UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

TESIS

"METODOLOGÍA PARA LA DEFINICIÓN DE PARÁMETROS CONCENTRADOS Y SU USO EN EL DISEÑO DE CHASISES TIPO ESCALERA"

PRESENTA

ARMANDO MANUEL MENDOZA CANTÚ

EN OPCIÓN PARA OBTENER EL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA AUTOMOTRIZ

NOVIEMBRE 2015

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

TESIS

“METODOLOGÍA PARA LA DEFINICIÓN DE PARÁMETROS CONCENTRADOS Y SU USO EN EL DISEÑO DE CHASISES TIPO ESCALERA”

PRESENTA

ING. ARMANDO MANUEL MENDOZA CANTÚ

EN OPCIÓN PARA OBTENER EL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA AUTOMOTRIZ

NOVIEMBRE 2015

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO

TESIS

“METODOLOGÍA PARA LA DEFINICIÓN DE PARÁMETROS CONCENTRADOS Y SU USO EN EL DISEÑO DE CHASISES TIPO ESCALERA”

PRESENTA

ING. ARMANDO MANUEL MENDOZA CANTÚ

EN OPCIÓN PARA OBTENER EL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA AUTOMOTRIZ

NOVIEMBRE 2015

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA
DIVISIÓN DE ESTUDIOS DE POSGRADO

Los miembros del Comité de Tesis recomendamos que la tesis **"METODOLOGÍA PARA LA DEFINICIÓN DE PARÁMETROS CONCENTRADOS Y SU USO EN EL DISEÑO DE CHASISES TIPO ESCALERA"** realizada por el alumno Armando Manuel Mendoza Cantú con número de matrícula **1651388** sea aceptada para su defensa como opción al grado de Maestro en Ciencias de la Ingeniería Automotriz.

Dr. Luis Arturo Reyes

Asesor

Dr. Jacobo Hernández Sandoval

Co-asesor

MC. Ricardo Héctor Martínez Ríos

Asesor externo

El comité de tesis

Vo. Bo.

Dr. Simón Martínez Martínez

Subdirector de estudios de posgrado

Noviembre de 2015

AGRADECIMIENTOS

A Metalsa S.A. de C.V. por brindarme el apoyo necesario para realizar tanto las materias como la tesis.

A Ricardo Héctor Martínez Ríos por darme su apoyo como coordinador, asesor y amigo.

A Ricardo Prado Gámez por brindar su soporte técnico cuando fue necesario.

A Edgar Alberto Fernández Montoya por apoyarme al inicio de los estudios de maestría.

A Gustavo González Buentello y a Héctor Saúl Gonzalez Reyes por el apoyo en la parte de CAE del proyecto.

A mis compañeros de la generación por formar un grupo muy ameno.

A mi familia por apoyarme incondicionalmente.

A mis amigos por siempre estar presentes en todas las etapas de mi vida.

Al CLAUT por el apoyo que brindó para la existencia del programa.

DEDICATORIA

A mi familia por siempre estar presente como parte fundamental de mi existencia y de mi bienestar.

TABLA DE CONTENIDO

RESUMEN

Actualmente en la industria automotriz debido a las nuevas regulaciones que estipulan reducciones en el consumo de combustible a partir del 2020, se está trabajando en mejoras en los vehículos tanto en los motores como en la reducción de masa para poder cumplir con el objetivo que plantean las regulaciones.

Teniendo presente dicho objetivo, el presente trabajo tiene como finalidad el desarrollo de una metodología de diseño que permita a los diseñadores de Metalsa S.A. de C.V. contar con una base científica al momento de generar propuestas de reducción de masa para chasises de camionetas.

Dicha metodología parte de la optimización de masa para un chasis tipo escalera teniendo como restricciones las rigideces tanto vertical como torsional esperadas como las dimensiones que se requieren por parte de manufactura y finalmente las distancias requeridas por parte del vehículo.

Por otra parte se hizo uso del método de desplazamientos en la cual se establece la matriz de rigidez de la estructura llámese chasis y se reacomoda para ambos casos particulares, de esta manera se obtienen los desplazamientos en ciertos puntos específicos, para posteriormente obtener las rigideces tanto torsional como vertical del chasis con las geometrías propuestas.

Se limitó la metodología a los recursos con los que cuentan los diseñadores de la empresa por lo cual se hizo uso del paquete Excel, el cual pertenece al Microsoft Office y del paquete de programación Visual Basic ya que de igual manera se encuentra en los ordenadores de los diseñadores.

A su vez se desarrolló un estudio de mercado el cual muestra una forma alternativa de obtener los datos de entrada necesarios tomando en cuenta el mercado, y de esta manera posibilita la generación de propuestas sin necesidad de objetivos específicos compartidos o definidos por parte de los clientes.

Por otra parte se generó un exportador de los resultados al paquete de diseño asistido por computadora (CAD) CATIA, para facilitar la visualización por parte de los usuarios.

Se realizaron cinco iteraciones en las cuales se probó la metodología haciendo una comparativa de los resultados contra los resultados que arrojó por el método de elementos finitos a lo que la variación que presentó es aceptable.

INTRODUCCIÓN

Actualmente, la National Highway Traffic Administration (NHTSA) perteneciente al Departamento de Transportación de los Estados Unidos de América y encargado de la seguridad de la vialidad, liberó una regulación actualizada de la Corporate Average Fuel Economy (CAFE) la cual actualmente es considerada como una de las regulaciones más estrictas del ahorro de combustible. [1]

Esta actualización muestra cambios considerables desde el año 2020 e irá reduciendo paulatinamente el consumo de combustible hasta llegar al año 2025. Los vehículos son catalogados dependiendo de su "huella" la cual es calculada al multiplicar el ancho y la distancia entre los ejes. En la siguiente gráfica (Figura.1) se muestra la regulación futura a través de los años de las huellas donde se incluyen camionetas, ya que son las que principalmente cuentan con bastidor en su estructura.

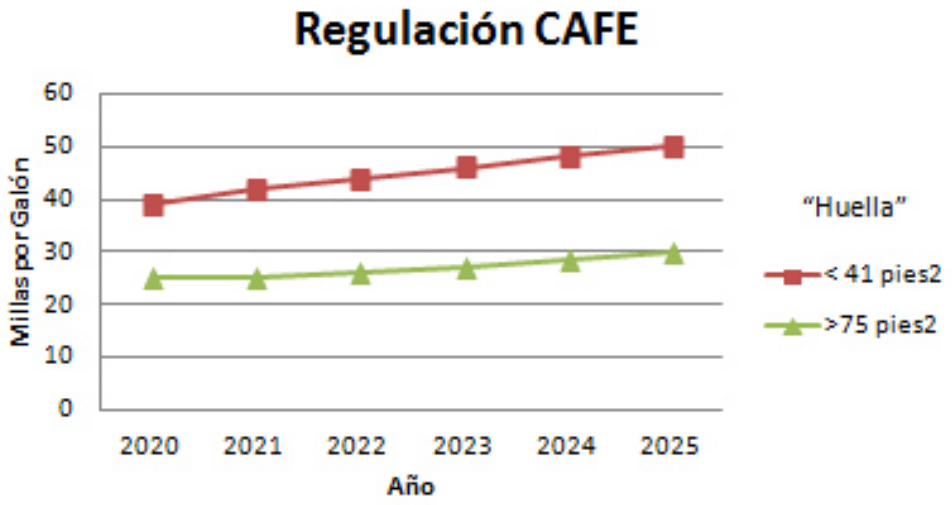


Figura 1. Gráfica de regulación de CAFE para los siguientes años [1]

Hoy en día la multa por el incumplimiento de la regulación del consumo de combustible es de $5.50 dólares americanos por cada 0.1 milla por galón que este fuera de lo estipulado, esto por cada unidad de vehículo doméstico de EEUU que no lo cumpla. De esta manera se puede explicar que si un fabricante del sector automotriz no logra una plataforma que cumpla con los requisitos de la CAFÉ se verá obligado a pagar esta sanción. [1]

Metalsa S.A. de C.V. es una compañía de origen Mexicano teniendo como base corporativa la ciudad de Monterrey Nuevo León, la cual tiene como una de sus numerosas labores el desarrollo y fabricación de estructuras vehiculares y tiene como misión actual proporcionar dichas estructuras a sus clientes para de esta manera lograr mover al mundo de una forma más eficiente, siendo responsable por el apoyo y cuidado del medio ambiente, así como ofreciendo una garantía a largo plazo de generar valor económico para los distintos grupos de interés con los cuales trabaja.

Asimismo, Metalsa cuenta con la visión de ofrecer soluciones innovadoras a sus clientes teniendo como principal enfoque las tecnologías de producto y proceso de manera segura y sustentable, logrando así una mejora en la calidad y servicio para los clientes manteniendo un costo competitivo.

Ya habiendo mencionado las actualizaciones sobre el consumo de combustible es importante agregar que Metalsa, siendo un proveedor del sector automotriz, busca brindar apoyo a sus clientes en dar las soluciones necesarias para que en conjunto logren cumplir dichas normas.

Por ende la finalidad de este proyecto es generar una herramienta que sea de apoyo para los diseñadores de Metalsa para reducir la cantidad de iteraciones entre el área de elementos finitos y el área de diseño, además de optimizar los recursos para poder dar un mejor soporte a los diferentes clientes.

CAPÍTULO 1: DESCRIPCIÓN DEL PROYECTO

1.1 JUSTIFICACIÓN DEL PROYECTO

Actualmente en el sector automotriz, como lo menciona Carbajal (2010); se ha buscado hacer un esfuerzo colectivo para la generación de valor agregado a lo largo de no solo las armadoras sino de toda la cadena. Esto implica que se tenga una mejor comunicación entre todas las partes involucradas además del impulso y desarrollo de proveedores haciendo uso de la subcontratación externa. [8]

A su vez, se comenta que los proveedores de autopartes han aumentado su roles en diseño ya que los automóviles se están diseñando como plataformas globales haciéndole algunos cambios para mercados locales. [32]

Por ende, Metalsa S.A. de C.V. siendo una compañía global que busca proveer y dar soporte integral a clientes del sector automotriz, se ha visto con la necesidad de involucrarse en los procesos de diseño y desarrollo de plataformas globales para diferentes clientes.

Aunado a esto, el área a tratar en este proyecto es la de diseño de Metalsa, la cual se encarga de modelar propuestas de chasises de camionetas con la ayuda de un software de diseño asistido por computadora (CAD). Otra área muy importante que se relaciona estrechamente con diseño es la de elementos finitos. En conjunto las dos áreas tienen como finalidad en primera instancia que el área de diseño, como su nombre lo dice, elabore una propuesta de diseño inicial basada en la experiencia del diseñador donde este proceso al ser finalizado sea validado por el área de elementos finitos, donde dicha área realiza una simulación con la cual se comprueba si el diseño debe ser modificado.

El área de oportunidad de la forma en que se trabaja actualmente en el área de diseño, es que depende directamente de la previa experiencia del diseñador, lo cual limita a los diseñadores menos experimentados a dar soporte en los proyectos.

Por lo tanto lo que se desea lograr con esta tesis es mejorar la precisión de la propuesta de diseño inicial por medio de la elaboración de una metodología que permita al diseñador contar con una base científica y no únicamente dejarlo en manos de su experiencia previa. De esta manera se busca que tanto los diseñadores con menos experiencia como los más experimentados puedan dar soporte a proyectos, y así tener un mejor aprovechamiento de los recursos.

1.2 Objetivo General del Proyecto

El objetivo general de este proyecto es el desarrollo de una metodología de diseño que permita a los diseñadores contar con una base científica al momento de generar propuestas para chasises de camionetas, y a su vez mejorar el aprovechamiento del recurso humano dentro del área.

1.3 Objetivos Específicos

1. Desarrollar un algoritmo para definir las geometrías óptimas que puedan ser utilizadas posteriormente para la base de diseño de un chasis.
2. Desarrollar un estudio de mercado para definir entradas para el algoritmo.
3. Generar una geometría de chasis utilizando la metodología establecida.
4. Comprobar los resultados por medio del método de elementos finitos.

1.4 Hipótesis

Un estudio de las principales variables durante el desarrollo de un chasis logrará reducir la cantidad de iteraciones entre las etapas de diseño y análisis numérico.

CAPÍTULO 2: DEFINICIONES ESTRUCTURALES

2.1 VÍNCULOS

Así como lo menciona Aguilar, R (2004) cuando se habla de vínculo se refiere a toda condición geométrica que tiene como finalidad restringir o limitar el movimiento de un cuerpo, estos pueden ser internos o externos dependiendo de la ubicación de los mismos. Se les llama internos a los cuales vinculan los cuerpos entre sí, en cambio se les llama externos a los que son vinculados de un cuerpo a la tierra.[2]

Se les clasifica por medio de clases dependiendo al tipo de limitación a la movilidad del cuerpo al cual se encuentran unidos (Figura.2):

- Primera clase: articulación móvil o rodillo, permite tanto la rotación como el desplazamiento al que está unido.
- Segunda clase: articulación fija, permite solamente la rotación del cuerpo al que se encuentra unido alrededor del punto de unión y también existe

el empotramiento móvil el cual permite el desplazamiento lineal del punto donde se encuentra unido en la dirección de su movimiento.

- Tercera clase: empotramiento fijo, no permite ningún tipo de movimiento en la unión al cuerpo.

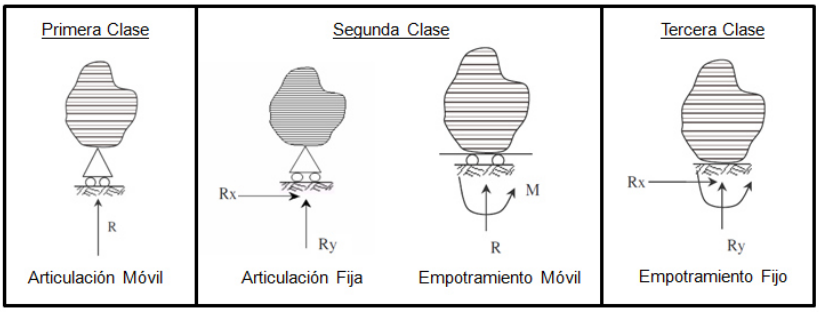


Figura 2. Clases de vínculos. [2]

2.2 ELEMENTOS LINEALES

Asimismo, Aguilar (2004) explica que un elemento lineal es formado por medio de un área plana, la cual cuenta con la característica que su centro de gravedad describe una curva que es mejor conocida como directriz o eje, la cual mantiene su plano perpendicular a la curva, el área móvil puede llegar a cambiar tanto en magnitud como en forma siempre y cuando se realice de forma continua (Figura 3). [2]

Otra característica es que las dimensiones del área transversal deben ser de menor magnitud a comparación de la longitud de la directriz, por lo cual es común que se represente a un elemento por la forma de su eje.

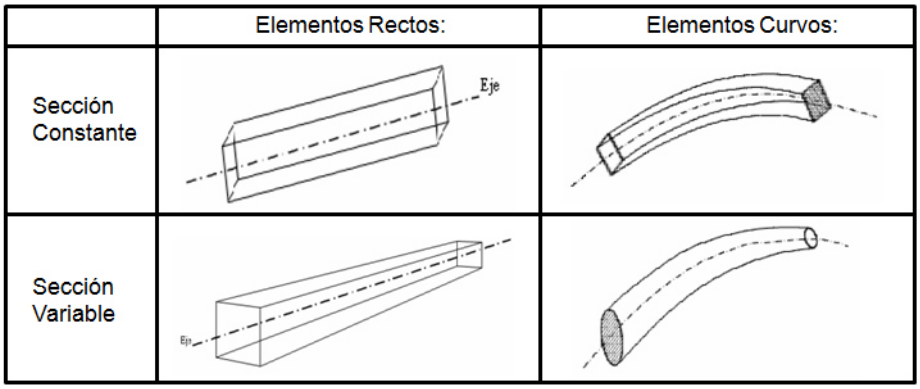


Figura 3. Tipos de elementos lineales. [2]

2.3 JUNTAS O NUDOS

Por otro lado a los puntos de unión de dos o más elementos se le conoce como juntas o nudos, normalmente se representa por medio de un punto, el cual identifica la intersección de los elementos que son interconectados en dicho nudo. Cabe destacar que a pesar de ser representado por un punto las juntas son uniones de diferentes elementos físicos que cuentan con dimensiones como bien es representado por la figura 4. [2]

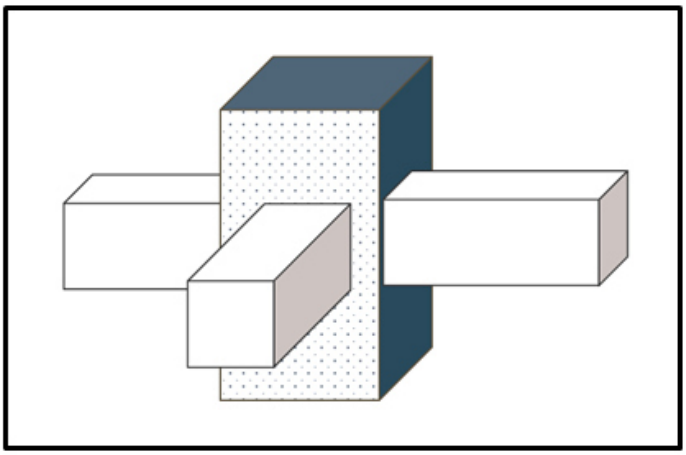


Figura 4 Representación de junta o nudo típico. [2]

2.4 ESTRUCTURAS

Se le denomina estructura a una cadena elástica estable (figura 5), la cual está formada por medio de elementos que están unidos mediante un número determinado de juntas o nudos; al mencionar que es elástico su comportamiento es lineal. [2]

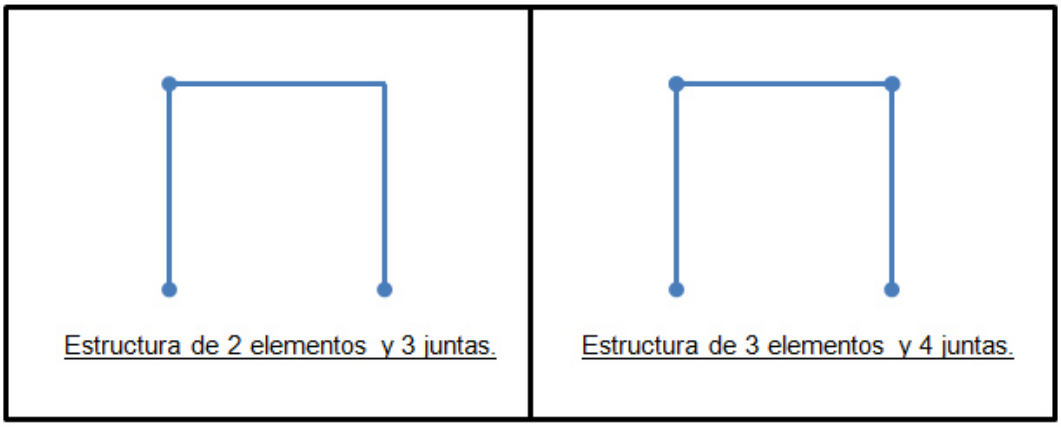


Figura 5. Ejemplos de representaciones de estructuras.

CAPÍTULO 3: DEFINICIONES SOBRE MECÁNICA

3.1 LEY DE HOOKE

Se le conoce como ley de Hooke a la relación lineal entre esfuerzo y deformación (figura 6) de una barra en tensión o compresión simple y es expresada como la siguiente ecuación 1:

$$\boldsymbol{\sigma} = E\varepsilon \qquad \text{(Ecuación 1)}$$

En donde σ representa al esfuerzo, ε es la deformación axial y E es una constante de proporcionalidad conocida como módulo de elasticidad del material. Se le conoce como ley de Hooke en honor al científico Robert Hooke (1635–1703), quien fue el primer hombre en investigar las propiedades elásticas de los materiales. [12]

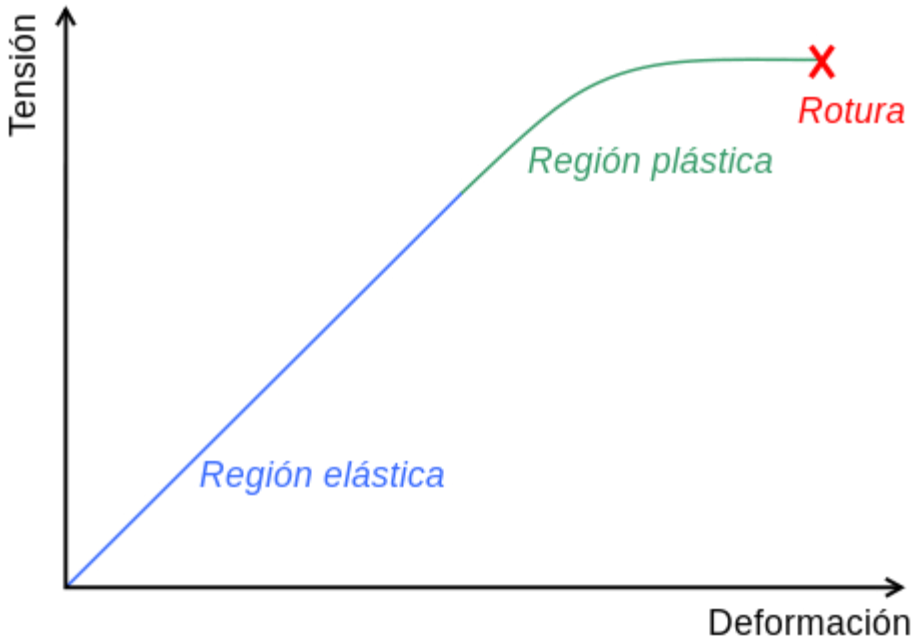


Figura 6. Ejemplo de gráfica de esfuerzo-deformación. [36]

3.2 Módulo de Elasticidad

El módulo de elasticidad (E) también es conocido como el módulo de Young, esto debido a que el científico Thomas Young (1773–1829), estuvo involucrado en la investigación de este efecto; el cual es la pendiente de la curva del diagrama esfuerzo-deformación en la parte elástica (figura 7) ya que es la parte lineal, tiene las mismas unidades que el esfuerzo (MPa, psi, etc.). [12]

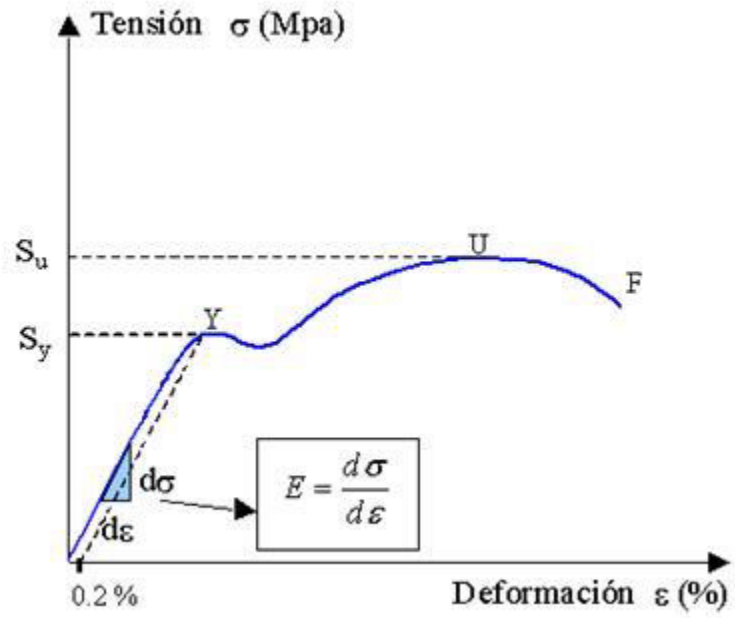


Figura 7. Diagrama esfuerzo-deformación haciendo énfasis en módulo de elasticidad. [5]

3.3 Razón de Poisson

Cuando una barra es cargada a tensión, al presentar deformación axial se contrae de forma lateral de forma normal a la carga aplicada, esto se puede observar en la figura 8 donde en (a) se muestra la barra antes de que se le aplique la carga y (b) después de que se le aplica la carga. [12]

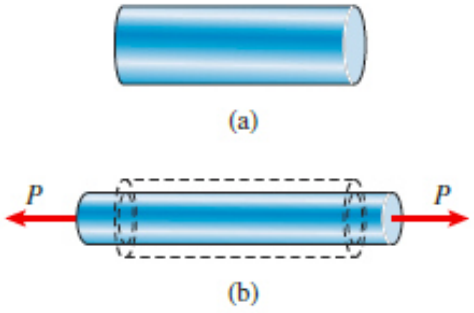


Figura 8. Deformación axial y contracción lateral de una barra prismática en tensión. [12]

La deformación lateral (ε') en cualquier punto de la barra es proporcional a la deformación axial (ε) en el mismo punto, si el material es linealmente elástico; la razón de estas tensiones se le conoce como razón de poisson ($\boldsymbol{\nu}$), así como se muestra en la ecuación 2. [12]

$$\boldsymbol{\nu} = \frac{Tensión\ Lateral}{Tensión\ Axial} = \frac{\varepsilon'}{\varepsilon} \qquad \text{(Ecuación 2)}$$

3.4 Coordenadas Generalizadas

Para la determinación de la configuración de una estructura se utilizan coordenadas, dichas coordenadas pueden ser dependientes o independientes (figura 9). Por otra parte se les conocen como coordenadas generalizadas a las que son independientes. [2]

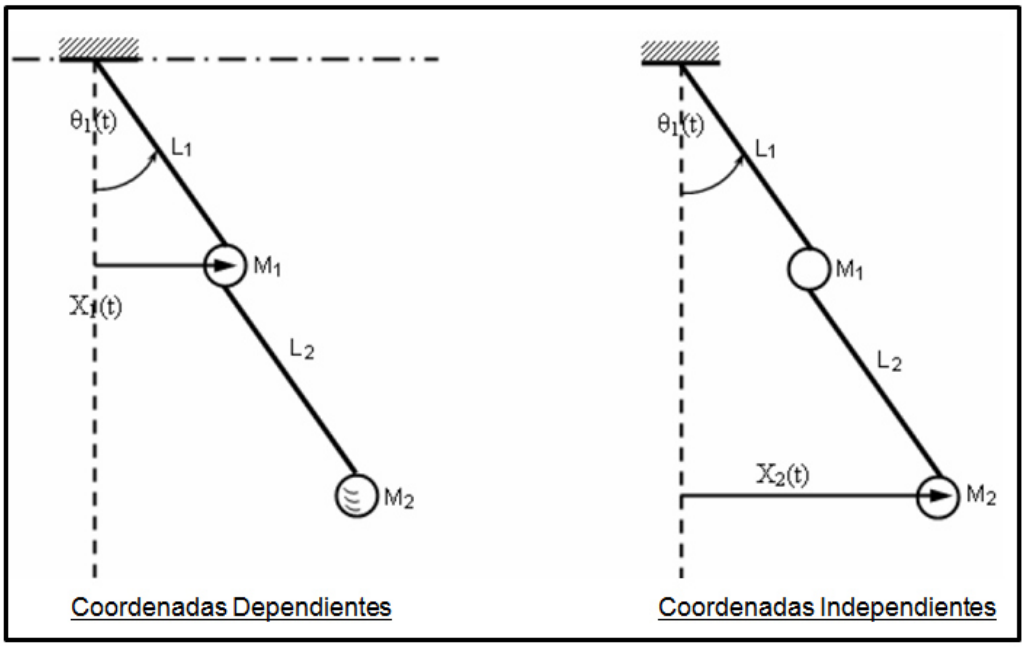


Figura 9. Coordenadas independientes y dependientes. [2]

3.5 GRADOS DE LIBERTAD

Se definen los grados de libertad en el caso de estructuras como “los desplazamientos de los nudos que son los necesarios para describir todas las posibles configuraciones deformadas de la estructura”. [24]

Asimismo, como lo menciona París, F (2006) cada uno de los grados de libertad está asociado a una coordenada generalizada por lo que en el conjunto de dichas coordenadas se puede expresar cualquier configuración deformada de la estructura (figura 10). [24]

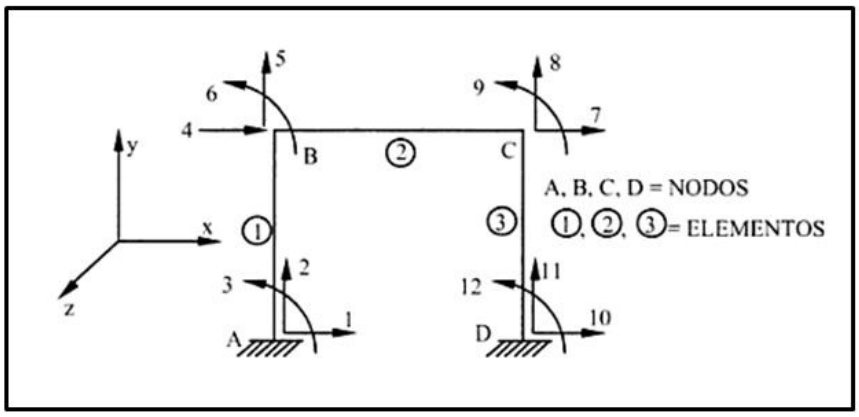


Figura 10. Nudos, elementos, grados de libertad y coordenadas generalizadas de una estructura. [24]

3.6 CONCEPTO DE RIGIDEZ

Es común decir que una estructura es rígida cuando se le aplica una carga y los desplazamientos que presenta son pequeños y por el contrario se dice que es flexible cuando presenta el caso contrario. [24]

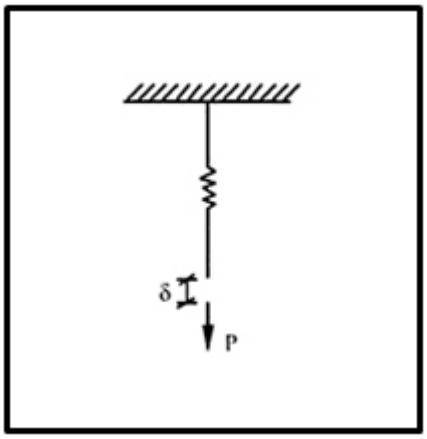


Figura 11. Rigidez de un muelle. [24]

Así como lo explica París, F (2006) con un sencillo ejemplo de un muelle (figura 11) que cuenta con un solo grado de libertad que sometido a una fuerza P que provoca un alargamiento δ que forman la siguiente ecuación (ecuación 3) [24]:

$$P = k\,\delta \qquad \text{(Ecuación 3)}$$

Donde k representa la constante de rigidez del muelle y a su vez representa la fuerza que se aplica para obtener un desplazamiento de una unidad. [24]

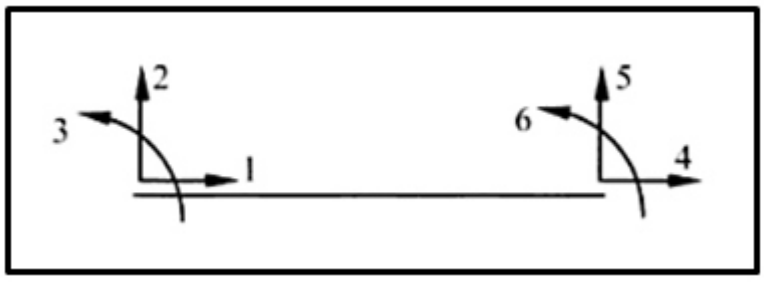


Figura 12. Grados de libertad de un elemento plano. [24]

Para el caso de un elemento plano (figura 12) los grados de libertad que presenta son 6 y el sistema de fuerzas es representado por el vector fuerzas (F) que cuenta con 6 componentes, para poder definir la configuración sobre la deformación de la misma es necesario hacer uso de un vector de desplazamientos (u) que cuenta con 6 componentes, los cuales incluyen giros y desplazamientos asociados a los grados de libertad. Estos vectores serán relacionados a través de una matriz de rigidez (ecuación 4). [24]

$$F = Ku \qquad \text{(Ecuación 4)}$$

Para el caso de una estructura de n grados de libertad las matriz K será de dimensión n x n y cada elemento de dicha matriz relacionará las fuerzas y desplazamientos. [24]

3.7 MATRIZ DE RIGIDEZ

La matriz de rigidez tiene distintas funciones, una de ellas es establecer una relación entre sus desplazamientos y las fuerzas que están asociadas. Operativamente relaciona los desplazamientos incógnita de una estructura con las fuerzas que le son aplicadas a la misma, esto permite que sea posible encontrar las reacciones, los esfuerzos y las tensiones en cualquier punto de la estructura. [34]

3.7.1 MATRIZ DE RIGIDEZ PARA UNA BARRA EN 2 DIMENSIONES.

Para este caso se emplea la configuración bi-empotrada, con lo cual cuenta con uniones rígidas en sus dos nudos de los extremos, esta barra puede

encontrarse en cualquier posición y orientación dentro del sistema global de coordenadas (x, y), a su vez se define un sistema local de coordenadas $(\hat{x}, \hat{y})$ en donde $\hat{x}$ se encuentra alineado con la dirección de la barra $i \rightarrow j$ como se muestra en la figura 13. [6]

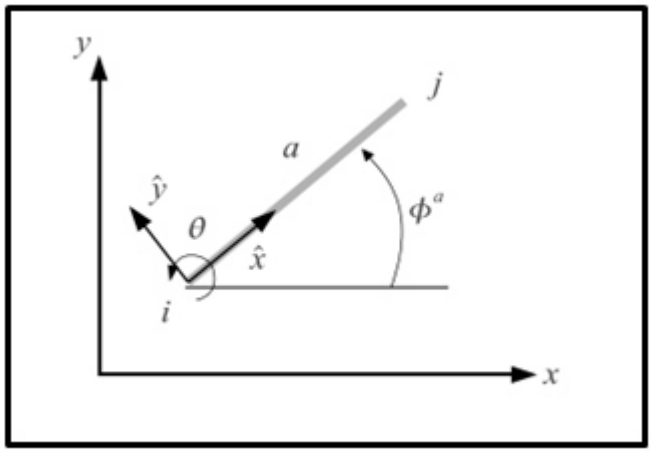


Figura 13. Sistema de coordenadas locales de una barra cuyos dos extremos están empotrados.[6]

Como lo menciona Blanco, (2012) haciendo uso del nuevo teorema de Maxwell-Betti, que para cualquier par de nudos i y j las submatrices $\widehat{K}_{ij}^{a}$ y $\widehat{K}_{ji}^{a}$ son traspuestas, por lo tanto $\widehat{K}_{ij}^{a} = \widehat{K}_{ji}^{a\,T}$. A continuación se muestran los desplazamientos unitarios para la barra (figura 14) [6]:

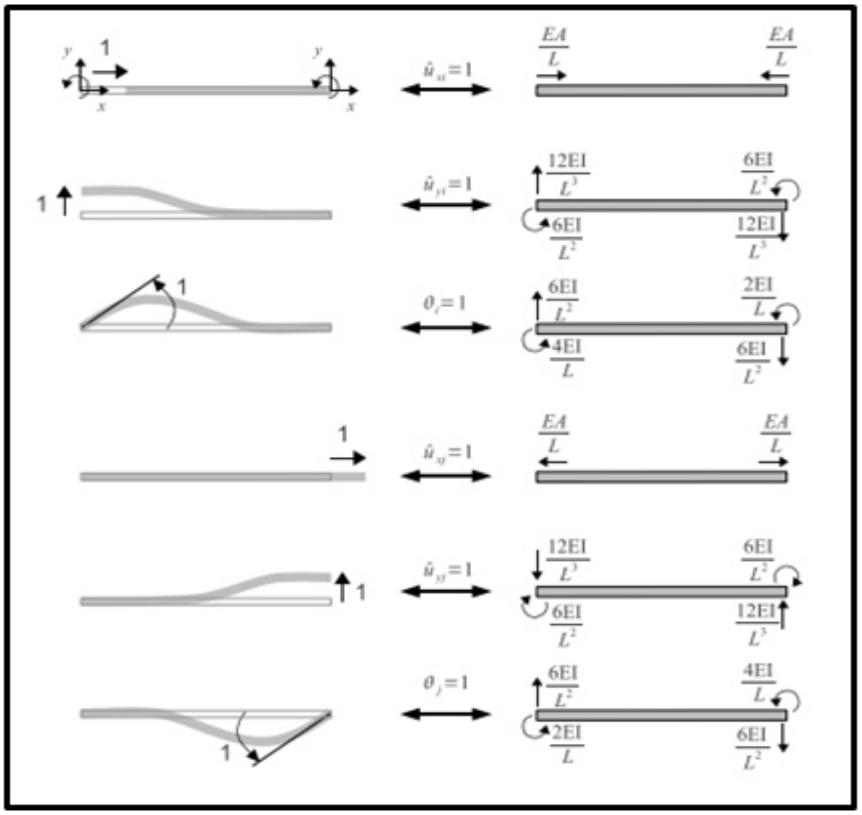


Figura 14. Desplazamientos unitarios para una barra con 6 grados de libertad. [6]

Por lo tanto la matriz de rigidez para una barra con 6 grados de libertad (2 dimensiones), es la que se muestra a continuación (ecuación 5):

$$\begin{bmatrix} \frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI}{L^3} & \frac{6EI}{L^2} & 0 & -\frac{12EI}{L^3} & \frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{4EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{2EI}{L} \\ -\frac{EA}{L} & 0 & 0 & \frac{EA}{L} & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & 0 & \frac{12EI}{L^3} & -\frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{2EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{4EI}{L} \end{bmatrix} \quad \text{(Ecuación 5)}$$

Donde E es el módulo de elasticidad, I representa al momento de inercia de la sección y L es la longitud de la barra. [18]

3.7.2 MATRIZ DE RIGIDEZ PARA UNA BARRA EN 3 DIMENSIONES.

Para obtener la matriz de rigidez de la barra en 3 dimensiones expresada en coordenadas locales se obtiene a partir de un desplazamiento unitario de cada uno de los movimientos posibles (figura 15), al restringir los otros 11 movimientos que son posibles. [18]

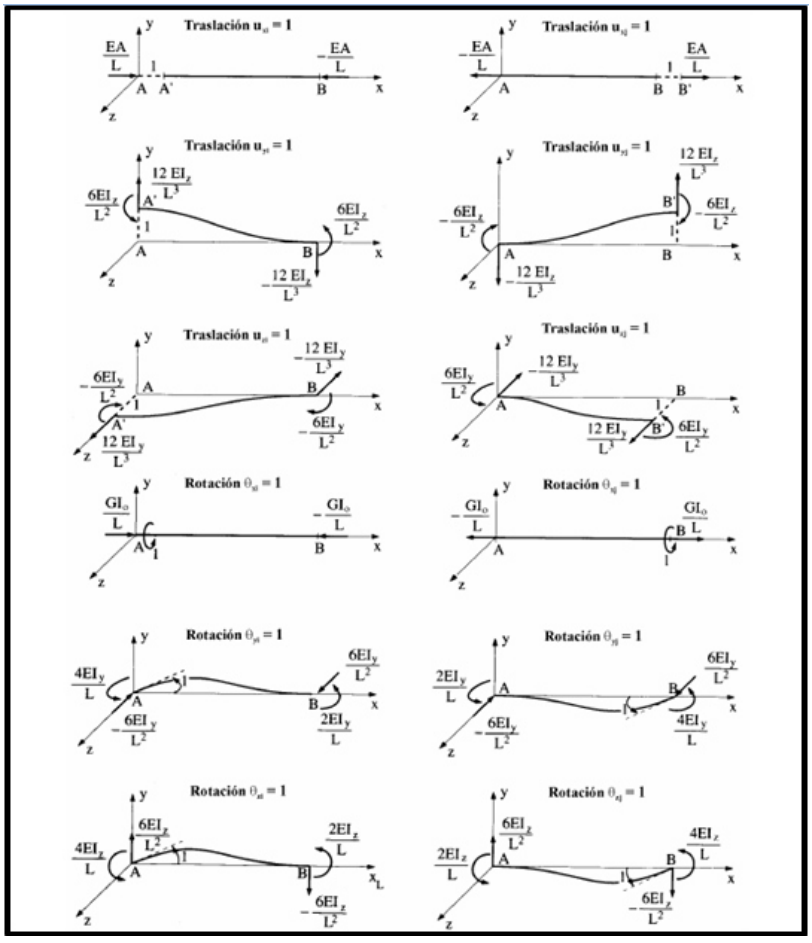


Figura 15. Movimientos unitarios en los extremos de una barra en 3 dimensiones. [18]

