UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

TRABAJO DE GRADO

Grado en Ingeniería mecánica

DESARROLLO PROGRAMA DE ANALISIS MATRICIAL DE ESTRUCTURAS PARA EL ESTUDIO DE CHASIS FORMULA STUDENT

Volumen I

Memoria

Autor: Eric Oller Farias
Director: Daniel Di Capua
Departamento RMEE

Convocatoria: Enero 2017

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

RESUM

El projecte que es presenta en aquest document, té com a finalitat l'estudi del comportament en estat estàtic d'una estructura reticular d'un xassís monoplaça per a la competició Formula Student.

El principal objectiu és la creació d'un programa mitjançant Matlab, valgut per simular, optimitzar i valorar les diferents configuracions estructurals que l'usuari convingui.

El projecte està dividit en tres grans blocs. El primer es plantegen les equacions que governa el mètode matricial de rigidesa, un mètode de càlcul aplicable a estructures hiperestàtiques i reticulades de barres que es comporten de forma elàstica i lineal, aquest mètode està dissenyat per realitzar anàlisis computat.

Seguidament en el segon bloc s'expliquen la interacció necessària entre l'usuari i el programa per a resoldre i comprendre com es comporta l'estructura estudiada en vers a una simulació de rigidesa torsional hi ha una màxima sol·licitació de rigidesa en circuit i així poder donar pas al tercer bloc .

En aquest últim bloc i possiblement el més important s'implementa un model d'optimització capaç de re calcular totes les variables i possibles configuracions a la recerca d'un mínim pes i una resistència torsional adequada per a la nostra estructura reticulada, en aquest bloc s'estudiaran els resultats obtinguts perquè el lector tingui un millor enteniment del conjunt del programa.

La missió d'aquest treball és la d'unificar totes les modelitzacions matemàtiques de la realitat física que governa un vehicle de carreres en un sol programa (Matlab), perquè en un futur l'optimització global sigui un procés immediat. Aquest petit programa és l'inici d'alguna cosa més gran.

RESUMEN

El proyecto que se presenta en este documento, tiene como finalidad el estudio del comportamiento en estado estático de una estructura reticular de un chasis monoplaza para la competición *Formula Student*.

El principal objetivo es la creación de un programa mediante Matlab, valido para simular, optimizar y valorar las distintas configuraciones estructurales que el usuario convenga.

El proyecto está dividido en tres grandes bloques. El primero se plantean las ecuaciones que gobierna el **método matricial de la rigidez**, un método de cálculo aplicable a estructuras hiperestáticas y reticuladas de barras que se comportan de forma elástica y lineal, este método está diseñado para realizar análisis computarizado.

Seguidamente en el segundo bloque se explicara la interacción necesaria entre el usuario y el programa para resolver y comprender cómo se comporta la estructura estudiada en verso a una **simulación de rigidez torsional** y a una máxima solicitación de rigidez en circuito y así poder dar paso al tercer bloque.

En este último bloque y posiblemente el más importante se implementa un modelo de optimización capaz de recalcular todas las variables y posibles configuraciones en busca de un mínimo peso y una resistencia torsional adecuada para nuestra estructura reticulada, en este bloque se estudiarán los resultados obtenidos para que el lector tenga un mejor entendimiento del conjunto del programa.

La misión de este trabajo es la de unificar todas las modelizaciones matemáticas de la realidad física que gobierna un vehículo de carreras en un solo programa (Matlab), para que en un futuro la optimización global sea un proceso inmediato. Este pequeño programa es el inicio de algo más grande.

ABSTRACT

The project presented in this document, aims to study the static behavior of a reticular structure of a single-seat chassis for Formula Student competition.

The main objective is the creation of a program using Matlab, validated to simulate, optimize and evaluate the different structural configurations that the user need.

The project is divided into three large blocks. The first one considers the equations that governs the direct stiffness method, a method of calculation applicable to hyperesthetic structures and reticulated bars that behave elastically and linearly, this method is designed to perform computerized analysis.

Then in the second block will explain the interaction between the user and the program to solve and understand how the structure studied in verse behaves to a simulation of torsional rigidity and a maximum request of rigidity in circuit and thus be able to give way to the third block.

In this last block and possibly the most important one is implemented an optimization model capable of recalculating all the variables and possible configurations in order to search of a minimum weight and a suitable torsional resistance for our reticulated structure, in this block will be studied the results obtained so that The reader has a better understanding of the whole program.

The mission of this work is to unify all the mathematical modeling of physical reality that governs a racing vehicle in a single program (Matlab), so that in the future global optimization is an immediate process. This small program is the beginning of something bigger.

AGRADECIMIENTOS

La realización de este trabajo así como la etapa de aprendizaje para obtener el grado en ingeniería mecánica ha sido la etapa a nivel intelectual más enriquecedora que he tenido, obteniendo unas fuertes bases ingenieriles para el desarrollo de mi futuro profesional.

Aprovecharé esta oportunidad para agradecer a todas las personas que han hecho posible esta etapa.

En primer lugar, quisiera dar las gracias a mi Director del trabajo aquí presente, Daniel Di Capua. Le agradezco que me introduzca a este complejo mundo y que me diera la libertad de poder expresar mis ideas en este trabajo. A tantos otros profesores que gracias a su profesionalidad me han hecho crecer como ingeniero, un profundo agradecimiento a todos ellos.

A mis compañeros y miembros del equipo e-Tech Racing, a los que se esforzaron y sacrificaron por el equipo y los que no pudieron con el ritmo y presión, a todos ellos porque me han mostrado distintos puntos de vista de la profesión que tanto amo. A ellos mis más sinceros agradecimientos ya que si ellos este trabajo no tendría sentido.

Llegados a este punto, quisiera agradecer a mis compañeros de estudio, Jordi, Ezequiel, Jorge, David. Por los grandes momentos que tuvimos juntos. Siempre recordaré el importante papel que han jugado en este viaje

Por último, pero no menos importante, me gustaría dar un enorme "Gracias" a mis padres José y Amalia y a mi hermano Héctor, ya que sin su ayuda no podría haber llegado a la meta.

También quería agradecer el incondicional apoyo a mi novia Laura, de ella he aprendido en estos años el valor de la disciplina y fuerza de voluntad, juntos empezamos este viaje y juntos lo hemos completado.

Le dedico este trabajo a mi amigo eterno, mi gran amigo Nacho, sé que él creía en mi ciegamente y a él van dedicados todos mis éxitos.

¡Gracias a todos y a cada uno de vosotros!

GLOSSARIO

Esta lista tiene como objeto especificar conceptos anglosajones utilizados en la Formula Student. Es por ello, que en la memoria se utilizará dichos conceptos en forma inglesa y no en su forma castellana. Se pasa a describir cada uno de estos conceptos:

- Main Hoop *"Arco principal"*: Arco antivuelco situado lateralmente o posterior al torso del conductor. Main Hoop es abreviado como *MH*.
- Front Hoop *"Arco intermedio"*: Arco antivuelco situado por encima de las piernas del piloto próximo al volante. Front Hoop es abreviado como *FH*.
- Front Bulkhead *"Arco frontal"*: Arco cuya función es proteger las piernas del piloto. Front Bulkhead es abreviado como *FB*.
- Impact Attenuator *"Atenuador de impactos"*: Dispositivo deformable que absorbe energía localizado en la Front Bulkhead.
- Side Impact Structure *"Estructura de impacto lateral"*: Estructura cuya función es proteger de impactos laterales al piloto.
- Head Restraint *"Reposacabezas"*: Elemento cuya función es limitar el movimiento hacia atrás de la cabeza del conductor.
- *Shoulder harness "Barra de cinturones"*: Barra en la cual se anclan los cinturones de seguridad.
- Set Up “Configuración de prestaciones”: Es el conjunto de ajustes hechos al vehículo para optimizar su comportamiento.

ÍNDICE DE MEMORIA

CAPÍTULO 1: INTRODUCCIÓN

1.1. Origen del proyecto. EUETIB e-Tech Racing

Este trabajo surge de una idea y un objetivo, basado en la creación de un equipo (formado únicamente por estudiantes de ingeniería) con el objetivo de diseñar y construir un monoplaza eléctrico capaz de participar en pruebas de *Formula Student*.

El proyecto es emprendido por un grupo de 30 estudiantes de media de la *Escola Universitària d'Enginyeria Tècnica Industrial de Barcelona* (EUETIB), de las especialidades de Ingeniería Mecánica, Eléctrica y Electrónica. La mayoría sin ninguna experiencia en el campo de la competición automovilística.

El equipo EUETIB e-Tech Racing nace en septiembre de 2012 e inicia una fase de aprendizaje y adquisición de conocimientos de dos años, hasta presentarse a las pruebas estáticas de *Formula Student Spain* 2014 con el monoplaza denominado "E79", figura 1.1.

Tras una primera toma de contacto con la competición, comienza una etapa de rediseño y construcción del monoplaza (denominado "Will-e", figura 1.2), donde se presenta a las pruebas estáticas y dinámicas de la edición 2015 de *Formula Student Spain.* A pesar de conseguir un monoplaza 100% funcional, por problemas técnicos no pudo disputar las pruebas dinámicas.

Tras la experiencia de dos ediciones, comienza la optimización de todas las partes del monoplaza, convirtiéndolo en otro monoplaza totalmente diferente (denominado “EV-a”, figura 1.3), donde se presentó a las pruebas estáticas y dinámicas de la edición 2016 de *Formula Student Spain.*

Figura 1.1 *E79, edición 2014 [1]*

Figura 1.2 *Will-e , edición 2015 [1]*

Figura 1.3 *Ev-a , edición 2016 [1]*

1.2. Motivación

El autor de este trabajo es miembro de EUETIB e-Tech Racing desde poco después de su creación en 2013, ha sido Team Leader del mismo, fruto de la pasión por el mundo de la automoción y de la ingeniería en concreto el mundo que rodea el "*Motorsport.*

Estos, son los causantes de la ilusión y el afán de conocimiento que comporta liderar un proyecto de este tipo.

La inmensa satisfacción de poder trabajar en un proyecto de tal magnitud y el hecho de formar parte de un equipo de personas de estas características, donde prima la interacción e intercambio tanto de información como de conocimientos, hace que represente otra de las razones de peso para la realización de este proyecto.

Además, la posibilidad de poder aplicar conocimientos aprendidos a lo largo de la formación universitaria, como el diseño de componentes y elementos de máquinas y mecanismos, el trato con componentes sometidos a esfuerzos (ya sean estáticos, dinámicos, sujetos a fatiga, etc.) y la elección de materiales específicos para cada aplicación, es la motivación definitiva para emprender un proceso como este.

- Aprender a trabajar en colaboración con un equipo grande de personas. Ser capaces de imponer el criterio de uno mismo cuando sea necesario, y ceder este derecho a los demás cuando sea oportuno.

- Conocer y formar parte del mundo de la *Formula Studen*t y en general el mundo de la competición en el sector de la automoción.

- Comprender y ser capaces de predecir al máximo nivel posible el comportamiento dinámico de un vehículo capaz de experimentar todas las situaciones que abarca la automoción de competición.

- Aprender a usar softwares de gran complejidad relacionados con el diseño asistido por ordenador (CAD) y la simulación por elementos finitos (SolidWorks® y ANSYS®), así como otros relacionados con la programación (MATLAB® y SIMULINK®).

- Realizar una primera toma de contacto con la ingeniería y la ciencia relacionada con los neumáticos, estructuras reticuladas, dinámica vehicular, adquisición de datos, organización… y ser capaces de analizar y optimizar en cierta medida su comportamiento.

- Diseño fiable y óptimo de componentes específicos para determinadas funciones. Diseño del ensamblaje del conjunto de los componentes que conforman el sistema de suspensión. Creación de los planos correspondientes a dichas partes y ensamblajes, para su posterior fabricación.

- Construir el vehículo en sí y legar el trabajo bien hecho a posteriores miembros para que las futuras mejoras devengan más deprisa

1.3. Objetivos

Los principales objetivos planteados se muestran a continuación:

- Conocer y formar parte del mundo de la Formula Student y en general el mundo de la competición del sector de la automoción.
- Conseguir la mayor rigidez torsional del chasis con la menor masa posible. Estudiando un buen valor de referencia para un óptimo funcionamiento dinámico del monoplaza.
- Reducir la masa del chasis respecto al Ev-a de la edición 2016.
- Aumentar la destreza en software de gran complejidad relacionados con el diseño asistido por ordenador tales como MATLAB® y como otros progamas (CAD), simulación por elementos finitos en SolidWorks® y ANSYS®.
- Construir un vehículo diseñado y optimizado para someterlo a todo tipo de pruebas previas a su presentación oficial en la Formula Student.
- Obtener la máxima puntuación en pruebas estáticas de la edición 2016 de Formula Student, y finalizar todas las pruebas dinámicas con un resultado satisfactorio.

CAPÍTULO 2: ENTORNO

1.4. Formula Student

La *Formula Student* es una competición interuniversitaria, en la que participan equipos de todo el mundo, formados exclusivamente por estudiantes de los centros implicados. Los equipos participantes compiten con un monoplaza diseñado y construido por los integrantes de cada equipo, y son evaluados tanto el diseño de dicho vehículo, como los resultados obtenidos en las pruebas realizadas el día de la competición.

La finalidad de la competición es fomentar el aprendizaje autónomo de los estudiantes involucrados, y así mejorar la formación de los mismos. En el desarrollo del proyecto, el estudiante pone a prueba los conocimientos adquiridos en una gran diversidad de campos, al aplicarlos en el diseño del monoplaza.

Los fondos necesarios para realizar el proyecto suelen ser obtenidos de empresas patrocinadoras e inversores interesados en colaborar en él. Además, ciertas ramas de la evaluación del diseño están relacionadas con la economía y el equipo como empresa. Es por ello que también se requieren i aplican conocimientos de marketing, análisis de costos o estrategias de negocio.

1.4.1. *Historia*

El programa tiene sus orígenes en América bajo el nombre de *Formula SAE*. La *Society of Automotive Engineering* (SAE) comienza en 1981 un proyecto denominado *Formula SAE* (como evolución a la competición *BAJA SAE*), en la Universidad de Texas, Austin, en el que 6 equipos formados por estudiantes del centro compiten con monoplazas autónomamente diseñados. La razón de la creación de este evento es la de proporcionar a las empresas del sector de la automoción y del mundo de la ingeniera una cantera de estudiantes altamente preparados para el mundo laboral, donde estos requieren menos recursos de la empresa para poder ser rentables. Siendo esta una de las razones principales por las que grandes empresas esponsorizan a los equipos.

El proyecto evoluciona considerablemente, y en 1998 atraviesa el atlántico y al crearse la competición *Formula Student* (con la colaboración del *Institute of mechanical engineers*) en Warwickshire y con la participación de 4 equipos de la región. Hoy en día existen 12 eventos de este tipo celebrados en distintos lugares del planeta, llegando estos a agrupar hasta 120 equipos y más de 2000 estudiantes, en algunos casos. En la imagen 2.1 se muestran los eventos geográficamente.

Figura 2.1 *Localización geográfica de los eventos Formula Student*

1.4.2. *Estructura del evento*

La competición está dividida en dos tipos de pruebas: pruebas estáticas y pruebas dinámicas. En las primeras se evalúa el diseño del monoplaza, el plan de negocio y el estudio económico, mientras en las segundas se evalúa el comportamiento en pista.

La evaluación de todas las partes de la competición se lleva a cabo por jueces especialistas en cada ámbito específico, con experiencia en la automoción de competición así como en el mundo de la *Formula Student*.

A continuación, se describen ambas fases de la evaluación con detalle.

- Pruebas estáticas (Static events):

  En las pruebas que constituyen este grupo, se evalúa el diseño del monoplaza, así como el coste, el plan de negocio y la inspección técnica del mismo.

  La puntuación máxima que se puede obtener es de 325 puntos, repartidos según se muestra en la tabla 2.1:

Prueba	**Puntuación máxima**
Inspección Técnica	No recibe puntos
Coste y construcción	100 puntos
Presentación	75 puntos
Diseño	150 puntos
Total	**325 puntos**

Tabla 2.1 *Tabla de puntuación de pruebas estáticas [1]*

- **Inspección técnica:**

En este apartado se determina si el vehículo cumple con la normativa FSAE que rige la competición. Esta prueba no proporciona puntuación al equipo, aunque es imprescindible superarla para poder ser evaluado en las pruebas dinámicas.
Se realizan distintas pruebas para determinar el cumplimiento de la normativa:

Parte 1: Revisión eléctrica y mecánica

Comprobación eléctrica y mecánica del vehículo, según la norma FSAE.

(Norma S2.7.1 "*Electrical and Mechanical Scrutineering*"
2015 Formula SAE® Rules) *[1]*

Parte 2: Prueba de inclinación

El monoplaza debe ser capaz de inclinarse 45º sin perder combustible ni ningún tipo de fluido, y 60º sin volcar.

(Norma S2.7.2 "*Tilt table test*" 2015 Formula SAE® Rules) *[1]*

Parte 3: Prueba de ruido, interruptor principal, test de lluvia y prueba de frenada.

Según el tipo de vehículo se realizaran las pruebas pertinentes de esta parte de la evaluación.

(Norma S2.7.2 "*Noise, Master Switch, Ready-To-Drive-Sound, Rain Test and Brake Test*" 2015 Formula SAE® Rules) *[1]*

- **Coste y construcción:**

La evaluación de esta parte se basa en la justificación de los costes que implica el proyecto.

Los equipos tienen que saber encontrar un balance adecuado entre el diseño de cada parte del vehículo, y el coste que éste comporta.

Los participantes deben aprender y comprender los principios básicos del diseño, fabricación y ensamblaje, con mínimas restricciones y una normativa que los rija durante el proceso.

Esta fase de evaluación consta de tres partes:

Parte 1: Informe de costes

Se debe presentar un informe de costes, y entregarlo a los jueces responsables antes de la celebración del evento. Este informe incluye materiales, componentes y procesos aplicados durante la construcción.

(Norma S4.3.1 "*Cost Report*" 2015 Formula SAE® Rules) *[1]*

Parte 2: Debate

En el día de evaluación, ya empezada la competición, se lleva a cabo un debate con los jueces alrededor del monoplaza, evaluando tanto el coste del proyecto como la capacidad del equipo para preparar estimaciones precisas de dichos costes.

(Norma S4.3.2 "*Discussion*" 2015 Formula SAE® Rules) *[1]*

Parte 3: Caso real

Los estudiantes deben enfrentarse a una hipótesis de caso real, relacionado con el coste y la fabricación del vehículo.

(Norma S4.3.3 "*Real Case*" 2015 Formula SAE® Rules) *[1]*

- **Presentación (Plan de negocio)**

En esta parte se evalúa la capacidad del equipo para desarrollar y ofrecer un modelo de negocio, y con ello convencer a los ejecutivos de una corporación de que el proyecto es el que mejor se adapta a sus intereses y a las exigencias de los clientes potenciales (mercado de competiciones de fin de semana, aficionados al *Motorsport*...). El negocio debe ser rentable y debe ser expuesto como tal.

(Norma S5 "*Presentation event*" 2015 Formula SAE® Rules) *[1]*

- **Diseño**

Esta parte dela evaluación considera el valor del diseño llevado a cabo por el equipo, y según criterios de ingeniería.

En función de la valoración obtenida y el grado en el que se superen los objetivos de diseño, se otorgarán más o menos puntos al equipo.

(Norma S6 "*Design event*" 2015 Formula SAE® Rules) *[1]*

- **Pruebas dinámicas (Dynamic events):**

Una vez superada la Inspección Técnica del vehículo y habiendo participado en las pruebas estáticas, el monoplaza puede ser evaluado en las pruebas dinámicas. En este evento, distintos aspectos del comportamiento dinámico del vehículo son puestos a prueba en las diferentes partes que lo componen.

Con un máximo de 675 puntos obtenibles, esta fase de la competición se estructura tal y como se muestra en la tabla 2.2.

Prueba	**Puntuación máxima**
Aceleración	75 puntos
Skid Pad	50 puntos
Autocross	150 puntos
Eficiencia	100 puntos
Resistencia	300 puntos
Total	**675 puntos**

Tabla 2.2 *Tabla de puntuación de pruebas dinámicas [1]*

- **Aceleración**

Este evento evalúa la aceleración longitudinal máxima que del monoplaza es capaz de desarrollar e línea recta y pavimento plano. El vehículo dispone de un tramo de 75m de longitud (entre línea de salida y línea de llegada), el cual debe ser recorrido en el menor tiempo posible.

La puntuación obtenida en esta prueba se corresponde con la expresión matemática siguiente (Ec. 2.1):

$$Puntuación\ Aceleración = \frac{71{,}5 \cdot \frac{T_{max}}{T_{your}} - 1}{\frac{T_{max}}{T_{min}} - 1} + 3{,}5 \qquad (2.1)$$

Dónde:

T_{your}: $Menor\ tiempo\ corregido\ por\ sanciones$

T_{min}: $Menor\ tiempo\ corregido\ por\ sanciones\ conseguido\ por\ el\ monoplaza\ más\ rápido$

: $150\%\ de\ T_{min}$

(Norma D5 "*Acceleration event*" 2015 Formula SAE® Rules) *[1]*

- **Skid-Pad:**

Esta prueba tiene la finalidad de evaluar el comportamiento en curva (pavimento llano y radio de curva constante), con las aceleraciones laterales que este comporta.

La forma y dimensiones del circuito están impuestos por normativa: Dos tramos circulares situados en forma de ocho. El diámetro interior de cada círculo es de 15,25m, y el exterior 21,25m. Los centros de ambos círculos se encuentran a 18,25m el uno del otro. El ancho del camino queda de 3m en todos los puntos del recorrido. La entrada y la salida del circuito son tramos rectos (de 3m de ancho) tangentes a ambos círculos.

En la figura 2.1 se halla representado gráficamente el circuito arriba descrito:

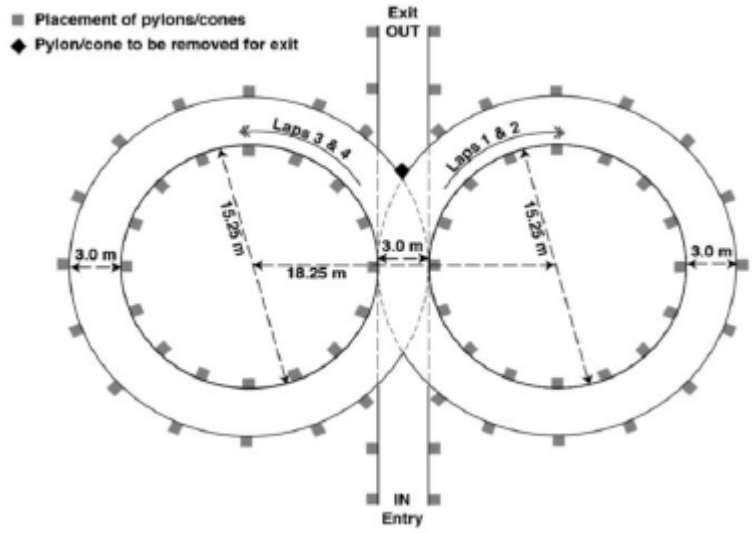


Figura 2.2 *Recorrido del Skid-Pad de Formula Student [1]*

La puntuación obtenida en esta prueba se corresponde con la expresión matemática siguiente (Ec. 2.2):

$$Puntuación\ Skid-Pad = \frac{47{,}5\cdot\left(\frac{T_{max}}{T_{your}}\right)^2-1}{\left(\frac{T_{max}}{T_{min}}\right)^2-1}+2{,}5 \qquad (2.2)$$

Dónde:

T_{your}: *Menor promedio de la vuelta a derechas e izquierdas corregido por sanciones*

T_{min}: *Menor tiempo corregido por sanciones conseguido por el monoplaza más rápido*

: $125\%\ de\ T_{min}$

(Norma D6 "*Skid-Pad event*" 2015 Formula SAE® Rules) *[1]*

- **Autocross:**

Esta prueba tiene como objetivo evaluar el comportamiento en circuito, sin considerar otros vehículos como estorbo. Se realiza en un circuito de aproximadamente 1km de longitud, que combina condiciones de aceleración, frenada y paso por curva.

La puntuación obtenida en esta prueba se corresponde con la expresión matemática siguiente (Ec. 2.3):

$$Puntuación\ Autocross = \frac{142{,}5 \cdot \frac{T_{max}}{T_{your}} - 1}{\frac{T_{max}}{T_{min}} - 1} + 7{,}5 \qquad (2.3)$$

Dónde:

T_{your}: $Mejor\ tiempo\ corregido\ por\ sanciones$

T_{min}: $Mejor\ tiempo\ corregido\ por\ sanciones\ conseguido\ por\ el\ monoplaza\ más\ rápido$

: $145\%\ de\ T_{min}$

(Norma D7 "*Autocross event*" 2015 Formula SAE® Rules) *[1]*

- **Resistencia:**

Es posiblemente la prueba más exigente de la competición y la de más peso (representa 300 puntos de los 675 totales de pruebas dinámicas). Tiene como finalidad evaluar el rendimiento general del monoplaza, siendo de capital importancia la durabilidad y fiabilidad del mismo.

El monoplaza debe ser capaz de realizar un total de 22 vueltas a un circuito de 1km de longitud. Aunque el tiempo es importante en esta prueba, hay una gran diversidad de temas a tener en cuenta para la puntuación final (por ejemplo, el vehículo no debe derramar fluidos de ningún tipo durante la prueba).

La puntuación obtenida en esta prueba se corresponde con la expresión matemática siguiente (Ec. 2.4):

$$Puntuación\ Resistencia = \frac{250 \cdot \frac{T_{max}}{T_{your}} - 1}{\frac{T_{max}}{T_{min}} - 1} + 50 \qquad (2.4)$$

Si un equipo no consigue un tiempo inferior a T_{max}, no recibirá puntuación en la prueba.

(Norma D8.20 "*Endurance Scoring Formula*" 2015 Formula SAE® Rules) *[1]*

- **Eficiencia:**

En esta prueba se considera la eficiencia del vehículo en condiciones normales de carrera. Su evaluación se lleva a cabo durante la prueba de resistencia previamente comentada.

(Norma D8.5 "*Efficiency*" 2015 Formula SAE® Rules) *[1]*

CAPÍTULO 3: GEOMETRÍA DEL CHASIS

1.5. Introducción

El chasis, o en opinión del autor, la estructura del vehículo, ya que en el entorno de *motorsport*, la palabra chasis está más focalizado en la interacción de conjuntos de elementos tales como, suspensión, neumáticos y dirección. Consiste en una estructura interna que sostiene, une todos los elementos del vehículo y aporta la rigidez necesaria para afrontar unas deformaciones mínimas para un correcto funcionamiento del vehículo. Se podría hacer la analogía al esqueleto de un animal

Es un elemento de principal importancia, ya que posee dos grandes variables que los fabricantes de vehículos deben tener en cuenta, el peso en términos de eficiencia energética o "performance", aquí es donde entra también la rigidez, pero esta relacionada con el peso, y el "packaging" que elimina un grado de libertad al diseño dificultando así la optimización de este, a causa de la extrema interacción con todos los elementos que constituyen un vehículo.

Desde los primeros modelos de vehículos, los conceptos y geometrías de los chasis han ido evolucionando a medida que la tecnología, la competencia y la sociedad han demandado mayor prestación sin sacrificar la economía. Por ello es necesario que entendamos que en la actualidad tanto a nivel de usuario utilitario como a nivel de competición automovilística existen varias tipologías de chasis.

1.5.1. *Monocascos CFRP / autoportante*

Se denominan monocasco o carrocería autoportante a los vehículos que incluyen el chasis y el habitáculo de componentes y de pasajeros en una sola pieza con punteras que sirven de soporte al motor. Este sistema es el usado en casi la totalidad de los turismos desde los años 1980. El primer automóvil en incorporar esta técnica junto a CFRP "*Fibras de Carbono Reforzadas por Polímero*" en competición viene de la mano de McLaren-Honda y su MP4/1 de 1992

Imagen 2.1 *Monocasco integro MP4/1 McLaren-Honda*

Esta tipología de chasis es la utilizada en alta competición, tales como F1, Formula e, LP1, DTM… Es así ya que aporta una inmejorable relación entre el peso y la rigidez que puede proporcionar, llegando a valores de rigidez inalcanzares por otro tipo de tecnologías. Las desventajas más claras de esta tecnología son que al tratarse de un material altamente anisótropico es necesaria la presencia de expertos en la materia para su diseño y optimización así como expertos en la fabricación de los moldes y utillajes necesarios para su construcción. Esto hace que el precio de un chasis monocasco multiplique por más de diez veces a otro tipo de tecnologías utilizadas para el mismo uso.

En turismos utilitarios hoy en día casi todos los automóviles se construyen con la técnica de monocasco, realizándose las uniones entre las distintas piezas mediante soldadura de punto. Existen vehículos en los cuales hasta los cristales forman parte de sus estructuras, brindando fortaleza y rigidez a todo el conjunto.

1.5.2. *Multitubulares*

La carrocería tubular o multitubular es un tipo de estructura utilizada pioneramente en la fabricación de aeronaves, mayormente del tipo aviones, ya que ofrece la gran ventaja de minimizar el peso si dejar de lado la rigidez que necesita una máquina de esas características. Se implementó en la automoción, de la mano de los vehículos deportivos de mediados del siglo XX y por los grupos B de los años 80.

Esta técnica utiliza como estructura del vehículo una red de finos tubos metálicos soldados, recubierta después con láminas metálicas, frecuentemente para aumentar la rigidez sin aumentar considerablemente el peso de la estructura

Tiene como ventajas conseguir una estructura de gran rigidez y resistencia con muy poco peso. Por otra parte, la fabricación es muy laboriosa lo que para un prototipo puede aumentar considerablemente los costes, pero no tanto como las de monocascos CFRP.

La técnica todavía se utiliza en modelos deportivos hechos a mano como es el caso de este proyecto documentado.

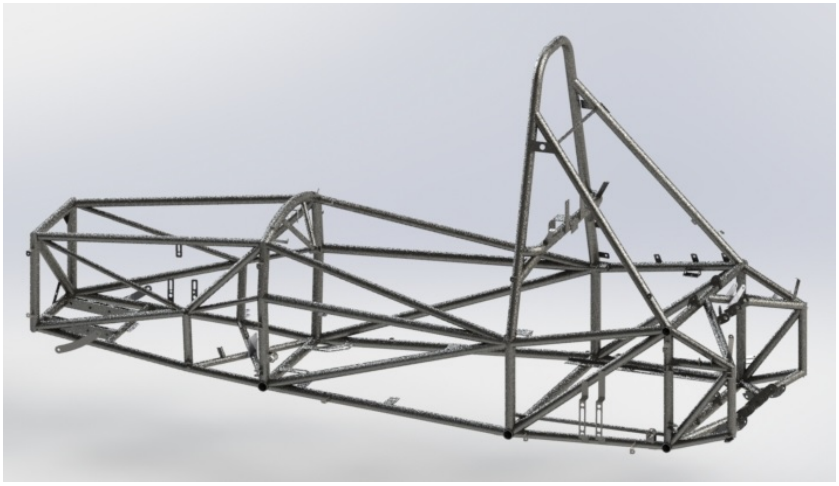


Imagen 2.2 *Estructura multitubular del EV-a, Formula Student EUETIB e-Tech Racing*

Existen muchas más tipos de tipologías en automoción, pero para el caso que nos atañe, en Formula Student estos son los dos grandes grupos existentes. Se puede encontrar un tercer grupo, pero o es más que la unión o mixta de ambos

En este trabajo se escoge la opción de conceptuar, diseñar, construir y optimizar una estructura multitubular, no solo porque para un equipo joven y con bajos recursos económicos, sino porque pesa muchísimo más la importancia de poder crear un modelo matemático tal como el método directo de rigidez que refleje la realidad física, y así poder optimizar la estructura y validarla mediante galgas extensimétricas u otras técnicas, llegando así al estudio más profundo de este tipo de elementos.

1.6. La normativa

La normativa, las leyes, las normas de calidad, son las reglas increbrantables que cualquier diseñador debe tener en cuenta para que un concepto llegue a buen fin, es decir, que pueda ser utilizado tanto en la vía pública como en un circuito de carreras, este último será nuestro caso.

En Formula Student existe una extensísima normativa focalizada en la seguridad de más de 170 páginas, algunas de estas normas ya las hemos visto en apartados anteriores, pero estas son afectadas indirectamente por el diseño del monoplaza. A continuación mostraremos parte de la normativa que atañe única y exclusivamente al diseño del chasis. Los demás hándicaps de fabricación, como la geometría de suspensión y el pakaging de los demás elementos, no se trataran en esta sección.

Cabe recalcar que el diseño optimo, a través de un algoritmo matemático, queda seriamente restringido por la norma que veremos a continuación, pero como no, aun con estas restricciones normativas se puede optimizar el sistema. Siendo este uno de los objetivos principales de este trabajo.

Para empezar, la normativa que se debe seguir es la normativa que nace en Formula SAE®. Esta normativa permite que haya competitividad entre equipos, y es adoptada por todas las competiciones, pudiendo estos variar algunos aspectos de la misma para adaptarse a las necesidades del país o región. Como es el caso de la Formula Student Germany, que modifica aspectos como la potencia máxima eléctrica que pueden suministrar los grupos propulsores.

Dicha normativa está dividida en siete diferentes partes, estas partes se dividen en tres grandes grupos y reflejan tanto el diseño, como la manera que se puntúan las distintas disciplinas y los requisitos mínimos e indispensables que debe cumplir un equipo para poder participar en la competición. De los tres grandes grupos solamente se hace referencia, en este trabajo, aquellas que afectan al diseño, y más específicamente a la parte de diseño que atañe al chasis o estructura principal, que son: parte T (*General Technical Requirements*) y parte EV (*Technical regulation – Electric Vehicles*).

1.6.1. *Generalidades*

A continuación, se muestran los puntos de la normativa mencionada que incumben al diseño del chasis del monoplaza, y por tanto a todo el contenido de este trabajo:

- **Configuración del vehículo** (T2.1 "*Vehicle configuration*")*[1]*:

Ninguna parte del vehículo puede acercarse más de 75 mm del exterior del neumático delantero y trasero en vista lateral, con las ruedas posicionadas rectas.

NOTA: esta normativa restringe en gran medida la posición estirada del piloto y la longitud del primer volumen del CockPit

- **Visibilidad de componentes** (T2.5 "*Visible access*")*[1]*:

Todas las partes a inspeccionar deben ser claramente visibles a los inspectores técnicos sin el uso de instrumentos como endoscopios o espejos. El acceso visible puede ser proporcionado mediante la eliminación de paneles de la carrocería o paneles de acceso desmontable.

- **Estructura del vehículo** (T3.1 "*Vehicle structure*")*[1]*:

Los equipos podrán elegir entre dos opciones para la realización del chasis.

a) Parte T, articulo 3 como se define a continuación.

b) Parte AF, “Alternative Frame Rules” donde se puede encontrar en el apéndice AF de la propia normativa.

NOTA: estos dos grandes grupos separan la libertad o no de poder escoger la perfilaría de las barras que conforman el chasis, la normativa AF está prohibida en algunas competiciones, sin antes no competir en algún evento en el cual si este permitida, esto dificulta su selección, además de necesitar testeos físicos para demostrar su seguridad. Por eso en este trabajo no se hablara más de este apartado normativo.

A continuación toda normativa expuesta está dentro del apartado normativo PARTE T.

- **Requerimientos generales** (T3.2 "*General requirements*")*[1]*:

Entre otros requisitos la estructura debe incluir dos arcos antivuelco, una estructura protectora delantera con sistema de apoyo, un atenuador de impacto frontal y estructuras de impacto lateral. Así como las medida mínimas y máximas que atañen a la batalla y vía del monoplaza

- **Definiciones** (T3.3 "*Definitions*")*[1]*:

Las siguientes definiciones se aplican a todo el documento:

a) Main Hoop: Arco principal antivuelco situado lateralmente o posterior al torso del conductor.

b) Front Hoop: Arco intermedio antivuelco situado por encima de las piernas del piloto próximo al volante.

c) Roll Hoops: Tanto el Main Hoop como el Front Hoop se clasifican como Roll Hoops.

d) Frame: Elemento estructural que soporta todos los componentes funcionales. Este puede ser una estructura soldada, múltiples estructuras soldadas, o un conjunto de materiales compuestos con estructura soldada.

e) Front Bulkhead: Arco frontal cuya función es proteger las piernas del piloto.

f) Impact Attenuator: Dispositivo deformable que absorbe energía localizado en la Front Bulkhead.

g) Side Impact zone: Área lateral del monoplaza en la que la barra más alta ha de estar a 350 mm del suelo y ha de conectar Front Hoop y Main Hoop.

h) Node-to-node triangulation: La estructura del chasis ha de ser triangulada para que, al aplicar fuerzas en cualquier dirección, resulte fuerzas de tracción o compresión a los miembros de la estructura.

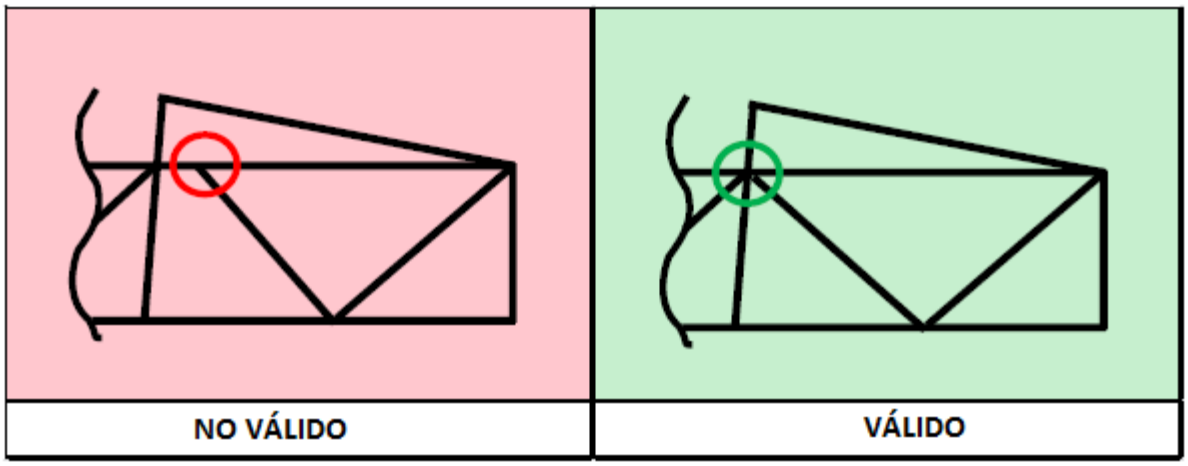


Figura 3.1 *Validación de la triangulación correcta de nodo a nodo*

NOTA: Es común que en el debate del diseño que los jueces penalicen mucho el no llevar toda carga a un nodo correctamente triangulado, esto es debido a que una carga de ese tipo genera un gran flector que podría doblar fácilmente una barra. Es complicado llevar nodos a los puntos de suspensión, así que deben estudiarse por el motivo dicho, siento esta una opción correcta que se acepta en la normativa

1.6.2. *Materiales y perfiles*

Este apartado de la normativa es la clave, aquí es donde más opciones y libertades te puede ofrecer la normativa para la optimización del chasis. No solo porque dan las opciones de utilizar materiales alternativos al acero como son el aluminio y las fibras de carbono, sino que te permiten modificar la perfilaría mínima obligatoria.

Esta optimización se verá en más detalle en las secciones pertinentes, pero a modo de resumen los apartados que mostramos a continuación tienen una variable en común, esta variable es $E \cdot I =$ Modulo de pandeo , una variable que debe mantenerse siempre por encima de los requerimientos mínimos. Esto significa que el peso, se mantiene casi invariante, lo cual no podemos reducir el peso, pero la rigidez torsional si puede aumentar, aumentando el momento de inercia de la sección. Si se crea una variable auxiliar llamada constante rígida, se puede optimizar el chasis en gran medida.

$$Rigidez\ torsional/Masa = \text{Constante rígida} \tag{3.1}$$

- **Requerimientos mínimos de materiales** (*T3.4 Minimum material requirements*) *[1]*:

La estructura principal del vehículo debe de estar construida de tubos de acero (mínimo 0,1% carbono), aleaciones, o alternativas aprobadas por las normas T3.5, T3.6, y T3.7.

ITEM or APPLICATION	OUTSIDE DIMENSION X WALL THICKNESS
Main & Front Hoops, Shoulder Harness Mounting Bar	Round 1.0 inch (25.4 mm) x 0.095 inch (2.4 mm) or Round 25.0 mm x 2.50 mm metric
Side Impact Structure, Front Bulkhead, Roll Hoop Bracing, Driver's Restraint Harness Attachment (except as noted above) EV: Accumulator Protection Structure	Round 1.0 inch (25.4 mm) x 0.065 inch (1.65 mm) or Round 25.0 mm x 1.75 mm metric or Round 25.4 mm x 1.60 mm metric or Square 1.00 inch x 1.00 inch x 0.047 inch or Square 25.0 mm x 25.0 mm x 1.20 mm metric
Front Bulkhead Support, Main Hoop Bracing Supports EV: Tractive System Components	Round 1.0 inch (25.4 mm) x 0.047 inch (1.20 mm) or Round 25.0 mm x 1.5 mm metric or Round 26.0 mm x 1.2 mm metric

Tabla 3.3 Tabla requerimientos mínimos de materiales

Nota 1: Excepto los agujeros de inspección, los agujeros perforados en cualquier tubo requieren de presentación en el SES (*Structural Equivalency Spreadsheet*).

Nota 2: Las propiedades del acero utilizado no debe ser menor a las propiedades del acero referencia en el SES:

i) Módulo elástico (E) = 200 GPa

j) Límite elástico (Sy) = 305 MPa

k) Límite a rotura (Su) = 365 MPa

- **Materiales Alternativos** (T3.5 "*Alternative tubing and material*")*[1]*

En el caso que se escoja tubos o materiales alternativos, se debe presentar el SES. Se ha de demostrar con cálculos que el material utilizado es equivalente a los mínimos requerimientos de T3.4.

En el caso de tubos doblados, estos han de ir acompañados de tubos de soporte del mismo diámetro y espesor que el tubo doblado.

- **Tubos de acero alternativos** (T3.6 "*Alternative steel tubing*")*[1]*

Espesor mínimo de tubo permitido para equipos que no realicen ensayos físicos, mostrado en tabla 2.4.

Espesor mínimo de tubo permitido para equipos que realicen ensayos físicos, mostrado en tabla 2.5.

Tabla 3.4 Tabla espesor mínimo de tubos (no requiere ensayo) *[1]*

MATERIAL & APPLICATION	MINIMUM WALL THICKNESS
Steel Tubing for Front and Main Roll Hoops, and Shoulder Harness Mounting Bar	2.0 mm (0.079 inch)
Steel Tubing for Roll Hoop Bracing, Roll Hoop Bracing Supports, Side Impact Structure, Front Bulkhead, Front Bulkhead Support, Driver's Harness Attachment (except as noted above), Protection of HV accumulators, and protection of HV tractive systems	1.2 mm (0.047 inch)

Tabla 3.5 Tabla espesor mínimo de tubos (requiere ensayo) *[1]*

MATERIAL & APPLICATION	MINIMUM WALL THICKNESS
Steel Tubing for Front and Main Roll Hoops, and Shoulder Harness Mounting Bar	1.6 mm (0.065 inch)
Steel Tubing for Roll Hoop Bracing, Roll Hoop Bracing Supports, Side Impact Structure, Front Bulkhead, Front Bulkhead Support, Driver's Harness Attachment (except as noted above), Protection of HV accumulators, and protection of HV tractive systems	0.9 mm (0.035 inch)

Nota 1: Todos los aceros son tratados iguales. No se permite aceros con menor espesor justificando que han sido tratados bajo condiciones especiales.

Nota 2: Para mantener el módulo de pandeo (EI), el diámetro exterior del tubo escogido ha de ser mayor al diámetro exterior del tubo de referencia de la tabla

T2.3.

Nota 3: Para mantener el límite elástico y el límite a rotura, el área del tubo escogido ha de ser mayor al área del tubo de referencia de la tabla T2.3.

1.6.3. *Estructura principal*

Seguidamente se mostrara la parte de la normativa que atañe a los conjuntos de barras que conforman unidades estructuras. Estas son las barras mínimas que ensamblan el conjunto total del chasis. Esto significa que si buscamos una optimización del conjunto total, solamente debemos tener dichos conjuntos, ya que añadir otro miembro u otro conjunto implicara muy posiblemente un aumento de peso que puede no contribuir en la rigidez torsional. Estos conjuntos se ven a continuación

- **Main Hoop y Front Hoop** (T3.10 "*Main and Front Roll Hoops*")*[1]*:

La cabeza y las manos de los pilotos no deben ponerse en contacto con el suelo en cualquier situación de vuelco.

