Conversión de Datos de Patran a MODULEF; patmod

patmod es un programa escrito en Fortran 90, pensado para ejecutarse desde Windows, que convierte archivos .bdf obtenidos mediante Patran (Patran 2003 release 2) en archivos de malla MODULEF (con o sin formato).

Este programa está destinado únicamente a convertir la geometría y la malla del archivo .bdf de Patran en un archivo de malla MODULEF, empleando distintas fuerzas definidas sobre la geometría para diferenciar distintas condiciones de contorno y distintos materiales para diferenciar distintos subdominios.

1. Estructura

patmod está formado por un programa principal, **patran-modulef.f90**, en el que se realiza toda la conversión y tres funciones auxiliares, **dos.f90**, **tres.f90** y **cuatro.f90**, que intervienen en el cálculo de las referencias de las aristas, la primera de ellas, y de las referencias de las caras, las otras dos.

2. Utilización

2.1. Entrada

Hay que situar **patmod.exe** y el archivo .bdf a convertir en el mismo directorio, ejecutar **patmod.exe** y completar los datos que el programa va solicitando (el nombre del archivo .bdf que se pide debe llevar extensión).

Se han de conocer de antemano el número de nodos y el número de elementos. Estos datos se pueden obtener desde Patran o consultando el archivo .bdf. Los nodos han de estar numerados de forma consecutiva del número uno en adelante. En el caso de los elementos no es necesario.

Las condiciones Dirichlet se imponen en Patran como desplazamientos (displacement) y las Neumann como fuerzas (force). De este modo, distintos desplazamientos y distintas fuerzas (el momento no influye) en Patran dan lugar a distintas condiciones Dirichlet y Neumann, respectivamente, en la malla MODULEF. En Patran, todas las condiciones de contorno tienen que estar definidas sobre la geometría y no sobre la malla. Además, hemos de tener en cuenta que en aquellos vértices en los que coincidan condiciones Dirichlet con Neumann, prevalece la condición Dirichlet y en aquellos en los que coincidan condiciones del mismo tipo, prevalecen las primeras impuestas. (La asignación de referencias necesita ser revisada).

Los tipos de elementos soportados por el conversor son

en dos dimensiones: Tria3 Tria6 Tria6* Quad4 Quad8 Quad8*

y en tres dimensiones: Tet4 Wedge6 Hex8

Los elementos Tria6* son elementos Tria6 de Patran que al pasar a la malla MODULEF se convierten en triángulos con nodos los puntos medios de los lados. Y, de igual modo, los elementos Quad8* son elementos Quad8 de Patran que al pasar a la malla MODULEF se convierten en cuadriláteros con nodos los puntos medios de los lados.

En Patran, los casos bidimensionales han de estar sobre el plano XY.

2.2. Salida

Tras la ejecución se obtiene un archivo de malla MODULEF, es decir con la siguiente estructura

número de elementos, número de nodos, número de vértices

matriz de conectividades de los nodos (en caso de que estos sean distintos a los vértices), matriz de conectividades de los vértices, número de referencia de caras (de haberlas), número de referencia de aristas, número de referencia de vértices, coordenadas de los vértices (dos o tres por vértice según estemos en una geometría en dos o tres dimensiones)

número de subdominio

Si han seleccionado los tipos de elemento Tria6* o Quad8*, al final del archivo de malla MODULEF creado, en un registro nuevo después de los números de subdominio, se dan las coordenadas de los nodos.

Todas las coordenadas del archivo generado están definidas como reales de doble precisión.

Por último, si se selecciona la opción de malla con formato, **patmod** tratará a los enteros con el formato $i10$ y a los reales con el $e12.5$

Departamento de Matemática Aplicada. USC

Inés Santos Atienza 28 - II - 2005

A. ANEXO: Modificaciones para el uso de patmod con ficheros de datos MD Nastran obtenidos a través del mallador Hypermesh

A.1. Introducción

Buena parte de la información de este anexo ha sido obtenida del fichero "MD Nastran 2006 Quick Reference Guide", incluido en la ayuda de la aplicación MD Nastran 2006.

A.1.1. El fichero de datos Nastran

Un fichero de datos Nastran (o *MD Nastran input data file*) se identifica por las extensiones `.dat` o `.bdf`. Su estructura se muestra en la Figura 1. Puede constar de cinco secciones:

- Las dos primeras, las sentencias Nastran (*Nastran statements*) y la sección de gestión de ficheros (*File management section*) son las únicas opcionales y pueden terminar con el delimitador opcional "ID A, B".
- La sección de control ejecutivo (*Executive control section*) termina con el delimitador obligatorio "CEND".
- La seccion de control de casos (*Case control section*) termina con el delimitador obligatorio "BEGIN BULK".
- La sección de datos primarios (*Bulk data section*) termina con el delimitador "ENDDATA".