Al obtener los esfuerzos originados en cada uno de los extremos de la barra al imponer el movimiento, se logran obtener los coeficientes de rigidez de la matriz (ecuación 6) que es mostrada a continuación:

$$\begin{bmatrix}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\
0 & 0 & \frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 & 0 & 0 & -\frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GI_0}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GI_0}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\
-\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} & 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} \\
0 & 0 & -\frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 & 0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & -\frac{GI_0}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GI_0}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L}
\end{bmatrix}$$

(Ecuación 6)

Donde E es el módulo de elasticidad, I representa al momento de inercia de la sección, G es el módulo de elasticidad a cortadura, I_0 representa a la constante de rigidez a la torsión de la sección y L es la longitud de la barra. [18]

3.7.3 TRANSFORMACIÓN DE COORDENADAS

Es necesario con anticipación de la imposición de las ecuaciones de equilibrio y compatibilidad la transformación del sistema de coordenadas local al global, ya que esto permitirá que las ecuaciones de los distintos elementos puedan ser expresados en conjunto. [30]

Se comienza estudiando las relaciones que existe entre dos ejes cartesianos planos de coordenadas, esto al girar uno de ellos (figura 16):

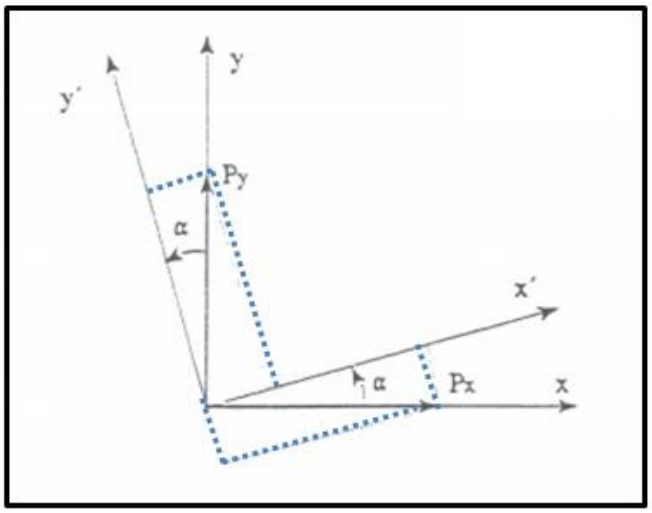


Figura 16. Transformación de coordenadas. [30]

Observando la imagen anterior se pueden deducir 2 ecuaciones (ecuación 7 y 8):

$$P'x = Px\ \cos\alpha + Py\ \sin\alpha \qquad \text{(Ecuación 7)}$$

$$P'y = -Px\ \sin\alpha + Py\ \cos\alpha \qquad \text{(Ecuación 8)}$$

O bien se pueden ver de forma matricial (ecuación 9):

$$\begin{Bmatrix} P'x \\ P'y \end{Bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{Bmatrix} Px \\ Py \end{Bmatrix} \qquad \text{(Ecuación 9)}$$

Si se le llama T (ecuación 10) a:

$$T = \begin{bmatrix} lx & ly \\ mx & my \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \qquad \text{(Ecuación 10)}$$

En donde (lx, mx) representan a los cosenos directores de x' respecto a (x, y), a su vez (ly, my) representan los cosenos directores de y'. Al ser ortogonal la matriz T, se cumple que su inversa es igual que su transpuesta (ecuación 11). [30]

$$T^T = T^{-1} \qquad \text{(Ecuación 11)}$$

Para el caso de una matriz de cambio de coordenadas de sistema local a global, para una barra en 3 dimensiones tiene la siguiente forma (ecuación 12):

$$[T] = \begin{bmatrix} l_x & l_y & l_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ m_x & m_y & m_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ n_x & n_y & n_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & l_x & l_y & l_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & m_x & m_y & m_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_x & n_y & n_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & l_x & l_y & l_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & m_x & m_y & m_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & n_x & n_y & n_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & l_x & l_y & l_z \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_x & m_y & m_z \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n_x & n_y & n_z \end{bmatrix}$$

(Ecuación 12)

En donde (l_x, m_x, n_x) representan a los cosenos directores de x local respecto a (X, Y, Z) globales, así como (l_y, m_y, n_y) representan los cosenos directores de y local respecto a (X, Y, Z) globales y finalmente (l_z, m_z, n_z) representan los cosenos directores de z local respecto a (X, Y, Z) globales. [18]

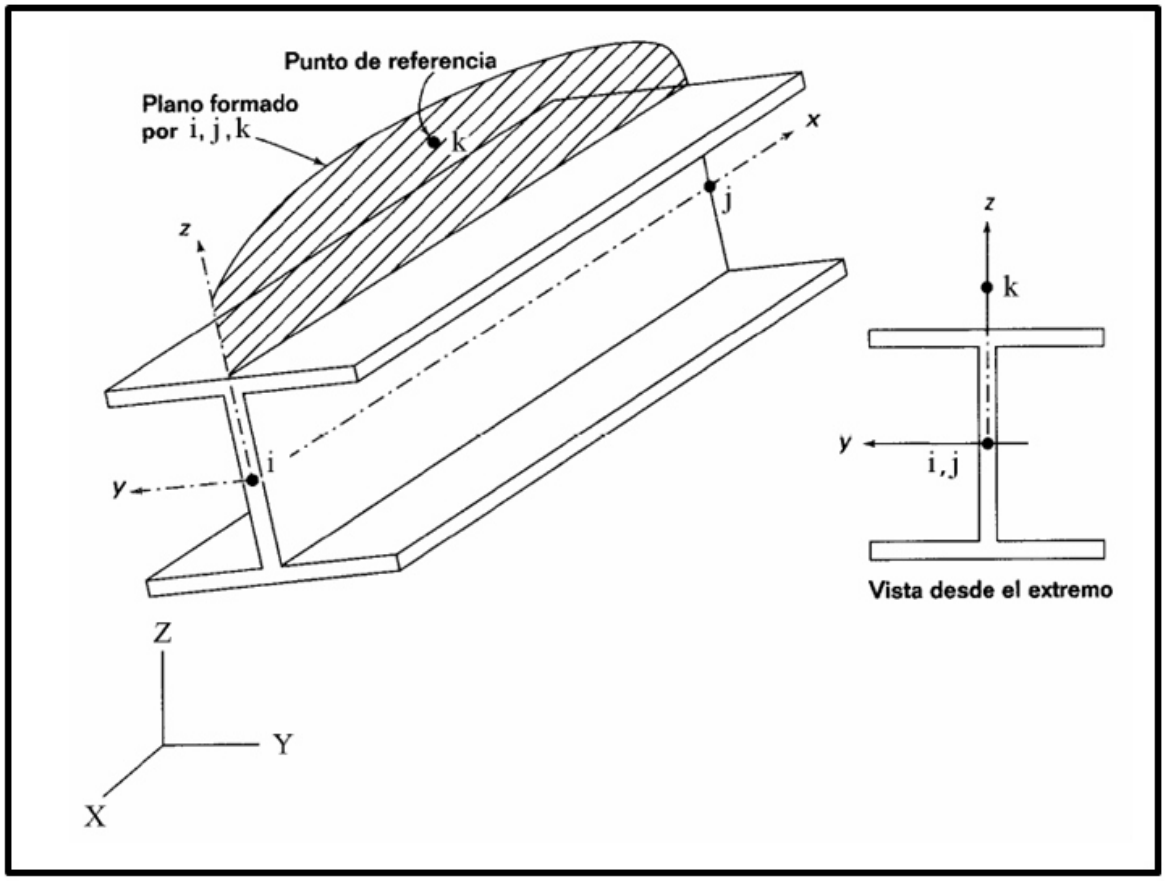


Figura 17. Sistema de referencia local y global para una barra en 3 dimensiones. [18]

Para obtener los cosenos directores se hacen uso las coordenadas de los puntos i, j, k (figura 17). Y se utilizan las siguientes formulas.

Para el eje x se utiliza la orientación de i a j, y el vector V_x (ecuación 13) representa los cosenos directores:

$$V_x = \begin{bmatrix} l_x \\ l_y \\ l_z \end{bmatrix} = \begin{bmatrix} \frac{X_j - X_i}{L} \\ \frac{Y_j - Y_i}{L} \\ \frac{Z_j - Z_i}{L} \end{bmatrix} \qquad \text{(Ecuación 13)}$$

Donde la L representa la longitud resultante de la barra y se obtiene con la siguiente fórmula (ecuación 14):

$$L = \sqrt{\left(X_j - X_i\right)^2 + \left(Y_j - Y_i\right)^2 + \left(Z_j - Z_i\right)^2} \qquad \text{(Ecuación 14)}$$

Asimismo, se debe obtener el valor de L_{ik} (ecuación 15) ya que será utilizado para obtener el vector V_{ik}:

$$L_{ik} = \sqrt{(X_k - X_i)^2 + (Y_k - Y_i)^2 + (Z_k - Z_i)^2} \qquad \text{(Ecuación 15)}$$

Por otra parte se utiliza el vector V_{ik} (ecuación 16) como apoyo para determinar el vector V_y (ecuación 17) y se obtienen de la siguiente manera:

$$V_{ik} = \begin{bmatrix} l_{ik} \\ l_{ik} \\ l_{ik} \end{bmatrix} = \begin{bmatrix} \frac{X_k - X_i}{L_{ik}} \\ \frac{Y_k - Y_i}{L_{ik}} \\ \frac{Z_k - Z_i}{L_{ik}} \end{bmatrix} \qquad \text{(Ecuación 16)}$$

$$V_y = \begin{bmatrix} l_y \\ l_y \\ l_y \end{bmatrix} = \frac{V_x \; X \; V_{ik}}{|V_x \; X \; V_{ik}|} \qquad \text{(Ecuación 17)}$$

Y por último la fórmula para obtener los cosenos directores para el eje z (ecuación 18):

$$V_z = \begin{bmatrix} l_z \\ l_z \\ l_z \end{bmatrix} = V_x \; X \; V_y \qquad \text{(Ecuación 18)}$$

Haciendo uso de la matriz de transformación pueden expresar las fuerzas y los desplazamientos de la barra de la siguiente manera:

$$[U]^{Globales} = [T][u]^{lcoales} \qquad \text{(Ecuación 19)}$$

$$[P]^{Globales} = [T][p]^{lcoales} \qquad \text{(Ecuación 20)}$$

En donde $[u]^{lcoales}$ son los desplazamientos en coordenadas locales mientras que $[U]^{Globales}$ (ecuación 19) son los desplazamientos en coordenadas globales, de igual manera que con las fuerzas $[p]^{lcoales}$ y $[P]^{Globales}$ (ecuación 20). [18]

3.7.4 MATRIZ DE RIGIDEZ DE UN ELEMENTO EN COORDENADAS GLOBALES

Partiendo de la resolución de desplazamientos nodales se plantea la siguiente ecuación (ecuación 21):

$$[p]^{locales} = [K]^{locales}[u]^{locales} \qquad \text{(Ecuación 21)}$$

Y sustituyendo las ecuaciones 19 y 20 se tiene que (ecuación 22):

$$[T]^{-1}[P]^{Globales} = [K]^{locales}[T]^{-1}[U]^{Globales} \qquad \text{(Ecuación 22)}$$

Si se multiplican ambos lados por la matriz de transformación $[T]$ se obtiene (ecuación 23):

$$[P]^{Globales} = [T][K]^{locales}[T]^{-1}[U]^{Globales} \qquad \text{(Ecuación 23)}$$

Usando la igualdad de la ecuación 11 la matriz de rigidez en coordenadas globales del elemento (ecuación 24) se puede expresar de la siguiente manera:

$$[K]^{Globales} = [T][K]^{locales}[T]^{T} \qquad \text{(Ecuación 24)}$$

3.7.5 MATRIZ DE RIGIDEZ DE UNA ESTRUCTURA

Al tener los las matrices de cada uno de los elementos que forman una estructura en coordenadas globales, es posible ensamblar las matrices para de esta manera lograr obtener la matriz de rigidez de la estructura.

Al tener la matriz de rigidez de un elemento en coordenadas globales se puede subdividir (ecuación 25) de la siguiente manera:

$$[K]^{Globales} = \begin{bmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{bmatrix}^{ij} \qquad \text{(Ecuación 25)}$$

Ya sabiendo la relación que tiene cada una de las partes de la matriz se puede hacer una sumatoria de sub-matrices (ecuación 26) como se muestra a continuación:

$$[K]_{Global}^{Estructura} = \sum_{ij} \begin{bmatrix} [0] & [0] & [0] & [0] \\ [0] & K_{ii} & K_{ij} & [0] \\ [0] & K_{ji} & K_{jj} & [0] \\ [0] & [0] & [0] & [0] \end{bmatrix}^{ij} \qquad \text{(Ecuación 26)}$$

CAPÍTULO 4: MÉTODOS DE SOLUCIÓN DE ESTRUCTURAS

Existen diferentes métodos para analizar una estructura, en el caso de esta tesis como se habló en el alcance solamente se verificarán análisis del tipo estático y nos interesaría saber la rigidez vertical y torsional del chasis; principalmente se busca hacer un análisis en reversa, es decir, establecer un método con el cual se pueda obtener las geometrías óptimas teniendo valores de rigideces.

4.1 MÉTODOS DE CÁLCULO MATRICIAL

Existen dos diferentes procedimientos en mecánica de medios continuos de sólidos deformables para tener la posibilidad de establecer las ecuaciones de comportamiento dependiendo del orden de aplicación así como se muestra en la figura 18. [6]

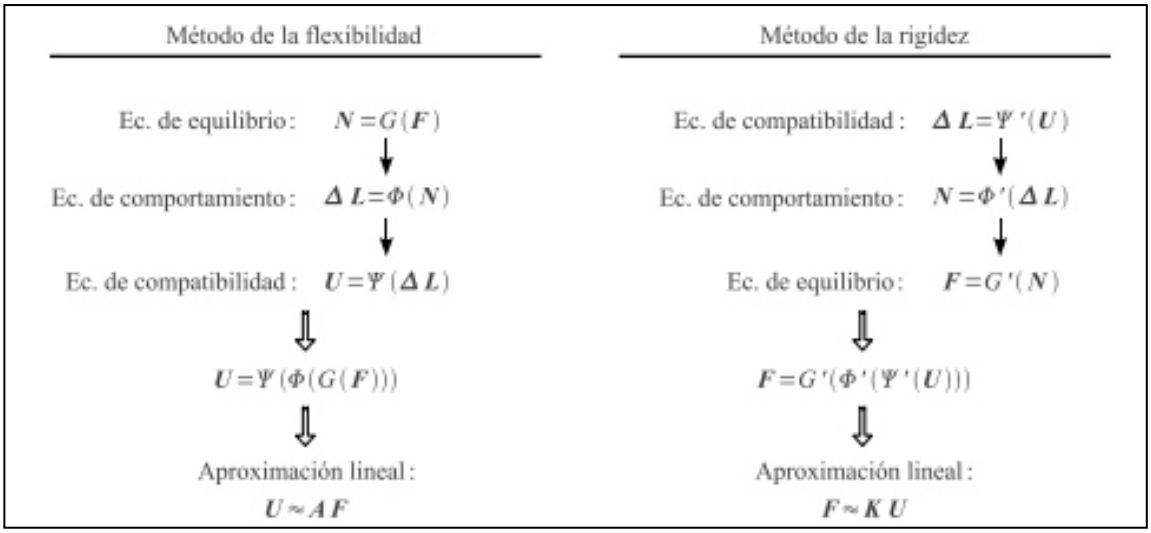


Figura 18. Métodos de cálculo matricial. [6]

Método de Flexibilidad o método de Compatibilidad: Cuando se inicia con las ecuaciones de equilibrio y son utilizadas al incorporarlas a las de comportamiento y al obtener el resultado lo incorporamos a las ecuaciones de compatibilidad. En este método se relacionan los desplazamientos en función de las cargas aplicadas. [6]

Método de Rigidez o método del Equilibrio: cuando se inicia relacionando las deformaciones y los desplazamientos al aplicar las ecuaciones de compatibilidad para incorporarlas a las de comportamiento y finalmente a las ecuaciones de equilibrio. [6]

Por otra parte Blanco (2012) afirma que de los dos métodos mencionados solamente el de la rigidez permite realizar un procedimiento automático y sistematizado por la misma razón es el método utilizado en este proyecto. [6]

CAPÍTULO 5: OPTIMIZACIÓN Y MÉTODO DE ELEMENTOS FINITOS

Por definición optimizar se refiere a la "búsqueda de la mejor manera de realizar una actividad" (RAE, 2001). Por lo definido es importante mencionar que existe una rama de las matemáticas que tiene como finalidad la selección del mejor elemento tomando en cuenta criterios específicos para una serie de variables. [26]

Generalmente los problemas de optimización se componen de 3 partes:

1. Función objetivo: Se le conoce como la medida cuantitativa del funcionamiento del sistema que se está optimizando, ya sea para maximizar o para minimizar. [28]
2. Variables: Representan los valores que se pueden modificar para afectar el valor de la función objetivo. [28]
3. Restricciones: son el conjunto de relaciones que ciertas variables están obligadas a satisfacer. [28]

Los métodos de optimización se pueden clasificar en:

- Métodos clásicos o iterativos: que son los métodos que usualmente se explican en los libros de optimización, algunos de ellos son la optimización lineal, lineal entera mixta, estocástica, no lineal, entre otros. [28]
- Métodos metaheurísticos: que son los que están ligados a la inteligencia artificial los cuales imitan fenómenos observados en la naturaleza, algunos de ellos son los algoritmos evolutivos, búsquedas heurísticas, sistemas multiagente, entre otros. [28]

5.1 Métodos Clásicos o Iterativos

A continuación se explicarán dos métodos clásicos o iterativos a manera de ejemplo para poder tener un mayor entendimiento sobre el tema de optimización.

5.1.1 Método Simplex

Así como le menciona López (1993), se refiere a un método algebraico iterativo que tiene la capacidad de resolver cualquier modelo de programación que sea lineal ya que examina paso a paso las posibles soluciones y se va aproximando sistemáticamente a la solución óptima, iniciando por un vértice y se mueve a través de las aristas del poliedro de soluciones hasta que alcanza el vértice óptimo de solución, así como se muestra en la figura 19. [17] [21]

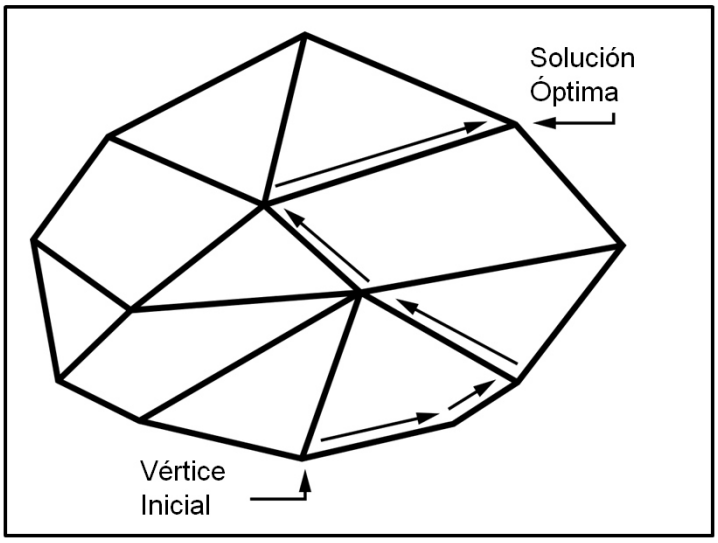


Figura 19. Algoritmo simplex de forma gráfica. [13]

5.1.2 MÉTODO NELDER-MEAD

El método de Nelder-Mead es uno de los más utilizados para la optimización no lineal sin restricciones, este método intenta minimizar una función no lineal escalar con valores variables de n reales utilizando únicamente valores de la función. [16]

5.2 MÉTODOS METAHEURÍSTICOS

Estos métodos son comúnmente utilizados para resolver problemas de optimización en los ámbitos de las ingenierías, ya que tienen una mayor dificultad que no logran ser resueltos por medio de las técnicas tradicionales. [29]

5.2.1 ALGORITMO GENÉTICO

Este método es basado en la evolución como una sucesión de cambios frecuentes en los genes (figura 20), trabajan a través de una población de cadenas binarias para hacer la representación del problema a resolver, y a su vez buscando las posibles soluciones a través de una exploración aplicando transformaciones a las soluciones previas tal y como se ve los organismos vivientes, a través de cruces, mutaciones e inversiones. [22]

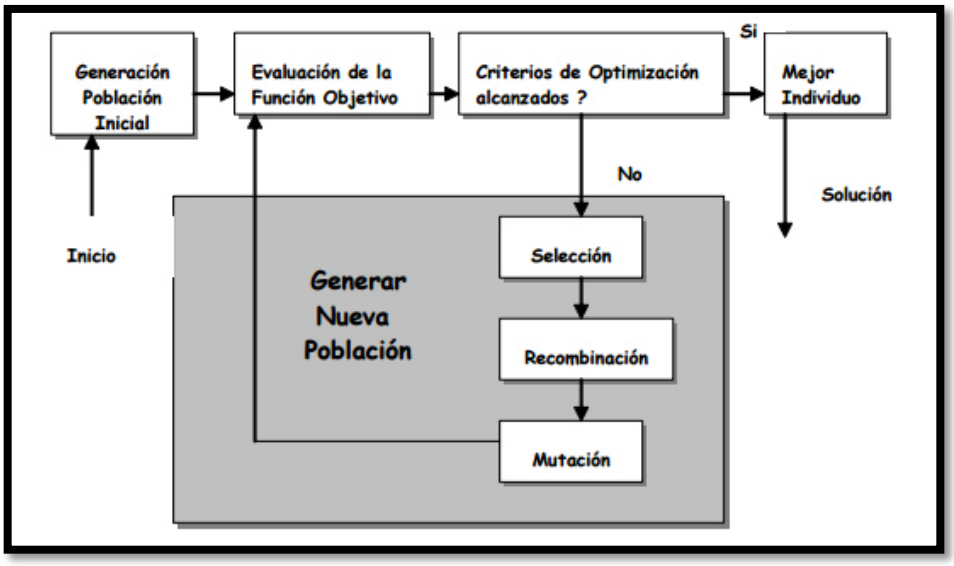


Figura 20. Diagrama de flujo del algoritmo genético. [3]

Por otra parte, tienden a ser muy flexibles ya que tienen la capacidad de adoptar nuevas ideas que surjan dentro de la computación evolutiva. [19]

5.3 MÉTODO DE ELEMENTOS FINITOS

El método de elementos finitos (FEM por sus siglas en inglés) ha adquirido una gran importancia en cuanto a la solución de problemas físicos e ingenieriles ya que ha permitido resolver casos que eran muy complicados para resolver por los métodos matemáticos tradicionales. [20]

Anteriormente era necesario fabricar prototipos, ensayarlos e ir iterando para mejorar los componentes lo cual ocasionaba un costo elevado y mayor cantidad de tiempo para el desarrollo de productos. El FEM permite realizar un modelo de cálculo matemático para un sistema real, el cual es más sencillo de modificar que un prototipo; a pesar de ello no deja de ser un método aproximado de cálculo, por lo que los prototipos siguen siendo necesarios pero en menor cantidad. [20]

5.3.1 HISTORIA DEL MÉTODO

En los años 40's Richard Courant desarrollo los elementos finitos tal y como se conocen en la actualidad como parte de sus cálculos estructurales en el campo aeroespacial, en los cuales propuso la utilización de funciones polinómicas para la solución de problemas elásticos, esto como una variación del método variacional de Rayleigh-Ritz para la aproximación de soluciones. [10]

Fueron M.J. Turner, R.W. Clough, H.C. Martin y L.J.Topp quienes presentaron la aplicación de elementos finitos simples (placas triangulares con cargas en su plano y barras) al análisis de estructuras aeronáuticas complejas utilizando conceptos de discretizado y funciones de forma. [33]

Posteriormente J.S. Przemieniecki para 1968 presenta el método de elementos finitos aplicándolo al análisis estructural, a su vez J.T. Oden presentó algunas de las contribuciones de mayor importancia en cuanto a matemáticas al método de elementos finitos en el año 1972. [23] [25]

Por otra parte O.C. Zienkiewicz junto con Y.K. Cheung fueron más allá e interpretaron el método de elementos finitos para una mayor cantidad de campos de aplicación, ya que demostraron que las ecuaciones pueden ser obtenidas utilizando un método de aproximación de pesos residuales; esto ayudo a que el método cobrara un mayor interés entre los matemáticos para la solución de ecuaciones diferenciales no lineales como las lineales mediante el método de elementos finitos y ha sido considerada como una de las herramientas más potentes de solución de problemas de ciencia aplicada tanto como de ingeniería. [37]

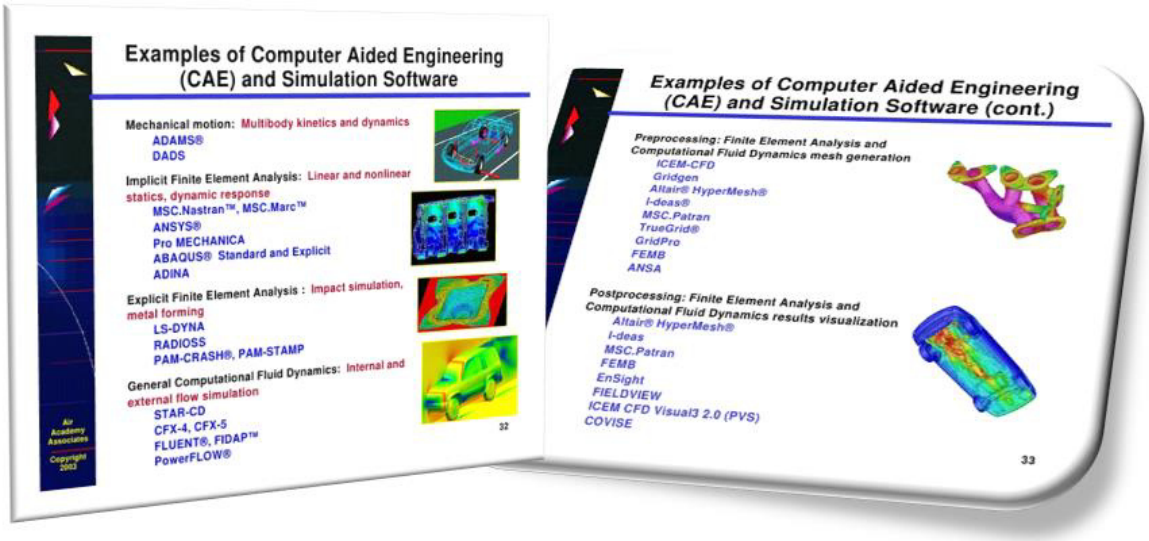


Figura 21. Ejemplos de software especializados que hacen uso del método de elementos finitos. [15]

Actualmente el método es ampliamente utilizado (figura 21) en diferentes industrias incluyendo la automotriz, esto debido al desarrollo en los ordenadores que han aportado un medio eficaz para poder resolver las ecuaciones que se plantea en el método de elementos finitos. [14]

5.3.2 Conceptos Generales del Método

Se puede decir que la idea general del método de elementos finitos es discretizar o dividir un sólido o continuo en un conjunto de elementos interconectados entre sí por medio de nodos; de esta manera las ecuaciones que rigen el comportamiento del sólido también rigen el del elemento. Por esta razón es posible pasar de un sistema continuo que cuenta con un número infinito de grados de libertad que a su vez es regido por una ecuación diferencial o bien por un conjunto de ellas, a un sistema cuyos grados de libertad sean finitos, el cual puede modelar su comportamiento a través un sistema de ecuaciones lineales o no lineales. [14]

Los sistemas se pueden dividir (figura 22) en:

- Dominio: Espacio geométrico en el cual se va a analizar el sistema.
- Condiciones de contorno o restricciones: Son variables que se conocen del sistema, ya sean fuerzas, desplazamientos, etc…
- Incógnitas o soluciones: Son las variables que se está buscando conocer al finalizar el estudio o bien al solucionar el sistema. Pueden ser de igual manera fuerzas, desplazamientos, etc…

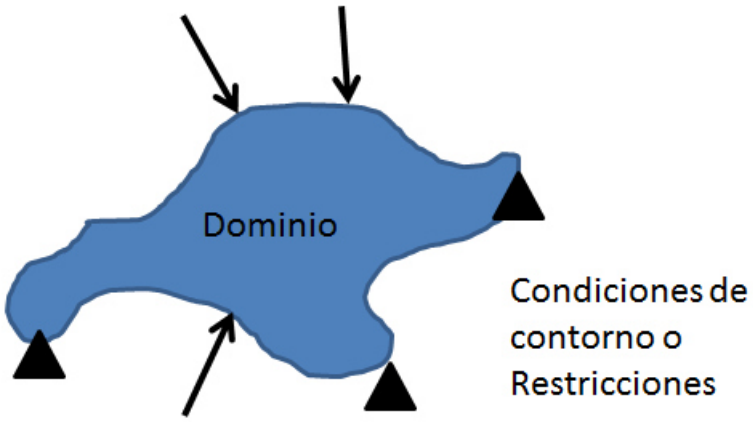


Figura 22. Explicación gráfica de dominio y condiciones de contorno o restricciones.

Lo que caracteriza al método de elementos finitos es que para poder solucionar el problema el dominio es dividido en subdominios que son llamados elementos, estos elementos se conectan entre sí a través de nodos en los cuales se aplican las incógnitas fundamentales del sistema. Para el caso de problemas estructurales las incógnitas son los desplazamientos nodales, ya que obteniendo dichos desplazamientos es posible hacer el cálculo del resto de las incógnitas que nos sean de interés así como las tensiones o bien las deformaciones. A estas incógnitas se les conoce como grados de libertad de cada nodo del modelo. [14]

CAPÍTULO 6: CHASIS

6.1 ¿QUÉ ES UN CHASIS?

El chasis constituye un armazón, así como lo establece Calvo (1997), cuyo principal propósito es realizar un vínculo o conexión de forma rígida entre la suspensión trasera y delantera de un vehículo, aunado a esto es el principal sustento para los diferentes sistemas de un vehículo, no obstante debe asegurar ante un impacto la integridad del conductor y de sus tripulantes, la forma en que lo protege es absorbiendo la energía del impacto y gracias al diseño, deformándose en direcciones contrarias a la dirección del conductor y de sus pasajeros. [7]

Hablando del diseño del chasis se puede decir que existe un punto de equilibrio entre la rigidez, el espacio (debido a los sistemas del vehículo), el peso y finalmente el costo del mismo. Para esto debe considerarse la resistencia estática y posteriormente a esfuerzos de fatiga, de esta manera se comprueba que las cargas en las uniones soporten y finalmente se comprueba la factibilidad de fabricación de la estructura. En este proyecto se tomarán en cuenta los esfuerzos que puedan considerarse estáticos.

Por otra parte los chasises no deben someterse a esfuerzos que alcancen los límites últimos de tensión sino que se diseñan para que trabajen en rangos menores ya que las características principales del chasis son indeformabilidad, resistencia a la flexión, resistencia a la torsión y finalmente la capacidad de carga; de no ser así estas variables formarían parte considerable de los parámetros para la conducción del vehículo. [7]

6.2 CHASIS TIPO ESCALERA

Así como lo menciona Sawyer (2003) no fue hasta los años 1970's que gracias a la utilización de diseño asistido por computadora mejor conocido como CAD (por sus siglas en inglés) y CAE (por sus siglas en inglés) ingeniería asistida por computadora que facilitó el diseño de los autos monocasco tanto los autos y las camionetas eran construidos con cuerpo y chasis separados. Este método de construcción era más sencillo y facilitaba soportar diferentes vehículos con la misma base. [31]

Figura 23. Chasis tipo escalera. [27]

El uso principal de la construcción de chasises de tipo escalera (figura 23) es para el sector de camionetas y camiones, hablando del beneficio que tiene este tipo de construcción es que facilita el uso de diferentes cabinas y componentes sobre el mismo marco o con algunos cambios que pueden llegar a ser adaptados en las mismas líneas de producción por lo que la fabricación tiende a ser de menor costo y con menor complejidad.

CAPÍTULO 7: DISEÑO DE UN CHASIS TIPO ESCALERA

7.1 INTRODUCCIÓN

La forma más común de construcción es utilizando dos vigas longitudinales y conectarlos con diferentes miembros transversales, dichos miembros transversales tienen usos específicos ya que son en función de los requerimientos del vehículo.

Asimismo, Wakeham (2009) menciona en su libro que el chasis perfecto sería un tubo con diámetro amplio y de pared delgada, ya que el chasis tiene dos objetivos principales [35]:

1. Soportar tanto el peso de los componentes del vehículo como de la carga.
2. Mantener fijos los componentes de la suspensión estando en movimiento.

Hablando del primer punto el autor menciona que los diseños originales de los chasises se basaron en los carruajes de caballos, además menciona que para cumplir con este objetivo existen diferentes formas comunes de vigas como lo son las vigas tipo “I” o bien una viga tipo “C” que son capaces de soportar altas cargas y que actualmente siguen siendo utilizadas por este mismo propósito.

Para hablar sobre el segundo punto es necesario hablar sobre el manejo, ya que al estar el vehículo en movimiento y hacer giros o simplemente al pasar por caminos desiguales ocasiona que el chasis sea sometido a torques que a su vez generan que el chasis se tuerza. La forma que menciona el autor para solucionar dicha torsión es simplemente el uso de un tubo, ya que al alejar el material lo más posible del centro de aplicación del torque, mayor será la resistencia a la deformación.

7.2 CRITERIOS DE RIGIDEZ APLICADO A CHASIS

Habiendo hablado sobre distintas expresiones sobre la rigidez, se puede deducir que teniendo un mayor momento de inercia, un mayor módulo de elasticidad, o una mayor área de sección transversal; mayor será la rigidez.

Si se habla específicamente de un chasis, se toman principalmente dos aspectos de la rigidez:

1. Rigidez a flexión: ya que es la resistencia a la deformación debida al peso de los distintos elementos que conforman un vehículo.
2. Rigidez torsional: debido a que dicha rigidez es necesaria al momento de que el vehículo se encuentra en movimiento y que las

ruedas delanteras y traseras se encuentran pasando por distintos caminos a distintos momentos (ejemplo: al pasar por un bache).

Por esta razón este proyecto se centra principalmente en el diseño de un chasis tomando en cuenta dichas rigideces como las principales.

7.3 CRITERIOS SOBRE EL PESO Y SU DISTRIBUCIÓN

Como parte del diseño de un chasis, si se habla sobre el peso y la distribución del mismo, deberán tomarse en cuenta los siguientes puntos:

- A menor peso contenga el chasis, mejor se aprovechará la potencia del motor; esto siempre y cuando se mantenga la rigidez necesaria del vehículo.
- Es conveniente que el centro de gravedad se localice lo más bajo posible, para que se disminuya el balanceo.

7.4 CRITERIOS SOBRE EL ESPACIO

Por otra parte existen diferentes componentes que van sujetados al chasis o bien que deben ser respetados debido a que son necesarios para el ensamble del vehículo completo; por lo que se enlistan a continuación algunos puntos que deberán ser tomados en cuenta al diseñar un chasis:

- Deberá tomarse en cuenta el espacio necesario para realizar los mantenimientos necesarios a las partes que generan el movimiento del vehículo.

- Es necesario tomar en cuenta la ergonomía tanto del pasajero como del conductor para que pueda transportarse cómodamente y sin tener algún problema al subir o bajar del vehículo.

7.5 CARGAS APLICADAS AL CHASIS

Si se habla sobre esfuerzos, el chasis se encuentra mayormente sometido a flexión y a torsión, de estos dos es más importante los esfuerzos de torsión ya que estos afectan a las cargas de las ruedas.

Por otra parte los esfuerzos de flexión deben ser soportados tanto en condiciones estáticas como cuando el vehículo se somete a esfuerzos provenientes de la aerodinámica, estos esfuerzos se deben a que el aire debe empujar el vehículo hacia abajo para mantener el mismo en contacto con el suelo por lo que esfuerza el chasis a flexión.

A su vez el chasis debe de diseñarse tomando en cuenta las peores condiciones, las cuales son:

- Cuando se acelera bruscamente el vehículo
- Al frenarse bruscamente en condiciones de manejo en recta como en curva.
- Cuando el vehículo se encuentra a la máxima velocidad en una curva.

Asimismo debido a las condiciones ya mencionadas se generan cargas que deben ser aplicadas al chasis en los siguientes puntos:

- Sujeciones de la suspensión

- Sujeciones donde el peso sea considerable ya sea por cargas del mismo vehículo o bien por cargas externas.

A todo esto es posible clasificar o catalogar las cargas de la siguiente manera:

1. Cargas permanentes
2. Cargas variables

CAPÍTULO 8: METODOLOGÍA

Para poder mejorar el proceso actual de la empresa se tomó la decisión que para reducir el tiempo de respuesta era necesario reducir la cantidad de iteraciones entre los departamentos de elementos finitos y de diseño por lo que era necesario aislar al diseñador en un principio del proceso y mejorar su asertividad en la propuesta inicial.

Por ende para poder lograr el objetivo fue necesario entender la limitante del software de análisis con que se cuenta en los ordenadores de dicho departamento por lo que el trabajo se realizó pensando en que el diseñador fuera capaz de obtener los datos de entrada requeridos y que además fuera capaz de interpretar los datos de salida para poder hacer uso de los mismos.

8.1 DISCRETIZACIÓN DEL CHASIS TIPO ESCALERA

Para poder trabajar con el chasis completo fue necesario discretizarlo (figuras 24 a 26) y esto quiere decir que se subdividió en elementos que asemejan la estructura de un chasis tipo escalera pero facilitan el cálculo estructural.