El chasis debe incluir tanto una Main Hoop como una Front Hoop.

El piloto sentado en su correcta posición de conducción y sujeto con los sistemas de sujeción, el casco de un varón de 95 percentil europeo y todos los pilotos debe cumplir:

l) Debe haber una distancia mínima de 50,8 mm desde el casco del conductor, hasta la línea generada desde la Main Hoop hasta la Front Hoop (figura 2.2).

m) Debe haber una distancia mínima de 50,8 mm desde el casco del conductor, hasta la línea generada desde la parte alta de la Main Hoop hasta la sujeción de su refuerzo (figura 2.2).

n) El casco del piloto no puede sobresalir por la parte trasera de la Main Hoop, si el refuerzo de este está proyectado hacia la parte delantera de la estructura (figura 2.3).

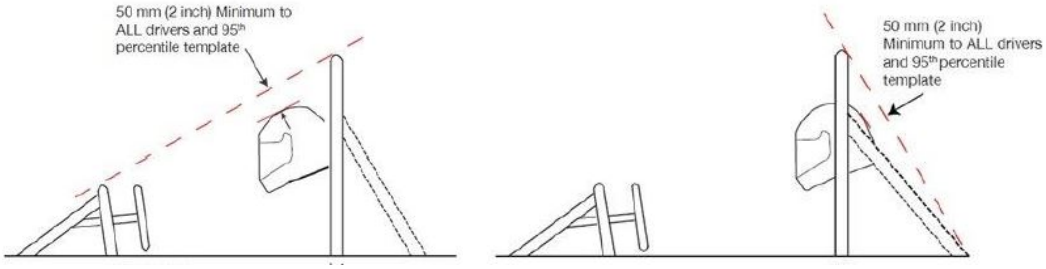


Figura 3.2 *Distancias mínimas a cumplir con percentil 95*

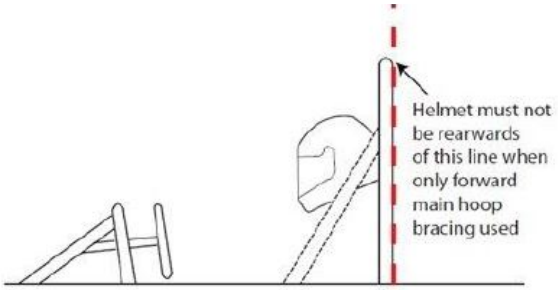


Figura 3.3 *Restricción en caso de llevar el refuerzo MH hacia adelante*

- **Main Hoop** (T3.11 "*Main Hoop*")*[1]*:

La Main Hoop debe de estar construida en una pieza continua de acero, siguiendo la tabla T2.3.

El uso de aleaciones de aluminio, titanio o materiales compuestos para realizar la Main Hoop está prohibido.

En vista lateral del vehículo, la Main Hoop no puede tener una inclinación superior a 10º con la vertical.

En vista frontal del vehículo, las barras verticales de la Main Hoop han de tener una anchura mínima de 380 mm.

- **Front Hoop** (T3.12 "*Front Hoop*")*[1]*:

La Front Hoop debe de estar construida de tubo metálico de sección cerrada, siguiendo la tabla T2.3.

Para una buena triangulación, se permite la construcción de la Front Hoop de más de una pieza.

El punto más alto de la Front Hoop debe estar más elevado que el volante en cualquier posición angular. Además, la Front Hoop no puede estar a más de 250 mm del volante.

En vista lateral del vehículo, la Front Hoop no puede tener una inclinación superior a 20º con la vertical.

- **Tirante Main Hoop** (T3.13 "*Main Hoop bracing*")*[1]*:

Los tirantes de la Main Hoop han de estar construidos de tubo de acero de sección de acero, siguiendo la tabla T2.3.

La Main Hoop ha de estar soportada por dos tirantes, uno en cada lado. Además, dichos refuerzos no pueden tener ningún tipo de doblez.

Los tirantes de la Main Hoop han de soportar como máximo a 160 mm del punto más alto de este. Además, dichos refuerzos no pueden tener una inclinación superior a 30º con la vertical (figura 2.4).

Los tirantes de la Main Hoop han de estar seguramente integrados en el chasis y han ser capaz de transmitir las fuerzas generadas por la Main Hoop a toda la estructura.

- **Tirante Front Hoop** (T3.14 "*Front Hoop bracing*")*[1]*:

Los tirantes de la Front Hoop han de estar construidos siguiendo la tabla T2.3 de la memoria.

La Front Hoop ha de estar soportada por dos tirantes, uno en cada lado. Además, estos han de soportar como máximo a 50,8 mm del punto más alto de la Front Hoop (figura 2.4).

En el caso que la Front Hoop estuviera inclinada más de 10º con la vertical, se ha de añadir un refuerzo extra.

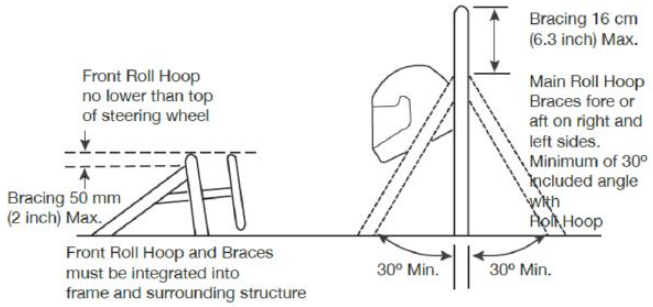


Figura 3.4 *Restricciones soportes Main Hoop y Front Hoop*

- **Otros requerimientos de** tirantes (T3.15 "*Other bracing requirements*")*[1]*:

Cuando los tirantes no son soldados a la estructura, dichos tirantes han de ir anclados a la estructura mediante tornillos de métrica 8.8 o superior.

- **Front Bulkhead** (T3.18 "*Bulkhead*"):

La Front Bulkhead debe de estar construida de sección cerrada, siguiendo la tabla T2.3.

La Front Bulkhead debe estar situada por delante de todos los objetos, incluidos los pedales.

- **Tirantes Front Bulkhead** (T3.19 "*Front Bulkhead support*")*[1]*:

La Front Bulkhead ha de ser soportada por un mínimo de tres miembros en cada lado del vehículo: refuerzo superior, refuerzo inferior y refuerzo diagonal.

El tirante superior ha de soportar como máximo a 50 mm del punto más alto de la Front Bulkhead, y ha de conectarse a 100 mm por encima y 50 mm por debajo de la barra superior de impacto lateral. En caso de ser más 100 mm por encima, entonces se requiere una triangulación para transferir las cargas correctamente.

El tirante inferior ha de conectar la base de la Front Bulkhead con la barra inferior de impacto lateral.

El tirante diagonal ha de conectar y triangular adecuadamente el refuerzo superior con el inferior.

- **Estructura de impacto lateral** (T3.24 "*Side impact structure for tube frame cars*")*[1]*:

La estructura de impacto lateral ha de constar como mínimo de tres miembros tubulares (superior, inferior y diagonal), localizados en cada lado del conductor cuando este esté sentado en posición de conducción (figura 2.5).

Los tres miembros tubulares han de estar construidos del material descrito en la tabla T2.3.

El miembro superior debe conectar el arco principal con el arco intermedio. Con un piloto de 77 Kg sentado en posición de conducir, el miembro superior ha de estar entre 300 y 350 mm del suelo.

El miembro inferior debe conectar la base del arco principal con la base del arco intermedio.

El miembro diagonal debe conectar y triangular adecuadamente el miembro superior con el inferior.

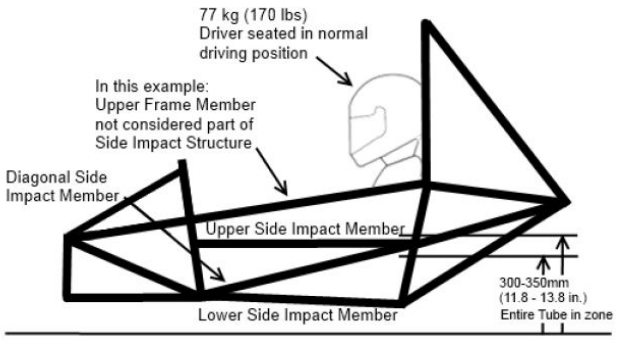


Figura 3.5 *Restricciones estructura de impacto lateral*

- **Agujeros de inspección** (T3.25 "*Inspection holes*")*[1]*:

Los inspectores pueden comprobar el cumplimiento de todas las barras. Esto será comprobado mediante test de ultrasonidos o agujereando zonas requeridas por el inspector.

- **Cockpit** (T4.1 "*Cockpit opening*"):

Para garantizar un tamaño adecuado del cockpit, una plantilla (figura 2.6) ha de poder ser insertada y pasada verticalmente hasta haber pasado el miembro superior de la estructura de impacto lateral.

- **Interior cockpit** (T4.2"Cockpit internal cross section")*[1]*:

Para garantizar un tamaño adecuado del interior del cockpit, una plantilla (figura 2.6) ha de poder ser insertada y pasada horizontalmente por el cockpit hasta 100 mm del pedal, estando este en la posición más retrasada posible.

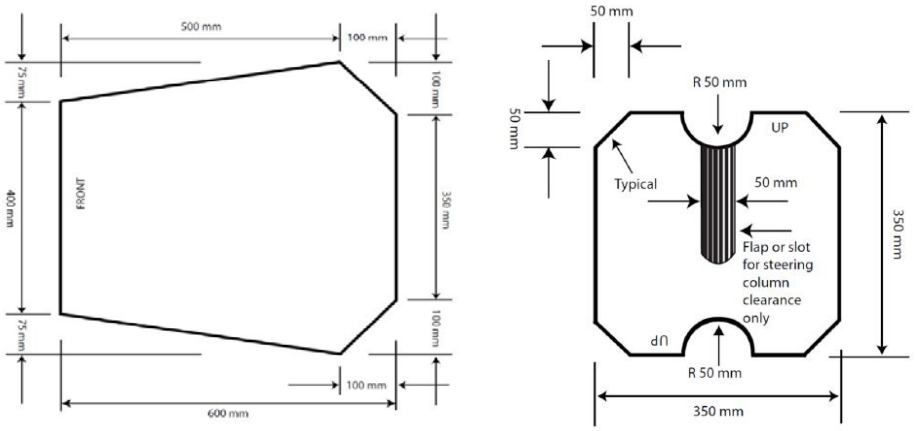


Figura 2.6 *Plantillas (izquierda: cockpit) (derecha: interior cockpit)*

- **Visibilidad del piloto** (T4.7 "*Driver visibility*")*[1]*:

El conductor debe tener una visibilidad adecuada a la parte frontal y los laterales del coche. Con el conductor sentado en una posición normal de conducción, este debe tener un campo de visión de 200º (un mínimo de 100º a cada lado del conductor). La visibilidad requerida se puede obtener por el conductor girando su cabeza y/o el uso de espejos.

- **Barra de cinturones** (T5.4 "*Shoulder harness*")*[1]*:

La barra de cinturones debe ser montada detrás del conductor y construido siguiendo los requerimientos de la tabla T2.3.

La barra de cinturones no puede ir instalada en el arco principal sin estar soportada por refuerzos. En el caso que esta barra no sea recta, las articulaciones entre la barra y la estructura han de ir reforzada por una correcta triangulación para prevenir rotación torsional.

La resistencia de la barra de cinturones debe ser reportado en la ficha correspondiente del SES.

- **Distancia al suelo** (T6.2 "*Ground clearance*")*[1]*:

La distancia al suelo debe ser suficiente para evitar que cualquier parte del vehículo, excepto los neumáticos, toque el suelo durante las pruebas. Contacto intencional con el suelo, conllevará la descalificación del evento.

- **Estabilidad antivuelco** (T6.7 "*Rollover stability*")*[1]*

La vía y el centro de gravedad del vehículo deben combinarse para proporcionar una estabilidad adecuada de vuelco.

La estabilidad antivuelco se evaluará en una mesa basculante mediante una prueba de pasa/no pasa. El vehículo no debe rodar cuando se inclina en un ángulo de 60º respecto a la horizontal, correspondiendo a 1,7 g. La prueba de inclinación se realizará con el conductor más alto en la posición normal de conducción.

- **Estructura accumulator container** (EV 3.4 "*Tractive system accumulator container*")*[1]*

El accumulator container, también llamado package de baterías, debe ser protegido de impactos posteriores y laterales por una estructura equivalente a la definida en T3.4. Además, dicha estructura debe ser incluida en el reporte del SES.

- **Colocación partes del sistema de tracción** (EV 4.2 "*Positioning of tractive system parts*")*[1]*

En el caso que hubiera partes del sistema de tracción que pudieran ser dañadas por un impacto lateral o posterior, por ejemplo, motores en el eje posterior, estos han de ir protegidos por una estructura apropiadamente triangulada equivalente a la definida en T3.4.

CAPÍTULO 4: MÉTODO MATRICIAL DE LA RIGIDEZ

1.7. Introducción

Los métodos de cálculo matricial (CM) de estructuras son un conjunto de métodos que tienen en común organizar toda la información en forma de matrices. En estos métodos, todas las relaciones entre las distintas partes de una estructura dan lugar a sistemas de ecuaciones con un alto número de variables pero donde no se han realizado suposiciones o simplificaciones en las que se pierda información relevante. Esta generalidad, junto a la estructura de la información en matrices, permite que su planteamiento y resolución pueda ser ejecutada de manera automática por medio de programas de ordenador, lo que ha hecho que en la actualidad sean la práctica habitual en la ingeniería.

En el presente trabajo se va a desarrollar el denominado método directo de la rigidez de cálculo matricial, aplicado a estructuras Tridimensionales formadas por barras y nodos. Este mismo esquema puede ser extendido a otras formas de discretizar una estructura o un medio continuo. De hecho, el método de los Elementos Finitos es la extensión del método de CM donde se trata con elementos que no son solo barras, sino volúmenes de distintas formas geométricas que modelan un mayor número de problemas mecánicos o físicos.

En todo el desarrollo del método aceptaremos las hipótesis generales en las que normalmente se desarrolla la Teoría de Estructuras y resistencia de materiales, esto es, comportamiento elástico y lineal del material y estado de pequeños desplazamientos

1.8. Características del método directo de rigidez

Es interesante hacer una breve explicación de las principales ventajas de estos métodos en verso a sus casi escasas desventajas

- **Dificultad:** La aplicación del CM, una vez que sus relaciones ya han sido desarrolladas, requiere un nivel de conocimiento para el usuario mucho más básico. No es necesario entender el sentido físico de estas relaciones para aplicarlas. Los métodos particulares exigen un conocimiento preciso del problema estructural a tratar y una toma de decisiones continúa sobre la influencia de diversos aspectos con el fin de simplificarlos. En el CM, al no tener que evaluar hipótesis o estimar efectos despreciables sobre el resultado final, la aplicación es directa.

- **Velocidad de cálculo:** Al incluirse todas las ecuaciones en CM, el tiempo de cálculo es mucho mayor por lo que, conocidas sus ecuaciones desde hace varios siglos, no han resultado útiles y de aplicación práctica hasta mediados del siglo XX. Los métodos particulares estaban desde el principio establecidos para poder aplicarse de manera manual y rápida, bien con ayuda de algún elemento de cálculo (reglas de cálculo) o incluso de manera gráfica (métodos de Maxwell-Cremona, Williot, etc.). hoy en día con la capacidad computacional de cualquier ordenador personal estos métodos son extremamente rápidos

- **Automatización del método:** Esta es una característica derivada de las anteriores y termina siendo la razón fundamental por la que los métodos matriciales son los que se han implantado actualmente, en particular el denominado método directo de la rigidez (que se desarrollará en el trabajo). La generalidad del método y el hecho de que se implementen todas las ecuaciones, reducen al mínimo las decisiones previas para modelar el problema matemáticamente. Si se organiza la información de manera que se puedan seguir pasos repetitivos para cada elemento (barra) que intervenga en la estructura, es muy fácil desarrollar un algoritmo de aplicación automática para todos los casos. En eso consiste el método matricial de la rigidez, y tiene como consecuencia que sea muy sencillo implementar programas de ordenador para aplicar el método. Con ello se salva la principal limitación en cuanto a la necesidad de resolución de grandes sistemas de ecuaciones y permite explotar todas las ventajas adicionales que tiene el CM.

1.9. Modelización del problema

Aunque el cálculo matricial está pensado para que las ecuaciones finales las resuelva un ordenador, existe un paso fundamental que es responsabilidad del programador y que no podrá ser realizada por un ordenador. Se trata de la modelización matemática del problema y de su correcta discretizacion. El cálculo puede estar bien realizado pero de nada sirve si el problema no responde a la realidad física que pretendemos representar.

El concepto de discretizacion debe ser establecido de manera precisa. Consiste en la representación del comportamiento de un medio continuo (la estructura) por medio de un conjunto finito de variables, en nuestro caso fuerzas aplicadas sobre el sólido y desplazamientos. Este número finito de variables son los desplazamientos en cada uno de los grados de libertad (GDL) de un sistema.

Determinar dichos grados de libertad y establecer todas sus relaciones son el punto de partida a partir del cual se resolverá el problema. El CM sólo aportara información en esos GDL, cualquier información adicional exigirá un paso adicional de interpretación de los resultados directos.

Para cada GDL, existirá una variable en fuerza y otra en desplazamiento. De ellas, una estará determinada por las condiciones de contorno (de carga o de desplazamiento impuesto) y la otra será la incógnita a despejar. En caso de ser incógnita de fuerza estaremos hablando de reacciones. Tanto los esfuerzos como cualquier incógnita interna de deformaciones, alargamientos o desplazamientos de puntos internos diferentes de los grados de libertad definidos en el problema deberán ser derivados posteriormente a partir de los resultados directos obtenidos en cada GDL definido.

1.10. Método de cálculo

En términos generales, existen dos procedimientos genéricos en mecánica de medios continuos de solidos deformables para poder establecer el sistema completo de ecuaciones dependiendo del orden en que las vayamos aplicando.

Las ecuaciones que podemos poner en juego son las ecuaciones de equilibrio, las de comportamiento y las de compatibilidad del problema. Cuando partiendo de las ecuaciones de equilibrio las utilizamos para incorporarlas a las de comportamiento y finalmente el resultado lo introducimos en las ecuaciones de compatibilidad, estamos aplicando el método denominado de la compatibilidad o de la flexibilidad. Hablando en términos de las variables implicadas, en este caso llegamos a formular los desplazamientos en función de las cargas aplicadas.

Si seguimos el procedimiento inverso, inicialmente relacionamos deformaciones y desplazamientos aplicando las ecuaciones de compatibilidad para posteriormente aplicar las leyes de comportamiento y finalmente las ecuaciones de equilibrio, en ese caso el método se denomina de la rigidez o del equilibrio

En cálculo matricial, es posible aplicar ambos procedimientos. Sin embargo, tal y como se desarrollara, únicamente es posible llegar a un procedimiento automático y sistematizado con el método de la rigidez, siendo este por tanto el que se ha implantado y generalizado.

Matriz de rigidez K, método más generalizado, que relaciona las solicitaciones $\vec{F}$ y desplazamientos $\vec{U}$ de manera que:

$$\vec{F} = K\vec{U} \tag{4.1}$$

Matriz de flexibilidad A, que relaciona esfuerzos y desplazamientos mediante

$$\vec{U} = A\vec{F} \tag{4.2}$$

1.11. Principio de superposición y linealidad

Los métodos de análisis de estructuras que se emplean en el presente trabajo se basa en los principios básicos de equilibrio y de compatibilidad. Además, al tratarse de pequeños desplazamientos y materiales estudiados únicamente en su rango lineal, la estructura cumple con el Principio de Linealidad.

En los problemas reales, no siempre es así de simple, por ello este método se basa en estas dos hipótesis argumentadas a continuación:

- Linealidad geométrica, es decir que los movimientos (desplazamientos y giros) que se producen son pequeños (hipótesis de los pequeños movimientos). Se entiende que los desplazamientos son pequeños comparados con las dimensiones geométricas de la estructura (espesor, luz, etc.); los giros son pequeños comparados con la unidad.

- Linealidad material, es decir, que la relación entre tensiones y deformaciones es elástica y lineal (los materiales cumplen la ley de Hooke generalizada).

El problema de determinar los esfuerzos y movimientos que se producen en una estructura por acción de las cargas es lineal, esto es, la respuesta de la estructura es una función lineal de la solicitación. Hay que resaltar que ninguna de las dos hipótesis anteriores son principios fundamentales, sino que se adoptan porque su aplicación se traduce en dos consecuencias importantes:

- Garantizan que la solución del problema estructural que satisface simultáneamente las condiciones de equilibrio y compatibilidad, existe y es única, y es independiente del método empleado para hallarla.

- Permiten adoptar simplificaciones importantes en el planteamiento del problema estructural, tanto al imponer las condiciones de equilibrio como las de compatibilidad.

La consecuencia directa de las hipótesis de linealidad es el Principio de Superposición que establece que los efectos que un sistema de fuerzas origina sobre una estructura son iguales a la suma de los efectos que cada una de las fuerzas origina por separado.

Alternativamente, se puede enunciar diciendo que los efectos que un sistema de fuerzas origina sobre una estructura no dependen del orden de aplicación de las fuerzas del sistema sobre la estructura

1.12. Constantes de rigidez de una barra elástica tridimensional

Las características definitorias del cálculo matricial de estructuras son la sistematización y la reutilización de las submatrices de los elementos. La sistematización se refiere a que, una vez planteados, todos los problemas se pueden resolver mediante un proceso fundamentalmente repetitivo, razón por la que se ajusta tan bien a su implementación en programas de ordenador. La reutilización se refiere a que existen fórmulas bien conocidas para las matrices que modelan cada uno de los elementos que pueden aparecer en una estructura, de forma que sólo es necesario estudiar estos elementos básicos (barras o vigas) una vez para poder emplear los resultados una y otra vez en innumerables problemas.

Modelizar la rigidez de un elemento de una estructura (una barra o un apoyo elástico) consiste en establecer qué relación existe entre los desplazamientos (y giros) que sufre en sus extremos y las solicitaciones (fuerzas y momentos) asociadas, también en los extremos. Si llamamos $\vec{u}$ y $\vec{f}$ a dos vectores que describen dichos desplazamientos y solicitaciones.

Es necesaria la obtención de las matrices de rigidez para los elementos y a continuación se detallara el proceso de cálculo utilizado.

En este trabajo directamente se utiliza la forma más completa de modelo de barra. Se trata de una barra con dos extremos y 12 GDL. Donde la matriz rigidez se expresa de la siguiente manera:

$$\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ M_4 \\ M_5 \\ M_6 \\ F_7 \\ F_8 \\ F_9 \\ M_{10} \\ M_{11} \\ M_{12} \end{Bmatrix} = \begin{bmatrix} A & & & & & & -A & & & & & \\ & B_y & & & & D_y & & -B_y & & & & D_y \\ & & B_z & & D_z & & & & -B_z & & D_z & \\ & & & E & & & & & & -E & & \\ & & D_z & & C_y & & & & -D_z & & C_y/2 & \\ & D_y & & & & C_z & & -D_y & & & & C_z/2 \\ -A & & & & & & A & & & & & \\ & -B_y & & & & -D_y & & B_y & & & & -D_y \\ & & -B_z & & -D_z & & & & B_z & & -D_z & \\ & & & -E & & & & & & E & & \\ & & D_z & & C_y/2 & & & & -D_z & & C_y & \\ & D_y & & & & C_z/2 & & -D_y & & & & C_y \end{bmatrix} \begin{Bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \\ \theta_{10} \\ \theta_{11} \\ \theta_{12} \end{Bmatrix} \quad (4.1.1)$$

En este caso la barra a tendrá 6 GDL en cada uno de sus dos nudos extremos, tres desplazamientos y tres giros. Su orientación en el espacio podrá ser arbitraria dentro de un sistema de coordenadas global del problema $\{X,Y,Z\}$ en el que definimos un sistema local de coordenadas de la barra tal que $\{X',Y',Z'\}$ está alineado con la dirección de la barra X → X' e Y' será normalmente uno de los dos ejes principales de su sección. De esta forma, en una barra espacial tenemos dos momentos de inercia I_y e I_z, nombrados según el eje del giro con respecto al cuál se definen. Así mismo, en lugar de un único momento (el flector M de la sección anterior), ahora tendremos dos momentos flectores M_y y M_z y un momento torsór M_x.

Una vez vista la matriz rigidez de un modelo de barra con 12 GDL, 6 por cada nodo, se explicara más detalladamente los valores de esta misma.

Una matriz de rigidez relaciona los desplazamientos generalizados (Traslaciones y giros) en las dos secciones extremas de un elemento o barra con las fuerzas necesarias para producir las deformaciones que dan lugar a dichos desplazamientos. Matemáticamente, una matriz de rigidez elemental es una relación entre el vector de fuerzas generalizadas (fuerzas y momentos) en las secciones extremas y el vector de desplazamientos generalizados (traslaciones y giros).

Las matrices de rigidez sólo dependen de las magnitudes geométricas de las barras y de los materiales de los que están hechas, pero no de las fuerzas. Propiedades como, modulo elástico, cizalla son proporcionados por el material a sí mismo, el módulo de torsión, el área y el momento de inercia son proporcionados a través de la geometría. Esto significa que si se consideran dos hipótesis de carga sobre una estructura, los dos cálculos pueden realizarse con las mismas matrices de rigidez aunque las fuerzas, reacciones y momentos sean diferentes.

1.12.1. *Cálculo de los coeficientes de rigidez*

Las componentes de la matriz de rigidez de una barra están formadas por diferentes rigideces o coeficientes de rigidez asociados a diferentes modos de deformación (extensión, cortadura, flexión, torsión, etc.). Para que el lector entienda el principio del modelo de resistencia de materiales es necesario definir los diferentes tipos de deformación y/o rigidez dependiendo de qué modelo deseemos utilizar.

Existe un número finito de maneras que podemos forzar una barra para que deforme, estas rigideces básicas para una barra recta son: la rigidez axial, la rigidez flexional, la rigidez frente a cortadura, la rigidez torsional y la rigidez mixta cortante-flexión.

El método que se utilizara será el problema que estará representado únicamente por tres GDL asociados al desplazamiento horizontal y vertical del extremo y a su giro. Esta elección nos limita el tipo de problemas que podremos resolver con esta discretización, por ejemplo, las cargas solo podrán estar aplicadas en ese extremo.

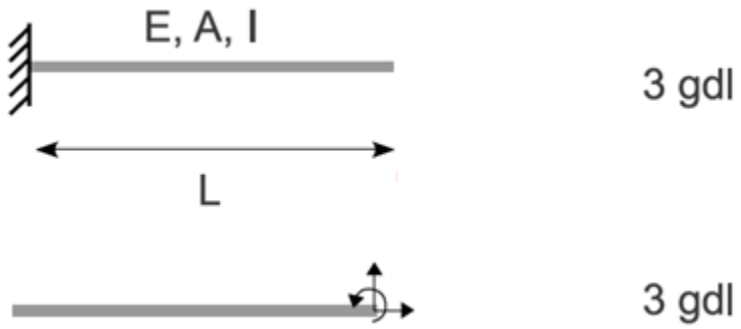


Figura 4.1 *discretización del elemento elástico*

Dónde:

E es el modulo elástico del materia
A es el área de la sección transversal del elemento elástico
L es la longitud

Una vez discretizado el problema, empezamos obteniendo su correspondiente matriz de rigidez K que relaciona las solicitaciones $\vec{F}$ y desplazamientos $\vec{U}$ de manera que:

$$\vec{F} = K\vec{U} \tag{4.1}$$

$$\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ M_4 \\ M_5 \\ M_6 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{Bmatrix} \tag{4.1.2}$$

Donde podemos obtener los valores de la matriz (los números Kij) por el método de inspeccionar cada uno de los problemas unitarios, uno por GDL. Dicho método consiste en plantear los problemas que corresponden con un vector de desplazamiento nulo en todos los GDL menos en uno (donde habrá un desplazamiento unitario) y calcular las solicitaciones asociadas a dicho desplazamiento. Los valores de esfuerzos así obtenidos se colocan en la columna correspondiente al GDL en el que se aplicó el desplazamiento, y así, columna a columna, se puede obtener la matriz completa.

El primer candidato es forzar una barra con desplazamiento en el eje de las X generando así una fuerza normal, la **rigidez axial** (4.2). Para una barra elástica recta y de sección constante la rigidez axial ***A*** se define simplemente como la relación F_{ax}/Δ_{ax} entre la fuerza horizontal F_{ax} y el desplazamiento horizontal Δ_{ax} . Podemos ver un ejemplo gráfico:

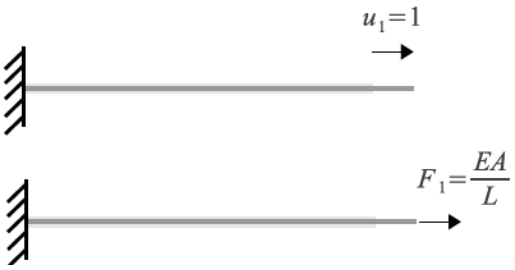


Figura 4.2 *Rigidez axial*

$$\boldsymbol{A} = \frac{F_{ax}}{\Delta_{ax}} = \frac{F_{ax} \cdot L}{F_{ax} \cdot L \,/\, EA} = \boldsymbol{\frac{EA}{L}} \tag{4.2}$$

Tomando el primer caso, donde U_1 = 1 (GDL primero, es decir deformación unitaria en el eje x) y los otros dos desplazamientos son cero. Si ahora sustituimos en la fórmula de la Ec. 4.1.2 estos desplazamientos y los valores de las fuerzas que deberíamos aplicar para obtenerlos tenemos:

$$\begin{Bmatrix} \frac{E\,A}{L} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} U_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} k_{11} \\ k_{21} \\ k_{31} \\ k_{41} \\ k_{51} \\ k_{61} \end{Bmatrix}$$

Es decir, ya hemos determinado los seis valores de la primera columna de la matriz de rigidez.

$$K = \begin{bmatrix} A & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ 0 & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ 0 & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ 0 & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ 0 & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ 0 & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix}$$

El segundo candidato a desplazar es en la dirección Y ó Z, dependiendo de dicha dirección podemos encontrarnos un desplazamiento U_2 o un desplazamiento U_3. Esto induce a una **rigidez a cortante** (4.3) de una barra elástica de sección constante que se define como una relación entre T_c/δ_c entre la fuerza vertical pura T_c aplicada en un extremo y el desplazamiento que sufre δ_c del mismo y la **rigidez mixta de cortante-flexión** (4.4) de una barra elástica de sección constante se define como una relación, M_e/δ entre el momento de empotramiento M_e cuando se aplica una fuerza vertical F_y en el extremo que produzca un desplazamiento δ en dicho nodo.

$$\boldsymbol{C} = \frac{T_c}{\delta_c} = \frac{T_c}{T_c \cdot L^3 \Big/ 12EI_z} = \frac{\boldsymbol{12EI_z}}{\boldsymbol{L^3}} \qquad (4.3)$$

Al mismo modo que la rigidez flexional, donde C_y / C_z depende del eje de momento de inercia que se requiera, para una barra circular $C_y = C_z = C$

$$\boldsymbol{D} = \frac{M_e}{\delta_c} = \frac{T_c \cdot L}{T_c \cdot L^3 / 12EI_z} = \frac{\boldsymbol{6EI_z}}{\boldsymbol{L^3}} \tag{4.4}$$

También al tratase de un modelo tridimensional se ha de tener en cuenta que D_y / D_z depende del eje de momento de inercia que se requiera, para una barra circular $D_y = D_z = D$

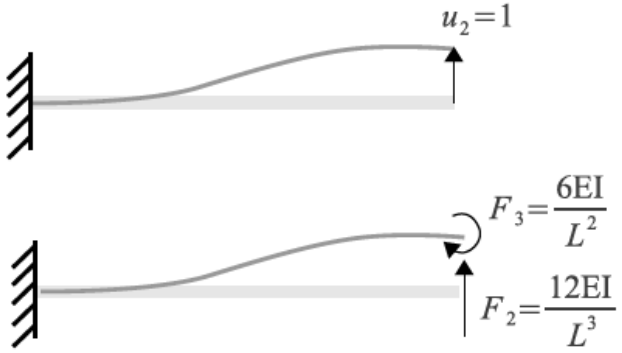


Figura 4.3 *Rigidez Flexional y rigidez mixta de cortante-flexión*

Donde se puede observar que las fuerzas necesarias para generar tal desplazamiento unitario en el grado de libertad U_2

$$\begin{Bmatrix} 0 \\ \frac{12EI_z}{L^2} \\ 0 \\ 0 \\ 0 \\ \frac{6EI_z}{L^3} \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} 0 \\ U_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} k_{12} \\ k_{22} \\ k_{32} \\ k_{42} \\ k_{52} \\ k_{62} \end{Bmatrix}$$

Es decir, ya hemos determinado los seis valores de la Segunda columna de la matriz de rigidez.

$$K = \begin{bmatrix} A & 0 & k_{13} & k_{14} & k_{15} & k_{16} \\ 0 & C & k_{23} & k_{24} & k_{25} & k_{25} \\ 0 & 0 & k_{33} & k_{34} & k_{35} & k_{36} \\ 0 & 0 & k_{43} & k_{44} & k_{45} & k_{46} \\ 0 & 0 & k_{53} & k_{54} & k_{55} & k_{56} \\ 0 & D & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix}$$

Es sencillo deducir que si se genera el desplazamiento en la dirección Z, U_3 los esfuerzos necesarios para llegar a esa condición son los siguientes:

$$\begin{Bmatrix} 0 \\ 0 \\ \frac{12EI_y}{L^2} \\ 0 \\ -\frac{6EI_y}{L^3} \\ 0 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} 0 \\ 0 \\ U_3 \\ 0 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} k_{13} \\ k_{23} \\ k_{33} \\ k_{43} \\ k_{53} \\ k_{63} \end{Bmatrix}$$

Observamos que ya hemos determinado los seis valores de la tercera columna de la matriz de rigidez.

$$K = \begin{bmatrix} A & 0 & 0 & k_{14} & k_{15} & k_{16} \\ 0 & C & 0 & k_{24} & k_{25} & k_{25} \\ 0 & 0 & C & k_{34} & k_{35} & k_{36} \\ 0 & 0 & 0 & k_{44} & k_{45} & k_{46} \\ 0 & 0 & -D & k_{54} & k_{55} & k_{56} \\ 0 & D & 0 & k_{64} & k_{65} & k_{66} \end{bmatrix}$$

El tercer candidato a desplazar es el grado de libertad U_5, dicho de un modo más correcto, más que un desplazamiento es una rotación en el plano las XZ, es decir una rotación en Y, afectando esto a la barra y debiendo existir una **rigidez flexional** (4.4) de una barra elástica de sección constante se define como una relación entre el esfuerzo aplicado en un extremo de la barra M_f y el giro que esta reporta en el mismo punto θ_f .

$$\boldsymbol{B} = \frac{M_f}{\theta_f} = \frac{M_f}{M_f \cdot L / 4EI_z} = \frac{\boldsymbol{4EI_z}}{\boldsymbol{L}} \qquad (4.4)$$

Donde B_y ó B_z depende del eje de momento de inercia que se requiera, para una barra circular $B_y = B_z = B$

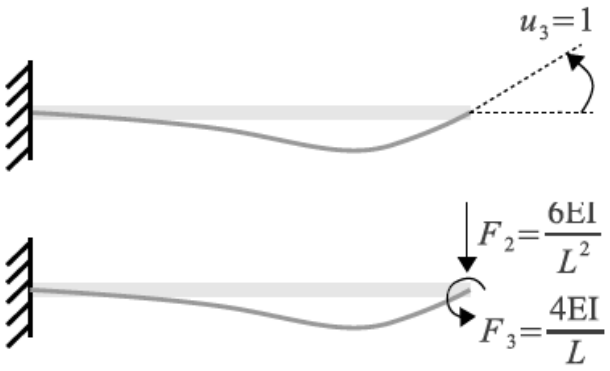


Figura 4.3 *Rigidez Flexional*

Donde se puede observar que las fuerzas necesarias para generar tal desplazamiento unitario en el grado de libertad U_5

$$\begin{Bmatrix} 0 \\ 0 \\ -\frac{6EI_z}{L^3} \\ 0 \\ \frac{4EI_z}{L} \\ 0 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ U_5 \\ 0 \end{Bmatrix} = \begin{Bmatrix} k_{15} \\ k_{25} \\ k_{35} \\ k_{45} \\ k_{55} \\ k_{65} \end{Bmatrix}$$

Esto resulta en seis valores más del matiz rigidez y son escritos en la quinta columna de la matriz.

$$K = \begin{bmatrix} A & 0 & 0 & k_{14} & 0 & k_{16} \\ 0 & C & 0 & k_{24} & -D & k_{25} \\ 0 & 0 & C & k_{34} & 0 & k_{36} \\ 0 & 0 & 0 & k_{44} & 0 & k_{46} \\ 0 & 0 & -D & k_{54} & B & k_{56} \\ 0 & D & 0 & k_{64} & 0 & k_{66} \end{bmatrix}$$

Se ve claramente que si mantenemos fijado el GDL U_5 las fuerzas que gobiernan el movimiento de dado grado de libertad son las mencionas en la figura 4.3, y estas son las causantes de una rotación en el eje Z, esta deformación se ve claramente en el plano YZ:

$$\begin{Bmatrix} 0 \\ \frac{6EI_z}{L^3} \\ 0 \\ 0 \\ 0 \\ \frac{4EI_z}{L} \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ U_6 \end{Bmatrix} = \begin{Bmatrix} k_{16} \\ k_{26} \\ k_{36} \\ k_{46} \\ k_{56} \\ k_{66} \end{Bmatrix}$$

Esto resulta en seis valores más del matiz rigidez y son escritos en la sexta y última columna de la matriz.

$$K = \begin{bmatrix} A & 0 & 0 & k_{14} & 0 & 0 \\ 0 & C & 0 & k_{24} & 0 & D \\ 0 & 0 & C & k_{34} & -D & 0 \\ 0 & 0 & 0 & k_{44} & 0 & 0 \\ 0 & 0 & -D & k_{54} & B & 0 \\ 0 & D & 0 & k_{64} & 0 & B \end{bmatrix}$$

Por último, el grado de libertad que queda por restringir es el U_4 , este se trata de una rotación en X, definida en el plano ZY y este descubre la **rigidez torsional** de una barra elástica de sección constante se define como una relación entre el momento torsor aplicado en el extremo M_t y el ángulo θ_t que reporta el mismo nodo como resultado del esfuerzo.

$$E = GJ/L \tag{4.5}$$

$$\begin{Bmatrix} 0 \\ 0 \\ 0 \\ \frac{GJ}{L} \\ 0 \\ 0 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \cdot \begin{Bmatrix} 0 \\ 0 \\ 0 \\ U_4 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} k_{14} \\ k_{24} \\ k_{34} \\ k_{44} \\ k_{54} \\ k_{64} \end{Bmatrix}$$

Esto finaliza los seis últimos valores de la matiz rigidez y son escritos en la cuarta columna de la matriz del siguiente modo:

$$K_{aa} = \begin{bmatrix} A & & & & & D_z \\ & C_z & & & & \\ & & C_y & & -D_y & \\ & & & E & & \\ & & -D_y & & B_y & \\ & D_z & & & & B_z \end{bmatrix} \qquad (4.1.2.1)$$

Finalizado esta explicación de donde salen los coeficientes de una barra elástica lineal, con módulo de alabeo despreciable al tratarse de materiales con constantes elásticas de alta magnitud. Se explica que para crear la matriz rigidez local, se necesitan 2x2 matrices de constantes nodales.

$$\begin{Bmatrix} F_a \\ F_b \end{Bmatrix} = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ab}^T & K_{bb} \end{bmatrix} \cdot \begin{Bmatrix} U_a \\ U_b \end{Bmatrix} \qquad (4.1.3)$$

Se deduce fácilmente las demás submatrizes del sistema, gracias a la explicación anteriormente formulada, solo se debe tener en cuenta que la matriz K_{ab} se encarga de magnificar las fuerzas que sufre el nodo a causa de las variaciones del nodo b y que la matriz $\mathrm{K_{bb}}$ no es más que los esfuerzos/reacciones que sufre el nudo b a causa de variaciones en deformaciones de este mismo. Así mismo es fácil interpretar que K_{ab}^T son los esfuerzos sufridos en el nodo b causados por el nodo a.

Una vez visto todo esto, queda por finalizada la explicación de la creación de la matriz rigidez de un elemento elástico-lineal sin alabeo.

1.13. Sistemas de referencia (global y local)

Al tratarse de un modelo matemático que intenta reflejar la realidad física, este debe estar localizado en un espacio eulcídeo tridimensional, es decir que dentro de este ortoedro, encontramos las dimensiones de ancho, largo y profundidad.

Además al estar vinculado este proyecto al sector de la automoción es necesario que respetemos la normativa.

Para definir el espacio de coordenadas eulcídeo, tanto sus ejes y planos como el origen se deben definir y se entienden por la siguiente razón:

- Origen de coordenadas: Es el punto de referencia de sistema de coordenadas. Es aquel punto en el que el valor de cualquiera de las tres coordenadas del sistema es nulo (0, 0, 0). Se encuentra coincidente con el suelo, en el punto medio del eje de la front bulkhead, este difiere del normativo, ya que en un proceso iterativo y deslocalizado tal y como es el diseño de un vehículo, facilita el trabajo de todos los departamentos. Dicho punto del eje se encuentra en el punto medio que une las huellas de las dos ruedas delanteras.

- Eje X: Este eje (también nombrado eje longitudinal) es paralelo al suelo, coincide con la dirección de avance del vehículo, y adquiere valores positivos en sentido contrario a dicho avance (del eje delantero al eje trasero del vehículo). El origen de coordenadas es coincidente con el eje X, y representa el punto donde éste adquiere valor nulo.

- Eje Y: Este eje (también nombrado eje transversal u horizontal) es paralelo al suelo y perpendicular al ya descrito Eje X. Si nos situamos en la posición del piloto, el Eje Y adquiere carácter positivo en sentido de izquierda a derecha. El origen de coordenadas es coincidente con el eje Y, y representa el punto donde éste adquiere valor nulo.

- Eje Z: Este eje (también nombrado eje vertical) es perpendicular al suelo y también a los otros dos ejes ya mencionados (X, Y). El origen de coordenadas es coincidente con el eje Z, y representa el punto donde éste adquiere valor nulo.

- Plano XY: Es el plano definido por los ejes X e Y (ambos coincidentes con el plano).

- Plano YZ: Es el plano definido por los ejes Y y Z (ambos coincidentes con el plano).

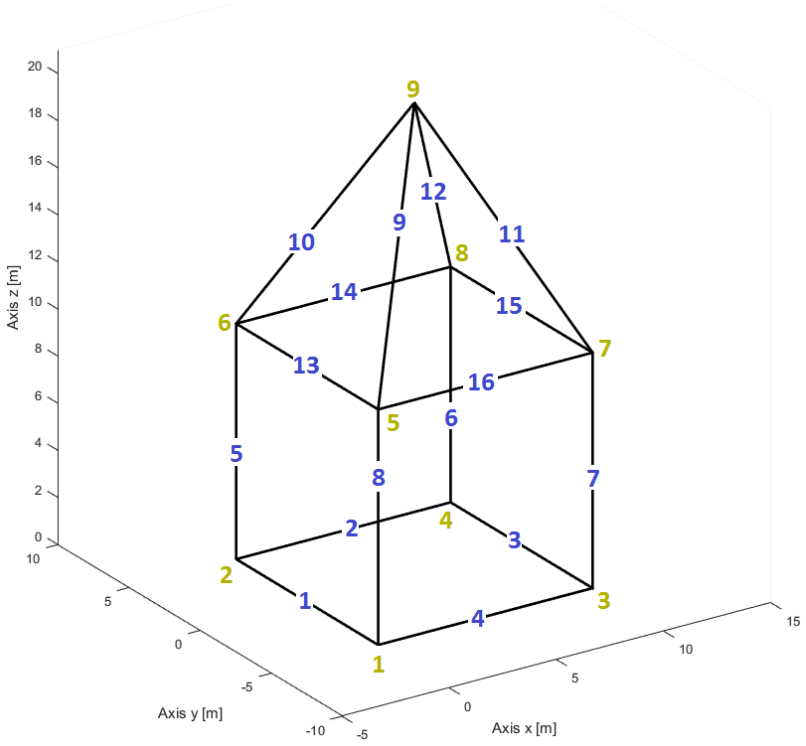
- Plano ZX: Es el plano definido por los ejes Z y X ambos coincidentes con el plano).

