

**Pontificia Universidad Católica del Ecuador**

Facultad de Ingeniería

Escuela de Civil

**Análisis Estructural II**

Séptimo Nivel - Paralelo 2

**Proyecto #1**

Análisis de un Marco Plano - Método de Rigidez

**Docente:**

Ing. Wilson Torres

**Integrantes (Grupo #12):**

Caicedo Germán

Cóndor Lino

Oñate Karen

**Fecha:**

Lunes, 23 de Julio del 2018

**Contenido**

[**1.** **Introducción** 3](#_Toc519878708)

[**2. Desarrollo** 4](#_Toc519878709)

[**2.1 Entrada de Datos para la Resolución de Marco Plano** 4](#_Toc519878710)

[**2.2 Resolución Exacta del Marco Plano en Matlab** 6](#_Toc519878711)

[**2.2.1 Determinación de la longitud de los elementos** 6](#_Toc519878712)

[**2.2.2 Determinación de la matriz de rigidez local de cada elemento** 7](#_Toc519878713)

[**2.2.2.1 Funciones de altura, área e inercia correspondientes a cada elemento** 7](#_Toc519878714)

[**2.2.2.2 Determinación de los coeficientes de flexibilidad del elemento** 8](#_Toc519878715)

[**2.2.2.3 Determinación de la matriz de rigidez del elemento (Columnas 4 – 6, Filas 4 - 6)** 8](#_Toc519878716)

[**2.2.2.4 Determinación de la matriz de rigidez del elemento (Columnas 1-3, Filas 1-3)** 9](#_Toc519878717)

[**2.2.3 Grado de indeterminacion Cinemática** 9](#_Toc519878718)

[**2.2.4 Numeración de los Grados de Libertad** 10](#_Toc519878719)

[**2.2.5 Determinación de matrices de rotación de cada elemento** 11](#_Toc519878720)

[**2.2.6 Matriz de Ubicación de los Grados de Libertad de la estructura en los elementos** 11](#_Toc519878721)

[**2.2.7 Matriz de colocación de los elementos (L)** 12](#_Toc519878722)

[**2.2.8 Determinación de la matriz de rigidez global de los elementos (Ki)** 12](#_Toc519878723)

[**2.2.9 Determinación de la matriz de rigidez global de la estructura (KE)** 13](#_Toc519878724)

[**2.2.10 Determinación de las Fuerzas de Empotramiento Perfecto {FEP}** 13](#_Toc519878725)

[**2.2.12 Determinación de Desplazamientos en GDL desconocidos** 15](#_Toc519878726)

[**2.2.13 Determinación de Reacciones de la Estructura** 16](#_Toc519878727)

[**2.7 Diagramas de Corte y Momento** 16](#_Toc519878728)

[**3.** **Conclusiones** 16](#_Toc519878729)

[**4. Script en el Programa Matlab** 16](#_Toc519878730)

# **Introducción**

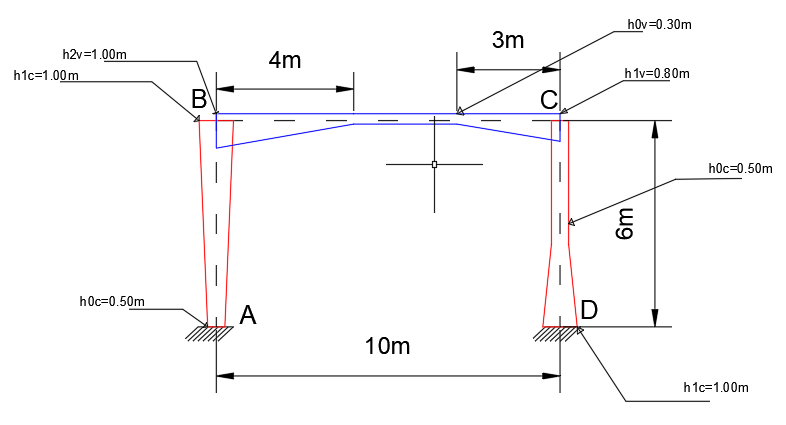
El presente trabajo trata del análisis de una estructura que posee secciones no prismáticas. Este tipo de estructuras son de uso común en las construcciones ya sea para ahorrar material o para disminuir el peso de la estructura.

Para realizar el análisis de este tipo de estructuras se parte del principio del método de flexibilidad, con el cual se obtiene la matriz de coeficientes de flexibilidad. Con dicha matriz de flexibilidad se procede a obtener las fuerzas necesarias para poder obtener desplazamientos unitarios y armar la matriz de rigidez de la estructura.

Los coeficientes de flexibilidad obtenidos dependen de las propiedades geométricas de la estructura.

**Estructura analizada**

La estructura analizada es un marco plano con secciones no prismáticas tanto en columnas como en vigas. (Figura 1)



*Figura 1. Marco plano con secciones no prismáticas*

La estructura se encuentra soportando diferentes tipos de cargas como son cargas distribuidas a lo largo de la columna AB y la viga BC. Además, de un momento puntual en el nodo B de la estructura. (Figura 2)

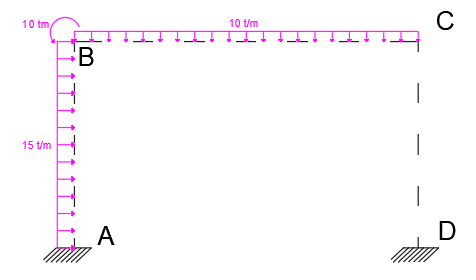


Figura 2. Cargas actuantes sobre la estructura

# **2. Desarrollo**

## **2.1 Entrada de Datos para la Resolución de Marco Plano**

Para la resolución y obtención de resultados de la estructura como se mencionó previamente en el *Inciso N.-1* se utilizará la herramienta MATLAB, para su uso es indispensable el ingreso de datos (Tablas de Ingreso) correspondientes al Marco Plano que contengan toda la información necesaria para un correcto desarrollo del programa. A continuación se detallan las tablas de datos a utilizarse.

**2.1.1 Tabla de Coordenadas (CO):** posee las coordenadas o posición (x, y) de cada uno de los nodos de la estructura medidos desde el eje de referencia establecido. En el ejercicio el origen de este sistema se encuentra en el nodo A.

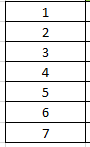
La dimensión de la tabla de coordenadas es:

**n x 2**

Donde:

**n=** número de nodos

La dimensión de la Tabla de Coordenadas, que se ingresa en nuestro proyecto para el caso del marco plano en análisis es 7 x 2:

*Tabla N.-1 Tabla de Coordenadas - Marco Plano*

En un principio con el objetivo de encontrar la solución exacta de los parámetros que el proyecto nos ha especificado encontrar, esta matriz no variará en dimensiones sino hasta que se empiece añadir tramos discretizados que correspondan a un número mayor de elementos que se encuentran inicialmente.

***Nota 1:*** Los datos ingresados, con el fin de ser resueltos por el programa, deben ser colocados en metros

**2.1.2 Tabla de Apoyos (A):** contiene únicamente los nodos que poseen algún tipo de restricción (apoyo) al desplazamiento o giro, ya sea en dirección horizontal “Rxi”, vertical “Ryi” o “Mi”. La restricción al desplazamiento o giro se denomina mediante la colocación de unos “1” en dirección de la restricción que exista.

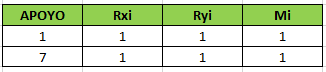
La dimensión de la tabla de apoyos es:

**nr x 4**

Donde:

**Nr =** número de nodos con alguna restricción

La dimensión de la Tabla de Apoyos que se ingresa para el marco plano en análisis es de 2 x 4, los nodos y sus restricciones son:



*Tabla N.-2 Tabla de Apoyos - Marco Plano*

Conforme la discretización de los tramos continúe los nodos restringidos mantendrán la misma numeración.

**2.1.3 Tabla de Conectividad (C):** presenta el origen y destino denominado “Nodo A” y “Nodo B” respectivamente de cada elemento de la estructura, Los nodos A y B de cada barra se denominan en función de la dirección asumida para cada una, así como se indica en la *Figura N.- 2 (FALTA)*, además el grado de discretización, la base, el peralte en el Nodo A, el peralte en el Nodo B y el módulo de elasticidad que correspondientes a cada elemento.

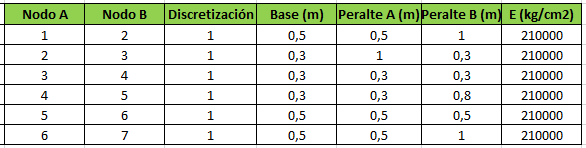
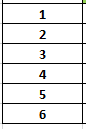
La dimensión de la tabla de conectividad es:

**m x 5**

Donde:

**m=** número de elementos

La dimensión de la Tabla de Conectividad que se ingresa para el marco plano en análisis es de 6x 7 y es:



*Tabla N.-3 Tabla de Conectividad - Marco Plano*

***Nota 2:*** Los valores del Módulo de Elasticidad y dimensiones de los elementos deberán estar en kg/cm2, m para obtener las unidades que se mencionan posteriormente al finalizar el programa, el módulo de elasticidad, en un principio, se convertirá a las unidades apropiadas T/m2.

**2.1.4 Tabla de Cargas (CA):** contiene únicamente el número de nodos cargados con su respectivo valor de carga puntual, es decir, aquellos que poseen fuerzas externas ya sean en dirección “x”, “y” o momentos.

La dimensión de la tabla de cargas es:

**CA x 4**

Donde:

**CA=** número de nodos cargados

La dimensión de la Tabla de Cargas a ingresar para la armadura en análisis es de 1x4 y es:



*Tabla N.-4 Tabla de Cargas Puntuales - Marco Plano*

***Nota 3:*** Los valores de Cargas deberán estar en las unidades Toneladas o Toneladas por metro para obtener las unidades que se mencionan posteriormente al finalizar el programa.