El criterio que se siguió para subdividir la estructura llámese chasis tipo escalera fue el siguiente:

- *Elementos*: al subdividir la estructura se crean elementos rectos de sección constante y son unidos entre sí por los diferentes tipos de nodos (representados por líneas negras en las siguientes imágenes).
- *Nodos primarios*: son las uniones necesarias para el armado del chasis, estos corresponden principalmente a las uniones entre los largueros y los travesaños. (representados por puntos verdes en las siguientes imágenes)
- *Nodos secundarios*: son aquellas uniones que dividen los elementos por requerimientos geométricos (por ejemplo: cambio de sección o cambio de dirección, etc.) (representados por puntos negros en las siguientes imágenes)
- *Nodos de cálculo*: son los puntos necesarios para poder realizar el cálculo estructural. (representados por cuadros rojos en las siguientes imágenes)

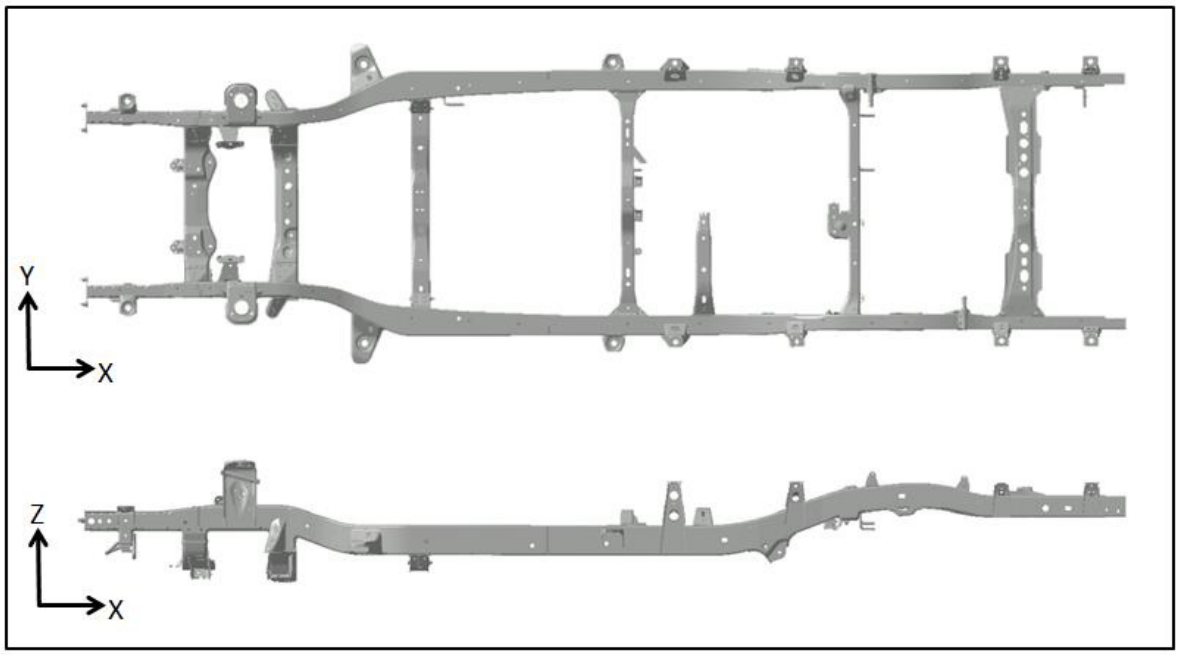


Figura 24. CAD de un chasis tipo escalera.

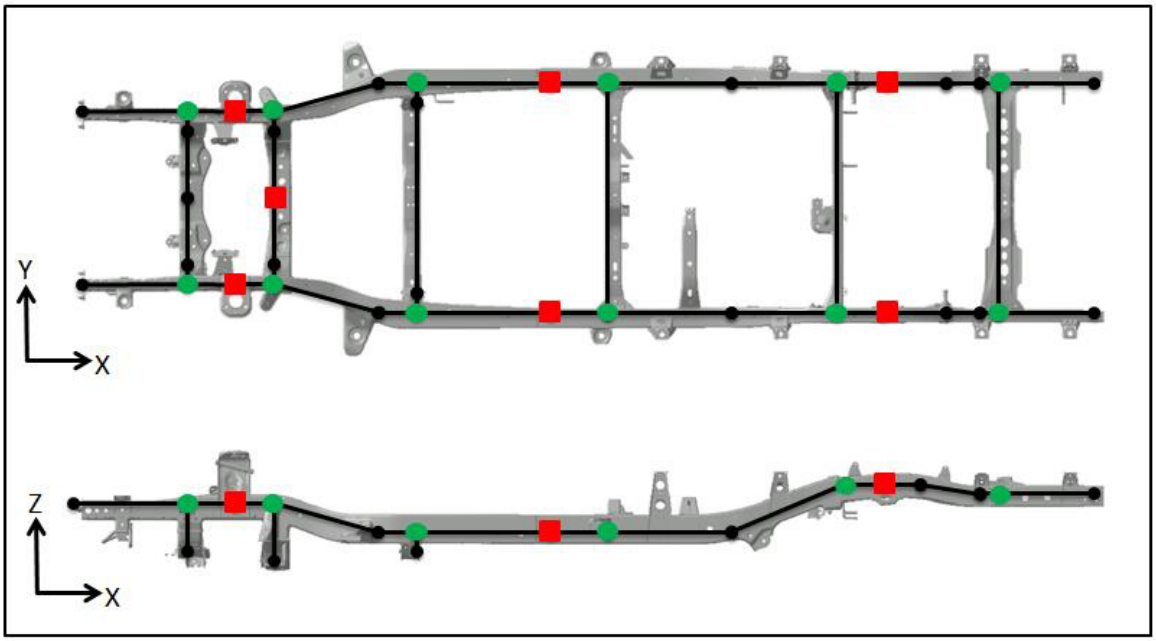


Figura 25. Discretización de un chasis tipo escalera montado sobre CAD de chasis.

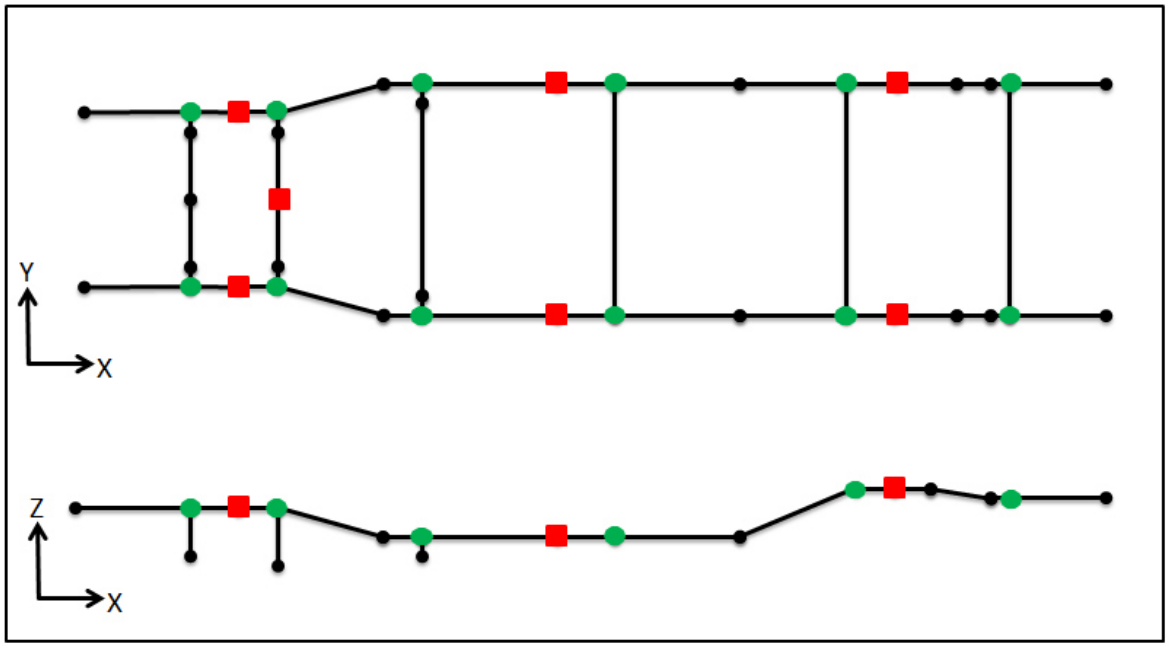


Figura 26. Discretización de un chasis tipo escalera.

8.2 Definición de los Elementos en Coordenadas Locales

Teniendo discretizado el chasis, se procede a definir cada una de los elementos, para poder tener un mejor control se nombraron los elementos y se enumeraron los nodos como se muestra en las figuras 27 y 28 respectivamente.

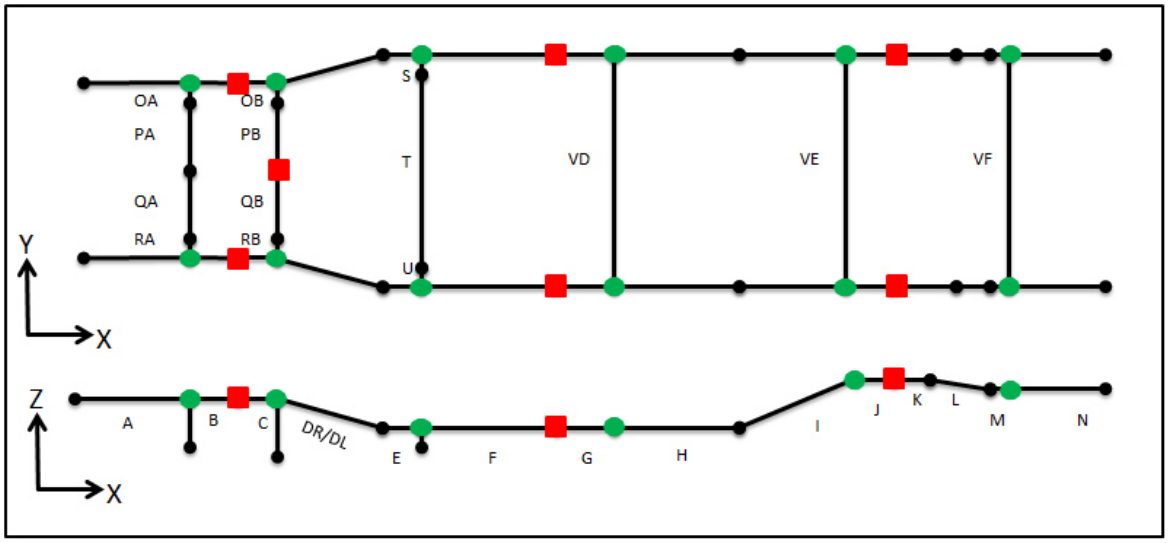


Figura 27. Discretización de chasis tipo escalera con los elementos nombrados.

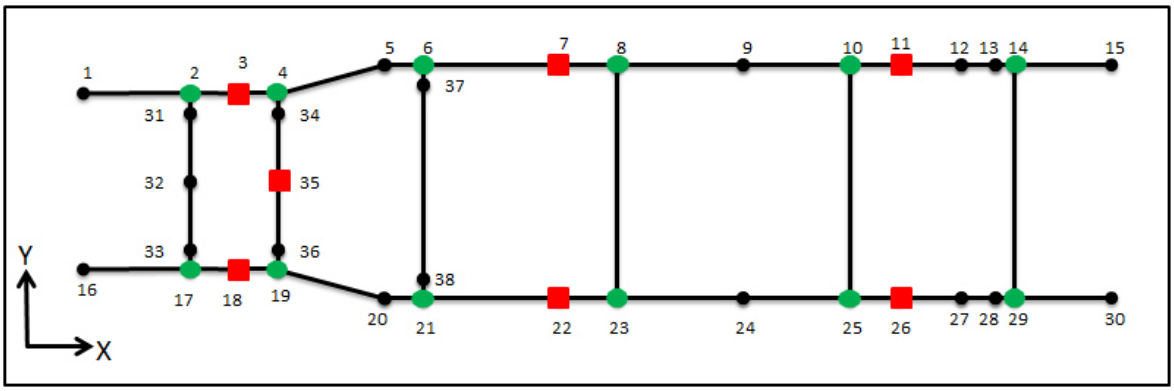


Figura 28. Discretización de chasis tipo escalera con los elementos enumerados.

Para la definición de cada uno de los elementos en 3 dimensiones se asignan coordenadas locales considerando el origen del elemento en el punto $(X_i,\ Y_i,\ Z_i)$ y su otro extremo como $(X_j,\ Y_j,\ Z_j)$ como se muestra en la figura 29.

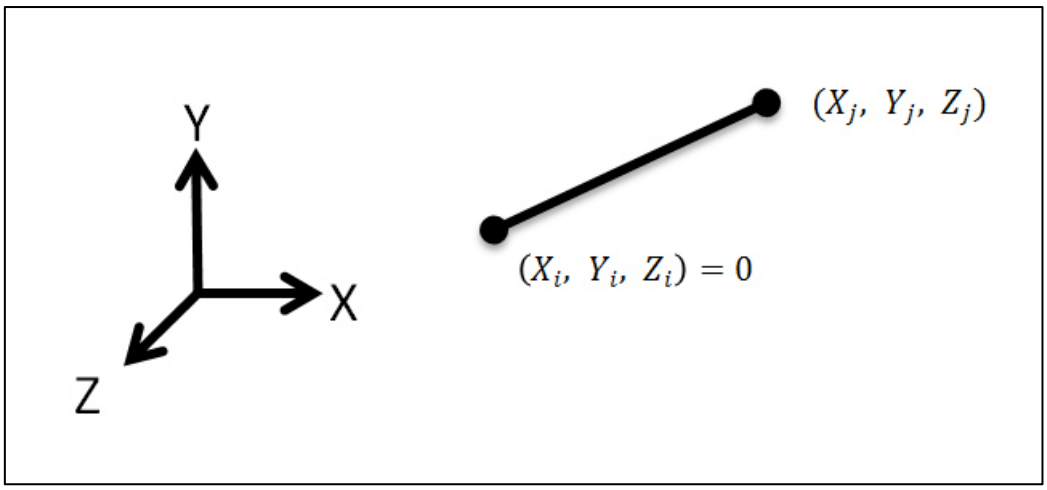


Figura 29. Representación de un elemento en coordenadas locales.

Por otra parte se establece la matriz de rigidez del elemento utilizando la ecuación 6 mencionada en el capítulo 3.7.2 tal y como se muestra a continuación:

$$\begin{bmatrix}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} & 0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & \frac{6EI_z}{L^2} \\
0 & 0 & \frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 & 0 & 0 & -\frac{12EI_y}{L^3} & 0 & -\frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & \frac{GI_0}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{GI_0}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{2EI_z}{L} \\
-\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} & 0 & \frac{12EI_z}{L^3} & 0 & 0 & 0 & -\frac{6EI_z}{L^2} \\
0 & 0 & -\frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 & 0 & 0 & \frac{12EI_y}{L^3} & 0 & \frac{6EI_y}{L^2} & 0 \\
0 & 0 & 0 & -\frac{GI_0}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GI_0}{L} & 0 & 0 \\
0 & 0 & -\frac{6EI_y}{L^2} & 0 & \frac{2EI_y}{L} & 0 & 0 & 0 & \frac{6EI_y}{L^2} & 0 & \frac{4EI_y}{L} & 0 \\
0 & \frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L} & 0 & -\frac{6EI_z}{L^2} & 0 & 0 & 0 & \frac{4EI_z}{L}
\end{bmatrix}$$

(Ecuación 6) repetida.

Como se explicó en el capítulo 3.7.2 usando la matriz (ecuación 6), y considerando las siguientes fórmulas o constantes (ecuación 27 a 33) para cada una de las variables o constantes de la matriz:

$$E = 210\ GPa\ (Constante\ del\ Acero)\ \text{(ArcelorMittal)} \quad \text{(Ecuación 27)}$$

$$G = 81\ GPa\ (Constante\ del\ Acero)\ \text{(ArcelorMittal)} \quad \text{(Ecuación 28)}$$

$$L = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2 (Z_j - Z_i)^2} \quad \text{(Ecuación 29)}$$

$$A = (b * h) - [(b - 2t)(h - 2t)] \quad \text{(Ecuación 30)}$$

Siendo b la base de la sección transversal, h la altura de la sección transversal y finalmente t el espesor de la lámina de acero.

$$I_{zz} = \sum(I_z) + \sum Ad^2 \quad \text{(Ecuación 31)}$$

Siendo I_z los momentos de inercia de cada una de las divisiones de la sección transversal del elemento con respecto al eje "z", A siendo el área de cada una de las divisiones y d representando la distancia entre el centroide de cada una de las secciones y el eje "z".

$$I_{yy} = \sum(I_y) + \sum Ad^2 \quad \text{(Ecuación 32)}$$

Siendo I_y los momentos de inercia de cada una de las divisiones de la sección transversal del elemento con respecto al eje “y”, A siendo el área de cada una de las divisiones y d representando la distancia entre el centroide de cada una de las secciones y el eje “y”.

$$I_o = \frac{4tA^2{}_m}{L_m} \qquad \text{(Ecuación 33)}$$

Siendo t el espesor de la lámina de acero, A_m representando al área encerrada por la línea media del espesor de la sección transversal y L_m representa el perímetro de la línea media del espesor de la sección transversal.

8.3 Definición de los Elementos en Coordenadas Globales

Teniendo la matriz de rigidez del elemento en coordenadas locales es necesario hacer una transformación a coordenadas globales para que se encuentre el elemento referenciado al origen global de la estructura. Para poder hacer la transformación es necesario generar una matriz de transformación como se explicó en el capítulo 3.7.3 y usando la matriz (ecuación 12):

$$[T] = \begin{bmatrix} l_x & l_y & l_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ m_x & m_y & m_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ n_x & n_y & n_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & l_x & l_y & l_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & m_x & m_y & m_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_x & n_y & n_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & l_x & l_y & l_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & m_x & m_y & m_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & n_x & n_y & n_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & l_x & l_y & l_z \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_x & m_y & m_z \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n_x & n_y & n_z \end{bmatrix}$$

(Ecuación 12) repetida.

Y calculando los valores de la matriz con las ecuaciones 13 a 18 se logra obtener la matriz de transformación. Posteriormente se procede a multiplicar la matriz de rigidez del elemento en coordenadas locales por la matriz de transformación y a su vez nuevamente multiplicarla por la matriz de trasformación transpuesta, siguiendo lo explicado en el capítulo 3.7.4, para poder obtener la matriz de rigidez del elemento en coordenadas globales (ecuación 24).

$$[K]^{Globales} = [T][K]^{locales}[T]^T \qquad \text{(Ecuación 24) repetida}$$

8.4 MATRIZ DE RIGIDEZ DEL CHASIS TIPO ESCALERA

Teniendo la matriz de rigidez de cada uno de los elementos que conforman la discretización del chasis en coordenadas globales se procede a hacer una sumatoria de sub matrices como se explica en el capítulo 3.7.5 (ecuación 26) y

se forma de esta manera la matriz de rigidez de la estructura que en este caso representa la discretización del chasis tipo escalera.

$$[K]_{Global}^{Estructura} = \sum_{ij} \begin{bmatrix} [0] & [0] & [0] & [0] \\ [0] & K_{ii} & K_{ij} & [0] \\ [0] & K_{ji} & K_{jj} & [0] \\ [0] & [0] & [0] & [0] \end{bmatrix}^{ij} \qquad \text{(Ecuación 26) repetida}$$

8.5 Planteamiento y Solución de la Estructura

Existen diferentes métodos de solución de estructuras como ya se mencionaron en el capítulo 4, en este caso se decidió optar por el método de la rigidez (ecuación 35) al tener la matriz global de rigidez (ecuación 34), por lo tanto se planteó de la siguiente manera:

$$K_G = [K]_{Global}^{Estructura} \qquad \text{(Ecuación 34)}$$

$$F = K_G U \qquad \text{(Ecuación 35)}$$

Siendo F el vector que representa las fuerzas, U que es el vector que representa los desplazamientos y finalmente la $\mathrm{K_G}$ la cual representa a la matriz de rigidez en coordenadas globales.

Para poder obtener los desplazamientos es necesario despejar el vector desplazamientos (ecuación 36) como se muestra a continuación:

$$U = K_G^{-1} F \qquad \text{(Ecuación 36)}$$

Como se mencionó anteriormente, para el presente trabajo se busca obtener la rigidez vertical y rigidez torsional del chasis por lo que se tienen que plantear dos soluciones distintas.

8.5.1 Rigidez Vertical

Para poder plantear el caso de rigidez vertical se entiende que se colocan 2 fuerzas de la misma magnitud (1000N) en el centro entre los ejes, además que se restringe en la posición donde se coloca el eje frontal en las tres direcciones (X,Y,Z) y en el eje trasero se restringe solamente en dos direcciones (Y,Z) lo cual permite un desplazamiento en X (figura 30) y se mide el desplazamiento en los puntos donde se colocan las fuerzas.

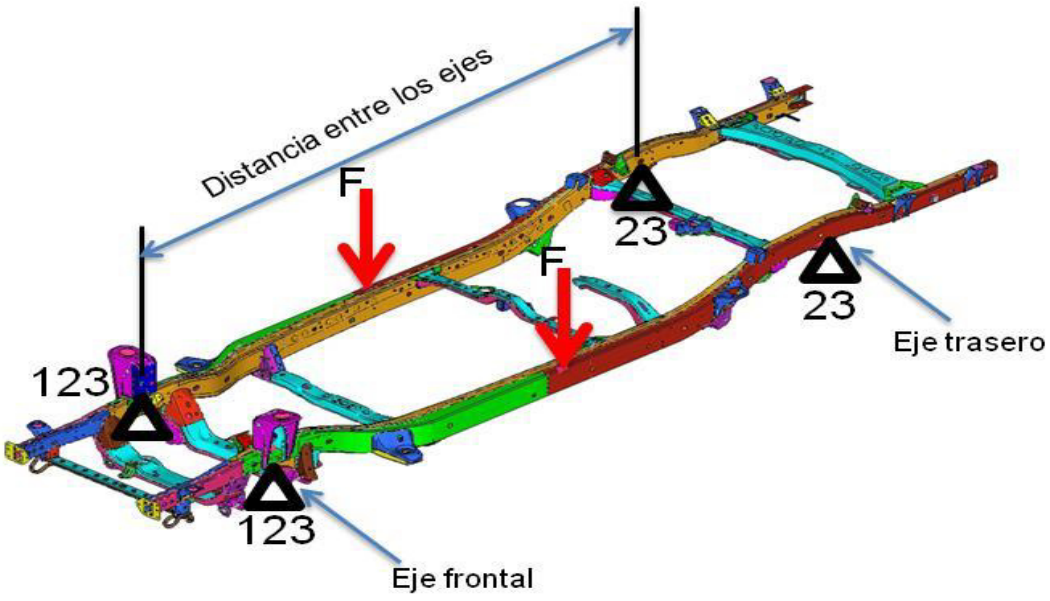


Figura 30. Representación gráfica de planteamiento para solución de rigidez vertical.

Para poder entender el planteamiento para la solución es necesario traducirlo a la numeración que se le colocó a los nodos de la discretización, esto quiere decir que para el caso de rigidez vertical se colocan las fuerzas en Z en los nodos 7 y 2 (color rojo) ; y las restricciones frontales en X,Y,Z son en los nodos 3 y 18 (color azul); para la parte trasera se restringen en Y, Z los nodos 11 y 26 (color azul) los cuales son mostrados en rojo en la figura 31.

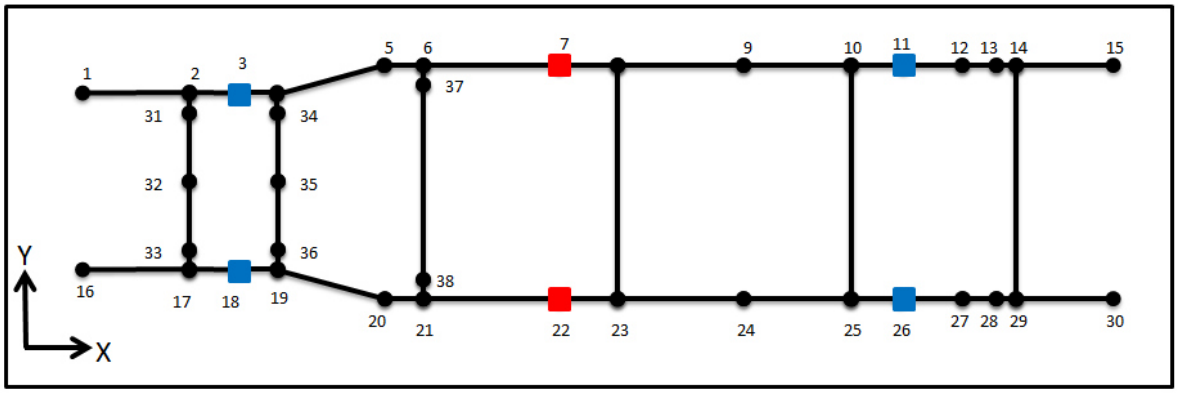


Figura 31. Representación para solución de rigidez vertical.

Teniendo los desplazamientos en los nodos se procede a calcular la rigidez vertical del chasis tipo escalera con la siguiente fórmula (ecuación 37):

$$RV = \frac{PW^3}{48\left(\frac{dz_7 + dz_{22}}{2}\right)} \qquad \text{(Ecuación 37)}$$

Siendo P la fuerza aplicada (1000N), Wes la distancia entre los ejes, finalmente dz_7 y dz_{22} representan los desplazamientos en “Z” en los nodos 7 y 22.

8.5.2 RIGIDEZ TORSIONAL

Para poder plantear el caso de rigidez torsional se entiende que se colocan 2 fuerzas encontradas de la misma magnitud (1000N) en los extremos del eje frontal, además se restringe en una dirección (Z) en el centro del segundo travesaño y en el eje trasero se restringe tres direcciones (X, Y, Z) lo cual permite que gire la parte frontal (figura 32) y se mide el desplazamiento en los puntos donde se colocan las fuerzas.

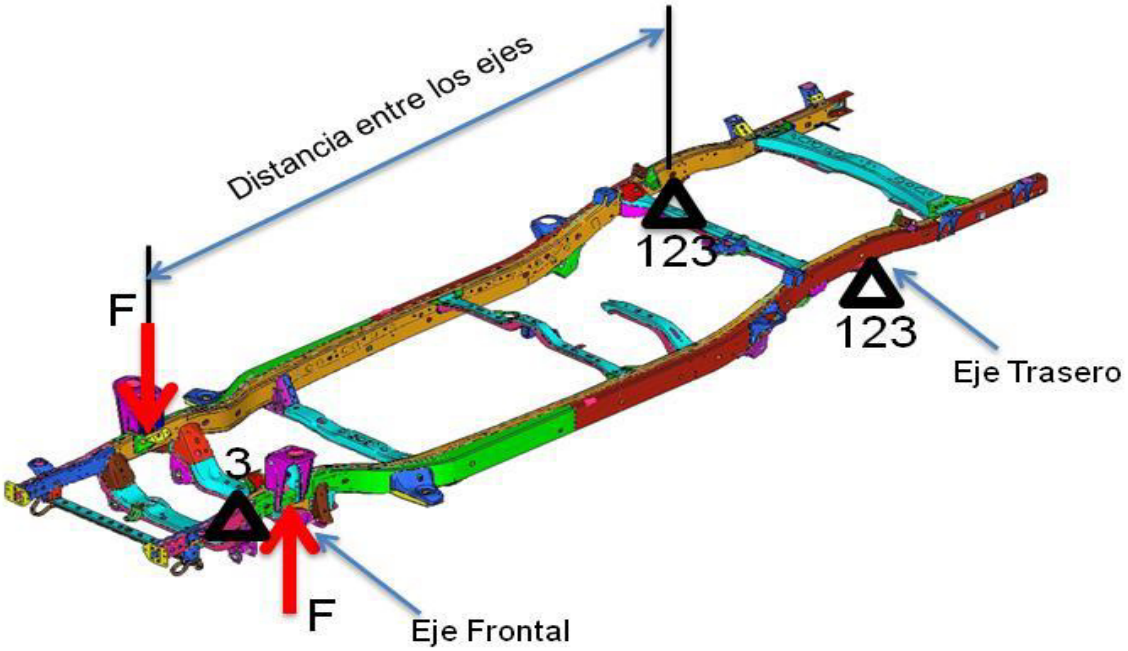


Figura 32. Representación gráfica de planteamiento para solución de rigidez torsional.

Para poder entender el planteamiento para la solución es necesario traducirlo a la numeración que se le colocó a los nodos de la discretización, esto quiere decir que para el caso de rigidez tosional se colocan las fuerzas en Z en los nodos 3 y 18 (color rojo) ; y la restricción frontal en Z son en el nodo 35 (color azul); para la parte trasera se restringen en X, Y, Z los nodos 11 y 26 (color azul) los cuales son mostrados en rojo en la figura 33.

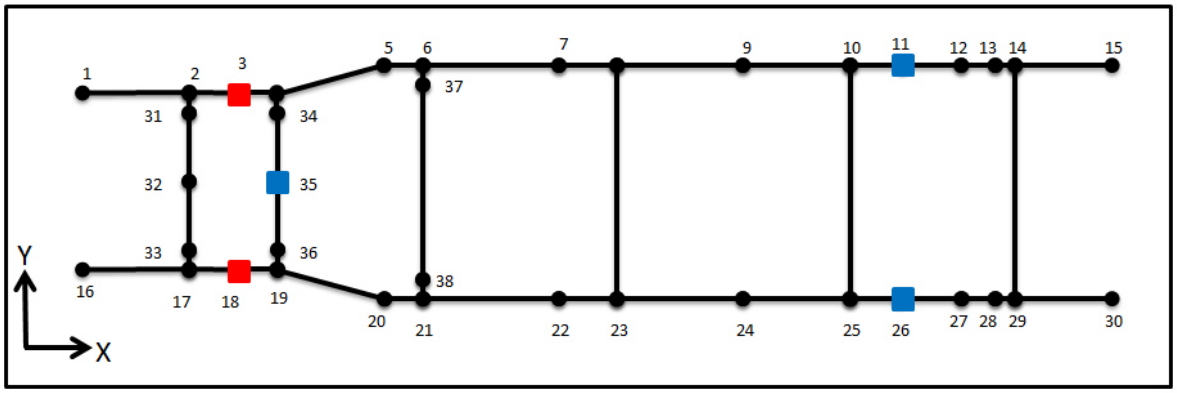


Figura 33. Representación para solución de rigidez torsional.

Teniendo los desplazamientos en los nodos se procede a calcular la rigidez torsional del chasis tipo escalera con la siguiente fórmula (ecuación 38):

$$RT = \left(\frac{PL^2}{dz_3 + dz_{18}}\right) W \qquad \text{(Ecuación 38)}$$

Siendo P la fuerza aplicada (1000N), Wes la distancia entre los ejes, L el ancho del chasis, es decir la distancia entre el nodo 3 y el nodo 12 y finalmente dz_3 y dz_{12} representan los desplazamientos en "Z" en los nodos 3 y 12.

8.6 Optimización de Chasis tipo Escalera

Ya habiendo planteado la forma de obtener las soluciones de la estructura para rigidez vertical y rigidez torsional de un chasis tipo escalera, el siguiente paso es optimizar el chasis; para esto se tienen que tomar en cuenta diversos parámetros para poder entender mejor esto se muestra el diagrama de flujo a continuación (figura 34).

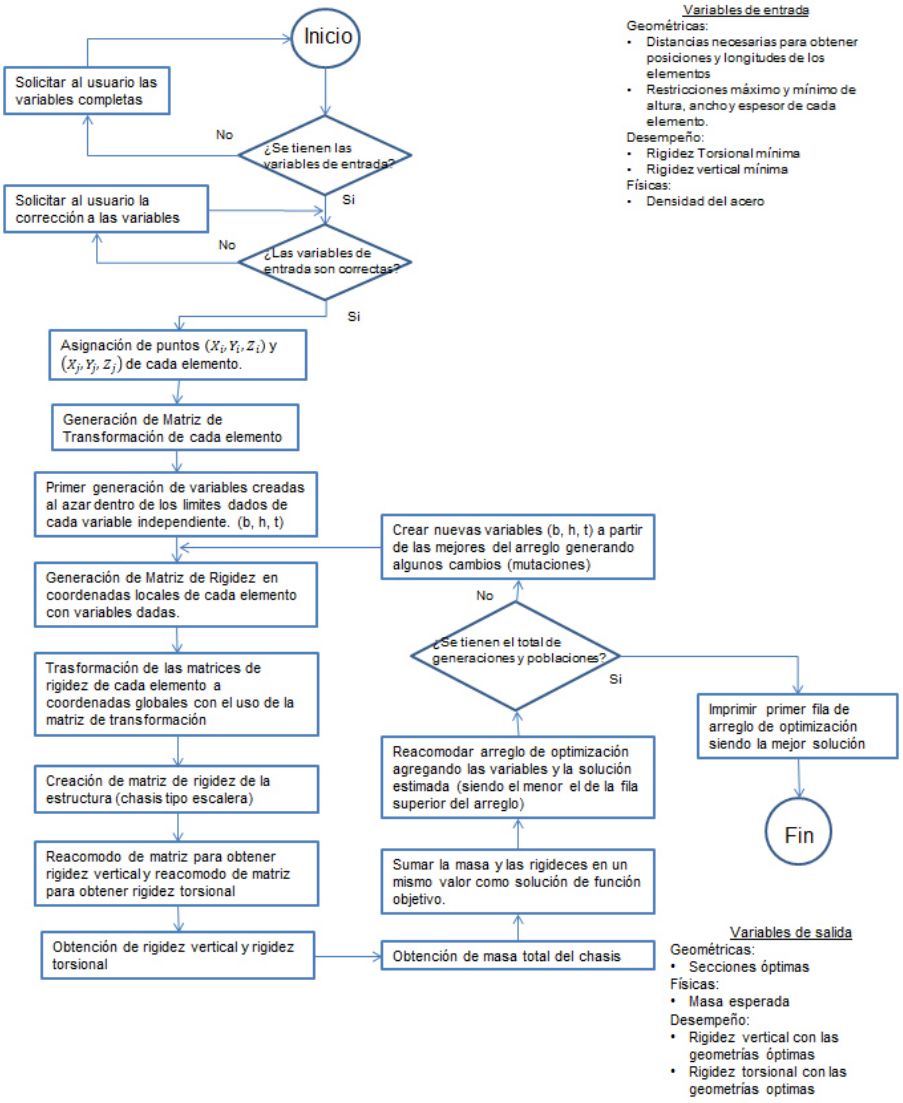


Figura 34. Diagrama de flujo de metodología.

8.6.1 FUNCIÓN OBJETIVO

Para la optimización, como se habló en el capítulo 5, es necesario plantear lo que se busca optimizar. En este caso en particular se busca reducir la mayor cantidad de masa total de la estructura y al mismo tiempo para optimizar la estructura se busca obtener la menor rigidez vertical y rigidez torsional sin sobrepasar los límites que se busca plantee el usuario, por lo tanto si planteamos como función objetivo la siguiente ecuación (ecuación 39):

$$f_{(x)} = \sum \rho AL + RT + RV \qquad \text{(Ecuación 39)}$$

Siendo ρ la densidad del acero (7850 kg/m^3), A el área de la sección transversal de cada elemento, L la longitud de cada elemento, RT representando la rigidez torsional y finalmente RV la rigidez vertical.

8.6.2 Variables de Independientes y Dependientes

Las variables independientes del sistema son la base, altura y espesor de la lámina de cada uno de los elementos lo cual afectan directamente el área de la sección transversal. Esto nos lleva a que la creación de la matriz de rigidez de la estructura sea una parte fundamental de la función objetivo ya que dichas variables afectan directamente dicha matriz haciéndola una variable dependiente del sistema a optimizar.

8.6.3 Restricciones

Para el análisis se agregaron los dos casos de estudio los cuales son la rigidez torsional y la rigidez vertical, para esto se le solicita al usuario introducir los limites inferiores, es decir un límite que restrinja la rigidez de la estructura al momento de optimizar la masa para que a pesar de reducir lo más que sea posible no se pierda el funcionamiento esperado de la estructura.

Por otra parte existen límites geométricos que restringen a los elementos en altura y ancho dependiendo del lugar en que se encuentren ya que al chasis se le ensamblan otros componentes los cuales no pueden ser modificados.

A su vez se tiene restricciones de calibre, ya que por cuestiones de corrosión se restringe el límite inferior y por cuestiones de capacidades de los sistemas de manufactura (ejemplo. prensas) los cuales limitan el calibre máximo dependiendo de las capacidades de la empresa.

8.6.4 PROGRAMACIÓN DE OPTIMIZACIÓN

Para optimizar el chasis tipo escalera se programó una serie de funciones y subrutinas en visual basic, ya que este paquete se encuentra en todas las computadoras con las que cuentan los diseñadores, se programó siguiendo los puntos anteriores como referencia.