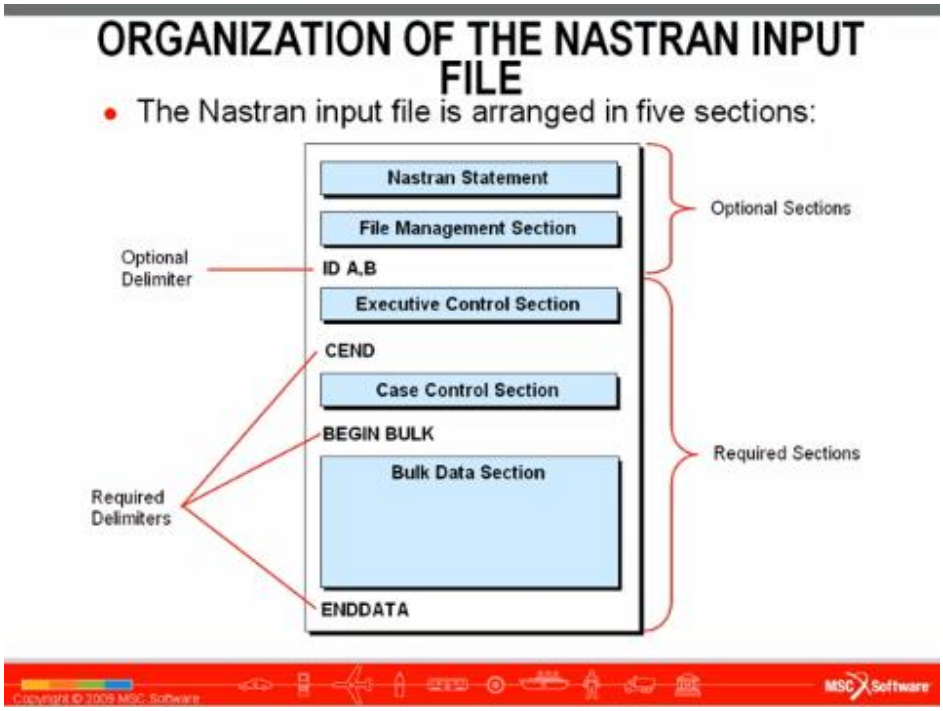


Figura 1: Estructura de un fichero de datos Nastran

Esta última sección es la que nos interesa, ya que permite especificar la geometría del modelo, la conectividad y las propiedades de los elementos, los materiales, las restricciones y las cargas[1].

A.1.2. La sección de datos primarios

La sección de datos primarios está compuesto de "entradas" y cada entrada contiene "datos" distribuidos en "campos". Cada linea de datos consta de 10 campos. El nombre de la entrada se da en el primer campo de la primera linea. El décimo campo de cada linea sirve para indicar si hay linea adicional de campos en la entrada. Un ejemplo de esta estructura puede verse en la Figura 2.

Los datos solo pueden ser enteros, reales o cadenas alfanuméricas. Las cadenas son de hasta 8 caracteres. Los reales pueden darse de varias maneras. Por ejemplo, son el mismo número `7.0`, `.7E1`, `0.7+1`, `.70+1`, `7.E+0` y `70.-1`.

MD Nastran tiene tres formatos distintos para los datos de una entrada:

- Formato de campo libre (*Free Field Format*): los campos de datos están separados por comas.

[1]Los que pretendan usar Nastran deben saber que esta aplicación, de entre todas las cargas definidas en los datos primarios, solo considera aquéllas que se indican a través del comando correspondiente en la sección de control de casos.