Definido el sistema de referencia donde el modelo matemático se asienta se precisa al mismo tiempo un convenio para referenciar la posición local de los elementos elásticos así como la posición real/global de los mismos. Por tanto, se definen:

- Un *sistema global de referencia*, es decir, un triedro dextrógiro como los de las Figuras 4.4, al que nos referiremos como (*X, Y, Z*). Este sistema se usa para referir a él la geometría de la estructura, las fuerzas y los movimientos incógnita. El "ensamblaje" de las matrices y vectores de las piezas, en general, se hace en este sistema.

- Un *sistema local de referencia* para cada pieza. Este sistema es útil para escribir las ecuaciones elásticas de las piezas, ya que se simplifican al referirlas a sus ejes locales. Nos referiremos a él como (*x, y, z*). Se elige el **eje *x* coincidente con la dirección y sentido positivo de la barra** (del extremo "*Nodo1*" al "*Nodo2*"), tal como indica la Figuras 4.5., en este programa los ejes secundarios de inercia siempre coinciden con la ortogonalidad del plano generado por el vector unitario $\hat{\imath}$ local del elemento elástico en el sistema global.

- En ciertos casos particulares, también es necesario definir sistemas locales de referencia de nudo. Es el caso de condiciones de contorno según ejes inclinados, en el caso que implica este proyecto, tales sistemas de referencia son inexistentes, ya que el resultado de esfuerzos nodales será descompuesto en los tres ejes coordenados.



NODOS	X[mm]	Y[mm]	Z[mm]	BARRAS	nodo 1	nodo 2	Dext[mm]	Dint[mm]
1	0,00	5,00	0,00	1	1	2	20	18
2	0,00	-5,00	0,00	2	2	4	20	18
3	10,00	5,00	0,00	3	4	3	20	18
4	10,00	-5,00	0,00	4	3	1	20	18
5	0,00	5,00	10,00	5	2	6	20	18
6	0,00	-5,00	10,00	6	4	8	20	18
7	10,00	5,00	10,00	7	3	7	20	18
8	10,00	-5,00	10,00	8	1	5	20	18
9	5,00	0,00	20,00	9	5	9	20	18
10				10	6	9	20	18
11				11	7	9	20	18
12				12	8	9	20	18
13				13	5	6	20	18
14				14	6	8	20	18
15				15	8	7	20	18
16				16	7	5	20	18

PROPIEDADES-BARRAS	Módulo Torsión [m4] (J)	Area [m2]	Momento de Inercia [m4] (Iz/Iy)
1	5,40E-09	5,97E-05	2,70E-09
2	5,40E-09	5,97E-05	2,70E-09
3	5,40E-09	5,97E-05	2,70E-09
4	5,40E-09	5,97E-05	2,70E-09
5	5,40E-09	5,97E-05	2,70E-09
6	5,40E-09	5,97E-05	2,70E-09
7	5,40E-09	5,97E-05	2,70E-09
8	5,40E-09	5,97E-05	2,70E-09
9	5,40E-09	5,97E-05	2,70E-09
10	5,40E-09	5,97E-05	2,70E-09
11	5,40E-09	5,97E-05	2,70E-09
12	5,40E-09	5,97E-05	2,70E-09
13	5,40E-09	5,97E-05	2,70E-09
14	5,40E-09	5,97E-05	2,70E-09
15	5,40E-09	5,97E-05	2,70E-09
16	5,40E-09	5,97E-05	2,70E-09

Figura 4.4 *Sistema global de referencia*

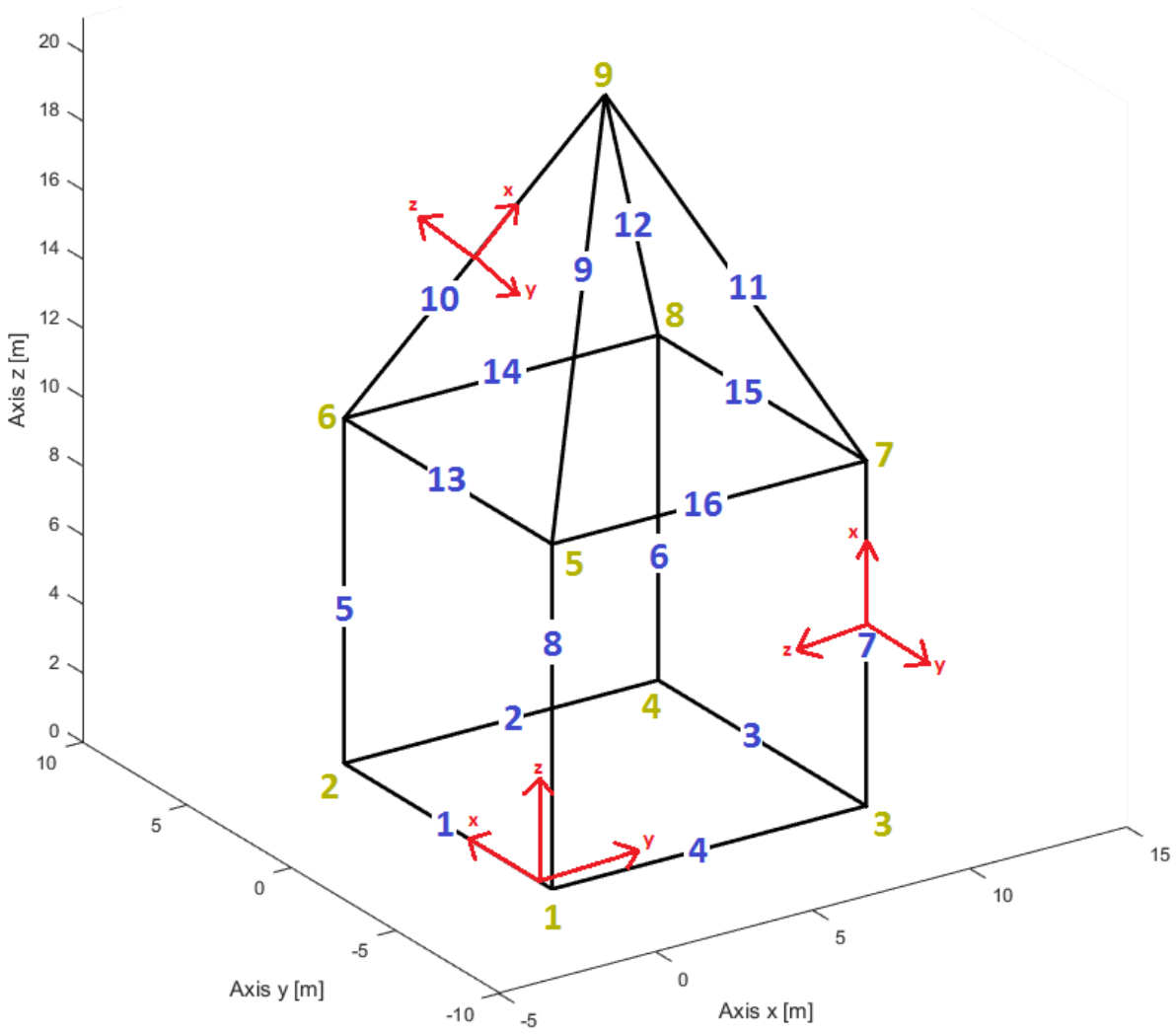


Figura 4.4.1 *ejemplo de sistema local de referencia de las barras*

1.14. Transformación de sistemas de referencia

En todo lo anterior, se ha definido la relación entre fuerzas y movimientos en los extremos de la pieza (ecuaciones elásticas) en el ***sistema local de referencia*** y considerada la pieza aislada. A continuación, se trata la transformación de dichas ecuaciones al ***Sistema global de referencia*** de la estructura a la que pertenece la barra.

En el caso de este trabajo únicamente daremos referencia al cambio de coordenadas de un sistema en tres dimensiones, que es el utilizado.

Para el caso tridimensional las matrices de rotación son una generalización de las vistas para estructuras planas. Una forma bastante extendida de representar esta matriz es en función de la matriz de cosenos directores R, de forma que esta matriz T_a de transformación queda:

$$\mathbf{T}^a = \left[\begin{array}{c|c|c|c} \mathbf{R} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \hline \mathbf{0}_{3\times3} & \mathbf{R} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \hline \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{R} & \mathbf{0}_{3\times3} \\ \hline \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{R} \end{array}\right]_{12\times12} \tag{4.6}$$

Dónde:

$$\mathbf{R} = \begin{bmatrix} \cos\theta_{\hat{x}x} & \cos\theta_{\hat{y}x} & \cos\theta_{\hat{z}x} \\ \cos\theta_{\hat{x}y} & \cos\theta_{\hat{y}y} & \cos\theta_{\hat{z}y} \\ \cos\theta_{\hat{x}z} & \cos\theta_{\hat{y}z} & \cos\theta_{\hat{z}z} \end{bmatrix}_{3\times3} \tag{4.7}$$

Donde el ángulo $\theta_{\hat{X}x}$ es el que forma el eje local $\hat{X}$ con el global X, $\theta_{\hat{y}Z}$ el que hace el eje local $\hat{Y}$ con el global z, y así sucesivamente.

La ecuación (4.7) puede ser representada mediante la figura 4.5 para así dar un mejor entendimiento geométrico del significado de la matriz R

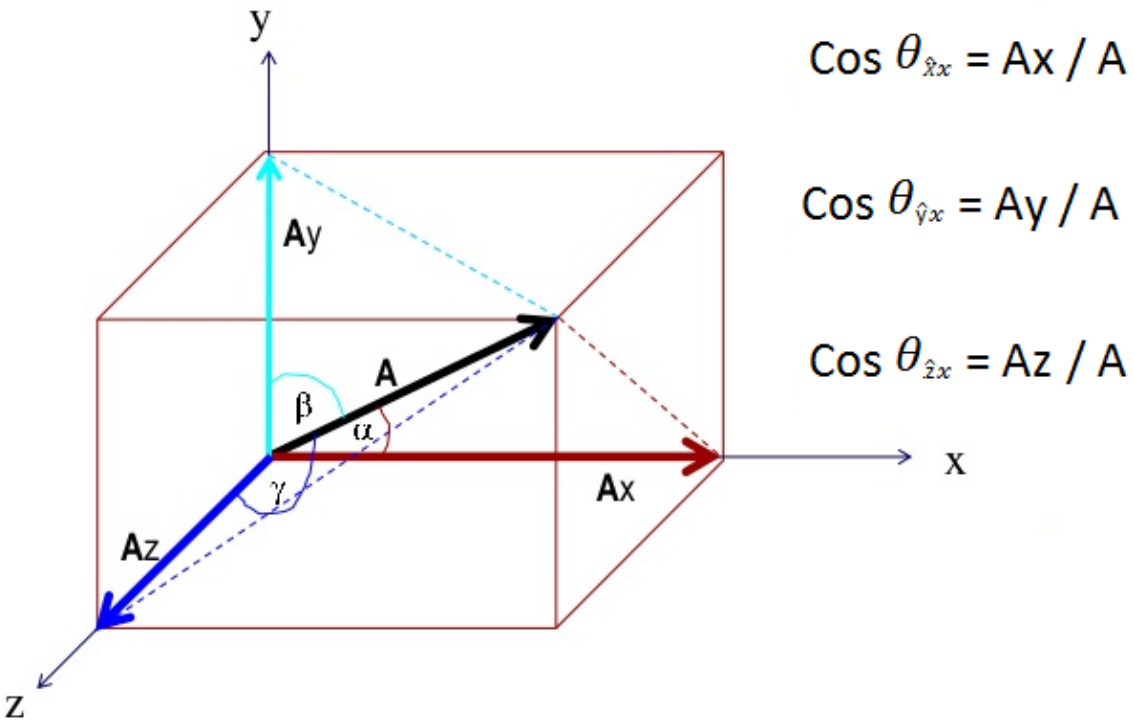


Figura 4.5 *Cosenos directores de un vector U en la base canónica.*

Estas matrices de rotación se pueden emplear para relacionar los vectores y matrices de rigidez de los sistemas locales y globales:

$$\vec{F} = T \cdot \hat{\vec{F}} \quad (4.8)$$

$$\vec{U} = T \cdot \hat{\vec{U}} \quad (4.9)$$

Dónde para cada elemento elástico (Barra):

$\vec{F}$ = Vector de fuerzas en el sistema de referencia global (base canónica)

$\hat{\vec{F}}$ = Vector de fuerzas en el sistema de referencia local.

$\vec{U}$ = Vector desplazamientos en el sistema de referencia global (base canónica)

$\hat{\vec{U}}$ = Vector desplazamiento en el sistema de referencia local.

T = Matriz de rotación

Entonces nuestra ecuación (4.1) en el sistema de referencia local quedaría de la siguiente forma:

$$\hat{\vec{F}} = \hat{K} \cdot \hat{\vec{U}} \quad (4.1.1)$$

Deduciéndose así que para transformar una matriz rigidez de un sistema de referencia local a un sistema global, la matriz rigidez debe ser tratada de la siguiente forma:

$$K = T \cdot \hat{K} \cdot T^T \quad (4.10)$$

Debido a la **Ortonormalidad**, la inversa de T es igual a la transpuesta de T

1.15. Características de una matriz rigidez

Con motivo de esclarecer un poco más el significado de una matriz de rigidez, conviene resumir aquí algunas de las principales características de las matrices de rigidez ya conocida como **K**:

- La matriz de rigidez es una propiedad del sistema estructural, no cambia en función del estado de cargas o de condiciones de contorno a que se someta a la estructura. Solo se verá afectada si se introduce algún elemento adicional.

- Cada columna representa las acciones necesarias para conseguir un desplazamiento unitario en el grado de libertad definido por el índice de la columna a la vez que se quedan fijados a cero el resto de los gdl.

- Una fila es un conjunto de multiplicadores que operados sobre el vector desplazamiento completo proporcionan el valor de la fuerza correspondiente al gdl definido por el índice de la fila.

- Cada termino k_{ij} se puede considerar una "función de peso" que representa la proporción de contribución a la fuerza del gdl *i* debido al desplazamiento del gdl *j*. En caso de que su valor sea cero significa que ambos gdl no está relacionados.

1.16. Ensamblaje, creación de la matriz global

El objetivo de esta sección es describir como las matrices de rigidez de los elementos (barras) individuales se ensamblan para formar la matriz de rigidez global K de una estructura, de forma que se pueda plantear el sistema de ecuaciones correspondiente a la estructura completa:

$$\vec{F}_{glob} = K_{GLOB} \cdot \vec{U}_{glob} \tag{4.1}$$

En primer lugar se expondrá la justificación teórica del método de ensamblaje, describiéndose a continuación el procedimiento para realizar el ensamblaje en sí de forma sistemática para que el usuario no deba de realizar ninguna tarea.

En el proceso se impone simultáneamente todas las condiciones de compatibilidad y de equilibrio de la estructura. Es decir, el método matricial implícitamente tiene en cuenta estas condiciones, debido a la estructuración parametrizada del método. Por lo cual, el usuario de nuevo no debe realizar un trabajo extra.

Una vez calculadas las ecuaciones elásticas de todas las piezas que forman la estructura en un sistema de referencia común, es decir, ya creada la matriz local en el sistema de referencia global, el siguiente paso en el método de rigidez es la construcción de la matriz global de rigidez de la estructura. Esta matriz global se obtiene mediante el ensamblaje de las matrices elementales de rigidez de las piezas.

Dicho ensamblaje se realiza considerando el equilibrio de fuerzas (y momentos, en su caso) que actúan sobre cada nudo de la estructura; de ahí el nombre de método de equilibrio con el que también se denomina al presente método de cálculo.

Para aclarar la operación de ensamblaje conviene definir el concepto de bloque o submatriz. Un bloque viene dado por un conjunto de componentes de una matriz de rigidez elemental. Usualmente las matrices de rigidez elementales tienen cuatro bloques: si llamamos a (izquierda) a la sección extrema situada más la izquierda y b (derecha) a la sección situada más a la derecha la matriz de rigidez de un elemento lineal, puede escribirse como:

$$K_{loc} = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ab}^{T} & K_{bb} \end{bmatrix} \tag{4.11}$$

Mismamente se pueden definir bloques para la matriz de rigidez global a partir de la numeración global usada para numerar los nudos. Para una barra elástica

recta de sección constante sin alabeo en 3D cada bloque es una submatriz de 6×6, eq (4.1.2.1).

La operación de ensamblaje consiste en formar una matriz de N×N bloques obtenidos copiando los bloques de las matrices de rigidez elemental y considerando sumas matriciales de los bloques cuando un nudo es común a varias barras.

A modo de ejemplo, para esclarecer la operación, creamos una estructura triangular, que conste de 3 nodos y 3 elementos elásticos, en la siguiente figura 4.6 podemos verlo en detalle:

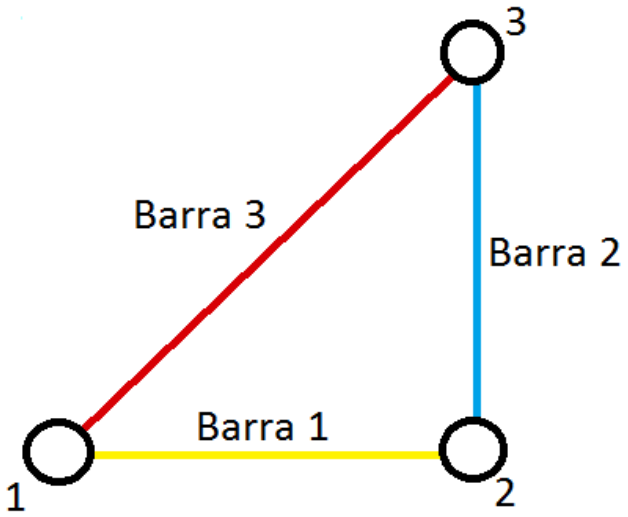


Figura 4.6 *ejemplo de estructura cuadrada n=3, m=3*

Aun tratándose de un ejemplo en 2 dimensiones, tengamos en cuenta que este trabajo está abarcando siempre las 3 dimensiones, por ello el ejemplo también estará centrado en estas dimensiones.

Al tratarse de una estructura de 3 barras (m) y 3 nodos (n), se genera una matriz de rigidez global siguiendo la expresión 4.12

$$Ancho\ de\ la\ matriz\ K_{glob} = n \cdot 6 \qquad (4.12)$$

Es decir se crea la siguiente matriz, esta matiz está vacía, o en el idioma de programación, está llena de ceros:

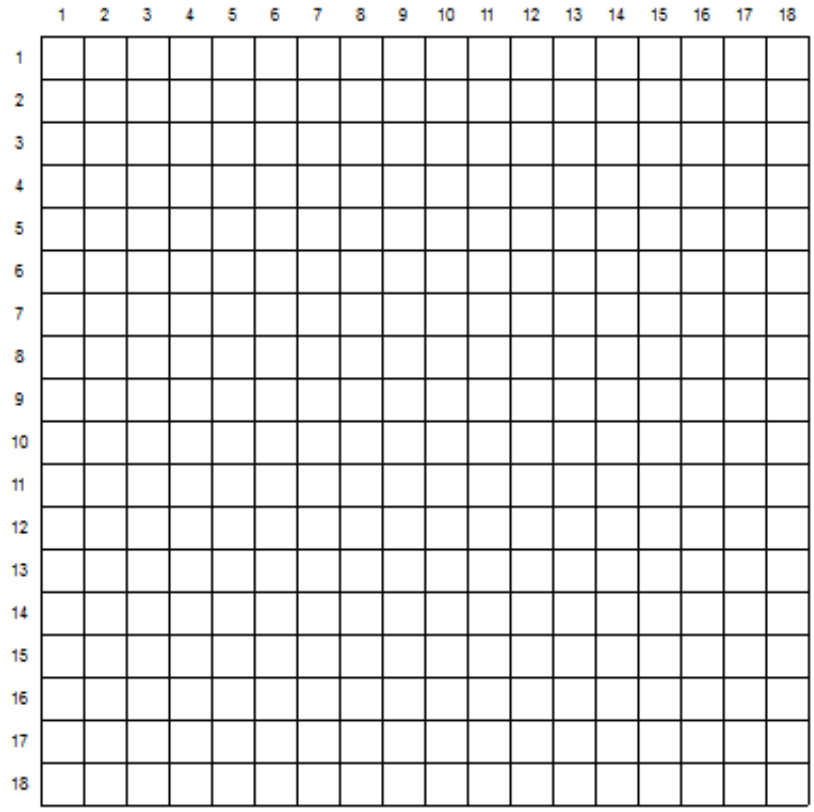


Figura 4.7 Matriz global $[K_{glob}]$ *con 18 Grados de libertad*

Seguidamente se debe empezar a ensamblar las matrices rigidez de cada elemento (barra), empezaremos por la barra 1, cabe decir que no importa el orden utilizado, pero en lenguaje computacional es imperativo realizar la iteración de ensamblado consecutivamente.

Así pues se puede ver que la barra 1 esta acoplado a los nodos globales 1 y 2 lo cual significa que el bloque $[K_{glob\ 11}]$ será el bloque $[K_{elemento\ aa}]$ (o una suma de bloques de ese tipo ellos si el nodo es común a varias barras), el bloque $[K_{glob\ 22}]$ será el bloque $[K_{elemento\ bb}]$ (o una suma de bloques de ese tipo ellos si el nodo es común a varias barras) y análogamente sucede con los bloques $[K_{elemento\ ab}]$ y $[K_{elemento\ ba}]$ Esto se puede ilustrar de manera simple siguiendo el ejemplo.

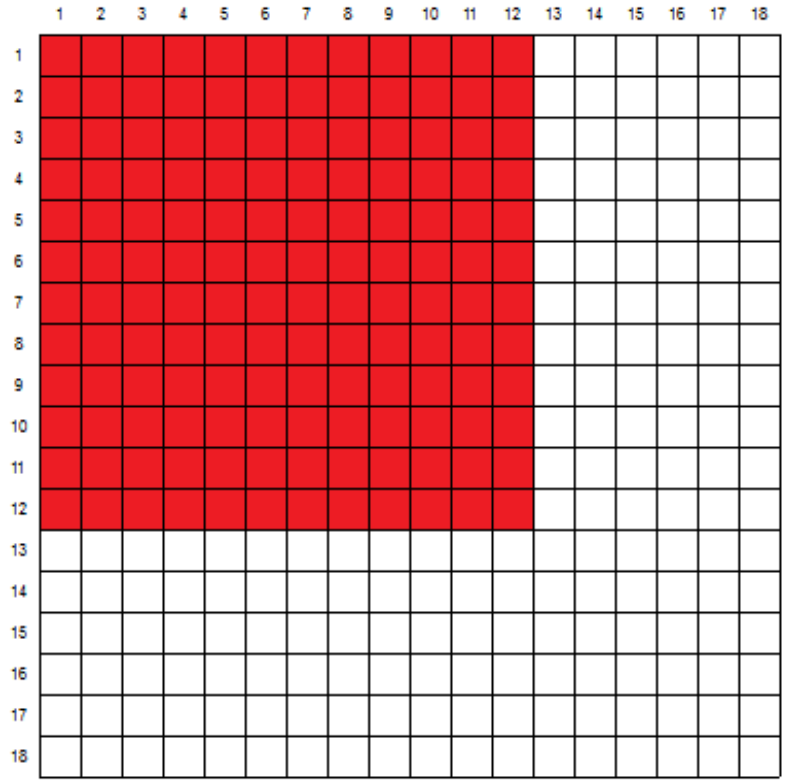


Figura 4.8 $[K_{glob}]$ *barra 1 ensamblada*

Hay que esclarecer que los cuadrados en rojo son coordenadas potenciales de añadir o adicionar un valor dado por la matriz local del elemento, dicho también es muy probable que en valor añadido o adicionado sea un valor nulo.

Veamos cómo se suceden las barras consiguientes, 2 y 3 en la composición total de la matriz global:

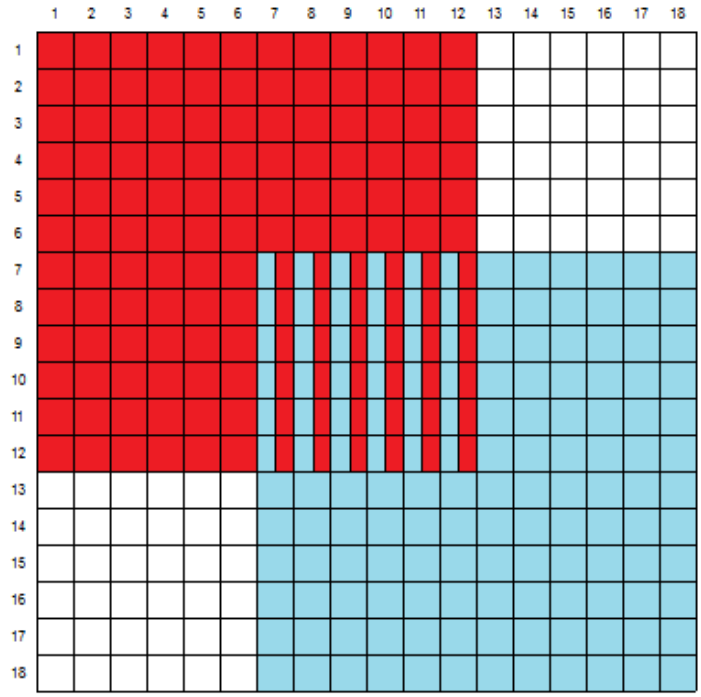


Figura 4.9 $[K_{glob}]$ *barra 1y barra 2 ensamblada*

Como se puede observar en la figura 4.9 existe una adición en los grados de libertad *i,j* =7-12. Estos grados de libertad tienen su razón de ser un nodo (nodo2) compartido por dos elementos elástico (barras).

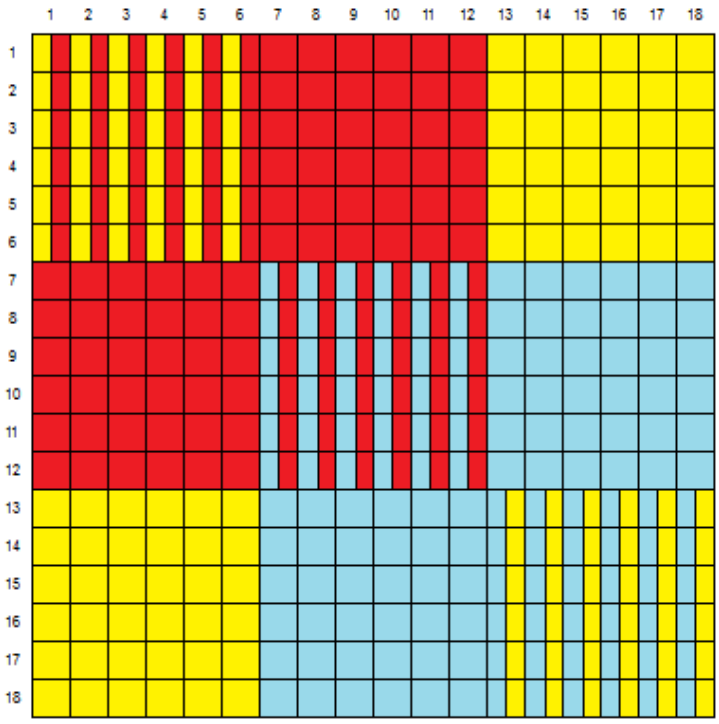


Figura 4.10 $[K_{glob}]$ *barra 1, barra 2y barra 3 ensamblada*

Es necesario decir que, esta es un ejemplo de una infinidad de estructuras, utilizar el método iterativo por elementos es más poderoso que utilizar el método iterativo de nodos, computacionalmente hablando.

Dado que el ejemplo anterior no es del todo ilustrativo, es decir, no muestra cuantas coordenadas de la matriz global tienen un valor nulo, a continuación se ilustrara en la figura 4.11 los valores no nulos (verde) junto a los valores nulos (blanco).

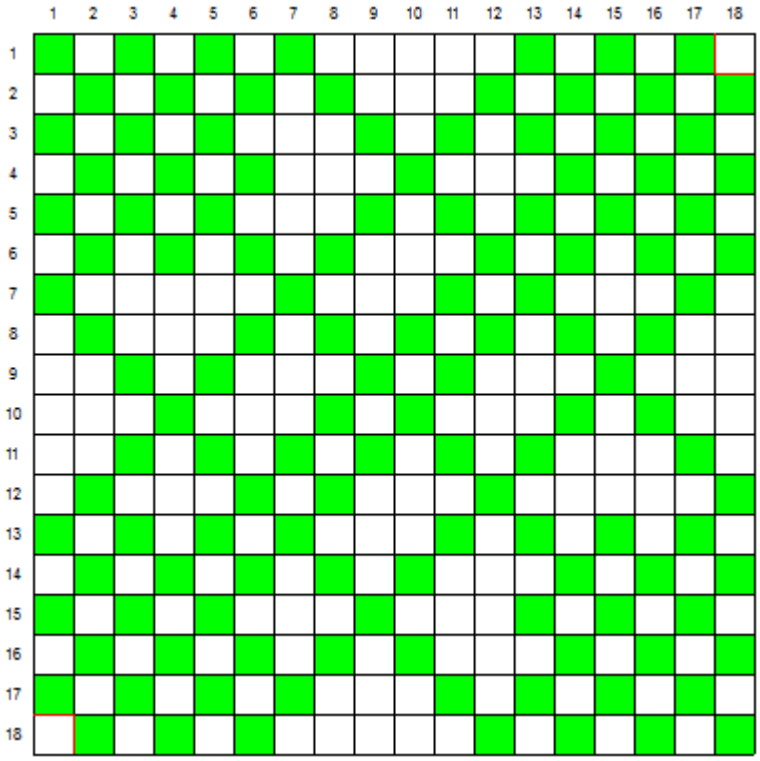


Figura 4.10 $[K_{glob}]$ *barra 1, barra 2y barra 3 ensamblada*

Se puede ver fácilmente que en el ejemplo mostrado, y así también el 95% de problemas matriciales, la matriz está guardando muchos valores nulos, es una simpleza requerir memoria de un computador cuando se tratan de matrices de enormes dimensiones, para ello hay procedimientos de almacenaje de este tipo de matriz.

En este último párrafo aparecen ambigüedades a la cuantificación de valores, tales como muchos y enormes. Para poder resolver estas dudas hay que navegar por técnicas avanzadas de cálculo de sistemas de ecuaciones para la programación. Al no ser esa la razón de este trabajo y entrando en un rango prudente de matriz global (300x300, tiempo de resolución ~ 8µs con MATLAB) se

dejara como lectura adicional al lector de este trabajo.

Una vez obtenida la matriz global ya ensamblada el siguiente paso es resolver la ecuación 4.1 Llegados a este punto nos encontramos que existe una singularidad en la matriz global, no tiene solución.

Esto es lo que las matemáticas expresan, que el sistema de ecuaciones no es compatible ya que el determinante de la matriz rigidez es siempre nulo. Como el determinante es nulo se deduce que existen valores propios nulos. De hecho, se puede demostrar que el número de valores propios nulos coincide con el número de movimientos de sólido rígido posibles.

Lo que la física de este problema expresa es que, la matriz global está sujeta a la nada, al vacío, es decir, que si introducimos una fuerza externa a un nodo, esta estructura, ya modelizada matemáticamente como una matriz global, se desplazara por el vacío sin restricción alguna de movimiento, no sufrirá deformación alguna y, por lo tanto, con tensiones y fuerzas nulas (despreciando los diferenciales de inercia debidos a la masa).

La solución a esta singularidad la explicaremos en el apartado 4.12

1.17. Condiciones de contornos

1.17.1. *Apoyos*

La especificación de las condiciones apoyo de la estructura se traduce en que algunos de los grados de libertad de ésta tienen sus valores prescritos. El número de grados de libertad prescritos debe ser el suficiente para impedir los movimientos de sólido rígido de la estructura.

Antes de abordar la resolución del problema de forma matricial es necesario identificar en que grados de libertad queda cada nudo libre y en cuales está su movimiento o giro restringido, y por tanto, aparecen reacciones. Un resumen de los apoyos más comunes y utilizados en este trabajo con los gdl que restringen se muestra a continuación

- Empotrado puro: es un tipo de unión entre sólido resistente y otro sólido inmóvil respecto a un sistema referencia también inmóvil, que elimina por completo la posibilidad de movimiento de un sólido respecto al otro en los puntos del empotramiento. Matemáticamente un empotramiento reduce el número de grados de libertad de los puntos del mismo, es decir, anula cualquier valor de desplazamiento o giro en todos los grados de libertad posible. En una estructura tridimensional se restringirían los 3 grados de libertad de movimiento en los tres ejes coordenados y además los 3 gdl de giro sobre los tres ejes coordenados.

  Estos pueden ser uniones atornilladas o soldadas.

- Empotrado deslizante: es una versión de empotrado igual que el anterior pero permitiendo un desplazamiento o un gdl en uno dos o tres de los ejes coordenados, restringiendo aun los 3 gdl de giro sobre los tres ejes coordenados.

- Apoyo simple o apoyo articulado: este apoyo está restringiendo únicamente los grados de libertad de desplazamiento en los ejes coordenados, dejando así libres los de rotación de los mismos.

  Estos pueden ser rotulas que permitan la rotación total o parcial en todos los ejes coordenados, en este grupo también se incluirían las rotulas que solo permitirían el giro en uno o dos ejes coordenados, tales como rotulas cilíndricas.

- Apoyo deslizante: es la versión del apoyo simple salvo que puede permitir liberar uno dos o tres gdl de desplazamiento en los ejes coordenados

Una vez conocidas estas condiciones de contorno de los apoyos, debemos aplicarlo al método matricial, creando un vector $\vec{U}_{glob}$ que no es más que la unión consecutiva de los vectores desplazamiento de cada nodo $\vec{U}$.

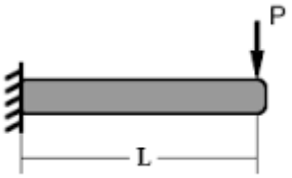


Figura 4.11 Ménsula

A modo de ejemplo imaginemos una ménsula, la cual sabemos que en el primer nodo esta empotrado y en el segundo esta libre, como muestra la figura 4.11 Vemos que el vector $\vec{U}$ de este elemento elástico debe quedar de la siguiente manera:

$$\vec{U}_{elemento} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \delta_7 \\ \delta_8 \\ \delta_9 \\ \theta_{10} \\ \theta_{11} \\ \theta_{12} \end{Bmatrix}$$

Se ve claramente que los primeros 6 gdl están restringidos debido a la naturaleza del apoyo y que en el nodo 2 los grados de libertad que van desde el 7-8 están sin restringir, es decir libres.

1.17.2. *Fuerzas*

Hay que tener en mente que el método de cálculo matricial solamente modela una serie discreta de nudos, estando los elementos entre ellos (típicamente barras o muelles) modelados por medio de las correspondientes matrices de rigidez. Las variables del problema, ya sean datos conocidos o incógnitas a averiguar, son todas las fuerzas y todos los desplazamientos de dicho conjunto discreto de nudos. Lo que ocurre entre un nudo y otro no es tenido en cuenta para nada, aparte de en las matrices de rigidez de cada elemento.

Una consecuencia de esta discretizacion del problema es que las cargas o fuerzas externas aplicadas a la estructura solamente pueden aparecer en los nudos. Para dejarlo claro: el método matricial no permite que existan cargas distribuidas o puntuales en mitad de las barras. Existen, por supuesto, maneras de abordar dichos problemas mediante el planteamiento de un problema equivalente donde cargas no nodales se convierten en nodales, pero el método matricial en si solo puede manejar cargas en los nudos.

Los esfuerzos solo serán lógicos aplicarlos en aquellos grados de libertad que estén libres, y estos para cada nodo son los siguientes:

$$\vec{F} = \begin{Bmatrix} F_x \\ F_y \\ F_z \\ M_x \\ M_y \\ M_z \end{Bmatrix}$$

1.18. Resolución del sistema de ecuaciones

En este punto, se ha conseguido plantear el problema de pequeñas deformaciones de una estructura como un sistema lineal en la forma $\vec{F} = K \cdot \vec{U}$. Sin embargo, el sistema no se puede resolver de manera inmediata por estar datos e incógnitas entremezcladas en los vectores $\vec{F}$ y $\vec{U}$.

Se hace necesario por tanto llegar a un sistema de ecuaciones reducido, esto significa que debemos eliminar del sistema de ecuaciones los grados de libertad prescritos como fijos.

Así si las condiciones de apoyo son suficientes para impedir los movimientos de solido rígido de la estructura, la matriz de rigidez global, que ya hemos comentado que es singular, devendrá modificada y se eliminará su singularidad. Llegando así a obtener un sistema de ecuaciones compatibles con una única solución. Esto es debido a que la matriz de rigidez reducida, el determinante es no nula y por consiguiente tiene inversa.

A continuación se expondrá un ejemplo grafico para mayor entendimiento. Volviendo al ejemplo de la figura 4.6 e imponiendo condiciones de contorno de empotramiento puro al nodo 3, además de una fuerza cualquiera en el nodo 1 o 2 para que el estudio tenga una razón. Veamos cómo se comportara la matriz de rigidez global de la figura 4.10

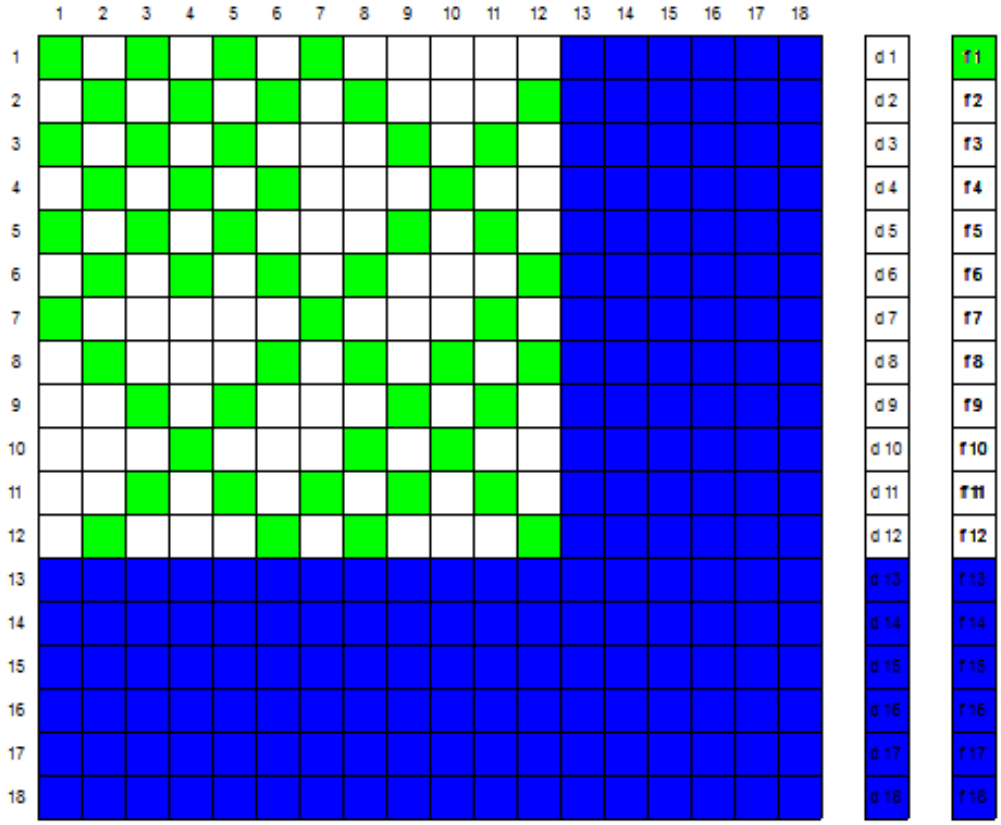


Figura 4.12 K_{glob} *reducida, sistema de ecuaciones compatible*

Como se puede ver en la figura 4.12 los gdl del nodo 3 están extinguidos por lo cual esos gdl se eliminan de la matriz rigidez, eliminando así la fila y columna que refiere al nodo 3, también se puede observar que es necesario introducir una fuerza para inducir un problema y esta está en el gdl nº 1 es decir, el primer nodo en la F_x de un valor arbitrario no nulo

1.19. Cálculo de los esfuerzos en los nodos y barras

El cálculo de reacciones en sí, no es parte del método, ya que este únicamente nos solventa la magnitud de desplazamiento o giro ocasionado en cada gdl libre.

Las reacciones en los nudos con movimientos impedidos o prescritos se calculan a partir del sistema original de ecuaciones, una vez conocidos los movimientos incógnita del problema.

Conocidos los movimientos de los nudos, principal incógnita del problema, se pueden determinar matricialmente los esfuerzos en los extremos de las barras (**esfuerzo axil, flector, cortante y torsor**). Estos esfuerzos conviene expresarlos en el sistema coordenado local para obtener directamente las leyes de esfuerzos en las barras.

Una vez se han resuelto los desplazamientos de la estructura completa como se ha descrito hasta ahora, tenemos perfectamente definido el valor del vector de desplazamientos en coordenadas globales U. A continuación vamos a mostrar cómo, a partir de únicamente esta información y las matrices de rigidez, es posible calcular los esfuerzos que soporta cada una de las barras de la estructura.

Supongamos que nos centramos en una barra en particular situada entre los nudos i y j. Los esfuerzos vendrán determinados por cómo ha sido obligada a deformarse en dichos extremos, valores que nombramos como U_a y U_b y que conocemos por ser una parte del vector U. Recodemos la ecuación 4.1.3

$$\begin{Bmatrix} F_a \\ F_b \end{Bmatrix} = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ab}^T & K_{bb} \end{bmatrix} \cdot \begin{Bmatrix} U_a \\ U_b \end{Bmatrix} \qquad (4.1.3)$$

Donde se ha expresado un elemento elástico general. Debemos calcular las incógnitas de los esfuerzos $F_{barra\,n}$, y que no deben confundise con las fuerzas del vector principal F. dicho también de paso la matriz rigidez que se refiere a 4.1.3 en este apartado es en la referencia local.

Por lo tanto únicamente hemos de calcular el vector de desplazamientos en coordenadas locales $U_{barra\,n}$ lo cual es muy sencillo a partir de los vectores correspondientes en coordenadas globales U_a y U_b que ya conocemos. Recordando que la matriz de transformación de coordenadas de la barra a, denotada como T, tiene la propiedad de convertir coordenadas locales en globales, es decir:

$$\vec{\boldsymbol{U}} = \boldsymbol{T} \cdot \hat{\vec{\boldsymbol{U}}} \qquad (4.9)$$

Se ve la relación inversa:

$$\hat{\vec{\boldsymbol{U}}} = \boldsymbol{T^{-1}} \cdot \vec{\boldsymbol{U}}$$

Entonces:

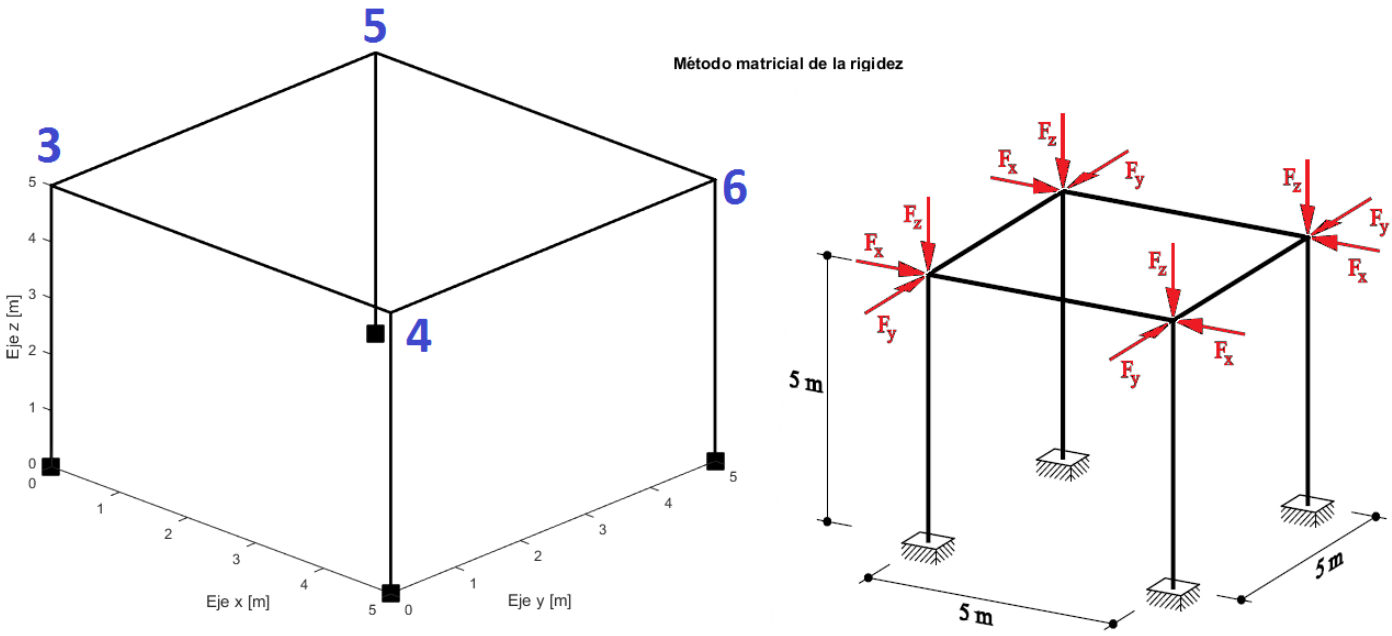
$$\hat{\vec{\boldsymbol{F}}} = \hat{\boldsymbol{K}} \cdot \boldsymbol{T^{-1}} \cdot \vec{\boldsymbol{U}} \qquad (4.13)$$

Formula que nos permite calcular los esfuerzos de la barra en coordenadas locales, al disponer de todos los términos de la derecha tras resolver el problema matricial.

