Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Mecánica

Sistematización del método directo para el cálculo elástico lineal de estructuras

Autor:

Yuste Yuste, Alejandro

Tutor(es):

Cacho Pérez, Mariano

Dpto. Construcciones Arquitectónicas, Ingeniería del Terreno y Mecánica de los Medios Continuos y Teoría de Estructuras

Valladolid, Junio 2020

Resumen

Durante el grado universitario de Ingeniería Mecánica se imparten distintos métodos para el cálculo lineal de estructuras en asignaturas como *Estructuras y Construcciones Industriales* o *Elasticidad y resistencia de materiales*. Pero, entre todos ellos no se encuentra el método directo aplicado al cálculo lineal, el cual se imparte relacionándolo con el cálculo plástico.

Debido a ello, se presenta este Trabajo Fin de Grado donde se comprobará si se puede utilizar dicho método para este fin. Y en caso afirmativo, se realizará un pequeño código informático que nos ayudará a aplicar esta metodología de forma sencilla e intuitiva.

Palabras clave

Cálculo lineal, método directo, momentos flectores, estructura, carga.

Abstract

During the university degree in Mechanical Engineering different methods are taught for the linear calculation of structures in subjects such as Industrial Structures and Constructions or Elasticity and resistance of materials. But, among all of them the direct method applied to the linear calculation is not found, which is taught by relating it to the plastic calculation.

Because of this, this Final Degree Project is presented where you can check if you can use this method for this purpose. And if so, a small computer code will be run to help us apply this methodology in a simple and intuitive way.

Keywords

Lineal calculus, direct method, bending moments, structure, load.

Índice General

Índice de figuras

Capítulo 1

INTRODUCCIÓN

1.Introducción

1.1 Objeto del proyecto

Dentro del Grado universitario de Ingeniería Mecánica existe un bloque de la docencia dedicada al cálculo de estructuras, el cual se podría dividir en dos partes. Por un lado, el cálculo elástico y por otro, el cálculo plástico.

Según el criterio de los diferentes departamentos que imparten las asignaturas de este bloque, ubicaron el método directo en la parte del cálculo plástico. De esta manera, a los alumnos se nos enseñó a utilizar este método para calcular las cargas y los mecanismos de colapso de estructuras. Pero este método tiene más utilidades que exclusivamente el cálculo plástico.

Por este motivo se realiza este trabajo, cuya finalidad será comprobar si el método directo se puede aplicar al cálculo lineal elástico de estructuras, para que, posteriormente, poder realizar un código informático sencillo e intuitivo que simplifique estos cálculos.

1.2 Contenido del proyecto

En primer lugar, para poder aplicar el método directo, se necesita conocer la base teórica que lo fundamenta. La parte teórica de este Trabajo Fin de Grado se explicará el método directo, englobando las diferentes partes en las que se divide, como son el Principio de Fuerzas Virtuales, el Principio de Desplazamientos Virtuales, etcétera.

Posteriormente, en la parte práctica del proyecto, se ejecutan las comprobaciones necesarias para establecer si la aplicación del método directo es factible de aplicar al cálculo de estructuras. Estas comprobaciones se realizarán utilizando varios ejemplos de estructuras, a las cuales se les aplicarán diferentes cargas puntuales y distribuidas.

Una vez comprobado que el método directo funciona, a continuación, se encuentra una descripción del código empleado en la sistematización del problema. Detallando cada punto e indicando como trabaja el programa a la hora de compilar el código.

Por último, se aplicará el código a los mismos ejemplos utilizados en la validación del método para comprobar que se haya programado correctamente. Y, adicionalmente, se utilizará el código para resolver un problema real, un pórtico a dos aguas.

Capítulo 2

MÉTODO DIRECTO

2. Método directo

Como se ha indicado, la base de este Trabajo Fin de Grado se encuentra en el método directo, por ello, en este capítulo se explicará brevemente en que se basa este método.

2.1 Análisis visual y primeras nociones

En primer lugar, al comenzar a aplicar este método se realizará una inspección visual con el objetivo de conocer los detalles de la estructura y así poder obtener en un segundo plano aquellos datos más relevantes, como son el grado de hiperestaticidad, el número de secciones candidatas para el cálculo de momentos y el número de ecuaciones de equilibrio necesarias.

Para conocer el grado de hiperestaticidad (GH) se aplicará la Ec. 1. De este dato se obtendrá el número de ecuaciones de compatibilidad necesarias, las cuales se calcularán posteriormente mediante la aplicación del Principio de Fuerzas virtuales

$$GH = (r - 3) + (3c - l) \qquad \text{Ec. 1}$$

Donde “r” es el número de reacciones en los diferentes apoyos existentes, “c” corresponde con el número de bucles internos y “l” corresponde con el número de libertades internas.

Para conocer el número de secciones candidatas (NPR) para la aplicación del método se realiza mediante una inspección visual a la estructura, eligiendo aquellos puntos que cumplan uno o más puntos de los indicados a continuación:

- Empotramientos, nudos rígidos y uniones no articuladas donde concurran dos o más elementos
- Los apoyos intermedios en las vigas continuas
- Los puntos de aplicación de las cargas puntuales

2. Método directo

- Un punto cualquiera, definido por el usuario, del elemento bajo el que soporta una carga distribuida. Para el punto elegido se tendrá en cuenta su posición respecto de los ejes de coordenadas relativos al elemento.

Para conocer el número de ecuaciones necesarias (EQ) primeramente se deben conocer el GH y el NPR. De esta manera, aplicando Ec. 2 se puede conocer este número.

$$EQ = NPR - GH \qquad \text{Ec. 2}$$

2.2 Mecanismo virtuales mediante Teoría de Mecanismos

Las ecuaciones de equilibrio se obtendrán mediante la aplicación del Principio de Desplazamientos Virtuales a mecanismos virtuales, basados en los mecanismos de colapso independientes que se pueden formar en la estructura del problema.

Por tanto, lo primero será conocer el número de mecanismos virtuales necesarios y como están formados cada uno de ellos. Para conocer e implementar en el código la búsqueda de estos mecanismos se hará utilidad de la Teoría de Mecanismos, en concreto se utilizarán las conocidas matrices homogéneas.

2.2.1 Matrices homogéneas

Para conocer los desplazamientos de los diferentes puntos de la estructura se hará uso de las matrices homogéneas, metodología que será la utilizada dentro del código. Una matriz homogénea (T) consta de:

$$T = \begin{pmatrix} R_{3x3} & p_{3x1} \\ f_{1x3} & w_{1x1} \end{pmatrix} \qquad \text{Ec. 3}$$

Siendo “R” la matriz de rotación, “f” representa a la transformación de perspectiva (es un vector con elementos nulos), “p” se refiere al vector posición (recorre desde el origen del sistema de referencia base hasta el origen del

sistema de referencia objetivo) y, por último, “w” correspondiente al factor de escala (que para nuestro caso siempre será igual a la unidad).

Las principales ventajas de estas matrices son:

- Simplicidad conceptual
- Eficiencia computacional
- Ahorro de memoria

De esta manera, para el caso de este Trabajo, la matriz homogénea queda como:

$$T = \begin{pmatrix} \bar{\bar{R}} & \bar{p} \\ \bar{0} & 1 \end{pmatrix} \qquad \text{Ec. 4}$$

La cual se utilizará principalmente para dos tipos de desplazamientos concretos:

- Traslación pura:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \bar{\bar{I}} & \bar{p} \\ \bar{0} & 1 \end{pmatrix} \begin{pmatrix} \bar{r} \\ 1 \end{pmatrix} =$$

$$= \begin{pmatrix} \bar{\bar{I}}\bar{r} + \bar{p} \cdot 1 \\ \bar{0}\bar{r} + 1 \cdot 1 \end{pmatrix} = \begin{pmatrix} \bar{r} + \bar{p} \\ 1 \end{pmatrix} = \begin{pmatrix} x + p_x \\ y + p_y \\ z + p_z \\ 1 \end{pmatrix} \qquad \text{Ec. 5}$$

- Rotación pura:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \bar{\bar{R}} & \bar{0} \\ \bar{0} & 1 \end{pmatrix} \begin{pmatrix} \bar{r} \\ 1 \end{pmatrix} =$$

Ec. 6

2. Método directo

$$= \begin{pmatrix} \bar{\bar{R}}\bar{r} + \bar{0} \cdot 1 \\ \bar{0}\bar{r} + 1 \cdot 1 \end{pmatrix} = \begin{pmatrix} \bar{\bar{R}}\bar{r} \\ 1 \end{pmatrix} = \begin{pmatrix} R_{11}x + R_{12}y + R_{13}z \\ R_{21}x + R_{22}y + R_{23}z \\ R_{31}x + R_{32}y + R_{33}z \\ 1 \end{pmatrix}$$

2.3 Principio de Desplazamientos Virtuales (PDV)

Una vez conocidos los mecanismos virtuales, ya podemos utilizarlos para obtener las diferentes ecuaciones de equilibrio necesarias. Para ello se utilizará el Principio de Desplazamientos Virtuales

Principio que fue formulado por Johann Bernouilli (1717) como : "Dado un cuerpo rígido mantenido en equilibrio por un sistema de fuerzas, el trabajo virtual efectuado por este sistema, durante un desplazamiento virtual, es nulo"

Entonces, si sobre una partícula A actúa un sistema de fuerzas F1,F2...Fn y sufre un desplazamiento δr, al cual denominaremos desplazamiento virtual. El trabajo realizado por estas fuerzas durante el desplazamiento será:

$$\delta U = \overrightarrow{F_1} \cdot \delta_r + \overrightarrow{F_2} \cdot \delta_r + \cdots + \overrightarrow{F_n} \cdot \delta_r = \qquad \text{Ec. 7}$$
$$= \left(\overrightarrow{F_1} + \overrightarrow{F_2} + \cdots + \overrightarrow{F_n}\right)\delta_r = \vec{R} \cdot \delta_r$$

El Principio de Desplazamientos Virtuales de una partícula establece que, si una partícula está en equilibrio, el trabajo virtual total de las fuerzas que concurren a la partícula es nulo, para cualquier desplazamiento virtual considerado.

2.4 Principio de Fuerzas Virtuales (PFV)

Tras la actuación de las fuerzas exteriores sobre la estructura que cumple con las hipótesis de pequeños desplazamientos, las componentes del grupo de acciones y reacciones estarán en equilibrio, y las componentes de las solicitaciones y deformaciones cumplirán con las condiciones de compatibilidad. Se impone entonces un estado virtual de fuerzas cuya puesta en carga es instantánea.

