=IAUX+1
            ENDIF

C       ENSAMBLAJE DE LA MATRIZ DE RIGUIDEZ GLOBAL INICIAL [KO]G


                  F(1:8)=1.0
                  DO ELEMF=1,3*NNODE
                        C(1:8)=1.0
                        DO ELEMC=1,3*NNODE

                        AUX1=(ELEMF*0.333)+0.9
                           AUX2=(ELEMC*0.333)+0.9
                        AUX3=AUX1
                              AUX4=AUX2

                              GLOBF=((CON(ELEM,AUX3)-1)*3)+F(AUX3)
                              GLOBC=((CON(ELEM,AUX4)-1)*3)+C(AUX4)

                              C(AUX4)=C(AUX4)+1

                              STIFFGO(GLOBF,GLOBC)=STIFFGO(GLOBF,GLOBC)
.                           +ESTIF(ELEMF,ELEMC)


                              MAS(GLOBF,GLOBC)=MAS(GLOBF,GLOBC)+EMASA(ELEMF,ELEMC)
```

```
                        ENDDO
                        F(AUX3)=F(AUX3)+1
                    ENDDO
                ESTIF(:,:)=0.0
            ENDDO !Cierra Bucle sobre los elementos

C       REDUCIMOS LA MATRIZ
            VRESTR=0.0
            DO I=1,NPOIN
                RESTX=(RESTR(I,1))/100
                RESTY=(RESTR(I,1)-(RESTX*100))*0.1
                RESTZ=RESTR(I,1)-RESTX*100-RESTY*10
                VRESTR(3*I-2)=RESTX
                VRESTR(3*I-1)=RESTY
                VRESTR(3*I)=RESTZ
                IF (RESTR(I,2).NE.0)THEN
                VRESTR(3*I-2)=0
                ENDIF
                IF (RESTR(I,3).NE.0)THEN
                VRESTR(3*I-1)=0
                ENDIF
                IF (RESTR(I,4).NE.0)THEN
                VRESTR(3*I)=0
                ENDIF
            ENDDO

            STIFAUX(:,:)=0.0
            STIFRED(:,:)=0.0
            MASAUX(:,:)=0.0
            MASRED(:,:)=0.0

            DO I=1,NTOTV
                IF(VRESTR(I).EQ.0)THEN
                STIFAUX(CRED+1,:)=STIFFGO(I,:)
              MASAUX(CRED+1,:)=MAS(I,:)
                CRED=CRED+1
             ENDIF
            ENDDO



            DO I=1,NTOTV
                IF(VRESTR(I).EQ.0) THEN
                STIFRED(:,AUXRED+1)=STIFAUX(:,I)
                MASRED(:,AUXRED+1)=MASAUX(:,I)
                AUXRED=AUXRED+1
                ENDIF
            ENDDO

C       LLAMADA AL ALMACEMANIENTO EN SPARSE



C       PARA EVITAR PROBLEMAS OVERFLOW CON GRANDES MATRICES

            ALLOCATE(STIFRED2(CRED,CRED))
            ALLOCATE(MASRED2(CRED,CRED))

            STIFRED2=STIFRED(1:CRED,1:CRED)
            MASRED2=MASRED(1:CRED,1:CRED)

C           DO I=1,CRED
C           WRITE(60,10)MASRED2(I,:)
C           WRITE(61,10)STIFRED(I,1:500)
C           ENDDO


            CALL ALMACEN(STIFRED2,CRED,SPARK)
            CALL ALMACEN(MASRED2, CRED,SPARM)


            DEALLOCATE(STIFRED2,MASRED2,MASRED,MASAUX,MAS)
            DEALLOCATE(STIFRED,STIFAUX)

C       CÁLCULO DE VALORES Y VECTORES PROPIOS PARA EL CASO BASE


C       LEER DATOS PARA EL CÁLCULO DE FRECUENCIAS
C       ABRIMOS FRECUENCIAS.DTS Y LOS LEEMOS


        OPEN(UNIT=59,FILE='Frecuencias.dts', STATUS='UNKNOWN',
```

```
     .POSITION='REWIND')

          READ(59,*)NEIG,TOLERSIM,MAXI,FANIM
C         NEIG=7 !Nº Eigen values que se quieren calcular
C         TOLERSIM=1.0e-3 !Tolerancia
C         MAXI=1000000 !Nº max. de iteraciones
C         FANIM=3 !Nº forma propia de la que se quiere crear una animación
          pred=0

          NPRED = min(SPARK%n, 2*neig, neig+8)


          ALLOCATE(EIGENFREQ(NEIG))
          ALLOCATE(SVAL(NEIG))
          ALLOCATE(SVEC(SPARK%N,NEIG))
          ALLOCATE(SVECC(NTOTV,NEIG))
          ALLOCATE(SPRED(SPARK%N,NPRED))


          CALL Eigen_SIM(SPARK,SPARM,0,sval,svec,neig,tolersim,maxi,ioutsim,
     .                         iitersim,2,0,0,pred, spred,npred)


C         DESALOCAR PARA AHORRAR MEMORIA
          CALL deallocate_SCSR(SPARK)


C         PASAR RESULTADOS A FRECUENCIAS Y VECTOR COMPLETO

               do i=1,neig
                    eigenfreq(i)= (Sqrt(sval(i)))/(2*3.14159)
               enddo

     CALL VectProp(svec,NEIG,CRED,NTOTV,SVECC,VRESTR)


C         GUARDAR RESULTADOS PARA PODER POSTPROCESARLOS

          CALL ESCRIBIR(eigenfreq,NEIG,NTOTV,SVECC,0,0,1,CPROB,NNODE)

C         TRANSFORMAR DATOS PARA PODER CREAR UNA ANIMACIÓN DEL VECTOR PROPIO DESEADO

          DO I=1,FANIM
          CALL ANIMACION(SVECC(:,I),I,NTOTV,CPROB)
          ENDDO


C         DESALOCAR PARA AHORRAR MEMORIA
          DEALLOCATE(EIGENFREQ,SVAL,SVEC,SVECC)


C         CÁLCULO DE LA RIGUIDEZ ELEMENTAL EN PLASTICIDAD [KP]E Y ACTUALIZAR LA MATRIZ GLOBAL

               CTRL6=1 !Para que sólo entre en el primer paso
               STIFFGP=0.0


C         ALOCAMOS DIMENSIONES DE LOS VECTORES QUE USAREMOS A CONTINUACIÓN

               ALLOCATE(VELEM(CTRL2+1)) ! +1 Para que no de error en el último paso de
               ALLOCATE(VAUX(CTRL2+1))  ! actualización.
               ALLOCATE(VPGAUS(CTRL2))

C         VOLVEMOS AL PRINCIPIO DEL ARCHIVO Y LLENAMOS LOS VECTORES QUE ACABAMOS DE CREAR
               REWIND(42)
               DO I=1,CTRL2

                READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
                VAUX(I)=AUX
                VELEM(I)=ELEM
                VPGAUS(I)=PGAUS

               ENDDO


 100           I=1+CTRL3
               DO WHILE (VAUX(I).EQ.VAUX(I+1))
                I=I+1
                CTRL3=CTRL3+1
               ENDDO
```

```
      CTRL3=CTRL3+1 !Porque el último no lo suma ya que es diferente del siguiente
      CTRL4=1 !Cuantos elementos distintos se tienen que actualizar
      DO I=AUX5+1,CTRL3-1
            IF(VELEM(I).NE.VELEM(I+1))THEN
             CTRL4=CTRL4+1
            ENDIF
            AUX5=I
      ENDDO


C     SÓLO LO HEMOS DE LEER AQUÍ LA PRIMERA VEZ LAS OTRAS YA SE LEEN DENTRO DEL BUCLE

      IF (CTRL6.EQ.1)THEN

            REWIND(42)
            READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
            IGAUSL=PGAUS*0.01
            JGAUSL=(PGAUS-(IGAUSL*100))*0.1
            LGAUSL=PGAUS-IGAUSL*100-JGAUSL*10
            CTRL6=2 !Para que sólo entre una vez
            AUX6=AUX6+1 !Contador para evitar que en el último caso intente leer
      ENDIF             !el archivo cuando ya se hayan terminado los datos.

      AUX7=1

C     MIRAMOS SI EL PUNTO DE GAUSS LEIDO COINCIDE CON EL QUE TOCARÍA,
C     SI COINCIDE UTILIZAMOS EL CTAN LEIDO, SINO USAMOS CO


      DO I=1,CTRL4

            DO I2=1,NNODE
                  ELCOD(1,I2)=COOR(CON(ELEM,I2),1)
                  ELCOD(2,I2)=COOR(CON(ELEM,I2),2)
                  ELCOD(3,I2)=COOR(CON(ELEM,I2),3)
            ENDDO

            KGASP=0
            ELEMV=ELEM

            DO IGAUS=1,NGAUX
                  DO JGAUS=1,NGAUY
                        DO LGAUS=1,NGAUZ


            IF ((IGAUSL.EQ.IGAUS).AND.(JGAUSL.EQ.JGAUS).AND.
     .      (LGAUSL.EQ.LGAUS).AND.(E LEMV.EQ.ELEM))THEN

             CALL COMPON_V3D(VCTAN,CTAN,36,1,1)
             CONTROL=1 !Sirve para indicar si hemos usado el Ctan y necesitamos leer el siguiente
             ELSE
             CALL COMPON_V3D(VCO(MAT(ELEMV),:),CTAN,36,1,1)
             !Se llama Ctan pero en este caso sería Co, pero es para no hacer
             !los mínimos cambios en el código
             CONTROL=0
            ENDIF

            EXISP=POSGP(IGAUS)
            ETASP=POSGP(JGAUS)
            EZETA=POSGP(LGAUS)

            CALL SFR3D(EXISP,ETASP,EZETA)

            KGASP=KGASP+1

            CALL JACOB3D(XJACM,DJACB,GPCOD)


            CALL BMATSOL_V3D(BMATX,GPCOD,FGRAD)

C     CALCULAR DVOL

            PESO=WEIGP(JGAUS)
       PESO1=WEIGP(LGAUS)
             DVOL=DJACB*WEIGP(IGAUS)*PESO*PESO1

            A1 = MATEULER(ELEM,1)
            A2 = MATEULER(ELEM,2)
            A3 = MATEULER(ELEM,3)

C     ------ ESTABLECE LA MATRIZ CONSTITUTIVA DEL COMPUESTO DE CADA ELEMENTO EN S. GLOBAL
```

```
                    IF(A1**2+A2**2+A3**2.GT.1.e-9) THEN
                        dir = 2  ! local Elemento to global Estructura
                        call ROTM(CTAN,A1,A2,A3, mstr1, dir)
                    END IF

C       CALCULAR MATRIZ DE RIGIDEZ

                    CALL KMATRI_V3D(BMATX,DVOL,CTAN,STRSG)

                    IF(AUX6.EQ.CTRL2)THEN !Si ya se ha leído el último valor del fichero
                     GO TO 200                !no entra a leer ya que se produciría un error.
                    ELSE
                     IF (CONTROL.EQ.1) THEN
                     READ(42)AUX,ELEM,PGAUS,CARG,PASON,VCTAN
                     IGAUSL=PGAUS*0.01
                     JGAUSL=(PGAUS-(IGAUSL*100))*0.1
                     LGAUSL=PGAUS-IGAUSL*100-JGAUSL*10
                     AUX6=AUX6+1
                     ENDIF
                    ENDIF
 200                AUX6=AUX6

                            ENDDO
                        ENDDO
                    ENDDO


C       BUSCAR LA MATRIZ DE RIGIDEZ ELEMENTAL NO DAÑADA
                    AESTIF=1
                    DO WHILE(ELEMV.NE.ESTIFES(AESTIF,1))
                        AESTIF=AESTIF+1
                    ENDDO

                    IKON=1
                    DO IMAT = 1, 3*NNODE
                     DO JMAT = 1, 3*NNODE
              IKON = IKON+1
              ESTIFO(IMAT,JMAT) = ESTIFES(AESTIF,IKON)
                     ENDDO
                    ENDDO