Todo esto es complicado de explicar sin un ejemplo, vamos a construir uno para que se entienda mejor.

El ejemplo se basa en el cálculo de todas las fases vistas hasta ahora pero se prestara más atención en la resolución de los esfuerzos, el ejemplo es simple y se trata de una estructura de 8 barras o elementos elásticos lineales, y contiene 8 nodos, 4 de los cuales se encuentra en una situación de empotramiento puro.

Para ver con más detalle lo explicado en la figura 4.13 se detallan estos datos gráficamente



NODOS	X[mm]	Y[mm]	Z[mm]	BARRAS	nodo 1	nodo 2
1	0,00	0,00	0,00	1	1	3
2	5,00	0,00	0,00	2	2	4
3	0,00	0,00	5,00	3	7	5
4	5,00	0,00	5,00	4	8	6
5	0,00	5,00	5,00	5	3	5
6	5,00	5,00	5,00	6	5	6
7	0,00	5,00	0,00	7	6	4
8	5,00	5,00	0,00	8	4	3

Figura 4.12 *Estructura ejemplo con numeracion de nodos y condiciones de contorno*

Pongamos como ejemplo las deformaciones del nodo 3, recordemos que los nodos 1, 2, 7 y 8, que están en la parte inferior de la estructura, son empotrados y su deformación es nula.

$$\overrightarrow{d_3} = \begin{Bmatrix} \delta_{13} \\ \delta_{14} \\ \delta_{15} \\ \theta_{16} \\ \theta_{17} \\ \theta_{18} \end{Bmatrix} = \begin{Bmatrix} \delta_{x3} \\ \delta_{y3} \\ \delta_{z3} \\ \theta_{x3} \\ \theta_{y3} \\ \theta_{z3} \end{Bmatrix} = \begin{Bmatrix} 1{,}25 \\ 1{,}25 \\ -2{,}50 \\ -0{,}250 \\ 0{,}250 \\ 0 \end{Bmatrix} \cdot 10^{-4}$$

El lector puede ver fácilmente como los movimientos de los nudos 4, 5 y 6 se deducen por simetría a partir de las relaciones. Los desplazamientos se expresan en metros y los giros en radianes.

Una vez calculada y partiendo del sistema completo de ecuaciones, se calculan las reacciones en los nudos con movimientos prescritos.

Cogiendo como ejemplo la barra que contiene el nudo 1 y el 3 y modificando la ecuación 4.13, vemos que:

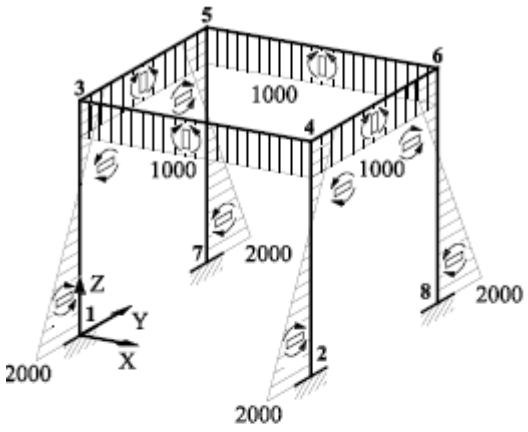
$$\widehat{\overrightarrow{\boldsymbol{F}}}_{\boldsymbol{13}} = \widehat{\boldsymbol{K_{aa}}}_{\boldsymbol{13}} \cdot \boldsymbol{T^{-1}} \cdot \overrightarrow{\boldsymbol{U}}_{\boldsymbol{1}} + \widehat{\boldsymbol{K_{ab}}}_{\boldsymbol{13}} \cdot \boldsymbol{T^{-1}} \cdot \overrightarrow{\boldsymbol{U}}_{\boldsymbol{3}}$$

Y solventando:

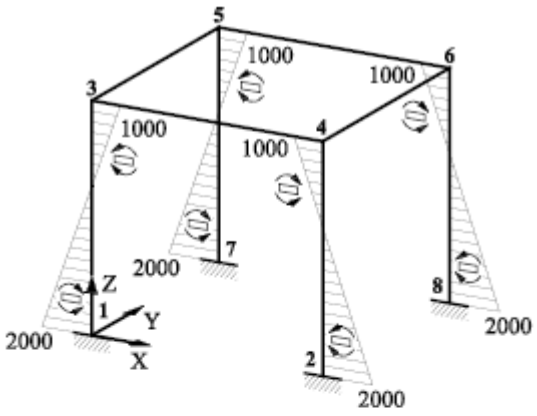
$$\overrightarrow{F_{13}} = \begin{Bmatrix} F_{13} \\ F_{14} \\ F_{15} \\ M_{16} \\ M_{17} \\ M_{18} \end{Bmatrix} = \begin{Bmatrix} F_{x3} \\ F_{y3} \\ F_{z3} \\ M_{x3} \\ M_{y3} \\ M_{z3} \end{Bmatrix} = \begin{Bmatrix} 500 \cdot 10^3 \\ 600 \\ 600 \\ 0 \\ -2000 \\ 2000 \end{Bmatrix}$$

Donde las fuerzas se expresan en N y los momentos en N·m. Nótese que los esfuerzos en el extremo 1 de la barra 13 coinciden con el valor de las reacciones en el nudo 1. Repitiendo el procedimiento se obtienen los restantes valores para los esfuerzos en los extremos de las barras, véase que el nodo 3 está siendo modificado por los siguientes 3 elementos elásticos

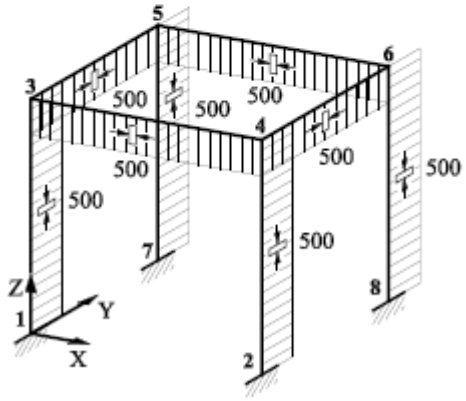
$$\overrightarrow{F_{13}} = \begin{Bmatrix} F_{x1} \\ F_{y1} \\ F_{z1} \\ M_{x1} \\ M_{y1} \\ M_{z1} \\ F_{x3} \\ F_{y3} \\ F_{z3} \\ M_{x3} \\ M_{y3} \\ M_{z3} \end{Bmatrix} = \begin{Bmatrix} 500 \cdot 10^3 \\ 600 \\ 600 \\ 0 \\ -2000 \\ 2000 \\ -500 \cdot 10^3 \\ -600 \\ -600 \\ 0 \\ -600 \\ 1000 \end{Bmatrix}; \overrightarrow{F_{34}} = \begin{Bmatrix} F_{x3} \\ F_{y3} \\ F_{z3} \\ M_{x3} \\ M_{y3} \\ M_{z3} \\ F_{x4} \\ F_{y4} \\ F_{z4} \\ M_{x4} \\ M_{y4} \\ M_{z4} \end{Bmatrix} = \begin{Bmatrix} 500 \cdot 10^3 \\ 0 \\ 0 \\ 0 \\ 1000 \\ 0 \\ -500 \cdot 10^3 \\ 0 \\ 0 \\ 0 \\ -1000 \\ 0 \end{Bmatrix}; \overrightarrow{F_{35}} = \begin{Bmatrix} F_{x3} \\ F_{y3} \\ F_{z3} \\ M_{x3} \\ M_{y3} \\ M_{z3} \\ F_{x5} \\ F_{y5} \\ F_{z5} \\ M_{x5} \\ M_{y5} \\ M_{z5} \end{Bmatrix} = \begin{Bmatrix} 500 \cdot 10^3 \\ 0 \\ 0 \\ 0 \\ 1000 \\ 0 \\ -500 \cdot 10^3 \\ 0 \\ 0 \\ 0 \\ -1000 \\ 0 \end{Bmatrix}$$



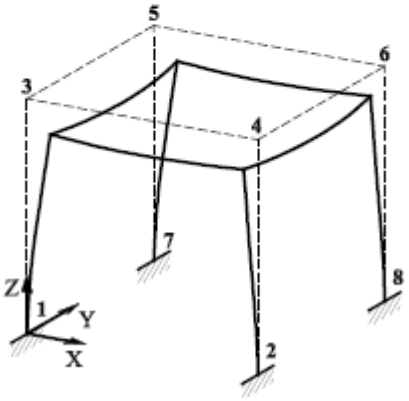
FLECTORES M_y [N×m]



FLECTORES M_z [N×m]



AXIL $\times 10^3$ [N]



DEFORMADA

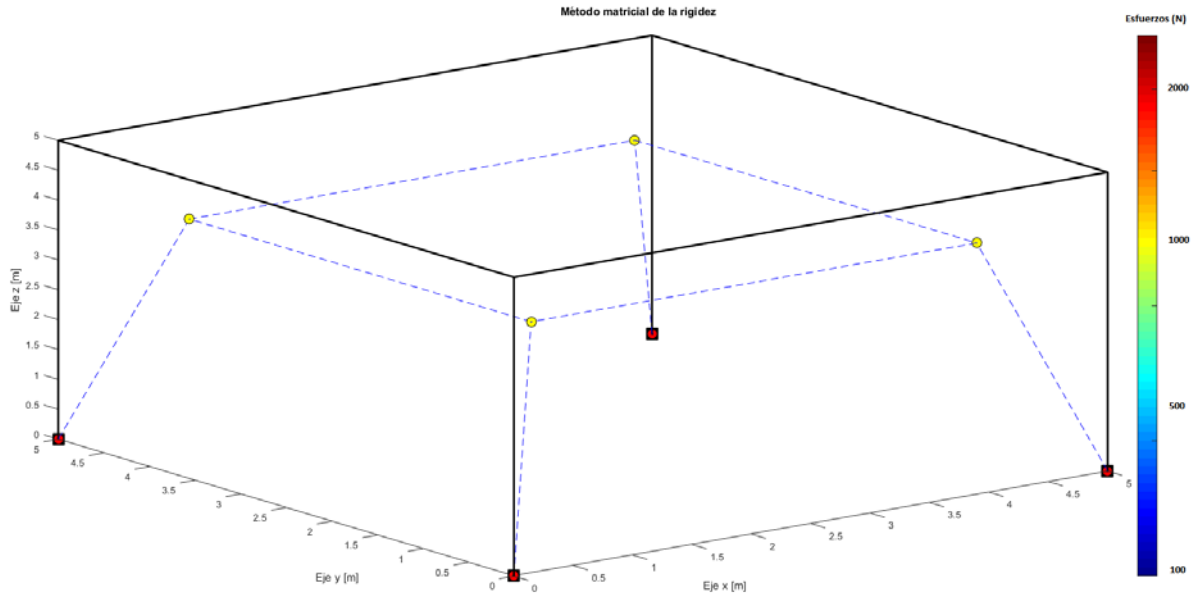


Figura 4.13 Diagrama de esfuerzos y visualizacion en matlab

Este ejemplo se puede ver en su formato original en la referencia bibliográfica [5]

Si bien un ingeniero está interesado en el cálculo de los esfuerzos que sufre su estructura lo está más en saber cuánto deformara, esto no es más que saber a qué tensión está sometida la estructura.

En este trabajo se focalizara más en el cálculo de tensiones y que tan lejos están de las tensiones máximas admitirles por el material

Una vez calculado los esfuerzos de los nodos en N y N·m nos encontramos que estos esfuerzos están referenciados sin tener en cuenta como está el material reaccionando con ellos. Existen distintos modos de clasificar estas tensiones, según generen tensiones normales o tangenciales

- La tracción y la flexión son tensiones normales
- La Cizalla y la tosían son tensiones tangenciales

De **las tensiones normales** podemos calcular la tensión axil que sufre un elemento elástico

$$\sigma_{axil} = \frac{Fuerza\ Normal}{Area} \tag{4.13}$$

NOTA: hay que tener en cuenta que si existe pandeo el problema se puede complicar de sobremanera

Las **tensiones de flexión** que sufre un elemento elástico, responder a la expresión de Navier

$$\sigma_{flex} = \frac{Mf}{I} \cdot y \tag{4.14}$$

Donde Mf es el momento flector y I es el momento de inercia a flexión. Y es la distancia entre el punto considerado más lejano a la fibra neutra en nuestro caso.

$$\sigma_{flex} = \frac{Mf}{I} \cdot y_{max} \tag{4.15}$$

Para tener una referencia a algunos momentos de inercia a flexión:

Sección circular maciza de diámetro d	Sección circular hueca de diámetro exterior D y diámetro interior d	Sección rectangular de altura h y base b
		
$I = \frac{\pi}{64} * d^4$	$I = \frac{\pi}{64} * (D^4 - d^4)$	$I = \frac{b * h^3}{12}$

Figura 4.14 Momentos de inercia transversales

Las **tensiones de cinceladura** son la distribución de tensiones tangenciales, este esfuerzo tiene un comportamiento parabólico.
Se puede definir un valor promedio de la tensión debida a la cortadura, pero como norma general se suele calcular a tensión máxima según el perfil.

Rectangular:

$$\tau_{max} = \frac{3}{2} \cdot \frac{C}{Area} \tag{4.16}$$

Circular:

$$\tau_{max} = \frac{4}{3} \cdot \frac{C}{Area} \tag{4.17}$$

Donde C es el esfuerzo generado por cortadura.

Otra tensión tangencial es la tensión debido a la torsión, este tipo de tensión depende mucho del tipo de perfil a utilizar, si son circulares, siguen las leyes de Hooke. Por lo contrario si no son circulares no verifican las leyes de Hook, y por lo tanto las deformaciones producidas no son proporcionales a las tensiones causantes.

Para una sección circular, la tensión tangencial debida a la torsión es una función lineal de la distancia al centro de la sección, de modo que para una fibra situada a una distancia y del centro es:

$$\tau_{tor} = \frac{Mt}{I_p} \cdot y_{max} \tag{4.18}$$

Donde Mt es el momento torsional y Ip es el momento polar de inercia
Ya teniendo en cuenta que donde queremos evaluar la tensión será en la zona más lejana a la fibra neutra., veamos distintos Ip según sección

Sección circular maciza de diámetro d	Sección circular hueca de diámetro exterior D y diámetro interior d
	
$I_p = \frac{\pi}{32} * d^4$	$I_p = \frac{\pi}{32} * (D^4 - d^4)$

Figura 4.15 Momentos de inercia polares

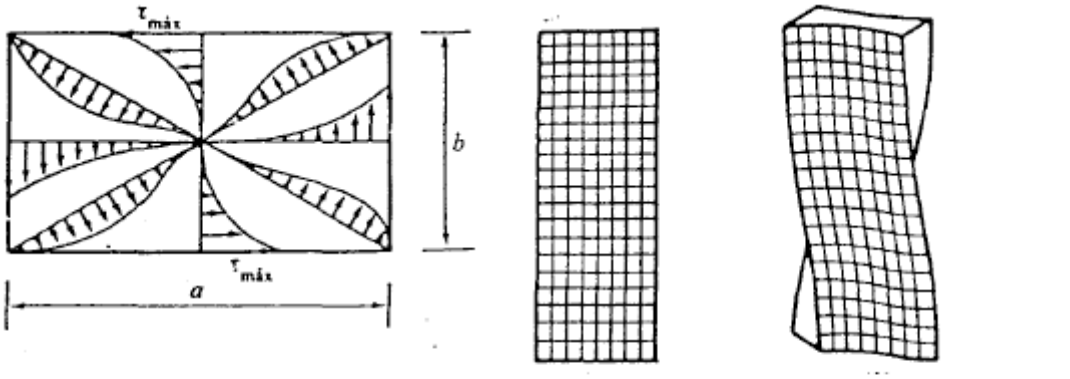
Para una sección no circular, tensiones y deformaciones no son proporcionales. Particularizando una sección rectangular, la tau máxima se presenta en el centro del lado mayor, y su valor depende de Mt y de que los valores de H y B pero también depende de la relación de aspecto h/B y lo hace de manera no lineal.

Tensión centro lado mayor:

$$\tau_{tor} = \frac{Mt}{\beta \cdot B^2 \cdot h} \tag{4.19}$$

Tensión centro lado menor:

$$\tau_{tor_{menor}} = \tau_{tor.} \cdot \alpha \tag{4.20}$$



a/b	1	1.5	2	2.5	3	4	6	10	∞
α	0.208	0.231	0.246	0.256	0.267	0.282	0.299	0.312	0.333
β	0.141	0.196	0.229	0.249	0.263	0.281	0.299	0.312	0.333

Figura 4.16 relación de aspecto y constantes de un perfil rectangular

Una vez calculadas las tensiones de cada elemento elástico vemos que están ubicados en planos y direcciones distintas, por ello debemos entender el principio del estado de tensiones.

Se considera el estado de tensión plano de un punto material, definido por los valores de las tensiones σx; σy y τxy
El estado tensional de un punto material de un elemento resistente sometido a solicitación es único, pero su expresión en tensiones depende del sistema de referencia que se considere.

Existen dos direcciones perpendiculares entre sí, en las que la expresión de dicho estado tensional solo tiene componentes normales: σ1 y σ2.
Dichas tensiones reciben el nombre de tensiones principales, y las direcciones correspondientes se llaman a su vez direcciones principales.

La expresión analítica de dichos valores es:

$$\sigma_{1,2} = \frac{\sigma_x + \sigma_y}{2} \pm \sqrt{\left(\frac{\sigma_x - \sigma_y}{2}\right)^2 + \tau_{xy}^2} \qquad (4.21)$$

En la figura 4.17 se puede ver esquemáticamente la explicación grafica de la ecuación 4.21

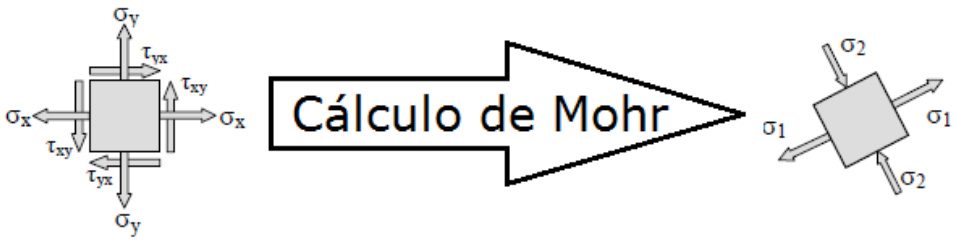


Figura 4.17 Representación gráfica de tensiones principales

Llegado a este punto, se teoriza la máxima energía de distorsión del sistema descrito, a modo de explicación, esta teoría se originó a partir de la observación de que los materiales dúctiles, sometidos a esfuerzos hidrostáticos, presentan resistencias a la fluencia mucho mayores que las obtenidas en el ensayo a tracción simple.

Los estudios concluyen que se puede considerar que la energía total de deformación tiene dos componentes: la energía relacionada con el cambio de volumen y la energía relacionada con la distorsión de la red cristalina, y se propone la siguiente teoría:
Se producirá el fallo cuando el estado de solicitación alcance el valor de la energía de distorsión presente en el ensayo de tracción cuando la tensión alcanza el valor de fallo.

Por este mismo hecho el autor escoge esta teoría para calcular la tensión equivalente del sistema.

La expresión analítica para el caso de tensión plana es:

$$\sigma_{eq} = +\sqrt{\sigma_1^2 + \sigma_2^2 - \sigma_1\sigma_2} \tag{4.22}$$

La aplicación del concepto de tensión equivalente de Von Mises, permite pasar desde un estado tensional expresado en sus tensiones principales, a un estado tensional uniaxial con una tensión de tracción ficticia, que produce un efecto equivalente a aquel.

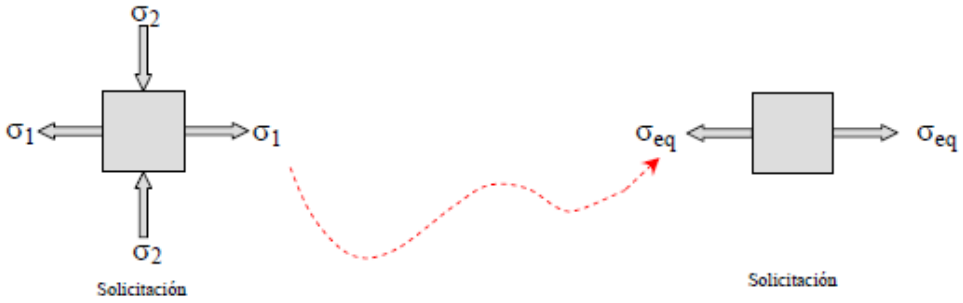


Figura 4.18 Representación grafica de la tensión equivalente

La teoría de la máxima energía de distorsión es menos conservadora que la teoría del esfuerzo tangencial máximo, y aun así es la más utilizada, porque sus resultados son igualmente válidos y su uso es el más sencillo porque requiere la utilización de solo una ecuación. Cabe decir también que esta teoría es mejor utilizarla en modelos de materiales dúctiles más que en materiales frágiles.

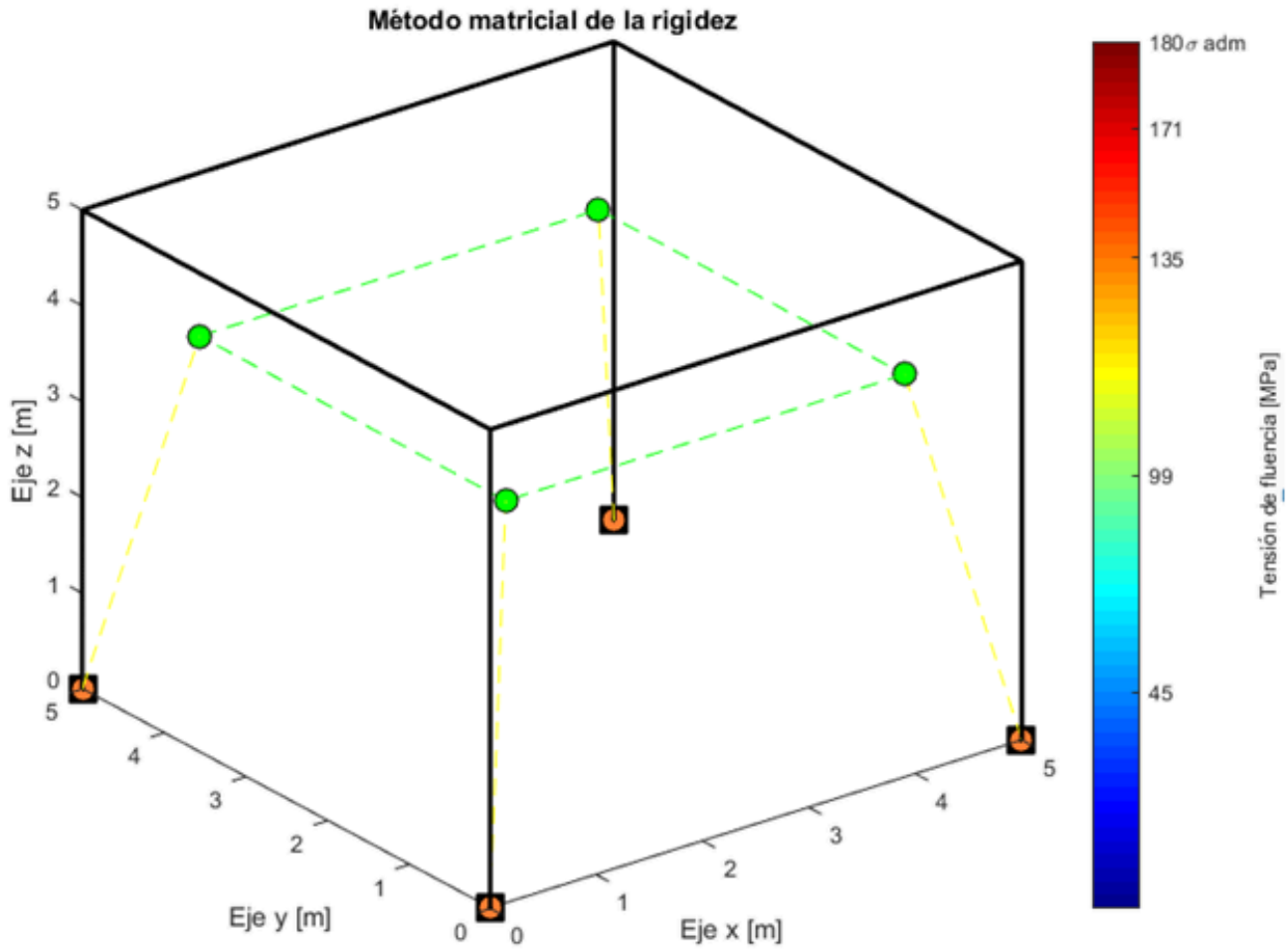


Figura 4.18 Representación gráfica de tensiones principales en MATLAB

CAPÍTULO 5: SIMULACIONES EN ESTADO ESTATICO DEL CHASIS

1.20. Introducción

En este capítulo se tratara de explicar el motivo por el cual es necesario el realizar simulaciones a la estructura principal de un vehículo de carreras. Las simulaciones en si no son más que el proceso de modelizar un sistema físico real y poder experimentar con él con la principal finalidad de comprender el comportamiento del sistema y así poder gobernarlo a fin de encontrar un punto óptimo sobre las variables que lo definen.

Existen diferentes tipos de simulaciones, en nuestro caso nos basaremos en las simulaciones físicas descritas a través de modelos matemáticos. La principal razón de utilizar este tipo de simulaciones computarizadas se han descrito en apartados anteriores, pero no va de más recordar que la gran ventaja de estas simulaciones es que pueden realizarse un enorme número de ellas a coste prácticamente cero en recursos económicos y temporales.

Este último no es el único motivo importante para simular la estructura principal, en realidad, si no existiera un motivo, un problema, no utilizaríamos tal herramienta. Por lo cual, en nuestro caso, si existe un motivo para el estudio y simulación.

El problema que se plantea al grupo de ingenieros que diseñan un vehículo de altas prestaciones es básico e inherente. La búsqueda ininterrumpida de alcanzar el límite sin sobrepasarlo. Esta lucha por mantenerse en el confín de un conjunto de sistemas es el sino de la ingeniería de alto nivel.

El principal problema de estudio se puede dividir en dos bloques que detallamos a continuación. Como ya es conocido en el campo de la ingeniería mecánica, el cálculo de elementos estructurales debe suponer el principal problema a solucionar

- Problema de tensiones: Se debe estudiar que la estructura aguante con cierto coeficiente de seguridad la tensión máxima admisible sin existencia de deformaciones plásticas remanentes. Se trata de un estudio complejo, ya que para poder calcular como es debido las tensiones, para ello se deben saber perfectamente las fuerzas y reacciones que perturban nuestro sistema. En el caso de un chasis de formula student, si selecciona cumplir las normas alternativas, es necesario certificar que la estructura puede soportar los esfuerzos que en caso de impacto puedan producirse.
  (Norma AF "*Alternative Frame Rules*" 2015 Formula SAE® Rules) *[1]*

  Además no solo estas tensiones son importantes, sino que un apartado de estas son las tensiones de rotura debida a fatiga, algo que es imperativo estudiar para conocer la vida del producto. Sabiendo también que un sistema dinámico que conste de masas, rigideces y viscosidades internas con carga variable, el estudio de vibraciones es necesario también

- Problema de deformaciones: Este posiblemente sea un problema más serio a estudiar, ya que en muchos tipos de estructuras, es más restrictivo certificar un cierto grado de deformación máxima debido a la tensión de trabajo, y este hace que la estructura sea más robusta, en contrapartida al peso. Esto significa que una estructura que solo se estudie para que aguante las tensiones de diseño, muy posiblemente deformen más de lo que desearía el ingeniero. Por la cual cosa nos centraremos en certificar cierta deformación máxima

  El valor esta deformación máxima es algo que se debe estudiar con conocimiento, cualquier elemento que sufra tensión, deformara inevitablemente pero, ¿cuánto es el límite aceptable? Es misión de este trabajo el poder responder a esta pregunta en los temas venideros.

Por este último motivo, como a problema planteado desde el punto de vista de ingeniería mecánica, debemos estudiar las deformaciones y posteriormente comprobar las tensiones.

Por ello se ha planteado dos problemas que resuelvan estas dos incógnitas mencionadas.

1.21. Rigidez torsional

Desde el punto de vista de la dinámica vehicular, el trabajo de un ingeniero es cerciorar que el comportamiento del vehículo es el adecuado en cualquier punto del circuito. Aquí es donde entra el concepto de rigidez torsional.

La rigidez torsional del chasis es relevante para ayudar a entender cómo los pares de ruedo de los ejes delantero y trasero interactúan a través del chasis.

Modificando del reparto de ruedo entre ejes es una forma muy común de modificar el equilibrio de un vehículo. Si el chasis es demasiado débil, la rigidez torsional es demasiado baja; cualquier diferencia en la rigidez al balanceo entre la parte delantera y trasera será absorbida como la torsión del chasis. En un coche de carreras, esto a veces puede ser observado como un coche que no responde a los cambios en la distribución o “Set Up”. Algo que debemos de tener en cuenta, pero que no podemos ver a simple vista.

En Motorsport se pueden calificar los vehículos de dos maneras distintas, referentes a su rigidez.

- Monoplazas y prototipos: este tipo de vehículos tienen una alta resistencia torsional, valores aproximados entre 10kN/º - 20kN/º Esto permite al ingeniero de pista poder variar en 1mm las cotas de “Set Up” del vehículo y que el mismo cambie totalmente su comportamiento, gracias a que la estructura no difumina interiormente esa deformación o cambio.

- Turismos: Los vehículos con una baja resistencia torsional son susceptibles a difuminar / absorber el cambio realizado al mismo, con la negativa consecuencia de poder ajustar bien el “Set Up”

En esta simulación el objetivo es calcular los desplazamientos de los nodos causados de cierta manera cumpliendo la normativa SAE.

SAE TECHNICAL PAPER SERIES 2002-01-3300 [2]

En resumen, a un recomendando su **lectura**, se realiza este ensayo porque en la situación de un coche de carreras, el paso por curva y las variaciones de alturas debidas a pianos y baches de la pista son los esfuerzos más significativos que sufre la estructura. Este experimento que simulamos se basa en el principio de torsión de una ménsula normal, en la figura 5.1 se puede observar lo comentado, una fuerza torsora (par de fuerzas) que deforman una estructura empotrada en su otro extremo.

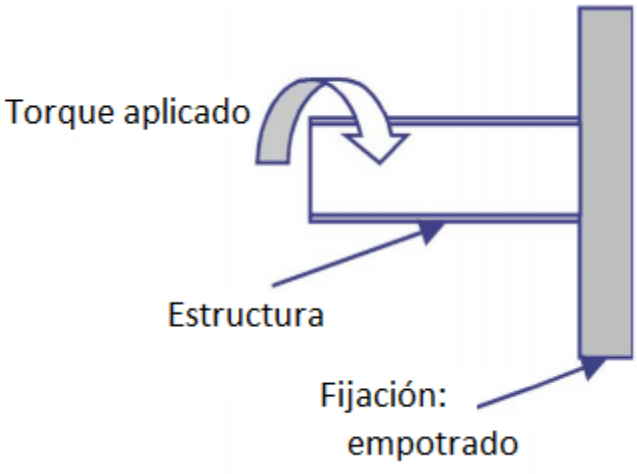


Figura 5.1 *Tubo torsionado*

Para poder recrear estas mismas condiciones pero en nuestro modelo de estructura de un vehículo, debemos restringir todos los grados de libertad de los nodos más posterior, y a su vez aplicar un par de fuerzas con tal de generar una torsión. Escogeremos los nodos de suspensión más frontales, para simplificar el modelo y porque estos acotan perfectamente las condiciones de uso del vehículo en pista. Podemos ver un ejemplo de cómo quedaría la estructura con estas condiciones de contorno aplicadas en la figura 5.2

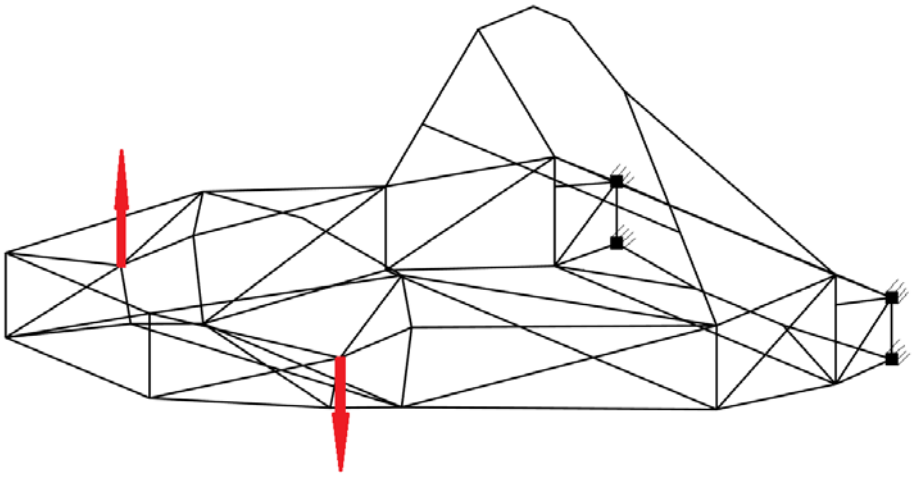


Figura 5.2 *Condiciones de contorno del modelo*

Una vez completada la simulación debemos calcular la torsión analíticamente.

Cabe recordar que la magnitud de fuerza simétrica aplicada es indiferente, ya que se está buscando un factor, pero el autor recomienda unas magnitudes de fuerza similares a las reales y estas rondan magnitudes entre 0.5KN – 5KN

La rigidez se puede calcular a través de la ecuación 5.1 que relaciona la fuerza torsora aplicada con el ángulo que esta genera.

$$K_{rigidez} = {}^{T}/_{\theta} \tag{5.1}$$

Y para poder encontrar el ángulo generado y la magnitud de torsión que generan nuestros pares de fuerza nos guiamos por la siguiente ecuación.

$$K_{rigidez} = \frac{P\,(L_1 + 2\,L_2)}{\tan^{-1}\left(\frac{\Delta a + \Delta b}{L_1}\right)} \tag{5.2}$$

Dónde:

P: Fuerza aplicada en el extremo

L1: Distancia de los puntos a estudiar (A y B)

L2: Distancia del centro al punto de aplicación de la fuerza P

$\Delta_{a/b}$: Deformación producida en eje Z de los puntos a estudiar

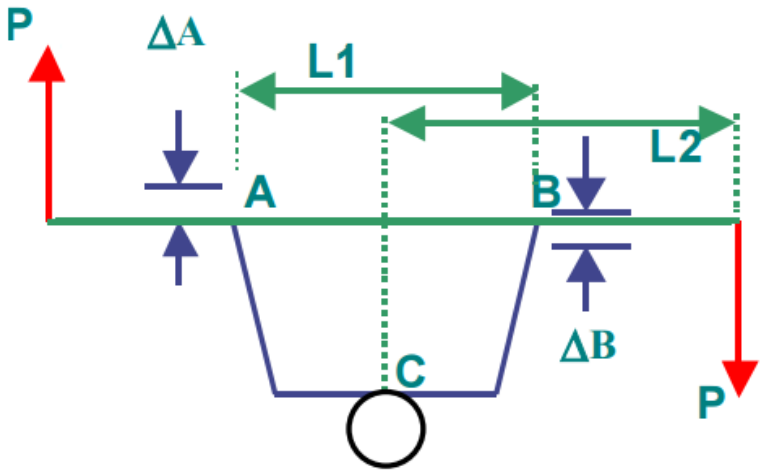


Figura 5.3 *Puntos frontales de la suspension y cargas*

1.21.1. *Valor de rigidez de referencia*

Como paso final el autor ve necesario generar un último coeficiente en razón a poder optimizar el sistema.

Como se ha explicado anteriormente debe existir una resistencia torsional que satisfaga un buen control por parte del piloto y por parte del ingeniero.

Eso se consigue entendiendo como se reparten las fuerzas en los neumáticos, y como las rigideces de los muelles y de las barras antiroleo influyen en estas.

El la figura 5.4 podemos observar en la imagen, como sería un sistema simple de suspensión de un vehículo y este se expone de manera que todos los elementos trabajasen de forma torsional, es decir, se convierte la constante de un muelle elástico de compresión/tracción a un muelle con constante torsional.

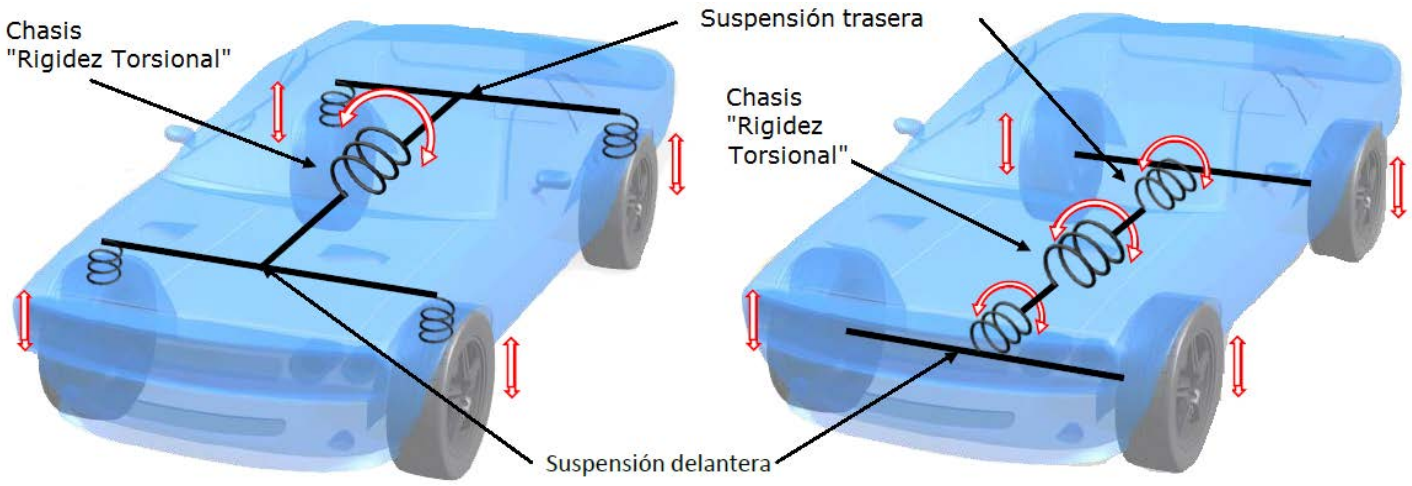


Figura 5.4 *Influencia de rigideces*

Una vez convertido el sistema en un sistema que pose constantes de muelle en serie se puede llegar a realizar el siguiente cálculo.

Pongamos que se tiene unas rigideces delanteras y traseras de distinto valor donde:

$$Ks_{front} = Rigidez\ de\ los\ muelles\ delanteros$$

$$Ks_{rear} = Rigidez\ de\ los\ muelles\ traseros$$

$$K_{Chasis} = Rigidez\ torsional\ de\ la\ estructura\ principal$$

Para simplificar la teoría, se escogerá aquellos muelles que sean más resistentes, ya que esto generara una situación más restrictiva.

Entones sabemos que si el sistema actúa en serie la Constante de rigidez total es función de:

$$K_{rigidez\ Total} = \frac{Ks \cdot K_{ch}}{Ks + K_{ch}} \tag{5.3}$$

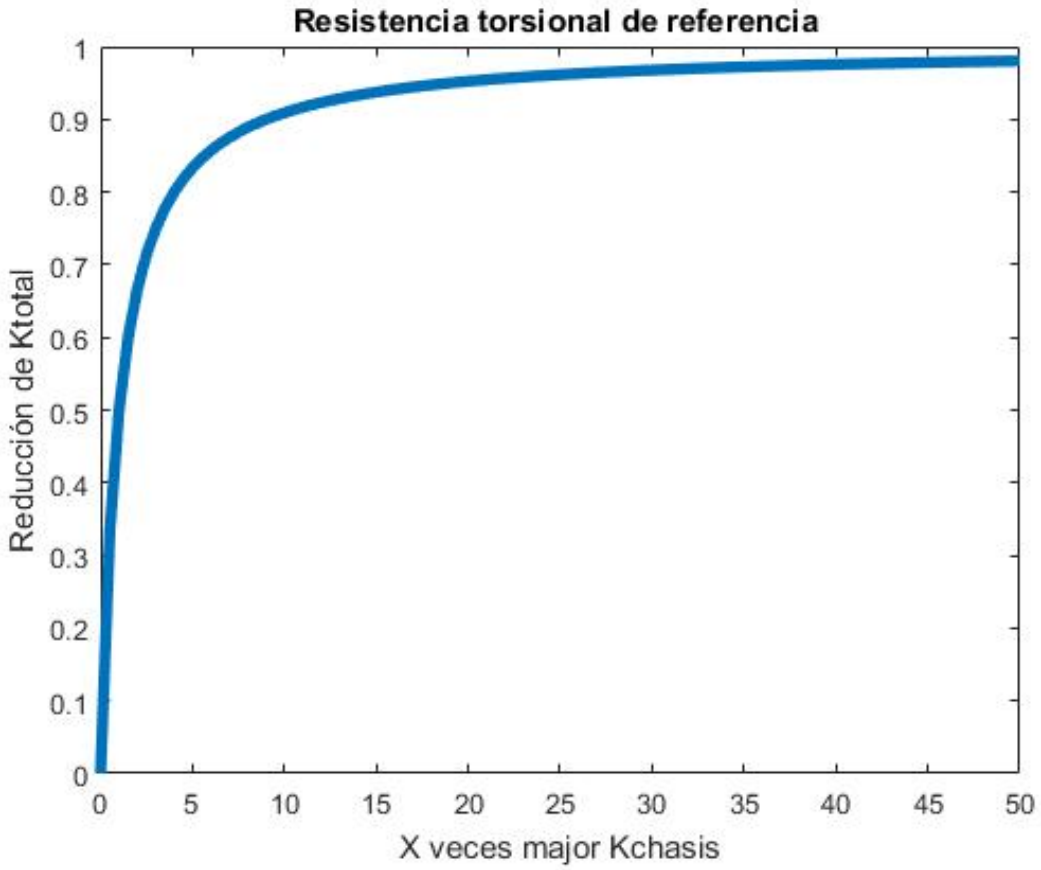


Figura 5.5 *Reducción de rigidez total*

Como podemos ver en la figura 5.5, si se tiene en cuenta la rigidez de la estructura principal, está siempre disminuye la rigidez total del sistema. Entonces llegado a este momento se decide minimizar está perdida de rigidez.

Es lógico pensar que cuanta más resistencia torsional tenga el chasis mejor, para minimizar el error, esto matemáticamente es totalmente cierto, pero si tenemos en cuenta que a más resistencia torsional, más pesado será el chasis, esto dará lugar a un empeoramiento del tiempo por vuelta, cosa que no será compensada

por una buena dinámica vehicular o “SetUp”.