**2.1.5 Tabla de Cargas de vano (CARVA):** contiene los elementos con algún tipo de carga de vano distribuida aplicada en el vano, las cargas que poseen los elementos se muestran en *Tabla N.-5.*

La dimensión de la tabla de cargas es:

**mc x 2**

Donde:

**mc=** número de barras/elementos con cargas de vano

La dimensión de la Tabla de Cargas de Vano a ingresar para el marco plano en análisis es de 4x2 y es:



*Tabla N.-5 Tabla de Cargas de Vano - Marco Plano*

***Nota 4:*** Los valores de Cargas de vano deberán estar en Toneladas/metro para obtener las unidades que se mencionan posteriormente.

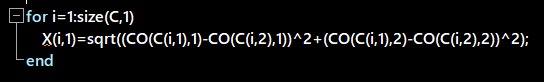
## **2.2 Resolución Exacta del Marco Plano en Matlab**

# **2.2.1 Determinación de la longitud de los elementos**

Para determinar las longitudes de cada elemento se establece un contador (i) desde 1 hasta el número de barras (6), de manera que extraemos la posición de los extremos de cada elemento, ubicando la posición de dichos nodos correspondientes a cada elemento en la columna 1 y 2 de la tabla “C” en la tabla “CO” de modo que queden guardados en un vector “X”.

La longitud se calcula a partir de la ecuación de distancia entre dos puntos en un plano cartesiano, expresada:

Se debe tomar en cuenta, para cálculos posteriores los valores de las longitudes así como también las coordenadas han sido ingresados en **metros.**

****

*Imagen N.- 2*

## **2.2.2 Determinación de la matriz de rigidez local de cada elemento**

### **2.2.2.1 Funciones de altura, área e inercia correspondientes a cada elemento**

Dado que la altura en las cartelas a analizarse varía de manera lineal, la función “hx” que expresa la ley de altura, será calculada con la definición de la función de dicho orden y=mx + b:

Donde las alturas están alojadas en la tabla “CO” y “d”, que es la longitud del elemento, en el vector “X” definido anteriormente, estas funciones se guardan en un vector simbólico “hx”

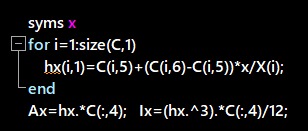
De la misma manera establecemos una ley de alturas y de inercias para cada elemento en vectores simbólicos “Ax” e “Ix”, partiendo de la definición:

* Área de un rectángulo:

Donde b, la base, la encontramos en la matriz “CO” y variará linealmente al ser multiplicada por la función de altura “” definidos en “hx”, este vector, igualmente simbólico, lo nombraremos como “Ax”.

* Inercia centroidal en el eje fuerte de un rectángulo:

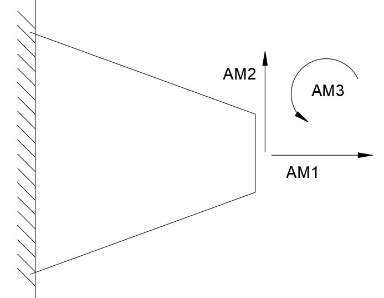
De la misma manera que con la altura, basados en la definición, guardaremos la función de inercia correspondiente a cada elemento en un vector “Ix”.



*Imagen N.- 3*

## **2.2.2.2 Determinación de los coeficientes de flexibilidad del elemento**

Los coeficientes de flexibilidad obtenidos a partir del principio del trabajo virtual, los obtenemos aplicando fuerzas unitarias en el extremo de la barra B, una fuerza axial, corte y momento como se indica en la *Figura N.- 3.*

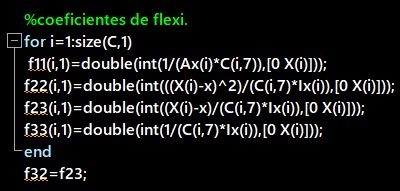


*Figura N.- 3 Obtención de los Coeficientes de Flexibilidad*

De esta manera obteniendo:

= =

El cálculo de los coeficientes *f11, f22, f23, f32 y f33* quedarán guardados en vectores con el mismo nombre de tamaño correspondiente al número de elementos, con su distancia función de áreas e inercias respectivas definidas en puntos anteriores y su módulo de elasticidad en la columna de conectividad.

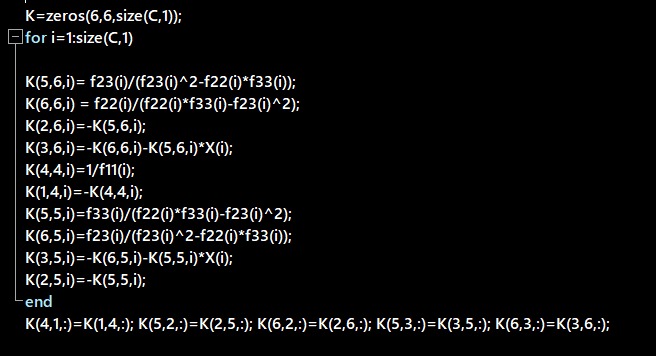


*Imagen N.- 4*

## **2.2.2.3 Determinación de la matriz de rigidez del elemento (Columnas 4 – 6, Filas 4 - 6)**

Estableceremos los coeficientes de rigidez que corresponden a desplazamientos unitarios en el Nodo B partiendo de:

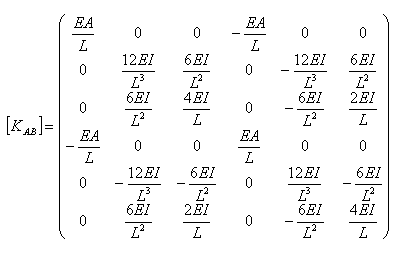
Obtendremos las ecuaciones correspondientes al equilibrio con desplazamientos igual a 1 en para obtener los coeficientes de rigidez despejando .

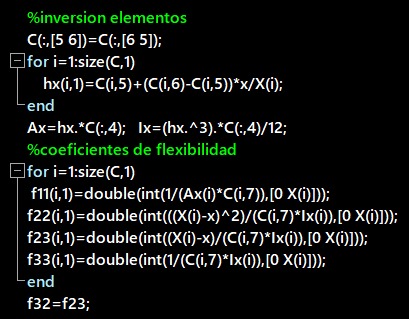
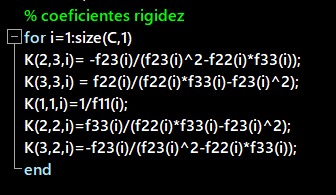


Conociendo que las matrices de rigidez de los elementos son simétricas con respecto a su diagonal, también podemos definir los coeficientes en las filas 4-6.

## **2.2.2.4 Determinación de la matriz de rigidez del elemento (Columnas 1-3, Filas 1-3)**

Con la permutación de las columnas 5 y 6 de la tabla “C”, realizando los mismos procedimientos de los puntos 2.2.2.1 hasta 2.2.2.3 podemos calcular los coeficientes pertenecientes al bloque definido entre las columnas 1-3 y filas 1-3 conociendo que los coeficientes de rigidez en dicho bloque corresponden al cálculo anterior de los coeficientes en las columnas 4-6 y filas 4-6 correspondientemente (Ej: k44 --> k11) tomando en cuenta que los coeficientes que no se encuentran en la diagonal son de signo contrario respecto a los calculados anteriormente como se puede ver en la matriz de rigidez local de un elemento de marco plano.



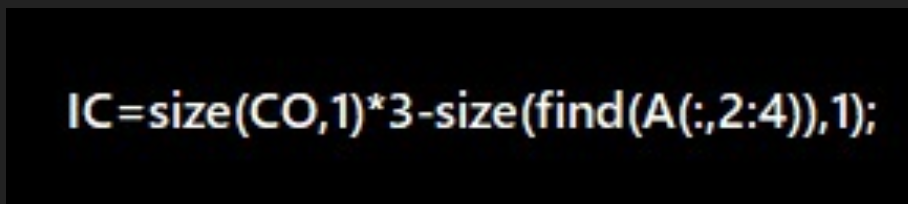
 

# **2.2.3 Grado de indeterminacion Cinemática**

El Grado de Indeterminación Cinemática está definido como los desplazamientos y giros posibles que los nodos de la estructura pueden experimentar, para el caso en estudio, las componentes de los desplazamientos se encuentran en eje horizontal y vertical más un giro.

Para el caso de los marcos planos, este valor lo podemos expresar como la suma de los tres posibles movimientos por nodo menos las restricciones que pueden impedir dichos efectos.

Dichas restricciones pueden ser contadas en función de los muchos “1” en la matriz “A” sin contar la primera columna que indica la numeración de los nodos restringidos.



### **2.2.4 Numeración de los Grados de Libertad**

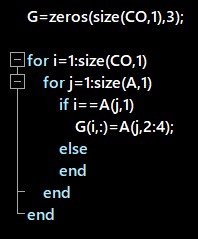
Para una resolución adecuada del marco plano se requiere de la enumeración de los grados de libertad de la estructura siguiendo un orden, es decir, primero se numeran los grados de libertad desconocidos, y posteriormente los grados de libertad conocidos. Esta numeración se basa en la información obtenida de la tabla de apoyos ya que en ella se indica los nodos se encuentran cual alguna restricción.