Las características de las fuerzas virtuales son:

- Serán independientes de las fuerzas exteriores reales que actúan sobre la estructura
- Cumplirán las condiciones de equilibrio constituyendo una configuración estática admisible. El equilibrio virtual de fuerzas se plantea sobre la geometría indeformada de la estructura
- Actuarán con un valor constante

Las fuerzas virtuales exteriores y los desplazamientos realizan, en cada estado virtual, un trabajo virtual instantáneo llamado trabajo virtual de las fuerzas exteriores (δW). Internamente, las tensiones virtuales y las deformaciones reales realizaran un trabajo virtual complementario interno que se acumula en forma de energía virtual complementaria de deformación (δU)

Las condiciones de compatibilidad de la estructura real no se modifican, por lo que el trabajo virtual complementario de las fuerzas exteriores será igual a la energía virtual complementaria de deformación. Se trata de una ecuación de balance energético que representa una condición de compatibilidad de la estructura:

$$\delta W = \delta U \qquad \text{Ec. 8}$$

Resumiendo todo esto, y para una aplicación del principio más sencilla se utilizará la siguiente ecuación a la hora de utilizar el método:

Para cargas puntuales:

$$0 = \frac{L}{6EI}[m_a(2M_a + M_b) + m_b(2M_b + M_a)] \qquad \text{Ec. 9}$$

Para cargas uniformes

$$0 = \frac{L}{6EI}\left[m_a(2M_a + M_b) + m_b(2M_b + M_a) + \frac{qL^2}{4}(m_a + m_b)\right] \qquad \text{Ec.10}$$

Capítulo 3

EJEMPLOS DE VALIDACIÓN

3. Ejemplos de validación

Para comenzar el apartado práctico del trabajo, lo primero de todo es comprobar si la aplicación del método directo explicado con anterioridad es factible para el cálculo de estructuras, sin que estás lleguen a plastificar. Para ello, en los subapartados de este capítulo se estudiarán varios ejemplos, dos de ellos basados en un pórtico recto a los que se le aplican, a uno de ellos cargas puntuales y al otro cargas distribuidas, y un tercer ejemplo compuesto por una viga soportando cargas puntuales.

El procedimiento seguido ha sido calcular en primera instancia, utilizando el método directo, los momentos en los distintos puntos característicos del problema y posteriormente comprobar dichos resultados utilizando el Método Directo de Rigidez (MDR) de estructuras, utilizando para ello el programa informático MDR-fx. Un detalle importante a la hora de aplicar el MDR es que se deben considerar nulos los desplazamientos longitudinales de los elementos de la estructura

Adelantando los resultados, efectivamente se puede utilizar la metodología del método directo para conocer los momentos en las estructuras. Por tanto, el siguiente paso será la realización del código que sistematice estos cálculos, el cual se describirá en el siguiente capítulo.

3.1 Ejemplo 1. Pórtico biempotrado con cargas puntuales

El caso que nos incumbe en este apartado será el cálculo de los momentos para un pórtico biempotrado con una carga vertical y hacia abajo en el punto medio de la flecha y otra carga horizontal y hacia la derecha en el punto más alto del pilar derecho. Toda la estructura está construida con la misma geometría, siendo las características de las mismas E, A e I.

3. Ejemplos de validación

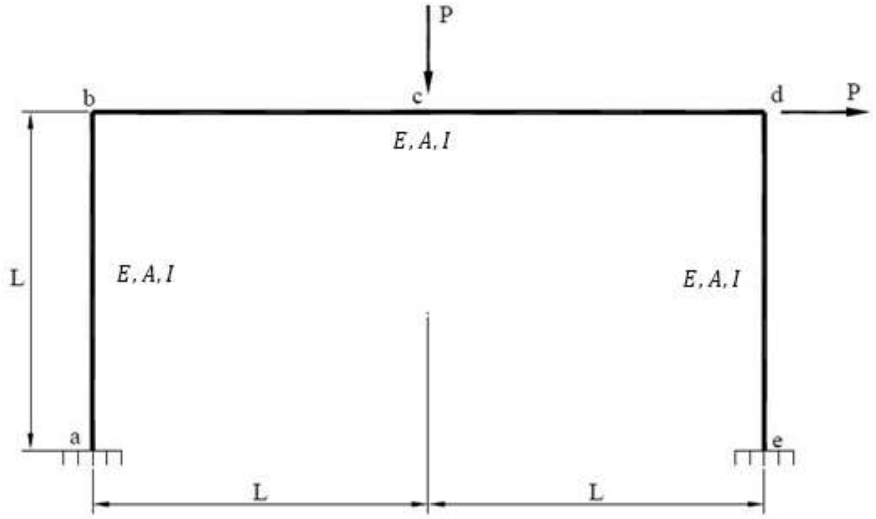

Figura 1: Ejemplo 1

Para la aplicación del cálculo plástico se han seguido las pautas descritas anteriormente. En primer lugar, se hace una inspección visual para localizar las secciones candidatas a la aparición de la rótula plástica (NPR), sumando un total de 5 secciones candidatas:

- Secciones “a” y “e” por encontrarse en esos nodos el arriostramiento al suelo de la estructura mediante un empotramiento
- Secciones “b” y “d” por ser el nodo donde concurren dos elementos de la estructura
- Sección “c” por ser el nodo de aplicación de una carga puntual (también se puede aplicar a la sección “d”)

El siguiente paso para realizar será calcular el grado de hiperestaticidad de la estructura, para que así, junto con el NPR, se calcule el número de ecuaciones de equilibrio necesarias. Entonces, aplicando la Ec. 1 se obtiene un grado de hiperestaticidad de 3.

Conocidos ambos datos, ya se puede conocer el número de ecuaciones de equilibrio necesarias, utilizando la Ec. 2, que será de 2.

Por tanto, para obtener estas ecuaciones se deben buscar 2 mecanismos independientes formados con la estructura y aplicando el Principio de Desplazamientos Virtuales, obtener las ecuaciones de equilibrio necesarias

- Mecanismo 1: Aquel formado por la aparición de articulaciones en las secciones “b”,”c” y “d”.

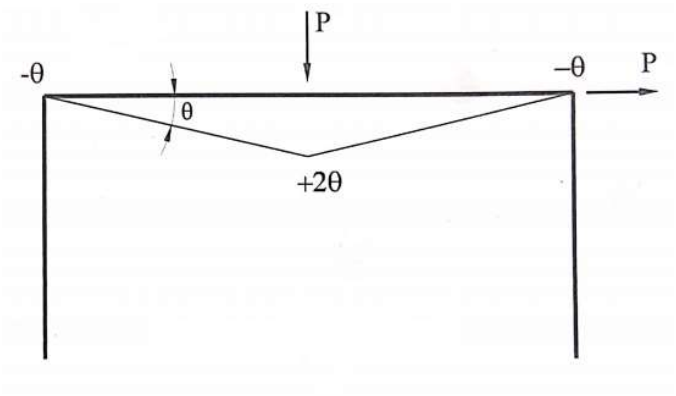

Figura 2: Mecanismo 1 (Ejemplo1)

$$PL\theta = M_b(-\theta) + M_c(2\theta) + M_d(-\theta)$$

$$PL = -M_b + 2M_c - M_d \qquad \text{Ec. 11}$$

La primera ecuación de equilibrio obtenida corresponde con la Ec. *11*.

- Mecanismo 2: Aquel formado por la aparición de articulaciones en las secciones “a”,”b”,”c” y “d”.

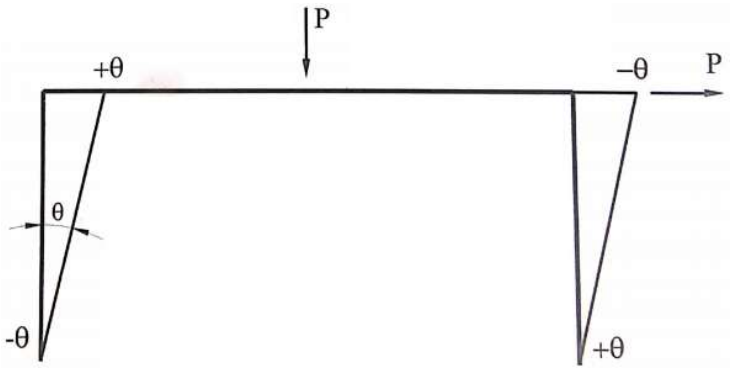

Figura 3: Mecanismo 2 (Ejemplo1)

$$PL\theta = M_a(-\theta) + M_b(\theta) + M_d(-\theta) + M_e(\theta)$$

$$PL = -M_a + M_b - M_d + M_e \qquad \text{Ec. 12}$$

La segunda ecuación de equilibrio obtenida corresponde con la Ec. *12*.

Llegados a este punto ya conocemos las dos ecuaciones de equilibrio necesarias y el siguiente paso será calcular las ecuaciones de compatibilidad. El número necesario de ecuaciones de compatibilidad lo indica el grado de hiperestaticidad, por lo que para este problema se necesitan tres ecuaciones de compatibilidad. Para calcularlas, en la siguiente tabla se mostrará el estado de los momentos flectores, que cumplen las ecuaciones de equilibrio para cargas nulas, elegidos para posteriormente proceder con la aplicación del Principio de Fuerzas Virtuales. También se mostrará una fila con los momentos flectores reales de la estructura que corresponden con las incógnitas del problema

Tabla 1: Equilibrio momentos flectores Ejemplo 1

Sección	a	b	c	d	e
Momentos reales	M_a	M_b	M_c	M_d	M_e
EC1	1	1	1/2	0	0
EC2	0	0	1/2	1	1
EC3	0	1	1	1	0

Para la aplicación del PFV se utilizará la forma sistematizada para cargas puntuales, la cual está representada por la Ec. 9 y como resultado se tiene:

- Ecuación de compatibilidad 1:

$$0 = 3M_a + \frac{11}{2}M_b + 3M_c + \frac{1}{2}M_e \qquad \text{Ec. 13}$$

- Ecuación de compatibilidad 2:

$$0 = \frac{1}{2}M_b + 3M_c + \frac{11}{2}M_d + 3M_e \qquad \text{Ec. 14}$$

- Ecuación de compatibilidad 3:

$$0 = M_a + 5M_b + 6M_c + 5M_d + M_e \qquad \text{Ec. 15}$$

Ya se tienen todas las ecuaciones necesarias, las dos ecuaciones de equilibrio y las tres ecuaciones de compatibilidad, para calcular las 5 incógnitas del problema. Entonces, el último paso sería resolver el sistema de ecuaciones siguiente:

$$\begin{pmatrix} 0 & -1 & 2 & -1 & 0 \\ -1 & 1 & 0 & -1 & 1 \\ 3 & 5.5 & 3 & 0.5 & 0 \\ 0 & 0.5 & 3 & 5.5 & 3 \\ 1 & 5 & 6 & 5 & 1 \end{pmatrix} \begin{pmatrix} M_a \\ M_b \\ M_c \\ M_d \\ M_e \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} PL \qquad \text{Ec. 16}$$