Entonces parece más lógico ver en qué punto la perdida de rigidez empieza a estabilizarse y a ser menor esta ganancia. Esto se ve derivando la gráfica anterior, que se muestra en la figura 5.6

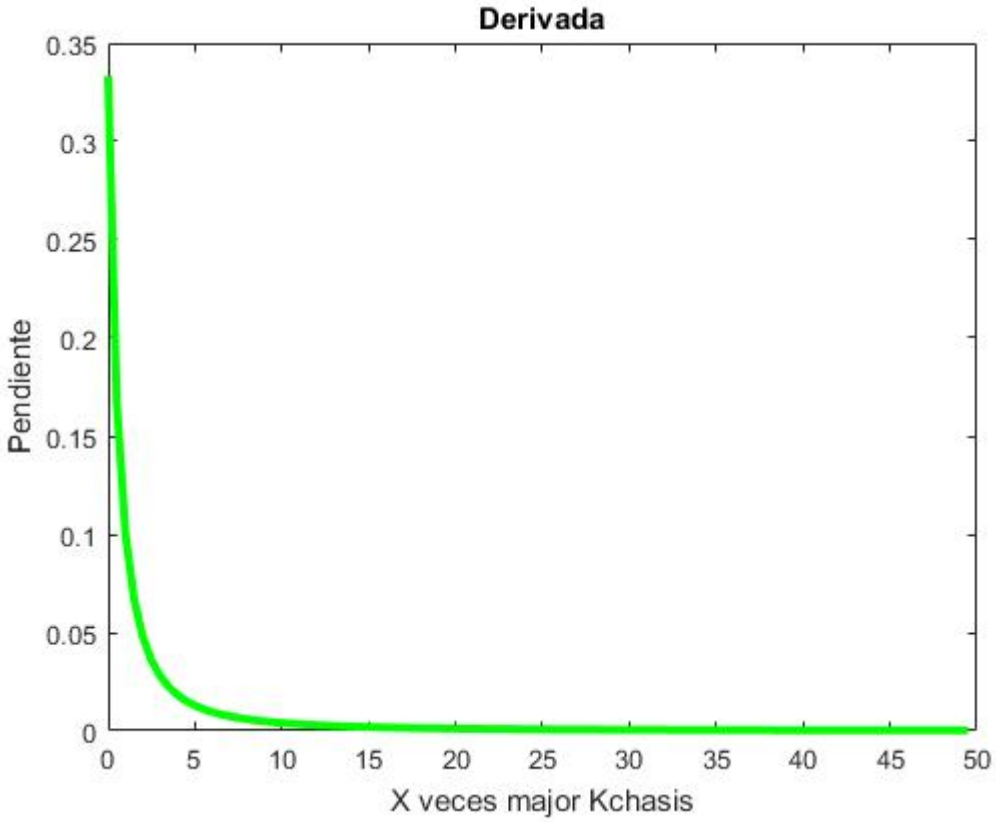


Figura 5.6 *derivada grafica* 5.5

La conclusión que podemos sacar estudiando las gráficas 5.4 y 5.5 es que la pendiente a partir de 5 veces más Kchasis que Ksuspensión empieza a ser constante, entonces ese debería ser nuestro punto óptimo. Estudiando la gráfica de referencia vemos que 5Kc generan mucho error, 0.833 que es un 16.7% de error para ser exactos, si escogemos **10Kc** que se encuentra a un punto de la derivada menos óptima nos genera **0.9091 un 9.09%** de error, valor que resulta cómodo ya que solo debemos aumentar la rigidez del chasis unas 10 veces la mayor de las suspensiones. Para encontrar un fallo del 5% debemos ir a buscar un valor de 19 veces superior.

NOTA: el valor de rigidez torsional puede ser tomado entre 5 y 20 veces la rigidez de suspensión. Dependiendo de la diferencia de pesados y la implicación de este en tiempo por vuelta, se puede escoger un valor menor para ganar tiempo por vuelta y tenerlo en cuenta en el cálculo de comportamiento dinámico del vehículo.

Llegados a este punto y seleccionado una resistencia torsional aceptable debemos clasificar la estructura a través de su frecuencia natural, esta nos clasificara las distintas estructuras que tengan una misma rigidez torsional, es decir que se clasifican mediante su rigidez especifica.

Esta constante se debe tomar con precaución, ya que un óptimo valor ente rigidez y masa puede significar el doble de peso a doble de rigidez, por lo que el ingeniero debe intentar fijar un valor mínimo o máximo como ya hemos explicado anteriormente, para que tenga sentido físico.

$$\omega_n = \frac{1}{2\,\pi} \cdot \sqrt{K_{rigidez}/_{Masa}} \tag{5.3}$$

Gracias al término de las frecuencias naturales del chasis podemos cuantificar la optimización teniendo en mente que el chasis debe estar pesando lo mínimo posible, es decir, que solamente se introduzca una unidad de peso si esta mejora las prestaciones del vehículo. Este último ejercicio es la clave del diseño competitivo y por ende la más complicada de todas.

Finalizado esta explicación, se le pueden venir al lector la pregunta siguiente:

¿Es suficiente tal resistencia torsional para no deformar los puntos de anclaje del chasis?

Esta es una pregunta que todo ingeniero debe preguntarse y resolver.

Como ya hemos visto anteriormente las deformaciones son generadas a través de unas fuerzas aplicadas así como sus apoyos, es fácil ver que las fuerzas aplicadas que debe soportar un F1 no son las mismas que las soportadas por un turismo en una superficie deslizante, para mostrar dos extremos. Dicho de otro modo, para tener la misma deformación en un F1 que en un turismo las rigideces de torsión deben ser distintas en proporción a las fuerzas aplicadas.

Por ello, en este trabajo se estudiara y se plasmara un resultado que tenga en cuenta las fuerzas aplicadas en una estructura y su deformación, referenciadas a una resistencia torsional.

1.22. Máxima solicitud real

En el apartado anterior hemos visto que es necesario calcular el valor real de las deformaciones cuando interviene una fuerza real, cabe decir que real en ingeniería solo puede tener un 5% de desviación a la simulada.

Es imperativo entender como ingeniero de diseño de vehículos de competición que las principales reacciones que puede soportar el conjunto se deben a las huellas de contacto de los cuatro neumáticos, y a interacción al paso por el fluido aire. Por ello debe entender que cada modelo de neumático es distinto a otro, así como la interacción aerodinámica.

Para poder obtener estas fuerzas, se puede abordar el problema desde dos perspectivas distintas.

La primera perspectiva y la más alejada de nuestro complejo sistema es la de obtener datos de aceleración con un sensor acelerómetro bien posicionado e instalado y la segunda es entrar en un nivel profundo de modelización matemática de los estados estático/dinámico y cinemáticos mediante ecuaciones físicas del vehículo.

1.22.1. *Método empírico*

El modelo empírico quizás sea el modelo generalista que más conviene utilizar por su simplicidad matemática, es un modelo de investigación científica, que se basa en la experimentación y la lógica empírica, que junta la observación de fenómenos y su análisis estadístico. Dicho de otro modo, si se consigue fabricar un prototipo real, la experimentación con este resulta muy sencilla y se obtienen valiosos datos, tanto para verificar modelos matemáticos como para un proceso de diseño posterior

Posicionando un acelerómetro, con corrección giroscópica, justo en el punto coordenado del centro de gravedad podemos analizar en todo momento del circuito el vector junto a su magnitud de la aceleración. Además como ya sabemos el peso real del vehículo con la ecuación 5.4 podemos averiguar la fuerza en la que los neumáticos están reaccionando con el asfalto. Nótese que la variable aceleración se simplifica como "*acc_x*" para evitar confusión con la distancia del CG al eje delantero llamada como "*a*"

$$F = m \cdot acc_{y,x,z} \tag{5.4}$$

Podemos poner un ejemplo de cómo son las aceleraciones tanto longitudinales como laterales, estas se muestran en la figura 5.4, sacadas de la bibliografía [4]

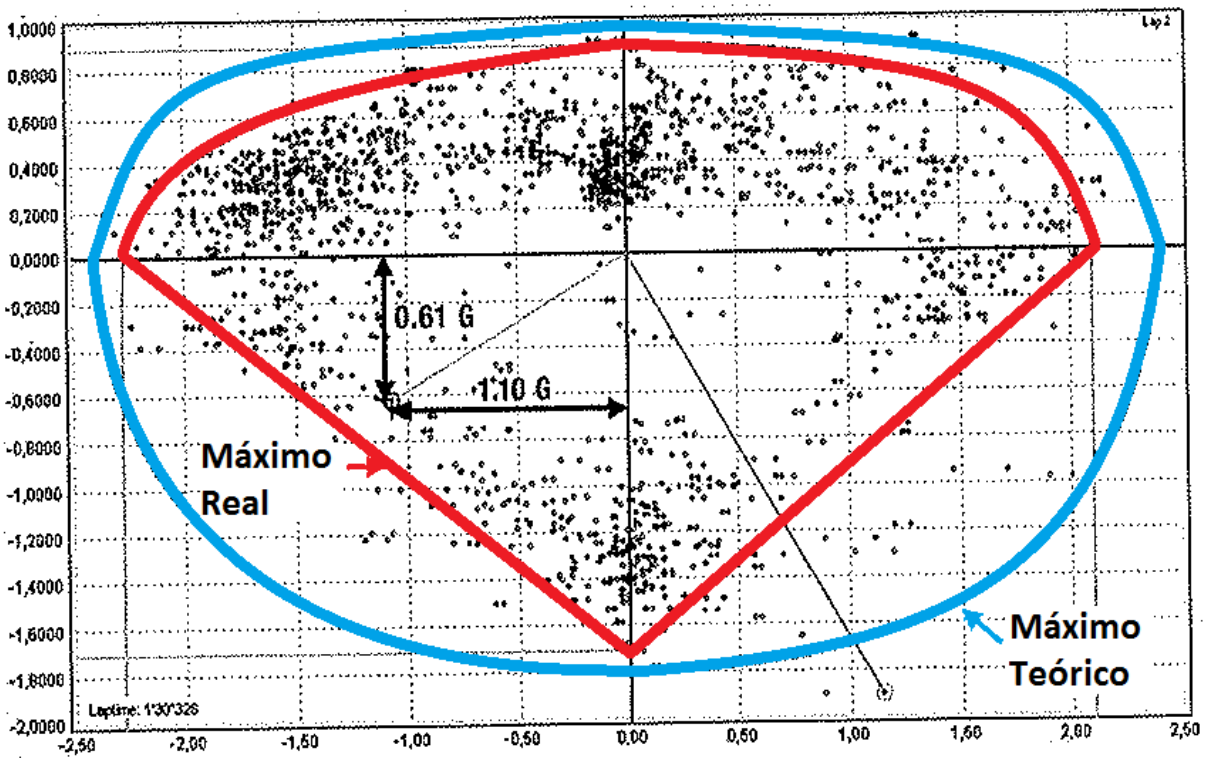


Figura 5.4 *Diagrama* G·G *Teoricoy Real* [4]

Como podemos observar en el la figura 5.4 estos son los típicos datos que se adquieren en un coche de carreras, cabe decir que las magnitudes máximas, tanto teóricas como reales varían dependiendo de cargas aerodinámicas, tracción a 2 ó 4 ruedas, Etc.

Para interpretar el diagrama, el eje coordenado muestra un coche sin aceleración alguna, en el eje horizontal, se pueden observar las aceleraciones laterales que sufre el vehículo, en el eje vertical positivo son las aceleraciones positivas, es decir cuando aceleras, las negativas como es lógico son las deceleraciones o aceleraciones negativas.
Se puede observar que normalmente un vehículo de estas características desacelera con mayor magnitud que acelera, esto es debido a que tienen dos reacciones (neumáticos) más decelerando, que acelerando.

1.22.2. *Modelo matemático*

Un modelo matemático se emplea para expresar relaciones, proposiciones sustantivas de hechos, variables, parámetros, entidades y relaciones entre variables de las operaciones, para estudiar comportamientos de sistemas complejos ante situaciones difíciles de observar en la realidad.

Se trata de un camino, una opción muy laboriosa y complicada. Pero en última instancia es extremadamente necesario tener consolidado un modelo que resuelva preguntas comprometidas. Dicho de otro modo, un correcto modelo matemático debería ser capaz de calcular el punto óptimo de tiempo por vuelta, por ejemplo, entre aumentar la carga de sustentación negativa y a la vez, el peso total del vehículo y la fuerza de arrastre.

Para solucionar este problema, el modelo debe ser capaz de en distintas áreas, como, dinámica vehicular, aerodinámica, etc. Todas estas a la vez y de manera iterativa.

Dentro del conjunto de dinámica vehicular, debe existir un modelo que describa el comportamiento de la estructura del vehículo, objetivo de este trabajo.

Pero para conocer las fuerzas de reacción que se generan, es decir, conocer las variables con menor perspectiva, las que describen un sistema complejo con pocas variables, es necesario introducirnos primero en el estudio del comportamiento del neumático así como el comportamiento dinámico de las fuerzas que debe reaccionar el neumático.

1.22.3. *Valor de deformación de referencia*

Como es lógico pensar, una fuerza aplicada a una estructura, genera una deformación, esta deformación en según qué situación puede ser más o menos critica.

En el interior de la estructura, estas pequeñas deformaciones pueden no tener una gran implicación a la dinámica vehículo, en cambio posiblemente si se trata de un “Packaging” reducido debería tenerse en cuenta.

Donde si realmente es importante ver la deformación de la estructura es en los puntos de anclaje de las suspensiones, ya que estos generan una cinemática calculada de los trapecios o de cualquier configuración de suspensión, esta si se deforma cambiara la cinemática y por ende el punto óptimo de trabajo de los neumáticos. En según qué situaciones esto puede ser importante, y por ello debe ser estudiado.

El mayor problema de estudiar esta deformación es la complejidad que dan los cálculos de los neumáticos, por ello en este trabajo no se dará un valor de referencia, pero si se implementara en el programa una herramienta capaz de simular el comportamiento real de solicitud de fuerzas de nuestra estructura.

Estas solicitudes reales no son más que las provocadas por las masas de cada uno de los elementos que la estructura principal debe soportar. En ingeniería es una práctica habitual agrupar todos estos elementos másicos en un único elemento llamado Centro de Gravedad. Eso se hace así para disminuir y/o facilitar el cálculo del problema planteado.

En nuestro caso, no simplificaremos tanto, en vez de agrupar todos los elementos másicos en uno solo, lo que se propone es agruparlos según departamento, es decir, cada departamento agrupara todos los elementos másicos que posee y dará como resultado un Centro de Gravedad de ese departamento, entonces así, se tendrán tantos CG como departamentos.

Cabe recordar que la suma o agrupación de estos CG daría como resultado un CG único.

NODOS	X[mm]	Y[mm]	Z[mm]	Masa [Kg]
Package	1630,89	17,26	206,35	54,90
Powertrain	2059,38	0,00	256,79	40,90
Refri	1601,58	-217,03	247,33	6,60
Ergonomía	1161,81	0,00	324,38	78,20
Aero	889,28	0,00	140,19	16,00
Transmi	2259,38	0,00	244,00	17,60
Sus/Dir	1064,77	0,00	209,46	7,27
Ruedas	1290,50	0,00	73,00	0,20
Electrónica	1442,26	0,42	248,47	8,00

Figura 5.5 *Centros de gravedad departamentales DATOS*

Como se puede observar en la figura 5.5, cada departamento debe agrupar en la hoja de cálculo la posición y la masa que contiene su CG respectivo.

Una vez elaborado esto el programa responderá a esta información como se muestra en la figura 5.6 se puede observar que cada departamento tiene un tamaño y color acorde con su departamento y masa.

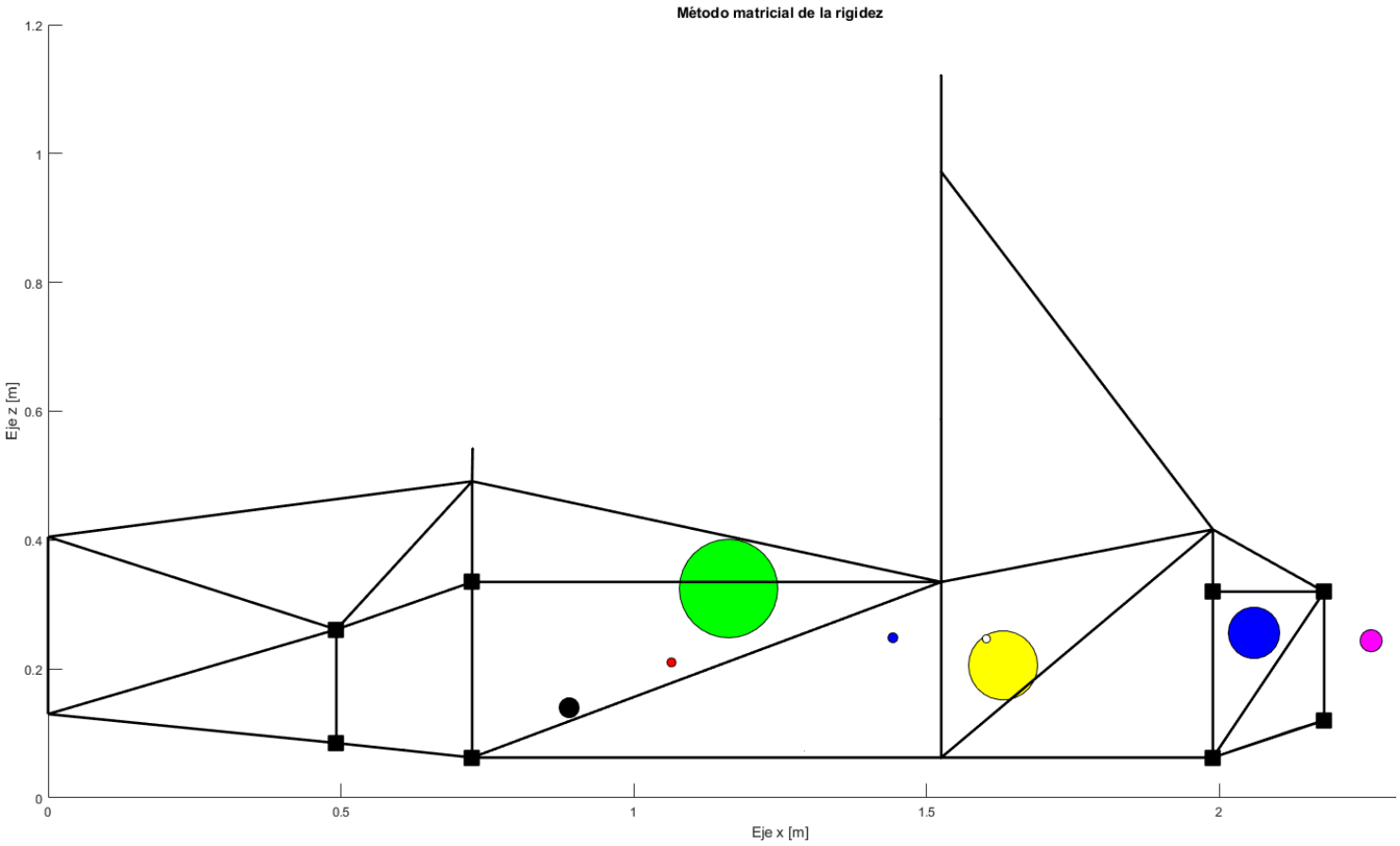
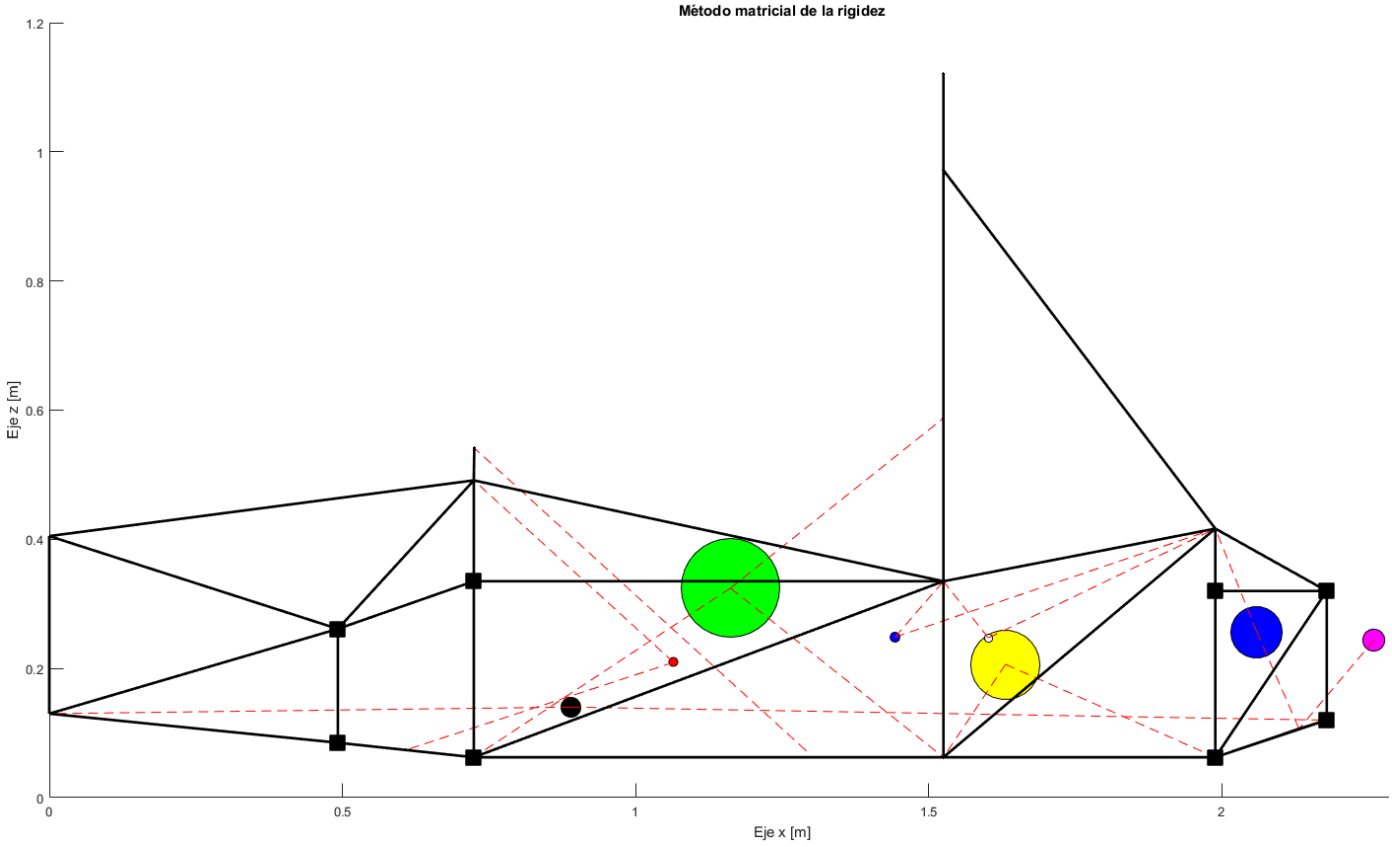


Figura 5.6 *Centros de gravedad departamentales PLOT*

Una vez explicada la posición de los Centros de Gravedad de cada departamento. Es en estos donde nosotros aplicaremos un vector de aceleración, que nos resultara, multiplicado por la masa, una fuerza total. Esta fuerza no es más a la que llamamos, fuerza real, ya que es la fuerza física real que los elementos másicos están sufriendo y estos aran tensionar y deformar la estructura del problema.

Para que estos CG se puedan comunicar con la estructura principal, debemos crear unos elementos que llamaremos “BarrasCG” estos elementos rígidos tienen la misión de transmitir la fuerza remotamente aplicada a los nodos reales de la estructura, con esto conseguimos que los nodos que soportan los elementos másicos de los distintos departamentos, transmitan el vector de fuerzas y momentos correctamente a la estructura principal y por ende podamos calcular las tensiones y deformaciones de esta.



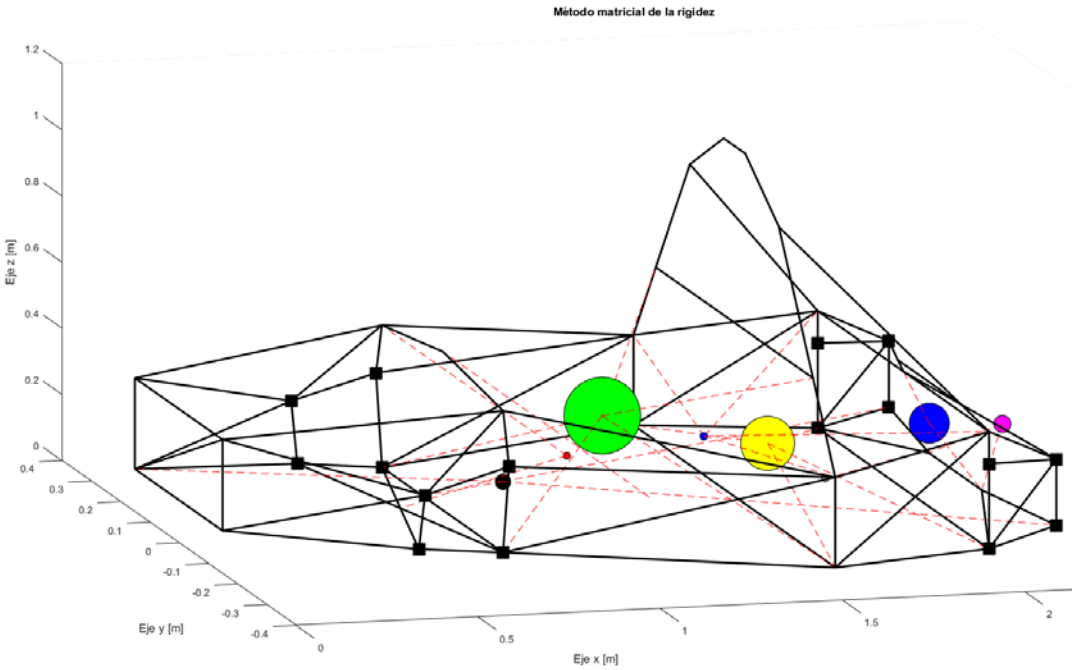


Figura 5.7 *Elementos rijidos*

Hasta aquí ya tenemos definido el núcleo del programa de solicitaciones reales, ya que con este código podemos calcular lo anteriormente propuesto. Además se pueden calcular también las cargas cíclicas de la estructura en verso a unes solicitudes de una pista de carreras, como se ha visto en la figura 5.4, pero esto lo detallaremos en el siguiente apartado.

CAPÍTULO 6: PASO A PASO DEL CÓDIGO

A continuación en este capítulo se mostrara detenidamente línea por línea el código principal del programa, para la obtención de deformaciones de una estructura reticulada en 3 dimensiones **Figura 6.1** mediante el método directo de rigidez. El método y código se basa en lo explicado en el **Capítulo 4** para resolver los desplazamientos y así completar toda la explicación para que el lector pueda, por sus propios medios recrear y entender con exactitud el programa. Con este capítulo daremos por finalizado el primer bloque del trabajo.

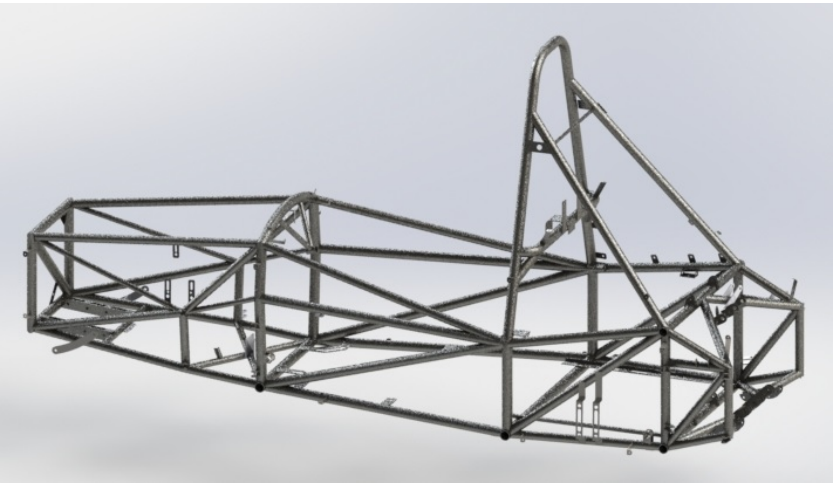


Imagen 6.1 *Estructura multitubular del EV-a, Formula Student EUETIB e-Tech Racing*

Para que el lector pueda implementar rápidamente el código en un programa de Matlab este capítulo se ha amoldado a las características internas del programa Matlab, por ello a continuación se mostraran recuadros de texto de color gris, estos recuadros son extractos de código que el usuario debe interpretar secuencialmente. Esto significa que el código se ejecuta de recuadro en recuadro no teniendo en cuenta las explicaciones que el autor introduce entremedio. Además Matlab permite añadir comentarios después de un paréntesis (%) y a continuación el texto que aparece en *verde* no será computable.

1.23. Puesta a punto del entorno de trabajo

El código mostrado a continuación esta para eliminar variables de códigos anteriores que pueden molestar al código posterior.

- Clear ALL : Elimina los objetos del *WorkSpace* , liberandolas de la memoria del sistema
- Clc : limpia la ventana de comandos (*Command Window*)
- Format shortEng : muestra la salida de datos del *Command Window* en notación ingenieril, con exponentes múltiples de 3, con 4 dígitos después de la coma decimal

```
clear ALL
clc
format shortEng
```

1.24. Adquisición de la Geometría

Para facilitar la comunicación entre usuario y maquina se ha previsto guardar los datos de manera más fácil y visual en un archivo de Microsoft Excel. Este apartado del programa interpreta esta base de datos, leyéndola y guardando sus variables en el *WorkSpace* del propio Matlab

```
file=('DSM16BASELANE.xlsx'); %--> Imput modificado por usuario. BASE DE DATOS
```

Se puede observar que el primer valor es el nombre del archivo que contiene los datos, y solamente cambiando el nombre en la línea de código superior, este ya será modificado en todo el programa, facilitando así el trabajo del usuario

```
nodos= xlsread(file,'B2:D1000')*1e-3; % mm a Metros
barras= xlsread(file,'F2:G100');
diametros= xlsread(file,'J2:k100');
opti_codigo= xlsread(file,'I2:I100');
optidata= xlsread(file,'Secciónes','C72:D84');
n=length(nodos);
m=length(barras);
```

Veamos en detalle que variables ha guardado en el *WorkSpace* el programa en un formato comprimido

```
format short,nodos,barras
```

nodos =

0	-0.1500	0.1300
0	-0.1500	0.4050
0	0.1500	0.4050
0	0.1500	0.1300
0.4922	-0.2080	0.0850
0.4922	-0.2288	0.2613
0.4922	0.2288	0.2613
0.4922	0.2080	0.0850
0.6081	0	0.0738
0.7240	-0.2070	0.0625
0.7240	-0.2275	0.3350
0.7240	-0.2075	0.4913
0.7248	0	0.5413
0.7240	0.2075	0.4913
0.7240	0.2275	0.3350
0.7240	0.2070	0.0625
1.5250	-0.3450	0.0625
1.5250	-0.3450	0.3350
1.5250	-0.2688	0.5875
1.5250	-0.1522	0.9716
1.5250	-0.0370	1.1220
1.5250	0.0370	1.1220
1.5250	0.1522	0.9716
1.5250	0.2688	0.5875
1.5250	0.3450	0.3350
1.5250	0.3450	0.0625
1.9888	-0.2930	0.0625
1.9888	-0.2930	0.3200
1.9888	-0.2930	0.4165
1.9888	0	0.4165
1.9888	0.2930	0.4165
1.9888	0.2930	0.3200
1.9888	0.2930	0.0625
2.1316	-0.0900	0.1067
2.0601	-0.1076	0.3802
2.0601	0.1076	0.3802
2.1316	0.0900	0.1067
2.1788	-0.2865	0.1200
2.1788	-0.2865	0.3200
2.1788	0.2865	0.3200

```
barras =

     1     2
     2     3
     3     4
     4     1
     4     2
     1     5
     1     6
     2     6
     2    12
     3    14
     3     7
     4     7
     4     8
     5     6
     7     8
     5    10
     5     9
     6    11
     6    12
     7    14
     7    15
     8     9
     8    16
    10    11
    11    12
    12    13
    13    14
    14    15
    15    16
    16    10
    10    17
    10    18
    11    18
    12    18
    14    25
    15    25
    16    25
    16    26
    17    18
    18    19
    19    20
    20    21
    21    22
    22    23
    23    24
    24    25
    25    26
    26    17
     9    10
     9    16
    17    27
    17    29
    18    29
    20    29
    23    31
    25    31
    26    31
    26    33
    27    28
    28    29
    29    30
    30    31
    31    32
    32    33
    33    27
    27    38
    27    39
    28    39
    29    39
    39    35
    36    40
    31    40
    32    40
    33    40
    33    41
    27    34
    35    36
    37    34
    38    39
    40    41
    41    37
    38    34
    35    30
    36    30
    19    24
    33    37
```

La variable nodos ha sido pasada de milímetros a metros, para que la introducción de datos en Excel sea más familiar para el usuario

1.25. Condiciones de contorno

Dependiendo del caso a estudiar, las condiciones que acotan o definen totalmente al sistema, serán dependiendo del problema que se desee solventar, en este capítulo los problemas que desearemos resolver serán los siguientes:

- Rigidez torsional
- Aceleraciones laterales y/o longitudinales
- Impactos frontales y laterales

Rigidez torsional

En este caso y como ya se ha explicado el método a utilizar en temas anteriores, queremos calcular la rigidez de la estructura.

- *Fuerzas*

```
CCF=[6,3,-500;7,3,500]; %--> Imput modificado por usuario. FUERZAS DEL SISTEMA
```

En la anterior línea de código, **el usuario nada más debe modificar la primera columna de la matriz**, esta es la que define en que nodo esta aplicándose una fuerza Z de una magnitud positiva de 500N, es decir, esta matriz en la posición (i,1) almacena el número del nodo, la posición (i,2) afecta al 3er grado de libertad, siendo Fz y la última posición (i,3) representa la magnitud en Newton

Dónde (i, 1) se refiere a cualquier fila columna 1

```
F=zeros (n*6, 1);

for i=1: size (CCF, 1);

        F (CCF (i, 1)*6-(6-CCF (i, 2))) = (CCF (i, 3));
        Fnodo=F;
end
```

La iteración que se ha programado no es más que para formar un vector de n nodos con 6 grados de libertad en cada uno de ellos y aplicar en el tercer GL la magnitud deseada, en este caso todas las mas magnitudes de fuerza deben ser 0 excepto la 33 y 39, comprobémoslo

```
Fz_Nodo6= F(33),Fz_Nodo7=F(39)
```

```
Fz_Nodo6 =

  -500


Fz_Nodo7 =

   500
```

- *Apoyos*

En la gran mayoría de casos, utilizaremos un empotrado puro restringiendo los 6 GL del nodo empotrado, poniendo una *FLAG* a 0 en el GL restringido.

```
CCGL=[38 39 40 41]; %--> Imput modificado por usuario. RESTRICCIÓN TOTAL (Empotrado)
```

En la anterior línea de código, **el usuario debe introducir el número de nodos que desea empotrar**.

```
GL=ones(n,6);
GL(CCGL,:)=0:
```

En el código anterior se genera una matriz de donde cada fila representa los GL de cada nodo y las columnas representan los movimientos restringidos y seguidamente los convierte en un vector de 246x1

```
GL =

     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
     1     1     1     1     1     1
```

```
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
1     1     1     1     1     1
0     0     0     0     0     0
0     0     0     0     0     0
0     0     0     0     0     0
0     0     0     0     0     0
```

```
GL=transpose(reshape(GL',1,n*6));
```

1.26. Definición del Material & Límite elástico

En esta sección el **usuario debe seleccionar el material** deseado, eliminando/saltando las otras líneas de código, pudiendo el usuario modificarlas a su razón

ACEROS

```
E=210e9; %Pa
G =81e9; %Pa
Densidad=7800; %Kg/m3
Limite_elastico= 300e6; %Pa
```

ALUMINIO 7075

```
E=70e9; %Pa
G =26e9; %Pa
Densidad=2700; %Kg/m3
Limite_elastico= 500e6; %Pa
```

PVC

```
E=3.15e9; %Pa
G =1.1e9; %Pa
Densidad=1400; %Kg/m3
Limite_elastico= 50e6;
```

1.27. Propiedades de sección del elemento elástico (BARRAS)

Lectura del archivo xlsx para la inicialización de las variables correspondientes a las propiedades geométricas de los elementos debido a su perfil

```
I =xlsread(file,'N2:N100');
J =xlsread(file,'L2:L100');
Area=xlsread(file,'M2:M100');
```

NOTA: en versiones posteriores, estas variables se generan dentro de la función crear_barras.

Todas las variables deben ser leídas ya en sistema internacional de medidas [SI], es decir el código de Matlab no modificara las variables entrantes del archivo de datos, este debe estar correctamente introducido

1.28. Programa (Condiciones de contorno de Rigidez Torsional)

A continuación se muestra el código principal del núcleo del programa, este código es muy parecido en todos los ensayos, variando algunas funciones que son específicas para cada modelo.

Núcleo del programa

El núcleo está estructurado de manera que está constantemente accediendo a diferentes funciones, estas funciones tienen la misión de simplificar el programa evitando repetir y/o sobre escribir variables que podrían *"buggear"* el programa.

1.28.1. *Función: Crear_barras*

La función Crear_barras se encarga de inicializar una variable del tipo **estructura** que contiene toda la información necesaria para el posterior funcionamiento, así como, variables en blanco para ser rellenadas en un futuro.

Genera una sola variable de salida, Barra. Necesita de 9 variables de entrada, todas ellas leídas de la base de datos de Excel y con las líneas de código anteriores, todas estas variables solo se refieren a las propiedades geométricas y mecánicas del elemento elástico.

```
Barra = Crear_barras(m,nodos,barras,E, Densidad,G,diametros,opti_codigo);
```

Al llamar la función, Crear_barras entramos dentro de su código, veamos ese código en detalle

Función Crear_barras

A continuación vemos que al llamar esta función el código va a buscar dentro de la carpeta, esta misma. Al encontrarla entiende que tiene una variable de salida [Barra], podrían ser más. Y 9 variables de entrada (m,nodos,barras,E, Densidad,G,diametros,opti_codigo); todas ellas ya inicializadas con anterioridad. Solo así podremos entrar dentro de esta función.

```
function [Barra] = Crear_barras(m,nodos,barras,E, Densidad,G,diametros,opti_codigo);
```

```
 for i=1:m;
```

El comando **for**, es una función de programación para crear un lazo, es este caso, y en el 99% de situaciones en este trabajo, el lazo será desde 1 hasta el número (m) de barras, es decir, el código que se encuentra a continuación se repetirá m veces, que para nuestro caso son 86 veces

```
aux=nodos(barras(i,:),:);
u= [(aux(2,1) - aux(1,1));(aux(2,2) - aux(1,2));(aux(2,3) - aux(1,3))];
L=norm(u);
```

Veamos un ejemplo de la varíale auxiliar cuando *i* =9 para que el lector tenga más información

aux =

0	**-0.1500**	**0.4050**
0.7240	**-0.2075**	**0.4913**

Como podemos ver en las anteriores líneas de código, se toma una variable auxiliar para guardar en ella una variable donde estén las coordenadas de los nodos de la barra *i* . Esto se realiza para posteriormente restarlas y generar un vector que va desde el nodo *i* hacía en nodo *j*. Posteriormente con la función **norm** calculamos la norma del vector

```
Area =((pi/4*(diametros(i,1)^2-diametros(i,2)^2))*0.000001);
I=(pi*(diametros(i,1)^4-diametros(i,2)^4)/64)*0.000000000001;
J=2*(pi*(diametros(i,1)^4-diametros(i,2)^4)/64)*0.000000000001;
A=(E*Area)/L;
C=(4*E*I)/L;
D=(6*E*I)/L^2; %Dz es la negativa de Dy
B=(12*E*I)/L^3;
T=(G*J)/L;
```

Seguidamente y como se ha visto en el **capítulo 4** se inician las constantes de rigidez de una barra elástica tridimensional **apartado 4.6**

```
Barra(i).nombre = ['Barra ' int2str(i)];
```

Se crea el nombre a cada barra, siendo *i* = 1 la "Barra 1"

```
Barra(i).Kloc = [A 0 0 0 0 0 -A 0 0 0 0 0;
   0 B 0 0 0 D 0 -B 0 0 0 D;
   0 0 B 0 -D 0 0 0 -B 0 -D 0;
   0 0 0 T 0 0 0 0 0 -T 0 0;
   0 0 -D 0 C 0 0 0 D 0 C/2 0;
   0 D 0 0 0 C 0 -D 0 0 0 C/2;
   -A 0 0 0 0 0 A 0 0 0 0 0;
   0 -B 0 0 0 -D 0 B 0 0 0 -D;
   0 0 -B 0 +D 0 0 0 B 0 +D 0;
   0 0 0 -T 0 0 0 0 0 T 0 0;
   0 0 -D 0 C/2 0 0 0 +D 0 C 0;
   0 D 0 0 0 C/2 0 -D 0 0 0 C];
```

En las líneas anteriores podemos observar la creación de la **Matriz de rigidez local (4.1.1).** El código tiene en cuenta que cada barra es distinta y el lector puede cerciorarse de que así es por las líneas de código anterior, la norma, los módulos y las áreas e inercias son únicos para cada barra, y por ende cada KLOC es única, salvo elementos elásticos idénticos en todas sus propiedades

```
Barra(i).base=[];
Barra(i).Kglob=[];
Barra(i).base_vector =[];
Barra(i).vector= u;
Barra(i).acoplamientos = barras(i,:);
Barra(i).nodos = nodos(barras(i,:),:);
Barra(i).D_ext = (diametros(i,1));
Barra(i).D_int = (diametros(i,2));
Barra(i).area= Area;
Barra(i).longitud= (L);
Barra(i).inercia= (I);
Barra(i).moduloelastico= (E);
Barra(i).Faxil =[];
Barra(i).Masa =Area*L*Densidad;
Barra(i).codigo= opti_codigo(i);
```

Finalizamos inicializando variables en nuestra estructura tales como, que acoplamientos tiene la barra en cuestión, los módulos, las áreas e inercias… y también iniciamos huecos para posteriormente volcar información valuosa como podría ser la matriz de cambio base, o las fuerzas internas a las que es sometida la barra.

```
 end;
```

Comando END para finalizar el lazo y así ser repetido m veces

```
end
```

Comando END para finalizar la función, guardar el trabajo en la estructura [Barras] del “workspace” y volver al código principal

1.28.2. *Función: Plot_undisplaced*

Vuelta al programa principal para seguidamente encontrarnos una función que sirve tanto para ayudar a interpretar los datos a posteriori como para ayudar al usuario a comprender si la geometría de la estructura cumple los requisitos programados y ver si se ha producido un error al introducir las barras y nodos, así como las condiciones de contorno.

Cabe destacar que los nodos empotrados, fijos sin ningún GL libre, se muestran como cuadrados de color negro.

En la siguiente línea de código, vemos que es llamada la función, seguidamente entramos en ella.

```
Figure (1)
```

NOTA: si tenemos más de una figura activa, y queremos visualizar la estructura, debemos llamar tantas veces como figuras tengamos la función de a continuación

```
Plot_undisplaced(n,m,nodos,barras,CCGL) %Dibujando nodos NO desplazados
```

Como se puede obsevar la funcion requiere de 5 parametros o variables de entrada y ninguna variable se genera de salida, esto es debido a que unicamente se genera una figura o “Plot” en lenguage Matlab.

```
function Plot_undisplaced(n,m,nodos,barras,CCGL)
```

Una vez dentro de la función, el primer apartado de código dibuja en la figura **“Plot3”** los Cuadrados negros **“sblack”** que simbolizan los empotrados, para ello se coge la variable CCGL “condición de contorno grado de libertad” que ha sido introducida por el usuario y con su longitud de vector genera un lazo cerrado que lee que nodos tiene cada variable de entrada de CCGL y en X, Y, Z lee la variable ya guardada que la localiza en un sistema de coordenadas.

Finalizamos con un **Hold ON** que mantendrá activa la figura creada para ir incluyendo otras figuras.

```
for i=CCGL;
    x=nodos(i,1);
    y=nodos(i,2);
    z=nodos(i,3);
    plot3(x,y,z,'sblack','MarkerSize',15,'MarkerFaceColor','black')
    hold on
end
```

Seguidamente el código dibuja nodo a nodo como una esfera completamente negra, además el código mostrado a continuación pone el nombre a los ejes y ala figura en sí.

```
for i=1:n;
    x(i)=nodos(i,1);
    y(i)=nodos(i,2);
    z(i)=nodos(i,3);
    plot3(x,y,z,'*black','MarkerSize',1)
    title('Método matricial de la rigidez')
    xlabel('Eje x [m]')
    ylabel('Eje y [m]')
    zlabel('Eje z [m]')
end
```

En este apartado que mostramos a continuación se dibujan las barras de nodo *i* a nodo *j*. En el código se utiliza la variable *aux* que guardara en memoria un vector doble que solamente tiene en cuenta que la barra *i* tiene nodo *a* y *b* guardando las coordenadas únicamente. Veamos un ejemplo

aux =

0	**-0.1500**	**0.4050**
0	**0.1500**	**0.4050**

Entonces se guarda en otra variable como *h, v, t*, para que esta tenga en cuenta donde dibujar una línea de barra a barra

v =

-0.1500 0.1500

Notase que *h* se equipara a X, *v* a Y e t se entiende como a *z*

```
hold on
for i=1:m;
      aux=nodos(barras(i,:),:);
   h(1)=aux(1,1);
   v(1)=aux(1,2);
   t(1)=aux(1,3);
   h(2)=aux(2,1);
   v(2)=aux(2,2);
   t(2)=aux(2,3);
plot3(h,v,t,'black','linewidth',2)
end

hold on
```

Como siempre se mantiene la figura active y se cierran todos los lazos

```
end
```

En este punto, el programa vuelve al código principal, ploteando la figura que deseamos. Es aquí cuando el usuario puede comprobar visualmente si la estructura está bien compuesta.