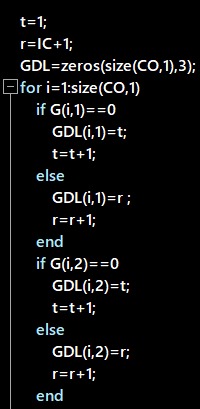
* **Matriz “G”**

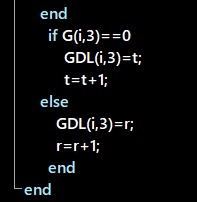
Para realizar este procedimiento, primero se forma una matriz de ceros denominada “G” de dimensiones ***(número de nodos x 3).*** De manera similar que la matriz que contiene las restricciones que puede contener cada nodo pero de numero de filas igual al número total de nodos, tengan o no restricciones.

* **Matriz “GDL”**

Partiendo de un contador t = 1 y r = IC+1, y una matriz de ceros “GDL” con las mismas dimensiones de “G” utilizaremos un bucle que revisará todos los elementos de una fila en busca de posible restricciones en “G” (1), en caso de encontrarla, en dicha posición en la matriz “GDL” pondremos el valor actual de t, caso contrario pondremos el de r, cada vez que esta condición sea verdadera y se haya colocado el contador respectivo el “GDL”, los contadores aumentarán en uno (+1).





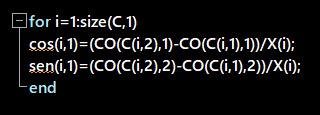


### **2.2.5 Determinación de matrices de rotación de cada elemento**

* **Determinación del seno y coseno**

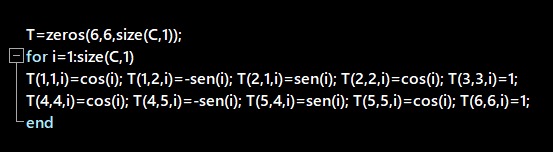
De la misma manera de la que extrajimos las coordenadas de los extremos de los elementos, procederemos a encontrar los valores del seno y coseno del ángulo de rotación de los elementos respeto a un ángulo cero, guardando esta información en vectores con el mismo nombre.

Siendo “d” la longitud de los elementos encontrada en el vector “X”.



* **Armado de la matriz de rotación**

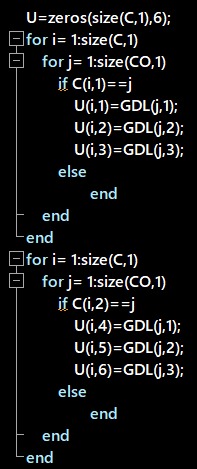
Se forma un arreglo tridimensional de manera que los resultados a obtener sean guardados en una sola variable donde colocaremos las matrices de rotación de cada elemento, cada matriz de rotación será colocada en un puesto de la tercera dimensión de esta variable. Las matrices de rotación correspondientes a cada elemento de marco plano son:



### **2.2.6 Matriz de Ubicación de los Grados de Libertad de la estructura en los elementos**

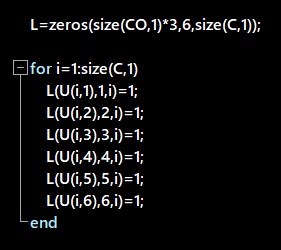
Se ha definido la matriz “U” de ceros de dimensiones ***(número de elementos x 6)*** en el cual se relacionarán los tres grados de libertad en el Nodo A y los tres grados de libertad en el Nodo B de cada elemento en este orden, para facilitar la creación de la matriz de colocación.

Para esto partiremos de que ya establecimos una numeración de los nodos de cada elemento en la matriz “C” y en base a esto ubicaremos en la fila respectiva para cada elemento sus grados de libertad en ambos extremos.



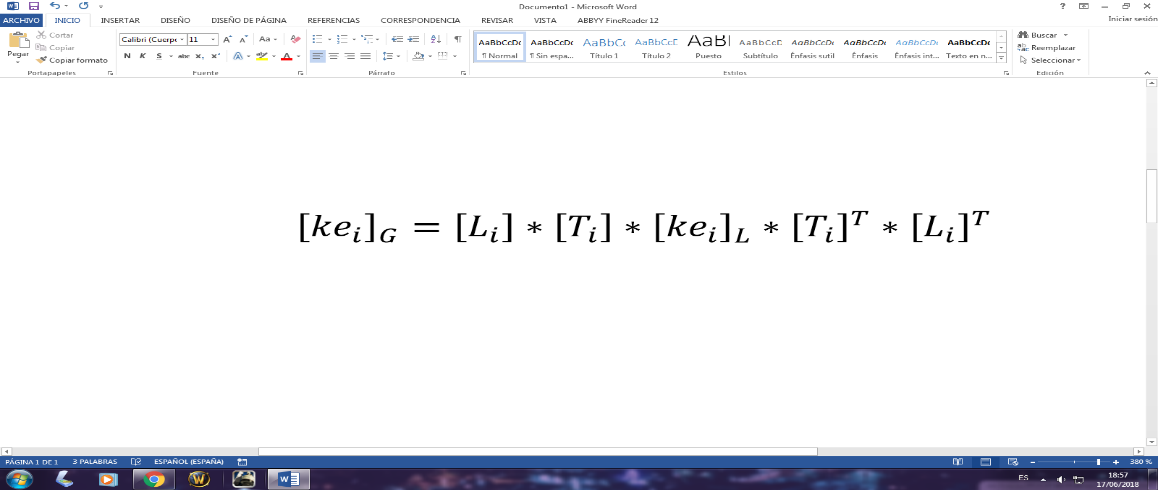
### **2.2.7 Matriz de colocación de los elementos (L)**

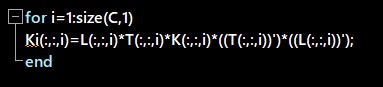
De la misma manera que creamos un arreglo tridimensional parea la matriz de rotación, partiremos de la matriz de dimensiones ***(GDL x 3 x número de elementos),*** para que, dependiendo de la numeración de los GDL en la matriz “U”, colocaremos un “1” en dicha fila dependiendo de la columna en la que se encuentre este GDL. Este proceso se lo realizará para los 6 GDL que se relacionan a un elemento. La matriz de colocación de cada elemento se guardará en diferentes niveles de la tercera dimensión.



### **2.2.8 Determinación de la matriz de rigidez global de los elementos (Ki)**

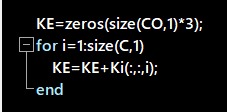
Calcularemos en una variable tridimensional, la matriz de rigidez global de cada elemento definida por el producto de las matrices previamente establecidas respectivas a cada nivel en la tercera dimensión de los arreglos [L], [T] y [K] pertenecientes a cada elemento.





### **2.2.9 Determinación de la matriz de rigidez global de la estructura (KE)**

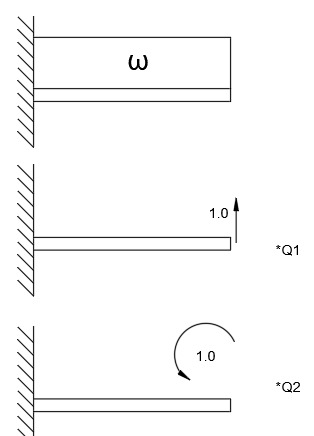
Se crean una matriz cuadrada de ceros de dimensión “GDL” y se procede a realizar sumas sucesivas de cada nivel de “Ki” hasta que contenga los aportes de rigidez de cada elemento en los GDL de la estructura globalmente.



### **2.2.10 Determinación de las Fuerzas de Empotramiento Perfecto {FEP}**

Utilizando el principio del trabajo virtual, estableceremos ecuaciones de compatibilidad, correspondientes a cargas unitarias en el Nodo B

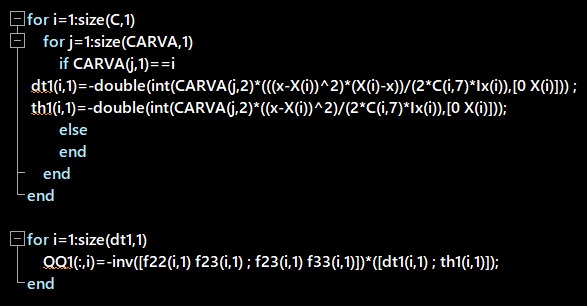
Y que: que de la misma manera dado los caso parciales será igual a y . Siendo los casos parciales:

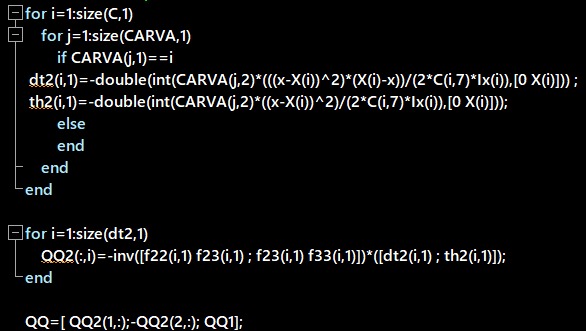


Las cargas distribuidas en cada tramo serán un dato que se extraerá de la matriz “CARVA”.

Guardaremos dichos movimientos en vectores de “dt1” y “th1”, para encontrar las redundantes Q1 y Q2 partiremos de las ecuaciones definidas al principio de este inciso. De manera matricial:

Con dichas redundantes, aplicadas en el nodo B, en un vector fila “QQ1”. El mismo procedimiento lo realizaremos antes y despues de realizada la permutación de las alturas, con la diferencia de que los movimientos quedarán guardados en vectores “dt2” y “th2” y las redundantes en el vector fila “QQ2”.





Para facilidad en el armado de los vectores que contendrán las fuerzas de empotramiento locales de los elementos guardaremos las matrices QQ1 y QQ2 en una sola variable QQ. Se debe tomar en cuenta que la redundante en A de corte tendrá signo contrario realizando este proceso de la manera en la que se ha explicado.