                    IF (AUX7.EQ.1) THEN
                      STIFFGP=STIFFGO
                      AUX7=2
                    ENDIF



                    F(1:8)=1.0
                    DO ELEMF=1,3*NNODE
                        C(1:8)=1.0
                        DO ELEMC=1,3*NNODE

                        AUX1=(ELEMF*0.333)+0.9
                           AUX2=(ELEMC*0.333)+0.9
                        AUX3=AUX1
                            AUX4=AUX2

                            GLOBF=((CON(ELEMV,AUX3)-1)*3)+F(AUX3)
                            GLOBC=((CON(ELEMV,AUX4)-1)*3)+C(AUX4)

                            C(AUX4)=C(AUX4)+1

                            STIFFGP(GLOBF,GLOBC)=STIFFGP(GLOBF,GLOBC)
   .                       -ESTIFO(ELEMF,ELEMC)+ESTIF(ELEMF,ELEMC)

                        ENDDO
                        F(AUX3)=F(AUX3)+1
                    ENDDO
                    ESTIF(:,:)=0.0
                ENDDO


C       REDUCIMOS LA MATRIZ

                ALLOCATE(STIFRED(NTOTV,NTOTV));
                ALLOCATE(STIFAUX(NTOTV,NTOTV));
                STIFAUX(:,:)=0.0
                STIFRED(:,:)=0.0
```

```
      AUXRED=0

      DO I=1,NTOTV
            IF(VRESTR(I).EQ.0)THEN
            STIFAUX(AUXRED+1,:)=STIFFGP(I,:)
        AUXRED=AUXRED+1
       ENDIF
      ENDDO

      AUXRED=0

      DO I=1,NTOTV
            IF(VRESTR(I).EQ.0) THEN
            STIFRED(:,AUXRED+1)=STIFAUX(:,I)
            AUXRED=AUXRED+1
            ENDIF
      ENDDO




      STIFFGP=0.0

C     GUARDAMOS LA RIGIDEZ EN FORMATO SPARSE UNA VEZ HEMOS ACUTALIZADO TODO EL PASO

      ALLOCATE(STIFRED2(CRED,CRED))
      STIFRED2=STIFRED(1:CRED,1:CRED)

            IF(aux.eq.18)THEN
            DO AUX7=1,CRED
            WRITE(61,12)STIFRED2(AUX7,:)
            ENDDO
            ENDIF

      CALL ALMACEN(STIFRED2,CRED,SPARK)
      DEALLOCATE(STIFRED2,STIFRED,STIFAUX)
      pred=1
      ALLOCATE(EIGENFREQ(NEIG))
      ALLOCATE(SVAL(NEIG))
      ALLOCATE(SVEC(SPARK%N,NEIG))
      ALLOCATE(SVECC(NTOTV,NEIG))


      CALL Eigen_SIM(SPARK,SPARM,0,sval,svec,neig,tolersim,maxi,
     .                  new_size,iitersim,2,0,0,pred,spred,npred)


C     DESALOCAR PARA AHORRAR MEMORIA
      CALL deallocate_SCSR(SPARK)

            IF(SVAL(1).LT.0)THEN
            GO TO 300
            ENDIF


C     PASAR RESULTADOS A FRECUENCIAS Y VECTOR COMPLETO

            do i=1,neig
                  eigenfreq(i)= (Sqrt(sval(i)))/(2*3.14159)
            enddo

   CALL VectProp(svec,NEIG,CRED,NTOTV,SVECC,VRESTR)


C     GUARDAR RESULTADOS PARA PODER POSTPRO CESARLOS

      IF(AUX6.EQ.CTRL2)THEN
      AUX=AUX+1
      ENDIF

      READ(58,*)DAN

      CALL ESCRIBIR(eigenfreq,NEIG,NTOTV,SVECC,DAN,1,AUX,CPROB,NNODE)


 300  DEALLOCATE(EIGENFREQ,SVAL,SVEC,SVECC)

      IF(AUX6.EQ.CTRL2)THEN !Para salir del bucle una vez haya llegado al final
            CONTINUE
```

```
      ELSE
            GO TO 100
      ENDIF



10    FORMAT(500F15.4)
11    FORMAT(48F15.4)
12 FORMAT(500F15.4)
96 FORMAT(4I6,36F13.3)
97 FORMAT(24F13.3)
98 FORMAT(24F13.10)
99 FORMAT(I2,36F13.3,1XF8.3)


      END
```

CÓDIGO MALLAS-ANIMACIÓN.BAT

```
@echo off
Title JBLL--Crear mallas para las animaciones
color F
echo ========================================
echo =            JBLL              =
echo =                              =
echo =                              =
echo =  CREACION DE LAS CARPETAS Y MALLAS   =
echo =   PARA USAR LAS ANIMACIONES EN EL    =
echo =            GID               =
echo =                              =
echo =                              =
echo ========================================
echo.
echo.
set /p NOM=Nombre del problema resuelto:
set /p NUM=Numero de animaciones creadas:
:BUCLE
set /a contador=contador+1
md Animacion-%contador%
copy “*.post.msh" Animacion-%contador%\"Animacion-F%contador%-%NOM%.post.msh"
copy “Animacion-F%contador%-%NOM%.post.res" Animacion-%contador%\"Animacion-F%contador%-
%NOM%.post.res"
del  “Animacion-F%contador%-%NOM%.post.res"
copy “*.post.msh" "%NOM%F.post.msh"
del *.61
del *.7

IF %contador%==%NUM% (goto SIGUIENTE) ELSE (goto BUCLE)
:SIGUIENTE

exit
```

Anejo II: Archivos de entrada/salida

Archivo de entrada --- Viga3d.dts

```
MEC3D
2
0 1 1
0
START1
Test02-Implicit_3D_threedimensional_problem
176      90      1      3      3      1      20      1      2      1      3  1      0
1120618211521501661681 56 154 171 173 0 0 0
...      ...     ...    ...    ...    ...    ...     ...    ...    ...    .....  .....  .....  ..... ..... ..... ..... ..... .. .. ..
901206182126111331    33  15   19   38  0 0 0
  1    1
 P   1 0   0   0   1   1  1
1       0  0.3   1
2       0  0.2   1
...     ...  .....  ...
176     0.3  0   0
---------------------------------------------- Material identifier: PVC ----------------------------------------
  1
3 3 4 2 1 3
37000000 0.2   0   2.4   30   30   30000
1   0   5000   0   0   1.0
1   0   0   0   0   50
1   0   0
0 0 0
1 0.1  3.5   1
30000 1 1 0  0 1.0
0 0 0
0.2    37000000  0.2    0.2    37000000   0.2  0.2   0   0   0
0   0   0
1 1 1 1 1 1
1 1 1 1 1 1
0   0
0   0   0   0
0   0
0   0   0   0
0
0
0
---------------------------------------------- END STANDARD MATERIAL PROPERTIES ----------------------------------
CURV,   1
Curve1
X:DISPL,
  75    1    1
Y:RCTOT,
  2
  1
Title: Stage1
  1    1
  1   20    0    0    0    0    1    0    0    0
  No_Thermal_Loads
  12    0
25    111   0.000000   0.000000   0.000000
....    .....    ..............    ..............    ..............
176   111  0.000000   0.000000   0.000000
  0
  0
 10 0.0005 0.05 0.0001   50   0 0 1 1   2 0 0 0 0   1 1 0   0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
```

Archivo de salida --- viga3d.post.res

```
GiD Post Results File 1.0

 GaussPoints  "HEXA-G8"       Elemtype   HexaHedra     "HEXA-G8"
 Number of Gauss Points:  8
 Natural Coordinates   : Internal
 End GaussPoints

Result "DISPLACEMENTS" "DISPLACEMENTS"      0.050000 Vector Onnodes
 ComponentNames "DX" "DY ""DZ"
 Values
   1    0.384967132663193E-06   -0.422805714267150E-04   -0.144124296590165E-03
   2   -0.160335254395913E-05   -0.419801836324316E-04   -0.682169064103610E-04
   3   -0.1207003489048l9E-05   -0.123090323600055E-03   -0.144282287117266E-03
   4    0.337074485093392E-06   -0.439747260913573E-04   -0.145227687231739E-03
   5   -0.376498478565562E-05   -0.120904899071491E-03   -0.6540663831207 63E-04
   6   -0.101741607600539E-05   -0.441888172014784E-04   -0.688664273523434E-04
  ...  ..........................................  ..........................................  ..........................................

 175    0.000000000000000E+00    0.000000000000000E+00    0.000000000000000E+00
 176    0.000000000000000E+00    0.000000000000000E+00    0.000000000000000E+00
 End Values
Result "STRESSES//COMPOSITE" "MECHANICAL"                    0.050000 Matrix OnGaussPoin    ts
"HEXA-G8"
ComponentNames "Sxx" "Syy" "Szz" "Sxy" "Sxz" "Syz"
 Values
   1   0.543672612539294E+03   -0.218151480670832E+04   -0.450335304466724E+04   -
0.380564402451932E+02   -0.771261944190164E+02   -0.388356794610545E+04
      0.541543793185471E+03   -0.226225338942508E+04   -0.443325855871959E+04
0.367072480594052E+03   -0.915665824095612E+00   -0.368631202815567E+04
     -0.173333606399585E+04   -0.121238781040996E+05   -0.686055947309076E+04   -
0.897623337884050E+03   0.351959868266399E+02   -0.102079014014947E+05
     -0.177769264499841E+04   -0.122290811228083E+05   -0.697713935939479E+04   -
0.530455971105427E+03   -0.410145417682810E+02   -0.103540735348199E+05
     -0.268965399103236E+04   -0.622509925195846E+04   -0.214879486673463E+05   -
0.773613219266641E+02   -0.648062972362442E+03   -0.767045794679169E+04
     -0.272583866680260E+04   -0.633989369109164E+04   -0.215540776070643E+05
0.327767598912581E+03   -0.134414873138539E+04   -0.740347392830733E+04
     -0.706017529096490E+04   -0.167163825975114E+05   -0.244962427131807E+05   -
0.867200119030947E+03   -0.135912086335928E+04   -0.139250633016464E+05
     -0.707047601555105E+04   -0.167875297598037E+05   -0.244765991738191E+05   -
0.500032752252324E+03   -0.663035104336329E+03   -0.141409635355561E+05
   2   0.416362572794534E+03   -0.218398474479090E+04   -0.456103171760828E+04   -
0.452927001331844E+02   0.403735435425343E+02   -0.395576862311659E+04
      0.416362572792465E+03   -0.218398474479479E+04   -0.456103171761472E+04
0.452927001272130E+02   -0.403735435342994E+02   -0.395576862312047E+04
     ..........................................  ..........................................  ..........................................
```

Archivo de salida --- viga3d.AA

```
#    1 (2E20.8)
#    1    0
#Curve1
#X:DISPL          75   1   1   0   0
#Y:RCTOT           2   0   0   0   0
     0.00000000E+00     0.00000000E+00
    -0.32977191E-05     0.73127442E+02
    -0.65954382E-05     0.14625488E+03
    -0.97787578E-05     0.21464509E+03
    -0.12861134E-04     0.27975669E+03
    -0.15780647E-04     0.34003745E+03
    -0.18762895E-04     0.40315489E+03
    -0.21538668E-04     0.46130597E+03
    -0.24332676E-04     0.51594304E+03
    -0.26888306E-04     0.56269818E+03
```

Archivo de salida --- viga3d.mec.sal

```
      ------------------------------------------------------
      !ooooooooo.   ooooo              .oooooo.      ooooooooooo. !
      !`888   `Y88.  `888             d8P   `Y8b     `888    `Y8b !
      ! 888    .d88   888             888              888      888!
      ! 888ooo88P    888             888              888      888!
      ! 888          888             888              888      888!
      ! 888          888          o `88b      ooo     888     d88 !
      !o888o        o888ooooooood8  `Y8bood8P     o888bood8P   !
      ------------------------------------------------------
      !                                    !
      !Programa de ELEMENTOS FINITOS para resolver:
      !   * Problemas con no linealidad geometrica y constitutiva.
      !   * Problemas de fr  actura y localizacion de deformaciones inelasticas.
      !   * Problemas de materiales compuestos mediante la teoria de mezclas.
      !   * Problemas de anisotropia general mediante la teoria se mapeo de espacios.
      !   * Resuelve problemas de: Tension plana.        !
      !                            Deformacion plana.    !
      !                            Axil Simetria.        !
      !                            Barras de Timoshenko, !
      !                            Barras articuladas.   !
      !   * Utiliza Elementos   : Planos  3,4,6,8 y 9 N, !
      !                           de Timosh. 2 y 3 N,    !
      !                           articul.   2 y 3 N.    !
      ------------------------------------------------------