En la figura 6.2 podemos observar lo explicado en el párrafo anterior.

En la herramienta que Matlab proporciona, se puede mover, rotar la vista a razón del usuario, y esto facilita enormemente la interpretación de la geometría, además de poder verla también en distintos planos de trabajo como se muestran en las figura 6.3

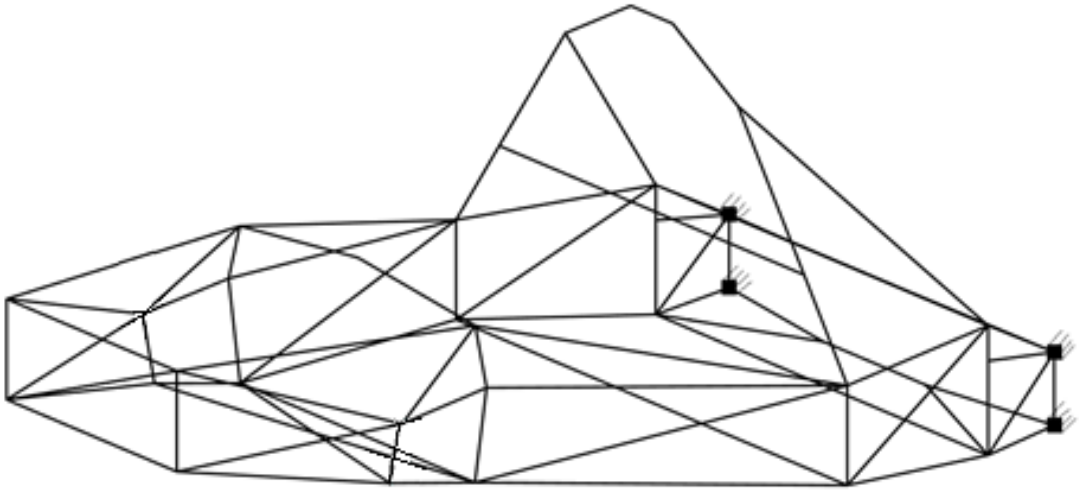


Imagen 6.1 *Estructura multitubular.*
Procesada por función Plot_undesplaced

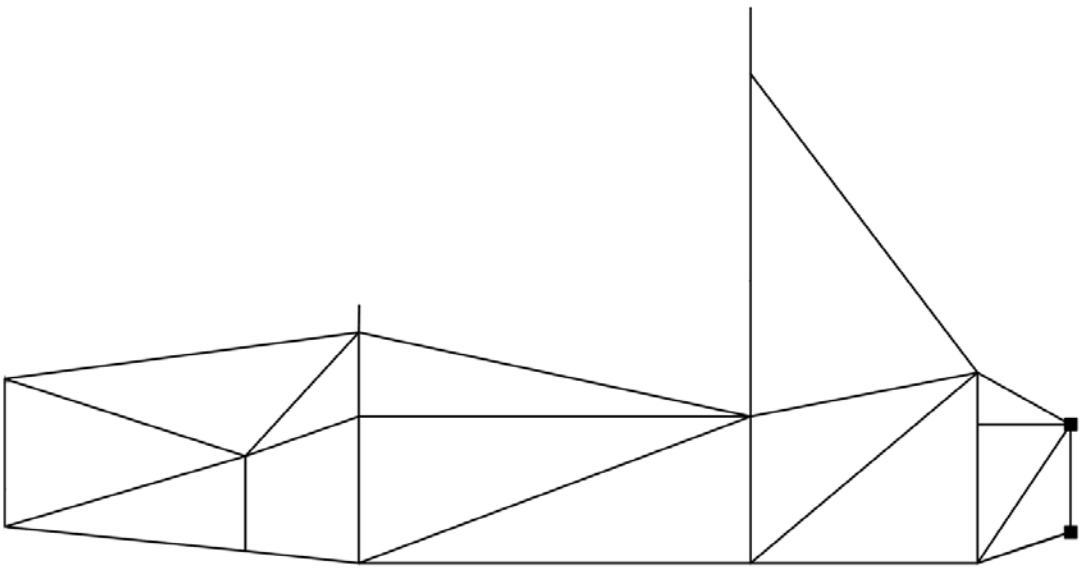


Imagen 6.1 *Estructura multitubular. Vista XZ*

1.28.3. *Función: VectorToBase*

Esta función tiene un desempeño clave a la hora de simplificar el trabajo del usuario. Se trata de una función que convierte un vector, el vector que se genera de nodo *i* a nodo *j*, en una base tridimensional. Esto se debe realizar porque el núcleo del programa cambia de sistema de referencia a través de la teoría de cambio de base de otra base. Esto se entenderá mejor si profundizamos en el código

VectorToBase necesita de 2 variables de entrada, la estructura Barra y la variable de cantidad de barras m, resolviendo/modificando la estructura Barra.

Como resumen podemos decir que si el vector (i) está en un plano XY, XZ, YZ entonces genera una base donde la componente <i> tiene la misma dirección que (i).

Para futuras modificaciones del código, cabe decir que la estructura Barra internamente ya tiene la variable m, pero no afecta a la velocidad del programa, solo a su organización.

```
function [Barra] = VectorToBase(Barra,m)
```

Se empieza en lazo desde 1 hasta la longitud total del vector barras, que en este caso es m con un total de 86.

```
for l=1:m;
```

Para poder entrar en el siguiente lazo, un lazo condicionado, se necesitan cumplir una serie de parámetros. En el siguiente caso estas condiciones no son más que dos.

El vector (i) tiene componentes X, es decir, su valor no es nulo, puede ser un real positivo o negativo.

&& significa **AND,** que significa que **además** NO debe tiene componentes en Z, es decir está en el PLANO XY

```
if Barra(l).vector(1)~=0 && Barra(l).vector(3)==0
```

Si el condicional es cumplido, el código entrara en la siguiente sección, si no es así saltara hacia ale próximo condicionante. Así sucesivamente hasta encontrar un final de código.

En la siguiente sección de código se crea la base ortogonal de la siguiente manera

Se crea la componente <i> de la base del vector (i), se le da los mismos valores que el vector (i) y normalizados, es decir, se obliga a tener en la componente de la base ortogonal una componente unitaria

Veamos cómo se comportan las variables guardadas en el *"Workspace"*

Barra (58).vector =

0.4637
-0.0520
0

Norm (Barra (58).vector) =

0.4667

Esto da como resultado un vector <i> de la base ortogonal con las siguientes componentes globales

<i> = Barra(58).vector/norm(Barra(6).vector)

0.9938
-0.1114
0

Finalizada la componente <i> y sabiendo que estamos en un plano XY se decide crear la componente <k> de la base ortogonal del vector (i) para COINCIDIR CON e3 (0, 0, 1)

Por último y para acabar de crear la base ortogonal a partir de nuestro vector *i* se trata de un simple cálculo de la base ortogonal que queda por definir <j> del vector (i) a través de un producto escalar de 90º para cumplir con la ortogonalidad de nuestra base, característica indispensable.

<j>=cross(k,i) =

0.1114
0.9938
0

En este punto ya hemos creado nuestra base ortogonal y es guardada en la estructura Barras.base_vector de la siguiente manera:

Barra (58).base_vector =

<i>	<j>	<k>
0.9938	0.1114	0
-0.1114	0.9938	0
0	0	1.0000

Veamos el código interpretable por Matlab

```
i=Barra(l).vector/norm(Barra(l).vector);
k=[0;0;1];
j=cross(k,i);
   Barra(l).base_vector =[i,j,k];
```

Una vez finaliza esta línea, la siguiente línea es un **ELSEIF,** esto el ordenador lo interpreta como un END, es decir, saldrá del lazo IF cambiara la l y sera l=l+1 dando lugar a otra iteración.

El siguiente apartado del lazo IF es cuando la primera condición no se cumple, entonces pasa a evaluar la siguiente condición que en este caso es cuando el vector (i) tiene componentes X y además NO tiene componentes en Y , lo cual significa que es un vector integrado en el PLANO XZ

```
elseif Barra(l).vector(1)~=0 && Barra(l).vector(2)==0
```

En las líneas de código que restan a continuación, son de la misma razón que las explicadas anteriormente, es fácil deducir el resultado de las mismas y el concepto geométrico que deriva del código.

El resultado es el mismo que el anteriormente comentado, lo único que cambia es que aquí el vector de la base ortogonal <j> es programado para que este alineado con el vector *j* de la base canónica del sistema de referencias global.

```
i=Barra(l).vector/norm(Barra(l).vector);
j=[0;1;0];
k=cross(i,j);
 Barra(l).base_vector =[i,j,k];
```

Una vez finalizado este apartado del código, el programa realiza una siguiente iteración.

Si el vector aún no se encuentra en ninguna de estas 2 condiciones, entrara a la tercera condición, y esta es que el vector (i) tiene componentes Z ó Y y además NO tiene componentes en X, lo cual significa que está integrado en el plano PLANO YZ

El Comando **||** significa **OR**, que puede aceptar una u otra variable y que entrara dentro de la condición.

```
elseif (Barra(l).vector(3)~=0 || Barra(l).vector(2)~=0 ) &&
Barra(l).vector(1)==0
```

Las siguientes líneas de código son de la misma intención que las anteriores, fácilmente interpretadas.

```
i=Barra(l).vector/norm(Barra(l).vector);
j=[-1;0;0];
k=cross(i,j);
   Barra(l).base_vector =[i,j,k];
```

El código, finalizadas estas líneas vuelve a iterar el código con la siguiente barra *m*.

La última condición se da cuando vector (i) tiene componentes X Y Z, es decir que no está en ningún plano. Esto sucede en el 90% de los casos

El código de a continuación muestra la condición para que entre en este lazo, aun con esto se podría programar esta cuarta condición como una condición ultima, es decir, que si no se cumplen las 3 anteriores, entre directamente en un lazo aparte.

```
   elseif Barra(l).vector(3)~=0 && Barra(l).vector(2)~=0 &&
Barra(l).vector(1)~=0
```

En este código se selecciona de nuevo la componente <i> de la base del vector (i), se le da los mismos valores que el vector (i) y los normaliza convirtiéndolos en vectores unitarios

Seguidamente se genera una variable U que es un vector igual al vector original pero anulando la componente *k*.

Barra(6).vector =

0.4922

-0.0580

-0.0450

```
<i> =

   0.9891
  -0.1165
  -0.0904
```

Veamos en detalle como la variable U modifica el vector de la barra 6, esta variable no es más que para guardar un nuevo vector, que tiene la misma dirección que el vector <i>.

Se respeta que el la componente unitaria de la base <j> se encuentre en el plano XY por ello las líneas de código que se ven, van referidas a ello.

Finalmente se genera un vector perpendicular y con este el ortogonal, finalizando así con la base que deseamos.

```
Barra (6).base_vector =

     <i>      <j>      <k>
   0.9891  -0.1170   0.0898
  -0.1165  -0.9931  -0.0106
  -0.0904        0   0.9959
```

A continuación se muestra el código explicado anteriormente

```
    i=Barra(l).vector/norm(Barra(l).vector);
        udir=[i(1) i(2)];
        inorm=norm(udir) ;
        u=udir/inorm;
    j=u*[0 -1;1 0];
    j(3)=0;
    j=transpose(j);
    k=cross(j,i);
        Barra(l).base_vector =[i,j,k];
```

```
end
```

Fin de la cuarta condición

```
end
```

Fin de la iteracion para cada barra

```
end
```

Fin de la función

1.28.4. *Función: cambio de base*

Una vez tenemos programadas para cada barra las bases correspondientes debemos cambiarlas o transfórmalas del sistema de referencia local a un sistema de referencias globales. Para ello se ha creado la función de “cambio_base”

```
function [Barra] = cambio_base(Barra,m)
```

Esta función únicamente necesita de la variable m para poder iterar a cada barra y poder leer y escribir la estructura Barra.

Como siempre se empieza iterando para cada barra

```
for l=1:m;
```

A continuación se genera una base canónica para tenerla como referencia, en realidad esta base canónica es la misma que el sistema de referencia global.

```
e1=[1;0;0];
e2=[0;1;0];
e3=[0;0;1];
```

Seguidamente y para cada barra, se leen las componentes de la base local, con esto se tiene guardada en el “workspace” ambas variables para poder tratarlas.

```
i=Barra(l).base_vector(:,1);
j=Barra(l).base_vector(:,2);
k=Barra(l).base_vector(:,3);
```

Como se ha visto anteriormente en el capítulo 4, y más específicamente en el apartado 4.8, para poder transformar o cambiar una base en referencia local a una referencia global, nos basaremos en los cosenos directores entre el vector unitario <i> de la base local y e1, e2 y e, y así sucesivamente para crear la matriz R

```
ie1=acosd(dot(i,e1));
ie2=acosd(dot(i,e2));
ie3=acosd(dot(i,e3));
 je1=acosd(dot(j,e1));
 je2=acosd(dot(j,e2));
 je3=acosd(dot(j,e3));
ke1=acosd(dot(k,e1));
ke2=acosd(dot(k,e2));
ke3=acosd(dot(k,e3));
```

Matriz de cosenos directores:

```
R= [ie1,je1,ke1;ie2,je2,ke2;ie3,je3,ke3];
```

Seguidamente se crea la Matriz transformación, que no es más que la acomodación de la matriz de cosenos directores para la modelización de 12gdl del elemento elástico.

```
T=[cosd(R),zeros(3),zeros(3),zeros(3);
    zeros(3),cosd(R),zeros(3),zeros(3);
    zeros(3),zeros(3),cosd(R),zeros(3);
    zeros(3),zeros(3),zeros(3),cosd(R)];

   Barra(l).base= T;
 Barra(l).Kglob= T*Barra(l).Kloc*transpose(T);
```

En la penúltima línea de código, donde se ve la ecuación algebraica descrita en el capítulo 4, es donde se pasa de tener una matriz expresada en su base local a expresarla en una base común para todas ellas, esta base de referencia no es más que la base global.

Por ello el autor como ya ha explicado trata sus variables con los nombres de “Kloc” refiriéndose a una matriz de rigidez de un elemento elástico lineal con sus bases referidas a un sistema local, en cambio “”Kglob” se refiere a esa misma matriz de rigidez pero ya expresando los valores a un sistema global y común de referencia.

```
end
```

Final de la función

1.28.5. *Función: Matriz Global*

Esta función tiene el principal propósito de crear la matriz que gobierna el problema, está dividida en dos secciones.

La primera de estas crea una matriz de tantas dimensiones como nodos tenga el problema.

La segunda, es el ensamblaje de todos los elementos elásticos, en el capítulo 5.10 se encontrara toda la explicación teórica del método.

Se empieza con la llamada de la función desde el programa principal y seguidamente se entra en ella

```
Matriz_ global(n,m,barras,Barra)
```

Se puede comprobar que esta función solo genera una variable de salida “Kglob”. Cabe esclarecer que aun teniendo el mismo nombre que la variable que define la base en la que es referida la matriz global de un elemento elástico, existe una diferenciación clave. “Kglob” es indiscutiblemente la matriz rigideces de todo el sistema de ecuaciones, pero “Barra (i).Kglob” es la matriz rigidez del elemento “i”.

Para poder acceder a esta función debemos tener en el “WorkSpace” variables como, numero de nodos, numero de barras, vector barras y la estructura barra.

```
function [Kglob] = Matriz_global(n,m,barras,Barra)
```

A continuación se inicializa la matriz principal del sistema de ecuaciones, esta matriz es para un sistema de 6 gdl por nodo, lo que significa que está en un dominio tridimensional, si por ejemplo el problema fuera de dos dimensiones se pondría 3 gdl.

NOTA: si el lector es ágil, después de unos instantes podrá percatarse de que substituyendo el valor numérico por: **length (Barra (1).Kglob) /2** el código estará parametrizado para cualquier dimensión.

Esta matriz estará en blanco, que informáticamente hablando contara con un cero o valor nulo en todas sus componentes

```
ancho=(n*6); %3D
Kglob = zeros(ancho);
```

Una vez inicializada la Matriz global se pasa al código que ensambla todas las matrices de cada elemento elástico a la principal

```
for i=1:m;
```

Es ya conocido el lazo para poder intervenir todas las barras de nuestro Sistema. Una vez realizado esto crearemos una variable de apoyo “aux2” esta no es más que la simplificación visual del código.

```
    aux2=barras(i,:);
```

Veamos el ejemplo más sencillo, o mayor dicho el primero que nos encontraremos, es decir para la barra 1.

i=1

aux2 = barras (i, :);

1.0000e+000 2.0000e+000

Los códigos de a continuación están separados para cada una de las partes de la matriz rigidez, como ya se ha explicado anteriormente.

Por ello el código sigue este mismo metro, veamos la primera sección en profundidad, las demás son todas clones de la primera, por supuesto respetando el orden.

% % % Kaa % % %

```
u=1;k=1;
for c = (((aux2(1)-1)*6)+1):(((((aux2(1)-1)*6)+1))+5);
 for r = (((aux2(1)-1)*6)+1):(((((aux2(1)-1)*6)+1))+5);
  Kglob(r,c) =Kglob(r,c) + Barra(i).Kglob(u,k);
  u=u+1;
 end;
 u=1;
 k=k+1;
end;
```

Se inicializan unas variables de control que sirven para situar al código en que porción de la matriz global del elemento elástico deben leer.

Vemos que existen dos lazos, uno interno a otro, esto no es más que el barrido de todas las filas y columnas de una matriz, en nuestro caso de la matriz global

“r” se refiere a filas

“c” se refiere a columnas

Se puede observar como el lazo se recorre a través de un algoritmo parametrizado, es imperativo que se entienda.

Este algoritmo tiene el propósito de entender dónde empieza la numeración de nodos de cada barra en la matriz global, es decir la barra 1, su primer nodo global es el 1 como caso trivial (al tratarse de una barra que su primer nodo es el 1). Pero que sucede para la barra 2 y 7 por ejemplo.

i = 2 **i = 7**

((i - 1) * 6) + 1 = 7 **((i - 1) * 6) + 1 = 37**

Vemos que es correcto para todo “i” porque como máximo cada nodo tiene 6gdl

Si sabemos que cada secuencia tiene que ir de 1 hacia 6 son lazos de 5 iteraciones cada una.

(((aux2 (1)-1)*6) +**1**) :(((((aux2 (1)-1)*6) +1)) +**5**);

Una vez dentro del lazo nos encontramos con una línea de código que escribe en la matriz global principal el valor que ya tenía, podría ser nulo o no nulo, y le suma el valor que tiene el elemento elástico "i" de las variables inicializadas al principio "u, k". Recordamos que estas variables tienen en cuenta el nodo.

Kglob(r, c) =Kglob(r, c) + Barra (i).Kglob (u, k);

Las líneas de código que encontramos a continuación no son más que el cambio a la siguiente línea o matriz, si cambia en el lazo de la matriz global principal, también lo debe hacer en la de la barra. Entendiendo también que cuando una fila se ha acabado, debe volver a inicializarse.

% % % Kab % % %

```
u=7;k=1;
for c = (((aux2(2)-1)*6)+1):(((((aux2(2)-1)*6)+1))+5);
 for r = (((aux2(1)-1)*6)+1):(((((aux2(1)-1)*6)+1))+5);
  Kglob(r,c) =Kglob(r,c) + Barra(i).Kglob(k,u);
  k=k+1;
 end;
 k=1;
 u=u+1;
end;
```

% % % Kba % % %

```
u=7;k=1;
for c = (((aux2(1)-1)*6)+1):(((((aux2(1)-1)*6)+1))+5);
 for r = (((aux2(2)-1)*6)+1):(((((aux2(2)-1)*6)+1))+5);
  Kglob(r,c) =Kglob(r,c) + Barra(i).Kglob(u,k);
  u=u+1;
 end;
 u=7;
 k=k+1;
end;

aux2=barras(i,:);
```

% % % Kbb % % %

```
u=7;k=7;
for c = (((aux2(2)-1)*6)+1):(((((aux2(2)-1)*6)+1))+5);
 for r = (((aux2(2)-1)*6)+1):(((((aux2(2)-1)*6)+1))+5);
  Kglob(r,c) =Kglob(r,c) + Barra(i).Kglob(u,k);
  u=u+1;
 end;
 u=7;
 k=k+1;
end;
```

Fin de lazo de barra

```
end
```

Fin de la función

```
end
```

1.28.6. *Función: Solver*

La siguiente función que nos encontramos es la que trata de resolver el sistema de ecuaciones que se plantea en el problema. Como ya hemos visto en temas anteriores es de vital importancia tener un sistema compatible determinado, es decir, que solo tenga una única solución.

Sabemos que la matriz global principal, la que juega un papel indispensable en nuestro sistema de ecuaciones, se trata de una matriz con determinante nulo, es decir, que no tiene inversa. Esto significa que la matriz es singular y que no tiene solución. Si calculáramos los valores propios de la matriz veríamos cuantas ecuaciones deberíamos eliminar para que esta sea invertible y tenga solución.

Veamos como la función trata este problema.

```
function [soldesp,desp] = solver(n,Kglob,GL,F)
```

Se trata de una función que reporta dos variables “soldesp” y “desp” donde la primera de ellas es la simplificación del vector de desplazamientos que resulta de la solución del sistema de ecuaciones de la matriz reducida. La segunda de ellas se trata de la reconstrucción del vector con todas las posiciones representadas. Esto lo veremos con detalle a continuación.

Como variables de entrada están, la cantidad de nodos, la matriz global principal, y vectores de fuerza y de grados de libertad.

Esta primera parte del código se inicializa las variables para poder eliminar la singularidad de “Kglob” a través de los Grados de libertad “GL”

```
ancho=(n*6); %3D
j=ancho; %inicialización
```

Inicializamos las variables para saber la longitud de los vectores, dicho de otra manera, la cantidad de ecuaciones que tiene el sistema.

```
  for gl= ancho:-1: 1
```

Vemos que el siguiente lazo trata la sucesión de manera inversa, esto es así porque como ya se verá, se eliminan filas y columnas, y con este método no se pisa el lazo con una fila ya eliminada, evitando así un fallo.

Veamos con más detalle que se realiza dentro de este lazo. El comando **IF** es un condicionante y su entrada está vetada si no se cumple no entra dentro de las siguientes líneas de código.

Se entrara cuando encuentre un valor nulo en el vector “GL”, es decir, si el grado de libertad es 0, eliminando la fila y la columna de la matriz rigidez principal

Si el grado de libertad no tiene un valor nulo, se ignora y se pasa al siguiente bajando la numeración de “j”

```
    if GL(gl)==0
        Kglob(j,:)=[];% fila fuera
        Kglob(:,j)=[];% columna fuera
        j=j-1;
    else
        j=j-1;
    end;
```

Fin de la condición

```
  end;
```

Fin del lazo

A su vez para que el sistema tenga las mismas ecuaciones, debemos simplificar el vector fuerzas del mismo modo que se ha descrito anteriormente

```
k=ancho; %inicialización
  for gl= ancho:-1:1
    if GL(gl)==0
        F(k)=[];% componente k fuera
        k=k-1;
    else
        k=k-1;
    end;
  end;
```

Una vez llegado a este paso, el sistema de ecuaciones esta formulado correctamente, la matriz reducida con los vectores de fuerza reducidos pueden ser ya solucionados a través de la siguiente línea de código.

```
soldesp=Kglob\F;
```

NOTA:

Matlab es una excelente herramienta de cálculo por situaciones como la actual.

Podemos ver que el sistema de ecuaciones empieza a tener una magnitud considerable, matrices del orden de 300 x 300, es decir, que si tratamos de encontrar la inversa de esta matriz, muy posiblemente tardemos más de lo esperado.

Por ello Matlab propone que se trate as sistema de ecuaciones de la manera que expresa el código **Kglob\F** ya que internamente el propio programa escogerá la manera más óptima de tratar este sistema de ecuaciones.

Una vez solucionado el sistema de ecuaciones, la variable que guarda la solución es la “soldesp” esta variable es un vector de desplazamientos reducido, para que sea más sencilla su interpretación y manipulación debemos reconstruir este vector con el siguiente código.

Se inicializan las variables pertinentes

```
h=ancho;
z=length(soldesp);
```

El código es muy parecido al anterior, salvo que se basa en rellenar los huecos vacíos.

La manera de realizar esto vuelve a ser con un lazo inverso, que interpreta cuando en el vector “GL” existe una variable no nula, le introduce la variable del vector “soldesp”, seguidamente cambia sus variables de inicialización para continuar el procedimiento. Si por el contrario en el vector “GL” aparece una variable nula, significa que ese grado de libertad está restringido y por ende el desplazamiento y rotación en todas las dimensiones se encuentran inmóviles es decir con un valor nulo de desplazamiento.

```
for gl= ancho:-1:1
    if GL(gl)==1
        desp(h)=soldesp(z);
        h=h-1;
        z=z-1;
    else
        desp(h)=0;
        h=h-1;
    end;

end;
```

Fin del lazo y de la condición

Transpuesta del vector fila por un vector columna en la siguiente línea de código

```
desp=transpose(desp);
```

Fin de la función

```
end
```

1.28.7. *Función: Tensión*

En esta función se tratan de evaluar las tensiones que soportan los elementos elásticos lineales, en tres pastes del mismo, en los nodos que lo definen y en la porción que los une.

Como se ha explicado anteriormente esta parte no es del propio método de cálculo matricial, es más la extensión lógica que deriva del método, y sirve para un sinfín de posibilidades. En este caso, lo utilizaremos para calcular donde se encuentran los puntos más débiles del sistema y así poder estudiar una solución con más acierto.

La base teórica de este apartado se puede encontrar en el capítulo 4.13 donde se detalla el cómo y porqué del cálculo de reacciones y tensiones.

Como siempre, al llamar la variable esta necesitara de unas entradas y reportara unas salidas, la línea que se ve a continuación muestra que variables de entrada y salida gobiernan la función.

```
function [Nodo,Barra] = tension(m,Barra,desp,n)
```

Es ya típico y conocido el lazo que cubre todos los elementos elásticos

```
for i=1:m;
```

Empezamos con los cálculos de reacciones de cada nodo de cada barra, como se puede observar en las líneas de código siguientes, en la Barra(i) se escoge aquellas rigideces que tienen que ver con el nodo A, por ello se ve como se cogen los valores de los 6 primeros grados de libertad. Al mismo tiempo se coge los mismos valores para el cambio de base, y juntando estas dos nuevas variables “Kaa” y “Q” más el vector de desplazamiento del nodo A, se puede saber qué esfuerzo ha realizado el nodo A al nodo A

```
  kaa=Barra(i).Kloc(1:6,1:6);
  Q=Barra(i).base(1:6,1:6);
  esf_int_aa=kaa*transpose(Q)*desp( (((Barra(i).acoplamientos(1))-
1)*6)+1:(((Barra(i).acoplamientos(1))-1)*6)+6);
```

Lo mismo sucede con el nodo B de la barra (i), vemos que la variable “Kab” es escogida en función a que nodo nos queremos referir, en este caso es el nodo B de la barra (i) pero que afecta al nodo A.

```
kab=Barra(i).Kloc(1:6,7:12);
Q=Barra(i).base(1:6,1:6);
esf_int_ab=kab*transpose(Q)*desp( (((Barra(i).acoplamientos(2))-
1)*6)+1:(((Barra(i).acoplamientos(2))-1)*6)+6);
```

Una vez guardados en variables dentro de la estructura Barras se suman los esfuerzos internos que el nodo A de la Barra (i) y los esfuerzos internos que el nodo B de la Barra (i) que afectan al NODO A.

Se ve claramente que los esfuerzos totales están en formato local, esto es una ventaja porque se podrá ver fácilmente que esfuerzos son axiles, cortantes y flexiónales.

Además se ha decidido guardar la variable en formato global, que no es más que el vector de fuerza/reacción que ve el nodo (i) en el sistema de referencia global. Esto es útil por si queremos sacar las reacciones con una mejor interpretación y así poderlas pasar fácilmente a otro programa.

```
Barra(i).esf_loc_total_nodo1=esf_int_aa+esf_int_ab;
Barra(i).esf_glob_nodo1=Barra(i).base(1:6,1:6)*Barra(i).esf_loc_total_nodo1;
```

En los bloques que se ven a continuación, el procedimiento es el mismo salvo que se refieren al nodo B, reacciones del nodo A y B que repercuten al nodo B.

```
kaa=Barra(i).Kloc(7:12,1:6);
Q=Barra(i).base(1:6,1:6);
esf_int_aa=kaa*transpose(Q)*desp( (((Barra(i).acoplamientos(1))-
1)*6)+1:(((Barra(i).acoplamientos(1))-1)*6)+6);
```

```
kab=Barra(i).Kloc(7:12,7:12);
Q=Barra(i).base(1:6,1:6);
esf_int_ab=kab*transpose(Q)*desp( (((Barra(i).acoplamientos(2))-
1)*6)+1:(((Barra(i).acoplamientos(2))-1)*6)+6);
```

```
Barra(i).esf_loc_total_nodo2=esf_int_aa+esf_int_ab;
Barra(i).esf_glob_nodo2=Barra(i).base(1:6,1:6)*Barra(i).esf_loc_total_nodo2;
```

Con esto finalizamos el lazo para cada barra y completemos así los esfuerzos locales y globales que sufren los nodos A y B de cada uno de los elementos elástico-lineales del problema.

```
end
```

Una vez finalizado el lazo anterior, se decide crear un lazo adicional pero esta vez para recorrer toda la cantidad de Nodos que existen en nuestro problema

Esta nueva estructura llamada Nodos (i) se trata de una variable auxiliar para ayudar a calcular las tensiones que se producen en los nodos, así como posteriormente poder plotearlas en la función que dibuja los nodos y barras desplazadas

```
for i=1:n;
```

Dentro de este lazo existe otro lazo más para poder iterar a cada barra, además de un contador “K” que sirve para poder poner más de una variable dentro de la estructura Nodos sin que se pisen/eliminen unas a otras.

```
    k=1;
    for j=1:m;
```

En esta sección de código nos encontramos con una condición, esta condición no es más que la de verificar si el nodo que se está evaluando <i> lo pose la barra <j>, además especifica que este nodo debe estar en la posición A, es decir, que sea el primer nodo de los dos.

```
if Barra(j).acoplamientos(1)==i
```

Si es así, entraremos en esta sección y creara dentro de la estructura NODO (i) distintas subvariables que reflejaran los esfuerzos globales que sufre cada nodo.

NOTA: esto se realiza así porque cada nodo tiene distintos esfuerzos debido a las distintas barras que son acopladas a estos, por ello este código guarda cada uno de estos esfuerzos, para posteriormente escoger el máximo de ellos.

```
Nodo(i).Fx(k)=Barra(j).esf_glob_nodo1(1);
Nodo(i).Fy(k)=Barra(j).esf_glob_nodo1(2);
Nodo(i).Fz(k)=Barra(j).esf_glob_nodo1(3);
Nodo(i).Mx(k)=Barra(j).esf_glob_nodo1(4);
Nodo(i).My(k)=Barra(j).esf_glob_nodo1(5);
Nodo(i).Mz(k)=Barra(j).esf_glob_nodo1(6);
k=k+1;
```

Se puede ver como el contador obliga a escribir consecutivamente dentro de una variable dentro de la estructura.

Podemos ver qué, sino se cumple la condición anterior se comprobara esta segunda, si ambas condiciones no se cumplen saldrá del bucle e iterara de nuevo.

En esta condición evalúa en nodo y barra para que sean iguales que en Nodo B de la barra (i).

```
    elseif Barra(j).acoplamientos(2)==i
        Nodo(i).Fx(k)=Barra(j).esf_glob_nodo2(1);
        Nodo(i).Fy(k)=Barra(j).esf_glob_nodo2(2);
        Nodo(i).Fz(k)=Barra(j).esf_glob_nodo2(3);
        Nodo(i).Mx(k)=Barra(j).esf_glob_nodo2(4);
        Nodo(i).My(k)=Barra(j).esf_glob_nodo2(5);
        Nodo(i).Mz(k)=Barra(j).esf_glob_nodo2(6);
        k=k+1;
    end
```

Finalizamos las condiciones

```
  end
```

Finalizamos los lazos para cada barra y cada nodo.

```
end
```

En esta sección, de la misma función, es donde realmente se calculan las tensiones que sufren tanto barras como nodos, y se realiza del siguiente modo.

Como siempre, empezamos con un lazo que recorre cada barra.

```
for i=1:m;
```

Ya vimos anteriormente la función de esta auxiliar “Daux”

```
Daux=diametros(i,:);
```

Empezaremos con el cálculo de las tensiones axiles de la barra (i), copiando el valor absoluto para evitar restas en posteriores cálculos. Cogiendo la ecuación 4.13 para poder evaluar la tensión axil. Y la guardamos en la estructura Barra

```
Faxil=abs(Barra(i).esf_loc_total_nodo1(1));
Barra(i).Saxil=Faxil/Barra(i).area;
```

Continuamos con las tensiones cortantes, en este caso al ser un problema de 3Dimensiones se debe tener en cuenta las tensiones esviadas, por ello se calcula la resultante de las fuerzas cortantes para tener el total, y con la ecuación 4.17 obtenemos la tensión cortante. Y la guardamos en la estructura Barra

```
Fcort=sqrt(abs(Barra(i).esf_loc_total_nodo1(2))^2+
abs(Barra(i).esf_loc_total_nodo1(3))^2) ;
Barra(i).Scort=(4/3)*(Fcort/Barra(i).area);
```

Pasamos a las tensiones de flexión, el procedimiento a seguir es el mismo, pero al tratarse de una barra que posee dos extremos con tensiones muy posiblemente distintas, se decide calcular la media aritmética de los nodos A y B de la barra (i) para poder tener un mínimo de referencia.

NOTA: esto es poco ortodoxo, ya que la barra en si sufre una tensión de flexión, lineal, cuadrática, cubica… dependiendo de la naturaleza de la carga aplicada, el autor la está tratando como constante, cosa matemáticamente imposible, ya que no existe un grado menor a una carga puntual.

Se utiliza la ecuación 4.15 para resolver la tensión.

```
 Mresnodo1=sqrt( ( Barra(i).esf_loc_total_nodo1(5) )^2 +
( Barra(i).esf_loc_total_nodo1(6) )^2 );
   Mresnodo2=sqrt( ( Barra(i).esf_loc_total_nodo2(5) )^2 +
( Barra(i).esf_loc_total_nodo2(6) )^2 );
   Mresmed=(Mresnodo1+Mresnodo2)/2;
   Barra(i).Sflex=(Mresmed*64*((Daux(1)*1e-3)/2))/( (Daux(1)*1e-3)^4-
(Daux(2)*1e-3)^4 *pi );
```

Lo mismo ocurre para las tensiones torsionales, en este caso, el autor comete un error a propósito para evitar posibles fallas del programa posteriores, esto se ve claramente porque se escoge el máximo de las torsiones de cada nodo, pero el físicamente, el método ha tenido en cuenta que si se aplica un esfuerzo torsional a un elemento elástico, este es el mismo para todos los nodos que contenga.

Esto no significa que el código este erróneo, sino más bien, que realiza un cálculo trivial, que no debería realizarse.

```
   Mtresnodo1=Barra(i).esf_loc_total_nodo1(4);
   Mtresnodo2=Barra(i).esf_loc_total_nodo2(4);
   Mtres=max(abs(Mtresnodo1),abs(Mtresnodo2));
   Barra(i).Stor=(Mtres*32*((Daux(1)*1e-3)/2))/( (Daux(1)*1e-3)^4-
(Daux(2)*1e-3)^4 *pi );
```

Una vez llegado a este paso, teniendo en cuenta todas las sigmas que sufre el elemento elástico, y como se ha prescrito en el capítulo 4.13 en el apartado de cálculo de tensiones. Se evalúa la sigma de tensión equivalente, una herramienta muy potente para unificar todas estas tensiones en una sola y compararla con la tensión admisible que tiene el material.

```
Sx=Barra(i).Saxil+Barra(i).Sflex;
Txy=Barra(i).Scort+Barra(i).Stor;
S1=(Sx/2)+sqrt( (Sx/2)^2 + Txy^2 );
S2=(Sx/2)-sqrt( (Sx/2)^2 + Txy^2 );
Barra(i).Seq=sqrt(S1^2+S2^2-(S1*S2));
```

Con esto finalizamos el cálculo para cada barra

```
end
```

Con todo lo anteriormente programado, de nuevo se utiliza la variable Estructural Nodo (i) para poder escribir sobre esta misma la tensión que soporta cada nodo.

El procedimiento es el mismo que el anteriormente explicado, con las mismas ecuaciones y condiciones tenidas en cuenta, todas salvo una. La condición de la tensión de flexión, aquí si es guardada la tensión que tiene el nodo, ya que es un lugar puntual de la barra y se trata del grado de libertad estudiado.

```
for i=1:n;
k=1;
for j=1:m;
```

```
  if Barra(j).acoplamientos(1)==i
```

```
        Daux=diametros(j,:);
        Mresnodo1=sqrt( ( Barra(j).esf_loc_total_nodo1(5) )^2 +
( Barra(j).esf_loc_total_nodo1(6) )^2 );
        Mtresnodo1=Barra(j).esf_loc_total_nodo1(4);
        Mtresnodo2=Barra(j).esf_loc_total_nodo2(4);
        Mtres=max(abs(Mtresnodo1),abs(Mtresnodo2));
```

```
        Nodo(i).Saxil(k)=Barra(j).Saxil;
        Nodo(i).Scort(k)=Barra(j).Scort;
        Nodo(i).Sflex(k)=(Mresnodo1*64*((Daux(1)*1e-3)/2))/( (Daux(1)*1e-
3)^4-(Daux(2)*1e-3)^4 *pi );
        Nodo(i).Stor(k)=(Mtres*32*((Daux(1)*1e-3)/2))/( (Daux(1)*1e-3)^4-
(Daux(2)*1e-3)^4 *pi );

        k=k+1;
```

```
    elseif Barra(j).acoplamientos(2)==i
```

```
        Daux=diametros(j,:);
        Mresnodo1=sqrt( ( Barra(j).esf_loc_total_nodo2(5) )^2 + (
Barra(j).esf_loc_total_nodo2(6) )^2 );
        Mtresnodo1=Barra(j).esf_loc_total_nodo1(4);
        Mtresnodo2=Barra(j).esf_loc_total_nodo2(4);
        Mtres=max(abs(Mtresnodo1),abs(Mtresnodo2));
```

```
        Nodo(i).Saxil(k)=Barra(j).Saxil;
        Nodo(i).Scort(k)=Barra(j).Scort;
        Nodo(i).Sflex(k)=(Mresnodo1*16)/(  (Daux(1)*1e-3)^3-(Daux(2)*1e-
3)^3  *pi  );
        Nodo(i).Stor(k)=(Mtres*32*((Daux(1)*1e-3)/2))/(  (Daux(1)*1e-3)^4-
(Daux(2)*1e-3)^4  *pi  );
        k=k+1;
```

```
    end
```

```
  end
  end
```

Es fácil para el lector cerciorarse de cómo operan las líneas de código anteriores,.

Una vez llegado a este punto, en el cual todos los nodos tienen las tensiones calculadas, el siguiente código calculara la tensión equivalente para poder tener una tensión de referencia. Una vez más, el código es fácilmente interpretable por el lector ya que anteriormente se ha explicado

```
        for h=1:n;
          Sx=max(Nodo(h).Saxil)+max(Nodo(h).Sflex);
          Txy=max(Nodo(h).Scort)+max(Nodo(h).Stor);
          S1=(Sx/2)+sqrt(  (Sx/2)^2 + Txy^2  );
          S2=(Sx/2)-sqrt(  (Sx/2)^2 + Txy^2  );
          Nodo(h).Seq=sqrt(S1^2+S2^2-(S1*S2));
        end
```

Finalización del lazo.

```
end
```

Final de la función.

1.28.8. *Función: Plot_displaced*

Esta función tiene como principal tarea, dibujar/plotear la estructura deformada. El usuario debe entender que, al tratarse de un modelo de desplazamientos mínimos, estos si se superponen con la estructura no deformada no podrá apreciar la estructura deformada, por ello en líneas anteriores a esta función, se recomienda escalar las deformaciones, con la finalidad de poder visualizar correctamente las deformaciones.

La siguiente línea del código, no es parte de la función, se debe introducir fuera de esta y sirve para poder escalar el vector desplazamiento.

```
deformada=desp; desp=desp*5e1; %escalando la Visualización
```

La función que se muestra a continuación no solo dibuja la estructura deformada, sino, que también tiene como objetivo secundario, poder mostrar visualmente la tensión equivalente que están sufriendo los nodos y elementos elásticos que los interconectan, todo esto mediante una barra codificada con colores.

Al llamar la función, vemos que esta no tiene variables de salida, únicamente necesita variables de entrada, estas variables de entrada son las siguientes; necesita de escalares como en número de barras y nodos “n,m” el vector de barras y nodos, el limite elástico que se ha impuesto al principio, y es función del material a utilizar. Así como también las estructuras Barra y Nodo.

```
function Plot_displaced(n,m,nodos,barras,desp,Limite_elastico,Barra,Nodo,deformada)
```

Una vez dento de la función, se procede a sumar el desplazamiento calculado con el método matricial a cada uno de los nodos, en las siguientes lianas de código podemos observar cómo se lee los nodos originales que el usuario dio y como estos son sustituidos, sumandos por el vector desplazamiento

Esto se realiza mediante un lazo que recorre todo el vector nodos, y con la variable “k” se selecciona únicamente grados de libertad de traslación, por ello al final del código esta variable auxiliar se suma 6 unidades, ya que significa que para el siguiente nodo, empezara 6gdl después, saltándose los gdl de giro.

```
Figure(1); k=1;
for i=1:n; %suma de desplazamientos en los nodos
   nodos(i,1)=nodos(i,1)+desp(k);
   nodos(i,2)=nodos(i,2)+desp(k+1);
   nodos(i,3)=nodos(i,3)+desp(k+2);
   k=k+6;
end
```

Finalizamos la sustitución de los valores del vector nodos y cerramos el lazo

Volvemos a abrir un lazo que recorre de nuevo el vector nodos

```
for i=1:n;
```

Esta vez se utiliza el lazo para poder dibujar la estructura en una figura, por ello se guardan en las variables “X, Y, Z” las coordenadas de cada nodo, además de calculase un cociente para entender la proporción que tiene la tensión equivalente en verso al límite elástico

```
hold on
   x=nodos(i,1);
   y=nodos(i,2);
   z=nodos(i,3);
   cociente=Nodo(i).Seq/Limite_elastico;
```

Una vez calculado este cociente, se genera una condición para escoger el color que la función dibujara en la figura.

En esta condición se evalúa si el cociente esta entre unos ciertos valores, como se ha convertido en una relación, si esta relación es mayor al 95% del límite elástico, este se dibujara de color rojo, dando a entender que ese nodo esta fuera del límite elástico

NOTA: por múltiples Hipótesis y por la naturaleza del método, el autor se guarda un 5%, pero el lector debe tener en cuenta que valor de coeficiente de seguridad desea. En cuanto a cuál es el valor de Coeficiente de seguridad deberíamos imponer, es función de la calidad del método, de las condiciones de contorno propuestas o de la normativa.

Si el nodo no se encuentra enzima de esos valores, entonces seguirá siendo evaluada la condición, se ve fácilmente que existen unos rangos para que las condiciones sean cumplidas, y cada una de estas, dibuja el nodo de un color, dependiendo de la condición que cumpla.

```
if (cociente>=0.95)
    plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','red')
elseif (cociente>=0.75) && (cociente<=0.95)
    plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','[1 0.5 0.2]')
elseif (cociente>=0.55) && (cociente<=0.75)
    plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','yellow')
elseif (cociente>=0.25) && (cociente<=0.55)
    plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','green')
elseif (cociente>=0) && (cociente<=0.25);
    plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','blue')
end
```

```
end
```

Finalizamos la condición y el lazo para cada nodo, en este paso los nodos han sido dibujados en la figura

A continuación se muestran las barras desplazadas, el procedimiento es muy similar al anterior, se empieza con un lazo que recorre todas las barras

```
for i=1:m;
```

Dentro de este lazo podemos encontrarnos de nuevo con el cociente que distribuirá la tensión en distintos colores, las variables “aux, h, v, t” ya fueron comentadas en la función “**plot_undisplaced**”. Pero a modo de recordatorio crea una serie de puntos espaciales los cuales sirven para unirlos mediante una línea, en nuestro caso de trazo discontinuo.

```
hold on
   cociente(i)=((Barra(i).Seq)/Limite_elastico); %  porcentage de limite elastico
   aux=nodos(barras(i,:),:);
   h(1)=aux(1,1);
   v(1)=aux(1,2);
   t(1)=aux(1,3);
   h(2)=aux(2,1);
   v(2)=aux(2,2);
   t(2)=aux(2,3);
```

Al igual que en el apartado anterior, dibujara la barra dependiendo del cociente, como se muestra en el código siguiente.

```
      if (cociente>=0.95)
         plot3(h,v,t,'--red')
      elseif (cociente(i)>=0.75) && (cociente(i)<=0.95)
         p = plot3(h,v,t,'--');
         set(p,'Color','[1 0.5 0.2]')
      elseif (cociente(i)>=0.55) && (cociente(i)<=0.75)
         plot3(h,v,t,'--yellow')
      elseif (cociente(i)>=0.25) && (cociente(i)<=0.55)
         plot3(h,v,t,'--green')
      else (cociente(i)>=0) && (cociente(i)<=0.25);
         plot3(h,v,t,'--blue')
      end
```

Finalizamos la condición y el lazo.

```
end
```

En este apartado se crea en la figura una barra de colores, esta barra esta para poder dar una ayuda al usuario, a modo de leyenda.