Para la optimización se utilizó una función traducida de la subrutina conocida como PIKAIA que es una función de optimización general que fue escrita en FORTRAN-77 y que es de dominio público que fue compartida por el High Altitude Observatory (HAO); y que a su vez fue traducida a visual basic para que pudiera ser utilizada por los diseñadores de la empresa. [9] [11]

La subrutina fue creada para maximizar por lo que se modificó en parte para que minimizara ya que es lo que se busca realizar para la optimización dentro del presente trabajo, para poder realizar esto se modificó la función PIKAIA_SORT_FUNC [11] invirtiendo los símbolos que se muestran en rojo en las figuras 35 a 37:

```
Private Function PIKAIA_SORT_FUNC(ByRef NROWS As Integer, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant)

Const Q_VAL As Integer = 11
' Q_VAL = smallest subfile to use quicksort on
Const LGN_VAL As Integer = 32
' LGN_VAL = log base 2 of maximum n;

Dim i As Integer
Dim j As Integer
Dim l As Integer
Dim m As Integer
Dim r As Integer
Dim s As Integer
Dim t As Integer

Dim XDATA_VAL As Double

On Error GoTo ERROR_LABEL

Dim LTEMP_ARR(1 To LGN_VAL) As Integer
Dim RTEMP_ARR(1 To LGN_VAL) As Integer

PIKAIA_SORT_FUNC = False

'Initialize the stack
LTEMP_ARR(1) = 1
RTEMP_ARR(1) = NROWS
s = 1
'Initialize the pointer array
For i = 1 To NROWS
    IFIT_ARR(i) = i
Next i
2:

If s > 0 Then
    l = LTEMP_ARR(s)
    r = RTEMP_ARR(s)
    s = s - 1
```

Figura 35. Función para re-acomodar original, con marcas de cambios. [11]

```
3:
    If (r - l) < Q_VAL Then 'Use straight insertion
        For i = l + 1 To r
            t = IFIT_ARR(i)
            XDATA_VAL = FITNS_ARR(t)
            For j = i - 1 To l Step -1
                If FITNS_ARR(IFIT_ARR(j)) <= XDATA_VAL Then GoTo 5
                IFIT_ARR(j + 1) = IFIT_ARR(j)
            Next j
            j = l - 1
5:            IFIT_ARR(j + 1) = t
        Next i
    Else
        'Use quicksort, with pivot as median of FITNS_ARR(l), FITNS_ARR(m), FITNS_ARR(r)
        m = (l + r) / 2
        t = IFIT_ARR(m)
        If FITNS_ARR(t) < FITNS_ARR(IFIT_ARR(l)) Then
            IFIT_ARR(m) = IFIT_ARR(l)
            IFIT_ARR(l) = t
            t = IFIT_ARR(m)
        End If
        If FITNS_ARR(t) > FITNS_ARR(IFIT_ARR(r)) Then
            IFIT_ARR(m) = IFIT_ARR(r)
            IFIT_ARR(r) = t
            t = IFIT_ARR(m)
            If FITNS_ARR(t) < FITNS_ARR(IFIT_ARR(l)) Then
                IFIT_ARR(m) = IFIT_ARR(l)
                IFIT_ARR(l) = t
                t = IFIT_ARR(m)
            End If
        End If

        'Partition
        XDATA_VAL = FITNS_ARR(t)
        i = l + 1
        j = r - 1

7:      If i <= j Then
8:        If FITNS_ARR(IFIT_ARR(i)) < XDATA_VAL Then
            i = i + 1
            GoTo 8
          End If
```

Figura 36. Función para re-acomodar original, con marcas de cambios. (Parte 2). [11]

```
9:          If XDATA_VAL < FITNS_ARR(IFIT_ARR(j)) Then
              j = j - 1
              GoTo 9
            End If

            If i <= j Then
              t = IFIT_ARR(i)
              IFIT_ARR(i) = IFIT_ARR(j)
              IFIT_ARR(j) = t
              i = i + 1
              j = j - 1
            End If
            GoTo 7
          End If

'Stack the larger subfile
          s = s + 1
          If (j - l) > (r - i) Then
              LTEMP_ARR(s) = l
              RTEMP_ARR(s) = j
              l = i
          Else
              LTEMP_ARR(s) = i
              RTEMP_ARR(s) = r
              r = j
          End If
          GoTo 3
      End If
      GoTo 2
End If

PIKAIA_SORT_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_SORT_FUNC = False
End Function
```

Figura 37. Función para re-acomodar original, con marcas de cambios. (Parte 3) [11]

Modificando la función se reacomoda cada una de las poblaciones y generaciones mostrando en el primer puesto del arreglo la mejor solución para este caso el mínimo valor de la función objetivo, a continuación se muestra la función modificada en las figuras 38 a 40:

```
Private Function PIKAIA_SORT_FUNC(ByRef NROWS As Integer, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant)

Const Q_VAL As Integer = 11
' Q_VAL = smallest subfile to use quicksort on
Const LGN_VAL As Integer = 32
' LGN_VAL = log base 2 of maximum n;

Dim i As Integer
Dim j As Integer
Dim l As Integer
Dim m As Integer
Dim r As Integer
Dim s As Integer
Dim t As Integer

Dim XDATA_VAL As Double

On Error GoTo ERROR_LABEL

Dim LTEMP_ARR(1 To LGN_VAL) As Integer
Dim RTEMP_ARR(1 To LGN_VAL) As Integer

PIKAIA_SORT_FUNC = False

'Initialize the stack
LTEMP_ARR(1) = 1
RTEMP_ARR(1) = NROWS
s = 1
'Initialize the pointer array
For i = 1 To NROWS
    IFIT_ARR(i) = i
Next i
2:

If s > 0 Then
    l = LTEMP_ARR(s)
    r = RTEMP_ARR(s)
    s = s - 1
```

Figura 38. Función para re-acomodar con cambios.

```
3:
    If (r - l) < Q_VAL Then 'Use straight insertion
        For i = l + 1 To r
            t = IFIT_ARR(i)
            XDATA_VAL = FITNS_ARR(t)
            For j = i - 1 To l Step -1
                If FITNS_ARR(IFIT_ARR(j)) >= XDATA_VAL Then GoTo 5
                IFIT_ARR(j + 1) = IFIT_ARR(j)
            Next j
            j = l - 1
5:            IFIT_ARR(j + 1) = t
        Next i
    Else
        'Use quicksort, with pivot as median of FITNS_ARR(l), FITNS_ARR(m), FITNS_ARR(r)
        m = (l + r) / 2
        t = IFIT_ARR(m)
        If FITNS_ARR(t) > FITNS_ARR(IFIT_ARR(l)) Then
            IFIT_ARR(m) = IFIT_ARR(l)
            IFIT_ARR(l) = t
            t = IFIT_ARR(m)
        End If
        If FITNS_ARR(t) < FITNS_ARR(IFIT_ARR(r)) Then
            IFIT_ARR(m) = IFIT_ARR(r)
            IFIT_ARR(r) = t
            t = IFIT_ARR(m)
            If FITNS_ARR(t) > FITNS_ARR(IFIT_ARR(l)) Then
                IFIT_ARR(m) = IFIT_ARR(l)
                IFIT_ARR(l) = t
                t = IFIT_ARR(m)
            End If
        End If

        'Partition
        XDATA_VAL = FITNS_ARR(t)
        i = l + 1
        j = r - 1

7:      If i <= j Then
8:        If FITNS_ARR(IFIT_ARR(i)) > XDATA_VAL Then
            i = i + 1
            GoTo 8
          End If
```

Figura 39. Función para re-acomodar con cambios. (Parte 2)

```
9:          If XDATA_VAL > FITNS_ARR(IFIT_ARR(j)) Then
              j = j - 1
              GoTo 9
            End If

            If i <= j Then
              t = IFIT_ARR(i)
              IFIT_ARR(i) = IFIT_ARR(j)
              IFIT_ARR(j) = t
              i = i + 1
              j = j - 1
            End If
            GoTo 7
          End If

'Stack the larger subfile
          s = s + 1
          If (j - l) > (r - i) Then
              LTEMP_ARR(s) = l
              RTEMP_ARR(s) = j
              l = i
          Else
              LTEMP_ARR(s) = i
              RTEMP_ARR(s) = r
              r = j
          End If
          GoTo 3
      End If
      GoTo 2
End If

PIKAIA_SORT_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_SORT_FUNC = False
End Function
```

Figura 40. Función para re-acomodar con cambios. (Parte 3)

8.6.4 EXPORTACIÓN A CATIA

Posteriormente teniéndose los resultados de las secciones se tomó la decisión de crear un modelo parametrizado en CATIA en el cual se puedan mostrar el chasis de manera virtual por lo que se crearon 2 exportadores:

1. Secciones constantes: se tienen extrusiones de secciones constantes lo cual muestra los resultados de la optimización aplicados en la geometría de chasis.
2. Secciones variables: se realizó haciendo variables tanto las secciones como los calibres por lo que muestra un patrón más parecido a lo que

sería a la realidad para facilitar el trabajo del diseñador al trabajar en las diferentes secciones.

Ambos exportadores siguen el siguiente diagrama de flujo (figura 41).

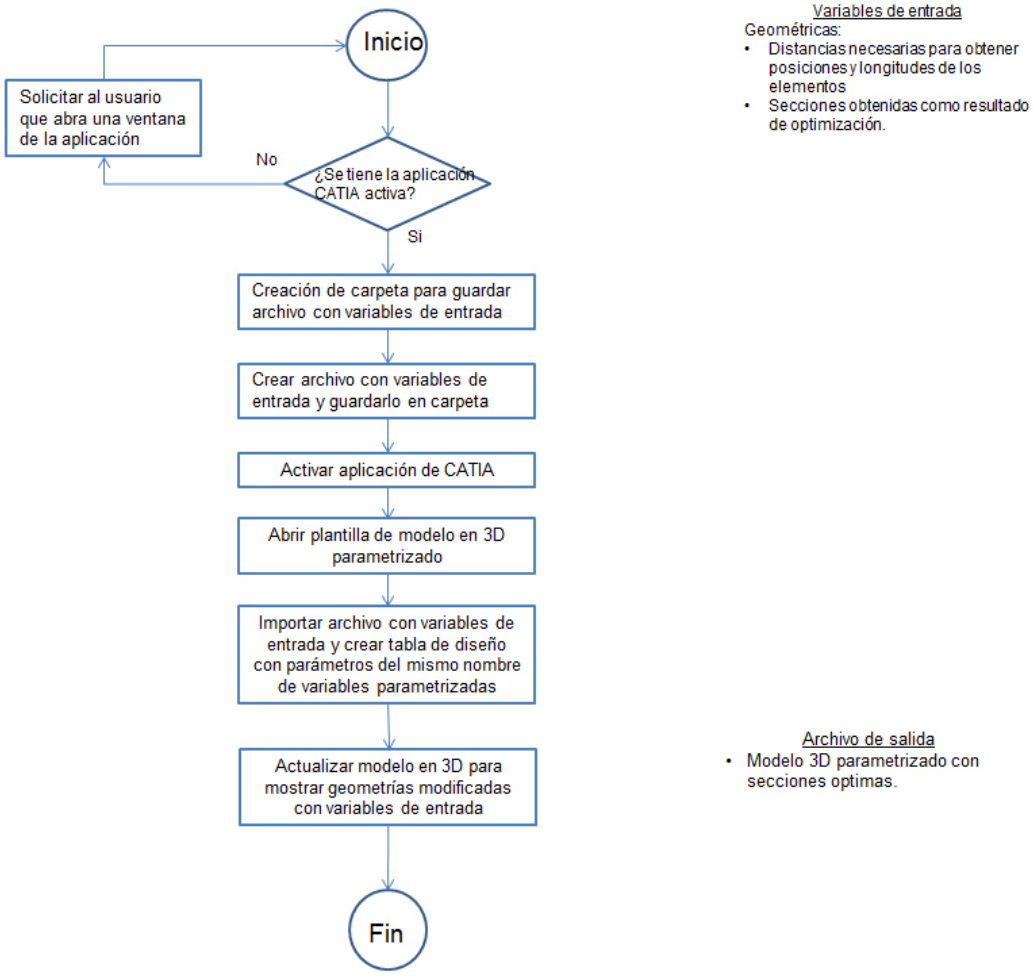


Figura 41. Diagrama de flujo de exportadores a CATIA.

8.6.4 INTERFACE CON EL USUARIO

Ya teniendo la programación se procedió a establecer una interface con el usuario amigable para que los diseñadores pudieran hacer uso de la herramienta y agregarla a sus actividades cotidianas para diseño de chasises tipo escalera de la empresa.

Para esto se creó una plantilla en Excel para los datos de entrada como se muestra en la figura 42.

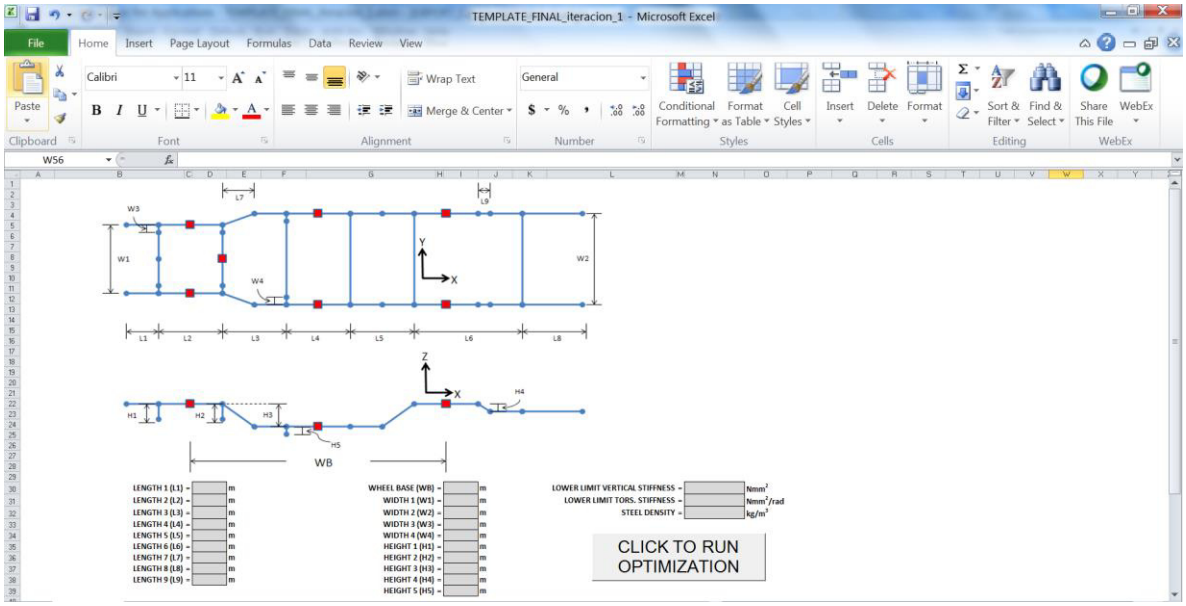


Figura 42. Interface de datos de entrada.

En donde el usuario alimenta los datos que se le solicitan para poder solicitar una optimización, ya teniendo los datos completos procede a dar click al único botón que se muestra en la imagen el cual es para correr la optimización con los datos de entrada dados.

Al dar inicio a la optimización se muestra un progreso de la optimización al usuario con la siguiente barra de progreso (figura 43):

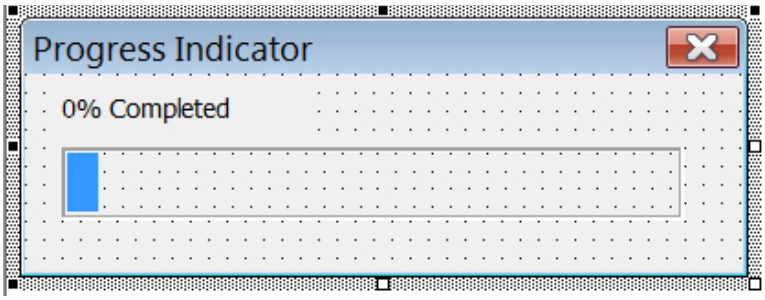


Figura 43. Barra de progreso de la optimización.

Posteriormente al completar el proceso de optimización se muestra una tabla con los resultados como se muestra a continuación (figura 44):

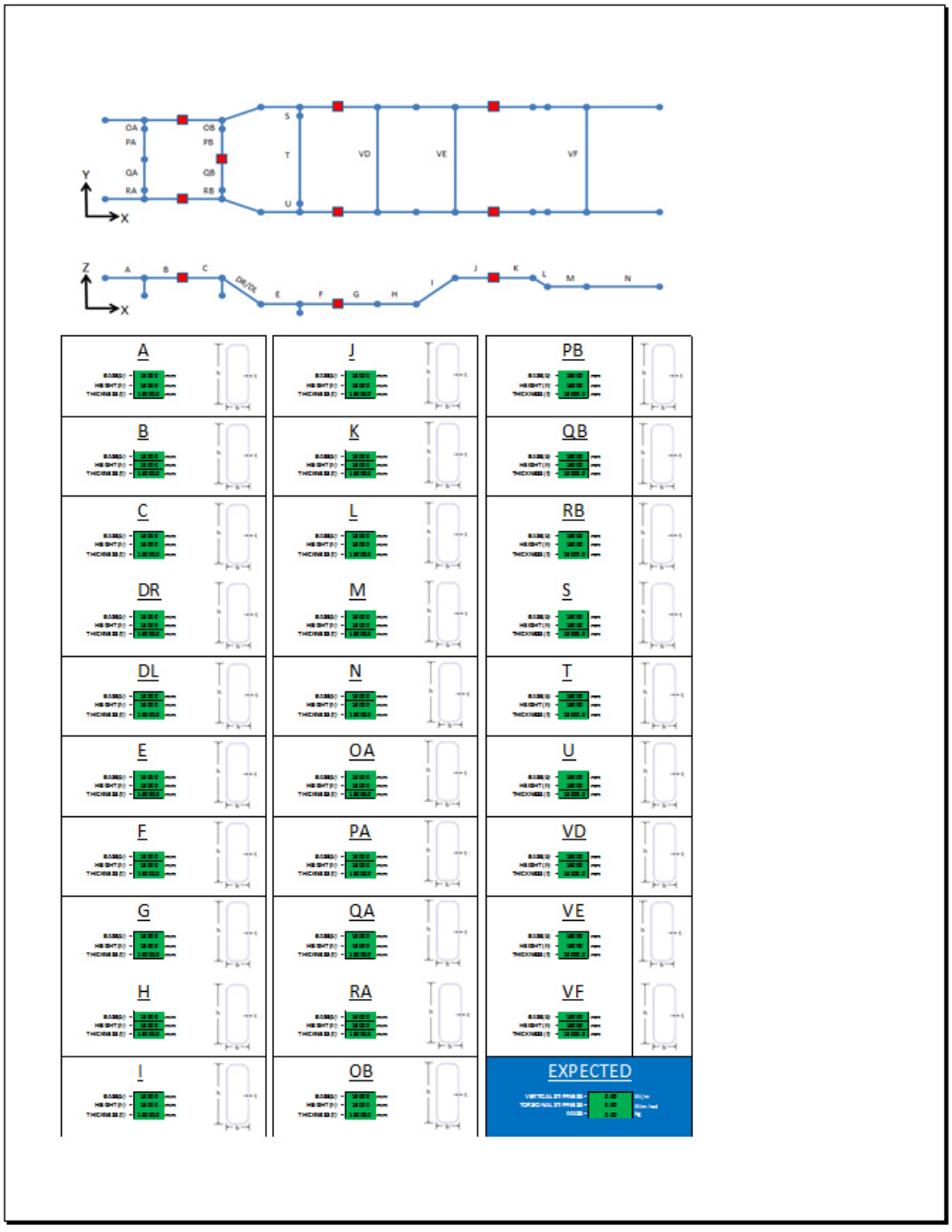


Figura 44. Tabla de resultados de optimización.

Además en la parte inferior de la hoja de resultados se encuentran los 2 botones que son utilizados para la exportación a CATIA (figura 45).

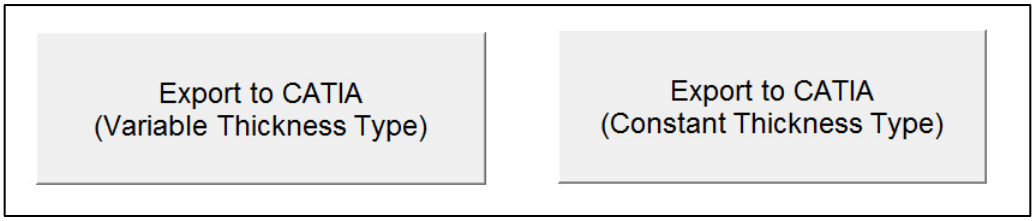


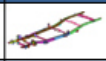
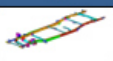
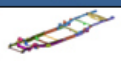
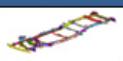
Figura 45. Botones de exportación a CATIA.

Esto facilita al usuario poder hacer uso de la herramienta y poder obtener intuitivamente utilizarlo para el beneficio que cada diseñador prefiera al tener las opciones de exportación o bien teniendo la tabla de resultados.

8.7 Estudio de Mercado

Por otra parte puede existir la posibilidad que no se tengan objetivos concretos de los clientes por lo que se recomienda generar un estudio de mercado para de esta manera definir los parámetros o variables de entrada y poder generar una propuesta sustentada y que de un valor agregado con respecto al mercado al que se está buscando enfocar la propuesta de diseño.

Tabla 1 Comparativa de chasises para definición de mercado.

	Plataforma	F1	F2	F3	F4	F5	F6
							
Tipo		4x4 D-Cabina	4X4 D-Cabina	4X4 D-Cabina	4X4 D-Cabina	4X4 D-Cabina	4x4 EX/L D
Volumen Anual		16,000	65,407	70,000	152,000	136,000	29,029
Tamaño	*Longitud (m)*	4.9	4.9	4.9	5.0	4.9	4.9
	Anchura(m)	1.3	1.4	1.4	1.4	1.5	1.5
	Altura(m)	0.5	0.6	0.6	0.5	0.6	0.6
Arquitectura del Larguero		C-en-C	C-en-C	C-en-C	C-en-C	C-en-C	Canal-C
Procesos							
	Estampado en caliente	0	0	0	0	0	0
	Estampado en frio	298	110	168	87	151	116
	Rolo formado	0	0	0	2	0	0
	Tubos	7	1	0	2	3	0
	Misceláneos	76	66	127	61	58	239
Total de Partes		**395**	**180**	**305**	**160**	**216**	**355**
Longitud de soldadura (m)		77.0	60.3	69.0	56.0	78.8	76.2

Para poder definir el estudio de mercado es necesario identificar los productos o bien en este caso de chasises que se van a comparar y para eso se tienen que comparar características físicas que los diferencian de entre los diferentes segmentos como se muestra en la Tabla 1.

Ya teniendo identificados los chasises a comparar se pueden identificar características mecánicas que son características secundarias de las camionetas que pueden confirmar o bien diferenciar los chasises de entre los diferentes segmentos, como se muestra en la Tabla 2.

Tabla 2. Comparativa de características de los vehículos a analizar para estudio de mercado.

	F1	F2	F3	F4	F5	F6
Base de Ruedas (mm)	3150	3220	3085	3200	3095	3239
GVW (Kg)	3250	3200	2835	2268	3040	2223
Carga (Kg)	1350	1255	1045	567	1303	612
Peso Vehicular (vacío)(Kg)	1900	1945	1790	1701	1737	1611
Tipo de Terreno	60-70% todo terreno 30-40% carretera	60-70% Carretera 30-40% todo terreno	-	-	60-70% carretera 30-40% todo terreno	-

Finalmente se procede a obtener características exclusivas de los chasises con respecto al rendimiento que en este caso nos interesa obtener las rigideces como datos principales (Tabla 3).

Tabla 3. Comparativa de características de rendimiento entre los chasises de los vehículos seleccionados.

	Plataforma	F1	F2	F3	F4	F5	F6
							
Rigidez	Torsional (KN-m/grado)	1.828	3.286	2.586	2.252	4.04	.916
	Flexión Vertical (KN/mm)	2.13	2.618	2.228	1.92	3.460	2.59
	Flexión Lateral(KN/mm)	0.152	0.296	0.285	0.180	0.298	0.092
NVH	Modo Torsional(Hz)	18.2	26.20	23.60	22.80	20.30	12.2
	Modo de flexión Vertical(Hz)	20.7	33.40	31.20	27.50	25.00	30.4
	Modo de Flexión Lateral(Hz)	25.7	39.60	36.40	33.30	37.50	33.1

Entendiendo las diferencias de rendimiento de los chasises dentro del mercado en que se está buscando realizar una propuesta se procede a definir objetivos específicos ya sean superar al resto de la competencia o bien colocar en dentro de competencia o bien lo que se esté buscando con dicho cliente en específico y esto facilita tener los 2 datos faltantes para poder correr la optimización que sin las rigideces tanto vertical como torsional mínimas que se busca obtener con la propuesta a realizar.

CAPÍTULO 9: RESULTADOS

Haciendo uso de la metodología propuesta en este trabajo se hicieron 5 iteraciones de optimización con diferentes datos de entrada basados en las variables de la figura 46.

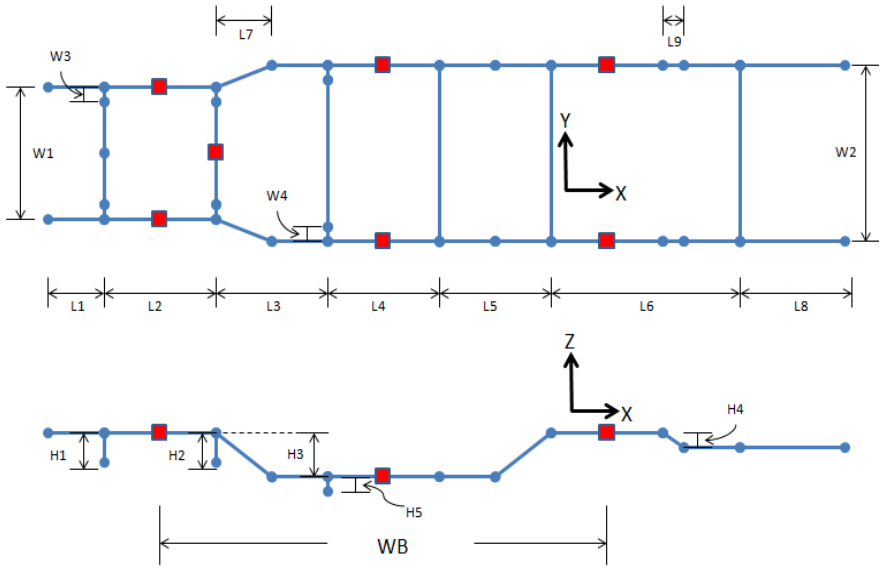


Figura 46. Dimensiones de un chasis descretizado.

Considerando la densidad del acero en común (7850 kg/m^3) se muestran en la tabla 4 los datos considerados como entrada al algoritmo propuesto:

Tabla 4. Datos de entrada para 5 iteraciones.

Dato \ Iteración	1	2	3	4	5
Longitud 1 (L1) [m]	0.70	0.50	0.70	0.70	0.65
Longitud 2 (L2) [m]	1.00	1.00	0.50	0.50	0.50
Longitud 3 (L3) [m]	1.10	1.10	0.70	0.70	0.80
Longitud 4 (L4) [m]	0.90	0.90	0.93	0.93	1.00
Longitud 5 (L5) [m]	0.70	0.70	0.95	0.95	0.95
Longitud 6 (L6) [m]	1.50	1.50	0.90	0.90	0.90
Longitud 7 (L7) [m]	0.55	0.55	0.50	0.50	0.50
Longitud 8 (L8) [m]	0.50	0.60	0.50	0.50	0.50
Longitud 9 (L9) [m]	0.20	0.20	0.25	0.25	0.25
Ancho 1 (W1) [m]	0.90	0.90	0.66	0.70	0.75
Ancho 2 (W2) [m]	1.00	1.00	0.90	0.90	1.00
Ancho 3 (W3) [m]	0.15	0.15	0.15	0.15	0.15
Ancho 4 (W4) [m]	0.10	0.10	0.10	0.10	0.10
Altura 1 (H1) [m]	0.30	0.25	0.25	0.20	0.25
Altura 2 (H2) [m]	0.30	0.25	0.25	0.20	0.25
Altura 3 (H3) [m]	0.30	0.25	0.25	0.20	0.25
Altura 4 (H4) [m]	0.20	0.25	0.13	0.15	0.15
Altura 5 (H5) [m]	0.08	0.08	0.05	0.10	0.10
Distancia entre ejes (WB) [m]	3.70	3.70	3.09	3.09	3.09
Límite inferior de rigidez vertical [$KNmm^2$]	2543.45	2207.25	1200.20	1200.00	1803.15
Límite inferior de rigidez torsional [$KNmm^2$/rad]	1240.08	1250.02	452.00	600.01	750.29

.

Teniendo dichos datos de entrada, se corrió el algoritmo de optimización y se obtuvieron los resultados basados en las variables de salida que se muestran en la figura 47.

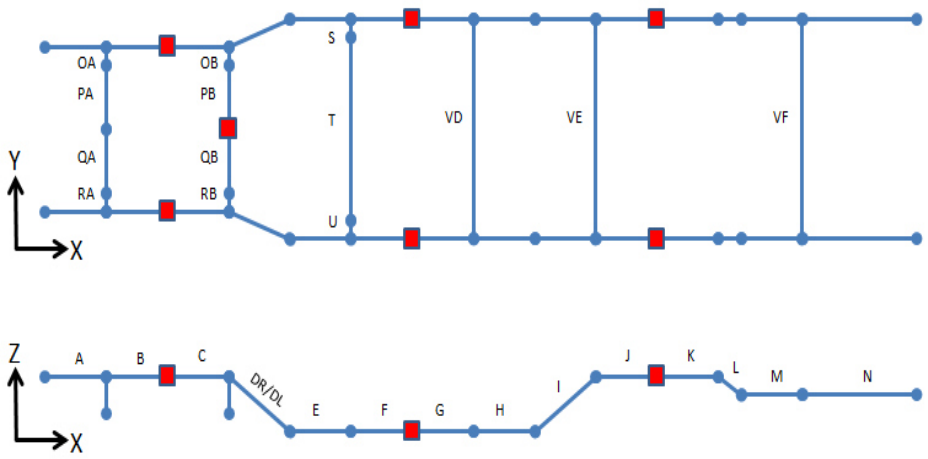


Figura 47. Elementos de salida del algoritmo.

Para las secciones se plantearon como variables de salida la altura, el ancho y el espesor del material así como se muestra en la figura 48.

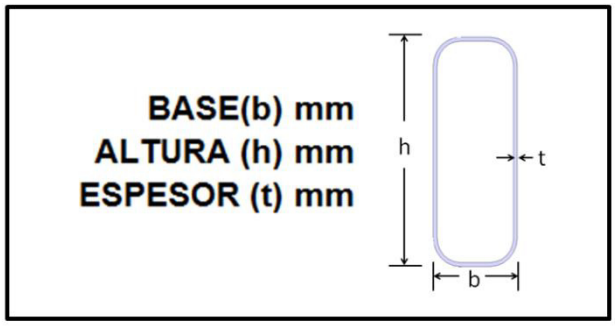


Figura 48. Variables de salida de cada elemento.

A continuación se muestran los resultados de las variables para las secciones de los largueros (tabla 5), además de mostrar las secciones que le

corresponden a los travesaños (tabla 6), esto después de correr las optimizaciones para cada iteración.

Tabla 5. Resultados de secciones para los largueros.

Resultado \ Iteración		1	2	3	4	5
A	altura (h) [mm]	183	143	122	121	176
	base (b) [mm]	74	128	96	51	77
	espesor (t) [mm]	1.7	1.7	2.4	1.5	1.8
B	altura (h) [mm]	251	248	179	239	262
	base (b) [mm]	120	84	52	59	66
	espesor (t) [mm]	1.5	1.9	2.3	2.3	2.1
C	altura (h) [mm]	253	222	103	111	192
	base (b) [mm]	108	114	119	119	140
	espesor (t) [mm]	1.6	2.4	2.6	2.6	2.0
DR	altura (h) [mm]	108	119	103	103	120
	base (b) [mm]	87	145	69	69	72
	espesor (t) [mm]	2.0	2.6	1.6	1.6	2.1
DL	altura (h) [mm]	215	248	174	174	230
	base (b) [mm]	120	56	101	113	121
	espesor (t) [mm]	1.6	2.4	1.7	1.9	1.8
E	altura (h) [mm]	166	152	139	131	213
	base (b) [mm]	133	148	60	128	105
	espesor (t) [mm]	2.1	2.0	2.7	1.6	1.6
F	altura (h) [mm]	200	121	114	242	199
	base (b) [mm]	118	116	111	86	118
	espesor (t) [mm]	2.0	2.5	2.7	1.6	2.7
G	altura (h) [mm]	197	285	180	139	154
	base (b) [mm]	143	85	80	64	51
	espesor (t) [mm]	2.3	1.5	2.0	2.6	2.9
H	altura (h) [mm]	180	182	226	115	168
	base (b) [mm]	102	134	88	64	79
	espesor (t) [mm]	2.9	2.4	2.1	2.6	1.8
I	altura (h) [mm]	203	197	114	139	138
	base (b) [mm]	136	143	56	90	66
	espesor (t) [mm]	2.5	2.8	2.3	2.1	3.0
J	altura (h) [mm]	194	158	137	137	116
	base (b) [mm]	104	76	52	132	77
	espesor (t) [mm]	2.1	2.6	2.9	1.8	2.2
K	altura (h) [mm]	182	182	113	165	254
	base (b) [mm]	116	111	67	138	66
	espesor (t) [mm]	2.9	2.3	1.7	2.6	2.5
L	altura (h) [mm]	244	238	192	128	154
	base (b) [mm]	126	104	80	89	140
	espesor (t) [mm]	2.0	2.3	1.8	2.0	1.6
M	altura (h) [mm]	169	188	103	172	285
	base (b) [mm]	97	97	69	57	95
	espesor (t) [mm]	2.9	2.9	1.8	1.8	2.4
N	altura (h) [mm]	177	177	122	294	150
	base (b) [mm]	79	84	51	102	54
	espesor (t) [mm]	1.5	1.8	2.5	1.5	1.7

Tabla 6. Resultados de secciones para los travesaños.

Resultado \ Iteración		1	2	3	4	5
OA	altura (h) [mm]	119	131	102	256	238
	base (b) [mm]	117	75	59	149	55
	espesor (t) [mm]	1.7	2.5	2.0	1.9	1.7
PA	altura (h) [mm]	113	94	69	126	51
	base (b) [mm]	134	84	110	142	122
	espesor (t) [mm]	2.5	2.8	1.7	2.3	2.5
QA	altura (h) [mm]	129	74	87	106	66
	base (b) [mm]	76	125	76	93	95
	espesor (t) [mm]	2.0	2.1	2.2	2.0	2.2
RA	altura (h) [mm]	107	216	123	194	171
	base (b) [mm]	60	71	118	85	138
	espesor (t) [mm]	2.9	1.9	1.6	2.5	2.1
OB	altura (h) [mm]	173	192	182	165	117
	base (b) [mm]	85	53	112	67	91
	espesor (t) [mm]	1.7	2.0	2.1	1.9	1.6
PB	altura (h) [mm]	102	52	66	66	77
	base (b) [mm]	64	111	54	54	70
	espesor (t) [mm]	2.0	2.9	2.0	2.1	1.5
QB	altura (h) [mm]	148	56	53	53	69
	base (b) [mm]	59	74	54	54	131
	espesor (t) [mm]	1.5	2.0	2.6	2.8	1.8
RB	altura (h) [mm]	83	71	96	82	63
	base (b) [mm]	77	134	84	64	88
	espesor (t) [mm]	2.3	2.4	2.3	1.5	1.9
S	altura (h) [mm]	234	116	208	150	201
	base (b) [mm]	105	62	98	105	132
	espesor (t) [mm]	2.1	2.3	2.0	2.1	1.9
T	altura (h) [mm]	183	188	133	185	108
	base (b) [mm]	53	120	65	65	52
	espesor (t) [mm]	2.2	2.1	1.6	1.6	2.0
U	altura (h) [mm]	114	204	124	123	245
	base (b) [mm]	121	87	144	145	82
	espesor (t) [mm]	2.7	1.8	2.9	2.9	1.8
VD	altura (h) [mm]	139	76	62	66	91
	base (b) [mm]	79	75	69	108	104
	espesor (t) [mm]	2.5	1.7	1.8	1.8	1.9
VE	altura (h) [mm]	60	117	94	94	70
	base (b) [mm]	86	147	56	56	138
	espesor (t) [mm]	2.1	2.8	2.9	2.9	2.2
VF	altura (h) [mm]	109	85	101	79	89
	base (b) [mm]	54	91	54	116	135
	espesor (t) [mm]	1.5	1.5	1.5	1.9	1.8

A su vez se exportaron las geometrías definidas a un software de diseño asistido por computadora (CATIA V5); lo cual facilita la visualización de los resultados de la optimización, asimismo, fue de utilidad para exportar los

resultados a un software de elementos finitos (Hypermesh) y son mostrados en la figura 49.

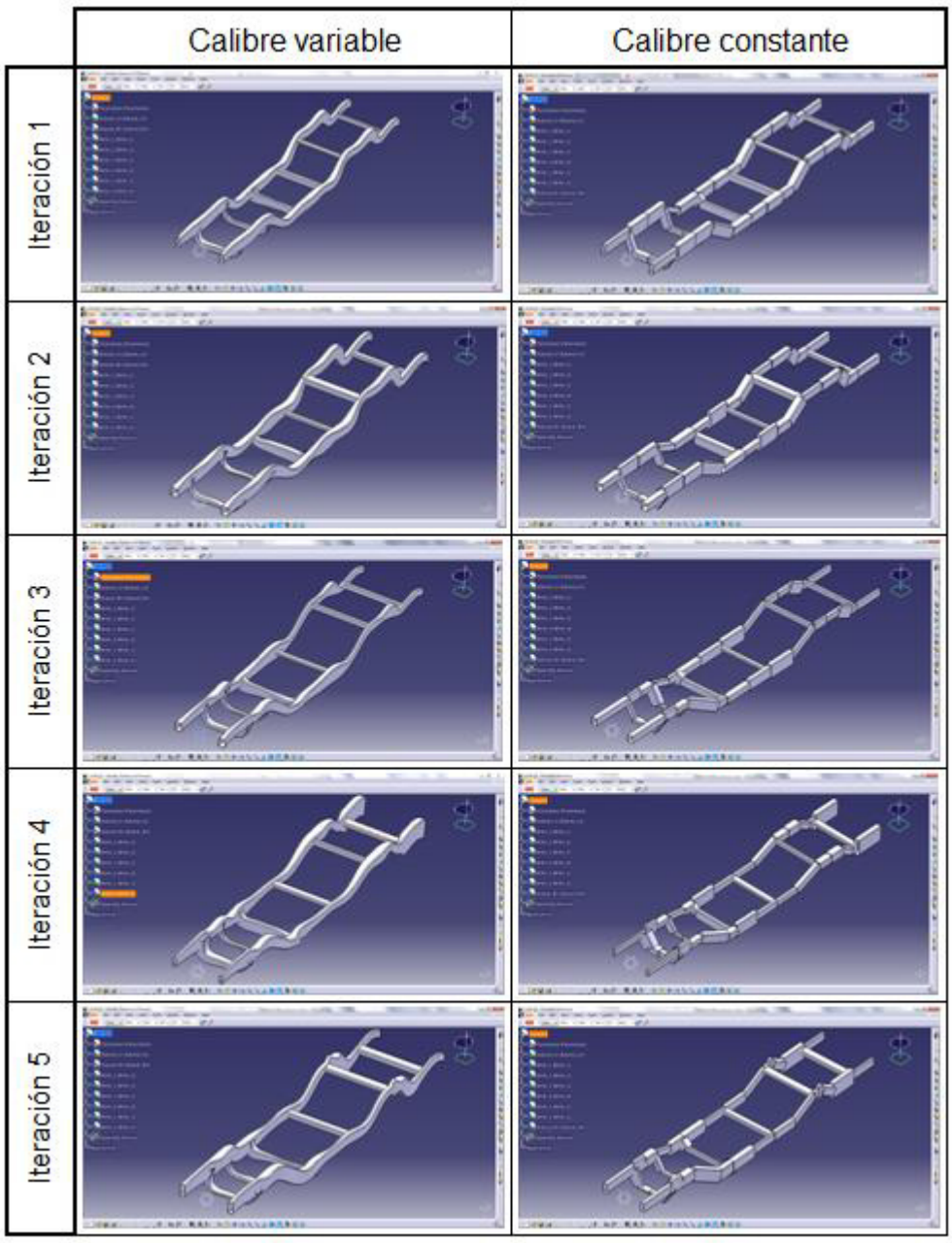


Figura 49. Resultados de exportaciones a CATIA de las 5 iteraciones.

Por otra parte se obtuvieron los resultados de las rigideces y de la masa estimada para el chasis considerando los resultados de las secciones anteriormente mostradas; para validar los resultados se hizo uso de las geometrías exportadas de sección constante para cada una de las iteraciones con las secciones que fueron marcadas por el algoritmo y se obtuvieron los resultados por el método de elementos finitos utilizando los parámetros que se muestran en la tabla 7 para la definición de la malla.

Tabla 7. Parámetros utilizados para el cálculo con elementos finitos.

Parámetro	Descripción
Modelo CAE:	CAD Superficie media
Tipo de malla:	Cuadrada (Quads)
Tamaño de malla:	6mm (estandar)
Conectores:	Elementos rígidos (RBE2)
Material:	Acero estándar

Por otra parte se colocaron las fuerzas y restricciones en cada caso para obtener los resultados, para el caso vertical (figura 50) las restricciones en los puntos de agarre de la suspensión delantera en las 3 direcciones y para la parte trasera solamente en 2 direcciones (“y” y “z”), para el caso de la fuerza se aplicó una en cada larguero en la parte media entre las 2 sujeciones de 1000N cada una.