* **Armado de las Fuerzas de Empotramiento Locales**

Considerando que las filas 1-4 de la matriz “QQ” contienen las acciones de corte y momento en cada columna perteneciendo a un elemento armaremos el vector de fuerzas de empotramiento locales a partir de esta información, no habrá componentes axiales debido a que las cargas son distribuidas y aplicadas perpendicularmente al eje del elemento. Esta información se guarda en una matriz en donde cada columna contiene las fuerzas de empotramiento perfecto, se considera que este proceso se realizará hasta el último tramo que contenga una carga por lo que tramos intermedios contendrán ceros en dicha columna lo cual no afectará el aporte de las fuerzas de empotramiento de la estructura que serán adicionadas.

D:\FREDDY\Downloads\WhatsApp Image 2018-07-20 at 6.18.38 PM.jpeg

**2.2.11 Determinación de las fuerzas de empotramiento perfecto globales**

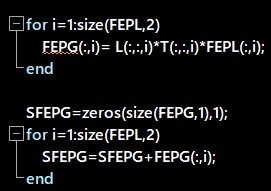
* **De los elementos**

Utilizaremos el producto matricial:

Para encontrar las fuerzas producto de cargas de vano en los Grados de Libertad de la estructura que adicionan un efecto en las acciones puntuales aplicadas [Q].

* **De la estructura**

En un vector “SFEPG” con las mismas dimensiones de los vectores de fuerzas de empotramiento perfecto (GDL \* 1), que en un principio será un vector de ceros, añadiremos los efectos de las cargas de vano pertenecientes a cada elemento cargado de igual manera que hicimos durante la conformación de la matriz de rigidez de la estructura.



### **2.2.12 Determinación de Desplazamientos en GDL desconocidos**

* **Cálculo de [KQQ]**

Para la determinación de estos desplazamientos, en un principio, definiremos el bloque [KQQ] de la matriz de rigidez de la estructura, en este caso [KE] con elementos pertenecientes a las filas y columnas 1-IC.

* **Matriz de ubicación de cargas “p”**

Nos apoyaremos en una matriz “p”, en un principio de ceros, similar a como hicimos con la matriz “G” pero en este caso con un numero de filas igual al número de nodos y de columnas igual a 3 (posibles acciones en los nodos, fuerzas en “x”, “y” y momento) partiendo de que dichas cargas puntuales se encuentran en la matriz “CA” siendo este arreglo similar a dicha matriz pero conteniendo todos los nodos, tengan o no cargas. Una acción no existente en estos nodos estará representada por un cero.

* **Vector de ubicación de las acciones en “p” (V)**

En el vector V estableceremos la ubicación en filas y en columnas de las cargas aplicadas en cada nodo utilizando la función “find” con dos parámetros (ubicación de la fila y columna de las cargas en “CC”).

* **Cálculo de [P]**

Dado que “p” y “GDL” van a tener dimensiones similares y cada una va a relacionar las cargas y la numeración de los Grados de Libertad respectivamente crearemos un vector [P] de ceros al cual dependiendo de la numeración en “GDL”, colocaremos en dicha fila, la información de la matriz “p” dependiendo del número de fila y columna definido en “V” para cada carga aplicada.

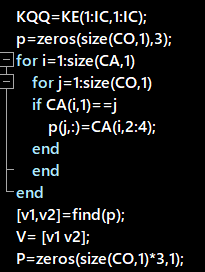
* **Cálculo de [Q]**

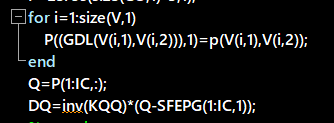
Las filas desde la 1-IC contendrán las cargas en los Grados de Libertad desconocidos mejor conocido como [Q].

* **Cálculo de [ΔQ] (DQ)**

Por definición los desplazamientos en los GDL desconocidos ΔQ vienen dados por la ecuación:

Donde serán las filas 1-IC del vector definido como SFEPG





## **2.2.13 Determinación de Reacciones de la Estructura**

* **Cálculo del bloque [KRQ]**

Este bloque, que relaciona los desplazamientos en un principio desconocidos {ΔQ} con las reacciones es un componente de la matriz de rigidez de la estructura [KE] con filas desde IC + 1 hasta la última fila de [KE] y de columnas de 1-IC.

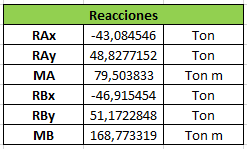
* **Cálculo de reacciones [R]**

La ecuación que define a las reacciones en el método de la rigidez es:

Donde serán las filas IC+1 hasta el último elemento del vector definido como SFEPG.

**E:\loli18.PNG**

Los valores de reacciones obtenidas se guardan en un vector:

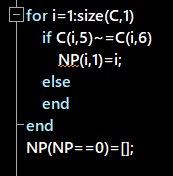


*Tabla N.-8 Reacciones – Marco Plano*

# **2.3 Resolución Discretizada del Marco Plano en Matlab**

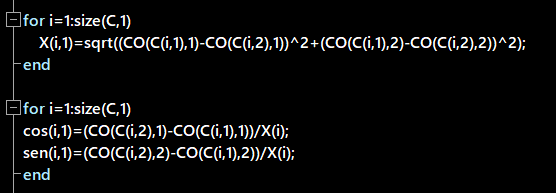
## **2.3.1 Definición de tramos no prismáticos**

Estableceremos un vector [NP] en el cual se evaluará si las alturas en el Nodo A y en el Nodo B en la tabla “C” coinciden y dependiendo de que, si esta condición es verdadera, la numeración del tramo se guardará en este vector. En este caso dichos tramos son 1, 2, 4,6.



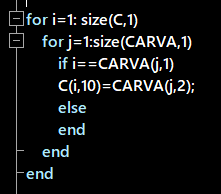
## **2.3.2 Determinación de la longitud, coseno y seno del ángulo de todos los tramos**

Para este punto se procederá de igual manera a la aproximación tomada en el punto 2.2.1 y 2.2.5 pertenecientes a la resolución exacta.



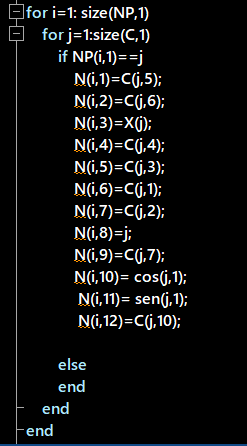
## **2.3.3 Relación de cargas de vano en tramos originales**

Con el objetivo de que posteriormente cada tramo discretizado contendrá las cargas distribuidas pertenecientes a su tramo original, en caso de necesitarlo utilizaremos la matriz de conectividad para relacionar cada tramo de acuerdo a su numeración (fila en matriz “C”) con su carga distribuida, en caso de tenerla (dato encontrado en “CARVA”). Para este propósito dependiendo de si la numeración del tramo se encuentra en “CARVA” colocaremos su carga respectiva en la columna 10 de la tabla “C”.



## **2.3.4 Propiedades de los tramos no prismáticos “N”**

Con el objetivo de discretizar los tramos no prismáticos guardaremos información de dichos elementos partiendo de que si su numeración se encuentra en el vector [NP], dicha información será guardada. La composición de esta matriz será “Ha”, “Hb”, “Longitud”, “Base”, “Nivel de discretización”, numeraciones: “Na” y “Nb” de los nodos de cada elemento, “numeración del tramo”, “módulo de elasticidad”, “coseno” y “seno” del ángulo de la dirección que toma el elemento y “las cargas de vano distribuidas relacionadas” en las columnas 1-12.

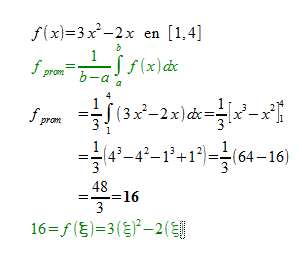


## **2.3.5 Funciones de altura, área e inercia**

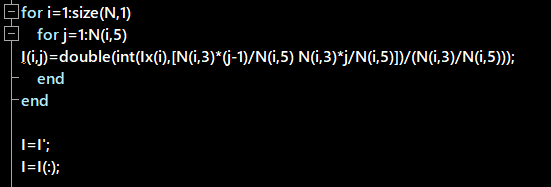
Se procederá de acuerdo a lo ya explicado en el punto 2.2.2.1

## **2.3.6 Valores medios de inercia y área**

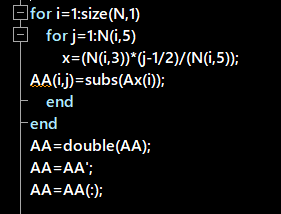
* **Inercia:** partiremos de que debido a una variación cúbica de la función de la inercia**,** el valor representativode la función definida en dos intervalos queda mejor expresada como:



Valores de inercia media, los cuales en un principio quedarán guardados en una matriz “I” donde cada fila contiene los dichos valores correspondientes a un solo elemento, en nuestro caso 4 filas. Este arreglo se redimensionará a manera de un vector para que este pueda ser integrado a la matriz “C” que posteriormente tendrá que ser adaptada para poder resolver la cartela como si fuera un marco formado de elementos prismáticos.



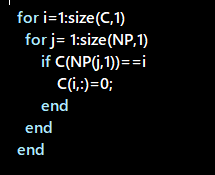
* **Área:** de igual manera que con la inerciapero tomando en cuenta que esta vez el área varía de manera lineal en los tramos no prismáticos se tomará como valor medio de área, el área en la sección media de cada tramoya discretizado. Esta información se guardará con el mismo criterio que en la matriz “I” pero en una matriz “AA” que también será redimensionada en un vector que se agregara a la matriz de conectividad.