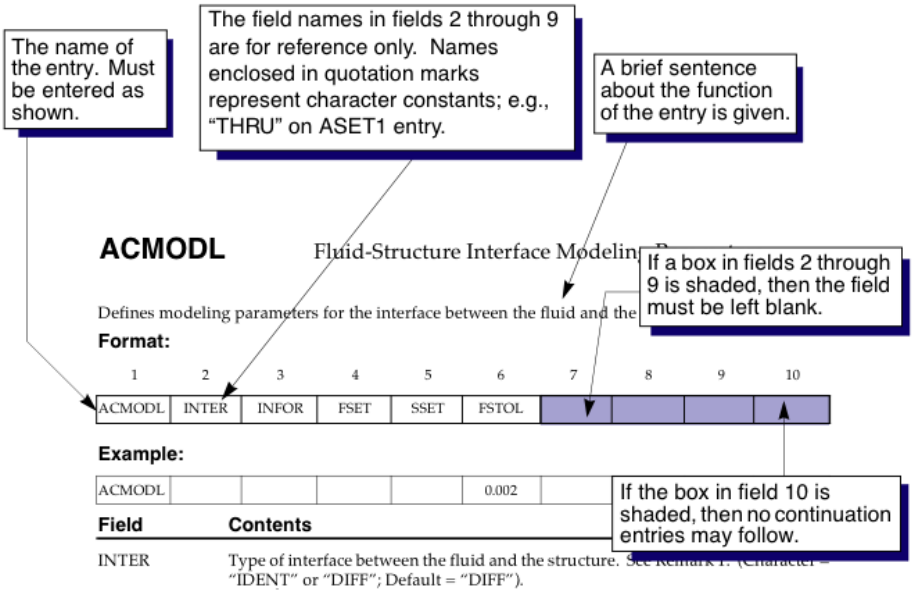


Figura 2: Esquema de una entrada de la sección de datos primarios.

- Formato de campo corto (*Small Field Format*): Hay 10 campos contiguos de 8 caracteres cada uno.
- Formato de campo largo (*Large Field Format*): Hay 10 campos contiguos de 16 caracteres cada uno.

El tipo de formato leído por patmod es el formato de campo corto.

A.1.3. Las entradas relevantes para patmod

Las entradas de las sección de datos primarios que patmod es capaz de reconocer son:

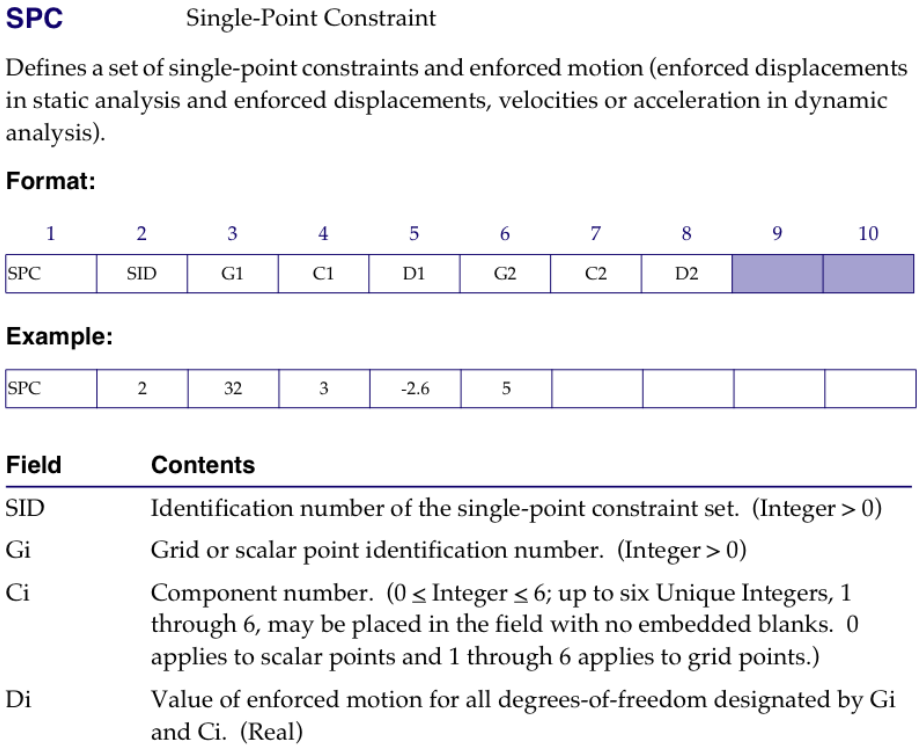
- Conectividad de los elementos: las entradas involucradas son CTRIA3, CTRIA6, CQUAD4, CQUAD8, CTETRA, CPENTA y CHEXA. Sirven para obtener las variables *mm* y *nn* de RECONVXX.
- Coordenadas de los vértices: la entrada GRID. Se obtiene la variable z de RECONVXX.
- Referencias Dirichlet de los vértices: la entrada SPC1, implementada por Inés y la entrada SPC, implementada por Fran. Se obtiene la variable *nrv*.
- Referencias Neumann de los vértices: la entrada LOAD, implementada por Inés (y modificada por Fran). Se obtiene la variable *nrv*.

A.2. Modificaciones al código de Inés

Respecto de las modificaciones hechas al código de Inés, resaltamos las siguientes:

- En el caso CTRIA3 y CTETRA, si el determinante de la matriz de la transformación de paso al elemento de referencia es negativo, se intercambian los dos últimos nodos para obtener un valor positivo.