Se llama la sub función “colormap” en su modalidad “Jet”, esta modalidad se puede cambiar a gusto, pero justamente esta modalidad brinda una barra de colores desde el rojo hasta el azul, como cualquier programa de elemento finito, con esto conseguimos que el usuario se encuentre en un entorno conocido.

Dentro del “colormap” podemos modificar ciertas características, tale como los “Ticks”, que son las particiones que mostrara en la figura de la barra de colores, estas porciones son proporciones a las deformaciones y el vector que las defines es proporcional entre toda la barra de color, es decir en la posición 0.5, esta se modificara a la mitad de la barra de color y este pondrá en la leyenda que el valor es de 55% del límite elástico mostrado en MPa.

Como se puede observar en el código, estas están parametrizadas, mostraran siempre una progresión desde el límite elástico admisible del material

```
colormap(jet)
c =
colorbar('Ticks',[0.25,0.5,0.75,0.9,1],'TickLabels',{int2str(0.25*Limite_elastico*
1e-6),...
   int2str(0.55*Limite_elastico*1e-6),int2str(0.75*Limite_elastico*1e-
6),int2str(0.95*Limite_elastico*1e-6),...
   [int2str(Limite_elastico*1e-6) '\sigma adm' ]});
c.Label.String = 'Tensión de fluencia [MPa]';
hold off
```

Con esto se finaliza la primera figura, la que dibuja las tensiones y visualiza las deformaciones.

```
end
```

En este apartado que describimos a continuación, es la creación de una segunda figura, en esta figura se mostrara gráficamente lo mismo que en la anterior, con la salvedad de que en vez de mostrar las tensiones se muestran las deformaciones totales.

Iniciamos la figura Nº 2 y inicializamos la variable “K”.

```
figure (2)
k=1;
```

como siempre utilizamos un lazo para cada nodo, este lazo se utiliza para calcular el vector de desplazamiento total de las coordenadas de 3D, la variable auxiliar “K” esta para escoger únicamente los GDL de traslación que son los 3 primeros de cada nodo de a un total de 6GDL por nodo. Pues en la variable “aux3” se guarda el vector desplazamiento de cada nodo.

```
for i=1:n; %suma de desplazamientos en los nodos
   aux3(i)=sqrt( ( desp(k) )^2 + ( desp(k+1) )^2 + ( desp(k+2) )^2);
   k=k+6;
end
```

A continuación se calcula el máximo valor del vector y se guarda en la variable “Masdesp”

```
Maxdesp=max(aux3);
k=1;
```

Se inicia de nuevo un lazo para cada nodo.

```
for i=1:n;
```

Como ya se ha visto anteriormente este lazo pretende guardar las coordenadas de los nodos, así como el cociente entre la deformación total del nodo con la máxima deformación del sistema.

```
hold on
   x=nodos(i,1);
   y=nodos(i,2);
   z=nodos(i,3);
   cociente=aux3(i)/Maxdesp;
   k=k+6;
```

De nuevo, se plotean con diferentes colores a través del cumplimiento o no de las condiciones impuestas por el código.

```
    if (cociente>=0.95)
       plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','red')
    elseif (cociente>=0.75) && (cociente<=0.95)
       plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','[1 0.5 0.2]')
    elseif (cociente>=0.55) && (cociente<=0.75)
       plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','yellow')
    elseif (cociente>=0.25) && (cociente<=0.55)
       plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','green')
    elseif (cociente>=0) && (cociente<=0.25);
       plot3(x,y,z,'oblack','MarkerSize',10,'MarkerFaceColor','blue')
    end
```

Finalizamos la condición y el lazo.

```
end
```

En este apartado del código, se dibujan las barras desplazadas en la figura 2. La única diferencia con el anterior código, es que todas ls barras son dibujadas sin condiciones de promedio, es decir todas serán dibujadas de color negro.

```
for i=1:m;
```

```
hold on
    aux=nodos(barras(i,:),:);
    h(1)=aux(1,1);
    v(1)=aux(1,2);
    t(1)=aux(1,3);
    h(2)=aux(2,1);
    v(2)=aux(2,2);
    t(2)=aux(2,3);

plot3(h,v,t,'--black')
```

```
end
```

Finalizamos

A continuación se muestra el código que calcula la proporción de desplazamiento máximo para poder configurar la barra de colores que tenemos como leyenda

Por ello cogemos la variable “deformada” que no es más que el vector “desplazamiento” pero guardado sin tener en cuenta escalamientos que anteriormente hemos modificado. Las líneas de código que suceden son fácilmente interpretables por el lector.

```
k=1;
for i=1:n;
   aux4(i)=sqrt( ( deformada(k) )^2 + ( deformada(k+1) )^2 + (
deformada(k+2) )^2);
   k=k+6;
end
```

```
 Maxdeformada=max(aux4);
```

```
colormap(jet)
c = colorbar('Ticks',[0,0.5,0.9,1],'TickLabels',{int2str(0),...
   int2str(0.55*Maxdeformada*1e6),int2str(Maxdeformada*1e6),...
   [ 'Def Max' ]});
c.Label.String = 'Deformación total [micrometros]';
hold off
```

Finalizamos lazo y función.

```
end
```

Con esto finalizamos el ploteo en dos figuras distintas, con tensiones y deformaciones.

1.28.9. *Función: resultados*

Esta función tiene dos principales objetivos, el primero y más claro es el de evaluar la resistencia torsional de la estructura que se está estudiando, esto como es lógico solo está disponible para estructuras de este tipo.

Esta función, reporta 3 distintas variables, "Rigidez, Masa, Kte" y necesita para su funcionamiento, el vector “nodos, desp y CCF” además como no de la estructura Barra

NOTA: esta función normalmente se encuentra antes de la función de “plot_undesplaced” ya que el vector “desp” es modificado.

```
function [Rigidez,Masa,Kte] = resultados(nodos,desp,Barra,CCF)
```

La teoría de este cálculo se ha explicado detalladamente en el capítulo 5, además la formula analítica de la rigidez es extraída directamente de la ecuación 5.2

```
Rigidez= (abs(CCF(1,3))*nodos(CCF(1),2))/atand((-desp(CCF(1,1)*6-
3)+desp(CCF(2,1)*6-3))/(2*nodos(CCF(1),2)));
```

A la vez, se llama el vector de la estructura Barra.Masa y se calcula el total y se reporta para saber la masa total de la estructura.

```
Masas=[Barra.Masa];
Masa=sum(Masas);
```

A demás se calcula como se ha dicho en el capítulo 5, la frecuencia natural de modo de torsión

```
w=(1/2*pi)*sqrt(Rigidez/Masa);
```

Finalización de la función

```
end
```

El segundo objetivo de esta función, es la de la clasificación de la estructura, con esta clasificación mediante estos 3 valores, se puede inicializar una optimización, encontrando el valor de la variable “Kte” mayor a una resistencia torsional o a un menor peso, dependiendo de lo que se busque.

1.29. Programa (Condiciones de contorno de solicitud real)

Al tratarse de dos simulaciones distintas el **núcleo del programa** debe ser distinto y diferenciado, esto se debe a que ambos códigos se solaparían y la programación no sería correcta dando como resultado el fallo.

Por ello para cada simulación ya sea “TS” que es el que anteriormente hemos descrito como simulación de resistencia torsional. Como para “RS” que será el que abarcaremos en esta sección, y es el que tratar de la solicitación real.

Cabe decir que las funciones de la simulación “TS” son muy parecidas o iguales que las que explicaremos a continuación.

Una de las principales diferencias es que la estructura “Barras” se complementa con otra estructura llamada “CGBarras” para posteriormente unirlas en la estructura “Barras” ya que esta es la estructura principal y la que utiliza el núcleo del programa

Además se ha cambiado a lo anteriormente explicado todas las inicializaciones de los lazos, cambiando la longitud máxima de “m” a la longitud máxima de la estructura “Barras” para así tener en cuenta también los elementos fijos.

Las siguientes líneas de código son líneas adicionales en comparación con todo lo anteriormente explicado, y estas están en el programa principal. Solamente están para poder inicializar, leer y computar todo lo referente a CG y elementos rígidos

```
CGnodos= xlsread(file,'CG','B2:D20')*1e-3; % mm a Metros
CGbarras= xlsread(file,'CG','G2:H40');
CGdiametros= xlsread(file,'CG','J2:K40');
E_CG=1e10; %CG K==inf
G_CG=1e10; %CG K==inf
CGn=length(CGnodos);
CGm=length(CGbarras);
Colors=char('y','b','w','g','k','m','r','y','b','g');
CGmass= xlsread(file,'CG','E2:E11');
```

Como podemos observar a parte de lo dicho anteriormente también se le da un valor de resistencia infinita a los elementos rígidos, o mejor dicho, un valor de elasticidad muy alto en comparación a otros materiales, así como su diámetro exagerado.

Entrando más profundamente en apartado de condiciones de contorno podemos ver que se inicializa un vector de aceleración “acc” este vector es el que proporcionara una dirección y una magnitud de fuerza a los CG

```
Acc=[0 5 1];
```

Seguidamente se inicializan los vectores de fuerza, estos se calculan a través del vector de aceleración y de la masa de los CG, para ello se genera un lazo que recorre tanto los nodos de la estructura como los nodos de los elementos rígidos, estos últimos son los que poseen los Centros de Gravedad. Dentro del lazo se puede apreciar cómo se introducen las fuerzas.

```
k=1;j=1;
for i=n+1:(CGn+n);
   CCF(k,:)=[i,1,Acc(1)*CGmass(j)*9.81];
   CCF(k+1,:)=[i,2,Acc(2)*CGmass(j)*9.81];
   CCF(k+2,:)=[i,3,Acc(3)*CGmass(j)*9.81];
   k=k+3;
   j=j+1;
end
```

Veamos un ejemplo de lo sucedido:

CCF =

42.0000	**1.0000**	**0**
42.0000	**2.0000**	**538.5690**
42.0000	**3.0000**	**538.5690**
43.0000	**1.0000**	**0**
43.0000	**2.0000**	**401.2290**
43.0000	**3.0000**	**401.2290**

Como se puede observar se genera un vector donde la primera fila se refiere a la numeración del nodo global, la segunda, la posición X,Y ó Z que se encuentra la fuerza, y la tercera, la magnitud.

Esto se procesa y se genera un vector único donde las fuerzas están bien posicionadas. A continuación el código explicado.

```
F=zeros((n+CGn)*6,1);
for i=1:size(CCF,1);
F(CCF(i,1)*6-(6-CCF(i,2)))=(CCF(i,3));
end
```

Una vez realizado esto el código pasa a inicializar el vector de grados de libertad. Como recordamos este vector realiza un empotramiento puro a los vectores introducidos, que en nuestro caso son los apoyos de la suspensión.

```
CCGL=[6 5 10 11 8 16 7 15 28 32 27 33 38 39 40 41]; % nodos de suspensión
% input---> CC empotrado puro
GL=ones((n+CGn),6);
GL(CCGL,:)=0;
GL=transpose(reshape(GL',1,(n+CGn)*6));
```

Generando al fin un vector total que tiene en cuenta la posición y la condición.

Después de estas inicializaciones entramos de lleno al núcleo del programa, con sus distintas funciones.

En los siguientes apartados solo se pondrán los códigos adicionales que no se han explicado en apartados interiores, cabe recordad, que se utilizan también otras funciones para el programa principal y para el núcleo de programación.

1.29.1. *Función: Crear_CGBarras*

El objetivo de esta función no es más que la de crear un conjunto de elementos rígidos del mismo modo que su homólogo, guardando en su interior todos los datos necesarios como son, los acoplamientos, las propiedades físicas.

Cabe decir que esta función reporta los resultados en la estructura “Barras”

```
Barra = Crear_CGbarras(barras,CGm,nodos,CGnodos,CGbarras,E,CGdiametros,G,Barra);
```

```
function [Barra] =
Crear_CGbarras(barras,CGm,nodos,CGnodos,CGbarras,E_CG,CGdiametros,G_CG,Barra);
```

A diferencia de su función homologa, se deben inicializar unas variables para el correcto funcionamiento que se pueden observar en las siguientes líneas de código

```
k=length(barras);
q=length(nodos);
```

La longitud del lazo en este caso solamente abarca los nodos de los CG

```
 for i=1:CGm;
```

Dentro del lazo el procedimiento es similar a la función homologa.

```
  aux=[CGnodos(CGbarras(i,1),:);nodos(CGbarras(i,2),:)];
  u= [(aux(2,1) - aux(1,1));(aux(2,2) - aux(1,2));(aux(2,3) - aux(1,3))];
  Area =((pi/4*(CGdiametros(i,1)^2-CGdiametros(i,2)^2))*0.000001);
  I=(pi*(CGdiametros(i,1)^4-CGdiametros(i,2)^4)/64)*0.000000000001;
  J=2*(pi*(CGdiametros(i,1)^4-CGdiametros(i,2)^4)/64)*0.000000000001;
  L=norm(u);
  A=(E_CG*Area)/L;
  C=(4*E_CG*I)/L;
  D=(6*E_CG*I)/L^2; %Dz es la negativa de Dy
  B=(12*E_CG*I)/L^3;
  T=(G_CG*J)/L;
```

```
  Barra(i+k).nombre = ['CGBarra ' int2str(i)];
  Barra(i+k).Kloc = [A 0 0 0 0 0 -A 0 0 0 0 0;
     0 B 0 0 0 D 0 -B 0 0 0 D;
     0 0 B 0 -D 0 0 0 -B 0 -D 0;
     0 0 0 T 0 0 0 0 0 -T 0 0;
     0 0 -D 0 C 0 0 0 D 0 C/2 0;
     0 D 0 0 0 C 0 -D 0 0 0 C/2;
     -A 0 0 0 0 0 A 0 0 0 0 0;
     0 -B 0 0 0 -D 0 B 0 0 0 -D;
     0 0 -B 0 +D 0 0 0 B 0 +D 0;
     0 0 0 -T 0 0 0 0 0 T 0 0;
     0 0 -D 0 C/2 0 0 0 +D 0 C 0;
     0 D 0 0 0 C/2 0 -D 0 0 0 C];
  Barra(i+k).base=[];
  Barra(i+k).Kglob=[];
  Barra(i+k).base_vector =[];
  Barra(i+k).vector= u;
  Barra(i+k).acoplamientos = [CGbarras(i,1)+q CGbarras(i,2)];
  Barra(i+k).nodos= aux;
  Barra(i+k).area= Area;
  Barra(i+k).longitud= (L);
  Barra(i+k).inercia= (I);
```

```
 end;
```

```
end
```

Finalizacion de la funcion

1.29.2. *Función: Peso propio*

Dentro de esta función se tiene en cuenta la fuerza que ejercen las barras de la estructura principal en la aplicación de un vector de aceleración.

```
function [F] = PesoPropio(F,m,Barra,Acc)
```

Esta función necesita para su correcto funcionamiento, variables como, el vector fuerza, vector aceleración, numero de barras normales y la estructura “Barras”

```
for i=1:m;
```

Se crea un lazo para que sea recorrido durante todo el vector de barras.

En el interior del lazo el código presentado lee los acoplamientos de la estructura “Barras” y escribe sobre el vector de fuerzas, lo que anteriormente se hallaba más la mitad del peso de la barra correspondiente, y esto lo hace teniendo en cuenta en qué grado de libertad debe posicionarlo y si el vector de aceleración da una aceleración que genere una fuerza.

```
F(Barra(i).acoplamientos(1)*6-5)=(F(Barra(i).acoplamientos(1)*6-
5)+(Barra(i).Masa/2)*Acc(1)*9.81);%Nodo 1 barra (i) X
F(Barra(i).acoplamientos(1)*6-4)=(F(Barra(i).acoplamientos(1)*6-
4)+(Barra(i).Masa/2)*Acc(2)*9.81);%Nodo 1 barra (i) Y
F(Barra(i).acoplamientos(1)*6-3)=(F(Barra(i).acoplamientos(1)*6-
3)+(Barra(i).Masa/2)*Acc(3)*9.81);%Nodo 1 barra (i) Z

F(Barra(i).acoplamientos(2)*6-5)=(F(Barra(i).acoplamientos(2)*6-
5)+(Barra(i).Masa/2)*Acc(1)*9.81);%Nodo 2 barra(i) X
F(Barra(i).acoplamientos(2)*6-4)=(F(Barra(i).acoplamientos(2)*6-
4)+(Barra(i).Masa/2)*Acc(2)*9.81);%Nodo 2 barra(i) Y
F(Barra(i).acoplamientos(2)*6-3)=(F(Barra(i).acoplamientos(2)*6-
3)+(Barra(i).Masa/2)*Acc(3)*9.81);%Nodo 2 barra(i) Z
```

```
End; End
```

Finalizamos la función

1.29.3. *Función: Plot_CG*

En esta función se dibuja la figura, donde podemos apreciar la localización de los CG así como su escalamiento y también las barras rígidas

```
function Plot_CG(CGmass,CGnodos,Colors,CGm,CGbarras,nodos)
```

Esta función necesita para su correcto funcionamiento, variables relacionadas con los CG tales como, el color, la masa, la localización, etc.

El procedimiento es similar al de otras funciones de Ploteo, como código especial, se puede observar que el código en la sección de Plot3 en el tamaño hay un código que atribuye el tamaño a las esferas de CG.

```
   for i=1:size(CGmass,1); % Plot de CG
   aux=CGnodos(i,:);
   x=aux(1,1);
   y=aux(1,2);
   z=aux(1,3);

plot3(x,y,z,'oblack','MarkerSize',(75*(CGmass(i)/max(CGmass))),'MarkerFaceCol
or',Colors(i));
   end
```

Codigo de dibujo de barras fijas, se visualizan de color rojo discontinuo

```
   for i=1:CGm; % Plot de Elementos rijidos
   aux=[CGnodos(CGbarras(i,1),:);nodos(CGbarras(i,2),:)];
   h(1)=aux(1,1);
   v(1)=aux(1,2);
   t(1)=aux(1,3);
   h(2)=aux(2,1);
   v(2)=aux(2,2);
   t(2)=aux(2,3);
      plot3(h,v,t,'--r','linewidth',0.5)
   end; end;
```

Finalizamos la funció

CAPÍTULO 7: OPTIMIZADOR DE ESTRUCTURAS

Este capítulo inicia el siguiente bloque de este trabajo, en este bloque se trata de explicar la necesidad de la creación de algoritmos optimizadores, para poder tener en cuenta las máximas posibilidades de diseño en búsqueda del compromiso entre varias variables de diseño. En este trabajo se ha creado un algoritmo que lee de un documento Exel, todos los posibles perfiles que dispone el equipo mediante la búsqueda de los mismos en el mercado actual, para calcular cual combinación de ellos es el que menor peso tiene y mayor resistencia torsional posé.

Este optimizador, solamente se basa en la iteración de perfiles, no cambia la posición de los nodos ni tampoco crea o destruye ningún elemento elástico.
Cabe entender que, según qué tipo de normativa se utiliza, el código optimizador debe ser modificado para satisfacerla, dando lugar a más de un código distinto, así mismo se ha creado un código optimizador que es el menos restrictivo de todos, permitido también por la normativa FSAE [1] Apartado de AF (alternative frame rules)

Pues es objetivo de este capítulo, entender el código optimizador e interpretar los resultados dados por el mismo

1.30. Introducción

Para tratar de poder ser lo más claro y conciso, se resolverá un ejemplo de estructura inicial para realizar un ejemplo simple y sencillo, tanto para el computador como para el usuario. Se explicara y se dará un ejemplo final de la solución final. Empezaremos exponiendo este ejemplo, es un ejemplo familiar para el lector, ya que anteriormente se ha utilizado.

Volvamos al Exel, donde el usuario debe introducir los parámetros geométricos de la estructura deseada. Como se ve en la figura 7.1, en el apartado de **Aplicación y Código** que aparecen en un recuadro rojo, es una sección única y exclusivamente para la optimización de la misma.

Nota: la figura 7.1 muestra una pequeña parte del Excel a modo de visualización.

NODOS	X[mm]	Y[mm]	Z[mm]	BARRAS	nodo 1	nodo 2	Aplicación	Código	Dext[mm]	Dint[mm]
1	0,00	-150,00	130,00	1	1	2	FB	3	25,4	22,2
2	0,00	-150,00	405,00	2	2	3	FB	3	25,4	22,2
3	0,00	150,00	405,00	3	3	4	FB	3	25,4	22,2
4	0,00	150,00	130,00	4	4	1	FB	3	25,4	22,2
5	492,20	-208,00	85,00	5	4	2	FB	3	25,4	22,2
6	492,20	-228,75	261,30	6	1	5	FBS	2	26	23,6
7	492,20	228,75	261,30	7	1	6	FBS	2	26	23,6
8	492,20	208,00	85,00	8	2	6	--	1	10	8
9	608,10	0,00	73,75	9	2	12	FBS	2	26	23,6
10	724,00	-207,00	62,50	10	3	14	FBS	2	26	23,6
11	724,00	-227,50	335,00	11	3	7	--	1	30	26

Figura 7.1 *Estructura ejemplo*

- **Aplicación:** En esta columna se da nombre al conjunto de barras que deben tener las mismas propiedades, es decir, en la figura 7.1 aparece las siglas FB, eso significa que las barras de la 1 a la 5 pertenecen al grupo de *Front Bulckhead*, el nombre aquí es arbitrario totalmente y se introducen a razón del usuario, como por ejemplo, si fuera una estructura tipo puente o casa, se podrían utilizar aplicaciones como, Montante, pilar…

- **Código:** El código es la columna que lee el optimizador, es de vital importancia que el usuario introduzca bien la numeración, que cada aplicación tenga asignado un número.

  No existe un número mínimo ni máximo en cuanto a la numeración del código. Si por ejemplo se decide que toda la estructura tenga solamente 2 tipos de perfiles o por el contrario que cada barra pose o pueda poseer un tipo de perfil distinto es elección del usuario, respetando las normativas.

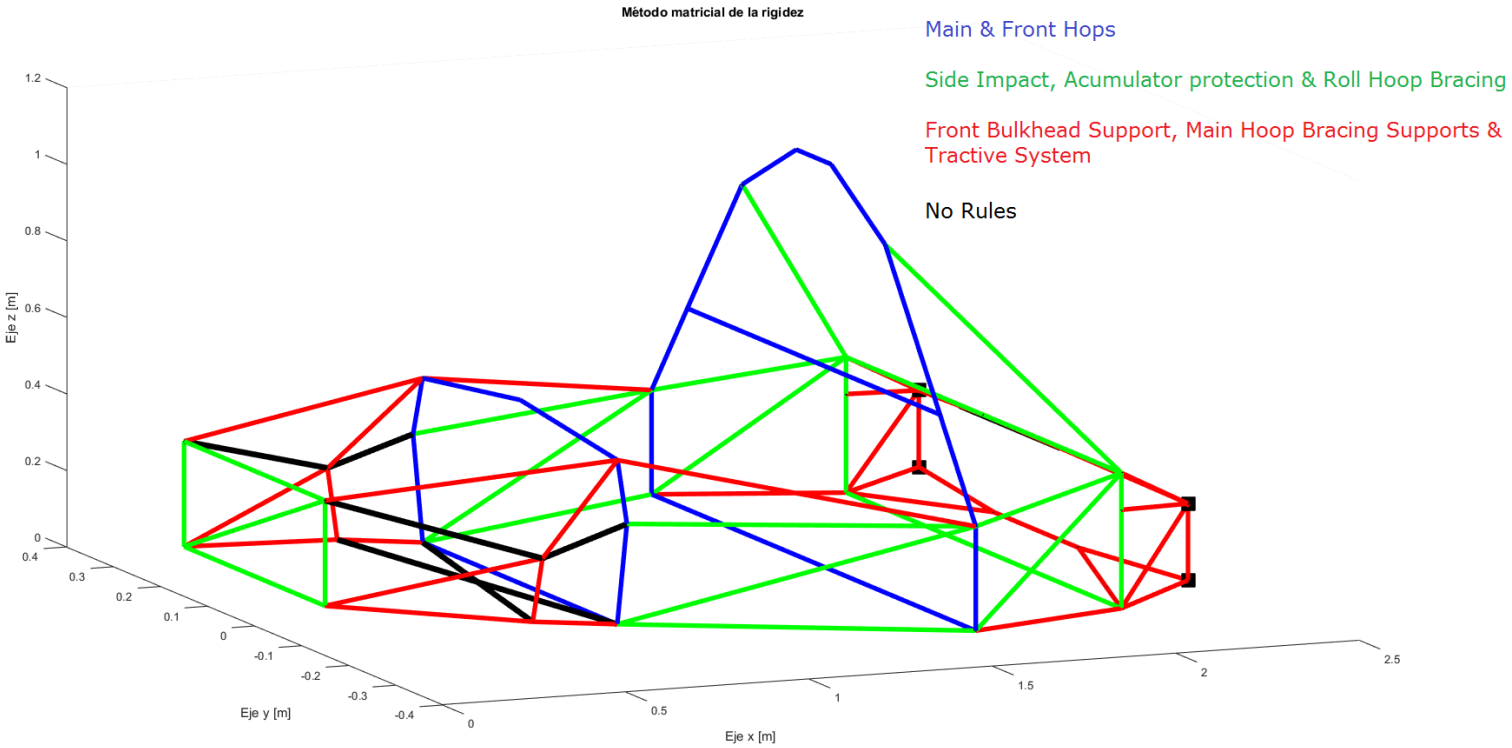


Figura 7.2 *Estructura ejemplo*

En la figura 7.2 se puede observar la estructura de un prototipo de formula student, en esta figura se han dibujado los distintos elementos elásticos, como se pueden percibir observando la leyenda de colores. Esto también se ve reflejado en el Exel, donde en el apartado código y aplicación se refiere.

Una vez programado todo lo anterior, las variables de contorno han sido configuradas y podemos proceder a cargar todos los perfiles que queramos incluir en nuestra optimización.

En la figura 7.3 podemos observar cómo deben introducirse los datos en la segunda hoja de Exel. En la primera columna vemos distribuidos y clasificados los distintos perfiles de menor a mayor diámetro exterior, y dentro de esta clasificación existe otra clasificando los diámetros interiores de menor a mayor, véase la diferencia entre diámetro interior y espesor. Además también está calculado el modulo resistente, que recordamos que la ecuación es la siguiente:

$$Módulo\ Resistente = E \cdot I_y \tag{7.1}$$

Dónde:
E es el modulo elástico del material
I_y es el momento de inercia, en este caso al tratarse de un circulo es simétrico a cualquier otro eje.

	Round (mm)	D_ext (mm)	D_int (mm)	I*E
10	10x1	10	8	6,09E-02
	10x1,5	10	7	7,83E-02
	10x2	10	6	8,97E-02
	10x2,5	10	5	9,66E-02
12	12x1	12	10	1,11E-01
	12x1,5	12	9	1,46E-01
	12x2	12	8	1,72E-01
	12x2,5	12	7	1,89E-01
14	14x1	14	12	1,82E-01
	14x1,5	14	11	2,45E-01
	14x2	14	10	2,93E-01
	14x2,5	14	9	3,28E-01
	14x3	14	8	3,54E-01
16	16x1	16	14	2,80E-01
	16x1,5	16	13	3,81E-01
	16x2	16	12	4,62E-01
	16x3	16	10	5,72E-01
	16x4	16	8	6,33E-01
18	18x1,5	18	15	5,60E-01
	18x2	18	14	6,86E-01
	18x2,5	18	13	7,88E-01
	18x3	18	12	8,68E-01
	18x4	18	10	9,79E-01
	18x5	18	8	1,04E+00
20	20x1	20	18	5,67E-01
	20x1,5	20	17	7,88E-01
	20x2	20	16	9,74E-01
	20x2,5	20	15	1,13E+00
	20x3	20	14	1,25E+00
	20x4	20	12	1,44E+00
	20x5	20	10	1,55E+00
	20x6	20	8	1,61E+00

Figura 7.3 *Estructura*

Para el programa, realmente solo necesita leer las columnas de diámetros exteriores y diámetros interiores, con esto el propio código de Matlab ya se puede calcular las áreas momentos de inercia y demás variables derivadas de a geometría inicia.

1.31. Algoritmo optimizador

Se denomina algoritmo a un grupo finito de operaciones organizadas de manera lógica y ordenada que permite solucionar un determinado problema. Se trata de una serie de instrucciones o reglas establecidas que, por medio de una sucesión de pasos, permiten arribar a un resultado o solución.
En el ámbito matemático, y cuando estamos decididos a llevar a cabo la descripción de uno de esos algoritmos hay que tener en cuenta que se puede efectuar mediante tres niveles. Así, en primer lugar, nos encontramos con el de alto nivel, lo que es la descripción formal y finalmente la tarea de implementación.
Asimismo tampoco podemos pasar por alto que los algoritmos se pueden expresar a través de lenguajes de programación, pseudocódigo, el lenguaje natural y también a través de los conocidos como diagramas de flujo.
Cabe mencionar por último que los algoritmos son muy importantes en la informática ya que permiten representar datos como secuencias de bits. Un programa es un algoritmo que indica a la computadora qué pasos específicos debe seguir para desarrollar una tarea.

En este trabajo utilizaremos un algoritmo secuencial para seleccionar un gran rango de distintas variables. Este algoritmo se basa en la sucesión de cambios y evaluaciones del cambio en frente a otros cambios, para así encontrar en todo los casos la configuración de perfiles, la masa total, la rigidez de la estructura, y la rigidez especifica de esta que hemos llamado “Kte”.

En la figura 7.1 podemos observar que cada grupo de barras que contienen en mismo código, deben tener en común el mismo perfil, a más a mas podemos ver que en la figura 7.2 nos encontramos más de 35 perfiles diferentes, pero esto puede ser tan grande como la disponibilidad de perfiles en el mercado. Esto significa que cada barra puede tener la opción de tener más de un perfil y a su vez, los perfiles de los otros conjuntos de barras también pueden tener distintos perfiles, con referencia a un único perfil del primer conjunto de barras. Esto nos deja un total de combinaciones que expresamos en la siguiente ecuación:

$$P^C = n^{\underline{o}}\ de\ combinaciones \tag{7.2}$$

Dónde:

C= Código, Conjunto de barras
P= Perfiles

Para que el lector tenga una referencia de la cantidad de combinaciones que se pueden tener en una estructura de un prototipo de Formula Student, teniendo como ejemplo un total de 4 tipos de conjuntos de barras distintas y un total de 70 tipos de perfiles distintos calculados con la ecuación 7.2.

$$70^4 = 24.010.000\ combinaciones$$

Este resultado refleja la ingente cantidad de distintos tipos de combinaciones podemos tener. Aun sabiendo que el cálculo de una estructura.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
DSM	1	0.179 s	0.009 s	
tension	1	0.087 s	0.087 s	
cambio_base	1	0.024 s	0.014 s	
Matriz_global	1	0.019 s	0.019 s	
Crear_barras	1	0.018 s	0.014 s	
VectorToBase	1	0.012 s	0.010 s	
dot	774	0.010 s	0.010 s	
solver	1	0.009 s	0.009 s	
int2str	86	0.003 s	0.003 s	
cross	86	0.002 s	0.002 s	
resultados	1	0.001 s	0.001 s	

Figura 7.4 *MATLAB performance time*

Utilizando la herramienta que nos brinda Matlab para poder mejorar el código, en cuestión de velocidad de procesamiento, vemos que una sola estructura tarda unos 0.179s. Realmente es un cálculo cuasi instantáneo, peor si lo multiplicamos con el total de combinaciones nos da un resultado de:

$$24.010.000\ combinaciones \cdot 0.179 segundos = \ 4297790 s$$

Esto son aproximadamente **50 días de cálculo.**

NOTA: Dependiendo del computador y de la optimización de rendimiento del mismo, esto podría traducirse en menos o más tiempo de computación, para tener una referencia este cálculo se ha realizado con un INTEL CORE i7 y 16Gb RAM, teniendo en cuenta que el programa Matlab está instalado en una unidad solida de memoria.

Al no poseer tal cantidad de días para poder diseñar la estructura, se deben tener consideraciones para reducir el tiempo total del algoritmo.
Hay que aclarar, que en sí, este método de optimizar la estructura mediante este algoritmo, es en que más variables abarcan y por ende es el que menos fallos hipotéticos puede tener. En contrapartida se ve la cantidad de tiempo que se necesita.
Para reducir este tiempo de cálculo, sin modificar el código ni el algoritmo, se opta por incurrir en la ecuación 7.2, y entendiendo que se trata de un cálculo exponencial, se puede optar por reducir el número de perfiles.

Pero antes de ver cómo podemos modificar las entradas para así descargar de faena al programa, veamos cómo funciona paso a paso el código del algoritmo.

1.31.1. *El Algoritmo*

A continuación expondremos el diagrama de flujo del funcionamiento lógico del algoritmo, como anteriormente se ha dicho se trata de una secuencia de números enteros.

Para entender mejor este concepto veamos la variable principal que da estructura al algoritmo

Range	Range	Range	Range	Code…

Estructura KK.OPTICODE *MATLAB workspace*

Como podemos ver en la estructura en forma de tabla, esta se genera de forma paramétrica, es decir, que se genera a través de unas variables como son “Range, Code”. Estos es fácil de interpretar que se tratan de la longitud de la estructura “Code” y del valor que se da a las casillas “Range”.

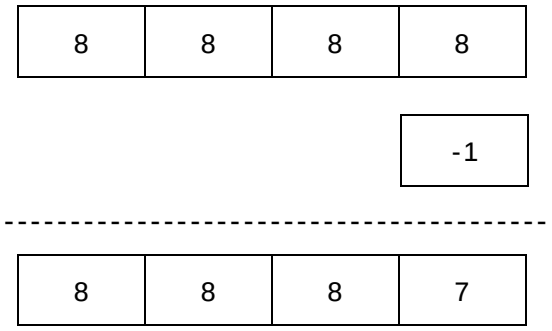
Dicho de otro modo, la longitud se refiere a la cantidad de conjuntos que posee la estructura y en valor numérico que tiene en el interior son las distintas posibilidades de perfiles distintos que puede tener. Aquí es donde la fórmula 7.2 obtiene un sentido más físico.

Pongamos un ejemplo donde la variable “Code” tiene un valor de 4 y “Range” de 8. Esto nos generaría la siguiente estructura:

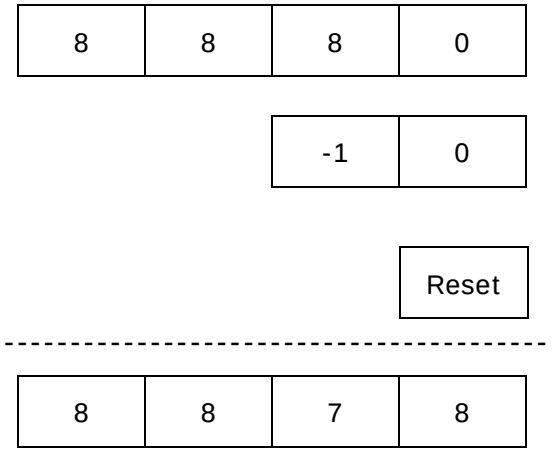
8	8	8	8

Una vez generada esta estructura el algoritmo optimizador da el valor de la todas las casillas de la estructura a los distintos perfiles y calcula sus características, tales como peso, resistencia torsional, frecuencia natural…

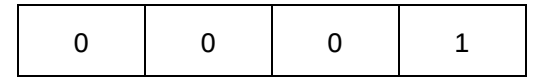
Una vez calculado esto, el algoritmo restara una unidad y recalculará de nuevo



Así hasta llegar al valor nulo, es ese momento el algoritmo restara una decena y reseteara las unidades a su valor predeterminado.



Esta secuencia se repite **mientras** que la última referencia de la estructura no obtenga un valor nulo siendo el siguiente valor el último que opera el algoritmo



Una vez alcanzado este valor el algoritmo finaliza exponiendo los resultados en un gráfico tridimensional.

1.31.2. *El código*

El código es una función que se llama a través del programa principal, esta función reporta una estructura llamada “Iter”, esta estructura guarda para cada iteración calculada distintos vectores, estos vectores son tales como la masa de la estructura calculada, la rigidez torsional, el coeficiente torsional y por último que configuración de barras de esa iteración.

Para funcionar necesita varias variables de entrada y estructura de entrada que se ven detalladas en la siguiente línea de código.

```
function [Iter] =
P_STIFFNESS(file,opti_codigo,optidata,Barra,m,nodos,barras,E,Densidad,G,diam
etros,n,GL,F,CCF)
```

Una vez llamada la función y ya dentro de ellas inician unas variables tales como el “Contador”, que sirve para conocer cuántas iteraciones se han llevado a cabo, inicializamos la máxima cantidad de conjuntos de perfiles pose el problema con la variable “maxcode”. Inicializamos el rango total de perfiles que intervienen en el problema, estos son leídos del Excel con anterioridad y en la variable “range” se guarda el valor de la cantidad de perfiles. Por ultimo guardamos en “Perfil_min” los perfiles básicos que debe tener la estructura.

```
contador=1;
maxcode=max(opti_codigo);
range=length(optidata);
pefil_min= xlsread(file,'Q2:R5');
```

En esta sección se crean unes estructures con la razón de inicializar también variables más complejas, en este caso se crea un lazo que recorre en número de conjuntos de perfiles.

Se crean dos tipos distintas de estructuras.

- **KK.opticode** = esta estructura tiene una importancia vital, ya que es la encargada de controlar la correcta sucesión de iteraciones, además está siendo constantemente modificada por el programa para llevar a cabo su cometido.

La estructura “Daux.D” no es más que guardar correctamente la variable de perfiles básicos.

```
for code=1:maxcode;
   KK(code).opticode=range;
   Daux(code).D=pefil_min(code,:);
end
```

Inicialicemos la variable “k” para utilizarla a posteriori y empezamos con un lazo cerrado, el comando **While** programa que mientras la condición que le sigue se cumpla, siempre estará iterando hasta que la condición inicial no se cumpla más, en este caso la condición que debe cumplir es que la estructura KK.opticode, en su posición (4) esta debe ser superior a 0, si es inferior saldrá del lazo.

```
k=1;
while KK(4).opticode>0
```

Dentro de este lazo se crea otro lazo que recorre todas las barras, pero con una excepción, que únicamente se tendrán en cuenta las condiciones impuestas, y estas son que, solamente esas barras que tengan la misma configuración, se les cambiara el perfil, esto se ve reflejado en la variable diámetros, que cojera de la variable “optidata” teniendo en cuenta la variable “k” dentro de la estructura .

```
   for i=1:m;
      if Barra(i).codigo==k;
        diametros(i,:)=optidata(KK(k).opticode,:);
      end
   end
```

Una vez modificado el vector diámetros, podemos decir que la estructura es totalmente nueva, y debe ser solventada.

Utilizamos entonces todas las funciones del programa Direct Stiffness Method.

```
    Barra =
Crear_barras(m,nodos,barras,E,Densidad,G,diametros,opti_codigo);
    Barra = VectorToBase(Barra,m);
    Barra = cambio_base(Barra,m);
    Kglob = Matriz_global(n,m,barras,Barra);
    [desp] =solver(n,Kglob,GL,F);
    [Rigidez,Masa,Kte] = resultados(nodos,desp,Barra,CCF);
```

Al finalizar el cálculo se guardan los resultados obtenido de esta estructura en la estructura “Iter”, que ya hemos definido anteriormente.

El contador se aumenta en 1, para saber que ya ha realizado una y que va por la siguiente iteración,

Aquí entonces también se calcula la cantidad de iteraciones que se debe realizar y al saber por cual iteración vamos, el código muestra a tiempo real el tanto por ciento de completado.

-

```
    Iter(contador).MasaVar=Masa;
    Iter(contador).RigidezVar=Rigidez;
    Iter(contador).wVar=w;
    Iter(contador).Profile=diametros;
    mplot(contador)=Masa;
    rplot(contador)=Rigidez;
    wplot(contador)=w;
```

```
    contador=contador+1;
    maxiter=((range+1)^maxcode)-((range+1)^(maxcode-1))+1;
    completo=(contador/maxiter)*100 %El porcentaje completado
```

Seguidamente en las siguientes líneas de código se realiza la secuencia, restando a la estructura “KK” una sucesión al conjunto de barras “k”, y seguidamente se re-inicializa la variable “k” a 1.

```
    KK(k).opticode=KK(k).opticode-1;
    k=1;
```

En las siguientes líneas de código, se comprueba si se ha llegado el código al final, con un lazo y una condición para ello, si se cumplen ambas cambia el perfil del conjunto de barras por el inicial

```
  while KK(k).opticode == 0
    if KK(k).opticode == 0;
      for i=1:m;
        if Barra(i).codigo==k;
         diametros(i,:)=Daux(k).D; % para el reinicio
        end
      end
      KK(k).opticode=range;
      k=k+1;
    end
  end
```

```
end
```

Se cierran todos los lazos y se finaliza la función, obteniendo los datos necesarios a través de la estructura “Iter”

A continuación y dentro de la misma función, se dibuja en un gráfico las distintas iteraciones en un gráfico en tres dimensiones, donde se puede comparar visualmente la masa la resistencia torsional y la frecuencia natural.

Adicionalmente se dibuja un circulo verde para mostrar cual de ello tiene mayor frecuencia natural entre masa.

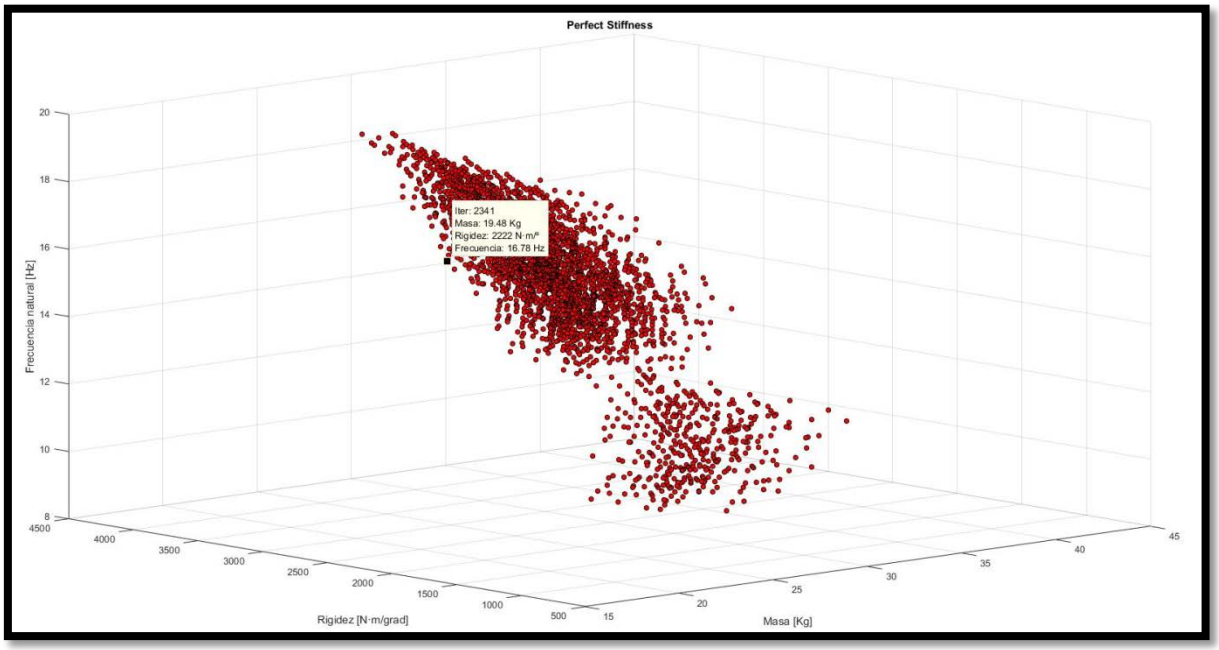
NOTA: Debe entenderse que se ha dividido la masa en esta constante un total de una vez y media debido a que la frecuencia natural ya lo realiza, eso el autor lo ve conveniente porque así el factor penaliza más cuanta más masa tenga.