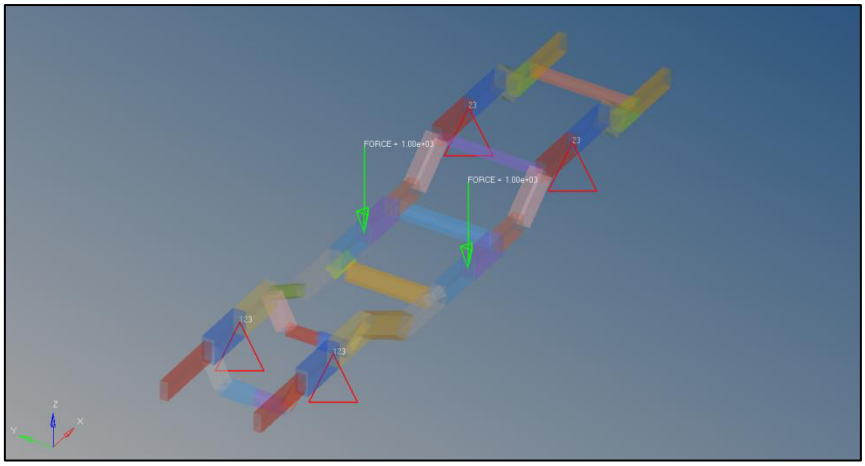


Figura 50. Restricciones y fuerzas para el análisis de rigidez vertical.

A su vez para el caso torsional (figura 51) las restricciones en los puntos de agarre de la suspensión trasera en las 3 direcciones y para la parte frontal solamente en 1 dirección (“z”) en el centro del segundo travesaño, para el caso de las fuerza se aplicaron 2 fuerzas encontradas en cada larguero en la suspensión frontal de 1000N cada una.

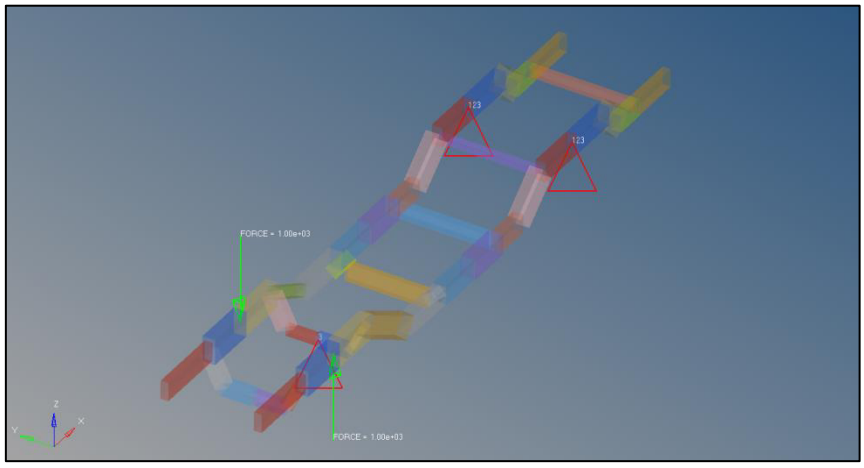


Figura 51. Restricciones y fuerzas para el análisis de rigidez torsional.

A continuación se muestran los desplazamientos que arrojó como resultado cada uno de los chasises evaluados por el método de elementos finitos (figuras 52 a 61), dichos desplazamientos se utilizan para el cálculo de rigideces utilizando las ecuaciones 37 y 38.

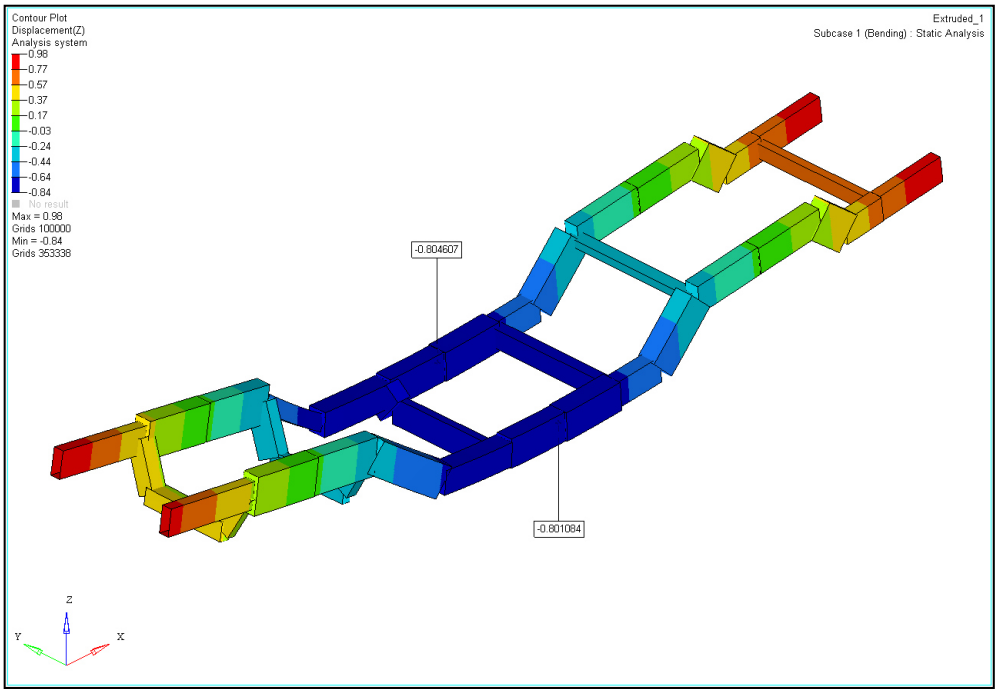


Figura 52. Desplazamientos para el caso vertical de la iteración 1 por el método de elementos finitos.

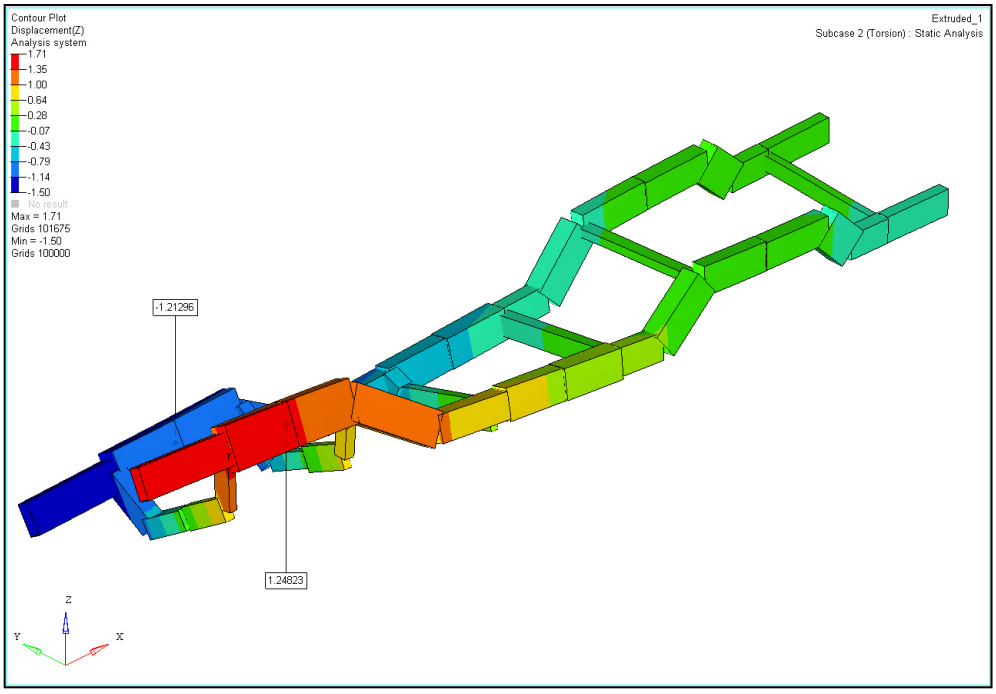


Figura 53. Desplazamientos para el caso torsional de la iteración 1 por el método de elementos finitos.

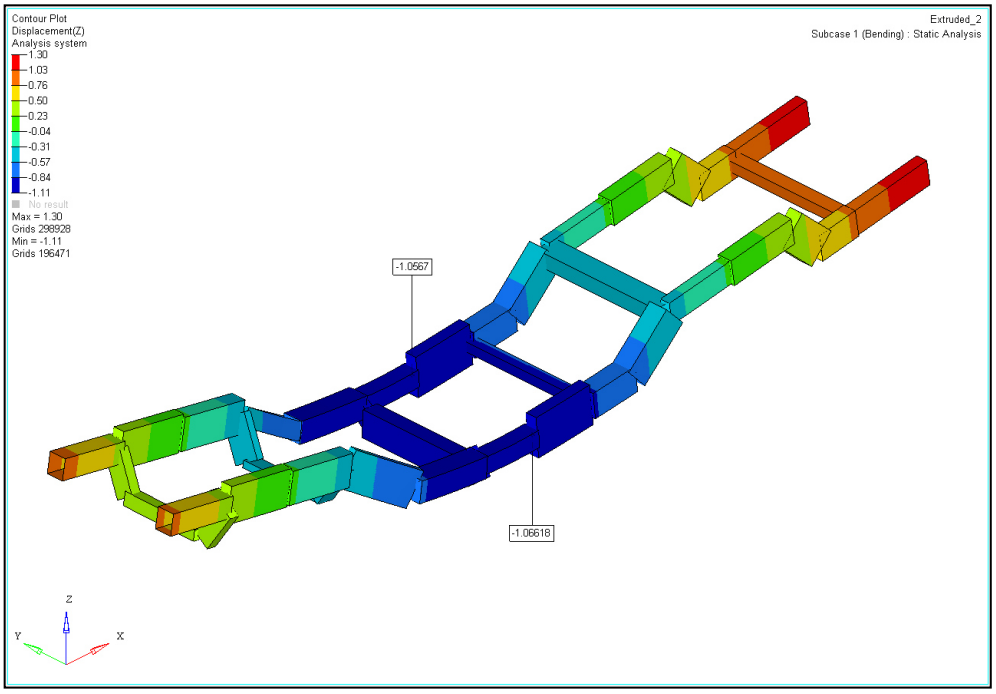


Figura 54. Desplazamientos para el caso vertical de la iteración 2 por el método de elementos finitos.

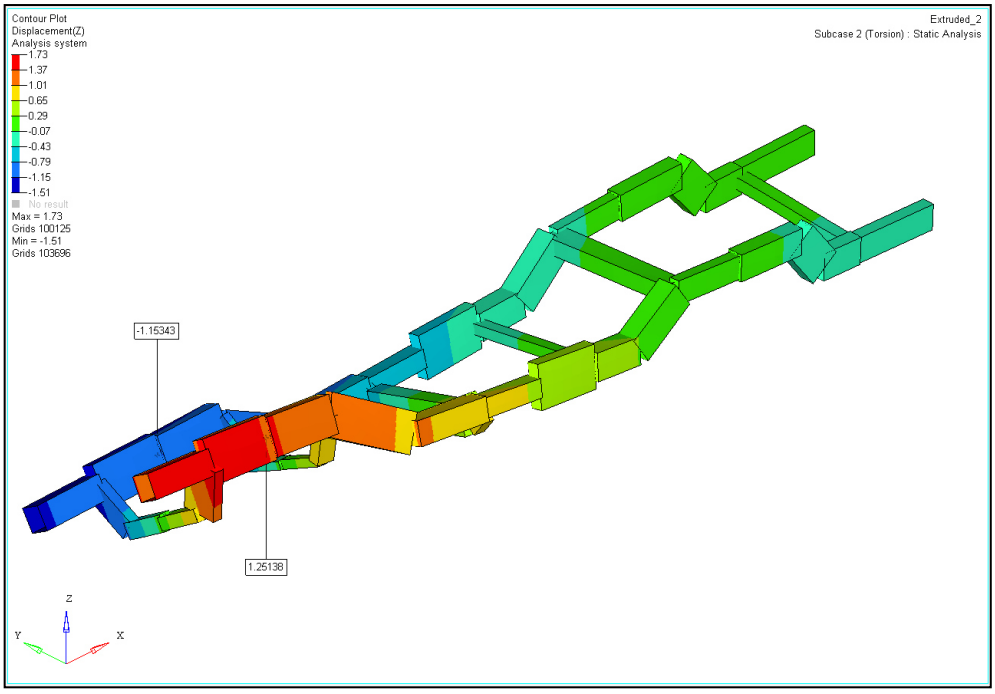


Figura 55. Desplazamientos para el caso torsional de la iteración 2 por el método de elementos finitos.

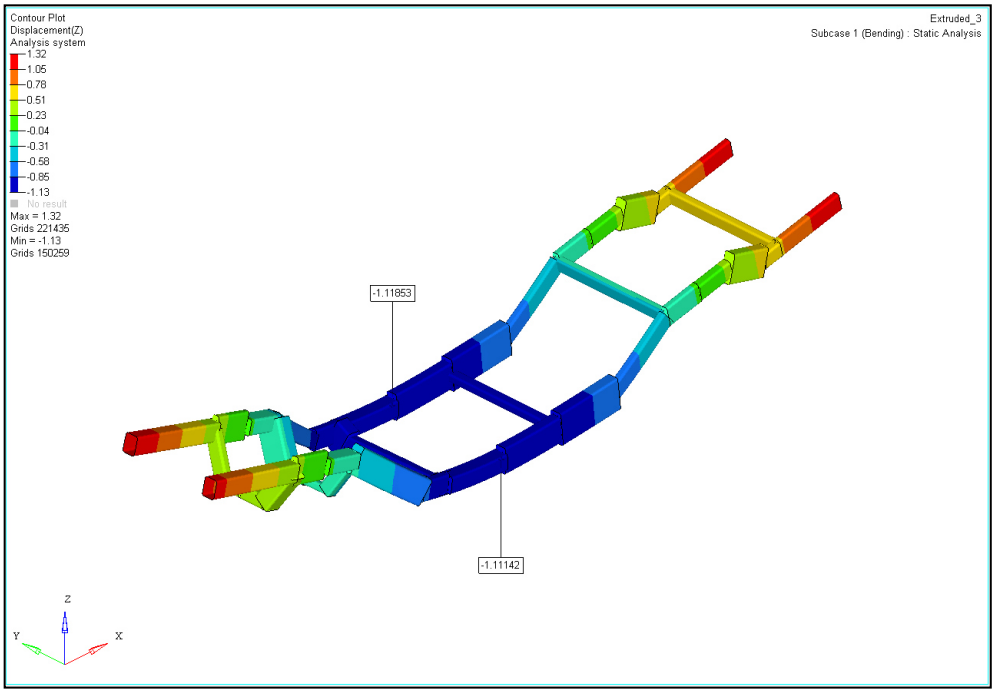


Figura 56. Desplazamientos para el caso vertical de la iteración 3 por el método de elementos finitos.

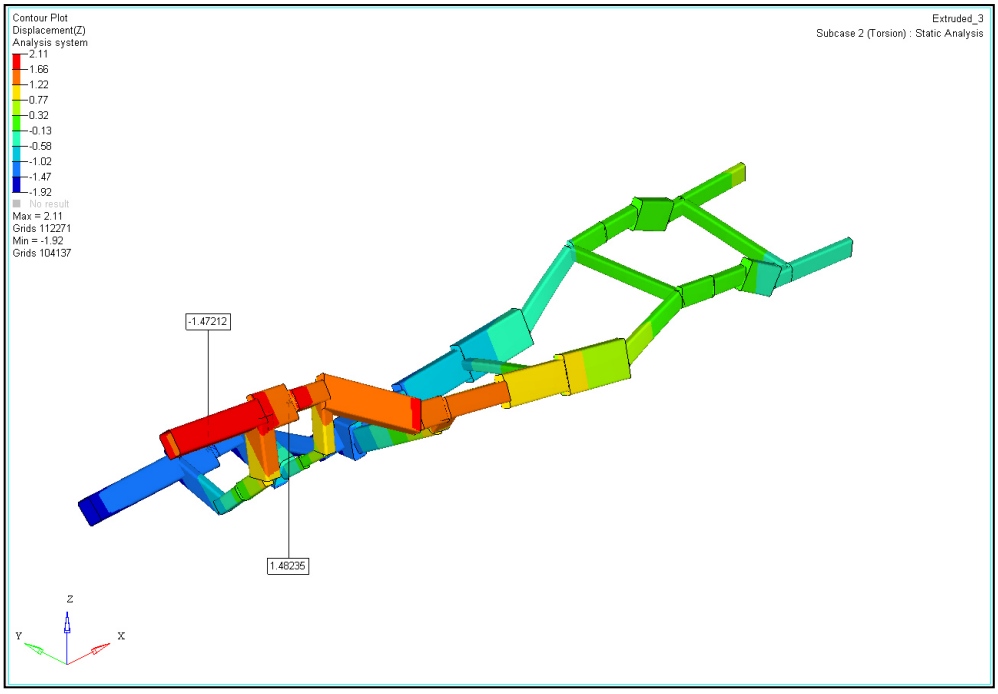


Figura 57. Desplazamientos para el caso torsional de la iteración 3 por el método de elementos finitos.

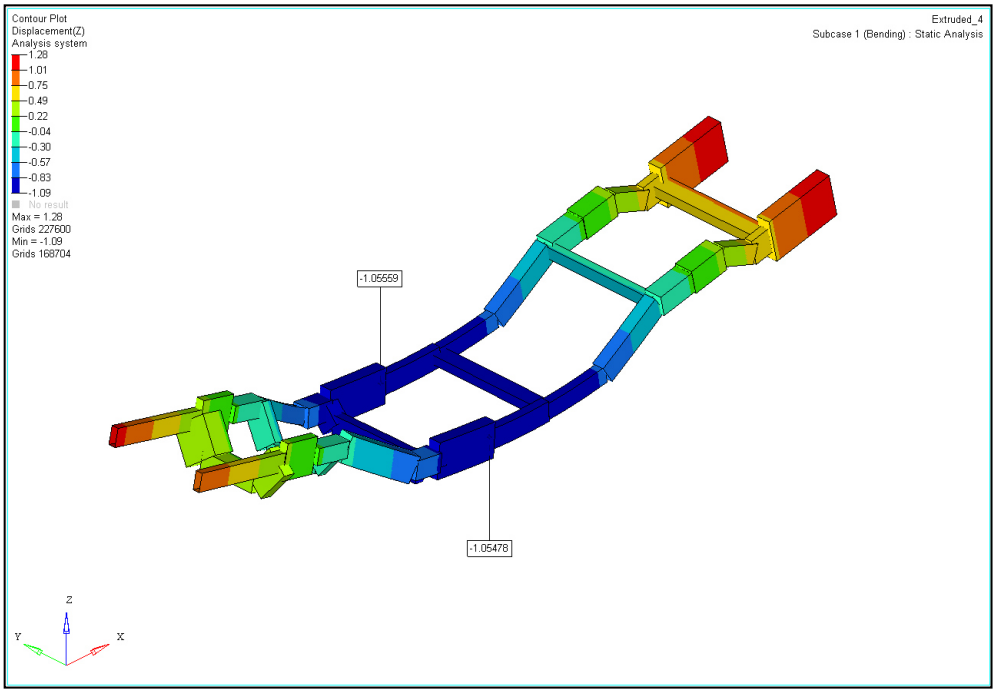


Figura 58. Desplazamientos para el caso vertical de la iteración 4 por el método de elementos finitos.

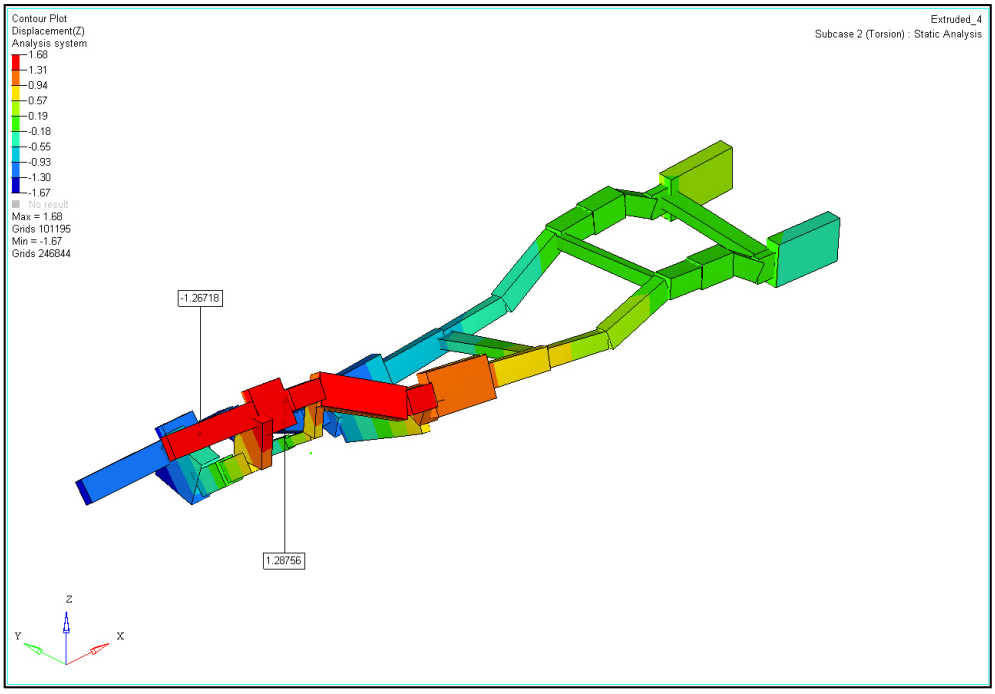


Figura 59. Desplazamientos para el caso torsional de la iteración 4 por el método de elementos finitos.

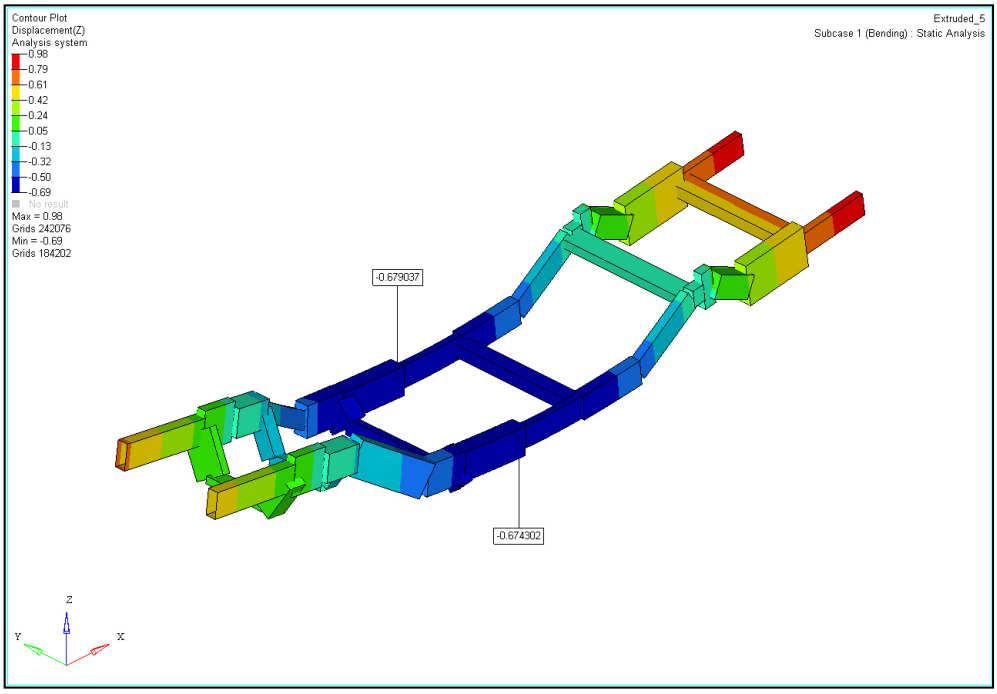


Figura 60. Desplazamientos para el caso vertical de la iteración 5 por el método de elementos finitos.

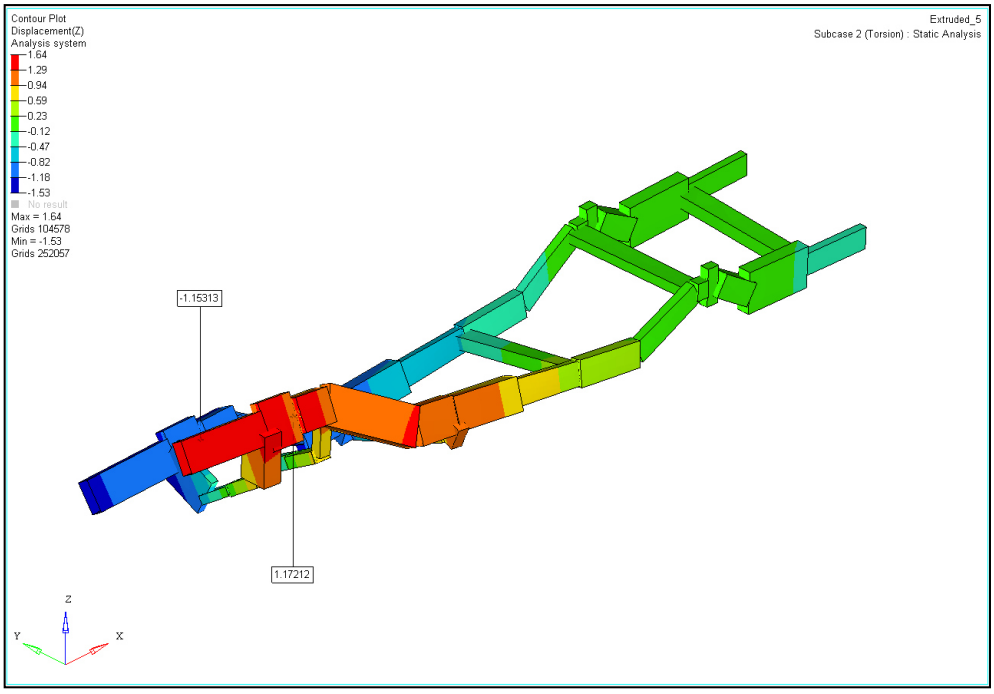


Figura 61. Desplazamientos para el caso torsional de la iteración 5 por el método de elementos finitos.

Al finalizar se hizo una comparativa entre los resultados (tabla 8) según el método propuesto y el método de elementos finitos para así obtener una variación entre los resultados promedio de -4.4% en rigidez vertical y -1.2% en rigidez torsional lo cual es aceptable.

Tabla 8. Resultados de rigideces y masa para las 5 iteraciones.

Resultado \ Iteración		1	2	3	4	5
Rigidez vertical [Nmm]	Metodología	2543.45	2217.35	1200.20	1204.94	1818.97
	Elementos Finitos	2628.83	1988.38	1097.15	1159.38	1807.91
	% Variación	3.2%	-11.5%	-9.4%	-3.9%	-0.6%
Rigidez torsional [Nmm/grado]	Metodología	1250.12	1250.02	460.12	600.01	750.30
	Elementos Finitos	1217.7	1246.25	454.85	591.70	746.29
	% Variación	-2.7%	-0.3%	-1.2%	-1.4%	-0.5%
Masa [kg]		172.54	179.11	110.26	115.11	130.94

CAPÍTULO 10: CONCLUSIONES Y RECOMENDACIONES

El propósito de esta tesis fue demostrar que es posible reducir la cantidad de iteraciones entre los departamentos de diseño y de elementos finitos de la empresa, el cual se logró debido a que haciendo uso de la metodología propuesta en el presente trabajo es posible tener mayor sensibilidad acerca de las geometrías por sección que facilitan la obtención de los objetivos del comportamiento estructural esperado al momento de diseñar un chasis.

A su vez, se cumplió con el objetivo general al desarrollar una metodología que permite a los diseñadores contar con una base científica al momento de generar propuestas para chasises de camionetas, facilitando el aprovechamiento del personal que forma parte del área de diseño al direccionar su trabajo en lugar de confiar solamente en las propias competencias de cada diseñador.

Por otra parte se cumplieron con los objetivos específicos establecidos al inicio de este proyecto al desarrollar un algoritmo que permite definir geometrías óptimas que a su vez puedan ser de utilidad como base para el diseño de un chasis de camioneta, asimismo, se desarrolló un estudio de mercado como

demostración de la posibilidad para definir el desempeño de un chasis sin necesidad de que sea definido por un cliente.

Adicionalmente se diseñó una estructura utilizando la metodología que se establece en este trabajo que fue de gran utilidad para que a través del método de elementos finitos se pudiera comprobar la factibilidad de hacer uso de la herramienta propuesta y de esta forma se completaron los objetivos planteados al inicio de este trabajo.

Para continuar con el desarrollo de esta metodología se recomienda asegurar a través de análisis modal que la estructura sea equivalente a la que se usa como base, esto sería pidiéndole los primeros tres modos al usuario como entrada y calculando los modos de la estructura como parte de la función objetivo.

Posteriormente se puede aplicar opciones de multi-material para facilitar al usuario el cálculo para chasises que contengan algún otro tipo de material como aleaciones de aluminio o magnesio.

Además se pueden hacer mejoras a la programación de la optimización para mejorar los tiempos de respuesta, para hacer el menor uso posible de los recursos tecnológicos, en este caso el uso de los procesadores de la computadora del diseñador.

Por otra parte se podría mejorar el resultado si se agregan variables que correspondan a las uniones, es decir, que se tomen en cuenta los diferentes tipos de uniones y sus respectivos comportamientos.

BIBLIOGRAFÍA

1. (NHTSA), National Highway Traffic Safety Administration. (2012). *Environmental Protection Agency.* US Government Information.

2. Aguilar, F. (2004). *Analisis Matricial de Estructuras* (3era ed.). Sangolquí: ESPE.

3. Andaluz, A. (2005). *Universidad Carlos III de Madrid*. Recuperado el 26 de 09 de 2015, de http://www.it.uc3m.es/~jvillena/irc/descarga.htm?url=practica/estudio/aeag.pdf

4. ArcelorMittal. (s.f.). *ArcelorMittal*. Recuperado el 06 de 10 de 2015, de http://sections.arcelormittal.com/fileadmin/redaction/2-Products_Services/1_Product_Range/ES-EN-IT/Coefficients.pdf

5. *Aulatecnología*. (s.f.). Recuperado el 28 de Septiembre de 2015, de Aulateconlogía: http://www.aulatecnologia.com/BACHILLERATO/1_bg/APUNTES/materiales/propiedadesmateriales.htm

6. Blanco, J. (2012). *Análisis Estático de Estructuras por el Método Matricial.* Málaga: Servicio de Publicaciones e Intercambio Científico de la Universidad de Málaga.

7. Calvo, J. (1997). *Mecánica del automóvil actual.* Reverte.

8. Carbajal, Y. (2010). Sector automotriz: reestructuración tecnológica y reconfiguración del mercado mundial. *Paradigma económico*, 24-52.

9. Charbonneau, P. (s.f.). *High Altitude Observatory*. Recuperado el 15 de Enero de 2015, de http://www.hao.ucar.edu/modeling/pikaia/pikaia.php#sec2

10. Courant, R. (1943). Variational Methods for the Solution of Problems of Equilibrium and Vibrations. *Bulletin of American Mathematical*.

11. Fermin, R. (19 de Marzo de 2009). *Quantcode*. Recuperado el 15 de Enero de 2015, de Directory of Open Source Software for Quantitative Finance & Trading: http://www.quantcode.com/modules/newbb/viewtopic.php?topic_id=211&forum=5#forumpost426

12. Gere, J. (2009). *Mecánica de Materiales* (Séptima ed.). (J. León, Trad.) México, D.F.: Cengage Learning.

13. Jacj. (5 de Mayo de 2005). *en.Wikipedia*. Obtenido de Wikipedia: https://upload.wikimedia.org/wikipedia/commons/7/78/Simplex_description.png

14. Jiménez, A. (2004). *Análisis y Optimización con Interacción de Dummy, de la Carrocería del Automóvil.* Puebla: Universidad de las Américas Puebla.

15. Kiemele, M. (27 de Febrero de 2003). *http://www.slideshare.net*. Recuperado el 27 de Septiembre de 2015, de Slideshare: http://www.slideshare.net/Sixsigmacentral/using-the-design-for-six-sigma-dfss-approach-to-design

16. Lagarias, J. (1998). Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions. *SIAM J. Optim*.

17. López, R. (1993). *Programación Lineal y Decisiones Económicas.* Caracas: Publicaciones UCAB.

18. Masiá, J. (2003). *Investigación de la Caracterización Experimental de las Uniones entre Componentes de Bastidores de Vehículos Pesados.* Alcoy: Universidad Politécnica de Valencia.

19. Mira, J. (1995). *Aspectos básicos de la Inteligencia Artificial.* Barcelona: Sanz y Torres.

20. Muñoz, B. (2014). *Introducción al Método de los Elementos Finitos.* Madrid: UNED.

21. Murty, K. (1983). *Linear Programming.* Michigan: Universidad de Michigan.

22. Nilsson, N. (2000). *Inteligencia Artificial: Una Nueva Síntesis.* Madrid: McGraw-Hill.

23. Oden, J. (1972). *Some Aspects of Recent Contributions to the Mathematical Theory of Finite Elements.* Huntsville: University of Alabama Press.

24. París, F. (2006). *Cálculo Matricial de Estructuras.* Oviedo: Ediciones de la Universidad de Oviedo.

25. Przemieniecki, J. (1968). *Theory of Matrix Structural Analysis.* Nueva York: Mc GRaw-Hill.

26. RAE. (2001). *Diccionario de la Lengua Española*. Obtenido de Real Academia Española: http://www.rae.es/recursos/diccionarios/drae

27. *RAM Trucks*. (s.f.). Recuperado el 27 de Septiembre de 2015, de Test Mideast Ram Trucks: http://test.mideast.ramtrucks.com/deprecated/en/groundbreakers/images/hd/frame_anim/RAM_HD_3500_LARAMIE_frameAnim_v02.0060.png

28. Ramos, A. (2010). *Modelos Matemáticos de Optimización.* Madrid: Universidad Pontífica de Comillas.

29. Rich, E. (1995). *Inteligencia Artificial: Un Enfoque Moderno.* Mexico: Prentice-Hall.

30. Sanmartín Quiroga, A. (2001). *Cálculo Matricial de Estructuras.* Colegio ICCP.

31. Sawyer, C. A. (2003). Framing The Question. *Automotive Design & Production*.

32. Sturgeon, T. (2009). Globalisation of the automotive industry: main features and trends. *International Journal of Technological Learning, Innovation and Development*, 7-24.

33. Turner, M. (1956). Stiffness and Deflection Analysis of Complex Structures. *Journal of the Aeronautical Sciences*.

34. Valencia, L. (2010). *Integracion Semi-analítica de la Matriz de Rigidez para un Elemento Lagrangiano Cuadrilátero de 9 Nodos Usando Matemática Simbólica.* Barquisimeto: UNIVERSIDAD CENTROCCIDENTAL“LISANDRO ALVARADO”.