**2.3.7 Armado de la nueva matriz de conectividad**

* **Borrado de la información de miembros no prismáticos en la tabla de conectividad “C”**

Para realizar este paso, se procedió a reemplazar todos los valores en las filas de los tramos no primaticos con ceros, el criterio con el que el programa realiza esta acción es revisando si la numeración de los tramos (fila) en los tramos de conectividad corresponde a un valor en la primera columna de la tabla “NP”



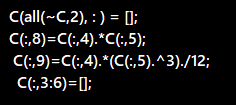
Hecho esto, revisaremos que filas tienen como 0 a todos sus elementos, en caso de que esta evaluación sea verdadera, se borrara toda esta fila

* Calculo del área e inercia en los tramos prismáticos

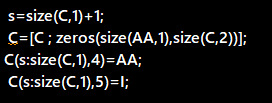
Dado que, en estos tramos, la altura no corresponde a una función de x, se procederá a calcular estos parámetros a partir de la base y la altura de estos tramos, las cuales se encuentran en las columnas 4 y 5 de la tabla de conectividad “C”. el área y la inercia calculada para estos tramos se guardará en las columnas agregadas 7 y 8 en la misma matriz de conectividad

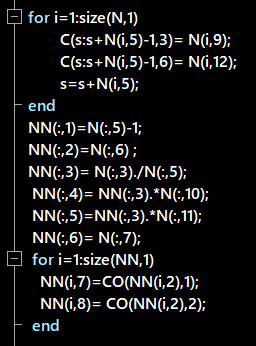
* Paso de la matriz de conectividad inicial a la matriz de conectividad para marco planos (Programa de análisis matricial)

Dado que las relaciones de nodo A y B, elasticidad, área e inercia, se encuentran en la tabla de conectividad en sus columnas 1,2,7,8 y 9 respectivamente, y la información del área e inercia media de los tramos no prismáticos ya ha sido definida en pasos anteriores, se eliminará las columnas 3-6 de la tabla de conectividad “C” que posee el grado de discretizacion y las dimensiones de los tramos antes de discretizarlos



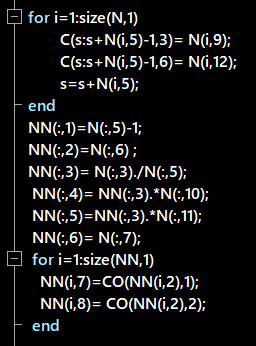
Hecho esto, se redimensionará la matriz al añadir los vectores de área e inercia medios encontrados anteriormente, colocándolos en las columnas 4 y 5, bajo las áreas e inercias de los tramos no prismáticos. De la misma manera, con la ayuda de un contador, el módulo de elasticidad y las cargas de vano relacionadas a los tramos originales, guardado respectivamente en las columnas 9 y 12 de la matriz de propiedades de tramos no prismáticos “N”, se guardarán en las columnas 3 y 6 de la tabla de conectividad “C”





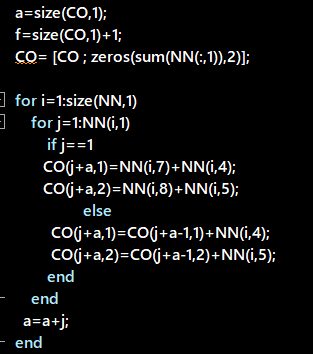
**2.3.8 Armado de la nueva matriz de coordenadas**

Con este fin, se armará una nueva matriz “NN” que constará de: el número de nodos a agregar, el nodo A y B del tramo original de los tramos discretizados, la longitud de cada tramo discretizado y el desplazamiento en “x” y “y “, además de las coordenadas en “x” y “y “ del nodo A de cada tramo a discretizar, que realizará cada nuevo nodo a partir de su nodo inicial, propiedades calculadas a partir de información sustraída de la matriz “N”



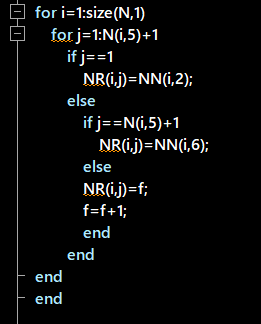
* Coordenadas de los nuevos nodos

Partiremos de un contador “a” igual al número de nodos actual (número de filas) de la tabla de coordenadas” CO” con el fin de ubicar nuevos nodos, de manera que, dependiendo del grado de discretizacion que se le haya dado a cada uno de los elementos, este valor menos uno será el número de nodos a agregar en cada tramo. Para esto, se utilizarán las coordenadas iniciales de un nodo y se irán añadiendo los desplazamientos en “x” y “y” conforme avance este proceso. Se establecerá una condición para que la primera coordenada parta de un movimiento del nodo inicial, mientras que las demás partirán de las coordenadas ya establecidas en este punto.



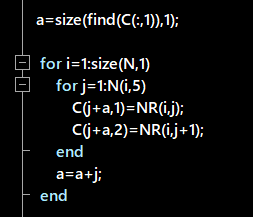
**2.3.7 Relación de los nuevos nodos con los tramos discretizados**

Con la ayuda de un contador “f”, igual al número de filas de la tabla de coordenadas actual más uno, establecido en el punto inferior, armaremos una matriz “NR”, que contendrá, dependiendo del grado de discretización del elemento original, en su fila respectiva, los nodos de la tabla de coordenadas en este instante, que pasan por el eje del elemento original, de este modo, si el elemento no prismático “1” partía del nodo “1” hacia el “2” y su grado de discretización “3”, provocaba que se añadan los nodos “8” y “9”, la fila “1” perteneciente a este elemento iba a ser [1 8 9 2]



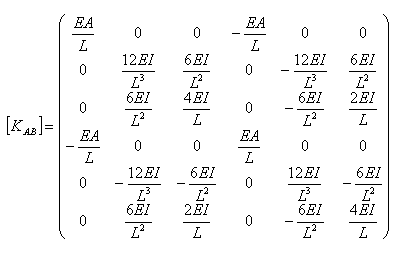
* Ubicación de los nodos en la tabla de conectividad

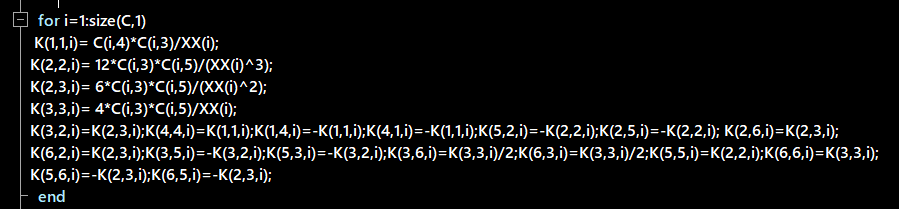
Con la ayuda de los nodos establecidos en las filas de ”NR”, se ubicará en la matriz de conectividad “C” su numeración respectiva, de manera que coincidan con las propiedades de área, inercia y elasticidad establecidas anteriormente, dado que el criterio con el que se armaron aquellas filas corresponden a los tramos entre los nodos armados en las columnas “1” y “2” de la matriz “C”



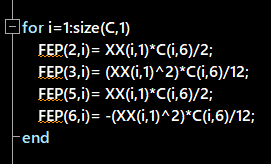
**2.3.7 Resolución del marco plano discretizado**

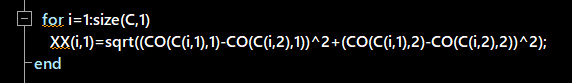
La resolución del marco plano, será correspondiente a lo explicado en los puntos 2.2.2.4 hasta el punto 2.2.2.13, con la excepción de que no será necesario el calcular los coeficientes de flexibilidad con el propósito de encontrar los coeficientes de rigidez, sino que estos vendrán definidos de la forma:

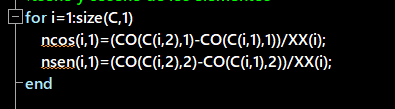




Además, la longitud y el seno y coseno de la dirección de estos tramos se guardarán en vectores “XX”, nsen y ncos con el fin de evitar definir o borrar estas variables, también es necesario mencionar como los vectores de fuerzas de empotramiento locales de cada elemento vienen definidos de la forma:



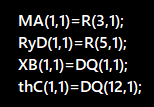




# **2.4 Variables a guardarse en cada iteración**

* Respuesta exacta

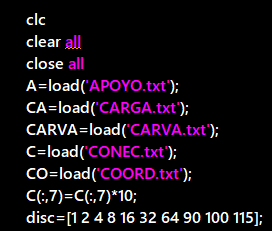
Dada la resolución exacta del marco plano, encontrada en la primera parte del programa, una vez haya definido los desplazamientos en los GDL desconocidos (DQ en el programa) y las reacciones “R”, guardaremos el primer y doceavo elemento de “DQ” y el tercer y quinto elemento de “R” en el primer elemento de matrices pertenecientes al momento de Reacción en “A”, la reacción vertical en “D”, el desplazamiento horizontal en X y el giro en C en los vectores mostrados, es posible que, conforme avance la discretización, los desplazamientos en DQ se calculen en los mismos GDL debido a que la numeración de los nodos encontrados inicialmente (7), es independiente del nivel de discretización, con el fin de no interferir en la resolución discretizada, la cual tiene nombres similares, se borraran todas las variables menos estas 4 definidas :

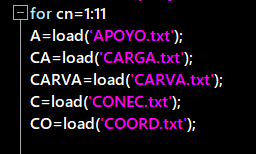


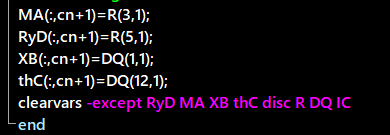


* **Evolución de los parámetros conforme la discretización**

Con el fin de graficar, lo mucho que se acercarán los parámetros mencionados en el punto anterior conforme avanza el nivel de discretización y por lo tanto de exactitud, a su solución exacta, toda la resolución del marco plano de manera discretizada, se encontrara en un bucle, de manera que se establecerán niveles de discretización en el vector fila “disc” en un principio con los valores 1,2,4,8,16,32,64,90,100,115 los cuales, conforme avance el contador del bucle “cn” serán valores que reemplazaran a la tercera columna de la matriz de conectividad “C”, la cual contiene el nivel de discretización de los elemento aplicados en esta parte del programa, cabe mencionar que el programa no tiene problema con colocar un nivel de discretización en tramos prismáticos, debido a que no tiene ningún efecto en la resolución discretizada