-->>> TIEMPO      0: 0: 0

-->>> TIEMPO   0.0000000E+00                HORA:   0: 0: 0


   PROBLEM TYPE TO SOLVE : MEC3D
   ---------------------

   PROBLEM TITLE : Test02-Implicit_3D_threedimensional_problem
   -------------

1.- PATH DEPENDENT SOLUTION
2.- INFINTESIMAL STRAIN
3.- SYMETRIC SOLVER


    NUMBER OF NODAL POINTS.....:    176(NPOIN)
    NUMBER OF ELEMENTS.....:     90(NELEM)
    NUMBER OF COMPOSITES MATERIALS.....:      1(NMATC)
    NUMBER OF MATERIALS.....  .....:      3(NMATS)
    NUMBER OF CONSTRUCTION STEPS.....:      1(NFASE)
    NUMBER OF LOAD INCREMENTS.....:     20(NINCM)

  .....  .....
    ELEMENTS DEFINITION :
    -------------------

 ELEMENT  MATNO    FRACL   NTYPE, NFASE NNODE NGAUS NINTE,     NODE   NUMBERS
ÁNGULOS
    1     1     0.000    6     1      8     2     1     152    150    166    168    156    154    171    173       0.00
0.00     0.00
    2     1     0.000    6     1     8     2     1     153    152    168    169    159        156    173    175       0.00
0.00     0.00
  .....    ....    ........   ....       ....  ....   ....  ....   ......    .....    .....    .....    .....    .....              .....    .....       .......
```

90 1 0.000 6 1 8 2 1 26 11 13 31 33 15 19 38 0.00
0.00 0.00

COORDINATES DEFINITION :

NODE	X-COORD	Y-COORD	Z-COORD	NODE	X-COORD	Y-COORD	Z-COORD
1	0.000	0.300	1.000	2	0.000	0.200	1.000
3	0.000	0.300	0.900	4	0.100	0.300	1.000
...	.....	.....	..	.....	...	.....	.....	.....
173	0.100	0.000	0.000	174	0.300	0.100	0.000
175	0.200	0.000	0.000	176	0.300	0.000	0.000

MATERIAL PROPERTIES DEFINITION :

MATERIAL NUMBER: 1

YIELD FUNCTION.....: MOHR-COULOMB
POTENTIAL FUNCTION.....: MOH R-COULOMB
INTEGRATION ALGORITHM.....: DMDMD
FLOW TYPE.....: VARIABLE
I..... .....

RENUMBERED NODES :

OLD	NEW	NEW	OLD
1	2	1	2
2	1	2	1
3	5	3	4
.....	.....	.....	.....
175	175	175	175
176	176	176	176

ECHO OF THERMAL LOADS DATA FOLLOWS :
==========

---------- LOAD NUMBER: 1 ----------

0Title: Stage1

NUMBER OF INCREMENTS..... .....: 20
POINTS LOADS.....: NO
GRAVITY LOADS.....: NO
FACE LOAD.....: NO
TYPE OF LOAD.....: STATIC

---

0 NUMERO TOTAL DE CARGAS POR ELEMENTO

1 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00

```
  90      0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
   0      0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00


       ECHO OF FIXITY DATA FOLLOWS :
       ============================

 NODE     CODE       FIXED VALUES
   25      111      0.000000  0.000000  0.000000
   32      111      0.000000  0.000000   0.000000
  ...      .....     ..............  ..............  ..............
 176      111      0.000000  0.000000  0.000000


       SOLVER INFORMATION :
       ==================


       ANCHO DE BANDA SIN RENUMERACION:  135  ELEMENTO:   54

       ANCHO DE BANDA CON RENUMERACION:   96  ELEMENTO:   22

       ANCHO DE BANDA CON RENUMERACION (CON RESTRICCIONES):       93  ELEMENTO:   22
 -->>> TIEMPO  0.1560000E+00              HORA:  0: 0: 0
========================= ============================================================
0  NUMERO DE CARGA   1  TIEMPO REAL  CALCULO   0.05  NUMERO DE INCREMENTO   1
0  FACTOR DE CARGA      =  0.00050
  TOLERANCIA          =  0.00010   MAX. Nr. ITERACIONES  =  50

  COD. SALIDA INICIAL  =   0   COD. DE SALIDA FINAL  =   0

  ALGORITMO DE SOLUCION =   2   TIPO DE ARC-LENGTH   =   0

  LONGITUD DE ARCO     =  0.000000E+00   NODO y GRAD. DE LIB  =  0,   0

  POST-PROC EN FEMVIEW  =   0   LARGE =   0   ISIME =   0      IDINA =   0
0  CALCULO DE LA LONGITUD CARACTERISTICA COMO DSQRT(TVOLU)
 ---------
0 IITER.=  6  COD. DE CONVERG.=   0   RELAC. DE NORM. RESIDUAL=  0.407128E-07
                RESIDUO MAXIMO  =  0.755497E-07
   1   6
--------------------------------------------------------------------------------
 Pot. deformativa          :   0.000000E+00
 Pot. cinetica             :   0.000000E+00
 Pot. disipada             :   0.000000E+00
 Pot. (def.)+(cin.)+(disip.):   0.000000E+00
 Pot. introducida          :   0.000000E+00
 Energ. introducida        :   0.000000E+00
 Energ. disipada           :   0.000000E+00
--------------------------------------------------------------------------------
********  DANO GLOBAL -medida energetica--:   0.215830E+00
********  DANO GLOBAL -medida en fuerza-- :   -0.147869E-06
        RESULTADOS EN LA SALIDA =   0.00050
        ***********************************
```

3.482401420998205E-003
3.987347113078210E-002
9.138556586773172E-002
0.163103396600391
0.232592765914423
0.291526714130732
0.364841225710857
0.430927783085663
0.513690785254786
0.549396034977606
0.582625628963865
0.654085720360045
0.709353951575367
0.776658327088419
0.913373063037344
0.923978769017817
0.932079793773579
0.938440383211111
0.943959348243244
0.948882810989565
0.953413767181545
0.969773282275184
0.978860149282096

Archivo de salida --- viga3d.post.msh

```
MESH "HEXA-G8"  Dimension 3 Elemtype HexaHedra    Nnode 8
 Coordinates
   1   0.000000    0.300000    1.000000
   2   0.000000    0.200000    1.000000
   3   0.000000    0.300000    0.900000
   4   0.100000    0.300000    1.000000
   5   0.000000    0.200000    0.900000
   6   0.100000    0.200000    1.000000
   7   0.100000    0.300000    0.900000
   8   0.100000    0.200000    0.900000
   9   0.000000    0.300000    0.800000
  10   0.000000    0.100000    1.000000
  11   0.200000    0.300000    1.000000
  ...   ..............    ..............    ..............

 173   0.100000    0.000000    0.000000
 174   0.300000    0.100000    0.000000
 175   0.200000    0.000000    0.000000
 176   0.300000    0.000000    0.000000
End Coordinates
Elements
    1   152   150   166   168   156   154   171   173     1
    2   153   152   168   169   159   156   173   175     1
    3   158   153   169   174   160   159   175   176     1
    4   136   13 3   150   152   141   139   154   156     1
    5   137   136   152   153   142   141   156   159     1
   ....   ......   ......   ......   ......   ......   ......   ......   ......   ....

   88    4    1    3    7    6    2    5    8    1
   89   11    4    7   13   15    6    8   19    1
   90   26   11   13   31   33   15   19   38    1
End elements
```

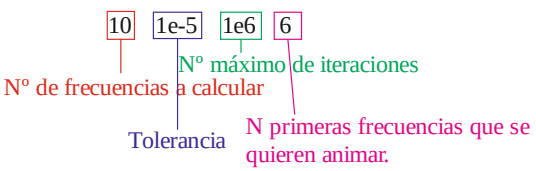
Archivo de salida --- fort.26

```
 ---------
0 IITER.= 1  COD. DE CONVERG.=  0   RELAC. DE NORM. RESIDUAL= 0.112992E-11
               RESIDUO MAXIMO  = 0.224931E-12
 ---------
0 IITER.= 1  COD. DE CONVERG.=  0   RELAC. DE NORM. RESIDUAL= 0.639619E-12
               RESIDUO MAXIMO  = 0.303979E-12
 ---------
0 IITER.= 1  COD. DE CONVERG.=  1   RELAC. DE NORM. RESIDUAL= 0.534998E+01
               RESIDUO MAXIMO  = 0.391045E+01
 ---------
0 IITER.= 2  COD. DE CONVERG.=  1   RELAC. DE NORM. RES  IDUAL= 0.127935E+01
               RESIDUO MAXIMO  = 0.704862E+00
 ---------
0 IITER.= 3  COD. DE CONVERG.=  1   RELAC. DE NORM. RESIDUAL= 0.216132E+00
               RESIDUO MAXIMO  = 0.153150E+00
 ---------
0 IITER.= 4  COD. DE CONVERG.=  1   RELAC. DE NORM. RESIDUAL= 0.523590E-01
               RESIDUO MAXIMO  = 0.336619E-01
 ---------
0 IITER.= 5  COD. DE CONVERG.=  1   RELAC. DE NORM. RESIDUAL= 0.130224E-01
               RESIDUO MAXIMO  = 0.81   1292E-02
 ---------
0 IITER.= 6  COD. DE CONVERG.=  1   RELAC. DE NORM. RESIDUAL= 0.339446E-02
               RESIDUO MAXIMO  = 0.202262E-02
```

Archivo de salida --- fort.31

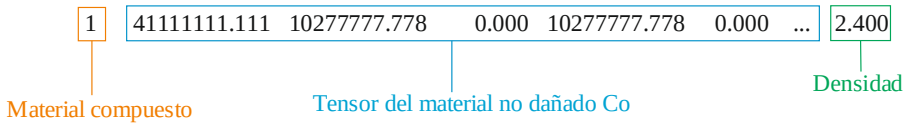
```
 SUMMARY OF COMPUTING TIMES :
       ==========================

          TOTAL CPU TIME          :    56.45
          COMPUTING MATRICES      :     0.00 (  0.00 % )
             INITIAL COMPUTATIONS     :     0.00 (  0.00 % )
             STIFFNESS MATRIX         :     0.00 (  0.00 % )
             ASSEMBLY PROCESS         :     0.00 (  0.00 % )
          SOLVING EQUATIONS       :     0.00 (  0.00 % )
             COMPUTING DECOMPOSITION      :     0.00 (  0.00 % )
             SOLVING LOWER MATRIX     :     0.00 (  0.00 % )
             SOLVING UPPER MATRIX     :     0.00 (  0.00 % )
          UPDATING DISPLACEMENTS    :     0.00 (  0.00 % )
          EVALUATING RESIDUAL FORCES :     0.00 (  0.00 % )
          CHECKING CONVERGENCE     :     0.00 (  0.00 % )
          I/O   OPERATIONS        :     0.00 (  0.00 % )
             INPUT OPERATIONS         :     0.00 (  0.00 % )
             OUTPUT TO EXTERNAL FILES  :     0.00 (  0.00 % )
             I/O RESTART FILE         :     0.00 (  0.00 % )
          RENUMBERING OPERATIONS    :     0.00 (  0.00 % )
          OTHER OPERATIONS        :    56.45 (100.00 % )
```

Archivo de entrada --- frecuencias.dts



Archivo de salida y entrada --- Co.txt

Formateado



Sin formato (Ahorra un 50% tamaño)