Del sistema de ecuaciones se obtienen las siguientes soluciones:

$$M_a = -\frac{17}{80} \cdot PL\ Nm$$

$$M_b = -\frac{1}{80} \cdot PL\ Nm$$

$$M_c = -\frac{3}{10} \cdot PL\ Nm$$

$$M_d = -\frac{31}{80} \cdot PL\ Nm$$

$$M_e = \frac{33}{80} \cdot PL\ Nm$$

Que particularizadas para P=6000N y L=2m quedan como:

$$M_a = -2550\ Nm$$

$$M_b = -150\ Nm$$

$$M_c = 3600\ Nm$$

$$M_d = -4650\ Nm$$

$$M_e = 4950\ Nm$$

Para conocer si la aplicación del método directo para el cálculo plástico se puede usar en el caso que nos incumbe se ha procedido a realizar el problema utilizando MDR-fx, cómo se indicó anteriormente. Así, los resultados obtenidos han sido:

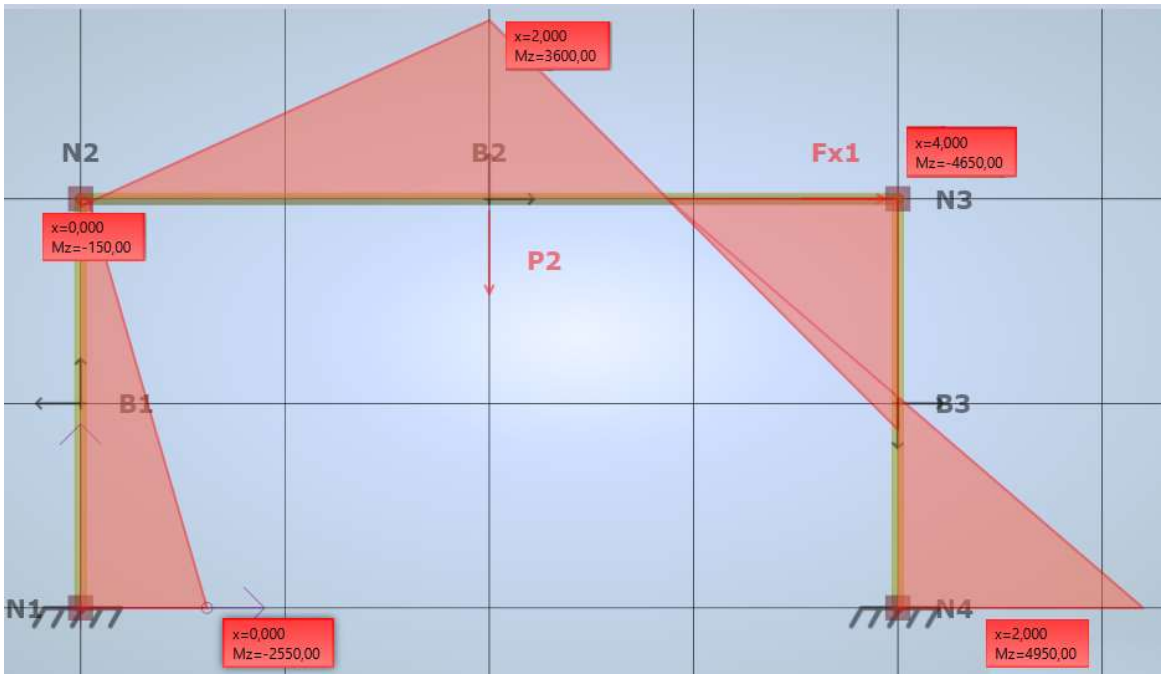

Figura 4: Resultados Ejemplo 1 con MDR-fx

Como se puede observar, los resultados concuerdan correctamente con los resultados obtenido aplicando el método tradicionalmente, por lo que se puede aplicar este procedimiento para el cálculo de los momentos que deben soportar los puntos críticos de una estructura.

3.2 Ejemplo 2. Pórtico biempotrado con carga uniforme

Para comprobar si el método se puede aplicar a problemas con cargas uniformes se ha utilizado el mismo tipo de pórtico biempotrado del problema anterior. La carga que soportará la estructura será una carga uniforme y hacia la derecha aplicada en el pilar izquierdo y las características de las secciones serán E, A e I.

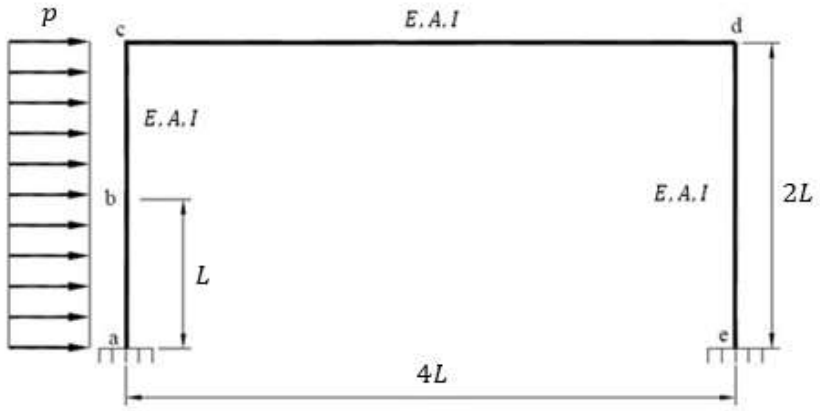

Figura 5: Ejemplo 2

Para este problema, las rótulas candidatas a plastificar son 5:

- Secciones “a” y “e” por encontrarse en esos puntos el arriostramiento de la estructura mediante un empotramiento
- Secciones “c” y “d” por ser puntos donde concurren dos elementos de la estructura
- Sección “b” por ser una sección intermedia dentro del elemento que soporta la carga uniforme, en este caso se ha considerado que es la sección en el punto medio del pilar

Una vez conocido el número de secciones que pueden formar una rótula y conocido también el grado de hiperestaticidad, el número de ecuaciones de equilibrio necesarias será, según la Ec. 2, de 2. Y para calcularlas es necesario plantear dos posibles mecanismos de colapso linealmente independientes. Además, cabe indicar que para agilizar los cálculos se ha considerado desde el principio los valores de q=6000 Nm y L=1 m

- Mecanismo 1: Aquel formado por la aparición de rótulas en las secciones “a”,”c”,”d” y “e”.

Figura 6: Mecanismo 1 (Ejemplo 2)

$$q \cdot L \cdot L \cdot \theta = M_a(-\theta) + M_c(\theta) + M_d(-\theta) + M_e(\theta)$$

$$12000 = -M_a + M_c - M_d + M_e \qquad \text{Ec. 17}$$

La primera ecuación de equilibrio coincide con la Ecuación 17.

- Mecanismo 2: Aquel formado por la aparición de rótulas en las secciones “a”,”b” y “c”

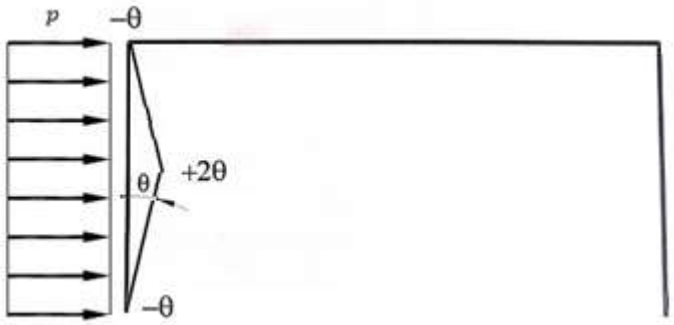

Figura 7: Mecanismo 2 (Ejemplo 2)

$$q \cdot L \cdot \frac{L}{2} \cdot \theta + q \cdot L \cdot \frac{L}{2} \cdot \theta = M_a(-\theta) + M_b(2\theta) + M_c(-\theta)$$

$$6000 = -M_a + 2M_b - M_c \qquad \text{Ec. 18}$$

La segunda ecuación de equilibrio coincide con la Ecuación 18.

Una vez conocidas las ecuaciones de equilibrio, el siguiente paso será calcular las ecuaciones de compatibilidad de la misma manera que se hizo para el pórtico anterior

Tabla 2: Equilibrio momentos flectores Ejemplo 2

Sección	a	b	c	d	e
Momentos reales	M_a	M_b	M_c	M_d	M_e
EC1	1	0	-1	-2	0
EC2	2	1	0	0	2
EC3	0	1/2	1	1	0

Una vez calculados los momentos virtuales se procede a la aplicación del PFV en su forma sistematizada, según la Ec.10. Así, las ecuaciones de compatibilidad son:

- Ecuación de compatibilidad 1:

$$0 = 2M_a - 18M_c - 28M_c - 4M_e \qquad \text{Ec. 19}$$

- Ecuación de compatibilidad 2:

$$-6000 = 5M_a + 6M_b + M_c + 4M_c + 8M_e \qquad \text{Ec. 20}$$

- Ecuación de compatibilidad 3:

$$-3000 = \frac{1}{2}M_a + 3M_b + \frac{29}{2}M_c + 16M_c + 2M_e \qquad \text{Ec. 21}$$

Con las dos ecuaciones de equilibrio y las tres ecuaciones de compatibilidad ya se puede calcular las cinco incógnitas del problema, que coinciden con los momentos en cada sección, resolviendo el siguiente sistema de ecuaciones:

$$\begin{pmatrix} -1 & 2 & -1 & 0 & 0 \\ -1 & 0 & 1 & -1 & 1 \\ 2 & 0 & -18 & -28 & -4 \\ 5 & 6 & 1 & 4 & 8 \\ 0.5 & 3 & 14.5 & 16 & 2 \end{pmatrix} \begin{pmatrix} M_a \\ M_b \\ M_c \\ M_d \\ M_e \end{pmatrix} = \begin{pmatrix} 6000 \\ 12000 \\ 0 \\ -6000 \\ -3000 \end{pmatrix} \quad \text{Ec. 22}$$