- La inclusión de la entrada SPC se debe a que es la utilizada por defecto por Hypermesh para incluir condiciones de tipo Dirichlet. La estructura de la entrada SPC puede verse en la Figura 3.



SPC Single-Point Constraint

Defines a set of single-point constraints and enforced motion (enforced displacements in static analysis and enforced displacements, velocities or acceleration in dynamic analysis).

Format:

1	2	3	4	5	6	7	8	9	10
SPC	SID	G1	C1	D1	G2	C2	D2		

Example:

SPC	2	32	3	-2.6	5				

Field	Contents
SID	Identification number of the single-point constraint set. (Integer > 0)
Gi	Grid or scalar point identification number. (Integer > 0)
Ci	Component number. (0 ≤ Integer ≤ 6; up to six Unique Integers, 1 through 6, may be placed in the field with no embedded blanks. 0 applies to scalar points and 1 through 6 applies to grid points.)
Di	Value of enforced motion for all degrees-of-freedom designated by Gi and Ci. (Real)

Figura 3: Estructura de la entrada SPC.

Nuestros objetivos son

- saber cuántas condiciones Dirichlet distintas definen los campos SPC de un fichero y
- asignar el número de condición a cada nodo.

Para la primera tarea, no podemos usar el campo SID ya que, desgraciadamente, Hypermesh asigna siempre (en los ejemplos realizados) el valor 1 al campo SID. En su lugar, debemos

contabilizar qué grados de libertad se restringen por nodo (pueden ser hasta 6) y qué valor se da a cada restricción. Este proceso no es trivial, ya que las restricciones para un nodo no tienen por qué ir todas en la misma entrada SPC. Por ejemplo, en una entrada SPC se pueden restringir los grados 1, 2 (y asignarles el valor 8); en la segundo entrada SPC, restringir los grados 3, 4 (con valor 10); en la tercera restringir el grado 2 (con valor 7). Lo que tendríamos aquí serían dos condiciones de contorno, la primera restringiendo los grados 1, 2, 3 y 4 (con valor 8 en los dos primeros y 10 en los otros) y la segunda restringiendo únicamente el grado 2, con valor 7.

Para poder gestionar todas las posibles variantes, Hypermesh hace ciertas hipótesis que debemos tener en cuenta:

- todos los SPC de una condición son contigüos y
- en los SPC de una condición, se fijan primero los grados de libertad menores (i.e., se fija el grado 1 antes que el 2).

De este modo, no conocemos el numero total de condiciones de contorno Dirichlet hasta que se han leído todas las entradas. Por eso, primero se llama al procedimiento `set_SPC` de `module_desplazamientos`, que guarda todas las condiciones de entradas SPC, para cada nodo y luego se llama al procedimiento `assign_SPC`, que agrupa los nodos que corresponden a la misma condición y asigna a cada nodo la condición Dirichlet **cuyo número es mayor** (es decir, si las condiciones 6 y 7 se aplican al mismo nodo, se asigna la 7).

- Las entradas FORCE se gestionan ahora con `module_fuerzas`. Esto permite distinguir las condiciones Neumann por el valor de la fuerza aplicada, y no por el SID de la entrada FORCE. La estructura de la entrada FORCE puede verse en la Figura 4.

  De modo análogo al explicado con SPC, el procedimiento `set_FORCE` de `module_fuerzas` guarda todas las condiciones y el procedimiento `assign_FORCE` asigna la condición Neumann **cuyo número es mayor, siempre que no tenga ya asignada una condición Dirichlet**.

- La sección de datos primarios solo guarda las condiciones de contorno sobre los nodos. El programa patmod se ha modificado para garantizar que, al asignar la condición de contorno a un nodo,

FORCE Static Force

Defines a static concentrated force at a grid point by specifying a vector.

Format:

1	2	3	4	5	6	7	8	9	10
FORCE	SID	G	CID	F	N1	N2	N3		

Example:

FORCE	2	5	6	2.9	0.0	1.0	0.0		

Field	Contents
SID	Load set identification number. (Integer > 0)
G	Grid point identification number. (Integer > 0)
CID	Coordinate system identification number. (Integer ≥ 0; Default = 0)
F	Scale factor. (Real)
Ni	Components of a vector measured in coordinate system defined by CID. (Real; at least one Ni ≠ 0.0. unless F is zero)

Figura 4: Estructura de la entrada FORCE.

- las Dirichlet hayan tenido prioridad sobre las Neumann, y
- dentro de cada tipo, las que cuentan con un número mayor, hayan tenido prioridad sobre las de número menor.