```
00000000h: 2C 01 00 00 01 00 00 00 3A 8E E3 38 72 9A 83 41 ; ,........:Ža8ršƒA
00000010h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Ža8rščA........
00000020h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Ža8rščA........
00000030h: 00 00 00 00 00 00 00 00 3A 8E E3 38 72 9A 63 41 ; ........:Ža8rščA
00000040h: 3A 8E E3 38 72 9A 83 41 00 00 00 00 00 00 00 00 ; :Ža8ršƒA........
00000050h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Ža8rščA........
00000060h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000070h: 00 00 00 00 00 00 00 00 56 55 55 55 AB 67 6D 41 ; ........VUUU«gmA
00000080h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000090h: 00 00 00 00 00 00 00 00 3A 8E E3 38 72 9A 63 41 ; ........:Ža8rščA
000000a0h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Ža8rščA........
000000b0h: 3A 8E E3 38 72 9A 83 41 00 00 00 00 00 00 00 00 ; :Ža8ršƒA........
000000c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000e0h: 00 00 00 00 00 00 00 00 56 55 55 55 AB 67 6D 41 ; ........VUUU«gmA
000000f0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000100h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000110h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
```

Archivo de salida --- Animacion-viga3d.post.res

```
GiD Post Results File 1.0

GaussPoints "HEXA-G8"      Elemtype  HexaHedra     "HEXA-G8"
Number of Gauss Points:  8
Natural Coordinates  : Internal
End GaussPoints

Result "ANIMACIÓN  6ª FRECUENCIA" "DISPLACEMENTS"     1 Vector Onnodes
ComponentNames "DX" "DY ""DZ"
Values
   1    0.498660071378960E-01    0.100000000000000E+01   -0.409441322477580E+00
  ...   ....................................   ....................................   ....................................   .....
 176   0.000000000000000E+00    0.000000000000000E+00    0.000000000000000E+00
End Values
Result "ANIMACIÓN  6ª FRECUENCIA" "DISPLACEMENTS"     2 Vector Onnodes
ComponentNames "DX" "DY ""DZ"
Values
   1    0.448794064241064E-01    0.900000000000000E+00   -0.368497190229822E+00
   2    0.437514467858132E-01    0.820300244532364E+00   -0.226461813796626E+00
   3    0.466760850587143E-01    0.789646209776235E+00   -0.367892884496011E+00
  ...   ....................................   ....................................   ....................................
```

Archivo de salida y entrada –- Csec.txt

Formateado

Aux	Indx	Pgaus	Icarg	Iincs	Csec							
1	96	112	1	6	39870654.465	9967663.616	0.000	9967663.616	0.000	0.000	9967663.616	39870654.465
1	96	122	1	6	39423033.558	9855758.389	0.000	9855758.389	0.000	0.000	9855758.389	39423033.558
1	97	112	1	6	39911989.594	9977997.398	0.000	9977997.398	0.000	0.000	9977997.398	39911989.594
1	97	122	1	6	39942153.652	9985538.413	0.000	9985538.413	0.000	0.000	9985538.413	39942153.652
1	98	112	1	6	39878593.812	9969648.453	0.000	9969648.453	0.000	0.000	9969648.453	39878593.812
1	98	122	1	6	39878600.329	9969650.082	0.000	9969650.082	0.000	0.000	9969650.082	39878600.329
1	99	112	1	6	39942126.964	9985531.741	0.000	9985531.741	0.000	0.000	9985531.741	39942126.964
1	99	122	1	6	39911972.947	9977993.237	0.000	9977993.237	0.000	0.000	9977993.237	39911972.947
1	100	112	1	6	39422998.399	9855749.600	0.000	9855749.600	0.000	0.000	9855749.600	39422998.399
1	100	122	1	6	39870624.258	9967656.065	0.000	9967656.065	0.000	0.000	9967656.065	39870624.258
2	86	112	1	7	38346137.699	9586534.425	0.000	9586534.425	0.000	0.000	9586534.425	38346137.699
2	86	122	1	7	38088013.237	9522003.309	0.000	9522003.309	0.000	0.000	9522003.309	38088013.237
2	86	212	1	7	38537712.463	9634428.116	0.000	9634428.116	0.000	0.000	9634428.116	38537712.463
2	86	222	1	7	38351883.703	9587970.926	0.000	9587970.926	0.000	0.000	9587970.926	38351883.703
2	87	112	1	7	38457980.136	9614495.034	0.000	9614495.034	0.000	0.000	9614495.034	38457980.136
2	87	122	1	7	38486163.365	9621540.841	0.000	9621540.841	0.000	0.000	9621540.841	38486163.365
2	87	212	1	7	38532880.851	9633220.213	0.000	9633220.213	0.000	0.000	9633220.213	38532880.851
2	87	222	1	7	38587060.978	9646765.244	0.000	9646765.244	0.000	0.000	9646765.244	38587060.978
2	88	112	1	7	38451081.669	9612770.417	0.000	9612770.417	0.000	0.000	9612770.417	38451081.669
2	88	122	1	7	38451089.004	9612772.251	0.000	9612772.251	0.000	0.000	9612772.251	38451089.004
2	88	212	1	7	38514140.538	9628535.134	0.000	9628535.134	0.000	0.000	9628535.134	38514140.538
2	88	222	1	7	38514147.324	9628536.831	0.000	9628536.831	0.000	0.000	9628536.831	38514147.324
2	89	112	1	7	38486141.775	9621535.444	0.000	9621535.444	0.000	0.000	9621535.444	38486141.775
2	89	122	1	7	38457964.170	9614491.043	0.000	9614491.043	0.000	0.000	9614491.043	38457964.170
2	89	212	1	7	38587034.171	9646758.543	0.000	9646758.543	0.000	0.000	9646758.543	38587034.171
2	89	222	1	7	38532865.369	9633216.342	0.000	9633216.342	0.000	0.000	9633216.342	38532865.369
2	90	112	1	7	38087978.309	9521994.577	0.000	9521994.577	0.000	0.000	9521994.577	38087978.309
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.

Aux --- Indica paso de acutalización de la matriz de rigidez
Indx --- Elemento
Pgaus --- Punto de Gauss concatenado
Icarg --- Paso de carga
Iincs --- Paso incremento
Csec --- Tensor secante

Sin formato