El cual nos da como resultado:

$$M_a = -5900\ Nm$$

$$M_b = 700\ Nm$$

$$M_c = 1300\ Nm$$

$$M_d = -1700\ Nm$$

$$M_e = 3100\ Nm$$

Para comprobar los resultados se han seguido las mismas pautas que para el ejemplo anterior. Como se puede observar en la Figura 8: Resultado Ejemplo 2 con MDR-fx los resultados coinciden. Por tanto, para el caso de cargas distribuidas también se podría utilizar el cálculo plástico para conocer los momentos que debe soportar cada sección

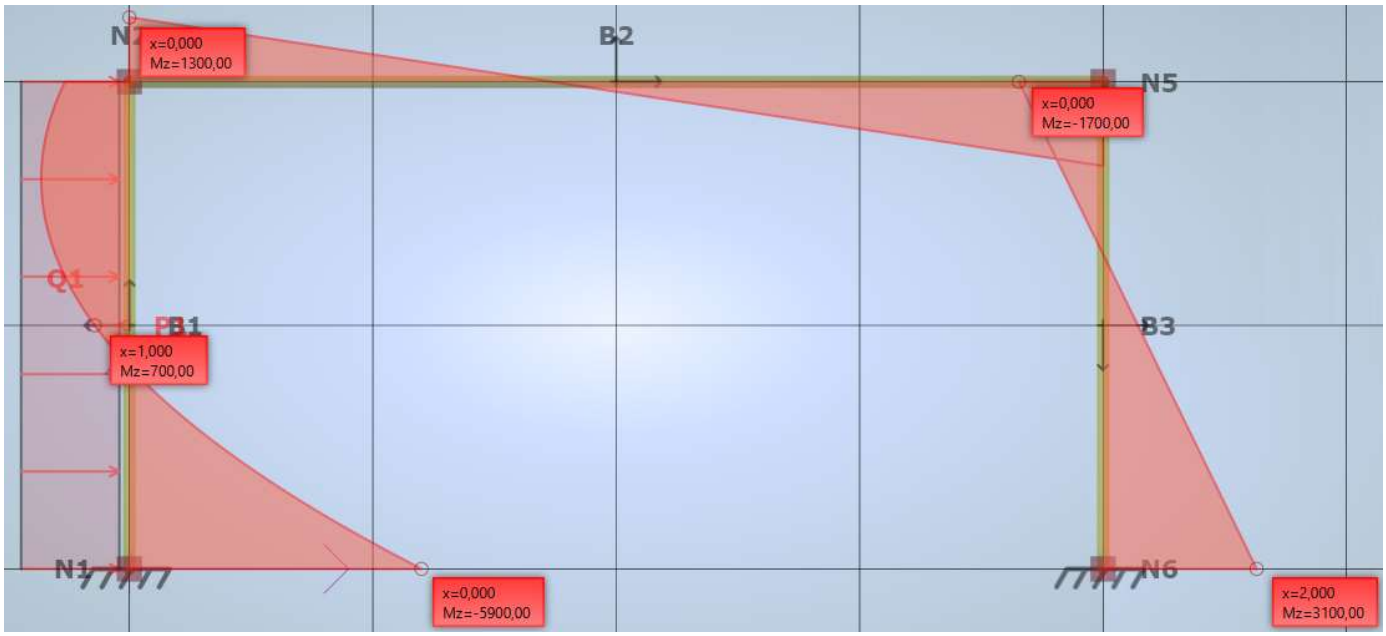

Figura 8: Resultado Ejemplo 2 con MDR-fx

3.3 Otros ejemplos

3.3.1 Ejemplo 3. Viga apoyada-empotrada con cargas puntuales

Para comprobar que esta teoría se cumple en otras configuraciones de barras, a continuación, se ejemplificará el caso de una viga apoyada-empotrada sometida a cargas puntuales. La viga está formada por 2 tipos de secciones, teniendo la barra central el doble de inercia que las barras de los extremos y con las cargas aplicadas en los puntos indicados en la Figura 9: Ejemplo 3.

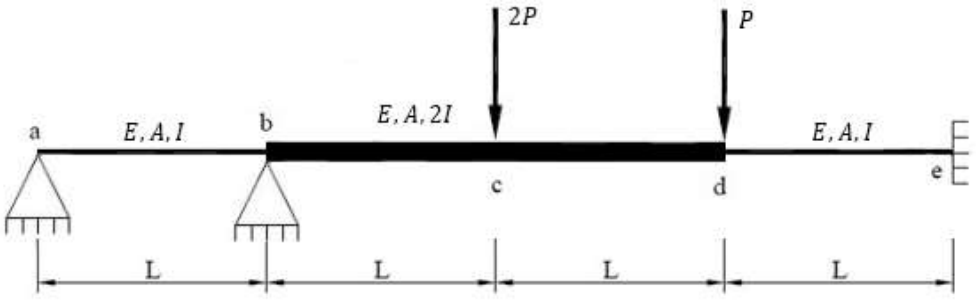

Figura 9: Ejemplo 3

Para este caso se tienen 4 secciones, “b”,”c”,”d” y “e”, candidatas a la aparición de rótulas plásticas. Estas secciones corresponden con la unión de dos elementos de la estructura (b y d), con el punto de aplicación de las cargas (c y d) y con el empotramiento (e). Ahora, el grado de hiperestaticidad es, según la Ec. 1, de 3 pero para este caso, al ser una estructura plana que solo trabaja dentro de un plano, nos deja un grado de hiperestaticidad operacional de 2.

Así, conociendo ambos datos, según la ecuación 4 se necesitan 2 ecuaciones de equilibrio independientes. Para calcular dichas ecuaciones se aplicará el Principio de Desplazamientos Virtuales a los mecanismos de colapso mostrados en la Figura 10: Mecanismos Ejemplo 3.

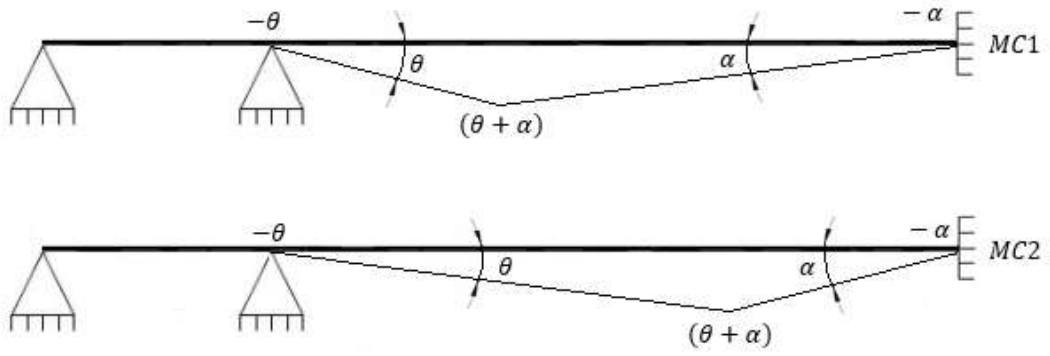

Figura 10: Mecanismos Ejemplo 3

Como resultado se obtienen las siguientes ecuaciones:

3. Ejemplos de validación

- Ecuación de equilibrio 1

$$5PL = -2M_b + 3M_C - M_e \qquad \text{Ec. 23}$$

- Ecuación de equilibrio 2

$$4PL = -M_b + 3M_d - 2M_e \qquad \text{Ec. 24}$$

Conocidas las ecuaciones de equilibrio, el paso siguiente será buscar el estado de momentos virtuales que cumplan con dichas ecuaciones y proceder al cálculo de las ecuaciones de compatibilidad aplicando el Principio de Fuerzas Virtuales (Ec. 9), que para este caso serán 2, como nos indica el grado de hiperestaticidad.

Tabla 3: Equilibrio momentos flectores Ejemplo 3

Sección	a	b	c	d	e
Momentos reales	0	M_b	M_c	M_d	M_e
EC1	0	0	1	2	3
EC2	0	3	2	1	0

Como resultado se obtienen las siguientes ecuaciones de compatibilidad:

- Ecuación de compatibilidad 1

$$0 = \frac{1}{2}M_b + 3M_c + \frac{19}{2}M_d + 8M_e \qquad \text{Ec. 25}$$

- Ecuación de compatibilidad 2

$$0 = 10M_b + 6M_c + 4M_d + M_e \qquad \text{Ec. 26}$$

Como último paso será resolver el siguiente sistema formado por las ecuaciones de equilibrio y las de compatibilidad, y particularizado para P=6000 N y L=2 m.

$$\begin{pmatrix} -2 & 3 & 0 & -1 \\ -1 & 0 & 3 & -2 \\ 1/2 & 3 & 19/2 & 8 \\ 10 & 6 & 4 & 1 \end{pmatrix} \begin{pmatrix} M_b \\ M_c \\ M_d \\ M_e \end{pmatrix} = \begin{pmatrix} 60000 \\ 48000 \\ 0 \\ 0 \end{pmatrix} \qquad \text{Ec. 27}$$

Con el sistema resuelto se obtienen los siguientes valores (teniendo en cuenta que el momento flector en el punto a es nulo):

$$M_a = 0\ Nm$$

$$M_b = -7980.30\ Nm$$

$$M_c = 11054.19\ Nm$$

$$M_d = 6088.67\ Nm$$

$$M_e = -10876.85\ Nm$$

Una vez aplicado el cálculo directo al problema, para comprobar que los resultados son correctos se ha vuelto a utilizar el programa MDR-fx. Y, efectivamente, como se demuestra en la Figura 11: Resultados Ejemplo 3 con MDR-fx para este caso también se puede aplicar esta metodología para calcular los momentos en las secciones de la estructura

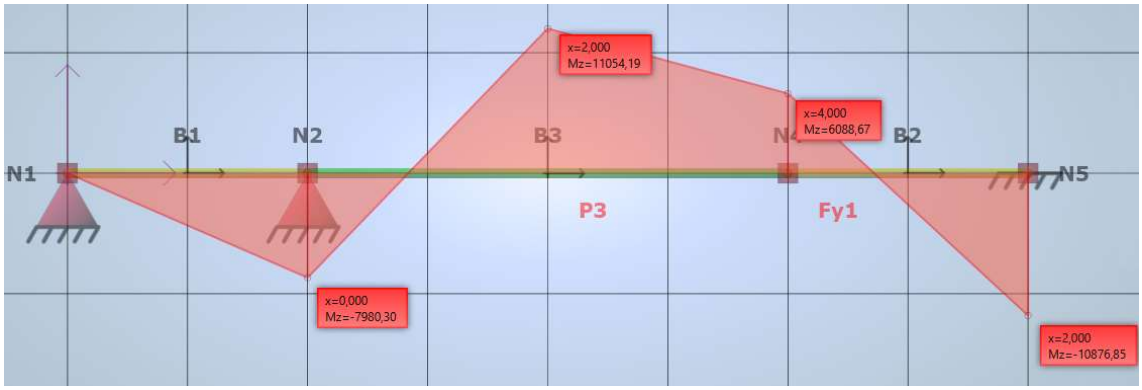

Figura 11: Resultados Ejemplo 3 con MDR-fx

Capítulo 4

SISTEMATIZACIÓN CON MATLAB

4. Sistematización con Matlab

Una vez comprobado la utilidad de la aplicación del método directo del cálculo plástico al cálculo de estructuras, el siguiente paso a realizar será la sistematización de este mediante un código utilizando Matlab. En el próximo capítulo se explicará detalladamente los pasos seguidos utilizando para ello, como ejemplo, el código utilizado para el ejemplo 1. Cabe decir que para el ejemplo 2 se utiliza el mismo código, pero utilizando las opciones de cargas uniformes que se indicarán igualmente a continuación.