La asignación de números de referencia a aristas se realiza en el procedimiento `dos` y, la asignación a caras, en los procedimientos `tres` y `cuatro`. Las normas que se siguen para la asignación de referencias a aristas o caras son las siguientes:

- Si todos los vértices de la entidad tienen el mismo número, se asigna ese número.
- Si alguno es cero, se asigna cero.
- Si hay números de distinto tipo, se asigna el número **Neumman de menor valor**.
- En otro caso, se asigna el número de **menor valor**.

De este modo, garantizamos que las aristas y caras de la frontera entre dos condiciones tienen asignadas las referencias de mayor valor. Esta regla puede fallar en ciertas esquinas. En la Figura 5, se muestra en trazo grueso la frontera entre las condiciones 1 y 2. Los vértices de la frontera tienen todos el número 2 asignado, lo que provoca que la cara de la esquina

reciba equivocadamente el número 2 (en rojo). Los números azules representan los números asignados correctamente al resto de caras.

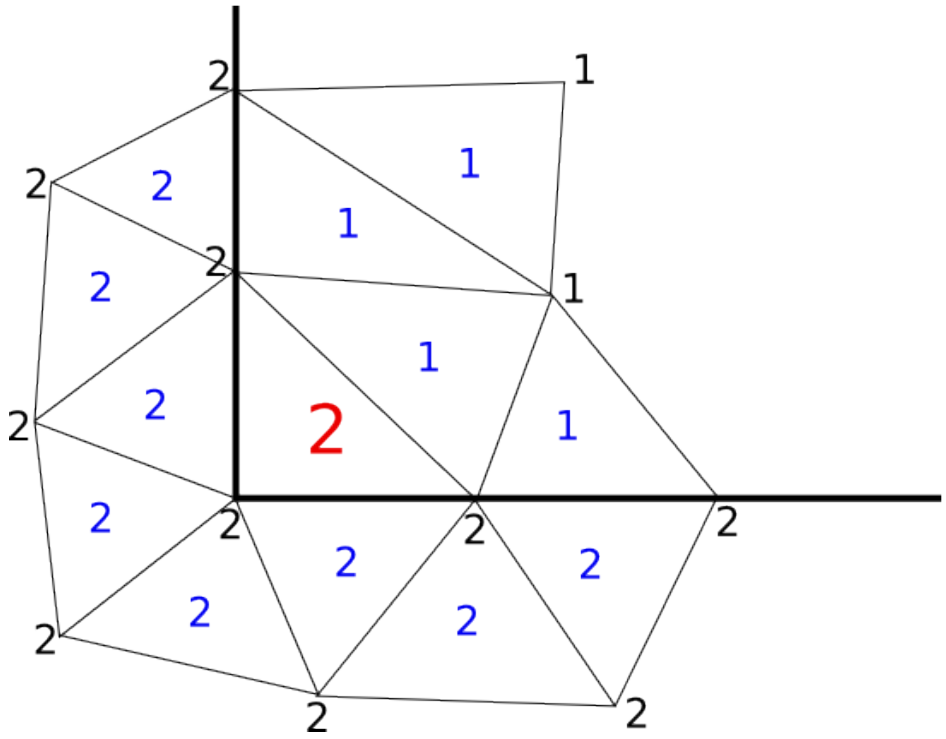


Figura 5: Ejemplo de mal funcionamiento de la asignación de referencias a caras.

Salvo casos de este tipo, creo que este algoritmo de asignación es correcto.

A.2.1. Otras modificaciones

Se resumen a continuación otras modificaciones menores al código original de Inés:

- Se han incluido módulos básicos de servicio para facilitar la programación:
  - `module_COMPILER_DEPENDANT`: agrupa los procedimientos que dependen de cada compilador. Implementado para los compiladores Intel, Gfortran y NAG.
  - `module_SO_DEPENDANT`: agrupa los procedimientos que dependen del sistema operativo. Implementado para Windows y Linux.
  - `module_REPORT`: gestiona la emisión de avisos y errores programados en el código.
  - `module_CONVERS`: agrupa los procedimientos para conversión de tipos.
  - `module_ALLOC`: agrupa los procedimientos para el alojamiento de arrays.

- `module_FILES`: agrupa los procedimientos para la gestión de ficheros.
- `module_MATH`: permite realizar operaciones matemáticas. En particular, permite calcular el determinante de una matriz cuadrada.

- Los nombres de fichero pasan de 20 a 255 caracteres.
- La apertura de ficheros es ahora más general.
- Se introduce el comando `rewind(3)` entre la lectura de coordenadas y elementos, para que ambas puedan hacerse en cualquier orden.