```
00000000h: 2C 01 00 00 01 00 00 00 3A 8E E3 38 72 9A 83 41 ; ,.......:Žã8ršƒA
00000010h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Žã8rščA........
00000020h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Žã8rščA........
00000030h: 00 00 00 00 00 00 00 00 3A 8E E3 38 72 9A 63 41 ; ........:Žã8rščA
00000040h: 3A 8E E3 38 72 9A 83 41 00 00 00 00 00 00 00 00 ; :Žã8ršƒA........
00000050h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Žã8rščA........
00000060h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000070h: 00 00 00 00 00 00 00 00 56 55 55 55 AB 67 6D 41 ; ........VUUU«gmA
00000080h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000090h: 00 00 00 00 00 00 00 00 3A 8E E3 38 72 9A 63 41 ; ........:Žã8rščA
000000a0h: 3A 8E E3 38 72 9A 63 41 00 00 00 00 00 00 00 00 ; :Žã8rščA........
000000b0h: 3A 8E E3 38 72 9A 83 41 00 00 00 00 00 00 00 00 ; :Žã8ršƒA........
000000c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000e0h: 00 00 00 00 00 00 00 00 56 55 55 55 AB 67 6D 41 ; ........VUUU«gmA
000000f0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000100h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000110h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
```

Archivo de salida --- viga3d.post.res (CFYFP)

GaussPoints "HEXA-G8" Elemtype HexaHedra "HEXA-G8"
Number of Gauss Points: 8
Natural Coordinates : Internal
End GaussPoints

Result "DEFORMADA, 1ª FRECUENCIA" "DISPLACEMENTS" 1.000000 Vector Onnodes
ComponentNames "DX" "DY ""DZ"
Values
1 0.733871858762587E+00 0.328341652274933E+00 -0.771900146816974E-01
2 0.500185617394868E+00 0.318623635357484E+00 -0.750254444851077E-01
... .
176 0.000000000000000E+00 0.000000000000000E+00 0.000000000000000E+00
End Values
Result "DEFORMADA, 2ª FRECUENCIA" "DISPLACEMENTS" 1.000000 Vector Onnodes
ComponentNames "DX" "DY ""DZ"
Values
1 0.741885746516222E-02 0.188218002315621E+00 0.391599734258916E+00
2 0.105518527012055E-01 0.181026973631551E+00 0.211923642147631E+00
...
17)0
En
Res les

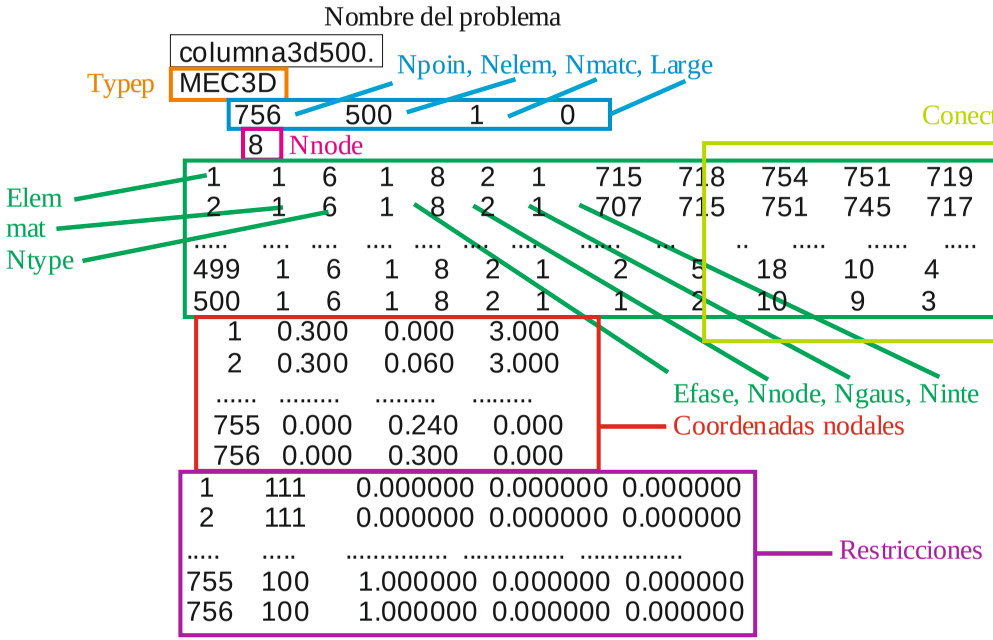
Archivo de salida --- viga3dF.post.res (CFYFP)

ComponentNames "DX" "DY ""DZ"
Values
1 0.573294074361978E+00 0.366578602585555E+00 0.267884726688366E+00
... .
176 0.000000000000000E+00 0.000000000000000E+00 0.000000000000000E+00
End Values

....

Result "DEFORMADA, 1ª FRECUENCIA" "DISPLACEMENTS" 2.000000 Vector Onnodes
ComponentNames "DX" "DY ""DZ"
Values
1 0.727741844097801E+00 0.325275950990273E+00 -0.792244212572895E-01
2 0.496134402556021E+00 0.315693738581783E+00 -0.773152102227423E-01
... .

Archivo de salida y entrada --- Datosgen.txt



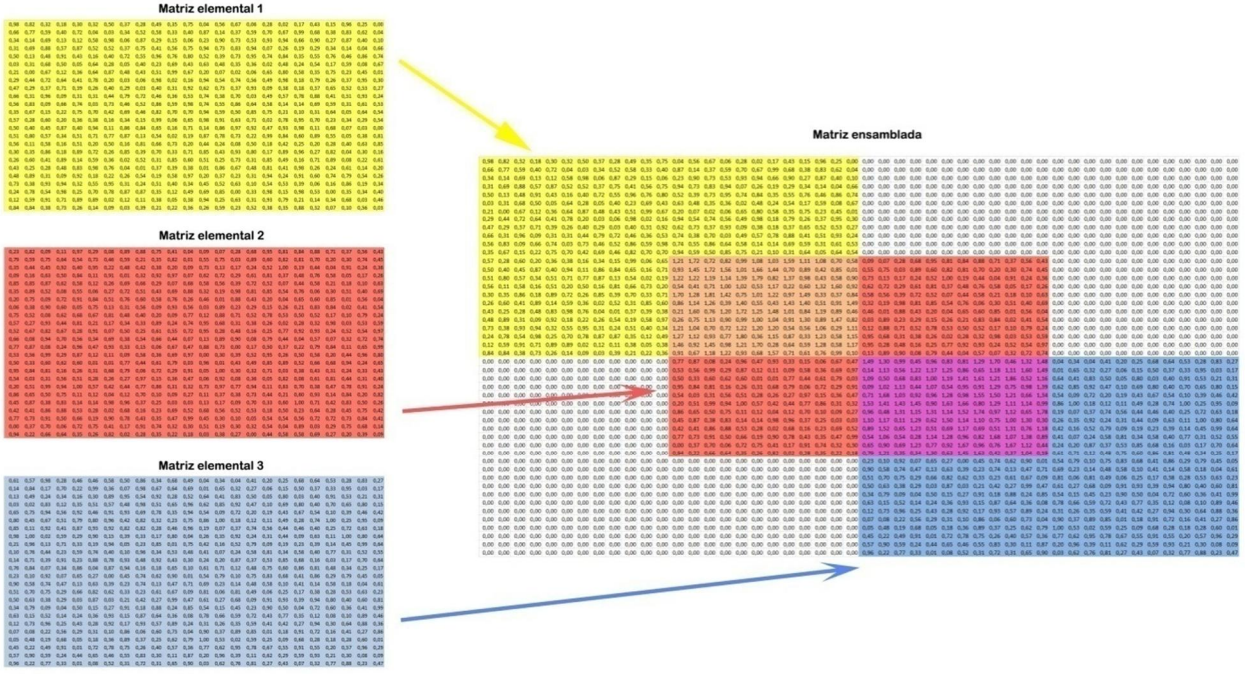
Typep --- Tipo de problema
Npoin --- Nº nodos totales del problema
Nelem --- Nº elementos
Nmatc --- Nº de materiales compuestos
Large --- Indicador de pequeñas (0) o grandes (1) deformaciones
Nnode --- Nº nodos de un elemento
Elem --- Elemento
mat --- Material
Ntype --- Tipo de problema (P. tridimensional)
Efase --- Fase de carga
Nnode --- Nº nodos de un elemento
Ngaus --- Nº puntos de integración a utilizar
Ninte --- Cuadratura a utilizar (Gauss)
Conectividad --- Nodos que forman un elemento
Coordenadas nodales --- Nº nodo, x, y, z
Restricciones --- Nodo, libertad (0) o restricción (1) en las tres direcciones, valor del desplazamiento

Anejo III: Validación ensamblaje

Como ya se ha comentado en la sección 7.6.2 en la parte del código correspondiente al ensamblaje, se quería simular el caso de un problema de 16 nodos, con tres elementos que presentaban las siguientes conectividades:

- Elemento 1: [1 2 3 4 5 6 7 8]
- Elemento 2: [5 6 7 8 9 10 11 12]
- Elemento 3: [9 10 11 12 13 14 15 16]

Las tres matrices elementales creadas de manera aleatoria y utilizadas para validar el ensamblaje son (por motivos de espacio disponible en un folio se muestra con solamente dos decimales y redondeo, además de tamaño reducido, en las hojas siguientes se muestran en más detalle. La validación se ha realizado con una precisión más elevada, 12 decimales):



Matriz del elemento 1

0,98	0,82	0,32	0,18	0,30	0,32	0,50	0,37	0,28	0,49	0,35	0,75	0,04	0,56	0,67	0,06	0,28	0,02	0,17	0,43	0,15	0,96	0,25	0,00
0,66	0,77	0,59	0,40	0,72	0,04	0,03	0,34	0,52	0,58	0,33	0,40	0,87	0,14	0,37	0,59	0,70	0,67	0,99	0,68	0,38	0,83	0,62	0,04
0,34	0,14	0,69	0,13	0,12	0,58	0,98	0,06	0,87	0,29	0,15	0,06	0,23	0,90	0,73	0,53	0,93	0,94	0,66	0,90	0,27	0,87	0,40	0,10
0,31	0,69	0,88	0,57	0,87	0,52	0,52	0,37	0,75	0,41	0,56	0,75	0,94	0,73	0,83	0,94	0,07	0,26	0,19	0,29	0,34	0,14	0,04	0,66
0,50	0,13	0,48	0,91	0,43	0,16	0,40	0,72	0,55	0,96	0,76	0,80	0,52	0,39	0,73	0,95	0,74	0,84	0,35	0,55	0,76	0,46	0,86	0,74
0,03	0,31	0,68	0,50	0,05	0,64	0,28	0,05	0,40	0,23	0,69	0,43	0,63	0,48	0,35	0,36	0,02	0,48	0,24	0,54	0,17	0,59	0,08	0,67
0,21	0,00	0,67	0,12	0,36	0,64	0,87	0,48	0,43	0,51	0,99	0,67	0,20	0,07	0,02	0,06	0,65	0,80	0,58	0,35	0,75	0,23	0,45	0,01
0,29	0,44	0,72	0,64	0,41	0,78	0,20	0,03	0,06	0,98	0,02	0,16	0,94	0,54	0,74	0,56	0,49	0,98	0,18	0,79	0,26	0,37	0,95	0,30
0,47	0,29	0,37	0,71	0,39	0,26	0,40	0,29	0,03	0,40	0,31	0,92	0,62	0,73	0,37	0,93	0,09	0,38	0,18	0,37	0,65	0,52	0,53	0,27
0,66	0,31	0,96	0,09	0,31	0,31	0,44	0,79	0,72	0,46	0,36	0,53	0,74	0,38	0,70	0,03	0,49	0,57	0,78	0,88	0,41	0,51	0,93	0,24
0,56	0,83	0,09	0,66	0,74	0,03	0,73	0,46	0,52	0,86	0,59	0,98	0,74	0,55	0,86	0,64	0,58	0,14	0,14	0,69	0,59	0,31	0,61	0,53
0,35	0,67	0,15	0,22	0,75	0,70	0,42	0,69	0,46	0,82	0,70	0,70	0,94	0,59	0,50	0,85	0,75	0,21	0,10	0,31	0,64	0,05	0,64	0,54
0,57	0,28	0,60	0,20	0,36	0,38	0,16	0,34	0,15	0,99	0,06	0,65	0,98	0,91	0,63	0,71	0,02	0,78	0,95	0,70	0,23	0,34	0,29	0,54
0,50	0,40	0,45	0,87	0,40	0,94	0,11	0,86	0,84	0,65	0,16	0,71	0,14	0,86	0,97	0,92	0,47	0,93	0,98	0,11	0,68	0,07	0,03	0,00
0,51	0,80	0,57	0,34	0,51	0,71	0,77	0,87	0,13	0,54	0,02	0,19	0,87	0,78	0,73	0,22	0,99	0,84	0,60	0,89	0,55	0,05	0,38	0,81
0,56	0,11	0,58	0,16	0,51	0,20	0,50	0,16	0,81	0,66	0,73	0,20	0,44	0,24	0,08	0,50	0,18	0,42	0,25	0,20	0,28	0,40	0,63	0,85
0,30	0,35	0,86	0,18	0,89	0,72	0,26	0,85	0,39	0,70	0,33	0,71	0,85	0,43	0,93	0,80	0,17	0,89	0,96	0,27	0,82	0,04	0,30	0,16
0,26	0,60	0,41	0,89	0,14	0,59	0,36	0,02	0,52	0,31	0,85	0,60	0,51	0,25	0,73	0,31	0,85	0,49	0,16	0,71	0,89	0,08	0,22	0,61
0,43	0,25	0,28	0,48	0,83	0,98	0,76	0,04	0,01	0,37	0,39	0,38	0,01	0,86	0,67	0,48	0,81	0,41	0,98	0,26	0,24	0,61	0,14	0,20
0,48	0,89	0,31	0,09	0,92	0,18	0,22	0,26	0,54	0,19	0,58	0,97	0,20	0,37	0,23	0,31	0,94	0,24	0,91	0,60	0,74	0,79	0,54	0,26
0,73	0,38	0,93	0,94	0,32	0,55	0,95	0,31	0,24	0,51	0,40	0,34	0,45	0,52	0,63	0,10	0,54	0,53	0,39	0,06	0,16	0,86	0,19	0,34
0,24	0,78	0,54	0,98	0,25	0,70	0,78	0,87	0,87	0,35	0,12	0,49	0,69	0,85	0,00	0,33	0,98	0,15	0,98	0,53	0,00	0,35	0,34	0,40
0,12	0,59	0,91	0,71	0,89	0,89	0,02	0,12	0,11	0,38	0,05	0,38	0,94	0,25	0,63	0,31	0,93	0,79	0,21	0,14	0,34	0,68	0,03	0,46
0,84	0,84	0,38	0,73	0,26	0,14	0,09	0,03	0,39	0,21	0,22	0,36	0,26	0,59	0,23	0,52	0,38	0,35	0,88	0,32	0,07	0,10	0,56	0,03

Matriz del elemento 2

0,23	0,82	0,09	0,11	0,97	0,29	0,08	0,89	0,88	0,75	0,41	0,04	0,09	0,07	0,28	0,68	0,95	0,81	0,84	0,88	0,71	0,37	0,56	0,43
0,79	0,59	0,75	0,64	0,54	0,73	0,46	0,59	0,21	0,35	0,82	0,01	0,55	0,75	0,03	0,89	0,60	0,82	0,81	0,70	0,20	0,30	0,74	0,45
0,35	0,44	0,45	0,92	0,40	0,95	0,22	0,48	0,42	0,38	0,20	0,09	0,73	0,13	0,17	0,24	0,52	1,00	0,19	0,44	0,04	0,91	0,24	0,36
0,09	0,16	0,63	0,50	0,84	0,11	0,91	0,01	0,32	0,92	0,97	0,07	0,62	0,72	0,29	0,61	0,81	0,37	0,48	0,76	0,58	0,05	0,17	0,26
0,85	0,85	0,87	0,62	0,58	0,12	0,26	0,69	0,66	0,29	0,07	0,68	0,58	0,56	0,39	0,72	0,52	0,07	0,44	0,58	0,21	0,18	0,10	0,63
0,35	0,89	0,52	0,08	0,55	0,06	0,27	0,72	0,51	0,43	0,69	0,88	0,32	0,19	0,98	0,81	0,85	0,54	0,76	0,06	0,30	0,51	0,40	0,69
0,20	0,75	0,09	0,72	0,91	0,84	0,51	0,76	0,60	0,58	0,76	0,26	0,46	0,01	0,88	0,43	0,20	0,04	0,65	0,60	0,85	0,01	0,56	0,04
0,06	0,38	0,90	0,60	0,05	0,75	0,13	0,31	0,56	0,09	0,93	0,56	0,03	0,89	0,23	0,29	0,15	0,26	0,21	0,83	0,84	0,02	0,41	0,54
0,75	0,52	0,08	0,62	0,68	0,67	0,81	0,48	0,40	0,20	0,09	0,77	0,12	0,88	0,71	0,52	0,78	0,53	0,50	0,52	0,17	0,10	0,79	0,24