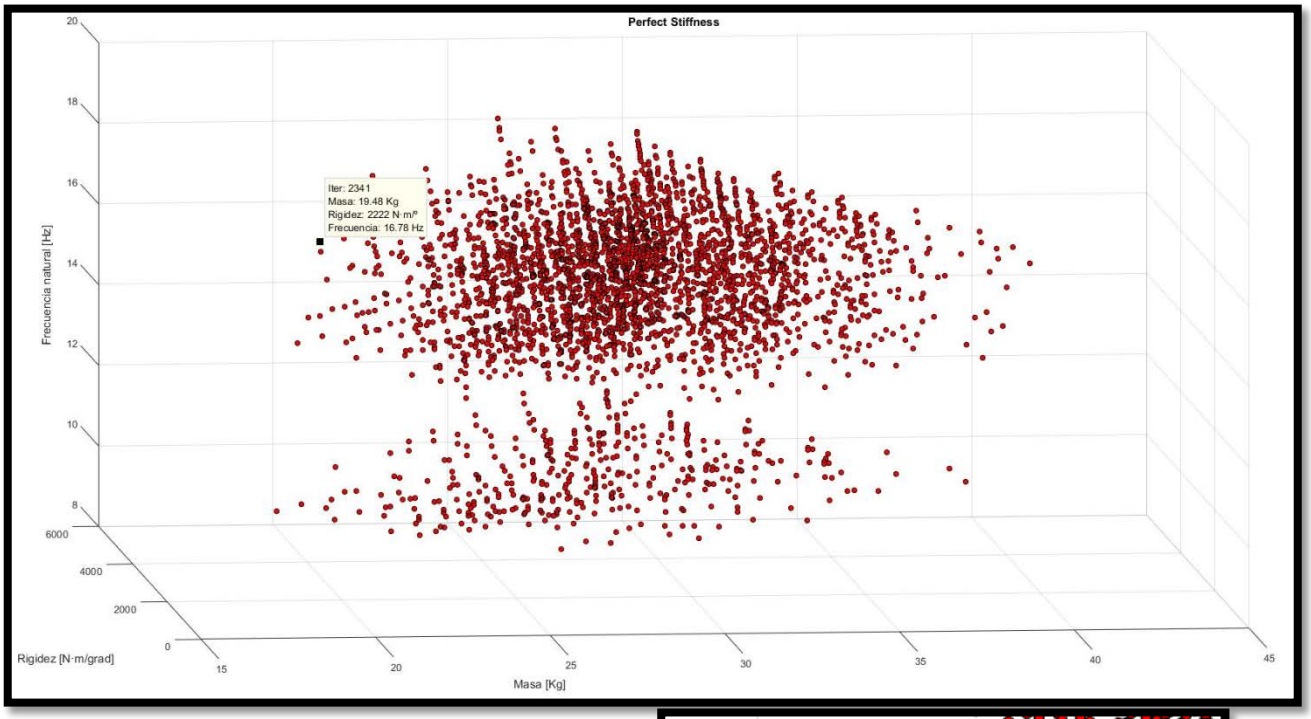
```
figure (1)

for c=1:contador-1;
      Kte(c)=Iter(c).wVar/Iter(c).MasaVar;
end

[M,I] = max(Kte)
plot3( mplot(I),rplot(I),wplot(I),'oblack','MarkerSize',8,'MarkerFaceColor','green')
grid on
hold on
plot3( mplot,rplot,wplot,'oblack','MarkerSize',5,'MarkerFaceColor','red')
title('Perfect Stiffness')
xlabel('Masa [Kg]')
ylabel('Rigidez [N·m/grad]')
zlabel('Frecuencia natural [Hz]')
```

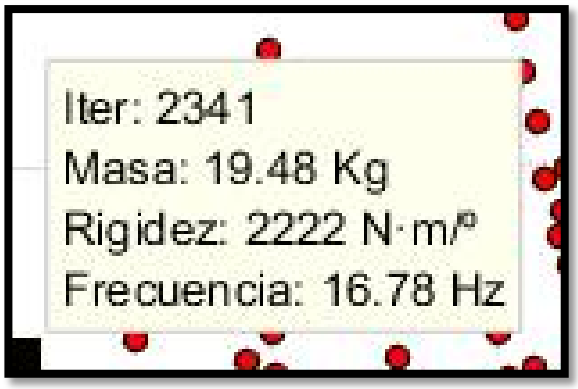
Para entender con mayor detalle que figura se crea en esta función con 4 tipos diferentes de configuraciones y 5 perfiles distintos, veamos las figuras 7.5.

En estas figuras se aprecia una nube de puntos donde el punto más óptimo se muestra en color verde y rojo, y si el usuario accede al cursor de datos del grafico puede ver que iteración es, sus características para así poder recrear la estructura calculada.

```
end
```







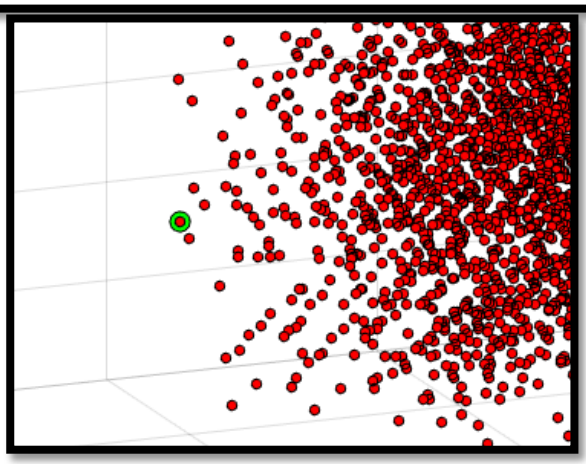


Figura 7.5 *Plot de resultados*

CAPÍTULO 8:

ESTUDIO DE RESULTADOS

En este apartado, nos encargaremos de explicar los siguientes cálculos necesarios para obtener unas conclusiones claras y lógicas, por ello se abarcaran en distintos tipos de disciplinas para poder llegar a un resultado buscado. Este resultado no es más que la búsqueda de una estructura lo más óptima posible.

En este tema se tratara de dar un ejemplo claro y focalizado de la estructura principal de un monoplaza de formula student para años venideros, la intención es poder concluir una estructura principal capaz de abarcar el mínimo peso posible sin comprometer el manejo del monoplaza mejor de Europa.

1.32. Análisis de resultados

Una vez están completadas las optimizaciones pertinentes podemos evaluar de qué manera se comporta el algoritmo optimizador. Para ello se han realizado las optimizaciones con el código mostrado en el capítulo 7 con distintos perfiles para así tener más valores de referencia.

A continuación se mostraran las primeras gráficas, estas dan referencia a la comparativa entre rigidez torsional y la masa del chasis y así veremos qué forma toman los datos y con ello estudiaremos la tendencia que está siguiendo.

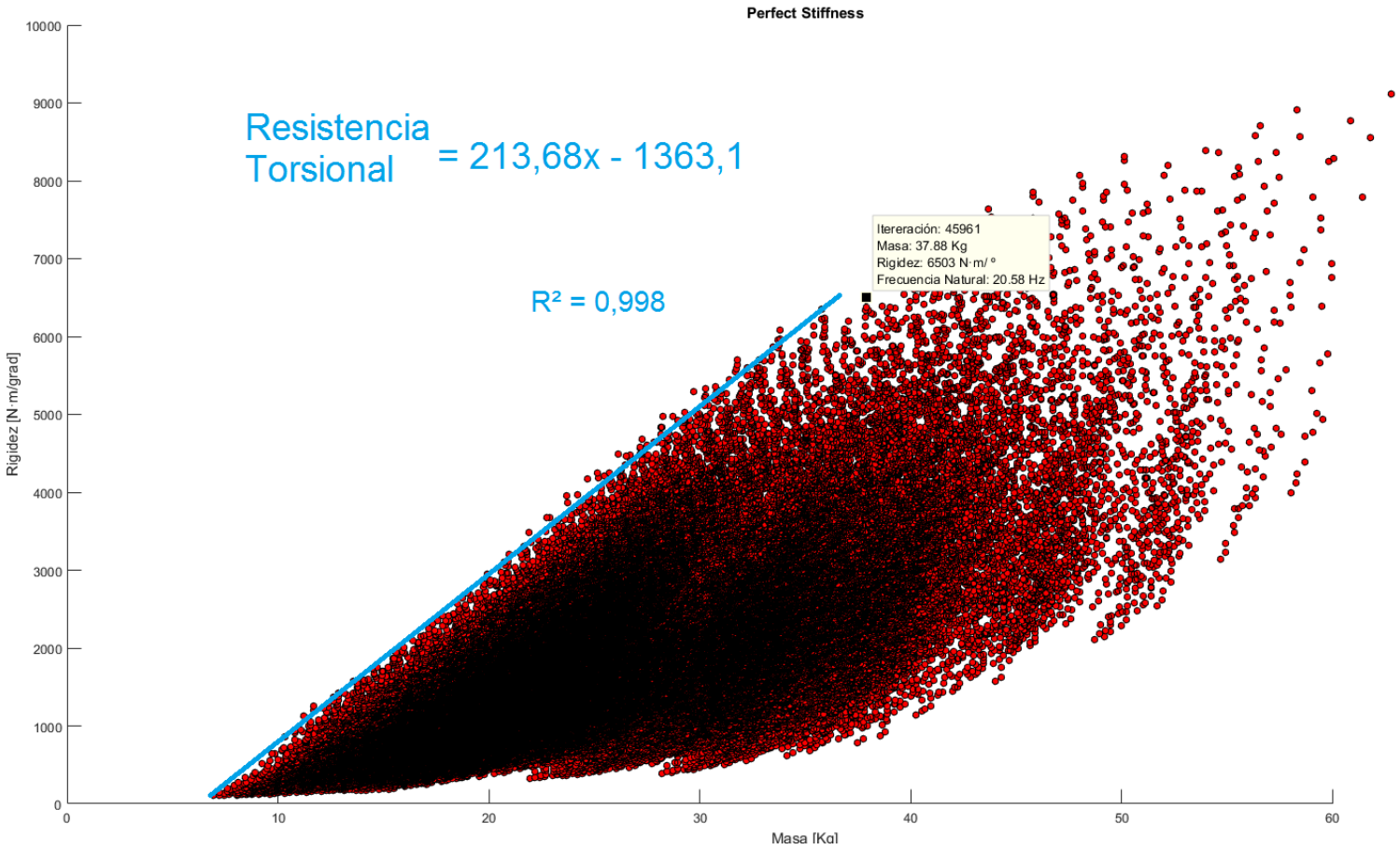


Figura 8.1a *Rigidez vs masa 100k resultados*

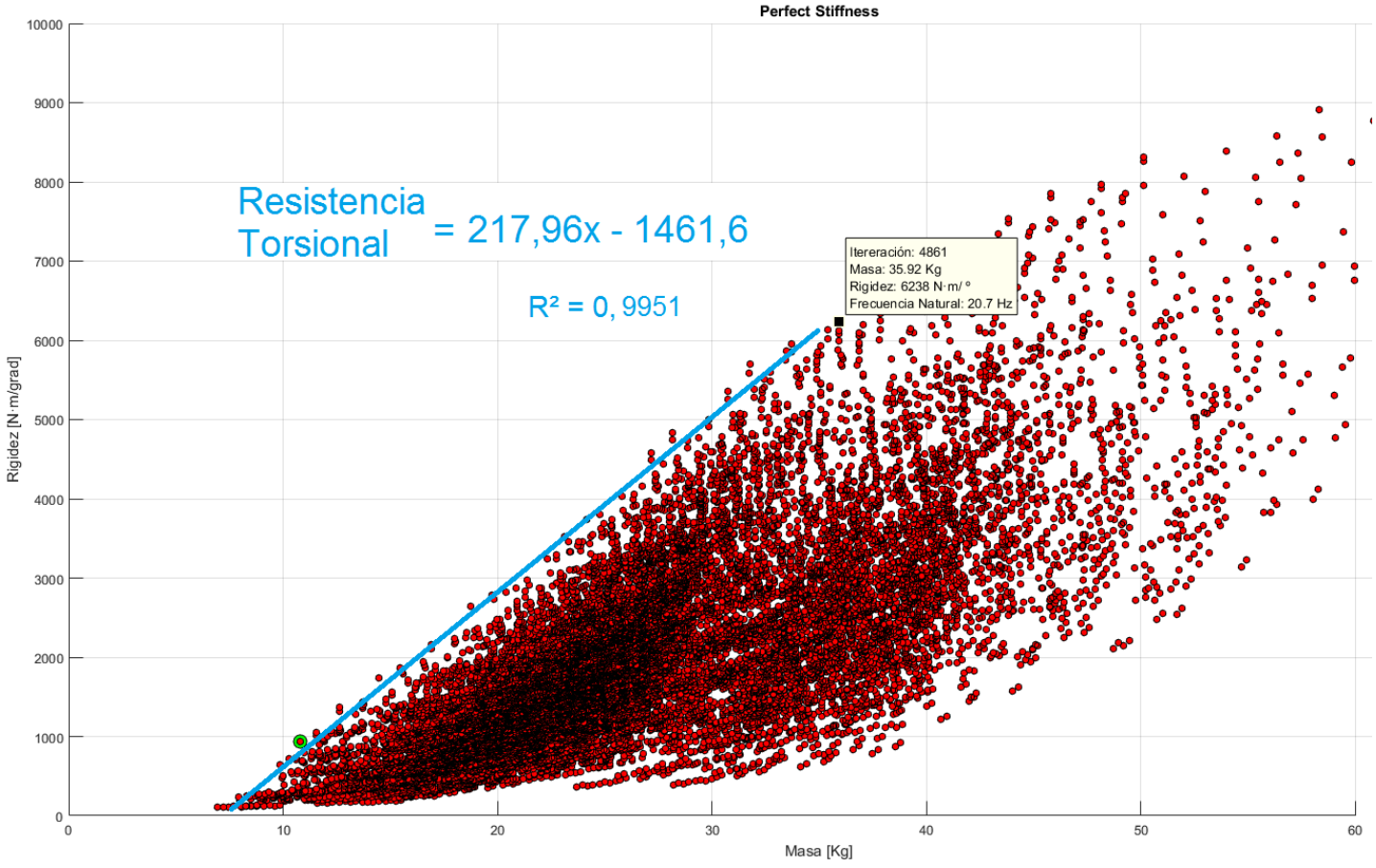
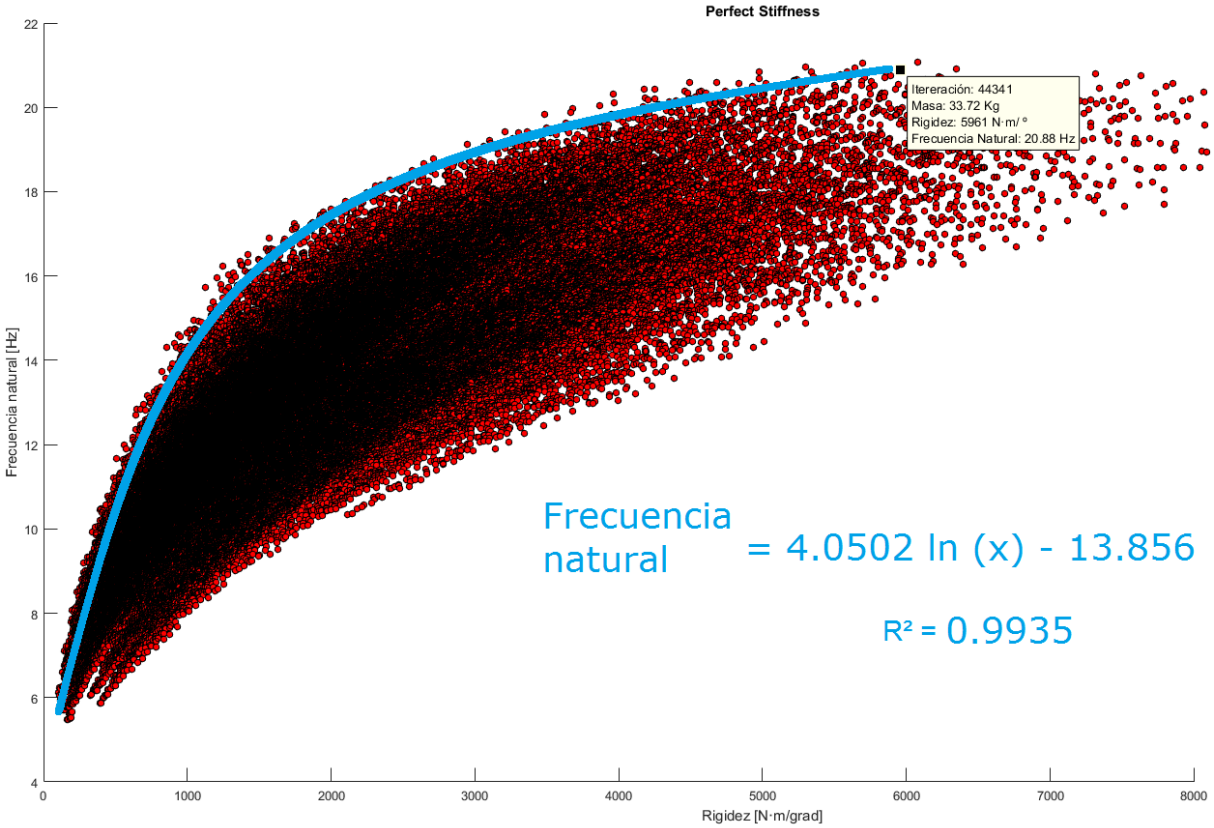


Figura 8.1b *Rigidez vs masa 20k resultados*

En la figura 8.1 podemos observar como el algoritmo optimizador tiene una tendencia a generar unos resultados con pendientes de 215 N·m/º por cada unidad de masa para un tipo de estructura dada.

Veamos que sucede cuando comparamos Rigidez y frecuencia natural.



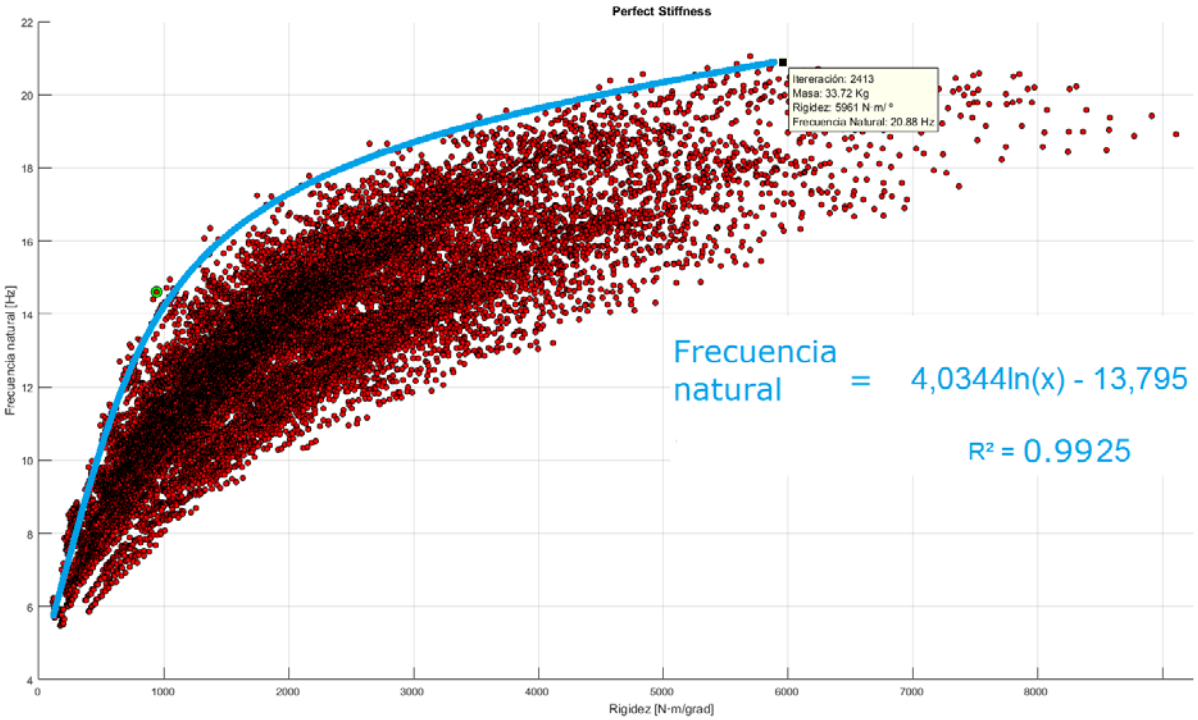


Figura 8.2a y 8.2b *Rigidez vs frecuencia natural 20k resultados*

Podemos observar el mismo patrón que anteriormente, la nube de resultados es más densa pero la tenencia se mantiene, en este caso se trata de una tendencia logarítmica.

Con esto tenemos dos ecuaciones pero que no se pueden comparar directamente, dichas ecuaciones se muestran a continuación y son para una estructura dada:

$$TS = 214\,(masa) - 1361.1 \tag{8.1}$$

$$W_n = 4.05 \cdot \ln(\,TS\,) - 13.795 \tag{8.2}$$

Para encontrar una ecuación que nos relaciona la masa con la frecuencia natural de la estructura estudiada debemos estudiar de nuevo las gráficas:

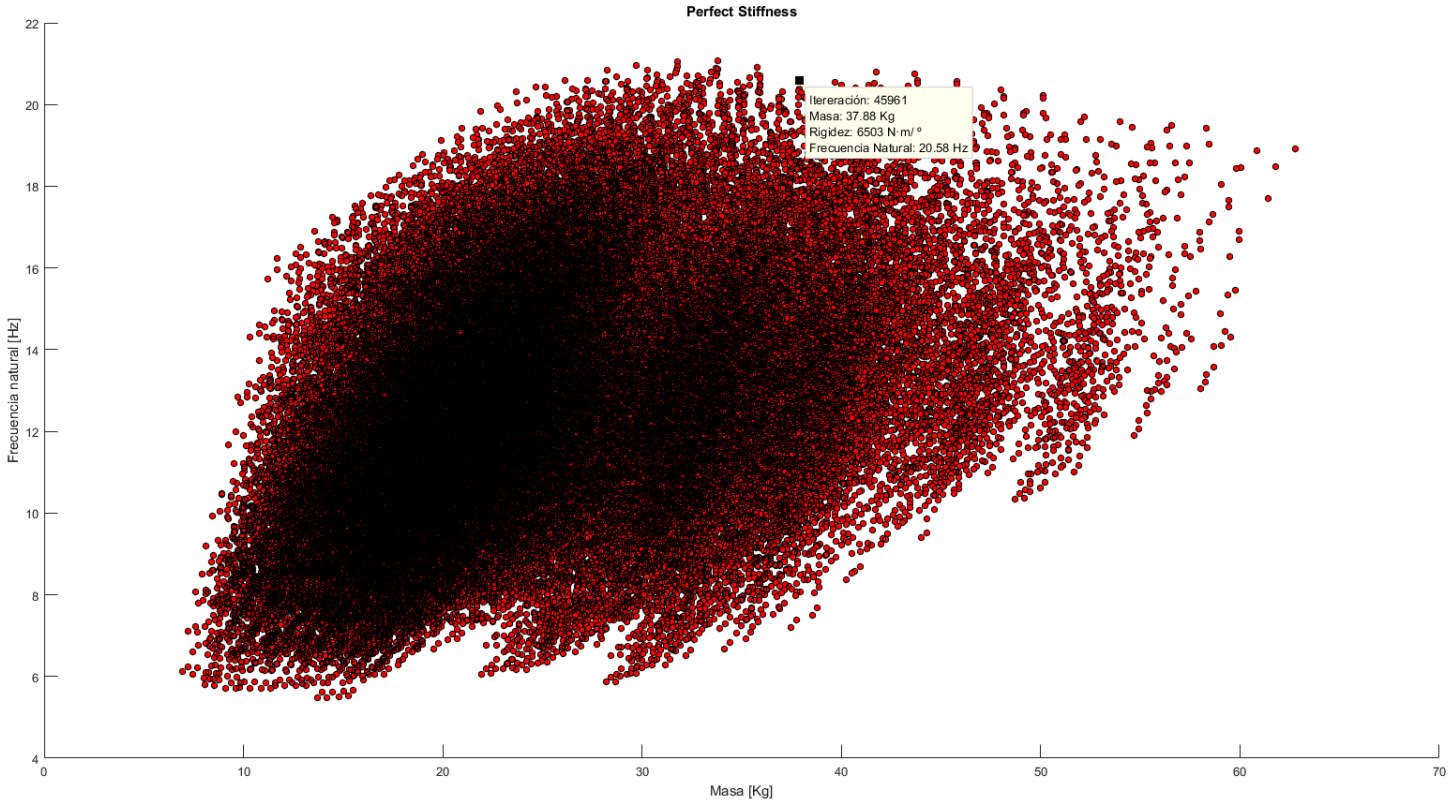


Figura 8.3 *Wn vs masa 100k resultados*

En la figura8.3 se muestra la relación que existe entre la masa y la frecuencia natural a torsión, como se puede observar no es de extrañar que siga una proporción logarítmica, per esta tendencia logarítmica no es precisa del todo, tiene un 93% de precisión por lo que el autor la descarta.

Para poder seguir extrayendo resultados, es necesario poder comparar con otra estructura, de modo que escogeremos el chasis de la temporada 2015 que aun siendo distinto tiene una tipología muy similar, por lo que será un ejemplo ideal para estudiar.

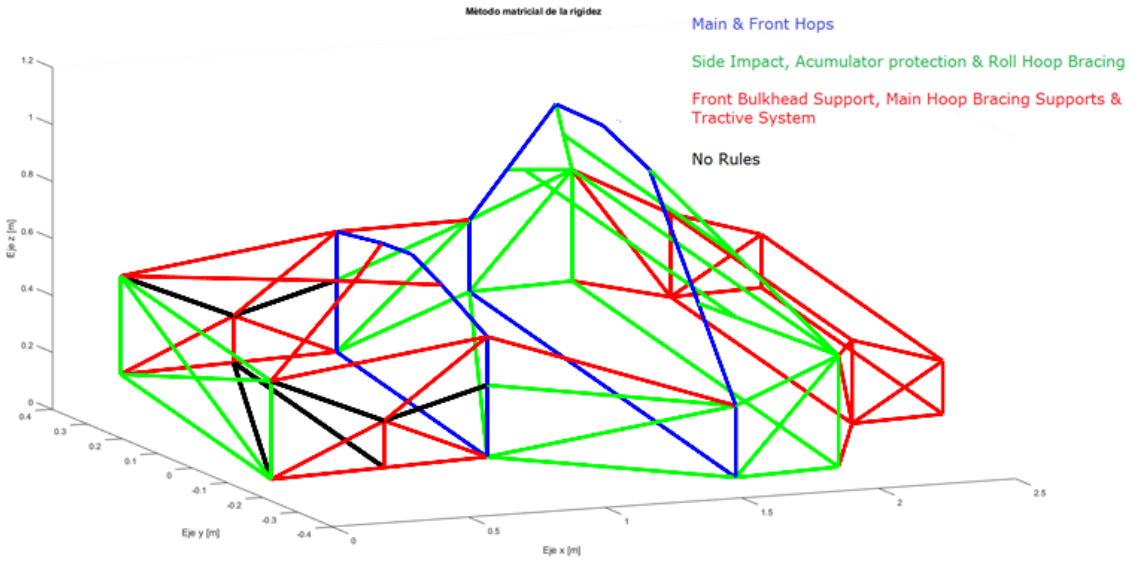


Figura 8.4 *Estructura chasis '15*

En esta estructura se han realizado las mismas operaciones con el algoritmo optimizador con el fin de comparar las tendencias de estas estructuras y sacar resultados más preciso

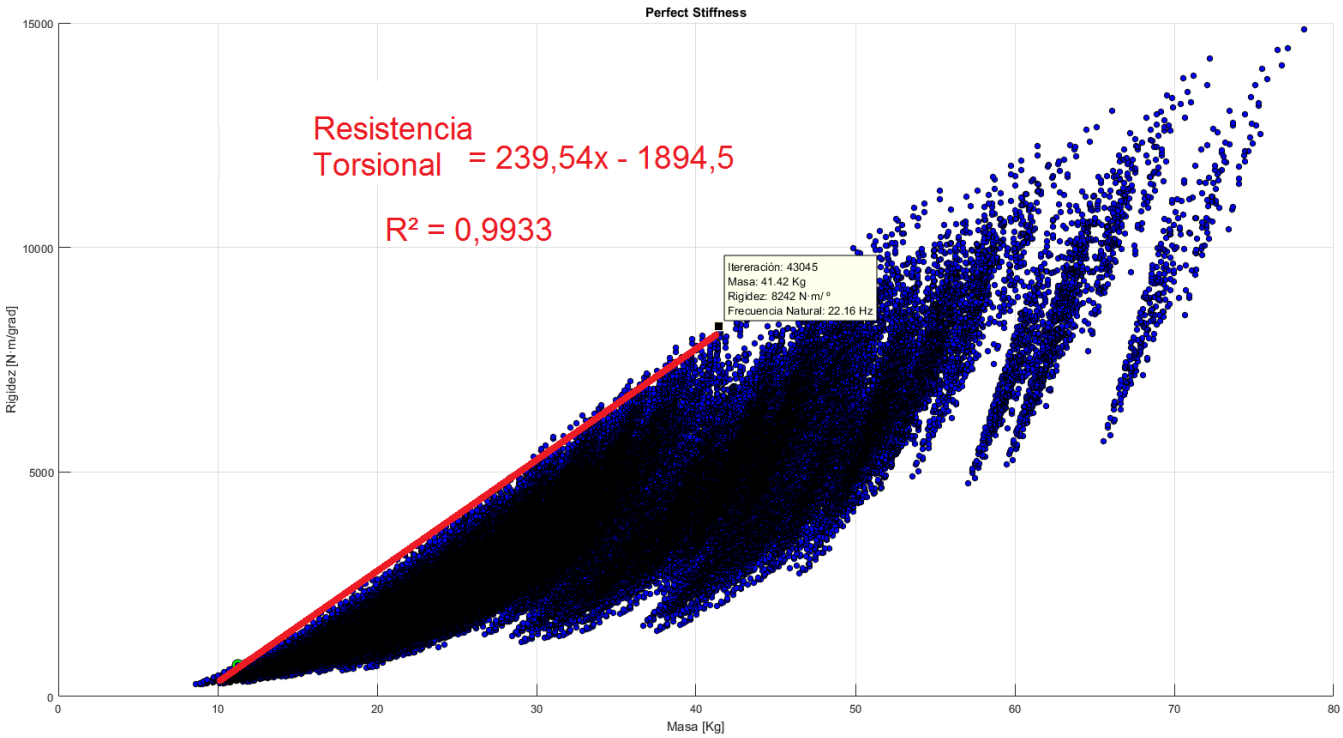


Figura 8.5 *Rigidez vs masa 100k resultados chasis '15*

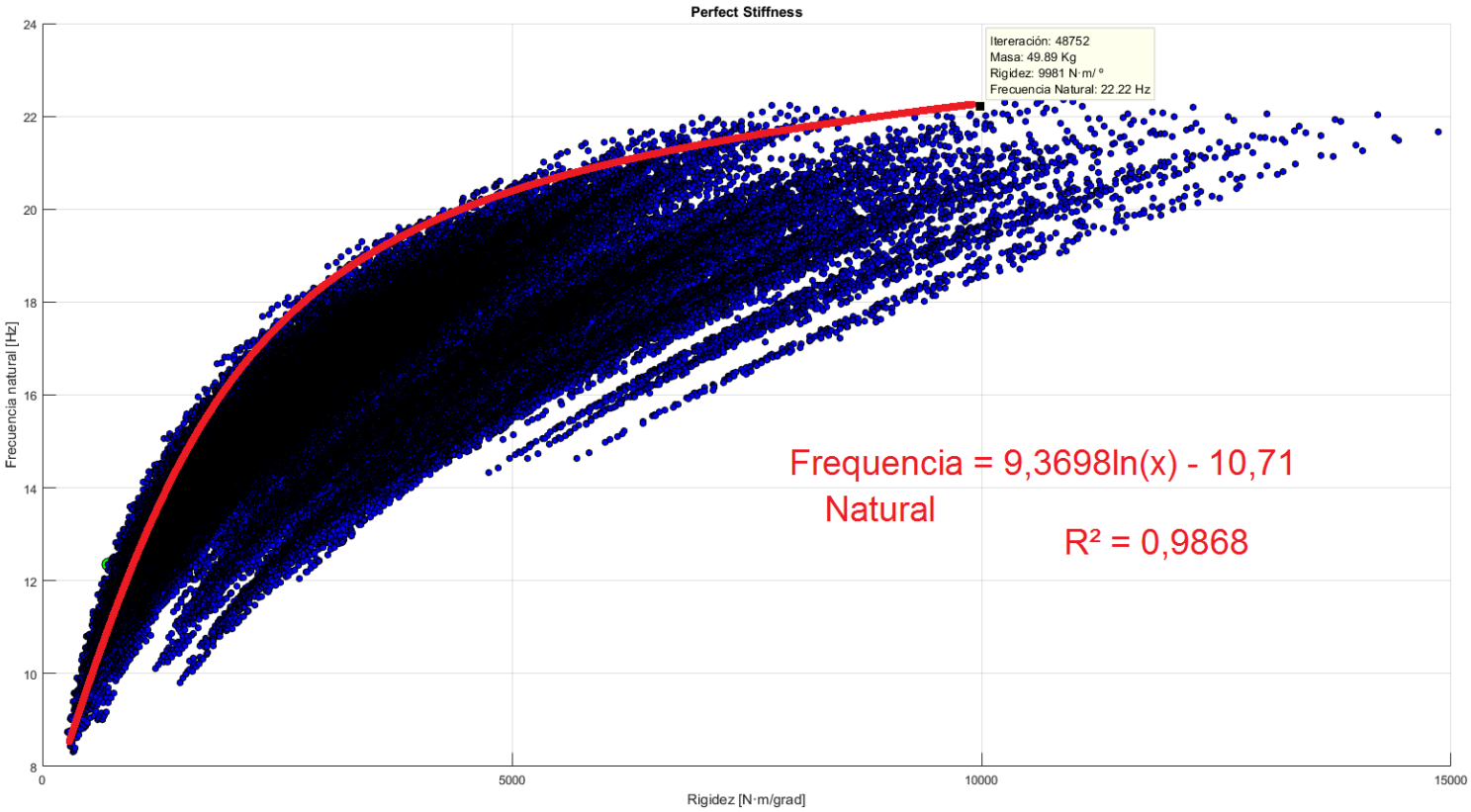


Figura 8.6 *Rigidez vs frecuencia natural 100k resultados chasis '15*

Con esto tenemos las dos ecuaciones que intentan interpretar matemáticamente cómo se comporta el algoritmo

$$TS = 239\,(masa) - 1894.5 \tag{8.3}$$

$$W_n = 9.37 \cdot \ln(\,TS\,) - 10.71 \tag{8.4}$$

Como se puede observar y a modo de conclusión, la geometría aportada por la segunda estructura tiene un comportamiento similar. Sus principales diferencias radican en la pendiente de mejora, que en el segundo caso es mejor, y que la estructura para obtener una torsión mínima debe satisfacer un valor másico mayor que la primera. En la figura 8.7 se puede observar gráficamente cual debe ser la elección de la estructura en verso a estas dos variables en la formula. La estructura roja, es la mejor elección si la resistencia torsional objetivo es menor a 3000N·m/º, sino, la estructura azul será la óptima. En cuanto a modo de vibración, ambos tienen un comportamiento similar por lo que no se debería ser un factor importante a tener en cuenta.

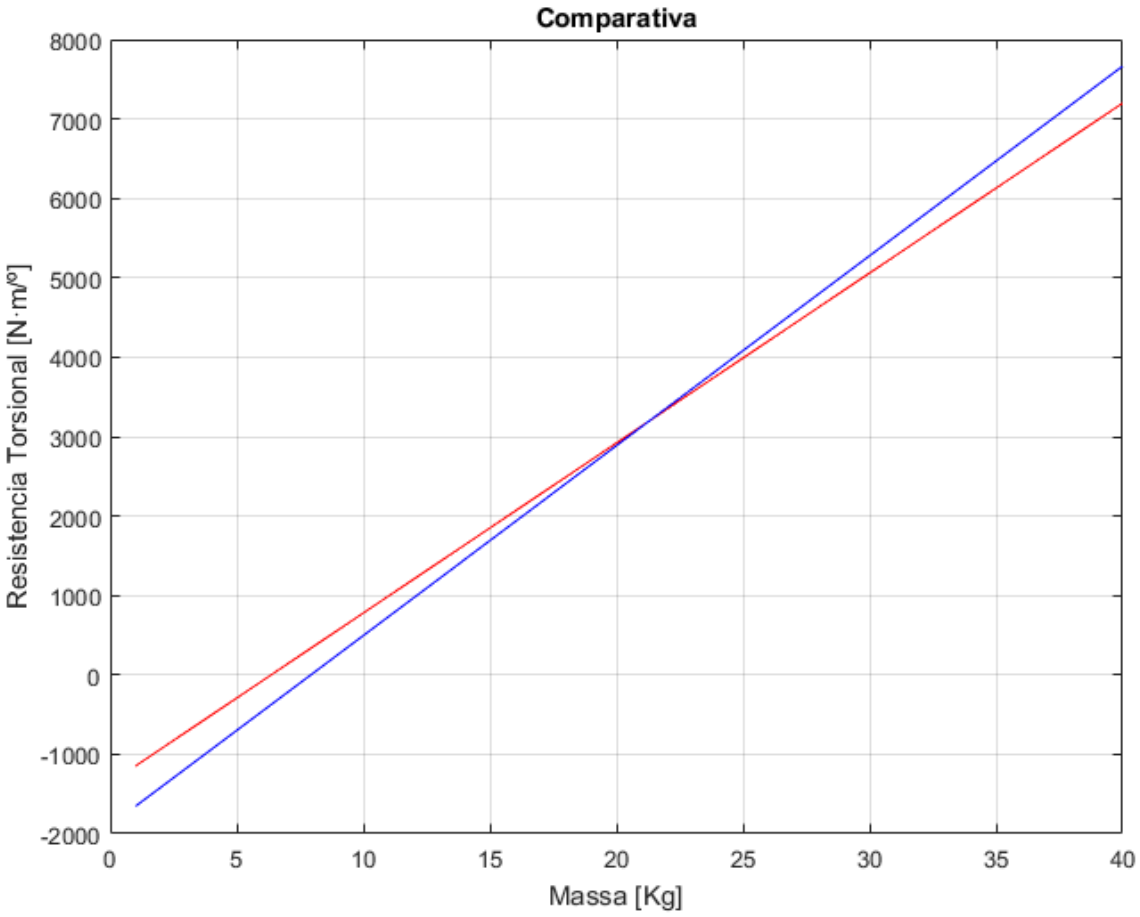


Figura 8.7 *comparativa de estructuras*

En cuanto a deformaciones reales, los resultados obtenidos para unas fuerzas de aceleración dadas por el sistema de vehículo- neumático, son despreciables internamente hablando, las deformaciones máximas y tensiones máximas están en el orden de micrómetros y decenas de Pascales. Por lo que para este tipo de vehículos la solicitud resistente de la estructura es menospreciable.
Otro tema es la deformación real que tendrían los trapecios y elementos elásticos del conjunto de suspensión, ya que estos al recibir una fuerza deformarían y cambiarían la cinemática del conjunto. Algo que es imperativo de estudiar.

CAPÍTULO 9: CONCLUSIONES

La finalidad de este proyecto es la de proporcionar una potente herramienta al lector para que con sus propios medios sea capaz de conceptuar, simular, optimizar una estructura reticulada compleja, como puede ser la de un monoplaza de competición para Formula Student.

Se ha creado con éxito un programa que, con pocos datos de entrada sea capaz de solventar:

1. Representar gráficamente la geometría de la estructura en 3D fielmente así como sus apoyos.

2. Calcular de manera casi inmediata el resultado analítico de las deformaciones y tensiones pertinentes

3. Reportar un valor de resistencia torsional y modo de vibración para satisfacer una solución y poder comparar con otras estructuras

4. Poder posicionar cargas remotas en distintos lugares para poder estudiar los efectos de estas en la estructura dado un vector de aceleración arbitraria.

5. Optimizar la estructura, cumpliendo las restricciones básicas de construcción y diseño.

Además, se ha explicado y referenciado los conceptos primordiales a la hora de escoger valores de referencia para poder inicializar un correcto diseño del monoplaza, a través de la resistencia torsional del mismo.

Más allá de todo esto, se ha estudiado el comportamiento del algoritmo optimizador, este código ha tomado una gran importancia en este proyecto ya que en el capítulo de estudio de resultados se puede observar como el lector puede interpretar de manera muy sencilla el comportamiento matemático de la estructura a estudiar. Esto último abre un sinfín de posibilidades para la mejora de la optimización estructural, no solo en el ámbito de las estructuras de un vehículo de carreras sino también para cualquier otro tipo de estructuras, por ello el autor decide finalizar el trabajo.

El final de este trabajo se podría describir como un final abierto, ya que un lector hábil podrá seguir su propio camino desde el punto de finalización de este proyecto como punto de partida.

El lector podrá inicializarse y adquirir un conocimiento amplio de la ciencia que hay detrás del cálculo de este tipo de estructuras, además de brindarle la escritura del código completo con ejemplos para su mejor entendimiento.

CAPÍTULO 10:

TRABAJOS POSTERIORES

Al tratarse un proyecto introductorio, el autor no puede plasmar todos los posibles caminos que se bifurcan a partir de la finalización de este, por ello ve necesario dar una guía, un consejo de por dónde deberían focalizarse los proyectos posteriores a este y siguiendo la línea de actuación.

Como anteriormente se ha comentado, la principal misión de este proyecto es la de unificar todas las modelizaciones matemáticas de un vehículo de carreras, en este apartado se ha tratado de la modelización de la estructura principal. Por ello y para poder optimizar de una mejor manera el autor recomienda:

1. La creación de un programa para cuantificar el ciclo de vida de la estructura, mediante las tensiones calculadas de cada barra y los vectores alternantes de las fuerzas de aceleración.

2. Introducir un mecanismo de creación de cartelas capaz de reforzar los nodos que sufran mayor deformación, esto permitirá a una dada estructura mejorar la pendiente de optimización de la misma.

3. La visualización y cálculo de puntos de inflexión para tener más información de donde colocar nuevos nodos que deban aguantar cargas remotas, esto permitirá poder colocar las sujeciones de los componentes de una manera lógica y estudiada

4. Un código capaz de calcular con pocas iteraciones el comportamiento filan de la estructura en verso al algoritmo optimizador, esto permitirá rebajar los tiempos de cálculo y por ende encontrar la geometría mas aceptada

5. Implementar un mecanismo que sea capaz de posicionar topológicamente los nodos de menor importancia espacial para poder conseguir distintas geometrías y así poder evaluar cuál de ellas es la mas optima

Tales trabajos han estado pensados por el autor como un claro camino a seguir para la optimización de la estructura.

Cabe decir que el lector es libre de tomar estos consejos de la manera que plazcan, pero el autor habla desde el conocimiento y la experiencia obtenida en todos estos años en el sector de la automoción y de la competición.

CAPÍTULO 10: BIBLIOGRAFÍA

1.1. Referencias bibliográficas

[1] Society of Automotive Engineers. *2016 Formula SAE Rules.* [en línea]. 2016 [Consulta: 20/01/2016]. Disponible en:

<http://www.fsaeonline.com/content/2016_FSAE_Rules.pdf>

[2] B. Riley, William; R. George, Albert. *Design, Analysis and Testing of a Formula SAE Car Chassis.* [en línea]. 2002 [Consulta 22/04/2016].

[3] Sánchez Molina, David Y González Drigo, Ramón González Drigo. *Cálculo de elementos estructurales.* UPC BARCELONATECH.2011

[4] Milliken, William F. Y Milliken, Douglas L. 1995. Race car vehicle dynamics.

[5] Segers, Jorge. 2008. Analysis Technics for Racecar Data Acquisition.

[6] Blanco Díaz, Elena. Cervera Ruiz, Miguel. Súarez Arroyo, Benjamín. Análisis Matricial de Estructuras

1.2. Bibliografía complementaria

[7] Apuntes de Metrología y Calidad de la asignatura *Procesos de fabricación*.

[8] Apuntes de SolidWorks de la asignatura *Simulación de máquinas y procesos*.

[9] Apuntes de la asignatura *Elasticidad y Resistencia de Materiales.*

[10] *ED-Tridim,* programa cedido para la verificación por el departamento de mecánica y resistencia de materiales de la universidad EUETIB UPC

[11] Apuntes de la asignatura *Maquinas y Mecanismos II.*