35. Wakeham, K. (2009). *Introduction To Chassis Design.* St. John's: Memorial University of Newfoundland And Labrador.

36. *wikimedia*. (s.f.). Recuperado el 28 de Septiembre de 2015, de wikimedia: https://commons.wikimedia.org/wiki/File:Stress-strain1-es.svg

37. Zienkiewicz, O. (1967). *The Finite Element Method in Structural and Continuum Mechanics.* Londres: Mc Graw-Hill.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ANEXO: MÓDULOS DE VISUAL BASIC

Módulo: EXPORT_CATIA_EXTRUDED

```
Sub CATIA_extruded()



Dim Response As VbMsgBoxResult
Response = MsgBox("Please Open CATIA Application", vbQuestion + vbYesNo)
If Response = vbNo Then

   MsgBox "You need to have CATIA to proceed", vbInformation

Else

   Application.ScreenUpdating = False
   'making visible the export worksheet
   Worksheets("EXPORT").Visible = -1

   'creating folder to save worksheet

   For variablename = 0 To 1000

        FolderName = "\CATIA_variable" & variablename
        If FileFolderExists(ThisWorkbook.Path & FolderName) Then

          GoTo nextfilename:

        Else

        Set ab = CreateObject("Scripting.FileSystemObject")
```

```
            ab.CreateFolder (ThisWorkbook.Path & FolderName & "/")

            GoTo continue:

          End If

nextfilename:
   Next

continue:

'creating workbook to be exported to CATIA
   Application.ScreenUpdating = False
   Application.DisplayAlerts = False

   Sheets("EXPORT").Range("A1:DB2").Copy

   Flname = "\Variable_input"

      Set NewWkbk = Workbooks.Add

      Sheets("Sheet1").Range("A1:DB2").PasteSpecial xlPasteValues

      NewWkbk.SaveAs ThisWorkbook.Path & FolderName & Flname


      ActiveWorkbook.Close

   'hiding the export worksheet
   Worksheets("EXPORT").Visible = 0

'OPENING CATIA
```

```
On Error Resume Next

Set ObjCATIA = GetObject(, "CATIA.Application")

If Err.Number <> 0 Then  ' this part of code is commented because is supposed that you have already CATIA opened

Set ObjCATIA = CreateObject("CATIA.Application")

CATIA.Visible = True

Err.Clear


End If

On Error GoTo 0


'OPENING TEMPLATE

ObjCATIA.Visible = True

ObjCATIA.DisplayFileAlerts = True

Set objProd = ObjCATIA.Documents.Open(ThisWorkbook.Path & "\Final_CATIA_Template_Extruded\Extruded.CATProduct")


'Importing excel to CATIA


Dim productDocument1 As ProductDocument

Set productDocument1 = objProd


Dim product1 As Product

Set product1 = productDocument1.Product


Dim part1 As Part

Set part1 = ObjCATIA.Documents.Item("Extruded_Parameters.CATPart").Part


Dim relations2 As Relations

Set relations2 = part1.Relations


Dim designTable1 As DesignTable

Set designTable1 = relations2.CreateDesignTable("DesignTable.1", "", False, ThisWorkbook.Path & FolderName & Flname & ".xlsx")
```

```
'Dim parameters1 As Parameters
Set parameters1 = part1.Parameters

Dim length1 As length
Set length1 = parameters1.Item("L1")

designTable1.AddAssociation length1, "L1"

'Dim parameters2 As Parameters
Set parameters2 = part1.Parameters

Dim length2 As length
Set length2 = parameters2.Item("L2")

designTable1.AddAssociation length2, "L2"

'Dim parameters3 As Parameters
Set parameters3 = part1.Parameters

Dim length3 As length
Set length3 = parameters3.Item("L3")

designTable1.AddAssociation length3, "L3"

'Dim parameters4 As Parameters
Set parameters4 = part1.Parameters

Dim length4 As length
Set length4 = parameters4.Item("L4")

designTable1.AddAssociation length4, "L4"
```

```
'Dim parameters5 As Parameters
Set parameters5 = part1.Parameters

Dim length5 As length
Set length5 = parameters5.Item("L5")

designTable1.AddAssociation length5, "L5"

'Dim parameters6 As Parameters
Set parameters6 = part1.Parameters

Dim length6 As length
Set length6 = parameters6.Item("L6")

designTable1.AddAssociation length6, "L6"

'Dim parameters7 As Parameters
Set parameters7 = part1.Parameters

Dim length7 As length
Set length7 = parameters7.Item("L7")

designTable1.AddAssociation length7, "L7"

'Dim parameters8 As Parameters
Set parameters8 = part1.Parameters

Dim length8 As length
Set length8 = parameters8.Item("L8")

designTable1.AddAssociation length8, "L8"
```

```
'Dim parameters9 As Parameters
Set parameters9 = part1.Parameters

Dim length9 As length
Set length9 = parameters9.Item("L9")

designTable1.AddAssociation length9, "L9"

'Dim parameters10 As Parameters
Set parameters10 = part1.Parameters

Dim length10 As length
Set length10 = parameters10.Item("W1")

designTable1.AddAssociation length10, "W1"

'Dim parameters11 As Parameters
Set parameters11 = part1.Parameters

Dim length11 As length
Set length11 = parameters11.Item("W2")

designTable1.AddAssociation length11, "W2"

'Dim parameters12 As Parameters
Set parameters12 = part1.Parameters

Dim length12 As length
Set length12 = parameters12.Item("W3")

designTable1.AddAssociation length12, "W3"
```

```
'Dim parameters13 As Parameters
Set parameters13 = part1.Parameters

Dim length13 As length
Set length13 = parameters13.Item("W4")

designTable1.AddAssociation length13, "W4"

'Dim parameters14 As Parameters
Set parameters14 = part1.Parameters

Dim length14 As length
Set length14 = parameters14.Item("H1")

designTable1.AddAssociation length14, "H1"

'Dim parameters15 As Parameters
Set parameters15 = part1.Parameters

Dim length15 As length
Set length15 = parameters15.Item("H2")

designTable1.AddAssociation length15, "H2"

'Dim parameters16 As Parameters
Set parameters16 = part1.Parameters

Dim length16 As length
Set length16 = parameters16.Item("H3")

designTable1.AddAssociation length16, "H3"
```

```
'Dim parameters17 As Parameters
Set parameters17 = part1.Parameters

Dim length17 As length
Set length17 = parameters17.Item("H4")

designTable1.AddAssociation length17, "H4"

'Dim parameters18 As Parameters
Set parameters18 = part1.Parameters

Dim length18 As length
Set length18 = parameters18.Item("H5")

designTable1.AddAssociation length18, "H5"

'Dim parameters19 As Parameters
Set parameters19 = part1.Parameters

Dim length19 As length
Set length19 = parameters19.Item("WB")

designTable1.AddAssociation length19, "WB"

'Dim parameters20 As Parameters
Set parameters20 = part1.Parameters

Dim length20 As length
Set length20 = parameters20.Item("ABASE")

designTable1.AddAssociation length20, "ABASE"
```

```
'Dim parameters21 As Parameters

Set parameters21 = part1.Parameters


Dim length21 As length

Set length21 = parameters21.Item("AHEIGHT")


designTable1.AddAssociation length21, "AHEIGHT"


'Dim parameters22 As Parameters

Set parameters22 = part1.Parameters


Dim length22 As length

Set length22 = parameters22.Item("ATHICKNESS")


designTable1.AddAssociation length22, "ATHICKNESS"


'Dim parameters23 As Parameters

Set parameters23 = part1.Parameters


Dim length23 As length

Set length23 = parameters23.Item("BBASE")


designTable1.AddAssociation length23, "BBASE"


'Dim parameters24 As Parameters

Set parameters24 = part1.Parameters


Dim length24 As length

Set length24 = parameters24.Item("BHEIGHT")


designTable1.AddAssociation length24, "BHEIGHT"
```

```
'Dim parameters25 As Parameters
Set parameters25 = part1.Parameters

Dim length25 As length
Set length25 = parameters25.Item("BTHICKNESS")

designTable1.AddAssociation length25, "BTHICKNESS"

'Dim parameters26 As Parameters
Set parameters26 = part1.Parameters

Dim length26 As length
Set length26 = parameters26.Item("CBASE")

designTable1.AddAssociation length26, "CBASE"

'Dim parameters27 As Parameters
Set parameters27 = part1.Parameters

Dim length27 As length
Set length27 = parameters27.Item("CHEIGHT")

designTable1.AddAssociation length27, "CHEIGHT"

'Dim parameters28 As Parameters
Set parameters28 = part1.Parameters

Dim length28 As length
Set length28 = parameters28.Item("CTHICKNESS")

designTable1.AddAssociation length28, "CTHICKNESS"
```

```
'Dim parameters29 As Parameters
Set parameters29 = part1.Parameters

Dim length29 As length
Set length29 = parameters29.Item("DLBASE")

designTable1.AddAssociation length29, "DLBASE"

'Dim parameters30 As Parameters
Set parameters30 = part1.Parameters

Dim length30 As length
Set length30 = parameters30.Item("DLHEIGHT")

designTable1.AddAssociation length30, "DLHEIGHT"

'Dim parameters31 As Parameters
Set parameters31 = part1.Parameters

Dim length31 As length
Set length31 = parameters31.Item("DLTHICKNESS")

designTable1.AddAssociation length31, "DLTHICKNESS"

'Dim parameters32 As Parameters
Set parameters32 = part1.Parameters

Dim length32 As length
Set length32 = parameters32.Item("DRBASE")

designTable1.AddAssociation length32, "DRBASE"
```

```
'Dim parameters33 As Parameters

Set parameters33 = part1.Parameters


Dim length33 As length

Set length33 = parameters33.Item("DRHEIGHT")


designTable1.AddAssociation length33, "DRHEIGHT"


'Dim parameters34 As Parameters

Set parameters34 = part1.Parameters


Dim length34 As length

Set length34 = parameters34.Item("DRTHICKNESS")


designTable1.AddAssociation length34, "DRTHICKNESS"


'Dim parameters35 As Parameters

Set parameters35 = part1.Parameters


Dim length35 As length

Set length35 = parameters35.Item("EBASE")


designTable1.AddAssociation length35, "EBASE"


'Dim parameters36 As Parameters

Set parameters36 = part1.Parameters


Dim length36 As length

Set length36 = parameters36.Item("EHEIGHT")


designTable1.AddAssociation length36, "EHEIGHT"
```

```
'Dim parameters37 As Parameters
Set parameters37 = part1.Parameters

Dim length37 As length
Set length37 = parameters37.Item("ETHICKNESS")

designTable1.AddAssociation length37, "ETHICKNESS"

'Dim parameters38 As Parameters
Set parameters38 = part1.Parameters

Dim length38 As length
Set length38 = parameters38.Item("FBASE")

designTable1.AddAssociation length38, "FBASE"

'Dim parameters39 As Parameters
Set parameters39 = part1.Parameters

Dim length39 As length
Set length39 = parameters39.Item("FHEIGHT")

designTable1.AddAssociation length39, "FHEIGHT"

'Dim parameters40 As Parameters
Set parameters40 = part1.Parameters

Dim length40 As length
Set length40 = parameters40.Item("FTHICKNESS")

designTable1.AddAssociation length40, "FTHICKNESS"
```

```
'Dim parameters41 As Parameters
Set parameters41 = part1.Parameters

Dim length41 As length
Set length41 = parameters41.Item("GBASE")

designTable1.AddAssociation length41, "GBASE"

'Dim parameters42 As Parameters
Set parameters42 = part1.Parameters

Dim length42 As length
Set length42 = parameters42.Item("GHEIGHT")

designTable1.AddAssociation length42, "GHEIGHT"

'Dim parameters43 As Parameters
Set parameters43 = part1.Parameters

Dim length43 As length
Set length43 = parameters43.Item("GTHICKNESS")

designTable1.AddAssociation length43, "GTHICKNESS"

'Dim parameters44 As Parameters
Set parameters44 = part1.Parameters

Dim length44 As length
Set length44 = parameters44.Item("HBASE")

designTable1.AddAssociation length44, "HBASE"
```

```
'Dim parameters45 As Parameters
Set parameters45 = part1.Parameters

Dim length45 As length
Set length45 = parameters45.Item("HHEIGHT")

designTable1.AddAssociation length45, "HHEIGHT"

'Dim parameters46 As Parameters
Set parameters46 = part1.Parameters

Dim length46 As length
Set length46 = parameters46.Item("HTHICKNESS")

designTable1.AddAssociation length46, "HTHICKNESS"

'Dim parameters47 As Parameters
Set parameters47 = part1.Parameters

Dim length47 As length
Set length47 = parameters47.Item("IBASE")

designTable1.AddAssociation length47, "IBASE"

'Dim parameters48 As Parameters
Set parameters48 = part1.Parameters

Dim length48 As length
Set length48 = parameters48.Item("IHEIGHT")

designTable1.AddAssociation length48, "IHEIGHT"
```

```
'Dim parameters49 As Parameters
Set parameters49 = part1.Parameters

Dim length49 As length
Set length49 = parameters49.Item("ITHICKNESS")

designTable1.AddAssociation length49, "ITHICKNESS"

'Dim parameters50 As Parameters
Set parameters50 = part1.Parameters

Dim length50 As length
Set length50 = parameters50.Item("JBASE")

designTable1.AddAssociation length50, "JBASE"

'Dim parameters51 As Parameters
Set parameters51 = part1.Parameters

Dim length51 As length
Set length51 = parameters51.Item("JHEIGHT")

designTable1.AddAssociation length51, "JHEIGHT"

'Dim parameters52 As Parameters
Set parameters52 = part1.Parameters

Dim length52 As length
Set length52 = parameters52.Item("JTHICKNESS")

designTable1.AddAssociation length52, "JTHICKNESS"
```

```
'Dim parameters53 As Parameters
Set parameters53 = part1.Parameters

Dim length53 As length
Set length53 = parameters53.Item("KBASE")

designTable1.AddAssociation length53, "KBASE"

'Dim parameters54 As Parameters
Set parameters54 = part1.Parameters

Dim length54 As length
Set length54 = parameters54.Item("KHEIGHT")

designTable1.AddAssociation length54, "KHEIGHT"

'Dim parameters55 As Parameters
Set parameters55 = part1.Parameters

Dim length55 As length
Set length55 = parameters55.Item("KTHICKNESS")

designTable1.AddAssociation length55, "KTHICKNESS"

'Dim parameters56 As Parameters
Set parameters56 = part1.Parameters

Dim length56 As length
Set length56 = parameters56.Item("LBASE")

designTable1.AddAssociation length56, "LBASE"
```

```
'Dim parameters57 As Parameters
Set parameters57 = part1.Parameters

Dim length57 As length
Set length57 = parameters57.Item("LHEIGHT")

designTable1.AddAssociation length57, "LHEIGHT"

'Dim parameters58 As Parameters
Set parameters58 = part1.Parameters

Dim length58 As length
Set length58 = parameters58.Item("LTHICKNESS")

designTable1.AddAssociation length58, "LTHICKNESS"

'Dim parameters59 As Parameters
Set parameters59 = part1.Parameters

Dim length59 As length
Set length59 = parameters59.Item("MBASE")

designTable1.AddAssociation length59, "MBASE"

'Dim parameters60 As Parameters
Set parameters60 = part1.Parameters

Dim length60 As length
Set length60 = parameters60.Item("MHEIGHT")

designTable1.AddAssociation length60, "MHEIGHT"
```

```
'Dim parameters61 As Parameters
Set parameters61 = part1.Parameters

Dim length61 As length
Set length61 = parameters61.Item("MTHICKNESS")

designTable1.AddAssociation length61, "MTHICKNESS"

'Dim parameters62 As Parameters
Set parameters62 = part1.Parameters

Dim length62 As length
Set length62 = parameters62.Item("NBASE")

designTable1.AddAssociation length62, "NBASE"

'Dim parameters63 As Parameters
Set parameters63 = part1.Parameters

Dim length63 As length
Set length63 = parameters63.Item("NHEIGHT")

designTable1.AddAssociation length63, "NHEIGHT"

'Dim parameters64 As Parameters
Set parameters64 = part1.Parameters

Dim length64 As length
Set length64 = parameters64.Item("NTHICKNESS")

designTable1.AddAssociation length64, "NTHICKNESS"
```

```
'Dim parameters65 As Parameters
Set parameters65 = part1.Parameters

Dim length65 As length
Set length65 = parameters65.Item("OABASE")

designTable1.AddAssociation length65, "OABASE"

'Dim parameters66 As Parameters
Set parameters66 = part1.Parameters

Dim length66 As length
Set length66 = parameters66.Item("OAHEIGHT")

designTable1.AddAssociation length66, "OAHEIGHT"

'Dim parameters67 As Parameters
Set parameters67 = part1.Parameters

Dim length67 As length
Set length67 = parameters67.Item("OATHICKNESS")

designTable1.AddAssociation length67, "OATHICKNESS"

'Dim parameters68 As Parameters
Set parameters68 = part1.Parameters

Dim length68 As length
Set length68 = parameters68.Item("PABASE")

designTable1.AddAssociation length68, "PABASE"
```

```
'Dim parameters69 As Parameters
Set parameters69 = part1.Parameters

Dim length69 As length
Set length69 = parameters69.Item("PAHEIGHT")

designTable1.AddAssociation length69, "PAHEIGHT"

'Dim parameters70 As Parameters
Set parameters70 = part1.Parameters

Dim length70 As length
Set length70 = parameters70.Item("PATHICKNESS")

designTable1.AddAssociation length70, "PATHICKNESS"

'Dim parameters71 As Parameters
Set parameters71 = part1.Parameters

Dim length71 As length
Set length71 = parameters71.Item("QABASE")

designTable1.AddAssociation length71, "QABASE"

'Dim parameters72 As Parameters
Set parameters72 = part1.Parameters

Dim length72 As length
Set length72 = parameters72.Item("QAHEIGHT")

designTable1.AddAssociation length72, "QAHEIGHT"
```

```
'Dim parameters73 As Parameters
Set parameters73 = part1.Parameters

Dim length73 As length
Set length73 = parameters73.Item("QATHICKNESS")

designTable1.AddAssociation length73, "QATHICKNESS"

'Dim parameters74 As Parameters
Set parameters74 = part1.Parameters

Dim length74 As length
Set length74 = parameters74.Item("RABASE")

designTable1.AddAssociation length74, "RABASE"

'Dim parameters75 As Parameters
Set parameters75 = part1.Parameters

Dim length75 As length
Set length75 = parameters75.Item("RAHEIGHT")

designTable1.AddAssociation length75, "RAHEIGHT"

'Dim parameters76 As Parameters
Set parameters76 = part1.Parameters

Dim length76 As length
Set length76 = parameters76.Item("RATHICKNESS")

designTable1.AddAssociation length76, "RATHICKNESS"
```

```
'Dim parameters77 As Parameters
Set parameters77 = part1.Parameters

Dim length77 As length
Set length77 = parameters77.Item("OBBASE")

designTable1.AddAssociation length77, "OBBASE"

'Dim parameters78 As Parameters
Set parameters78 = part1.Parameters

Dim length78 As length
Set length78 = parameters78.Item("OBHEIGHT")

designTable1.AddAssociation length78, "OBHEIGHT"

'Dim parameters79 As Parameters
Set parameters79 = part1.Parameters

Dim length79 As length
Set length79 = parameters79.Item("OBTHICKNESS")

designTable1.AddAssociation length79, "OBTHICKNESS"

'Dim parameters80 As Parameters
Set parameters80 = part1.Parameters

Dim length80 As length
Set length80 = parameters80.Item("PBBASE")

designTable1.AddAssociation length80, "PBBASE"
```

```
'Dim parameters81 As Parameters

Set parameters81 = part1.Parameters


Dim length81 As length

Set length81 = parameters81.Item("PBHEIGHT")


designTable1.AddAssociation length81, "PBHEIGHT"


'Dim parameters82 As Parameters

Set parameters82 = part1.Parameters


Dim length82 As length

Set length82 = parameters82.Item("PBTHICKNESS")


designTable1.AddAssociation length82, "PBTHICKNESS"


'Dim parameters83 As Parameters

Set parameters83 = part1.Parameters


Dim length83 As length

Set length83 = parameters83.Item("QBBASE")


designTable1.AddAssociation length83, "QBBASE"


'Dim parameters84 As Parameters

Set parameters84 = part1.Parameters


Dim length84 As length

Set length84 = parameters84.Item("QBHEIGHT")


designTable1.AddAssociation length84, "QBHEIGHT"
```

```
'Dim parameters85 As Parameters
Set parameters85 = part1.Parameters

Dim length85 As length
Set length85 = parameters85.Item("QBTHICKNESS")

designTable1.AddAssociation length85, "QBTHICKNESS"

'Dim parameters86 As Parameters
Set parameters86 = part1.Parameters

Dim length86 As length
Set length86 = parameters86.Item("RBBASE")

designTable1.AddAssociation length86, "RBBASE"

'Dim parameters87 As Parameters
Set parameters87 = part1.Parameters

Dim length87 As length
Set length87 = parameters87.Item("RBHEIGHT")

designTable1.AddAssociation length87, "RBHEIGHT"

'Dim parameters88 As Parameters
Set parameters88 = part1.Parameters

Dim length88 As length
Set length88 = parameters88.Item("RBTHICKNESS")

designTable1.AddAssociation length88, "RBTHICKNESS"
```

```
'Dim parameters89 As Parameters

Set parameters89 = part1.Parameters


Dim length89 As length

Set length89 = parameters89.Item("SBASE")


designTable1.AddAssociation length89, "SBASE"


'Dim parameters90 As Parameters

Set parameters90 = part1.Parameters


Dim length90 As length

Set length90 = parameters90.Item("SHEIGHT")


designTable1.AddAssociation length90, "SHEIGHT"


'Dim parameters91 As Parameters

Set parameters91 = part1.Parameters


Dim length91 As length

Set length91 = parameters91.Item("STHICKNESS")


designTable1.AddAssociation length91, "STHICKNESS"


'Dim parameters92 As Parameters

Set parameters92 = part1.Parameters


Dim length92 As length

Set length92 = parameters92.Item("TBASE")


designTable1.AddAssociation length92, "TBASE"
```

```
'Dim parameters93 As Parameters

Set parameters93 = part1.Parameters


Dim length93 As length

Set length93 = parameters93.Item("THEIGHT")


designTable1.AddAssociation length93, "THEIGHT"


'Dim parameters94 As Parameters

Set parameters94 = part1.Parameters


Dim length94 As length

Set length94 = parameters94.Item("TTHICKNESS")


designTable1.AddAssociation length94, "TTHICKNESS"


'Dim parameters95 As Parameters

Set parameters95 = part1.Parameters


Dim length95 As length

Set length95 = parameters95.Item("UBASE")


designTable1.AddAssociation length95, "UBASE"


'Dim parameters96 As Parameters

Set parameters96 = part1.Parameters


Dim length96 As length

Set length96 = parameters96.Item("UHEIGHT")


designTable1.AddAssociation length96, "UHEIGHT"
```

```
'Dim parameters97 As Parameters

Set parameters97 = part1.Parameters


Dim length97 As length

Set length97 = parameters97.Item("UTHICKNESS")


designTable1.AddAssociation length97, "UTHICKNESS"


'Dim parameters98 As Parameters

Set parameters98 = part1.Parameters


Dim length98 As length

Set length98 = parameters98.Item("VDBASE")


designTable1.AddAssociation length98, "VDBASE"


'Dim parameters99 As Parameters

Set parameters99 = part1.Parameters


Dim length99 As length

Set length99 = parameters99.Item("VDHEIGHT")


designTable1.AddAssociation length99, "VDHEIGHT"


'Dim parameters100 As Parameters

Set parameters100 = part1.Parameters


Dim length100 As length

Set length100 = parameters100.Item("VDTHICKNESS")


designTable1.AddAssociation length100, "VDTHICKNESS"
```

```
'Dim parameters101 As Parameters
Set parameters101 = part1.Parameters

Dim length101 As length
Set length101 = parameters101.Item("VEBASE")

designTable1.AddAssociation length101, "VEBASE"

'Dim parameters102 As Parameters
Set parameters102 = part1.Parameters

Dim length102 As length
Set length102 = parameters102.Item("VEHEIGHT")

designTable1.AddAssociation length102, "VEHEIGHT"

'Dim parameters103 As Parameters
Set parameters103 = part1.Parameters

Dim length103 As length
Set length103 = parameters103.Item("VETHICKNESS")

designTable1.AddAssociation length103, "VETHICKNESS"

'Dim parameters104 As Parameters
Set parameters104 = part1.Parameters

Dim length104 As length
Set length104 = parameters104.Item("VFBASE")

designTable1.AddAssociation length104, "VFBASE"
```

```
'Dim parameters105 As Parameters
Set parameters105 = part1.Parameters

Dim length105 As length
Set length105 = parameters105.Item("VFHEIGHT")

designTable1.AddAssociation length105, "VFHEIGHT"

'Dim parameters106 As Parameters
Set parameters106 = part1.Parameters

Dim length106 As length
Set length106 = parameters106.Item("VFTHICKNESS")

designTable1.AddAssociation length106, "VFTHICKNESS"

designTable1.Configuration = 1

 On Error Resume Next
product1.Update

End If

Application.ScreenUpdating = True

End Sub
```

Módulo: EXPORT_CATIA_VARIABLE

```
Public variablename As Integer
Public Function FileFolderExists(strFullPath As String) As Boolean
'Author       : Ken Puls (www.excelguru.ca)
'Macro Purpose: Check if a file or folder exists
    On Error GoTo EarlyExit
    If Not Dir(strFullPath, vbDirectory) = vbNullString Then FileFolderExists = True

EarlyExit:
    On Error GoTo 0
End Function



Sub CATIA_variable()



Dim Response As VbMsgBoxResult
Response = MsgBox("Please Open CATIA Application", vbQuestion + vbYesNo)
If Response = vbNo Then

    MsgBox "You need to have CATIA to proceed", vbInformation

Else

    Application.ScreenUpdating = False
    'making visible the export worksheet
    Worksheets("EXPORT").Visible = -1

    'creating folder to save worksheet

    For variablename = 0 To 1000
```

```
        FolderName = "\CATIA_variable" & variablename
        If FileFolderExists(ThisWorkbook.Path & FolderName) Then

           GoTo nextfilename:

        Else

        Set ab = CreateObject("Scripting.FileSystemObject")
           ab.CreateFolder (ThisWorkbook.Path & FolderName & "/")

           GoTo continue:

        End If

nextfilename:
   Next

continue:

'creating workbook to be exported to CATIA
   Application.ScreenUpdating = False
   Application.DisplayAlerts = False

   Sheets("EXPORT").Range("A1:DB2").Copy

   Flname = "\Variable_input"

      Set NewWkbk = Workbooks.Add

      Sheets("Sheet1").Range("A1:DB2").PasteSpecial xlPasteValues
```

```
        NewWkbk.SaveAs ThisWorkbook.Path & FolderName & Flname



        ActiveWorkbook.Close


    'hiding the export worksheet

    Worksheets("EXPORT").Visible = 0


'OPENING CATIA

On Error Resume Next

Set ObjCATIA = GetObject(, "CATIA.Application")

If Err.Number <> 0 Then  ' this part of code is commented because is supposed that you have already CATIA opened

Set ObjCATIA = CreateObject("CATIA.Application")

CATIA.Visible = True

Err.Clear


End If

On Error GoTo 0


'OPENING TEMPLATE

ObjCATIA.Visible = True

ObjCATIA.DisplayFileAlerts = True

Set objProd = ObjCATIA.Documents.Open(ThisWorkbook.Path & "\Final_CATIA_Template\Variable_Thickness.CATProduct")


'Importing excel to CATIA


Dim productDocument1 As ProductDocument

Set productDocument1 = objProd


Dim product1 As Product

Set product1 = productDocument1.Product
```

```
Dim part1 As Part

Set part1 = ObjCATIA.Documents.Item("Parameters.CATPart").Part


Dim relations2 As Relations

Set relations2 = part1.Relations


Dim designTable1 As DesignTable

Set designTable1 = relations2.CreateDesignTable("DesignTable.1", "", False, ThisWorkbook.Path & FolderName & Flname & ".xlsx")


'Dim parameters1 As Parameters

Set parameters1 = part1.Parameters


Dim length1 As length

Set length1 = parameters1.Item("L1")


designTable1.AddAssociation length1, "L1"


'Dim parameters2 As Parameters

Set parameters2 = part1.Parameters


Dim length2 As length

Set length2 = parameters2.Item("L2")


designTable1.AddAssociation length2, "L2"


'Dim parameters3 As Parameters

Set parameters3 = part1.Parameters


Dim length3 As length

Set length3 = parameters3.Item("L3")
```

```
designTable1.AddAssociation length3, "L3"

'Dim parameters4 As Parameters
Set parameters4 = part1.Parameters

Dim length4 As length
Set length4 = parameters4.Item("L4")

designTable1.AddAssociation length4, "L4"

'Dim parameters5 As Parameters
Set parameters5 = part1.Parameters

Dim length5 As length
Set length5 = parameters5.Item("L5")

designTable1.AddAssociation length5, "L5"

'Dim parameters6 As Parameters
Set parameters6 = part1.Parameters

Dim length6 As length
Set length6 = parameters6.Item("L6")

designTable1.AddAssociation length6, "L6"

'Dim parameters7 As Parameters
Set parameters7 = part1.Parameters

Dim length7 As length
Set length7 = parameters7.Item("L7")
```

```
designTable1.AddAssociation length7, "L7"

'Dim parameters8 As Parameters
Set parameters8 = part1.Parameters

Dim length8 As length
Set length8 = parameters8.Item("L8")

designTable1.AddAssociation length8, "L8"

'Dim parameters9 As Parameters
Set parameters9 = part1.Parameters

Dim length9 As length
Set length9 = parameters9.Item("L9")

designTable1.AddAssociation length9, "L9"

'Dim parameters10 As Parameters
Set parameters10 = part1.Parameters

Dim length10 As length
Set length10 = parameters10.Item("W1")

designTable1.AddAssociation length10, "W1"

'Dim parameters11 As Parameters
Set parameters11 = part1.Parameters

Dim length11 As length
Set length11 = parameters11.Item("W2")
```

```
designTable1.AddAssociation length11, "W2"

'Dim parameters12 As Parameters

Set parameters12 = part1.Parameters

Dim length12 As length

Set length12 = parameters12.Item("W3")

designTable1.AddAssociation length12, "W3"

'Dim parameters13 As Parameters

Set parameters13 = part1.Parameters

Dim length13 As length

Set length13 = parameters13.Item("W4")

designTable1.AddAssociation length13, "W4"

'Dim parameters14 As Parameters

Set parameters14 = part1.Parameters

Dim length14 As length

Set length14 = parameters14.Item("H1")

designTable1.AddAssociation length14, "H1"

'Dim parameters15 As Parameters

Set parameters15 = part1.Parameters

Dim length15 As length

Set length15 = parameters15.Item("H2")
```

```
designTable1.AddAssociation length15, "H2"

'Dim parameters16 As Parameters
Set parameters16 = part1.Parameters

Dim length16 As length
Set length16 = parameters16.Item("H3")

designTable1.AddAssociation length16, "H3"

'Dim parameters17 As Parameters
Set parameters17 = part1.Parameters

Dim length17 As length
Set length17 = parameters17.Item("H4")

designTable1.AddAssociation length17, "H4"

'Dim parameters18 As Parameters
Set parameters18 = part1.Parameters

Dim length18 As length
Set length18 = parameters18.Item("H5")

designTable1.AddAssociation length18, "H5"

'Dim parameters19 As Parameters
Set parameters19 = part1.Parameters

Dim length19 As length
Set length19 = parameters19.Item("WB")
```

```
designTable1.AddAssociation length19, "WB"

'Dim parameters20 As Parameters
Set parameters20 = part1.Parameters

Dim length20 As length
Set length20 = parameters20.Item("ABASE")

designTable1.AddAssociation length20, "ABASE"

'Dim parameters21 As Parameters
Set parameters21 = part1.Parameters

Dim length21 As length
Set length21 = parameters21.Item("AHEIGHT")

designTable1.AddAssociation length21, "AHEIGHT"

'Dim parameters22 As Parameters
Set parameters22 = part1.Parameters

Dim length22 As length
Set length22 = parameters22.Item("ATHICKNESS")

designTable1.AddAssociation length22, "ATHICKNESS"

'Dim parameters23 As Parameters
Set parameters23 = part1.Parameters

Dim length23 As length
Set length23 = parameters23.Item("BBASE")
```

```
designTable1.AddAssociation length23, "BBASE"

'Dim parameters24 As Parameters
Set parameters24 = part1.Parameters

Dim length24 As length
Set length24 = parameters24.Item("BHEIGHT")

designTable1.AddAssociation length24, "BHEIGHT"

'Dim parameters25 As Parameters
Set parameters25 = part1.Parameters

Dim length25 As length
Set length25 = parameters25.Item("BTHICKNESS")

designTable1.AddAssociation length25, "BTHICKNESS"

'Dim parameters26 As Parameters
Set parameters26 = part1.Parameters

Dim length26 As length
Set length26 = parameters26.Item("CBASE")

designTable1.AddAssociation length26, "CBASE"

'Dim parameters27 As Parameters
Set parameters27 = part1.Parameters

Dim length27 As length
Set length27 = parameters27.Item("CHEIGHT")
```

```
designTable1.AddAssociation length27, "CHEIGHT"

'Dim parameters28 As Parameters
Set parameters28 = part1.Parameters

Dim length28 As length
Set length28 = parameters28.Item("CTHICKNESS")

designTable1.AddAssociation length28, "CTHICKNESS"

'Dim parameters29 As Parameters
Set parameters29 = part1.Parameters

Dim length29 As length
Set length29 = parameters29.Item("DLBASE")

designTable1.AddAssociation length29, "DLBASE"

'Dim parameters30 As Parameters
Set parameters30 = part1.Parameters

Dim length30 As length
Set length30 = parameters30.Item("DLHEIGHT")

designTable1.AddAssociation length30, "DLHEIGHT"

'Dim parameters31 As Parameters
Set parameters31 = part1.Parameters

Dim length31 As length
Set length31 = parameters31.Item("DLTHICKNESS")
```

```
designTable1.AddAssociation length31, "DLTHICKNESS"

'Dim parameters32 As Parameters
Set parameters32 = part1.Parameters

Dim length32 As length
Set length32 = parameters32.Item("DRBASE")

designTable1.AddAssociation length32, "DRBASE"

'Dim parameters33 As Parameters
Set parameters33 = part1.Parameters

Dim length33 As length
Set length33 = parameters33.Item("DRHEIGHT")

designTable1.AddAssociation length33, "DRHEIGHT"

'Dim parameters34 As Parameters
Set parameters34 = part1.Parameters

Dim length34 As length
Set length34 = parameters34.Item("DRTHICKNESS")

designTable1.AddAssociation length34, "DRTHICKNESS"

'Dim parameters35 As Parameters
Set parameters35 = part1.Parameters

Dim length35 As length
Set length35 = parameters35.Item("EBASE")
```

```
designTable1.AddAssociation length35, "EBASE"

'Dim parameters36 As Parameters
Set parameters36 = part1.Parameters

Dim length36 As length
Set length36 = parameters36.Item("EHEIGHT")

designTable1.AddAssociation length36, "EHEIGHT"

'Dim parameters37 As Parameters
Set parameters37 = part1.Parameters

Dim length37 As length
Set length37 = parameters37.Item("ETHICKNESS")

designTable1.AddAssociation length37, "ETHICKNESS"

'Dim parameters38 As Parameters
Set parameters38 = part1.Parameters

Dim length38 As length
Set length38 = parameters38.Item("FBASE")

designTable1.AddAssociation length38, "FBASE"

'Dim parameters39 As Parameters
Set parameters39 = part1.Parameters

Dim length39 As length
Set length39 = parameters39.Item("FHEIGHT")
```

```
designTable1.AddAssociation length39, "FHEIGHT"

'Dim parameters40 As Parameters
Set parameters40 = part1.Parameters

Dim length40 As length
Set length40 = parameters40.Item("FTHICKNESS")

designTable1.AddAssociation length40, "FTHICKNESS"

'Dim parameters41 As Parameters
Set parameters41 = part1.Parameters

Dim length41 As length
Set length41 = parameters41.Item("GBASE")

designTable1.AddAssociation length41, "GBASE"

'Dim parameters42 As Parameters
Set parameters42 = part1.Parameters

Dim length42 As length
Set length42 = parameters42.Item("GHEIGHT")

designTable1.AddAssociation length42, "GHEIGHT"

'Dim parameters43 As Parameters
Set parameters43 = part1.Parameters

Dim length43 As length
Set length43 = parameters43.Item("GTHICKNESS")
```

```
designTable1.AddAssociation length43, "GTHICKNESS"

'Dim parameters44 As Parameters
Set parameters44 = part1.Parameters

Dim length44 As length
Set length44 = parameters44.Item("HBASE")

designTable1.AddAssociation length44, "HBASE"

'Dim parameters45 As Parameters
Set parameters45 = part1.Parameters

Dim length45 As length
Set length45 = parameters45.Item("HHEIGHT")

designTable1.AddAssociation length45, "HHEIGHT"

'Dim parameters46 As Parameters
Set parameters46 = part1.Parameters

Dim length46 As length
Set length46 = parameters46.Item("HTHICKNESS")

designTable1.AddAssociation length46, "HTHICKNESS"

'Dim parameters47 As Parameters
Set parameters47 = part1.Parameters

Dim length47 As length
Set length47 = parameters47.Item("IBASE")
```

```
designTable1.AddAssociation length47, "IBASE"

'Dim parameters48 As Parameters
Set parameters48 = part1.Parameters

Dim length48 As length
Set length48 = parameters48.Item("IHEIGHT")

designTable1.AddAssociation length48, "IHEIGHT"

'Dim parameters49 As Parameters
Set parameters49 = part1.Parameters

Dim length49 As length
Set length49 = parameters49.Item("ITHICKNESS")

designTable1.AddAssociation length49, "ITHICKNESS"

'Dim parameters50 As Parameters
Set parameters50 = part1.Parameters

Dim length50 As length
Set length50 = parameters50.Item("JBASE")

designTable1.AddAssociation length50, "JBASE"

'Dim parameters51 As Parameters
Set parameters51 = part1.Parameters

Dim length51 As length
Set length51 = parameters51.Item("JHEIGHT")
```

```
designTable1.AddAssociation length51, "JHEIGHT"

'Dim parameters52 As Parameters
Set parameters52 = part1.Parameters

Dim length52 As length
Set length52 = parameters52.Item("JTHICKNESS")

designTable1.AddAssociation length52, "JTHICKNESS"

'Dim parameters53 As Parameters
Set parameters53 = part1.Parameters

Dim length53 As length
Set length53 = parameters53.Item("KBASE")

designTable1.AddAssociation length53, "KBASE"

'Dim parameters54 As Parameters
Set parameters54 = part1.Parameters

Dim length54 As length
Set length54 = parameters54.Item("KHEIGHT")

designTable1.AddAssociation length54, "KHEIGHT"

'Dim parameters55 As Parameters
Set parameters55 = part1.Parameters

Dim length55 As length
Set length55 = parameters55.Item("KTHICKNESS")
```

```
designTable1.AddAssociation length55, "KTHICKNESS"

'Dim parameters56 As Parameters
Set parameters56 = part1.Parameters

Dim length56 As length
Set length56 = parameters56.Item("LBASE")

designTable1.AddAssociation length56, "LBASE"

'Dim parameters57 As Parameters
Set parameters57 = part1.Parameters

Dim length57 As length
Set length57 = parameters57.Item("LHEIGHT")

designTable1.AddAssociation length57, "LHEIGHT"

'Dim parameters58 As Parameters
Set parameters58 = part1.Parameters

Dim length58 As length
Set length58 = parameters58.Item("LTHICKNESS")

designTable1.AddAssociation length58, "LTHICKNESS"

'Dim parameters59 As Parameters
Set parameters59 = part1.Parameters

Dim length59 As length
Set length59 = parameters59.Item("MBASE")
```

```
designTable1.AddAssociation length59, "MBASE"

'Dim parameters60 As Parameters
Set parameters60 = part1.Parameters

Dim length60 As length
Set length60 = parameters60.Item("MHEIGHT")

designTable1.AddAssociation length60, "MHEIGHT"

'Dim parameters61 As Parameters
Set parameters61 = part1.Parameters

Dim length61 As length
Set length61 = parameters61.Item("MTHICKNESS")

designTable1.AddAssociation length61, "MTHICKNESS"

'Dim parameters62 As Parameters
Set parameters62 = part1.Parameters

Dim length62 As length
Set length62 = parameters62.Item("NBASE")

designTable1.AddAssociation length62, "NBASE"

'Dim parameters63 As Parameters
Set parameters63 = part1.Parameters

Dim length63 As length
Set length63 = parameters63.Item("NHEIGHT")
```

```
designTable1.AddAssociation length63, "NHEIGHT"

'Dim parameters64 As Parameters
Set parameters64 = part1.Parameters

Dim length64 As length
Set length64 = parameters64.Item("NTHICKNESS")

designTable1.AddAssociation length64, "NTHICKNESS"

'Dim parameters65 As Parameters
Set parameters65 = part1.Parameters

Dim length65 As length
Set length65 = parameters65.Item("OABASE")

designTable1.AddAssociation length65, "OABASE"

'Dim parameters66 As Parameters
Set parameters66 = part1.Parameters

Dim length66 As length
Set length66 = parameters66.Item("OAHEIGHT")

designTable1.AddAssociation length66, "OAHEIGHT"

'Dim parameters67 As Parameters
Set parameters67 = part1.Parameters

Dim length67 As length
Set length67 = parameters67.Item("OATHICKNESS")
```

```
designTable1.AddAssociation length67, "OATHICKNESS"

'Dim parameters68 As Parameters
Set parameters68 = part1.Parameters

Dim length68 As length
Set length68 = parameters68.Item("PABASE")

designTable1.AddAssociation length68, "PABASE"

'Dim parameters69 As Parameters
Set parameters69 = part1.Parameters

Dim length69 As length
Set length69 = parameters69.Item("PAHEIGHT")

designTable1.AddAssociation length69, "PAHEIGHT"

'Dim parameters70 As Parameters
Set parameters70 = part1.Parameters

Dim length70 As length
Set length70 = parameters70.Item("PATHICKNESS")

designTable1.AddAssociation length70, "PATHICKNESS"

'Dim parameters71 As Parameters
Set parameters71 = part1.Parameters

Dim length71 As length
Set length71 = parameters71.Item("QABASE")
```

```
designTable1.AddAssociation length71, "QABASE"

'Dim parameters72 As Parameters
Set parameters72 = part1.Parameters

Dim length72 As length
Set length72 = parameters72.Item("QAHEIGHT")

designTable1.AddAssociation length72, "QAHEIGHT"

'Dim parameters73 As Parameters
Set parameters73 = part1.Parameters

Dim length73 As length
Set length73 = parameters73.Item("QATHICKNESS")

designTable1.AddAssociation length73, "QATHICKNESS"

'Dim parameters74 As Parameters
Set parameters74 = part1.Parameters

Dim length74 As length
Set length74 = parameters74.Item("RABASE")

designTable1.AddAssociation length74, "RABASE"

'Dim parameters75 As Parameters
Set parameters75 = part1.Parameters

Dim length75 As length
Set length75 = parameters75.Item("RAHEIGHT")
```

```
designTable1.AddAssociation length75, "RAHEIGHT"

'Dim parameters76 As Parameters
Set parameters76 = part1.Parameters

Dim length76 As length
Set length76 = parameters76.Item("RATHICKNESS")

designTable1.AddAssociation length76, "RATHICKNESS"

'Dim parameters77 As Parameters
Set parameters77 = part1.Parameters

Dim length77 As length
Set length77 = parameters77.Item("OBBASE")

designTable1.AddAssociation length77, "OBBASE"

'Dim parameters78 As Parameters
Set parameters78 = part1.Parameters

Dim length78 As length
Set length78 = parameters78.Item("OBHEIGHT")

designTable1.AddAssociation length78, "OBHEIGHT"

'Dim parameters79 As Parameters
Set parameters79 = part1.Parameters

Dim length79 As length
Set length79 = parameters79.Item("OBTHICKNESS")
```

```
designTable1.AddAssociation length79, "OBTHICKNESS"

'Dim parameters80 As Parameters
Set parameters80 = part1.Parameters

Dim length80 As length
Set length80 = parameters80.Item("PBBASE")

designTable1.AddAssociation length80, "PBBASE"

'Dim parameters81 As Parameters
Set parameters81 = part1.Parameters

Dim length81 As length
Set length81 = parameters81.Item("PBHEIGHT")

designTable1.AddAssociation length81, "PBHEIGHT"

'Dim parameters82 As Parameters
Set parameters82 = part1.Parameters

Dim length82 As length
Set length82 = parameters82.Item("PBTHICKNESS")

designTable1.AddAssociation length82, "PBTHICKNESS"

'Dim parameters83 As Parameters
Set parameters83 = part1.Parameters

Dim length83 As length
Set length83 = parameters83.Item("QBBASE")
```

```
designTable1.AddAssociation length83, "QBBASE"

'Dim parameters84 As Parameters
Set parameters84 = part1.Parameters

Dim length84 As length
Set length84 = parameters84.Item("QBHEIGHT")

designTable1.AddAssociation length84, "QBHEIGHT"

'Dim parameters85 As Parameters
Set parameters85 = part1.Parameters

Dim length85 As length
Set length85 = parameters85.Item("QBTHICKNESS")

designTable1.AddAssociation length85, "QBTHICKNESS"

'Dim parameters86 As Parameters
Set parameters86 = part1.Parameters

Dim length86 As length
Set length86 = parameters86.Item("RBBASE")

designTable1.AddAssociation length86, "RBBASE"

'Dim parameters87 As Parameters
Set parameters87 = part1.Parameters

Dim length87 As length
Set length87 = parameters87.Item("RBHEIGHT")
```

```
designTable1.AddAssociation length87, "RBHEIGHT"

'Dim parameters88 As Parameters
Set parameters88 = part1.Parameters

Dim length88 As length
Set length88 = parameters88.Item("RBTHICKNESS")

designTable1.AddAssociation length88, "RBTHICKNESS"

'Dim parameters89 As Parameters
Set parameters89 = part1.Parameters

Dim length89 As length
Set length89 = parameters89.Item("SBASE")

designTable1.AddAssociation length89, "SBASE"

'Dim parameters90 As Parameters
Set parameters90 = part1.Parameters

Dim length90 As length
Set length90 = parameters90.Item("SHEIGHT")

designTable1.AddAssociation length90, "SHEIGHT"

'Dim parameters91 As Parameters
Set parameters91 = part1.Parameters

Dim length91 As length
Set length91 = parameters91.Item("STHICKNESS")
```

```
designTable1.AddAssociation length91, "STHICKNESS"

'Dim parameters92 As Parameters
Set parameters92 = part1.Parameters

Dim length92 As length
Set length92 = parameters92.Item("TBASE")

designTable1.AddAssociation length92, "TBASE"

'Dim parameters93 As Parameters
Set parameters93 = part1.Parameters

Dim length93 As length
Set length93 = parameters93.Item("THEIGHT")

designTable1.AddAssociation length93, "THEIGHT"

'Dim parameters94 As Parameters
Set parameters94 = part1.Parameters

Dim length94 As length
Set length94 = parameters94.Item("TTHICKNESS")

designTable1.AddAssociation length94, "TTHICKNESS"

'Dim parameters95 As Parameters
Set parameters95 = part1.Parameters

Dim length95 As length
Set length95 = parameters95.Item("UBASE")
```

```
designTable1.AddAssociation length95, "UBASE"

'Dim parameters96 As Parameters
Set parameters96 = part1.Parameters

Dim length96 As length
Set length96 = parameters96.Item("UHEIGHT")

designTable1.AddAssociation length96, "UHEIGHT"

'Dim parameters97 As Parameters
Set parameters97 = part1.Parameters

Dim length97 As length
Set length97 = parameters97.Item("UTHICKNESS")

designTable1.AddAssociation length97, "UTHICKNESS"

'Dim parameters98 As Parameters
Set parameters98 = part1.Parameters

Dim length98 As length
Set length98 = parameters98.Item("VDBASE")

designTable1.AddAssociation length98, "VDBASE"

'Dim parameters99 As Parameters
Set parameters99 = part1.Parameters

Dim length99 As length
Set length99 = parameters99.Item("VDHEIGHT")
```

```
designTable1.AddAssociation length99, "VDHEIGHT"

'Dim parameters100 As Parameters
Set parameters100 = part1.Parameters

Dim length100 As length
Set length100 = parameters100.Item("VDTHICKNESS")

designTable1.AddAssociation length100, "VDTHICKNESS"

'Dim parameters101 As Parameters
Set parameters101 = part1.Parameters

Dim length101 As length
Set length101 = parameters101.Item("VEBASE")

designTable1.AddAssociation length101, "VEBASE"

'Dim parameters102 As Parameters
Set parameters102 = part1.Parameters

Dim length102 As length
Set length102 = parameters102.Item("VEHEIGHT")

designTable1.AddAssociation length102, "VEHEIGHT"

'Dim parameters103 As Parameters
Set parameters103 = part1.Parameters

Dim length103 As length
Set length103 = parameters103.Item("VETHICKNESS")
```

```
designTable1.AddAssociation length103, "VETHICKNESS"

'Dim parameters104 As Parameters
Set parameters104 = part1.Parameters

Dim length104 As length
Set length104 = parameters104.Item("VFBASE")

designTable1.AddAssociation length104, "VFBASE"

'Dim parameters105 As Parameters
Set parameters105 = part1.Parameters

Dim length105 As length
Set length105 = parameters105.Item("VFHEIGHT")

designTable1.AddAssociation length105, "VFHEIGHT"

'Dim parameters106 As Parameters
Set parameters106 = part1.Parameters

Dim length106 As length
Set length106 = parameters106.Item("VFTHICKNESS")

designTable1.AddAssociation length106, "VFTHICKNESS"

designTable1.Configuration = 1

 On Error Resume Next
product1.Update
```

```
End If

Application.ScreenUpdating = True

End Sub
```