0,57	0,27	0,93	0,44	0,81	0,21	0,17	0,34	0,33	0,89	0,24	0,74	0,95	0,68	0,31	0,38	0,26	0,02	0,28	0,32	0,98	0,03	0,53	0,59
0,52	0,67	0,82	0,67	0,28	0,91	0,07	0,50	0,25	0,61	0,55	0,72	0,95	0,28	0,48	0,16	0,25	0,77	0,92	0,93	0,24	0,52	0,54	0,97
0,66	0,08	0,94	0,70	0,56	0,34	0,69	0,38	0,54	0,66	0,44	0,07	0,13	0,89	0,90	0,08	0,79	0,44	0,04	0,57	0,07	0,32	0,72	0,74
0,77	0,87	0,08	0,24	0,96	0,47	0,93	0,33	0,15	0,06	0,67	0,47	0,88	0,73	0,00	0,17	0,50	0,37	0,22	0,79	0,84	0,11	0,65	0,99
0,53	0,56	0,99	0,29	0,87	0,12	0,11	0,09	0,58	0,36	0,69	0,97	0,00	0,30	0,39	0,52	0,95	0,26	0,50	0,58	0,20	0,44	0,96	0,80
0,50	0,33	0,60	0,62	0,60	0,01	0,01	0,77	0,44	0,61	0,79	0,03	0,96	0,01	0,43	0,49	0,85	0,89	0,52	0,66	0,68	0,94	0,24	0,65
0,95	0,84	0,81	0,16	0,26	0,31	0,68	0,79	0,06	0,72	0,29	0,91	0,05	1,00	0,30	0,32	0,71	0,03	0,38	0,43	0,31	0,24	0,33	0,43
0,54	0,03	0,31	0,56	0,51	0,28	0,26	0,27	0,97	0,15	0,36	0,47	0,06	0,92	0,08	0,36	0,05	0,82	0,08	0,61	0,81	0,44	0,31	0,40
0,20	0,51	0,99	0,94	1,00	0,57	0,42	0,44	0,77	0,86	0,31	0,32	0,73	0,97	0,77	0,94	0,11	0,83	0,70	0,38	0,47	0,78	0,91	0,24
0,86	0,65	0,50	0,75	0,11	0,12	0,04	0,12	0,70	0,10	0,09	0,27	0,11	0,37	0,38	0,73	0,44	0,21	0,60	0,93	0,14	0,84	0,20	0,82
0,45	0,87	0,38	0,83	0,14	0,14	0,98	0,96	0,37	0,25	0,03	0,03	0,13	0,17	0,09	0,70	0,33	0,60	1,00	0,71	0,42	0,83	0,50	0,26
0,42	0,41	0,86	0,88	0,53	0,28	0,02	0,68	0,16	0,23	0,69	0,52	0,68	0,56	0,52	0,53	0,18	0,50	0,23	0,64	0,28	0,45	0,75	0,42
0,77	0,73	0,91	0,50	0,66	0,19	0,90	0,78	0,43	0,35	0,47	0,99	0,45	0,30	0,10	0,05	0,54	0,54	0,56	0,72	0,72	0,73	0,84	0,41
0,00	0,37	0,70	0,06	0,72	0,75	0,41	0,17	0,91	0,74	0,32	0,30	0,51	0,19	0,30	0,32	0,54	0,04	0,89	0,03	0,29	0,75	0,68	0,14
0,94	0,22	0,66	0,64	0,35	0,26	0,82	0,02	0,28	0,35	0,22	0,18	0,03	0,38	0,27	0,00	0,44	0,58	0,58	0,69	0,27	0,20	0,39	0,09

Matriz del elemento 3

$$\begin{bmatrix}
0,61 & 0,57 & 0,98 & 0,28 & 0,46 & 0,46 & 0,58 & 0,50 & 0,86 & 0,34 & 0,68 & 0,49 & 0,04 & 0,34 & 0,04 & 0,41 & 0,20 & 0,25 & 0,68 & 0,64 & 0,53 & 0,28 & 0,83 & 0,27 \\
0,14 & 0,84 & 0,17 & 0,70 & 0,22 & 0,99 & 0,36 & 0,07 & 0,98 & 0,67 & 0,64 & 0,69 & 0,01 & 0,65 & 0,32 & 0,27 & 0,06 & 0,15 & 0,50 & 0,37 & 0,33 & 0,95 & 0,03 & 0,17 \\
0,13 & 0,49 & 0,24 & 0,34 & 0,16 & 0,30 & 0,89 & 0,95 & 0,54 & 0,92 & 0,28 & 0,52 & 0,64 & 0,41 & 0,83 & 0,50 & 0,05 & 0,80 & 0,03 & 0,40 & 0,91 & 0,53 & 0,21 & 0,31 \\
0,03 & 0,02 & 0,83 & 0,12 & 0,35 & 0,51 & 0,57 & 0,48 & 0,98 & 0,51 & 0,65 & 0,96 & 0,62 & 0,85 & 0,92 & 0,47 & 0,10 & 0,69 & 0,80 & 0,40 & 0,70 & 0,65 & 0,80 & 0,15 \\
0,65 & 0,75 & 0,94 & 0,56 & 0,92 & 0,46 & 0,91 & 0,93 & 0,69 & 0,78 & 0,35 & 0,94 & 0,54 & 0,09 & 0,72 & 0,20 & 0,19 & 0,43 & 0,67 & 0,54 & 0,10 & 0,39 & 0,46 & 0,42 \\
0,80 & 0,45 & 0,67 & 0,51 & 0,79 & 0,80 & 0,96 & 0,42 & 0,82 & 0,32 & 0,23 & 0,75 & 0,86 & 1,00 & 0,18 & 0,12 & 0,11 & 0,49 & 0,28 & 0,74 & 1,00 & 0,25 & 0,95 & 0,09 \\
0,85 & 0,11 & 0,92 & 0,41 & 0,87 & 0,93 & 0,92 & 0,82 & 0,82 & 0,28 & 0,46 & 0,96 & 0,19 & 0,07 & 0,37 & 0,74 & 0,56 & 0,44 & 0,46 & 0,40 & 0,25 & 0,72 & 0,63 & 0,18 \\
0,98 & 1,00 & 0,02 & 0,59 & 0,29 & 0,90 & 0,15 & 0,39 & 0,33 & 0,17 & 0,80 & 0,04 & 0,26 & 0,35 & 0,92 & 0,24 & 0,31 & 0,44 & 0,09 & 0,63 & 0,11 & 1,00 & 0,80 & 0,64 \\
0,21 & 0,96 & 0,13 & 0,71 & 0,33 & 0,19 & 0,94 & 0,05 & 0,23 & 0,85 & 0,01 & 0,75 & 0,42 & 0,16 & 0,52 & 0,79 & 0,09 & 0,19 & 0,23 & 0,39 & 0,14 & 0,45 & 0,99 & 0,64 \\
0,10 & 0,76 & 0,44 & 0,23 & 0,59 & 0,74 & 0,40 & 0,10 & 0,96 & 0,34 & 0,53 & 0,48 & 0,41 & 0,07 & 0,24 & 0,58 & 0,81 & 0,34 & 0,58 & 0,40 & 0,77 & 0,31 & 0,52 & 0,55 \\
0,14 & 0,71 & 0,39 & 0,91 & 0,23 & 0,88 & 0,78 & 0,93 & 0,48 & 0,92 & 0,43 & 0,30 & 0,24 & 0,20 & 0,87 & 0,37 & 0,53 & 0,85 & 0,68 & 0,16 & 0,03 & 0,17 & 0,70 & 0,64 \\
0,76 & 0,84 & 0,07 & 0,34 & 0,86 & 0,04 & 0,87 & 0,94 & 0,16 & 0,18 & 0,65 & 0,10 & 0,61 & 0,71 & 0,12 & 0,48 & 0,75 & 0,60 & 0,86 & 0,81 & 0,48 & 0,34 & 0,25 & 0,17 \\
0,23 & 0,10 & 0,92 & 0,07 & 0,65 & 0,27 & 0,00 & 0,45 & 0,74 & 0,62 & 0,90 & 0,01 & 0,54 & 0,79 & 0,10 & 0,75 & 0,83 & 0,68 & 0,41 & 0,86 & 0,29 & 0,79 & 0,45 & 0,05 \\
0,90 & 0,58 & 0,74 & 0,47 & 0,13 & 0,63 & 0,39 & 0,23 & 0,74 & 0,13 & 0,47 & 0,71 & 0,69 & 0,23 & 0,14 & 0,48 & 0,58 & 0,10 & 0,41 & 0,14 & 0,58 & 0,18 & 0,04 & 0,61 \\
0,51 & 0,70 & 0,75 & 0,29 & 0,66 & 0,82 & 0,62 & 0,33 & 0,23 & 0,61 & 0,67 & 0,09 & 0,81 & 0,06 & 0,81 & 0,49 & 0,06 & 0,25 & 0,17 & 0,38 & 0,28 & 0,53 & 0,63 & 0,23 \\
0,50 & 0,63 & 0,38 & 0,29 & 0,03 & 0,87 & 0,03 & 0,21 & 0,42 & 0,27 & 0,99 & 0,47 & 0,61 & 0,27 & 0,68 & 0,09 & 0,91 & 0,93 & 0,39 & 0,94 & 0,80 & 0,40 & 0,60 & 0,81 \\
0,34 & 0,79 & 0,09 & 0,04 & 0,50 & 0,15 & 0,27 & 0,91 & 0,18 & 0,88 & 0,24 & 0,85 & 0,54 & 0,15 & 0,45 & 0,23 & 0,90 & 0,50 & 0,04 & 0,72 & 0,60 & 0,36 & 0,41 & 0,99 \\
0,63 & 0,15 & 0,52 & 0,14 & 0,24 & 0,36 & 0,93 & 0,15 & 0,87 & 0,64 & 0,36 & 0,08 & 0,78 & 0,66 & 0,59 & 0,72 & 0,43 & 0,77 & 0,35 & 0,12 & 0,08 & 0,10 & 0,89 & 0,46 \\
0,12 & 0,73 & 0,96 & 0,25 & 0,43 & 0,28 & 0,92 & 0,17 & 0,93 & 0,57 & 0,89 & 0,24 & 0,31 & 0,26 & 0,35 & 0,59 & 0,41 & 0,42 & 0,27 & 0,94 & 0,30 & 0,64 & 0,88 & 0,36 \\
0,07 & 0,08 & 0,22 & 0,56 & 0,29 & 0,31 & 0,10 & 0,86 & 0,06 & 0,60 & 0,73 & 0,04 & 0,90 & 0,37 & 0,89 & 0,85 & 0,01 & 0,18 & 0,91 & 0,72 & 0,16 & 0,41 & 0,27 & 0,86 \\
0,05 & 0,48 & 0,19 & 0,68 & 0,05 & 0,18 & 0,36 & 0,89 & 0,37 & 0,25 & 0,62 & 0,79 & 1,00 & 0,53 & 0,02 & 0,59 & 0,25 & 0,09 & 0,68 & 0,28 & 0,18 & 0,28 & 0,60 & 0,01 \\
0,45 & 0,22 & 0,49 & 0,91 & 0,01 & 0,72 & 0,78 & 0,75 & 0,26 & 0,40 & 0,57 & 0,36 & 0,77 & 0,62 & 0,95 & 0,78 & 0,67 & 0,55 & 0,91 & 0,55 & 0,20 & 0,57 & 0,96 & 0,29 \\
0,57 & 0,90 & 0,59 & 0,24 & 0,44 & 0,65 & 0,46 & 0,55 & 0,83 & 0,30 & 0,11 & 0,87 & 0,20 & 0,96 & 0,39 & 0,11 & 0,62 & 0,29 & 0,59 & 0,93 & 0,21 & 0,30 & 0,08 & 0,09 \\
0,96 & 0,22 & 0,77 & 0,33 & 0,01 & 0,08 & 0,52 & 0,31 & 0,72 & 0,31 & 0,65 & 0,90 & 0,03 & 0,62 & 0,76 & 0,81 & 0,27 & 0,43 & 0,07 & 0,32 & 0,77 & 0,88 & 0,23 & 0,47
\end{bmatrix}$$

Matriz ensamblada usando la rutina implementada

0,98	0,82	0,32	0,18	0,30	0,32	0,50	0,37	0,28	0,49	0,35	0,75	0,04	0,56	0,67	0,06	0,28	0,02	0,17	0,43	0,15	0,96	0,25	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,66	0,77	0,59	0,40	0,72	0,04	0,03	0,34	0,52	0,58	0,33	0,40	0,87	0,14	0,37	0,59	0,70	0,67	0,99	0,68	0,38	0,83	0,62	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,34	0,14	0,69	0,13	0,12	0,58	0,98	0,06	0,87	0,29	0,15	0,06	0,23	0,90	0,73	0,53	0,93	0,94	0,66	0,90	0,27	0,87	0,40	0,10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,31	0,69	0,88	0,57	0,87	0,52	0,52	0,37	0,75	0,41	0,56	0,75	0,94	0,73	0,83	0,94	0,07	0,26	0,19	0,29	0,34	0,14	0,04	0,66	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,50	0,13	0,48	0,91	0,43	0,16	0,40	0,72	0,55	0,96	0,76	0,80	0,52	0,39	0,73	0,95	0,74	0,84	0,35	0,55	0,76	0,46	0,86	0,74	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,03	0,31	0,68	0,50	0,05	0,64	0,28	0,05	0,40	0,23	0,69	0,43	0,63	0,48	0,35	0,36	0,02	0,48	0,24	0,54	0,17	0,59	0,08	0,67	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,21	0,00	0,67	0,12	0,36	0,64	0,87	0,48	0,43	0,51	0,99	0,67	0,20	0,07	0,02	0,06	0,65	0,80	0,58	0,35	0,75	0,23	0,45	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,29	0,44	0,72	0,64	0,41	0,78	0,20	0,03	0,06	0,98	0,02	0,16	0,94	0,54	0,74	0,56	0,49	0,98	0,18	0,79	0,26	0,37	0,95	0,30	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,47	0,29	0,37	0,71	0,39	0,26	0,40	0,29	0,03	0,40	0,31	0,92	0,62	0,73	0,37	0,93	0,09	0,38	0,18	0,37	0,65	0,52	0,53	0,27	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,66	0,31	0,96	0,09	0,31	0,31	0,44	0,79	0,72	0,46	0,36	0,53	0,74	0,38	0,70	0,03	0,49	0,57	0,78	0,88	0,41	0,51	0,93	0,24	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,56	0,83	0,09	0,66	0,74	0,03	0,73	0,46	0,52	0,86	0,59	0,98	0,74	0,55	0,86	0,64	0,58	0,14	0,14	0,69	0,59	0,31	0,61	0,53	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,35	0,67	0,15	0,22	0,75	0,70	0,42	0,69	0,46	0,82	0,70	0,70	0,94	0,59	0,50	0,85	0,75	0,21	0,10	0,31	0,64	0,05	0,64	