4.1 Datos y geometría del problema

Para comenzar con la redacción de nuestro código, el primer paso será, como cabe de esperar, la introducción de los datos del problema de la manera en la que se muestra en la Figura 12: Código introducción de datos. Los datos introducidos corresponden con las características geométricas (Longitud), las características de la sección (Inercia y Sección), las características del material (Modulo de Young, Tensión de rotura y Módulo plástico) y las cargas utilizadas. En cuanto a las cargas, la carga “F” corresponde con el valor para las fuerzas puntuales requeridas y la carga “q” con el valor de las cargas uniformes.

```
% Datos
% Características geométricas
L=1.0;
% Características de la sección
Iz=8360.0*10^-8;
S=628.0*10^-6;
% Características del material
vE=2.1*10^11;
sigma_F=275*10^6;
Mp=S*sigma_F;
% Cargas
p=0*6000.0;
F=6000;
```

Figura 12: Código introducción de datos

A continuación de introducir los datos, para terminar de concretar el enunciado del problema, el siguiente paso será introducir la geometría del problema, determinando la posición de los nodos y las barras que discurren entre ellos.

```
% Geometria - Ejemplo 3
nodos = {{[0, 0], {'Fx', 'Fy', 'Mz'}}, % empotramiento
    {[0, 2*L], {0 ,0, 0}},
    {[2*L, 2*L], {0, -F, 0}},
    {[4*L, 2*L], {F, 0, 0}},
    {[4*L,0], {'Fx', 'Fy', 'Mz'}} % apoyo fijo
    }; % coordenadas / fuerzas
material = {{vE}}; % una lista por material
perfil = {{Iz, Mp}}; % una lista por tipo de perfil
barras = {{[1, 2], [1, 1], p},
    {[2, 3], [1, 1], p},
    {[3, 4], [1, 1], 0},
    {[4, 5], [1, 1], 0}
    }; % definicion, propiedades(material,perfil)

nd=length(nodos);
nb=length(barras);
```

Figura 13: Código introducción de geometría

Como se puede observar en la Figura 13: Código introducción de geometría los datos de la geometría se introducen en una serie de vectores que se dividen en:

- Vector “nodos”: Vector utilizado para la introducción de los datos de los puntos característicos necesarios para poder describir todos los aspectos del problema. Estos pueden ser los puntos de aplicación de cargas puntuales, los puntos de inicio y fin de una barra, los puntos de arriostramiento de la estructura, etcétera.
- Este elemento está formado por otros dos vectores internos en los que, en el primero, se introducen las coordenadas x e y del nodo, y, en el segundo, las fuerzas aplicadas sobre ese punto, tanto las reacciones (Fx, Fy, Mz) como las cargas puntuales aplicadas a la estructura.
- Vector “material”: Vector utilizado para la introducción de las características del material. Este elemento es un vector fila con tantas columnas como tipos de materiales existan en el problema.

- Vector “perfil”: Vector utilizado para la introducción de las características de la sección de los elementos. Este elemento es un vector fila con tantas columnas como número de secciones diferentes existan en el problema.
- Vector “barras”: Vector utilizado para la introducción de las características de los elementos que forman la estructura. Este elemento tendrá tantas filas como número de barras que existan y, dentro de cada fila, en primera posición se indican los nodos donde comienza y acaba cada barra y en segunda posición se representa las características de la barra correspondiente, haciendo referencia a los vectores “material” y “perfil”.
- Para un problema de cargas distribuidas se añadirá una tercera posición a cada barra, indicando que barra soporta una carga de este tipo e indicando su valor.

A mayores, también es interesante conocer el número de nodos y barras de las que está formado el problema, para ello, se utilizan las variables “nd”, para guardar en ella el número de nodos, y “nb”, para el número de barras.

4.2 Número de ecuaciones

Como ya sabemos, existen un número de ecuaciones necesarias, la cuales corresponden con las ecuaciones de equilibrio y de compatibilidad. El número de ecuaciones de equilibrio nos lo marca la Ec. 2 y el número de ecuaciones de compatibilidad corresponde con el grado de hiperestaticidad.

Entonces, para conocer el número de ecuaciones se ha redactado el siguiente código:

```
% numero de ecuaciones
NPR=nd;
R=0;
for i=1:nd
    kk1=nodos{i}{2}{1};
    kk2=nodos{i}{2}{2};
    kk3=nodos{i}{2}{3};
    if (kk1=='Fx')&(kk2=='Fy')&(kk3=='Mz')
        R=R+3;  % empotramiento
    elseif (kk1=='Fx')&(kk2=='Fy')&(kk3==0)
        R=R+2;    %apoyo fijo
        NPR=NPR-1;
    elseif (kk1=='Fx')&(kk3==0)
        R=R+1;    %apoyo móvil eje x
        NPR=NPR-1;
    elseif (kk2=='Fy')&(kk3==0)
        R=R+1;    %apoyo móvil eje y
        NPR=NPR-1;
    end
end

GH=(3*nb+R)-(3*nd+1);
EQ=NPR-GH;

fprintf('Ecuaciones de equilibrio = %d \n',EQ);
fprintf('Ecuaciones de compatibilidad = %d \n',GH);
```

Figura 14: Código número de ecuaciones

Los pasos seguidos han sido, en primer lugar, considerar que el número de secciones candidatas a plastificación (NPR) coincide con el número de nodos totales para que, posteriormente, restarle, si es necesario, el número de puntos en los que no aparecerían rótulas, como por ejemplo en un apoyo fijo. De esta manera ya se conoce el número NPR.

En cuanto al grado de hiperestaticidad, se calculará acorde a la Ec. 1. De antemano, ya se conocían el número de barras y el de nodos, por tanto, quedarían calcular el número de reacciones y de libertades internas. En cuanto a las libertades, para este problema son nulas. Para conocer el número de reacciones, el procedimiento seguido ha sido, mediante la variable "R" ir sumando el número de reacciones que añade cada tipo de apoyo (un empotramiento añade 3 reacciones, un apoyo fijo añade 2 reacciones y un

apoyo móvil 1 reacción). Finalmente se presentan por pantalla los resultados indicando el número necesario de ecuaciones de ambos tipos.

4.3 Giros y desplazamientos virtuales

Llegados a este punto, en primer lugar, se hará hincapié en calcular los giros y desplazamientos virtuales necesarios para aplicar el PDV, para obtener como resultado las ecuaciones de equilibrio. Aunque en este paso no se van a obtener como tales dichas ecuaciones, sino los datos necesarios para poder calcularlas posteriormente.

```
% Giros y desplazamientos virtuales de los nodos
dv=0.0001;
mt=zeros(nd,3,3); % matrices Ti
mmT=zeros(3,3); % matriz T
qi=zeros(EQ,nd); % giros virtuales
di=zeros(EQ,2,nd); % desplazamientos virtuales
Mi=ones(1,nd); % momentos reales
% mecanismo (MC1)
q0=[0 1 1 1 0];
options = optimoptions('fsolve','Display','iter','TolFun',1e-30,'TolX',1e-50);
[q,fun]=fsolve(@posicion,dv*q0,options);
qi(1,:)=q;
% desplazamiento de los nodos
di(1,:,:)=delta(q);
% mecanismo (MC2)
q0=[1 1 0 1 1];
[q,fun]=fsolve(@posicion,dv*q0,options);
qi(2,:)=q;
di(2,:,:)=delta(q);
```

Figura 15: Código giros y desplazamientos

Como podemos observar en la Figura 15: Código giros y desplazamientos se aplicará el procedimiento tantas veces como número de ecuaciones de equilibrio existan, en el caso que nos incumbe es de 2. Para la obtención de los giros virtuales, nos ayudaremos de la función "**posición**" (Figura 16: Función "posicion"). Se introducirá como dato la geometría del mecanismo de colapso mediante el vector "**q0**", con un 1 cuando exista rótula y un 0 cuando no,

multiplicada por un diferencial de giro (dv). Como resultado, se obtendrán los valores de los giros virtuales en cada nodo.

```
102     function v=posicion(q)
103 -       q=q.*q0;
104 -       kk=matrizT(q);
105 -       vP=cat(1,kk(1:2,3),kk(2,1));
106 -       v0=cat(2,nodos{nd}{1},0.0);
107 -       v=v0'-vP;
108 -   end
```

Figura 16: Función "posicion"

En cuanto a los desplazamientos virtuales, éstos se calcularán utilizando la función "**delta**" (Figura 17: Función "delta") a la cual se le introducen los giros virtuales y devuelve los desplazamientos que sufren cada nodo.

```
142     function v=delta(q)
143 -       v=zeros(2,nd);
144 -       for i=1:nd
145 -           kk1=diag(ones(1,3));
146 -           for j=1:i
147 -               kk1=kk1*reshape(mt(j,:,:),[3,3]);
148 -           end
149 -           kk=kk1(1:2,3);
150 -           kk2=nodos{i}{1}';
151 -           v(:,i)=kk-kk2;
152 -       end
153 -   end
```

Figura 17: Función "delta"

4.4 Momentos virtuales

Como ya se ha indicado, para la aplicación del PFV se necesitan un conjunto de momentos virtuales que cumplan las ecuaciones de equilibrio para cargas nulas. Aunque en este punto aún se desconocen dichas ecuaciones, mediante

los giros virtuales calculados en el punto anterior podemos obtener los valores para estos momentos.

```
% Momentos virtuales
mi=zeros(GH,nd); % momentos virtuales
m0=[1 0 0 0 1]; % EC1
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(1,:)=m;
m0=[1 0 0 0 0]; % EC2
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(2,:)=m;
m0=[0 0 1 0 0]; % EC3
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(3,:)=m;
```