Módulo: MAIN_OPTIMIZATION_SCRIPT

```
Public Torsional_Stiffness As Double

Public Vertical_Stiffness As Double

Public Mass_total As Double

Public WB, W1, W2, W3, W4, H1, H2, H3, H4, H5, L1, L2, L3, L4, L5, L6, L7, L8, L9 As Double

Public length(29) As Double


Public Steel_Density, Limit_Tors_Stiff, Limit_Vert_Stiff As Double

Public Tmatrix1(11, 11), Tmatrix2(11, 11), Tmatrix3(11, 11), Tmatrix4(11, 11), Tmatrix5(11, 11), Tmatrix6(11, 11), Tmatrix7(11, 11) _

, Tmatrix8(11, 11), Tmatrix9(11, 11), Tmatrix10(11, 11), Tmatrix11(11, 11), Tmatrix12(11, 11), Tmatrix13(11, 11), Tmatrix14(11, 11) _

, Tmatrix15(11, 11), Tmatrix16(11, 11), Tmatrix17(11, 11), Tmatrix18(11, 11), Tmatrix19(11, 11), Tmatrix20(11, 11), Tmatrix21(11, 11) _

, Tmatrix22(11, 11), Tmatrix23(11, 11), Tmatrix24(11, 11), Tmatrix25(11, 11), Tmatrix26(11, 11), Tmatrix27(11, 11), Tmatrix28(11, 11) _

, Tmatrix29(11, 11) As Double

Sub Code()


    Dim i As Integer


        Application.ScreenUpdating = False

        Worksheets("OPTIMIZATION").Visible = -1

        Worksheets("SOLUTION").Visible = -1




    If Range("WB").Value = "" Or Range("Width1").Value = "" Or Range("Width2").Value = "" Or Range("Width3").Value = "" Or Range("Width4").Value = "" _

          Or Range("Height1").Value = "" Or Range("Height2").Value = "" Or Range("Height3").Value = "" Or Range("Height4").Value = "" Or Range("Height5").Value = "" _
```

```
    Or Range("Length1").Value = "" Or Range("Length2").Value = "" Or Range("Length3").Value = "" Or Range("Length4").Value = "" Or Range("Length5").Value = "" _

    Or Range("Length6").Value = "" Or Range("Length7").Value = "" Or Range("Length8").Value = "" Or Range("Length9").Value = "" Or Range("Density").Value = "" _

    Or Range("LIMITOR").Value = "" Or Range("LIMITVER").Value = "" Then


      UserForm1.Hide

      MsgBox ("PLEASE COMPLETE ALL THE REQUIRED DATA")



  Else


    WB = Range("WB").Value          'm

    W1 = Range("Width1").Value      'm

    W2 = Range("Width2").Value      'm

    W3 = Range("Width3").Value      'm

    W4 = Range("Width4").Value      'm

    H1 = Range("Height1").Value     'm

    H2 = Range("Height2").Value     'm

    H3 = Range("Height3").Value     'm

    H4 = Range("Height4").Value     'm

    H5 = Range("Height5").Value     'm

    L1 = Range("Length1").Value     'm

    L2 = Range("Length2").Value     'm

    L3 = Range("Length3").Value     'm

    L4 = Range("Length4").Value     'm

    L5 = Range("Length5").Value     'm

    L6 = Range("Length6").Value     'm

    L7 = Range("Length7").Value     'm

    L8 = Range("Length8").Value     'm

    L9 = Range("Length9").Value     'm

    Steel_Density = Range("Density").Value  'kg/m3
```

```
    Limit_Tors_Stiff = Range("LIMITOR").Value

    Limit_Vert_Stiff = Range("LIMITVER").Value


If L3 < L7 Then


    UserForm1.Hide

    MsgBox ("GEOMETRY IS NOT VALID, L3 MOST BE BIGGER THAN L7")


    Else


If (WB - ((L2 / 2) + L3 + L4 + L5)) > L6 Or (2 * (WB - ((L2 / 2) + L3 + L4 + L5)) + L9) > L6 Then



UserForm1.Hide

MsgBox ("GEOMETRY IS NOT VALID, PLEASE CONFIRM L6 MEASUREMENTS")


  Else


If WB < ((L2 / 2) + L3 + L4 + L5) Then


UserForm1.Hide

MsgBox ("GEOMETRY IS NOT VALID, SUM OF LENGTH 3,4,5 AND HALF OF 2 SHOULD BE SHORTER THAN
WB")


    Else


    Dim Xii(29), Xjj(29), Yii(29), Yjj(29), Zii(29), Zjj(29), Xk(29), Yk(29), Zk(29), LengthK(29) As Double


    Dim f As Integer



    Erase Xii
```

```
Erase Yii

Erase Zii


Xjj(0) = L1

Xjj(1) = L2 / 2

Xjj(2) = L2 / 2

Xjj(3) = L7

Xjj(4) = L7

Xjj(5) = L3 - L7

Xjj(6) = (WB / 2) - ((L2 / 2) + L3)

Xjj(7) = (WB / 2) - L5 - (L2 / 2)

Xjj(8) = L5 / 2

Xjj(9) = L5 / 2

Xjj(10) = L2 / 2

Xjj(11) = L2 / 2

Xjj(12) = L9

Xjj(13) = L6 - L2 - L9

Xjj(14) = L8

Xjj(15) = 0

Xjj(16) = 0

Xjj(17) = 0

Xjj(18) = 0

Xjj(19) = 0

Xjj(20) = 0

Xjj(21) = 0

Xjj(22) = 0

Xjj(23) = 0

Xjj(24) = 0

Xjj(25) = 0

Xjj(26) = 0

Xjj(27) = 0

Xjj(28) = 0
```

Yjj(0) = 0

Yjj(1) = 0

Yjj(2) = 0

Yjj(3) = (W2 - W1) / 2

Yjj(4) = -(W2 - W1) / 2

Yjj(5) = 0

Yjj(6) = 0

Yjj(7) = 0

Yjj(8) = 0

Yjj(9) = 0

Yjj(10) = 0

Yjj(11) = 0

Yjj(12) = 0

Yjj(13) = 0

Yjj(14) = 0

Yjj(15) = -W3

Yjj(16) = -((W1 / 2) - W3)

Yjj(17) = -((W1 / 2) - W3)

Yjj(18) = -W3

Yjj(19) = -W3

Yjj(20) = -((W1 / 2) - W3)

Yjj(21) = -((W1 / 2) - W3)

Yjj(22) = -W3

Yjj(23) = -W4

Yjj(24) = -(W2 - 2 * (W4))

Yjj(25) = -W4

Yjj(26) = -W2

Yjj(27) = -W2

Yjj(28) = -W2

Zjj(0) = 0

```
Zjj(1) = 0

Zjj(2) = 0

Zjj(3) = -H3

Zjj(4) = -H3

Zjj(5) = 0

Zjj(6) = 0

Zjj(7) = 0

Zjj(8) = 0

Zjj(9) = H3

Zjj(10) = 0

Zjj(11) = 0

Zjj(12) = -H4

Zjj(13) = 0

Zjj(14) = 0

Zjj(15) = -H1

Zjj(16) = 0

Zjj(17) = 0

Zjj(18) = H1

Zjj(19) = -H2

Zjj(20) = 0

Zjj(21) = 0

Zjj(22) = H2

Zjj(23) = -H5

Zjj(24) = 0

Zjj(25) = H5

Zjj(26) = 0

Zjj(27) = 0

Zjj(28) = 0



For i = 0 To 28
```

```
        'defining length of the element
        length(i) = Sqr((Xjj(i) - Xii(i)) ^ 2 + (Yjj(i) - Yii(i)) ^ 2 + (Zjj(i) - Zii(i)) ^ 2)

        'defining the support vector
        Xk(i) = (Xjj(i) - Xii(i)) / 2
        Yk(i) = (Yjj(i) - Yii(i)) / 2
        Zk(i) = length(i) / 2

        LengthK(i) = Sqr((Xk(i) - Xii(i)) ^ 2 + (Yk(i) - Yii(i)) ^ 2 + (Zk(i) - Zii(i)) ^ 2)

    Next i

'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

'Generate the transformation matrix to transform the coordinates into global
    Dim Transfmatrix(11, 347) As Double
    Dim g As Integer

  '//// initialize with 0 all the matrix

    Erase Transfmatrix

'Transformation Matrix from local coordinates to global

    Dim lx(29), mx(29), nx(29), ly(29), my(29), ny(29), lz(29), mz(29), nz(29), lik(29), mik(29), nik(29), Tempvalue(29) As Double
    Dim lmnx(2, 28), lmnik(2, 28), lmnz(3, 29), lmny(2, 28), TempArray(3, 29) As Variant

    i = 1
    For i = 0 To 28

        lx(i) = (Xjj(i) - Xii(i)) / length(i)
```

```
mx(i) = (Yjj(i) - Yii(i)) / length(i)
nx(i) = (Zjj(i) - Zii(i)) / length(i)

lik(i) = (Xk(i) - Xii(i)) / LengthK(i)
mik(i) = (Yk(i) - Yii(i)) / LengthK(i)
nik(i) = (Zk(i) - Zii(i)) / LengthK(i)



lmnx(0, i) = lx(i)
lmnx(1, i) = mx(i)
lmnx(2, i) = nx(i)

lmnik(0, i) = lik(i)
lmnik(1, i) = mik(i)
lmnik(2, i) = nik(i)

'cross product

TempArray(0, i) = lmnx(1, i) * _
   lmnik(2, i) - lmnx(2, i) * _
   lmnik(1, i)
TempArray(1, i) = lmnx(2, i) * _
   lmnik(0, i) - lmnx(0, i) * _
   lmnik(2, i)
TempArray(2, i) = lmnx(0, i) * _
   lmnik(1, i) - lmnx(1, i) * _
   lmnik(0, i)


Tempvalue(i) = Sqr((TempArray(0, i)) ^ 2 + (TempArray(1, i)) ^ 2 + (TempArray(2, i)) ^ 2)
```

```
lmnz(0, i) = TempArray(0, i) / Tempvalue(i)
lmnz(1, i) = TempArray(1, i) / Tempvalue(i)
lmnz(2, i) = TempArray(2, i) / Tempvalue(i)

lmny(0, i) = lmnz(1, i) * _
    lmnx(2, i) - lmnz(2, i) * _
    lmnx(1, i)
lmny(1, i) = lmnz(2, i) * _
    lmnx(0, i) - lmnz(0, i) * _
    lmnx(2, i)
lmny(2, i) = lmnz(0, i) * _
    lmnx(1, i) - lmnz(1, i) * _
    lmnx(0, i)

'1st column of each submatrix
Transfmatrix(0, 12 * i) = lmnx(0, i)
Transfmatrix(1, 12 * i) = lmnx(1, i)
Transfmatrix(2, 12 * i) = lmnx(2, i)

'2nd column of each submatrix
Transfmatrix(0, 12 * (i + 1) - 11) = lmny(0, i)
Transfmatrix(1, 12 * (i + 1) - 11) = lmny(1, i)
Transfmatrix(2, 12 * (i + 1) - 11) = lmny(2, i)

'3rd column of each submatrix
Transfmatrix(0, 12 * (i + 1) - 10) = lmnz(0, i)
Transfmatrix(1, 12 * (i + 1) - 10) = lmnz(1, i)
Transfmatrix(2, 12 * (i + 1) - 10) = lmnz(2, i)

'4th column of each submatrix
Transfmatrix(3, 12 * (i + 1) - 9) = lmnx(0, i)
Transfmatrix(4, 12 * (i + 1) - 9) = lmnx(1, i)
```

```
Transfmatrix(5, 12 * (i + 1) - 9) = lmnx(2, i)

'5th column of each submatrix
Transfmatrix(3, 12 * (i + 1) - 8) = lmny(0, i)
Transfmatrix(4, 12 * (i + 1) - 8) = lmny(1, i)
Transfmatrix(5, 12 * (i + 1) - 8) = lmny(2, i)

'6th column of each submatrix
Transfmatrix(3, 12 * (i + 1) - 7) = lmnz(0, i)
Transfmatrix(4, 12 * (i + 1) - 7) = lmnz(1, i)
Transfmatrix(5, 12 * (i + 1) - 7) = lmnz(2, i)

'7th column of each submatrix
Transfmatrix(6, 12 * (i + 1) - 6) = lmnx(0, i)
Transfmatrix(7, 12 * (i + 1) - 6) = lmnx(1, i)
Transfmatrix(8, 12 * (i + 1) - 6) = lmnx(2, i)

'8th column of each submatrix
Transfmatrix(6, 12 * (i + 1) - 5) = lmny(0, i)
Transfmatrix(7, 12 * (i + 1) - 5) = lmny(1, i)
Transfmatrix(8, 12 * (i + 1) - 5) = lmny(2, i)

'9th column of each submatrix
Transfmatrix(6, 12 * (i + 1) - 4) = lmnz(0, i)
Transfmatrix(7, 12 * (i + 1) - 4) = lmnz(1, i)
Transfmatrix(8, 12 * (i + 1) - 4) = lmnz(2, i)

'10th column of each submatrix
Transfmatrix(9, 12 * (i + 1) - 3) = lmnx(0, i)
Transfmatrix(10, 12 * (i + 1) - 3) = lmnx(1, i)
Transfmatrix(11, 12 * (i + 1) - 3) = lmnx(2, i)
```

```
        '11th column of each submatrix
        Transfmatrix(9, 12 * (i + 1) - 2) = lmny(0, i)
        Transfmatrix(10, 12 * (i + 1) - 2) = lmny(1, i)
        Transfmatrix(11, 12 * (i + 1) - 2) = lmny(2, i)

        '12th column of each submatrix
        Transfmatrix(9, 12 * (i + 1) - 1) = lmnz(0, i)
        Transfmatrix(10, 12 * (i + 1) - 1) = lmnz(1, i)
        Transfmatrix(11, 12 * (i + 1) - 1) = lmnz(2, i)

    Next i
'

    Dim p As Integer

    For k = 0 To 11

        For p = 0 To 11
            Tmatrix1(k, p) = Transfmatrix(k, p)
            Tmatrix2(k, p) = Transfmatrix(k, p + 12)
            Tmatrix3(k, p) = Transfmatrix(k, p + 24)
            Tmatrix4(k, p) = Transfmatrix(k, p + 36)
            Tmatrix5(k, p) = Transfmatrix(k, p + 48)
            Tmatrix6(k, p) = Transfmatrix(k, p + 60)
            Tmatrix7(k, p) = Transfmatrix(k, p + 72)
            Tmatrix8(k, p) = Transfmatrix(k, p + 84)
            Tmatrix9(k, p) = Transfmatrix(k, p + 96)
            Tmatrix10(k, p) = Transfmatrix(k, p + 108)
            Tmatrix11(k, p) = Transfmatrix(k, p + 120)
            Tmatrix12(k, p) = Transfmatrix(k, p + 132)
```

```
        Tmatrix13(k, p) = Transfmatrix(k, p + 144)
        Tmatrix14(k, p) = Transfmatrix(k, p + 156)
        Tmatrix15(k, p) = Transfmatrix(k, p + 168)
        Tmatrix16(k, p) = Transfmatrix(k, p + 180)
        Tmatrix17(k, p) = Transfmatrix(k, p + 192)
        Tmatrix18(k, p) = Transfmatrix(k, p + 204)
        Tmatrix19(k, p) = Transfmatrix(k, p + 216)
        Tmatrix20(k, p) = Transfmatrix(k, p + 228)
        Tmatrix21(k, p) = Transfmatrix(k, p + 240)
        Tmatrix22(k, p) = Transfmatrix(k, p + 252)
        Tmatrix23(k, p) = Transfmatrix(k, p + 264)
        Tmatrix24(k, p) = Transfmatrix(k, p + 276)
        Tmatrix25(k, p) = Transfmatrix(k, p + 288)
        Tmatrix26(k, p) = Transfmatrix(k, p + 300)
        Tmatrix27(k, p) = Transfmatrix(k, p + 312)
        Tmatrix28(k, p) = Transfmatrix(k, p + 324)
        Tmatrix29(k, p) = Transfmatrix(k, p + 336)
    Next p
Next k

    Dim Result As Variant
    Dim Name_function As String
    Dim Param As Variant
    Dim Constr As Variant


        Name_function = Range("Name").Value
        Param = Range("Param").Value
        Constr = Range("Constr").Value

        Torsional_Stiffness = 0
        Vertical_Stiffness = 0
```

```
            Mass_total = 0

            Result = CALL_TEST_MULTVAR_FRAME_FUNC(Name_function, Param, Constr)

            Range("SOLUTION") = Result

            Range("TORSIONAL") = Torsional_Stiffness
            Range("VERTICAL") = Vertical_Stiffness
            Range("MASS") = Mass_total

            Worksheets("OPTIMIZATION").Visible = 0
            Worksheets("DATASHEET").Visible = 0


            Application.ScreenUpdating = True

            UserForm1.Hide


        End If

        End If
        End If
        End If
    End Sub
```

Módulo: MATRIX_RC_TRANSPONSE_LIBR

```
Option Explicit     'Requires that all variables to be declared explicitly.
Option Base 1       'The "Option Base" statement allows to specify 0 or 1 as the
                    'default first index of arrays.



'*********************************************************************************
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION      : MATRIX_TRANSPOSE_FUNC
'DESCRIPTION   : TRANSPOSE AN ARRAY (FROM N X M TO M X N)
'LIBRARY       : MATRIX
'GROUP         : RC_TRANSPOSE
'ID            : 001
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 01/19/2009
'*********************************************************************************
'*********************************************************************************

Function MATRIX_TRANSPOSE_FUNC(ByRef DATA_RNG As Variant)

Dim i As Long
Dim j As Long

Dim SROW As Long
Dim SCOLUMN As Long
```

```
Dim NROWS As Long
Dim NCOLUMNS As Long

Dim TEMP_MATRIX As Variant
Dim DATA_MATRIX As Variant

On Error GoTo ERROR_LABEL

DATA_MATRIX = DATA_RNG

If IS_MATRIX_FUNC(DATA_MATRIX) = True Then
   SROW = LBound(DATA_MATRIX, 1)
   SCOLUMN = LBound(DATA_MATRIX, 2)

   NROWS = UBound(DATA_MATRIX, 1)
   NCOLUMNS = UBound(DATA_MATRIX, 2)

   ReDim TEMP_MATRIX(SCOLUMN To NCOLUMNS, SROW To NROWS)
   For j = SCOLUMN To NCOLUMNS
      For i = SROW To NROWS
         TEMP_MATRIX(j, i) = DATA_MATRIX(i, j)
      Next i
   Next j
Else
   If IS_ARRAY_FUNC(DATA_MATRIX) Then

      SROW = LBound(DATA_MATRIX)
      NROWS = UBound(DATA_MATRIX)

      ReDim TEMP_MATRIX(SROW To NROWS, SROW To SROW)
      For i = SROW To NROWS
         TEMP_MATRIX(i, SROW) = DATA_MATRIX(i)
```

```
        Next i
    Else
        GoTo ERROR_LABEL
    End If
End If

MATRIX_TRANSPOSE_FUNC = TEMP_MATRIX


Exit Function
ERROR_LABEL:
MATRIX_TRANSPOSE_FUNC = Err.Number
End Function

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'FUNCTION      : MATRIX_REVERSE_FUNC
'DESCRIPTION   : REVERSE THE ENTRIES IN AN ARRAY
'LIBRARY       : MATRIX
'GROUP         : RC_TRANSPOSE
'ID            : 002
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 01/19/2009
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function MATRIX_REVERSE_FUNC(ByRef DATA_RNG As Variant)
```

```
Dim i As Long
Dim j As Long

Dim SROW As Long
Dim SCOLUMN As Long

Dim NROWS As Long
Dim NCOLUMNS As Long

Dim TEMP_MATRIX As Variant
Dim DATA_MATRIX As Variant

On Error GoTo ERROR_LABEL

DATA_MATRIX = DATA_RNG

'---------------------------------------------------------------------------
If IS_MATRIX_FUNC(DATA_MATRIX) = True Then
'---------------------------------------------------------------------------
   NCOLUMNS = UBound(DATA_MATRIX, 2)
   NROWS = UBound(DATA_MATRIX, 1)

   SCOLUMN = LBound(DATA_MATRIX, 2)
   SROW = LBound(DATA_MATRIX, 1)

   ReDim TEMP_MATRIX(SROW To NROWS, SCOLUMN To NCOLUMNS)

   For j = SCOLUMN To NCOLUMNS
      For i = SROW To NROWS
         TEMP_MATRIX(NROWS + SROW - i, j) = DATA_MATRIX(i, j)
      Next i
   Next j
```

```
'------------------------------------------------------------------------
Else
'------------------------------------------------------------------------
   If IS_ARRAY_FUNC(DATA_MATRIX) Then

      SROW = LBound(DATA_MATRIX, 1)
      NROWS = UBound(DATA_MATRIX, 1)

      ReDim TEMP_MATRIX(SROW To NROWS, SROW To SROW)

      For j = SCOLUMN To NCOLUMNS
         For i = SROW To NROWS
            TEMP_MATRIX(NROWS + SROW - i, SROW) = DATA_MATRIX(i)
         Next i
      Next j
   Else
      GoTo ERROR_LABEL
   End If
'------------------------------------------------------------------------
End If
'------------------------------------------------------------------------

MATRIX_REVERSE_FUNC = TEMP_MATRIX

Exit Function
ERROR_LABEL:
MATRIX_REVERSE_FUNC = Err.Number
End Function
```