0,54	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,57	0,28	0,60	0,20	0,36	0,38	0,16	0,34	0,15	0,99	0,06	0,65	1,21	1,72	0,72	0,82	0,99	1,08	1,03	1,59	1,11	1,08	0,70	0,58	0,09	0,07	0,28	0,68	0,95	0,81	0,84	0,88	0,71	0,37	0,56	0,43	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,50	0,40	0,45	0,87	0,40	0,94	0,11	0,86	0,84	0,65	0,16	0,71	0,93	1,45	1,72	1,56	1,01	1,66	1,44	0,70	0,89	0,42	0,85	0,01	0,55	0,75	0,03	0,89	0,60	0,82	0,81	0,70	0,20	0,30	0,74	0,45	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,51	0,80	0,57	0,34	0,51	0,71	0,77	0,87	0,13	0,54	0,02	0,19	1,22	1,22	1,19	1,14	1,39	1,79	0,82	1,37	0,98	0,43	0,58	0,90	0,73	0,13	0,17	0,24	0,52	1,00	0,19	0,44	0,04	0,91	0,24	0,36	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,56	0,11	0,58	0,16	0,51	0,20	0,50	0,16	0,81	0,66	0,73	0,20	0,54	0,41	0,71	1,00	1,02	0,53	1,17	0,22	0,60	1,32	1,60	0,92	0,62	0,72	0,29	0,61	0,81	0,37	0,48	0,76	0,58	0,05	0,17	0,26	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,30	0,35	0,86	0,18	0,89	0,72	0,26	0,85	0,39	0,70	0,33	0,71	1,70	1,28	1,81	1,42	0,75	1,01	1,22	0,97	1,49	0,33	0,37	0,84	0,58	0,56	0,39	0,72	0,52	0,07	0,44	0,58	0,21	0,18	0,10	0,63	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,26	0,60	0,41	0,89	0,14	0,59	0,36	0,02	0,52	0,31	0,85	0,60	0,86	1,14	1,26	0,39	1,40	0,55	0,43	1,43	1,40	0,51	0,91	1,49	0,32	0,19	0,98	0,81	0,85	0,54	0,76	0,06	0,30	0,51	0,40	0,69	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,43	0,25	0,28	0,48	0,83	0,98	0,76	0,04	0,01	0,37	0,39	0,38	0,21	1,60	0,76	1,20	1,72	1,25	1,48	1,01	0,84	1,19	0,89	0,46	0,46	0,01	0,88	0,43	0,20	0,04	0,65	0,60	0,85	0,01	0,56	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,48	0,89	0,31	0,09	0,92	0,18	0,22	0,26	0,54	0,19	0,58	0,97	0,26	0,75	1,13	0,90	0,99	1,00	1,04	0,91	1,30	0,89	1,47	0,82	0,03	0,89	0,23	0,29	0,15	0,26	0,21	0,83	0,84	0,02	0,41	0,54	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,73	0,38	0,93	0,94	0,32	0,55	0,95	0,31	0,24	0,51	0,40	0,34	1,21	1,04	0,70	0,72	1,22	1,20	1,20	0,54	0,56	1,06	0,29	1,11	0,12	0,88	0,71	0,52	0,78	0,53	0,50	0,52	0,17	0,10	0,79	0,24	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,24	0,78	0,54	0,98	0,25	0,70	0,78	0,87	0,87	0,35	0,12	0,49	1,27	1,12	0,93	0,77	1,80	0,36	1,15	0,87	0,33	1,23	0,58	1,15	0,95	0,68	0,31	0,38	0,26	0,02	0,28	0,32	0,98	0,03	0,53	0,59	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,12	0,59	0,91	0,71	0,89	0,89	0,02	0,12	0,11	0,38	0,05	0,38	1,46	0,92	1,45	0,98	1,21	1,70	0,28	0,64	0,59	1,28	0,58	1,17	0,95	0,28	0,48	0,16	0,25	0,77	0,92	0,93	0,24	0,52	0,54	0,97	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,84	0,84	0,38	0,73	0,26	0,14	0,09	0,03	0,39	0,21	0,22	0,36	0,91	0,67	1,18	1,22	0,93	0,68	1,57	0,71	0,61	0,76	0,99	0,10	0,13	0,89	0,90	0,08	0,79	0,44	0,04	0,57	0,07	0,32	0,72	0,74	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,77	0,87	0,08	0,24	0,96	0,47	0,93	0,33	0,15	0,06	0,67	0,47	1,49	1,30	0,99	0,45	0,96	0,83	0,81	1,29	1,70	0,46	1,32	1,48	0,04	0,34	0,04	0,41	0,20	0,25	0,68	0,64	0,53	0,28	0,83	0,27
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,53	0,56	0,99	0,29	0,87	0,12	0,11	0,09	0,58	0,36	0,69	0,97	0,14	1,13	0,56	1,22	1,17	1,25	0,86	0,65	1,18	1,11	1,60	1,49	0,01	0,65	0,32	0,27	0,06	0,15	0,50	0,37	0,33	0,95	0,03	0,17
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,50	0,33	0,60	0,62	0,60	0,01	0,01	0,77	0,44	0,61	0,79	0,03	1,09	0,50	0,68	0,83	1,00	1,19	1,41	1,61	1,21	1,86	0,52	1,16	0,64	0,41	0,83	0,50	0,05	0,80	0,03	0,40	0,91	0,53	0,21	0,31
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,95	0,84	0,81	0,16	0,26	0,31	0,68	0,79	0,06	0,72	0,29	0,91	0,09	1,02	1,13	0,44	1,07	0,54	0,95	0,91	1,29	0,75	0,98	1,39	0,62	0,85	0,92	0,47	0,10	0,69	0,80	0,40	0,70	0,65	0,80	0,15
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,54	0,03	0,31	0,56	0,51	0,28	0,26	0,27	0,97	0,15	0,36	0,47	0,71	1,68	1,03	0,92	0,96	1,28	0,98	1,55	1,50	1,21	0,66	1,34	0,54	0,09	0,72	0,20	0,19	0,43	0,67	0,54	0,10	0,39	0,46	0,42
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,20	0,51	0,99	0,94	1,00	0,57	0,42	0,44	0,77	0,86	0,31	0,32	1,53	1,41	1,43	1,45	0,90	1,63	1,66	0,80	1,29	1,11	1,14	0,99	0,86	1,00	0,18	0,12	0,11	0,49	0,28	0,74	1,00	0,25	0,95	0,09
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,86	0,65	0,50	0,75	0,11	0,12	0,04	0,12	0,70	0,10	0,09	0,27	0,96	0,48	1,31	1,15	1,31	1,14	1,52	1,74	0,97	1,12	0,65	1,78	0,19	0,07	0,37	0,74	0,56	0,44	0,46	0,40	0,25	0,72	0,63	0,18
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,45	0,87	0,38	0,83	0,14	0,14	0,98	0,96	0,37	0,25	0,03	0,03	1,10	1,17	0,11	1,29	0,62	1,50	1,14	1,10	0,75	1,00	1,30	0,30	0,26	0,35	0,92	0,24	0,31	0,44	0,09	0,63	0,11	1,00	0,80	0,64
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,42	0,41	0,86	0,88	0,53	0,28	0,02	0,68	0,16	0,23	0,69	0,52	0,89	1,52	0,65	1,23	0,51	0,69	1,17	0,69	0,51	1,31	0,76	1,18	0,42	0,16	0,52	0,79	0,09	0,19	0,23	0,39	0,14	0,45	0,99	0,64
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,77	0,73	0,91	0,50	0,66	0,19	0,90	0,78	0,43	0,35	0,47	0,99	0,54	1,06	0,54	0,28	1,14	1,28	0,96	0,82	1,68	1,07	1,38	0,89	0,41	0,07	0,24	0,58	0,81	0,34	0,58	0,40	0,77	0,31	0,52	0,55
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,37	0,70	0,06	0,72	0,75	0,41	0,17	0,91	0,74	0,32	0,30	0,65	0,90	0,69	1,23	0,77	0,92	1,67	0,96	0,76	1,67	1,12	0,44	0,24	0,20	0,87	0,37	0,53	0,85	0,68	0,16	0,03	0,17	0,70	0,64
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,94	0,22	0,66	0,64	0,35	0,26	0,82	0,02	0,28	0,35	0,22	0,18	0,79	1,21	0,35	0,34	1,30	0,63	1,45	1,63	0,42	0,37	1,04	0,19	0,61	0,71	0,12	0,48	0,75	0,60	0,86	0,81	0,48	0,34	0,25	0,17
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,23	0,10	0,92	0,07	0,65	0,27	0,00	0,45	0,74	0,62	0,90	0,01	0,54	0,79	0,10	0,75	0,83	0,68	0,41	0,86	0,29	0,79	0,45	0,05
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,90	0,58	0,74	0,47	0,13	0,63	0,39	0,23	0,74	0,13	0,47	0,71	0,69	0,23	0,14	0,48	0,58	0,10	0,41	0,14	0,58	0,18	0,04	0,61
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,51	0,70	0,75	0,29	0,66	0,82	0,62	0,33	0,23	0,61	0,67	0,09	0,81	0,06	0,81	0,49	0,06	0,25	0,17	0,38	0,28	0,53	0,63	0,23
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,50	0,63	0,38	0,29	0,03	0,87	0,03	0,21	0,42	0,27	0,99	0,47	0,61	0,27	0,68	0,09	0,91	0,93	0,39	0,94	0,80	0,40	0,60	0,81
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,34	0,79	0,09	0,04	0,50	0,15	0,27	0,91	0,18	0,88	0,24	0,85	0,54	0,15	0,45	0,23	0,90	0,50	0,04	0,72	0,60	0,36	0,41	0,99
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,63	0,15	0,52	0,14	0,24	0,36	0,93	0,15	0,87	0,64	0,36	0,08	0,78	0,66	0,59	0,72	0,43	0,77	0,35	0,12	0,08	0,10	0,89	0,46
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,12	0,73	0,96	0,25	0,43	0,28	0,92	0,17	0,93	0,57	0,89	0,24	0,31	0,26	0,35	0,59	0,41	0,42	0,27	0,94	0,30	0,64	0,88	0,36
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,07	0,08	0,22	0,56	0,29	0,31	0,10	0,86	0,06	0,60	0,73	0,04	0,90	0,37	0,89	0,85	0,01	0,18	0,91	0,72	0,16	0,41	0,27	0,86
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,05	0,48	0,19	0,68	0,05	0,18	0,36	0,89	0,37	0,25	0,62	0,79	1,00	0,53	0,02	0,59	0,25	0,09	0,68	0,28	0,18	0,28	0,60	0,01
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,45	0,22	0,49	0,91	0,01	0,72	0,78	0,75	0,26	0,40	0,57	0,36	0,77	0,62	0,95	0,78	0,67	0,55	0,91	0,55	0,20	0,57	0,96	0,29
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,57	0,90	0,59	0,24	0,44	0,65	0,46	0,55	0,83	0,30	0,11	0,87	0,20	0,96	0,39	0,11	0,62	0,29	0,59	0,93	0,21	0,30	0,08	0,09
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,96	0,22	0,77	0,33	0,01	0,08	0,52	0,31	0,72	0,31	0,65	0,90	0,03	0,62	0,76	0,81	0,27	0,43	0,07	0,32	0,77	0,88	0,23	0,47

Matriz ensamblada manualmente

Diferencias entre las dos matrices

| | | |

Anejo IV: Validación SIM

EJEMPLO I

DATOS DE ENTRADA

$$\boldsymbol{M} = 10^4 \begin{bmatrix} 20 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 10 \end{bmatrix} \quad , \quad \boldsymbol{K} = 10^7 \begin{bmatrix} 5 & -2 & 0 \\ -2 & 3 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