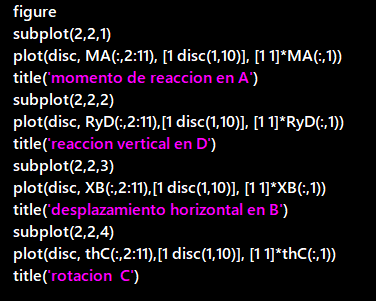




El contador de este bucle, ira de 1 hasta 11 de modo que guarde las respuestas de cada nivel de discretización, aplicado en todos los elementos de la misma manera, en los elementos 2-11 de los vectores de respuestas mencionados en este y el punto anterior, mientras el contador “cn” avance desde “1” hasta “10” ,se borrara la variables antes de reiniciar el bucle con el fin de no interferir con las nuevas variables, en caso de que la tabla de entrada “CONEC.txt”, en cuanto a los niveles de discretización, se modifique de alguna manera, las respuestas para dicho caso se encontraran en el elemento 12 de estos vectores de respuesta

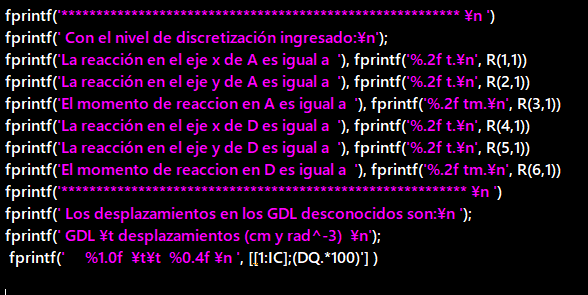
# **2.5 Gráfica de la evolución de los parámetros**

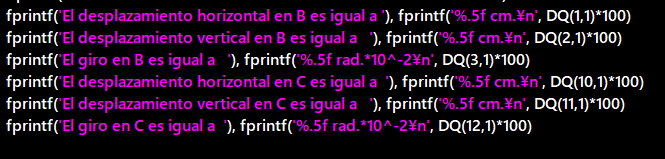
A partir de las respuestas en MA, RyD, XB y thC utilizaremos la función ‘subplot’ con el fin de cumplir este punto, manteniendo el primer elemento (respuesta exacta), como una función constante, y las respuestas, productos de las discretización en el vector “disc” como puntos que tenderán a acercarse más a esta respuesta exacta (asíntota)



# **Despliegue amigable de los resultados**

Debido a que es necesario el mostrar, en la interfaz del usuario, los resultados de la discretización ingresada; las reacciones y los desplazamientos y giros (numero dependiente del grado de discretización presente) de los GDL desconocidos, se establece una rutina para presentar estos resultados en la command window:





Se debe tener en cuenta que, como ya ha sido explicado, la numeración de los nodos y por lo tanto también la de los grados de libertad se mantendrá, por lo que los desplazamientos horizontal, vertical y giro del nodo B y C siempre se encontraran en los elementos 1,2,3 y 10,11,12 respectivamente, en caso de que sea necesario analizar estos nodos. La condición para que esta numeración se mantenga, es que solo el nodo A y D se encuentren empotrados

# **Diagramas de Corte y Momento**

Desplazamientos exactos, guardados en “DQe”

|  |
| --- |
| 3,7121 |
| -0,0387 |
| -0,8722 |
| 3,6609 |
| -4,7293 |
| -0,9932 |
| 3,5864 |
| -2,7431 |
| 1,6642 |
| 3,5426 |
| -0,0513 |
| 0,1595 |
| 0,7739 |
| -0,0162 |
| -0,7711 |

Reacciones exactas, guardadas en “Re”

|  |
| --- |
| -43,08 |
| 48,83 |
| 79,50 |
| -46,92 |
| 51,17 |
| 168,77 |

# **Conclusiones**

* Se recomienda, no usar una discretización mayor a 90 en todos los elementos para no tener problemas con tiempos de espera muy largos. Con relación a problemas por la falta de memoria, no es posible realizar una discretización mayor a 115 en los elementos en la versión estudiantil de MATLAB con 8 GB de RAM, debido a que el programa limita el uso de los muchos datos que se manejan simultáneamente, principalmente refiriéndonos a la matriz de rigidez global de los elementos, que, en el caso presente, es un arreglo tridimensional con matrices en cada nivel producto matricial de otras 5 matrices.
* La aproximación por medio de elementos finitos, introducida mediante este programa, sea útil para definir matrices de rigidez locales de elementos que tienen ejes curvos o de formas muy complejas en varios elementos más manejables de modo que se pueda realizar una aproximación precisa, que de otra manera no sería posible
* Deben aplicarse varias condiciones en las rutinas durante la creación de cualquier programa, de modo que se eviten errores debido a variaciones en los imput imprevistas
* Ciertas soluciones como los algoritmos utilizados con las “matrices dispersas” o con grandes cantidades de ceros, pueden ser aplicadas en casos como el presente, donde existe problemas de memoria debido al manejo de esta información, sin embargo, estos métodos de programación están fuera de nuestro alcance
* En lo posible, los resultados mostrados en la interfaz del usuario deben ser lo más concisos, para evitar que se muestre información no requerida, pero si inherente a la resolución.