Figura 18: Código momentos virtuales

Así, aplicando la función "**momentosVirtuales**" (Figura 19: Función "momentosVirtuales") se obtiene una matriz "**mi**" con el valor de los momentos virtuales en cada nodo para cada función de compatibilidad necesaria. A esta función solo basta con introducirle un vector aleatorio que trabaje como semilla.

```
function v=momentosVirtuales(m)
    v=zeros(1,EQ);
    q0=qi;
    for j=1:EQ
        q0(j,:)=q0(j,:)/max(q0(j,:));
        for i=1:nd
            v(j)=v(j)-m*(q0(j,:)');
        end
    end
end
```

Figura 19: Función "momentosVirtuales"

4.5 Resultados

Finalmente, habiendo obtenido todos los datos necesarios, el siguiente paso sería aplicar ambos principios. Para ello usamos la función "**ecuaciones**" (Figura 20: Función "ecuaciones"), mediante la cual se aplican ambos principios y se resuelven las incógnitas del problema. Para la aplicación de los principios se ha apoyado en la realización de otras dos funciones dentro de la propia función "**ecuaciones**", para el PDV corresponde con la función "**PDV**" y para el PFV corresponde con la función "**PFV**".

```
166     function v=ecuaciones(s)
167         M=s(:); % momentos reales
168         v=zeros(1,EQ+GH);
169         % aplicar el Principio de los Desplazamientos Virtuales (PDV)
170         v(1:EQ)=PDV(di); % ecuaciones de equilibrio
171         % aplicar PFV sistematico
172         v(EQ+1:end)=PFV(mi); % GH ecuaciones de compatibilidad
173         function v=PDV(di) % aplica el PDV -> obtener EQ ecuaciones de equilibrio
174             v=zeros(1,EQ);
175             for j=1:EQ
176                 desp=zeros(2,nd);
177                 desp=reshape(di(j,:,:),[2,nd]);
178                 for i=1:nd % nudos
179                     v(j)=v(j)-M(i)*qi(j,i);
180                     kk=nodos{i}{2};
181                     if kk{1}~='Fx'
182                      v(j)=v(j)+(desp(1,i)*kk{1}+desp(2,i)*kk{2});
183                     end
184                 end
185                 for i=1:nb % barras
186                     ij=barras{i}{1};
187                     Li=norm(nodos{i+1}{1}-nodos{i}{1});
188                     p0=barras{i}{3};
189                     a=desp(1,ij(1));
190                     b=(desp(1,ij(2))-desp(1,ij(1)))/Li;
191                     d=Li*(a+b*Li/2); % integral del desplazamiento
192                     v(j)=v(j)+p0*d;
193                 end
194             end
195         end
196         function v=PFV(mi) % aplica el PFV -> obtener GH ecuaciones de compatibilidad
197             v=zeros(1,GH);
198             for j=1:GH
199                 for i=1:nb
200                     ij=barras{i}{1};
201                     Li=norm(nodos{i+1}{1}-nodos{i}{1});
202                     vEi=material{barras{i}{2}(1)}{1};
203                     Izi=perfil{barras{i}{2}(2)}{1};
204                     Ma=M(ij(1)); Mb=M(ij(2));
205                     ma=mi(j,ij(1)); mb=mi(j,ij(2));
206                     p0=barras{i}{3};
207                     v(j)=v(j)+Li/(6*vEi*Izi)*(ma*(2*Ma+Mb)+mb*(Ma+2*Mb)+p0*Li^2*(ma+mb)/4);
208                 end
209                 for i=1:NPR
210                     v(j)=v(j)+mi(j,i)*q(i);
211                 end
212             end
213         end
214     end
215 end
```

Figura 20: Función "ecuaciones"

Una vez calculados los momentos en los puntos de la estructura correspondientes, como último paso será la presentación de los mismos en pantalla y así dar por resuelto nuestro problema.

```
92      % Resultados
93 -    s0=zeros(1,EQ+GH);
94 -    options = optimoptions('fsolve','Display','iter','TolFun',1e-30,'TolX',1e-50);
95 -    [s,fval]=fsolve(@ecuaciones,s0,options);
96 -    fprintf('Momento flector en "a": %5.2f \n',s(1));
97 -    fprintf('Momento flector en "b": %5.2f \n',s(2));
98 -    fprintf('Momento flector en "c": %5.2f \n',s(3));
99 -    fprintf('Momento flector en "d": %5.2f \n',s(4));
100 -   fprintf('Momento flector en "e": %5.2f \n',s(5));
```

Figura 21: Código resultados

Capítulo 5

VALIDACIÓN DEL CÓDIGO

5. Validación del código

Para verificar si el código funciona y si de él se obtienen los datos correctos basta con ejecutarlo y comparar resultados. Para ello, en este capítulo se presentarán los resultados obtenidos para los ejemplos utilizados en el capítulo 3, para los cuales, el código funciona a la perfección.

Los resultados se mostrarán en una tabla, en la que la columna "Método directo" se muestras los resultados obtenidos utilizando la metodología del capítulo 3, y en la columna "Código" se muestran los resultados obtenido mediante la ejecución del código.

- Ejemplo 1 (P=6000N y L=2m):

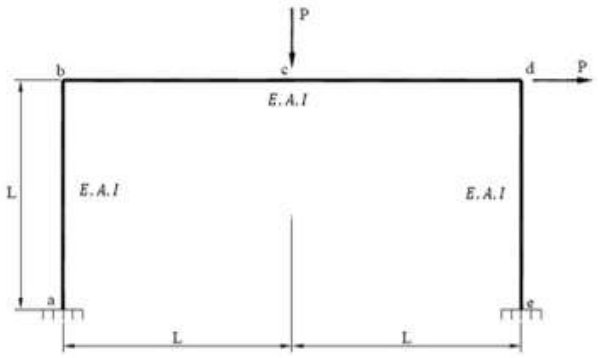

Figura 22: Resultados Matlab Ejemplo 1

Tabla 4: Resultados Ejemplo *1*

SECCIÖN	Método directo (Nm)	Código (Nm)
a	-2550	-2550.08
b	-150	-149.97
c	3600	3600.02
d	-4650	-4650.00

e	4950	4949.98

- Ejemplo 2 (q=6000Nm y L=1m):

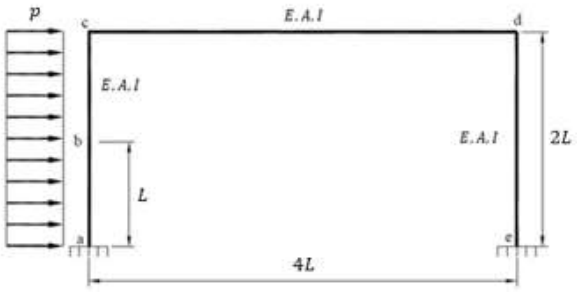

Figura 23: Resultados Matlab Ejemplo 2

Tabla 5: Resultados Ejemplo 2

SECCIÖN	Método directo (Nm)	Código (Nm)
a	-5900	-5899.98
b	700	700.00
c	1300	1299.99
d	-1700	-1700.00
e	3100	3100.0

- Ejemplo 3 (P=6000N y L=2m):

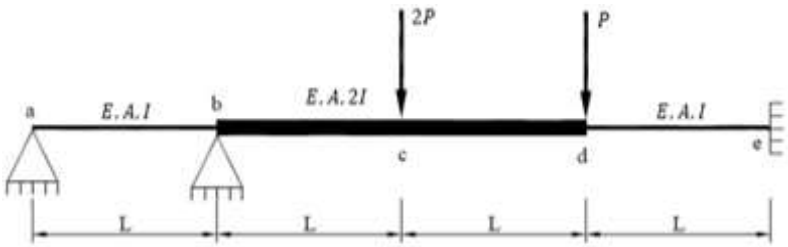

Figura 24: Resultados Matlab Ejemplo 2

Tabla 6: Resultados Ejemplo 3

SECCIÖN	Método directo (Nm)	Código (Nm)
a	0	0.00
b	-7980.3	-7980.30
c	11054.19	11054.19
d	6088.67	6088.67
e	-10876.85	-10876.85

5.1 Ejemplo aplicación

Una vez visto que la metodología es válida y que el código que se ha realizado funciona correctamente, en este apartado se aplicará el código a un ejemplo nuevo, correspondiente con un caso real. En este ejemplo se utilizará una geometría de pórtico a dos aguas al cual se le aplicaran varias cargas uniformes y puntuales.

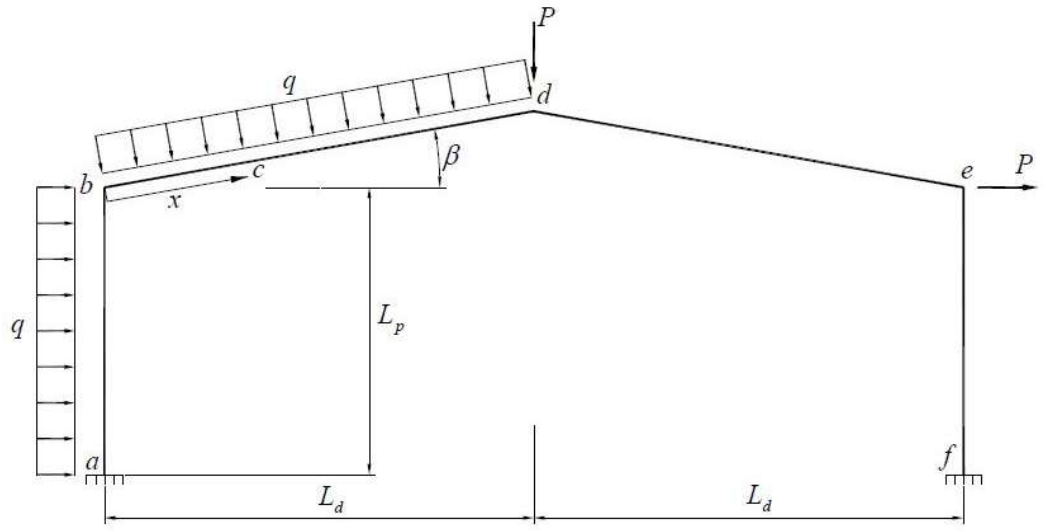

Figura 25: Ejemplo aplicación

La geometría se ha particularizado para los siguientes datos:

$$Lp = 4\ m$$
$$Ld = 6\ m$$
$$\beta = 10º$$
$$q = 1000\ Nm$$
$$F = 10000\ N$$

Una vez conocidos los datos del problema, se ha aplicado el código a este caso obteniendo los resultados representados en la siguiente tabla, en la que también se encuentran los resultados obtenidos tras aplicar el método directo.