RESULTADOS ESPERADOS

$$\boldsymbol{\Phi} = [\boldsymbol{\phi}_1 \quad \boldsymbol{\phi}_2 \quad \boldsymbol{\phi}_3] = \begin{bmatrix} 1 & 1 & 1 \\ 2.15 & 0.89 & -1.04 \\ 3.31 & -1.46 & 0.41 \end{bmatrix}$$

$$\boldsymbol{\Omega} = \begin{Bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{Bmatrix} = \begin{Bmatrix} 5.92 \\ 12.69 \\ 18.81 \end{Bmatrix}$$

RESULTADOS OBTENIDOS

w	frecuencias
5,928400	0,943542
12,675200	2,017316
18,820000	2,995301

Formas de vibración		
0,000709	0,001400	-0,001600
0,001500	0,001200	0,001700
0,002300	-0,002000	-0,000665

→

Formas de vibración normalizadas		
1,000000	1,000000	1,000000
2,115984	0,857143	-1,062500
3,244509	-1,428571	0,415781

COMPARACIÓN DE LOS RESULTADOS

w		
E.absoluto	E. relativo	%
0,008400	0,001419	0,141892
-0,014800	-0,001166	-0,116627
0,010000	0,000532	0,053163

Formas de vibración								
E. relativo			%			E. absoluto		
0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
-0,015821	-0,036918	0,021635	-1,582133	-3,691814	2,163462	-0,034016	-0,032857	-0,022500
-0,019786	-0,021526	0,014101	-1,978579	-2,152642	1,410061	-0,065491	0,031429	0,005781

EJEMPLO II

DATOS DE ENTRADA

$$\mathbf{M} = \begin{bmatrix} 105750 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 99750 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 95138 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 93075 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 92325 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 90638 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 88950 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 86888 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 85575 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 63638 \end{bmatrix}$$

$$\mathbf{K} = 1E9 \begin{bmatrix} 0.848 & -0.5964 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.5964 & 1.1928 & -0.5964 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.5964 & 0.884 & -0.2876 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.2876 & 0.5752 & -0.2876 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.2876 & 0.5752 & -0.2876 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.2876 & 0.4054 & -0.1178 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.1178 & 0.2356 & -0.1178 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.1178 & 0.1551 & -0.0373 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0373 & 0.0746 & -0.0373 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0373 & 0.0373 \end{bmatrix}$$

RESULTADOS ESPERADOS

$$\omega^2 = \begin{bmatrix} 8.185352772 \\ 16.79285562 \\ 27.85677655 \\ 36.01388621 \\ 43.61192497 \\ 59.78294071 \\ 72.73926038 \\ 93.1611507 \\ 105.1094667 \\ 138.3003977 \end{bmatrix}$$

$$\phi = \begin{bmatrix} 0.000207 & -0.000486 & -0.000768 & 0.000536 & -0.001108 & 0.00067 & -0.00117 & 0.001614 & 0.001086 & -0.001212 \\ 0.000292 & -0.000666 & -0.000986 & 0.000639 & -0.001202 & 0.000528 & -0.000566 & -0.000189 & -0.000583 & 0.002387 \\ 0.000374 & -0.000815 & -0.001076 & 0.000603 & -0.000913 & 0.000071 & 0.000539 & -0.001718 & -0.001175 & -0.00165 \\ 0.000535 & -0.001049 & -0.000987 & 0.00027 & 0.00026 & -0.000961 & 0.001887 & 0.000044 & 0.001892 & 0.000419 \\ 0.000684 & -0.001186 & -0.000649 & -0.000176 & 0.001273 & -0.000881 & 0.000004 & 0.001683 & -0.001806 & -0.000107 \\ 0.000819 & -0.001217 & 0.00015 & -0.000549 & 0.001509 & 0.00021 & -0.001886 & -0.001366 & 0.0009 & 0.000023 \\ 0.001107 & -0.001027 & 0.001158 & -0.000911 & -0.000123 & 0.002296 & 0.001176 & 0.000314 & -0.000145 & -0.000002 \\ 0.001339 & -0.000619 & 0.001787 & -0.000381 & -0.001579 & -0.001813 & -0.000461 & -0.000062 & 0.000021 & 0 \\ 0.001863 & 0.001078 & 0.000543 & 0.002448 & 0.000821 & 0.000302 & 0.000046 & 0.000003 & -0.000001 & 0 \\ 0.002102 & 0.002077 & -0.001669 & -0.002014 & -0.000366 & -0.000059 & -0.000006 & 0 & 0 & 0 \end{bmatrix}$$

RESULTADOS OBTENIDOS

Frecuencias	**w**
1,299508131	8,165050393
2,672236892	16,790159577
4,434784709	27,864574127
5,734211926	36,029116122
6,941458765	43,614471720

Formas de vibración				
-0,000207	0,000486	-0,000768	0,000536	0,001108
-0,000292	0,000666	-0,000986	0,000639	0,001202
-0,000374	0,000815	-0,001076	0,000603	0,000913
-0,000535	0,001049	-0,000986	0,000270	-0,000260
-0,000685	0,001186	-0,000649	-0,000177	-0,001273
-0,000820	0,001217	-0,000150	-0,000550	-0,001509
-0,001107	0,001027	0,001158	-0,000911	0,000124
-0,001339	0,000618	0,001787	-0,000379	0,001579
-0,001863	-0,001078	0,000542	0,002447	-0,000823
-0,002102	-0,002077	-0,001669	-0,002015	0,000366

COMPARACIÓN DE LOS RESULTADOS

Frecuencias			**w**	
E.absoluto	E. relativo	%	E. relativo	%
-0,003231224	-0,00248033	-0,248033032	-0,00248033	-0,248033032
-0,000429089	-0,000160547	-0,016054702	-0,000160547	-0,016054702
0,001241023	0,000279917	0,027991669	0,000279917	0,027991669
0,002423916	0,00042289	0,042288998	0,00042289	0,042288998
0,000405328	5,83957E-05	0,005839573	5,83957E-05	0,005839573

Formas de vibración

E. relativo

0,000000	0,000000	0,000000	0,000000	0,000000
0,000000	0,001100	0,000264	-0,000394	-0,000242
0,000000	0,001262	0,000411	-0,000303	0,000099
-0,000021	0,000503	-0,000022	-0,002032	-0,000051
0,000892	0,001074	0,000788	0,006309	0,000011
0,000708	0,000472	0,001026	0,001581	-0,000214
0,000060	0,000438	0,000598	-0,000297	0,008807
-0,000136	-0,000622	0,000684	-0,005486	-0,000243
-0,000099	0,000738	-0,001632	-0,000642	0,002091
-0,000052	0,000712	0,000403	0,000013	0,001096

%

0,000000	0,000000	0,000000	0,000000	0,000000
0,000000	0,110026	0,026352	-0,039408	-0,024232
0,000000	0,126247	0,041131	-0,030314	0,009904
-0,002056	0,050260	-0,002227	-0,203192	-0,005138
0,089181	0,107400	0,078758	0,630899	0,001075
0,070818	0,047190	0,102554	0,158107	-0,021377
0,005996	0,043834	0,059832	-0,029722	0,880723
-0,013576	-0,062166	0,068358	-0,548645	-0,024282
-0,009946	0,073841	-0,163247	-0,064189	0,209140
-0,005233	0,071203	0,040307	0,001338	0,109594

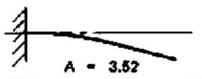
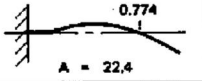
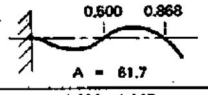
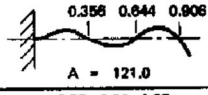
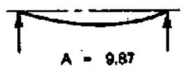
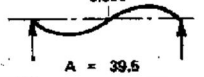
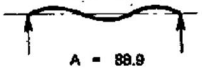
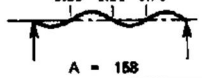
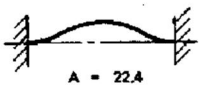
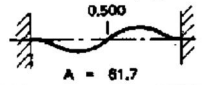
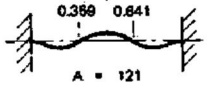
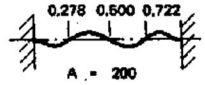
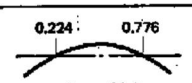
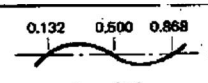
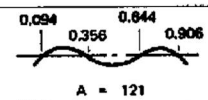
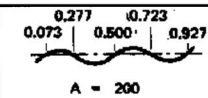
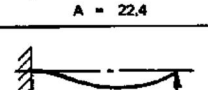
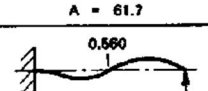
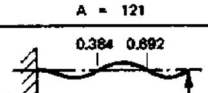
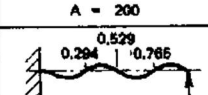
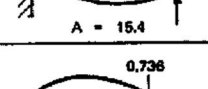
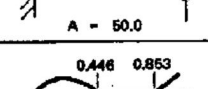
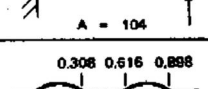
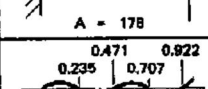
Anejo V

Tabla para determinar la frecuencia natural de vigas con distintas condiciones de vinculo

Beams of uniform section and uniformly distributed load

Natural frequencies $f_n = \frac{A}{2\pi}\sqrt{\frac{EI}{\rho S l^4}}$

where E = Young's modulus
I = Area moment of inertia of beam cross section
l = Length of beam
ρ = Mass density of beam material
S = Area of cross-section
A = Coefficient from table below

Clamped – free (Cantilever)	 A = 3.52	 0.774 A = 22.4	 0.500 0.868 A = 61.7	 0.356 0.644 0.906 A = 121.0
Hinged – hinged (Simple)	 A = 9.87	 0.500 A = 39.5	 0.333 0.667 A = 88.9	 0.25 0.50 0.75 A = 158
Clamped – clamped (Built-in)	 A = 22.4	 0.500 A = 61.7	 0.359 0.641 A = 121	 0.278 0.500 0.722 A = 200
Free – free	 0.224 0.776 A = 22.4	 0.132 0.500 0.868 A = 61.7	 0.094 0.644 0.356 0.906 A = 121	 0.277 0.723 0.073 0.500 0.927 A = 200
Clamped – hinged	 A = 15.4	 0.560 A = 50.0	 0.384 0.692 A = 104	 0.529 0.294 0.765 A = 178
Hinged – free	 0.736 A = 15.4	 0.446 0.853 A = 50.0	 0.308 0.616 0.898 A = 104	 0.471 0.922 0.235 0.707 A = 178

271281