# **6. Script en el Programa Matlab**

**clc**

**clear all**

**close all**

**A=load('APOYO.txt');**

**CA=load('CARGA.txt');**

**CARVA=load('CARVA.txt');**

**C=load('CONEC.txt');**

**CO=load('COORD.txt');**

**C(:,7)=C(:,7)\*10;**

**disc=[1 2 4 8 16 32 64 90 100 115];**

**for i=1:size(C,1)**

**X(i,1)=sqrt((CO(C(i,1),1)-CO(C(i,2),1))^2+(CO(C(i,1),2)-CO(C(i,2),2))^2);**

**end**

**syms x**

**for i=1:size(C,1)**

**hx(i,1)=C(i,5)+(C(i,6)-C(i,5))\*x/X(i);**

**end**

**Ax=hx.\*C(:,4); Ix=(hx.^3).\*C(:,4)/12;**

**for i=1:size(C,1)**

**f11(i,1)=double(int(1/(Ax(i)\*C(i,7)),[0 X(i)]));**

**f22(i,1)=double(int(((X(i)-x)^2)/(C(i,7)\*Ix(i)),[0 X(i)]));**

**f23(i,1)=double(int((X(i)-x)/(C(i,7)\*Ix(i)),[0 X(i)]));**

**f33(i,1)=double(int(1/(C(i,7)\*Ix(i)),[0 X(i)]));**

**end**

**f32=f23;**

**K=zeros(6,6,size(C,1));**

**for i=1:size(C,1)**

**K(5,6,i)= f23(i)/(f23(i)^2-f22(i)\*f33(i));**

**K(6,6,i) = f22(i)/(f22(i)\*f33(i)-f23(i)^2);**

**K(2,6,i)=-K(5,6,i);**

**K(3,6,i)=-K(6,6,i)-K(5,6,i)\*X(i);**

**K(4,4,i)=1/f11(i);**

**K(1,4,i)=-K(4,4,i);**

**K(5,5,i)=f33(i)/(f22(i)\*f33(i)-f23(i)^2);**

**K(6,5,i)=f23(i)/(f23(i)^2-f22(i)\*f33(i));**

**K(3,5,i)=-K(6,5,i)-K(5,5,i)\*X(i);**

**K(2,5,i)=-K(5,5,i);**

**end**

**K(4,1,:)=K(1,4,:); K(5,2,:)=K(2,5,:); K(6,2,:)=K(2,6,:); K(5,3,:)=K(3,5,:); K(6,3,:)=K(3,6,:);**

**for i=1:size(C,1)**

**for j=1:size(CARVA,1)**

**if CARVA(j,1)==i**

**dt1(i,1)=-double(int(CARVA(j,2)\*(((x-X(i))^2)\*(X(i)-x))/(2\*C(i,7)\*Ix(i)),[0 X(i)])) ;**

**th1(i,1)=-double(int(CARVA(j,2)\*((x-X(i))^2)/(2\*C(i,7)\*Ix(i)),[0 X(i)]));**

**else**

**end**

**end**

**end**

**for i=1:size(dt1,1)**

**QQ1(:,i)=-inv([f22(i,1) f23(i,1) ; f23(i,1) f33(i,1)])\*([dt1(i,1) ; th1(i,1)]);**

**end**

**C(:,[5 6])=C(:,[6 5]);**

**for i=1:size(C,1)**

**hx(i,1)=C(i,5)+(C(i,6)-C(i,5))\*x/X(i);**

**end**

**Ax=hx.\*C(:,4); Ix=(hx.^3).\*C(:,4)/12;**

**for i=1:size(C,1)**

**f11(i,1)=double(int(1/(Ax(i)\*C(i,7)),[0 X(i)]));**

**f22(i,1)=double(int(((X(i)-x)^2)/(C(i,7)\*Ix(i)),[0 X(i)]));**

**f23(i,1)=double(int((X(i)-x)/(C(i,7)\*Ix(i)),[0 X(i)]));**

**f33(i,1)=double(int(1/(C(i,7)\*Ix(i)),[0 X(i)]));**

**end**

**f32=f23;**

**for i=1:size(C,1)**

**K(2,3,i)= -f23(i)/(f23(i)^2-f22(i)\*f33(i));**

**K(3,3,i) = f22(i)/(f22(i)\*f33(i)-f23(i)^2);**

**K(1,1,i)=1/f11(i);**

**K(2,2,i)=f33(i)/(f22(i)\*f33(i)-f23(i)^2);**

**K(3,2,i)=-f23(i)/(f23(i)^2-f22(i)\*f33(i));**

**end**

**for i=1:size(C,1)**

**for j=1:size(CARVA,1)**

**if CARVA(j,1)==i**

**dt2(i,1)=-double(int(CARVA(j,2)\*(((x-X(i))^2)\*(X(i)-x))/(2\*C(i,7)\*Ix(i)),[0 X(i)])) ;**

**th2(i,1)=-double(int(CARVA(j,2)\*((x-X(i))^2)/(2\*C(i,7)\*Ix(i)),[0 X(i)]));**

**else**

**end**

**end**

**end**

**for i=1:size(dt2,1)**

**QQ2(:,i)=-inv([f22(i,1) f23(i,1) ; f23(i,1) f33(i,1)])\*([dt2(i,1) ; th2(i,1)]);**

**end**

**QQ=[ QQ2(1,:);-QQ2(2,:); QQ1];**

**IC=size(CO,1)\*3-size(find(A(:,2:4)),1);**

**G=zeros(size(CO,1),3);**

**for i=1:size(CO,1)**

**for j=1:size(A,1)**

**if i==A(j,1)**

**G(i,:)=A(j,2:4);**

**else**

**end**

**end**

**end**

**t=1;**

**r=IC+1;**

**GDL=zeros(size(CO,1),3);**

**for i=1:size(CO,1)**

**if G(i,1)==0**

**GDL(i,1)=t;**

**t=t+1;**

**else**

**GDL(i,1)=r ;**

**r=r+1;**

**end**

**if G(i,2)==0**

**GDL(i,2)=t;**

**t=t+1;**

**else**

**GDL(i,2)=r;**

**r=r+1;**

**end**

**if G(i,3)==0**

**GDL(i,3)=t;**

**t=t+1;**

**else**

**GDL(i,3)=r;**

**r=r+1;**

**end**

**end**

**for i=1:size(C,1)**

**cos(i,1)=(CO(C(i,2),1)-CO(C(i,1),1))/X(i);**

**sen(i,1)=(CO(C(i,2),2)-CO(C(i,1),2))/X(i);**

**end**

**T=zeros(6,6,size(C,1));**

**for i=1:size(C,1)**

**T(1,1,i)=cos(i); T(1,2,i)=-sen(i); T(2,1,i)=sen(i); T(2,2,i)=cos(i); T(3,3,i)=1;**

**T(4,4,i)=cos(i); T(4,5,i)=-sen(i); T(5,4,i)=sen(i); T(5,5,i)=cos(i); T(6,6,i)=1;**

**end**

**U=zeros(size(C,1),6);**

**for i= 1:size(C,1)**

**for j= 1:size(CO,1)**

**if C(i,1)==j**

**U(i,1)=GDL(j,1);**

**U(i,2)=GDL(j,2);**

**U(i,3)=GDL(j,3);**

**else**

**end**

**end**

**end**

**for i= 1:size(C,1)**

**for j= 1:size(CO,1)**

**if C(i,2)==j**

**U(i,4)=GDL(j,1);**

**U(i,5)=GDL(j,2);**

**U(i,6)=GDL(j,3);**

**else**

**end**

**end**

**end**

**L=zeros(size(CO,1)\*3,6,size(C,1));**

**for i=1:size(C,1)**

**L(U(i,1),1,i)=1;**

**L(U(i,2),2,i)=1;**

**L(U(i,3),3,i)=1;**

**L(U(i,4),4,i)=1;**

**L(U(i,5),5,i)=1;**

**L(U(i,6),6,i)=1;**

**end**

**for i=1:size(C,1)**

**Ki(:,:,i)=L(:,:,i)\*T(:,:,i)\*K(:,:,i)\*((T(:,:,i))')\*((L(:,:,i))');**

**end**

**KE=zeros(size(CO,1)\*3);**

**for i=1:size(C,1)**

**KE=KE+Ki(:,:,i);**

**end**

**FEPL= [zeros(1,size(QQ,2)); QQ(1,:); QQ(2,:);zeros(1,size(QQ,2)); QQ(3,:);QQ(4,:)];**

**for i=1:size(FEPL,2)**

**FEPG(:,i)= L(:,:,i)\*T(:,:,i)\*FEPL(:,i);**

**end**

**SFEPG=zeros(size(FEPG,1),1);**

**for i=1:size(FEPL,2)**

**SFEPG=SFEPG+FEPG(:,i);**

**end**

**KQQ=KE(1:IC,1:IC);**

**p=zeros(size(CO,1),3);**

**for i=1:size(CA,1)**

**for j=1:size(CO,1)**

**if CA(i,1)==j**

**p(j,:)=CA(i,2:4);**

**end**

**end**

**end**

**[v1,v2]=find(p);**

**V= [v1 v2];**

**P=zeros(size(CO,1)\*3,1);**

**for i=1:size(V,1)**

**P((GDL(V(i,1),V(i,2))),1)=p(V(i,1),V(i,2));**

**end**

**Q=P(1:IC,:);**

**DQ=inv(KQQ)\*(Q-SFEPG(1:IC,1));**

**KRQ=KE(IC+1:size(KE,1),1:IC);**

**R=KRQ\*DQ+SFEPG(IC+1:size(SFEPG,1),1);**

**MA(1,1)=R(3,1);**

**RyD(1,1)=R(5,1);**

**XB(1,1)=DQ(1,1);**

**thC(1,1)=DQ(12,1);**

**clearvars -except RyD MA XB thC disc**

**for cn=1:11**

**A=load('APOYO.txt');**

**CA=load('CARGA.txt');**

**CARVA=load('CARVA.txt');**

**C=load('CONEC.txt');**

**CO=load('COORD.txt');**

**C(:,7)=C(:,7)\*10;**

**if cn <= 10**

**C(:,3)=disc(cn);**

**else**

**end**

**for i=1:size(C,1)**

**if C(i,5)~=C(i,6)**

**NP(i,1)=i;**

**else**

**end**

**end**

**NP(NP==0)=[];**

**for i=1:size(C,1)**

**X(i,1)=sqrt((CO(C(i,1),1)-CO(C(i,2),1))^2+(CO(C(i,1),2)-CO(C(i,2),2))^2);**

**end**

**for i=1:size(C,1)**

**cos(i,1)=(CO(C(i,2),1)-CO(C(i,1),1))/X(i);**

**sen(i,1)=(CO(C(i,2),2)-CO(C(i,1),2))/X(i);**

**end**

**for i=1: size(C,1)**

**for j=1:size(CARVA,1)**

**if i==CARVA(j,1)**

**C(i,10)=CARVA(j,2);**

**else**

**end**

**end**

**end**

**for i=1: size(NP,1)**

**for j=1:size(C,1)**

**if NP(i,1)==j**

**N(i,1)=C(j,5);**

**N(i,2)=C(j,6);**

**N(i,3)=X(j);**

**N(i,4)=C(j,4);**

**N(i,5)=C(j,3);**

**N(i,6)=C(j,1);**

**N(i,7)=C(j,2);**

**N(i,8)=j;**

**N(i,9)=C(j,7);**

**N(i,10)= cos(j,1);**

**N(i,11)= sen(j,1);**

**N(i,12)=C(j,10);**

**else**

**end**

**end**

**end**

**syms x**

**for i=1:size(N,1)**

**hx(i,1)=N(i,1)+(N(i,2)-N(i,1))\*x/N(i,3);**

**end**

**Ax=hx.\*N(:,4);**

**Ix=(hx.^3).