Tabla 7: Resultados Ejemplo aplicación

SECCIÖN	Método directo (Nm)	Código (Nm)
a	-6473.40	-6473.39
b	-6368.70	-6368.71
c	-10264.00	-10264.03
d	8178.50	8178.47
e	17341.15	17341.15
F	-25138.40	-25138.39
g	30883.0	30883.00

Como paso último, se procederá a comprobar los resultados como se ha realizado hasta ahora, utilizando el programa informático MDR-fx, los cuales vienen representados en la Figura 26: Resultados MDR-fx Ejemplo adicional. Y como efectivamente, los resultados son correctos, ya nos encontramos en la disposición de dar por bueno el código realizado.

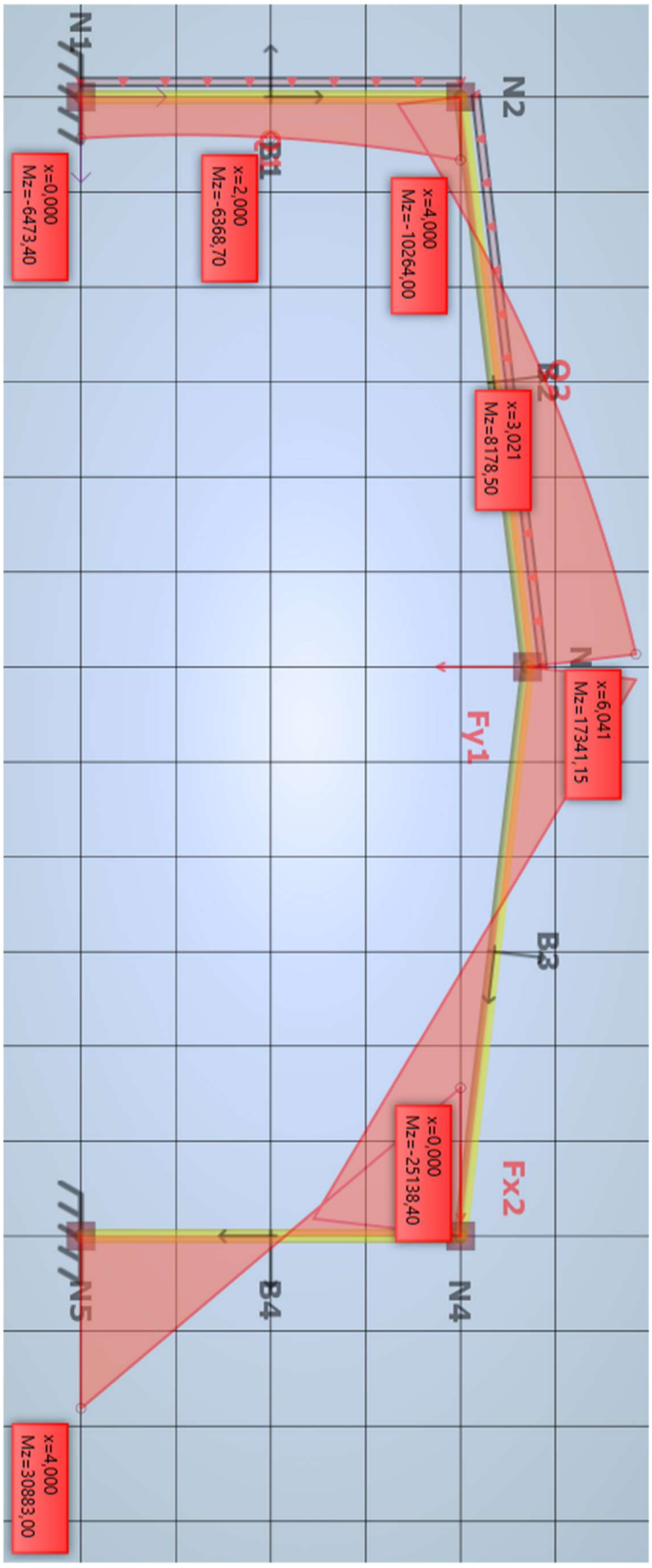

Figura 26: Resultados MDR-fx Ejemplo adicional

Capítulo 6

CONCLUSIONES

6. Conclusiones y líneas futuras

6.1 Conclusiones

Dando respuesta al objeto del proyecto, una vez realizados todos los cálculos y comprobaciones necesarias, se puede decir que la aplicación del método directo es factible para el cálculo lineal de estructuras. Así, la creación de un código para su aplicación supone ciertas ventajas e inconvenientes sobre otras formas de cálculo línea.

- Ventajas: Aplicar el método directo es una tarea menos tediosa comparada con otros métodos del cálculo de estructuras, como, por ejemplo, el Método Directo de Rigidez, a la hora de hacer los cálculos.

  Los Principios utilizado (PFV y PDV) son fáciles de aplicar ya que existen fórmulas predefinidas de las cuales, aplicándolas correctamente, se obtienen todas las ecuaciones necesarias.

  Y, en el caso de este trabajo, al sistematizar el proceso mediante Matlab, únicamente basta con correr el código para obtener la solución.
- Inconvenientes: Cómo resultado de la sistematización, únicamente se obtienen los momentos flectores en ciertas secciones. Para conocer todas las tensiones de la estructura sería necesario realizar un análisis a mayores, donde, conociendo el tipo de distribución de los momentos flectores (lineal, parabólico, etc.) se pueda aplicar equilibrios estáticos.

  Además, parte de este trabajo adicional sería calcular los desplazamientos de los distintos puntos (si fuera necesario).

  Por otro lado, el código no es del todo intuitivo, teniendo que cambiar varios datos para cada nuevo problema a resolver. Por ello, se necesita conocer que parámetros es necesario modificar y, sobre todo, teniendo en cuenta que muchos vectores se deben

rellenar a mano, que estos tengan las dimensiones correctas. Un ejemplo de ello corresponde con el número de nodos existentes, debiendo tener todos los vectores principales una longitud igual a ese número

6.2 Líneas futuras

Cómo se ha indicado dentro de los inconvenientes de esta metodología obtenemos los momentos de ciertas secciones elegidas y siempre se aplica a estructuras planas. De esta manera queda fuera de es Trabajo Fin de Grado:

- Los giros y desplazamientos de la estructura
- Los esfuerzos de cada elemento de la barra

Todo esto para resolver el problema de forma completa.

Bibliografía

[1] Área de Mecánica de Medios Continuos y Teoría de estructuras de la Universidad de Valladolid (2009): Apuntes para una breve introducción a la Resistencia de Materiales y temas relacionados. Valladolid (España)

[2] Basset Salom, Luisa. "El Principio de las Fuerzas Virtuales". Escuela Técnica Superior de Arquitectura (Univeristat Politécnica de Valéncia)

[3] Cervera Ruiz, Miguel/Blanco Díaz, Elena (2002). Mecánica de estructuras. Libro 2. Métodos de análisis. Universidad Politécnica de Cataluña (Barcelona, España)

[4] Gómez Carretero, Juan (2019). Cálculo plástico. Método directo sistemático mediante Teoría de Mecanismos. Universidad de Valladolid (Valladolid, España)

[5]Javier L. Mroginski, "UNIDAD 7:Principio de Trabajos Virtuales" [Disponible en: http://ing.unne.edu.ar/mecap/Apuntes/Estabilidad_1/TeoriaPTV.pdf]

[6] Mathworks. Centro de ayuda de Matlab [Disponible en: https://es.mathworks.com/help/index.html]

[7] "MDR-fx". Eduardo Tuñon Cabeza/José Pereda Llamas. Universidad de Valladolid.

[8] M.R.Dalmau/J.Vilardell (2003). Análisis plástico de estructuras. Introducción. Universidad Politécnica de Cataluña (Barcelona, España)

[9] William D. Callister, Jr. (1996). Introducción a la Ciencia e Ingeniería de Materiales. Editorial Reverte

ANEXOS

ANEXOS

Anexo 1. Matlab

MATLAB (abreviatura de ***MAT**rix **LAB**oratory*, «laboratorio de matrices») es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). La plataforma de MATLAB está optimizada para resolver problemas científicos y de ingeniería. El lenguaje de MATLAB, basado en matrices, es la forma más natural del mundo para expresar las matemáticas computacionales. Las gráficas integradas facilitan la visualización de los datos y la obtención de información a partir de ellos. Una vasta biblioteca de herramientas integradas permite empezar a trabajar inmediatamente con algoritmos esenciales para su dominio. El entorno de escritorio invita a experimentar, explorar y descubrir. Todas estas herramientas y funciones de MATLAB están probadas rigurosamente y diseñadas para trabajar juntas.

Las características principales son:

- Lenguaje de alto nivel para cálculos científicos y de ingeniería
- Entorno de escritorio optimizado para la exploración iterativa, el diseño y la solución de problemas
- Gráficas para visualizar datos y herramientas para crear diagramas personalizados
- Aplicaciones para ajustar curvas, clasificar datos, analizar señales, ajustar sistemas de control y muchas otras tareas
- Herramientas complementarias para una amplia variedad de aplicaciones científicas y de ingeniería
- Herramientas para crear aplicaciones con interfaces de usuario personalizadas
- Interfaces para C/C++, Java®, .NET, Python, SQL, Hadoop y Microsoft® Excel®
- Opciones de implementación libras de derechos para compartir programas de MATLAB con los usuarios finales

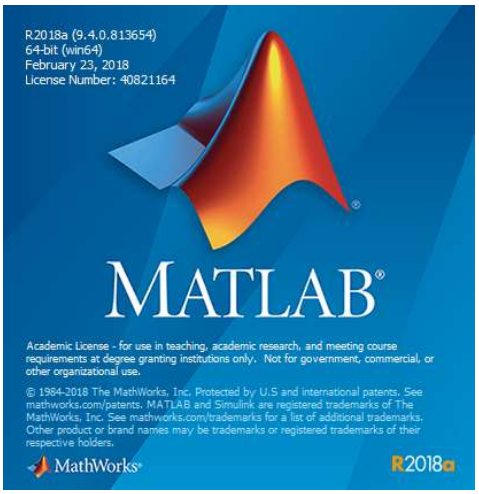

Figura 27: MATLAB

Anexo 1.1 Definición de comandos

Los comandos utilizados en los códigos realizados han sido:

- Bucles "**if**": evalúa una expresión y ejecuta un grupo de instrucciones cuando la expresión es verdadera. Una expresión es verdadera cuando su resultado no está vacío y contiene solo elementos no nulos, de lo contrario, la expresión es falsa.