\*N(:,4)/12;**

**for i=1:size(N,1)**

**for j=1:N(i,5)**

**I(i,j)=double(int(Ix(i),[N(i,3)\*(j-1)/N(i,5) N(i,3)\*j/N(i,5)])/(N(i,3)/N(i,5)));**

**end**

**end**

**I=I';**

**I=I(:);**

**for i=1:size(N,1)**

**for j=1:N(i,5)**

**x=(N(i,3))\*(j-1/2)/(N(i,5));**

**AA(i,j)=subs(Ax(i));**

**end**

**end**

**AA=double(AA);**

**AA=AA';**

**AA=AA(:);**

**for i=1:size(C,1)**

**for j= 1:size(NP,1)**

**if C(NP(j,1))==i**

**C(i,:)=0;**

**end**

**end**

**end**

**C(all(~C,2), : ) = [];**

**C(:,8)=C(:,4).\*C(:,5);**

**C(:,9)=C(:,4).\*(C(:,5).^3)./12;**

**C(:,3:6)=[];**

**s=size(C,1)+1;**

**C=[C ; zeros(size(AA,1),size(C,2))];**

**C(s:size(C,1),4)=AA;**

**C(s:size(C,1),5)=I;**

**for i=1:size(N,1)**

**C(s:s+N(i,5)-1,3)= N(i,9);**

**C(s:s+N(i,5)-1,6)= N(i,12);**

**s=s+N(i,5);**

**end**

**NN(:,1)=N(:,5)-1;**

**NN(:,2)=N(:,6) ;**

**NN(:,3)= N(:,3)./N(:,5);**

**NN(:,4)= NN(:,3).\*N(:,10);**

**NN(:,5)=NN(:,3).\*N(:,11);**

**NN(:,6)= N(:,7);**

**for i=1:size(NN,1)**

**NN(i,7)=CO(NN(i,2),1);**

**NN(i,8)= CO(NN(i,2),2);**

**end**

**a=size(CO,1);**

**f=size(CO,1)+1;**

**CO= [CO ; zeros(sum(NN(:,1)),2)];**

**for i=1:size(NN,1)**

**for j=1:NN(i,1)**

**if j==1**

**CO(j+a,1)=NN(i,7)+NN(i,4);**

**CO(j+a,2)=NN(i,8)+NN(i,5);**

**else**

**CO(j+a,1)=CO(j+a-1,1)+NN(i,4);**

**CO(j+a,2)=CO(j+a-1,2)+NN(i,5);**

**end**

**end**

**a=a+j;**

**end**

**for i=1:size(N,1)**

**for j=1:N(i,5)+1**

**if j==1**

**NR(i,j)=NN(i,2);**

**else**

**if j==N(i,5)+1**

**NR(i,j)=NN(i,6);**

**else**

**NR(i,j)=f;**

**f=f+1;**

**end**

**end**

**end**

**end**

**a=size(find(C(:,1)),1);**

**for i=1:size(N,1)**

**for j=1:N(i,5)**

**C(j+a,1)=NR(i,j);**

**C(j+a,2)=NR(i,j+1);**

**end**

**a=a+j;**

**end**

**for i=1:size(C,1)**

**XX(i,1)=sqrt((CO(C(i,1),1)-CO(C(i,2),1))^2+(CO(C(i,1),2)-CO(C(i,2),2))^2);**

**end**

**for i=1:size(C,1)**

**K(1,1,i)= C(i,4)\*C(i,3)/XX(i);**

**K(2,2,i)= 12\*C(i,3)\*C(i,5)/(XX(i)^3);**

**K(2,3,i)= 6\*C(i,3)\*C(i,5)/(XX(i)^2);**

**K(3,3,i)= 4\*C(i,3)\*C(i,5)/XX(i);**

**K(3,2,i)=K(2,3,i);K(4,4,i)=K(1,1,i);K(1,4,i)=-K(1,1,i);K(4,1,i)=-K(1,1,i);K(5,2,i)=-K(2,2,i);K(2,5,i)=-K(2,2,i); K(2,6,i)=K(2,3,i);**

**K(6,2,i)=K(2,3,i);K(3,5,i)=-K(3,2,i);K(5,3,i)=-K(3,2,i);K(3,6,i)=K(3,3,i)/2;K(6,3,i)=K(3,3,i)/2;K(5,5,i)=K(2,2,i);K(6,6,i)=K(3,3,i);**

**K(5,6,i)=-K(2,3,i);K(6,5,i)=-K(2,3,i);**

**end**

**IC=size(CO,1)\*3-size(find(A(:,2:4)),1);**

**G=zeros(size(CO,1),3);**

**for i=1:size(CO,1)**

**for j=1:size(A,1)**

**if i==A(j,1)**

**G(i,:)=A(j,2:4);**

**else**

**end**

**end**

**end**

**t=1;**

**r=IC+1;**

**GDL=zeros(size(CO,1),3);**

**for i=1:size(G,1)**

**for j=1:size(G,2)**

**if G(i,j)==1**

**GDL(i,j)=r;**

**r=r+1;**

**else**

**GDL(i,j)=t;**

**t=t+1;**

**end**

**end**

**end**

**for i=1:size(C,1)**

**ncos(i,1)=(CO(C(i,2),1)-CO(C(i,1),1))/XX(i);**

**nsen(i,1)=(CO(C(i,2),2)-CO(C(i,1),2))/XX(i);**

**end**

**T=zeros(6,6,size(C,1));**

**for i=1:size(C,1)**

**T(1,1,i)=ncos(i); T(1,2,i)=-nsen(i); T(2,1,i)=nsen(i); T(2,2,i)=ncos(i); T(3,3,i)=1;**

**T(4,4,i)=ncos(i); T(4,5,i)=-nsen(i); T(5,4,i)=nsen(i); T(5,5,i)=ncos(i); T(6,6,i)=1;**

**end**

**U=zeros(size(C,1),6);**

**for i= 1:size(C,1)**

**for j= 1:size(CO,1)**

**if C(i,1)==j**

**U(i,1)=GDL(j,1);**

**U(i,2)=GDL(j,2);**

**U(i,3)=GDL(j,3);**

**else**

**end**

**end**

**end**

**for i= 1:size(C,1)**

**for j= 1:size(CO,1)**

**if C(i,2)==j**

**U(i,4)=GDL(j,1);**

**U(i,5)=GDL(j,2);**

**U(i,6)=GDL(j,3);**

**else**

**end**

**end**

**end**

**L=zeros(size(CO,1)\*3,6,size(C,1));**

**for i=1:size(C,1)**

**L(U(i,1),1,i)=1;**

**L(U(i,2),2,i)=1;**

**L(U(i,3),3,i)=1;**

**L(U(i,4),4,i)=1;**

**L(U(i,5),5,i)=1;**

**L(U(i,6),6,i)=1;**

**end**

**for i=1:size(C,1)**

**Ki(:,:,i)=L(:,:,i)\*T(:,:,i)\*K(:,:,i)\*((T(:,:,i))')\*((L(:,:,i))');**

**end**

**KE=zeros(size(CO,1)\*3);**

**for i=1:size(C,1)**

**KE=KE+Ki(:,:,i);**

**end**

**KQQ=KE(1:IC,1:IC);**

**p=zeros(size(CO,1),3);**

**for i=1:size(CA,1)**

**for j=1:size(CO,1)**

**if CA(i,1)==j**

**p(j,:)=CA(i,2:4);**

**end**

**end**

**end**

**[v1,v2]=find(p);**

**V= [v1 v2];**

**P=zeros(size(CO,1)\*3,1);**

**for i=1:size(V,1)**

**P((GDL(V(i,1),V(i,2))),1)=p(V(i,1),V(i,2));**

**end**

**Q=P(1:IC,:);**

**for i=1:size(C,1)**

**FEP(2,i)= XX(i,1)\*C(i,6)/2;**

**FEP(3,i)= (XX(i,1)^2)\*C(i,6)/12;**

**FEP(5,i)= XX(i,1)\*C(i,6)/2;**

**FEP(6,i)= -(XX(i,1)^2)\*C(i,6)/12;**

**end**

**for i=1:size(C,1)**

**FEPG(:,i)=L(:,:,i)\*T(:,:,i)\*FEP(:,i) ;**

**end**

**SFEPG=zeros(size(FEPG,1),1);**

**for i=1:size(FEP,2)**

**SFEPG=SFEPG+FEPG(:,i);**

**end**

**DQ=inv(KQQ)\*(Q-SFEPG(1:IC,1));**

**KRQ=KE(IC+1:size(KE,1),1:IC);**

**R=KRQ\*DQ+SFEPG(IC+1:size(SFEPG,1),1);**

**MA(:,cn+1)=R(3,1);**

**RyD(:,cn+1)=R(5,1);**

**XB(:,cn+1)=DQ(1,1);**

**thC(:,cn+1)=DQ(12,1);**

**clearvars -except RyD MA XB thC disc R DQ IC**

**end**

**fprintf('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \n ')**

**fprintf(' Con el nivel de discretización ingresado:\n');**

**fprintf('La reacción en el eje x de A es igual a '), fprintf('%.2f t.\n', R(1,1))**

**fprintf('La reacción en el eje y de A es igual a '), fprintf('%.2f t.\n', R(2,1))**

**fprintf('El momento de reaccion en A es igual a '), fprintf('%.2f tm.\n', R(3,1))**

**fprintf('La reacción en el eje x de D es igual a '), fprintf('%.2f t.\n', R(4,1))**

**fprintf('La reacción en el eje y de D es igual a '), fprintf('%.2f t.\n', R(5,1))**

**fprintf('El momento de reaccion en D es igual a '), fprintf('%.2f tm.\n', R(6,1))**

**fprintf('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \n ')**

**fprintf(' Los desplazamientos en los GDL desconocidos son:\n ');**

**fprintf(' GDL \t desplazamientos (cm y rad^-2) \n');**

**fprintf(' %1.0f \t\t %0.4f \n ', [[1:IC];(DQ.\*100)'] )**

**fprintf('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \n ')**

**fprintf('El desplazamiento horizontal en B es igual a '), fprintf('%.5f cm.\n', DQ(1,1)\*100)**

**fprintf('El desplazamiento vertical en B es igual a '), fprintf('%.5f cm.\n', DQ(2,1)\*100)**

**fprintf('El giro en B es igual a '), fprintf('%.5f rad.\*10^-2\n', DQ(3,1)\*100)**

**fprintf('El desplazamiento horizontal en C es igual a '), fprintf('%.5f cm.\n', DQ(10,1)\*100)**

**fprintf('El desplazamiento vertical en C es igual a '), fprintf('%.5f cm.\n', DQ(11,1)\*100)**

**fprintf('El giro en C es igual a '), fprintf('%.5f rad.\*10^-2\n', DQ(12,1)\*100)**

**figure**

**subplot(2,2,1)**

**plot(disc, MA(:,2:11), [1 disc(1,10)], [1 1]\*MA(:,1))**

**title('Momento de reaccion en A [tm.]')**

**subplot(2,2,2)**

**plot(disc, RyD(:,2:11),[1 disc(1,10)], [1 1]\*RyD(:,1))**

**title('Reaccion vertical en D [t.]')**

**subplot(2,2,3)**

**plot(disc, XB(:,2:11),[1 disc(1,10)], [1 1]\*XB(:,1))**

**title('Desp. horizontal en B [m.]')**

**subplot(2,2,4)**

**plot(disc, thC(:,2:11),[1 disc(1,10)], [1 1]\*thC(:,1))**

**title('Rotacion C [rad.]')**