  Los bloques "elseif" y "else" son opcionales. Las instrucciones se ejecutan solo si las expresiones anteriores del bloque "if...end" son falsas.

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

Figura 28: Bucle "if"

- Bucle "**for**": ejecuta un grupo de instrucciones en un bucle un número especificado de veces, utilizando una variable determinada. "Values" tiene una de las siguientes formas:
    - "initVal:endVal": incrementa la variable de "initVal" a "endVal" en 1 y repite la ejecución de "statements" hasta que la variable es superior a "endVal".
    - "initVal:step:endVal": incrementa la variable según el valor de salto en cada iteración o disminuye la variabale cuando el salto es negativo.
    - Existen varias formas, pero no nos incumben en nuestro problema

```
for index = values
   statements
end
```

Figura 29: Bucle "for"

- "**fsolve**": Resuelve el sistema de ecuaciones no lineales:
    - x = fsolve(fun,x0,options) resuelve las ecuaciones con las opciones de optimización especificadas en "Opciones". Se usa para establecer estas opciones "optimoptions"
    - [x,fval] = fsolve(___), para cualquier sintaxis, devuelve el valor de la función objetiva en la solución "funx"

- "**Optimoptions**": crea opciones optimizadas, por ejemplo, acotando el número máximo de iteraciones o el error que se puede cometer.

```
options = optimoptions(SolverName)
options = optimoptions(SolverName,Name,Value)
```

Figura 30: Sintaxis "optimoptions"

- "**cat**": concatena arrays.
    - C = cat (dim, A, B) concatena "B" al final de "A" a lo largo de la dimensión "dim" cuando "A" y "B" tienen tamaños compatibles (las longitudes de las dimensiones coinciden excepto para la dimensión operativa "dim")
    - C = cat (dim, A1, A2, …, An) concatena "A1", "A2", …,"An" una dimensión a lo largo de "dim

- "**Reshape**": El bloque "Reshape" cambia la dimensionalidad de la señal de entrada a una dimensionalidad que usted especifique, utilizando el parámetro de dimensionalidad de salida. Por ejemplo, puede usar el bloque para cambiar un vector de elemento N a una señal de matriz 1 por N o N por 1.

Anexo 2. Código

```
function CODIGO_PORTICOS
% Análisis Lineal
% Teoría de Mecanismos (TM)
% Carga distribuida (p) y puntual (F)

% Datos
% Características geométricas
L=1.0;
% Características de la sección
Iz=8360.0*10^-8;
S=628.0*10^-6;
% Características del material
vE=2.1*10^11;
sigma_F=275*10^6;
Mp=S*sigma_F;
% Cargas
p=0*6000.0;
F=6000;
% Geometría - Ejemplo 3
nodos = {{[0, 0], {'Fx', 'Fy', 'Mz'}}, % empotramiento
    {[0, L], {0 ,0, 0}},
    {[0, 2*L], {0, -F, 0}},
    {[4*L, 2*L], {F, 0, 0}},
    {[4*L,0], {'Fx', 'Fy', 'Mz'}} % apoyo fijo
    }; % coordenadas / fuerzas
material = {{vE}}; % una lista por material
perfil = {{Iz, Mp}}; % una lista por tipo de perfil
barras = {{[1, 2], [1, 1], p},
    {[2, 3], [1, 1], p},
    {[3, 4], [1, 1], 0},
    {[4, 5], [1, 1], 0}
    }; % definicion, propiedades(material,perfil)

nd=length(nodos);
nb=length(barras);
% número de ecuaciones de equilibrio
NPR=nd;
R=0;
```

```
for i=1:nd
    kk1=nodos{i}{2}{1};
    kk2=nodos{i}{2}{2};
    kk3=nodos{i}{2}{3};
    if (kk1=='Fx')&(kk2=='Fy')&(kk3=='Mz')
        R=R+3;  % empotramiento
    elseif (kk1=='Fx')&(kk2=='Fy')&(kk3==0)
        R=R+2;    %apoyo fijo
        NPR=NPR-1;
    elseif (kk1=='Fx')&(kk3==0)
        R=R+1;    %apoyo móvil eje x
        NPR=NPR-1;
    elseif (kk2=='Fy')&(kk3==0)
        R=R+1;    %apoyo móvil eje y
        NPR=NPR-1;
    end
end

GH=(3*nb+R)-(3*nd);
EQ=NPR-GH;

fprintf('Ecuaciones de equilibrio = %d \n',EQ);
fprintf('Ecuaciones de compatibilidad = %d \n',GH);
% Giros y desplazamientos virtuales de los nodos
dv=0.0001;
mt=zeros(nd,3,3); % matrices Ti
mmT=zeros(3,3); % matriz T
qi=zeros(EQ,nd); % giros virtuales
di=zeros(EQ,2,nd); % desplazamientos virtuales
Mi=ones(1,nd); % momentos reales
% mecanismo (MC1)
q0=[1 1 1 0 0];
options    =    optimoptions('fsolve','Display','iter','TolFun',1e-
30,'TolX',1e-50);
[q,fun]=fsolve(@posicion,dv*q0,options);
qi(1,:)=q;
% desplazamiento de los nodos
di(1,:,:)=delta(q);
% mecanismo (MC2)
q0=[1 0 1 1 1];
[q,fun]=fsolve(@posicion,dv*q0,options);
```

```
qi(2,:)=q;
di(2,:,:)=delta(q);
% Momentos virtuales
mi=zeros(GH,nd); % momentos virtuales
m0=[1 0 0 0 1]; % EC1
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(1,:)=m;
m0=[1 0 0 0 0]; % EC2
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(2,:)=m;
m0=[0 0 1 0 0]; % EC3
[m,fun]=fsolve(@momentosVirtuales,m0,options);
mi(3,:)=m;
% Resultados
s0=zeros(1,EQ+GH);
options    =    optimoptions('fsolve','Display','iter','TolFun',1e-
30,'TolX',1e-50);
[s,fval]=fsolve(@ecuaciones,s0,options);
fprintf('Momento flector en "a": %5.2f \n',s(1));
fprintf('Momento flector en "b": %5.2f \n',s(2));
fprintf('Momento flector en "c": %5.2f \n',s(3));
fprintf('Momento flector en "d": %5.2f \n',s(4));
fprintf('Momento flector en "e": %5.2f \n',s(5));


    function v=posicion(q)
       q=q.*q0;
       kk=matrizT(q);
       vP=cat(1,kk(1:2,3),kk(2,1));
       v0=cat(2,nodos{nd}{1},0.0);
       v=v0'-vP;
    end



    function v=matrizT(q)

        lx=0;
        ly=0;
        i=1;
        Li=norm(nodos{i+1}{1}-nodos{i}{1});
        v=mT(q(1),Li,lx,ly);
```

```
        mt(1,:,:)=v;

        for i=1:nb
            Li=norm(nodos{i+1}{1}-nodos{i}{1});
            kk=(nodos{i+1}{1}-nodos{i}{1})/Li;
            lx=kk(1);
            ly=kk(2);
            kk=mT(q(i+1),Li,lx,ly);
            mt(i+1,:,:)=kk;
            v=v*kk;

        end

        mmT=v;

        function v=mT(x,Li,lx,ly)
            v=[[1,-x,Li*lx]
                [x,1,Li*ly]
                [0,0,1]
                ];
        end

    end

    function v=delta(q)
        v=zeros(2,nd);
        for i=1:nd
            kk1=diag(ones(1,3));
            for j=1:i
                kk1=kk1*reshape(mt(j,:,:),[3,3]);
            end
            kk=kk1(1:2,3);
            kk2=nodos{i}{1}';
            v(:,i)=kk-kk2;
        end
    end

    function v=momentosVirtuales(m)
        v=zeros(1,EQ);
        q0=qi;
```

```
        for j=1:EQ
            q0(j,:)=q0(j,:)/max(q0(j,:));
            for i=1:nd
                v(j)=v(j)-m*(q0(j,:)');
            end
        end
    end

    function v=ecuaciones(s)
        M=s(:); % momentos reales
        v=zeros(1,EQ+GH);
        % aplicar el Principio de los Desplazamientos Virtuales
(PDV)
        v(1:EQ)=PDV(di); % ecuaciones de equilibrio
        % aplicar PFV sistematico
        v(EQ+1:end)=PFV(mi); % GH ecuaciones de compatibilidad
        function v=PDV(di) % aplica el PDV -> obtener EQ ecuaciones
de equilibrio
            v=zeros(1,EQ);
            for j=1:EQ
                desp=zeros(2,nd);
                desp=reshape(di(j,:,:),[2,nd]);
                for i=1:nd % nudos
                    v(j)=v(j)-M(i)*qi(j,i);
                    kk=nodos{i}{2};
                    if kk{1}~='Fx'
                     v(j)=v(j)+(desp(1,i)*kk{1}+desp(2,i)*kk{2});
                    end
                end
                for i=1:nb % barras
                    ij=barras{i}{1};
                    Li=norm(nodos{i+1}{1}-nodos{i}{1});
                    p0=barras{i}{3};
                    a=desp(1,ij(1));
                    b=(desp(1,ij(2))-desp(1,ij(1)))/Li;
                    d=Li*(a+b*Li/2); % integral del desplazamiento
                    v(j)=v(j)+p0*d;
                end
            end
        end
```

```
        function v=PFV(mi) % aplica el PFV -> obtener GH ecuaciones
de compatibilidad
            v=zeros(1,GH);
            for j=1:GH
                for i=1:nb
                    ij=barras{i}{1};
                    Li=norm(nodos{i+1}{1}-nodos{i}{1});
                    vEi=material{barras{i}{2}(1)}{1};
                    Izi=perfil{barras{i}{2}(2)}{1};
                    Ma=M(ij(1)); Mb=M(ij(2));
                    ma=mi(j,ij(1)); mb=mi(j,ij(2));
                    p0=barras{i}{3};

v(j)=v(j)+Li/(6*vEi*Izi)*(ma*(2*Ma+Mb)+mb*(Ma+2*Mb)+p0*Li^2*(ma+mb
)/4);
                end
                for i=1:NPR
                    v(j)=v(j)+mi(j,i)*q(i);
                end
            end
        end
    end
end
```