Módulo: OPTIM_GENETIC_PIKAIA_LIBR

```
Option Explicit     'Requires that all variables to be declared explicitly.


Option Base 1       'The "Option Base" statement allows to specify 0 or 1 as the
                    'default first index of arrays.

'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------

Private XDATA_ARR() As Double
'Array  x(1:n)  is the "fittest" (optimal) solution found,
'i.e., the solution which maximizes fitness function ff

Private PUB_Y_VAL As Double
'is the value of the fitness function at x

Private XTEMP_ARR() As Double
'temporary scratch array for x() to build and pass to ff

Private PUB_CONVERG_VAL As Integer
'is an indicator of the success or failure
'of the call to pikaia (0=success; non-zero=failure)

'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
Private Const NO_PARAM As Integer = 256 'is the maximum number of
'adjustable parameters

Private Const MAX_POP As Integer = 512
```

```
'maximum population (CTRL(1) <= MAX_POP

Private Const MAX_GENES As Integer = 6
'maximum number of Genes (digits) per Chromosome
'segement(Parameter)(CTRL(3) <= DMax)
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------

'for sub PIKAIA_REPORT_FUNC
Private PUB_BEST_FIT_VAL As Double
Private PUB_PMUTPV_VAL As Double
Private PUB_SEED_VAL As Long 'for random number generator
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------

Private PARAM_ARR() As Double
Private LOWER_ARR() As Double
Private UPPER_ARR() As Double

Private PUB_REPORT_GROUP As Variant
Private PUB_FUNC_STR_NAME As String
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------

'*******************************************************************************
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION      : PIKAIA_OPTIMIZATION_FUNC
```

```
'DESCRIPTION   : fully self-contained, general purpose optimization subroutine.
'The routine maximizes a user-supplied FORTRAN function, the name of which
'is passed in as an argument.

'PIKAIA (pronounced ``pee-kah-yah'') is a general purpose function optimization
'FORTRAN-77 subroutine based on a genetic algorithm. The subroutine is particularly
'useful (and robust) in treating multimodal optimization problems. The development
'of genetic algorithm-based inversion methods is but one aspect of the research
'in helioseismology carried out in the Solar Interior Section of the High Altitude
'Observatory, a scientific division of the National Center for Atmospheric Research
'in Boulder, Colorado.


'     PIKAIA is an optimization (maximization) of user-supplied "fitness" function
'     over n-dimensional parameter space  x  using a basic genetic algorithm method.
'
'     Genetic algorithms are heuristic search techniques that
'     incorporate in a computational setting, the biological notion
'     of evolution by means of natural selection.  This subroutine
'     implements the three basic operations of selection, crossover,
'     and mutation, operating on "genotypes" encoded as strings.
'
'     This version of the PIKAIA algorithm includes (1) two-point crossover,
'     (2) creep mutation, and (3) dynamical adjustment of the mutation rate
'     based on metric distance in parameter space.

'
'      o Integer  n  is the parameter space dimension, i.e., the number
'        of adjustable parameters.
'
'      o Function  ff  is a user-supplied scalar function of n vari-
```

```
'       ables, which must have the calling sequence f = ff(n,x), where
'       x is a real parameter array of length n.  This function must
'       be written so as to bound all parameters to the interval [0,1];
'       that is, the user must determine a priori bounds for the para-
'       meter space, and ff must use these bounds to perform the appro-
'       priate scalings to recover true parameter values in the
'       a priori ranges.
'
'       By convention, ff should return higher values for more optimal
'       parameter values (i.e., individuals which are more "fit").
'       For example, in fitting a function through data points, ff
'       could return the inverse of chi**2.
'
'       In most cases initialization code will have to be written
'       (either in a driver or in a separate subroutine) which loads
'       in data values and communicates with ff via one or more labeled
'       common blocks.  An example exercise driver and fitness function
'       are provided in the accompanying file, xpkaia.f.
'
'
'     Input/Output:
'      real CTRL(12)
'
'     o Array  CTRL  is an array of control flags and parameters, to
'       control the genetic behavior of the algorithm, and also printed
'       output.  A default value will be used for any control variable
'       which is supplied with a value less than zero.  On exit, CTRL
'       contains the actual values used as control variables.  The
'       elements of CTRL and their defaults are:
'
'          CTRL( 1) - number of individuals in a population (default
'                     is 100)
```

```
'       CTRL( 2) - number of generations over which solution is
'            to evolve (default is 500)
'       CTRL( 3) - number of significant digits (i.e., number of
'            genes) retained in chromosomal encoding (default
'            is 6)  (Note: This number is limited by the
'            machine floating point precision.  Most 32-bit
'            floating point representations have only 6 full
'            digits of precision.  To achieve greater preci-
'            sion this routine could be converted to double
'            precision, but note that this would also require
'            a double precision random number generator, which
'            likely would not have more than 9 digits of
'            precision if it used 4-byte integers internally.)
'       CTRL( 4) - crossover probability; must be  <= 1.0 (default
'            is 0.85). If crossover takes place, either one
'            or two splicing points are used, with equal
'            probabilities
'       CTRL( 5) - mutation mode; 1/2/3/4/5 (default is 2)
'            1=one-point mutation, fixed rate
'            2=one-point, adjustable rate based on fitness
'            3=one-point, adjustable rate based on distance
'            4=one-point+creep, fixed rate
'            5=one-point+creep, adjustable rate based on fitness
'            6=one-point+creep, adjustable rate based on distance
'       CTRL( 6) - initial mutation rate; should be small (default
'            is 0.005) (Note: the mutation rate is the proba-
'            bility that any one gene locus will mutate in
'            any one generation.)
'       CTRL( 7) - minimum mutation rate; must be >= 0.0 (default
'            is 0.0005)
'       CTRL( 8) - maximum mutation rate; must be <= 1.0 (default
'            is 0.25)
```

```
'          CTRL( 9) - relative fitness differential; range from 0
'                     (none) to 1 (maximum).  (default is 1.)
'          CTRL(10) - reproduction plan; 1/2/3=Full generational
'                     replacement/Steady-state-Replace-random/Steady-
'                     State - Replace - worst(Default Is 3)
'          CTRL(11) - elitism flag; 0/1=off/on (default is 0)
'                     (Applies only to reproduction plans 1 and 2)
'          CTRL(12) - printed output 0/1/2=None/Minimal/Verbose
'                     (Default Is 0)
'
'
' Output:
'      real x(n), f
'      integer   status
'
'      o Array  x(1:n)  is the "fittest" (optimal) solution found,
'        i.e., the solution which maximizes fitness function ff
'
'      o Scalar  f  is the value of the fitness function at x
'
'      o Integer  status  is an indicator of the success or failure
'        of the call to pikaia (0=success; non-zero=failure)


' References:
'       Charbonneau, Paul. "An introduction to gemetic algorithms for
'          numerical optimization", NCAR Technical Note TN-450+IA
'          (April 2002)
'       Charbonneau, Paul. "Release Notes for PIKAIA 1.2",
'          NCAR Technical Note TN-451+STR (April 2002)
'       Charbonneau, Paul, and Knapp, Barry. "A User's Guide
'          to PIKAIA 1.0" NCAR Technical Note TN-418+IA
'          (December 1995)
```

```
'       Goldberg, David E.  Genetic Algorithms in Search, Optimization,
'          & Machine Learning.  Addison-Wesley, 1989.
'       Davis, Lawrence, ed.  Handbook of Genetic Algorithms.
'          Van Nostrand Reinhold, 1991.

'LIBRARY      : OPTIMIZATION
'GROUP        : GENETIC_PIKAIA
'ID           : 001
'AUTHOR       : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function PIKAIA_OPTIMIZATION_FUNC(ByVal FUNC_STR_NAME As String, _
ByVal CONST_RNG As Variant, _
Optional ByVal TRACE_FLAG As Boolean = False, _
Optional ByRef ERROR_STR As String = "", _
Optional ByVal RND_NUMB_SEED As Long = 123456, _
Optional ByVal NUMB_INDIV_POPUP As Integer = 1, _
Optional ByVal NUMB_GENER_EVOL As Integer = 10, _
Optional ByVal NUMB_DIGITS_ENCODE As Integer = 4, _
Optional ByVal CROSS_PROBAB As Double = 0.85, _
Optional ByVal MUTAT_MODE As Integer = 2, _
Optional ByVal INIT_MUTAT_RATE As Double = 0.00001, _
Optional ByVal MIN_MUTAT_RATE As Double = 0.000005, _
Optional ByVal MAX_MUTAT_RATE As Double = 0.00001, _
Optional ByVal RELAT_FITNESS As Integer = 1, _
Optional ByVal REPROD_PLAN As Integer = 1, _
Optional ByVal ELITISM As Integer = 1, _
Optional ByVal OUTPUT As Integer = 0)

PUB_FUNC_STR_NAME = FUNC_STR_NAME
```

```
Dim i As Long
Dim j As Integer
Dim NSIZE As Integer
'the number of parameters in the fitness function
Dim PARAM_VECTOR As Variant
Dim TRACE_MATRIX As Variant
Dim CONST_BOX As Variant

On Error GoTo ERROR_LABEL

Dim CTRL_ARR(1 To 12) As Variant
'is an input/output array of control flags and parameters, to
'control the genetic behavior of the algorithm, and also printed
'output.  A default value will be used for any control variable
'which is supplied with a value less than zero.  On exit, CTRL
'contains the actual values used as control variables.

ERROR_STR = ""
CTRL_ARR(1) = NUMB_INDIV_POPUP
'number of individuals in a population (default is 100)
CTRL_ARR(2) = NUMB_GENER_EVOL
'number of generations over which solution is to evolve (default is 500)
CTRL_ARR(3) = NUMB_DIGITS_ENCODE
'number of significant digits (i.e., number of genes) retained in
'chromosomal encoding (default is 6)  (Note: This number is limited by the
'machine floating point precision.  Most 32-bit floating point representations
'have only 6 full digits of precision.  To achieve greater preci-
'sion this routine could be converted to double precision, but note that
'this would also require a double precision random number generator, which
'likely would not have more than 9 digits of precision if it used 4-byte
'integers internally.)
```

```
CTRL_ARR(4) = CROSS_PROBAB
'crossover probability; must be  <= 1.0 (default is 0.85). If crossover takes
'place, either one or two splicing points are used, with equal probabilities
CTRL_ARR(5) = MUTAT_MODE
'mutation mode; 1/2/3/4/5 (default is 2)
'1=one-point mutation, fixed rate
'2=one-point, adjustable rate based on fitness
'3=one-point, adjustable rate based on distance
'4=one-point+creep, fixed rate
'5=one-point+creep, adjustable rate based on fitness
'6=one-point+creep, adjustable rate based on distance
CTRL_ARR(6) = INIT_MUTAT_RATE
'initial mutation rate; should be small (default
'is 0.005) (Note: the mutation rate is the proba-
'bility that any one gene locus will mutate in
'any one generation.)
CTRL_ARR(7) = MIN_MUTAT_RATE
'minimum mutation rate; must be >= 0.0 (default is 0.0005)
CTRL_ARR(8) = MAX_MUTAT_RATE
'maximum mutation rate; must be <= 1.0 (default is 0.25)
CTRL_ARR(9) = RELAT_FITNESS
'relative fitness differential; range from 0
'(none) to 1 (maximum).  (default is 1.)
CTRL_ARR(10) = REPROD_PLAN
'reproduction plan; 1/2/3=Full generational
'replacement/Steady-state-Replace-random/Steady-
'State - Replace - worst(Default Is 3)
CTRL_ARR(11) = ELITISM
'elitism flag; 0/1=off/on (default is 0)
'(Applies only to reproduction plans 1 and 2)
CTRL_ARR(12) = OUTPUT
'printed output 0/1/2=None/Minimal/Verbose
```

```
For i = 1 To 11
   Select Case i
   Case 1
      If CTRL_ARR(i) > MAX_POP Or CTRL_ARR(i) < 2 Then
         ERROR_STR = "This population size must be between 2 and " & MAX_POP
         GoTo ERROR_LABEL
      End If
   Case 2
      If CTRL_ARR(i) < 1 Then
         ERROR_STR = "There must be at least one generation."
         GoTo ERROR_LABEL
      End If
   Case 3
      If CTRL_ARR(i) > MAX_GENES Or CTRL_ARR(i) < 1 Then
         ERROR_STR = "This number of digits for encoding must be between 1 and " & _
         MAX_GENES
         GoTo ERROR_LABEL
      End If
   Case 4
      If CTRL_ARR(i) < 0 Or CTRL_ARR(i) > 1 Then
         ERROR_STR = "This crossover probability must be between 0 and 1."
         GoTo ERROR_LABEL
      End If
   Case 5
      If CTRL_ARR(i) <> 1 And CTRL_ARR(i) <> 2 And CTRL_ARR(i) <> 3 And _
         CTRL_ARR(i) <> 4 And CTRL_ARR(i) <> 5 And CTRL_ARR(i) <> 6 Then
         ERROR_STR = "This mutation mode must be 1, 2, 3, 4, 5, or 6."
         GoTo ERROR_LABEL
      End If
   Case 6
   If CTRL_ARR(i) < 0 Or CTRL_ARR(i) > 1 Then
```

```
        ERROR_STR = "This initial mutation rate must be between 0 and 1."
        GoTo ERROR_LABEL
    End If
    Case 7
        If CTRL_ARR(i) < 0 Or CTRL_ARR(i) > 1 Then
        ERROR_STR = "This minimum mutation rate must be between 0 and 1."
        GoTo ERROR_LABEL
    End If
    Case 8
        If CTRL_ARR(i) < 0 Or CTRL_ARR(i) > 1 Then
        ERROR_STR = "This maximum mutation rate must be between 0 and 1."
        GoTo ERROR_LABEL
    End If
    Case 9
        If CTRL_ARR(i) < 0 Or CTRL_ARR(i) > 1 Then
        ERROR_STR = "This relative fitness differential must be between 0 and 1."
        GoTo ERROR_LABEL
    End If
    Case 10
        If CTRL_ARR(i) <> 1 And CTRL_ARR(i) <> 2 And CTRL_ARR(i) <> 3 Then
        ERROR_STR = "This reproduction plan must be 1, 2, or 3."
        GoTo ERROR_LABEL
    End If
    Case 11
        If CTRL_ARR(i) <> 0 And CTRL_ARR(i) <> 1 Then
        ERROR_STR = "Elitism must be 0 or 1."
        GoTo ERROR_LABEL
    End If
    End Select
Next i

CONST_BOX = CONST_RNG
```

```
NSIZE = UBound(CONST_BOX, 2)

If TRACE_FLAG = True Then
   ReDim TRACE_MATRIX(0 To CTRL_ARR(1) * (CTRL_ARR(2) + 1), 1 To NSIZE + 6)
End If

ReDim XDATA_ARR(1 To NSIZE) As Double
ReDim XTEMP_ARR(1 To NSIZE) As Double
ReDim LOWER_ARR(1 To NSIZE) As Double
ReDim UPPER_ARR(1 To NSIZE) As Double
ReDim PARAM_ARR(1 To NSIZE) As Double


For i = 1 To NSIZE
   LOWER_ARR(i) = CONST_BOX(1, i)
   UPPER_ARR(i) = CONST_BOX(2, i)
Next i

Call INIT_RND_GENER_FUNC(RND_NUMB_SEED)
ERROR_STR = ""
PUB_CONVERG_VAL = GENETIC_PIKAIA_OPTIM_FUNC(NSIZE, CTRL_ARR, _
            TRACE_MATRIX, ERROR_STR)

If ERROR_STR <> "" Then: GoTo ERROR_LABEL

'-------------------------------------------------------------------------------
If OUTPUT <> 0 Then
   PIKAIA_OPTIMIZATION_FUNC = PUB_REPORT_GROUP
   Exit Function
End If

'-------------------------------------------------------------------------------
```

```
If TRACE_FLAG = True Then
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
'These entries shows the value of 1-fitness during the evolution. This is a
'measure of the error (smaller values of 1-ff means less error in this example).
'For a different problem this chart can be changed to show either the fitness
'value or 1/fitness of any other measure of fitness.
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
   TRACE_MATRIX(0, NSIZE + 5) = "1/Fitness"
   TRACE_MATRIX(0, NSIZE + 6) = "1-Fitness"
   For i = 1 To CTRL_ARR(1) * (CTRL_ARR(2) + 1) 'optimum value
      TRACE_MATRIX(i, NSIZE + 5) = 1 / TRACE_MATRIX(i, NSIZE + 4) '1 / Fitness
      TRACE_MATRIX(i, NSIZE + 6) = 1 - TRACE_MATRIX(i, NSIZE + 4) '1 - Fitness
   Next i
   PIKAIA_OPTIMIZATION_FUNC = TRACE_MATRIX
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
Else
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
   ReDim PARAM_VECTOR(1 To NSIZE, 1 To 1)
   'PARAM_VECTOR(0, 1) = PUB_Y_VAL
   For j = 1 To NSIZE
      PARAM_VECTOR(j, 1) = PIKAI_PARAM_SCALE_FUNC(j, XDATA_ARR(j))
   Next j
   PIKAIA_OPTIMIZATION_FUNC = PARAM_VECTOR
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
End If
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
```

```
Exit Function
ERROR_LABEL:
PIKAIA_OPTIMIZATION_FUNC = PUB_CONVERG_VAL
End Function


'*********************************************************************************
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION      : GENETIC_PIKAIA_OPTIM_FUNC

'DESCRIPTION   :

'PIKAIA incorporates only the two basic genetic operators: uniform one-point
'crossover, and uniform one-point mutation. Unlike many GA packages available
'commercially or in the public domain, the encoding within PIKAIA is based on
'a decimal alphabet made of the 10 simple integers (0 through 9); this is because
'binary operations are usually carried out through platform-dependent functions
'in FORTRAN. Three reproduction plans are available: Full generational replacement,
'Steady-State-Delete-Random, and Steady-State-Delete-Worst. Elitism
'is available and is a default option. The mutation rate can be dynamically
'controlled by monitoring the difference in fitness between the current best and
'median in the population (also a default option). Selection is rank-based and
'stochastic, making use of the Roulette

'Wheel Algorithm.
'PIKAIA is supplied with a ranking subroutine based on the Quicksort algorithm,
```

```
'and a random number generator based on the minimal standard Lehmer multiplicative
'linear congruential generator of Park and Miller (1988, Communications of the
'ACM, 31, 1192).

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 002
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************
'*****************************************************************************

Private Function GENETIC_PIKAIA_OPTIM_FUNC(ByVal NSIZE As Integer, _
ByRef CTRL_ARR As Variant, _
Optional ByRef TRACE_MATRIX As Variant, _
Optional ByRef ERROR_STR As String = "")

Dim i As Long
Dim j As Integer
Dim k As Integer

Dim ND_INT As Integer
Dim NPOP_INT As Integer
Dim NGEN_INT As Integer

Dim IMUT_INT As Integer
Dim IREP_INT As Integer
Dim IELITE_INT As Integer

Dim IP1_INT As Integer
Dim IP2_INT As Integer
```

```
Dim IVRB_INT As Integer
Dim IPOP_INT As Integer
Dim IGEN_INT As Integer

Dim NEWW_INT As Integer
Dim NEWTOT_INT As Integer

Dim FDIF_VAL As Double
Dim PMUT_VAL As Double
Dim PMUTMN_VAL As Double
Dim PMUTMX_VAL As Double
Dim PCROSS_VAL As Double

On Error GoTo ERROR_LABEL

Dim IFIT_ARR(1 To MAX_POP) As Integer
Dim JFIT_ARR(1 To MAX_POP) As Integer

Dim FITNS_ARR(1 To MAX_POP) As Double
Dim PH_MAT(1 To NO_PARAM, 1 To 2) As Double

Dim OLDPH_MAT(1 To NO_PARAM, 1 To MAX_POP) As Double
Dim NEWPH_MAT(1 To NO_PARAM, 1 To MAX_POP) As Double

Dim GN1_ARR(1 To NO_PARAM * MAX_GENES) As Integer
Dim GN2_ARR(1 To NO_PARAM * MAX_GENES) As Integer

'Set control variables from input and defaults

PUB_CONVERG_VAL = PIKAIA_CTRL_FUNC(CTRL_ARR, NSIZE, NPOP_INT, NGEN_INT, _
        ND_INT, PCROSS_VAL, PMUTMN_VAL, PMUTMX_VAL, _
        PMUT_VAL, IMUT_INT, FDIF_VAL, IREP_INT, IELITE_INT, _
```

```
        IVRB_INT, ERROR_STR)

If (PUB_CONVERG_VAL <> 0) Then 'Program stopped because control vector (CTRL)
'argument(s) invalid
   GoTo ERROR_LABEL
End If

'Make sure locally-dimensioned arrays are big enough

If (NSIZE > NO_PARAM Or NPOP_INT > MAX_POP Or ND_INT > MAX_GENES) Then
   ERROR_STR = "Program stopped because number of parameters, " & _
   "population, or genes too large"
   PUB_CONVERG_VAL = -1
   GoTo ERROR_LABEL
End If



If IsArray(TRACE_MATRIX) = True Then
'gp set the header row and pointer for output of the populations
'during each generation
   TRACE_MATRIX(0, 1) = "Generation"
   TRACE_MATRIX(0, 2) = "Rank"
   TRACE_MATRIX(0, 3) = "Quantile"
   For k = 1 To NSIZE
      TRACE_MATRIX(0, 3 + k) = "X(" & k & ")"
   Next k
   TRACE_MATRIX(0, 3 + NSIZE + 1) = "Fitness"
End If

'Compute initial (random but bounded) phenotypes

For IPOP_INT = 1 To NPOP_INT
```

```
    For k = 1 To NSIZE
        OLDPH_MAT(k, IPOP_INT) = NEXT_PSEUDO_RND_FUNC()
        XTEMP_ARR(k) = OLDPH_MAT(k, IPOP_INT)
    Next k
    FITNS_ARR(IPOP_INT) = PIKAIA_OBJ_FUNC(NSIZE, XTEMP_ARR())
Next IPOP_INT


'Rank initial population by fitness order
Call PIKAIA_RANK_FUNC(NPOP_INT, FITNS_ARR, IFIT_ARR, JFIT_ARR)

If IsArray(TRACE_MATRIX) = True Then
    'output of ranked populations for each generation
    i = 1
    For j = NPOP_INT To 1 Step -1
        TRACE_MATRIX(i, 1) = 0 'generation number 0 for initial population
        TRACE_MATRIX(i, 2) = j 'rank
        TRACE_MATRIX(i, 3) = j / (NPOP_INT + 1) 'quantile

        For k = 1 To NSIZE
            TRACE_MATRIX(i, 3 + k) = OLDPH_MAT(k, IFIT_ARR(j))
            'ranked phenomes of this individual
        Next k
        TRACE_MATRIX(i, 3 + NSIZE + 1) = FITNS_ARR(IFIT_ARR(j))
        'fitness of that individual
        i = i + 1
    Next j
End If

'main generation loop
For IGEN_INT = 1 To NGEN_INT
```

```
Dim Percent As Single
'updating of progress bar
Percent = (IGEN_INT / NGEN_INT) * 100
progress Percent

   NEWTOT_INT = 0
   For IPOP_INT = 1 To NPOP_INT / 2
      'pick two parents
      Call PIKAIA_PARENT_FUNC(NPOP_INT, JFIT_ARR, FDIF_VAL, IP1_INT) 'pick dad
21:
      Call PIKAIA_PARENT_FUNC(NPOP_INT, JFIT_ARR, FDIF_VAL, IP2_INT) 'pick mom
      If (IP1_INT = IP2_INT) Then GoTo 21 'no breeding with yourself!

      'encode parent phenotypes
      For k = 1 To NSIZE
         XTEMP_ARR(k) = OLDPH_MAT(k, IP1_INT)
      Next k

      Call PIKAIA_ENCODE_FUNC(NSIZE, ND_INT, XTEMP_ARR, GN1_ARR)
      For k = 1 To NSIZE
         XTEMP_ARR(k) = OLDPH_MAT(k, IP2_INT)
      Next k

      Call PIKAIA_ENCODE_FUNC(NSIZE, ND_INT, XTEMP_ARR, GN2_ARR)

      'breed
      Call PIKAIA_CROSS_FUNC(NSIZE, ND_INT, PCROSS_VAL, GN1_ARR, GN2_ARR)
      Call PIKAIA_MUTATE_FUNC(NSIZE, ND_INT, PMUT_VAL, GN1_ARR, IMUT_INT)
      Call PIKAIA_MUTATE_FUNC(NSIZE, ND_INT, PMUT_VAL, GN2_ARR, IMUT_INT)
      Call PIKAIA_DECODE_FUNC(NSIZE, ND_INT, GN1_ARR, XTEMP_ARR)

      For k = 1 To NSIZE
```

```
      PH_MAT(k, 1) = XTEMP_ARR(k)
   Next k

   'decode offspring genotypes
   Call PIKAIA_DECODE_FUNC(NSIZE, ND_INT, GN2_ARR, XTEMP_ARR)
   For k = 1 To NSIZE
      PH_MAT(k, 2) = XTEMP_ARR(k)
   Next k

   'insert into population
   If (IREP_INT = 1) Then
      Call PIKAIA_GENREP_FUNC(NO_PARAM, NSIZE, NPOP_INT, _
          IPOP_INT, PH_MAT, NEWPH_MAT)
   Else
      Call PIKAIA_STDREP_FUNC(NO_PARAM, NSIZE, NPOP_INT, _
         IREP_INT, IELITE_INT, PH_MAT, _
         OLDPH_MAT, FITNS_ARR, IFIT_ARR, JFIT_ARR, NEWW_INT)
         NEWTOT_INT = NEWTOT_INT + NEWW_INT
   End If
Next IPOP_INT

'if running full generational replacement: swap populations
If (IREP_INT = 1) Then
   Call PIKAIA_NEWPOP_FUNC(IELITE_INT, NO_PARAM, NSIZE, _
      NPOP_INT, OLDPH_MAT, NEWPH_MAT, _
      IFIT_ARR, JFIT_ARR, FITNS_ARR, NEWTOT_INT)
End If

If (IMUT_INT = 2 Or IMUT_INT = 3 Or IMUT_INT = 5 Or IMUT_INT = 6) Then
'adjust mutation rate?
   PUB_CONVERG_VAL = VARIAB_MUTAT_RATE_FUNC(NO_PARAM, NSIZE, _
            NPOP_INT, OLDPH_MAT, FITNS_ARR, _
```

```
                IFIT_ARR, PMUTMN_VAL, PMUTMX_VAL, _
                PMUT_VAL, IMUT_INT, ERROR_STR)
        If PUB_CONVERG_VAL <> 0 Then: GoTo ERROR_LABEL
    End If



    If (IVRB_INT > 0) Then
        If IGEN_INT = 1 Then: ReDim PUB_REPORT_GROUP(1 To NGEN_INT)
        PUB_REPORT_GROUP(IGEN_INT) = PIKAIA_REPORT_FUNC(IVRB_INT, NO_PARAM, _
            NSIZE, NPOP_INT, ND_INT, OLDPH_MAT, FITNS_ARR, _
            IFIT_ARR, PMUT_VAL, IGEN_INT, NEWTOT_INT)
    End If


    If IsArray(TRACE_MATRIX) = True Then
  'output of ranked populations for each generation
        For j = NPOP_INT To 1 Step -1
            TRACE_MATRIX(i, 1) = IGEN_INT 'generation number
            TRACE_MATRIX(i, 2) = j 'rank
            TRACE_MATRIX(i, 3) = j / (NPOP_INT + 1) 'quantile
            For k = 1 To NSIZE
                TRACE_MATRIX(i, 3 + k) = OLDPH_MAT(k, IFIT_ARR(j))
                'ranked phenomes of this individual
            Next k
            TRACE_MATRIX(i, 3 + NSIZE + 1) = FITNS_ARR(IFIT_ARR(j))
            'fitness of that individual
            i = i + 1
        Next j
    End If
'End of Main Generation Loop
Next IGEN_INT


'Return best phenotype and its fitness
```

```
For k = 1 To NSIZE
    XDATA_ARR(k) = OLDPH_MAT(k, IFIT_ARR(NPOP_INT))
Next k
PUB_Y_VAL = FITNS_ARR(IFIT_ARR(NPOP_INT))

GENETIC_PIKAIA_OPTIM_FUNC = PUB_CONVERG_VAL

Exit Function
ERROR_LABEL:
GENETIC_PIKAIA_OPTIM_FUNC = PUB_CONVERG_VAL
End Function

'*************************************************************************************
'*************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*************************************************************************************
'*************************************************************************************
'FUNCTION      : PIKAIA_CTRL_FUNC
'DESCRIPTION   : Set control variables and flags from input and defaults
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 003
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*************************************************************************************
'*************************************************************************************

Private Function PIKAIA_CTRL_FUNC(ByRef CTRL_ARR As Variant, _
ByRef NSIZE As Integer, _
ByRef NPOP_INT As Integer, _
```

```
ByRef NGEN_INT As Integer, _
ByRef ND_INT As Integer, _
ByRef PCROSS_VAL As Double, _
ByRef PMUTMN_VAL As Double, _
ByRef PMUTMX_VAL As Double, _
ByRef PMUT_VAL As Double, _
ByRef IMUT_INT As Integer, _
ByRef FDIF_VAL As Double, _
ByRef IREP_INT As Integer, _
ByRef IELITE_INT As Integer, _
ByRef IVRB_INT As Integer, _
ByRef ERROR_STR As String)

Dim i As Integer

On Error GoTo ERROR_LABEL

Dim DEFAULT_ARR(1 To 12) As Variant
'for sub setctl to set defaults for CTRL array

ERROR_STR = ""
DEFAULT_ARR(1) = 100
DEFAULT_ARR(2) = 500
DEFAULT_ARR(3) = 5
DEFAULT_ARR(4) = 0.85
DEFAULT_ARR(5) = 2
DEFAULT_ARR(6) = 0.005
DEFAULT_ARR(7) = 0.0005
DEFAULT_ARR(8) = 0.25
DEFAULT_ARR(9) = 1
DEFAULT_ARR(10) = 1
DEFAULT_ARR(11) = 1
```

```
DEFAULT_ARR(12) = 0
'IVRB_INT for PIKAIA Genetic Algorithm Report: 0=off, 1=on (normal default is
'0 for off)

For i = 1 To 12
   If (CTRL_ARR(i) < 0) Then CTRL_ARR(i) = DEFAULT_ARR(i)
Next i

NPOP_INT = CTRL_ARR(1)
NGEN_INT = CTRL_ARR(2)
ND_INT = CTRL_ARR(3)
PCROSS_VAL = CTRL_ARR(4)
IMUT_INT = CTRL_ARR(5)
PMUT_VAL = CTRL_ARR(6)
PMUTMN_VAL = CTRL_ARR(7)
PMUTMX_VAL = CTRL_ARR(8)
FDIF_VAL = CTRL_ARR(9)
IREP_INT = CTRL_ARR(10)
IELITE_INT = CTRL_ARR(11)
IVRB_INT = CTRL_ARR(12)
PUB_CONVERG_VAL = 0

'Check some control values

If (IMUT_INT <> 1 And IMUT_INT <> 2 And IMUT_INT <> 3 And IMUT_INT <> 4 And _
   IMUT_INT <> 5 And IMUT_INT <> 6) Then
   ERROR_STR = "ERROR: illegal value for IMUT_INT (CTRL(5))"
   PUB_CONVERG_VAL = 5
End If

If (FDIF_VAL > 1) Then
   ERROR_STR = "ERROR: illegal value for FDIF_VAL (CTRL(9))"
```

```
    PUB_CONVERG_VAL = 9
End If

If (IREP_INT <> 1 And IREP_INT <> 2 And IREP_INT <> 3) Then
    ERROR_STR = "ERROR: illegal value for IREP_INT (CTRL(10))"
    PUB_CONVERG_VAL = 10
End If

If (PCROSS_VAL > 1# Or PCROSS_VAL < 0) Then
    ERROR_STR = "ERROR: illegal value for PCROSS_VAL (CTRL(4))"
    PUB_CONVERG_VAL = 4
End If

If (IELITE_INT <> 0 And IELITE_INT <> 1) Then
    ERROR_STR = "ERROR: illegal value for IELITE_INT (CTRL(11))"
    PUB_CONVERG_VAL = 11
End If

If (IREP_INT = 1 And IMUT_INT = 1 And PMUT_VAL > 0.5 And IELITE_INT = 0) Then
    ERROR_STR = _
    "WARNING: dangerously high value for PMUT_VAL (CTRL(6)). " & _
    "Should enforce elitism with CTRL(11)=1"
End If

If (IREP_INT = 1 And IMUT_INT = 2 And PMUTMX_VAL > 0.5 And IELITE_INT = 0) Then
    ERROR_STR = _
    "WARNING: dangerously high value for PMUTMX_VAL (CTRL(8)). " & _
    "Should enforce elitism with CTRL(11)=1"
End If

If (FDIF_VAL < 0.33 And IREP_INT <> 3) Then
    ERROR_STR = _
```

```
    "WARNING: dangerously low value of FDIF_VAL (CTRL(9))"
End If

If NPOP_INT Mod 2 > 0 Then
    NPOP_INT = NPOP_INT - 1
    ERROR_STR = _
    "WARNING: decreasing population size (CTRL(1)) to npop=" & NPOP_INT
End If

PIKAIA_CTRL_FUNC = PUB_CONVERG_VAL

Exit Function
ERROR_LABEL:
PIKAIA_CTRL_FUNC = PUB_CONVERG_VAL
End Function

'*********************************************************************************************
'*********************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************************
'*********************************************************************************************
'FUNCTION      : VARIAB_MUTAT_RATE_FUNC
'DESCRIPTION   : Implements variable mutation rate
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 004
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*********************************************************************************************
'*********************************************************************************************
```

```
Private Function VARIAB_MUTAT_RATE_FUNC(ByRef NO_PARAM As Integer, _
ByRef NSIZE As Integer, _
ByRef NPOP_INT As Integer, _
ByRef OLDPH_MAT As Variant, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant, _
ByRef PMUTMN_VAL As Double, _
ByRef PMUTMX_VAL As Double, _
ByRef PMUT_VAL As Double, _
ByRef IMUT_INT As Integer, _
ByRef ERROR_STR As String)

'dynamical adjustment of mutation rate;
'IMUT_INT=2 or IMUT_INT=5 : adjustment based on fitness differential
'between best and median individuals
'IMUT_INT=3 or IMUT_INT=6 : adjustment based on metric distance
'between best and median individuals

Dim i As Integer
Dim RDIF_VAL As Double

On Error GoTo ERROR_LABEL

Const RDIFLO_VAL As Double = 0.05
Const RDIFHI_VAL As Double = 0.25
Const DELTA_VAL As Double = 1.5

ERROR_STR = ""
PUB_CONVERG_VAL = 0
' Adjustment based on fitness differential
If (IMUT_INT = 2 Or IMUT_INT = 5) Then
```

```
    If FITNS_ARR(IFIT_ARR(NPOP_INT)) + FITNS_ARR(IFIT_ARR(NPOP_INT / 2)) = 0 Then
        ERROR_STR = "Invalid fitness function"
        PUB_CONVERG_VAL = -1
        GoTo ERROR_LABEL
    End If
    RDIF_VAL = Abs(FITNS_ARR(IFIT_ARR(NPOP_INT)) - _
          FITNS_ARR(IFIT_ARR(NPOP_INT / 2))) / _
         (FITNS_ARR(IFIT_ARR(NPOP_INT)) + FITNS_ARR(IFIT_ARR(NPOP_INT / 2)))
ElseIf (IMUT_INT = 3 Or IMUT_INT = 6) Then
    RDIF_VAL = 0
    For i = 1 To NSIZE
        RDIF_VAL = RDIF_VAL + (OLDPH_MAT(i, IFIT_ARR(NPOP_INT)) - _
             OLDPH_MAT(i, IFIT_ARR(NPOP_INT / 2))) ^ 2
    Next i
    RDIF_VAL = Sqr(RDIF_VAL) / NSIZE
End If

'Adjustment based on normalized metric distance
If (RDIF_VAL <= RDIFLO_VAL) Then
    PMUT_VAL = MINIMUM_FUNC(PMUTMX_VAL, PMUT_VAL * DELTA_VAL)
ElseIf (RDIF_VAL >= RDIFHI_VAL) Then
    PMUT_VAL = MAXIMUM_FUNC(PMUTMN_VAL, PMUT_VAL / DELTA_VAL)
End If

VARIAB_MUTAT_RATE_FUNC = PUB_CONVERG_VAL

Exit Function
ERROR_LABEL:
VARIAB_MUTAT_RATE_FUNC = PUB_CONVERG_VAL
End Function

'*******************************************************************************
```

```
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION      : PIKAIA_REPORT_FUNC
'DESCRIPTION   : Write generation report to standard output
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 005
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*********************************************************************************
'*********************************************************************************

Private Function PIKAIA_REPORT_FUNC(ByRef IVRB_INT As Integer, _
ByRef NO_PARAM As Integer, _
ByRef NSIZE As Integer, _
ByRef NPOP_INT As Integer, _
ByRef ND_INT As Integer, _
ByRef OLDPH_MAT As Variant, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant, _
ByRef PMUT_VAL As Double, _
ByRef IGEN_INT As Integer, _
ByRef NNEW_INT As Integer)

Dim k As Long
Dim NDPWR_INT As Long
Dim RPT_FLAG As Boolean
Dim GENOT_MAT As Variant
```

```
On Error GoTo ERROR_LABEL

PUB_BEST_FIT_VAL = 0
PUB_PMUTPV_VAL = 0
RPT_FLAG = False

If (PMUT_VAL <> PUB_PMUTPV_VAL) Then
   PUB_PMUTPV_VAL = PMUT_VAL
   RPT_FLAG = True
End If

If (FITNS_ARR(IFIT_ARR(NPOP_INT)) <> PUB_BEST_FIT_VAL) Then
   PUB_BEST_FIT_VAL = FITNS_ARR(IFIT_ARR(NPOP_INT))
   RPT_FLAG = True
End If

If (RPT_FLAG Or IVRB_INT >= 2) Then 'Power of 10 to make integer
'genotypes for display
 NDPWR_INT = Round(10 ^ ND_INT)
 ReDim GENO_MAT(1 To NSIZE + 1, 1 To 3)
 GENO_MAT(1, 1) = FITNS_ARR(IFIT_ARR(NPOP_INT)) 'igen
 GENO_MAT(1, 2) = FITNS_ARR(IFIT_ARR(NPOP_INT - 1)) 'nnew
 GENO_MAT(1, 3) = FITNS_ARR(IFIT_ARR(NPOP_INT / 2)) 'pmut
 For k = 1 To NSIZE
   GENO_MAT(k + 1, 1) = Round(NDPWR_INT * OLDPH_MAT(k, IFIT_ARR(NPOP_INT)))
   GENO_MAT(k + 1, 2) = Round(NDPWR_INT * OLDPH_MAT(k, IFIT_ARR(NPOP_INT)))
   GENO_MAT(k + 1, 3) = Round(NDPWR_INT * OLDPH_MAT(k, IFIT_ARR(NPOP_INT)))
 Next k
End If

PIKAIA_REPORT_FUNC = GENO_MAT
```

```
Exit Function

ERROR_LABEL:

PIKAIA_REPORT_FUNC = Err.Number

End Function


'--------------------------------------------------------------------------------


'********************************************************************************

'********************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca

'********************************************************************************

'********************************************************************************

'FUNCTION      : PIKAIA_ENCODE_FUNC

'DESCRIPTION   : Encodes phenotype into genotype

'LIBRARY       : OPTIMIZATION

'GROUP         : GENETIC_PIKAIA

'ID            : 006

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'********************************************************************************

'********************************************************************************


Private Function PIKAIA_ENCODE_FUNC(ByRef NSIZE As Integer, _

ByRef ND_INT As Integer, _

ByRef PH_MAT As Variant, _

ByRef GN_ARR As Variant)


Dim i As Long

Dim j As Long
```

```
Dim ii As Long
Dim jj As Long

Dim Z_VAL As Double

On Error GoTo ERROR_LABEL

PIKAIA_ENCODE_FUNC = False

'encode phenotype parameters into integer genotype
'PH_MAT(k) are x,y coordinates [ 0 < x,y < 1 ]

Z_VAL = 10 ^ ND_INT
ii = 0
For i = 1 To NSIZE
   jj = Int(PH_MAT(i) * Z_VAL)
   For j = ND_INT To 1 Step -1
      GN_ARR(ii + j) = jj Mod 10
      jj = Int(jj / 10)
         'gp debug add Int to force VBA not to round to nearest value

   Next j
   ii = ii + ND_INT
Next i

PIKAIA_ENCODE_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_ENCODE_FUNC = False
End Function
```

```
'*******************************************************************************
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION      : PIKAIA_DECODE_FUNC
'DESCRIPTION   : decodes genotype into phenotype
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 007
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*******************************************************************************
'*******************************************************************************

Private Function PIKAIA_DECODE_FUNC(ByRef NSIZE As Integer, _
ByRef ND_INT As Integer, _
ByRef GN_ARR As Variant, _
ByRef PH_MAT As Variant)

'decode genotype into phenotype parameters
'PH_MAT(k) are x,y coordinates [ 0 < x,y < 1 ]

Dim i As Long
Dim j As Long
Dim ii As Long
Dim jj As Long
Dim Z_VAL As Double
```

```
On Error GoTo ERROR_LABEL

PIKAIA_DECODE_FUNC = False

Z_VAL = 10 ^ (-ND_INT)
ii = 0
For i = 1 To NSIZE
   jj = 0
   For j = 1 To ND_INT
      jj = Int(10 * jj + GN_ARR(ii + j))
         'gp add Int for force VBA not to round

   Next j
   PH_MAT(i) = jj * Z_VAL
   ii = ii + ND_INT
Next i

PIKAIA_DECODE_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_DECODE_FUNC = False
End Function

'*******************************************************************************************
'*******************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************************
'*******************************************************************************************
'FUNCTION      : PIKAIA_CROSS_FUNC
```

```
'DESCRIPTION   : Breeds two offspring from two parents
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 008
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************************
'*****************************************************************************************

Private Function PIKAIA_CROSS_FUNC(ByRef NSIZE As Integer, _
ByRef ND_INT As Integer, _
ByRef PCROSS_VAL As Double, _
ByRef GN1_ARR As Variant, _
ByRef GN2_ARR As Variant)

'breeds two parent chromosomes into two offspring chromosomes
'breeding occurs through crossover. If the crossover probability
'test yields true (crossover taking place), either one-point or
'two-point crossover is used, with equal probabilities.
'Compatibility with version 1.0: To enforce 100% use of one-point
'crossover, un-comment appropriate line in source code below

Dim h As Integer
Dim i As Integer
Dim j As Integer '
Dim k As Integer
Dim l As Integer

On Error GoTo ERROR_LABEL

PIKAIA_CROSS_FUNC = False
```

```
'Use crossover probability to decide whether a crossover occurs
If (NEXT_PSEUDO_RND_FUNC() < PCROSS_VAL) Then
'Compute first crossover point
   j = Int(NEXT_PSEUDO_RND_FUNC() * NSIZE * ND_INT) + 1
   ' Now choose between one-point and two-point crossover
   If (NEXT_PSEUDO_RND_FUNC() < 0.5) Then
      k = NSIZE * ND_INT
   Else
      k = Int(NEXT_PSEUDO_RND_FUNC() * NSIZE * ND_INT) + 1
      'Un-comment following line to enforce one-point crossover
         If (k < j) Then
            l = k
            k = j
            j = l
         End If
   End If
   ' Swap genes from j to k
   For i = j To k
      h = GN2_ARR(i)
      GN2_ARR(i) = GN1_ARR(i)
      GN1_ARR(i) = h
   Next i
End If

PIKAIA_CROSS_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_CROSS_FUNC = False
End Function

'*******************************************************************************
```

```
'*****************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*****************************************************************************************
'*****************************************************************************************
'FUNCTION     : PIKAIA_MUTATE_FUNC
'DESCRIPTION   : Introduces random mutation in a genotype
'LIBRARY      : OPTIMIZATION
'GROUP        : GENETIC_PIKAIA
'ID           : 009
'AUTHOR       : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************************
'*****************************************************************************************

Private Function PIKAIA_MUTATE_FUNC(ByRef NSIZE As Integer, _
ByRef ND_INT As Integer, _
ByRef PMUT_VAL As Double, _
ByRef GN_ARR As Variant, _
ByRef IMUT_INT As Integer)

'Mutations occur at rate PMUT_VAL at all gene loci
'IMUT_INT=1    Uniform mutation, constant rate
'IMUT_INT=2    Uniform mutation, variable rate based on fitness
'IMUT_INT=3    Uniform mutation, variable rate based on distance
'IMUT_INT=4    Uniform or creep mutation, constant rate
'IMUT_INT=5    Uniform or creep mutation, variable rate based on
'fitness
'IMUT_INT=6    Uniform or creep mutation, variable rate based on
'distance
```

```
Dim i As Integer

Dim j As Integer

Dim k As Integer

Dim l As Integer


Dim ii As Integer

Dim jj As Integer

Dim kk As Integer


On Error GoTo ERROR_LABEL


PIKAIA_MUTATE_FUNC = False


'Decide which type of mutation is to occur



If (IMUT_INT >= 4 And NEXT_PSEUDO_RND_FUNC() <= 0.5) Then

  'CREEP MUTATION OPERATOR

  'Subject each locus to random +/- 1 increment at the rate PMUT_VAL


    For i = 1 To NSIZE

       For j = 1 To ND_INT

       'Construct integer

          If (NEXT_PSEUDO_RND_FUNC() < PMUT_VAL) Then

             kk = (i - 1) * ND_INT + j

             jj = Round(NEXT_PSEUDO_RND_FUNC()) * 2 - 1

             ii = (i - 1) * ND_INT + 1

             GN_ARR(kk) = GN_ARR(kk) + jj

             If (jj < 0 And GN_ARR(kk) < 0) Then

       'This is where we carry over the one (up to two digits)

       'first take care of decrement below 0 case

                If (j = 1) Then
```

```
            GN_ARR(kk) = 0
         Else
            For k = kk To ii + 1 Step -1
               GN_ARR(k) = 9
               GN_ARR(k - 1) = GN_ARR(k - 1) - 1
               If (GN_ARR(k - 1) >= 0) Then GoTo 4
            Next k
            If (GN_ARR(ii) < 0) Then
               For l = ii To kk
                  GN_ARR(l) = 0
               Next l
            End If
4:
         End If
      End If
      'we popped under 0.00000 lower bound; fix it up
      If (jj > 0 And GN_ARR(kk) > 9) Then
         If (j = 1) Then
            GN_ARR(kk) = 9
         Else
            For k = kk To ii + 1 Step -1
               GN_ARR(k) = 0
               GN_ARR(k - 1) = GN_ARR(k - 1) + 1
               If (GN_ARR(k - 1) <= 9) Then GoTo 7
            Next k
            'we popped over 9.99999 upper bound; fix it up
            If (GN_ARR(ii) > 9) Then
               For l = ii To kk
                  GN_ARR(l) = 9
               Next l
            End If
7:
```

```
                End If
            End If
        End If
    Next j
  Next i
Else
 'UNIFORM MUTATION OPERATOR
 'Subject each locus to random mutation at the rate PMUT_VAL

  For i = 1 To NSIZE * ND_INT
    If (NEXT_PSEUDO_RND_FUNC() < PMUT_VAL) Then
      GN_ARR(i) = Int(NEXT_PSEUDO_RND_FUNC() * 10)
    End If
  Next i
End If

PIKAIA_MUTATE_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_MUTATE_FUNC = False
End Function

'*******************************************************************************
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION     : PIKAIA_PARENT_FUNC
'DESCRIPTION  :
```

```
'Selects a parent from the population, using roulette wheel
'algorithm with the relative fitnesses of the phenotypes as
'the "hit" probabilities [see Davis 1991, chap. 1].

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 010
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************
'*****************************************************************************

Private Function PIKAIA_PARENT_FUNC(ByRef NPOP_INT As Integer, _
ByRef JFIT_ARR As Variant, _
ByRef FDIF_VAL As Double, _
ByRef IP_INT As Integer)

Dim i As Integer
Dim j As Integer

Dim DICE_VAL As Double
Dim FIT_VAL As Double

On Error GoTo ERROR_LABEL

PIKAIA_PARENT_FUNC = False

j = NPOP_INT + 1
DICE_VAL = NEXT_PSEUDO_RND_FUNC() * NPOP_INT * j
FIT_VAL = 0

For i = 1 To NPOP_INT
```

```
    FIT_VAL = FIT_VAL + j + FDIF_VAL * (j - 2 * JFIT_ARR(i))
    If (FIT_VAL >= DICE_VAL) Then
        IP_INT = i
        GoTo 2
    End If
Next i
2:
'Assert: loop will never exit by falling through

PIKAIA_PARENT_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_PARENT_FUNC = False
End Function

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'FUNCTION      : PIKAIA_RANK_FUNC
'DESCRIPTION   : Ranks initial population
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 011
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
Private Function PIKAIA_RANK_FUNC(ByRef NPOP_INT As Integer, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant, _
ByRef JFIT_ARR As Variant)

'Calls external sort routine to produce key index and rank order
'of input array FITNS_ARR (which is not altered).

Dim i As Integer

On Error GoTo ERROR_LABEL

PIKAIA_RANK_FUNC = False

Call PIKAIA_SORT_FUNC(NPOP_INT, FITNS_ARR, IFIT_ARR)
'External sort subroutine external PIKAIA_SORT_FUNC
'Compute the key index
'...and the rank order
For i = 1 To NPOP_INT
   JFIT_ARR(IFIT_ARR(i)) = NPOP_INT - i + 1
Next i

PIKAIA_RANK_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_RANK_FUNC = False
End Function

'*******************************************************************************
'*******************************************************************************
```

```
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION      : PIKAIA_GENREP_FUNC
'DESCRIPTION   : Full generational replacement: accumulate offspring into new
'population array. Inserts offspring into population, for full generational replacement

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 012
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*******************************************************************************
'*******************************************************************************

Private Function PIKAIA_GENREP_FUNC(ByRef NO_PARAM As Integer, _
ByRef NSIZE As Integer, _
ByRef NPOP_INT As Integer, _
ByRef IPOP_INT As Integer, _
ByRef PH_MAT As Variant, _
ByRef NEWPH_MAT As Variant)

Dim i As Integer
Dim j As Integer
Dim k As Integer

On Error GoTo ERROR_LABEL

PIKAIA_GENREP_FUNC = False
```

```
'Insert one offspring pair into new population
i = 2 * IPOP_INT - 1
j = i + 1
For k = 1 To NSIZE
   NEWPH_MAT(k, i) = PH_MAT(k, 1)
   NEWPH_MAT(k, j) = PH_MAT(k, 2)
Next k

PIKAIA_GENREP_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_GENREP_FUNC = False
End Function

'*********************************************************************************
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION     : PIKAIA_STDREP_FUNC
'DESCRIPTION   : Inserts offspring into population, for steady-state reproduction
'LIBRARY      : OPTIMIZATION
'GROUP        : GENETIC_PIKAIA
'ID           : 013
'AUTHOR       : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*********************************************************************************
'*********************************************************************************
```

```
Private Function PIKAIA_STDREP_FUNC(ByRef NO_PARAM As Integer, _
ByRef NSIZE As Integer, _
ByRef NPOP_INT As Integer, _
ByRef IREP_INT As Integer, _
ByRef IELITE_INT As Integer, _
ByRef PH_MAT As Variant, _
ByRef OLDPH_MAT As Variant, _
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant, _
ByRef JFIT_ARR As Variant, _
ByRef NNEW_INT As Integer)

'steady-state reproduction: insert offspring pair into population
'only if they are fit enough (Replace-random if irep=2 or
'Replace-worst if irep=3).


Dim h As Integer
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim l As Integer

Dim FIT_VAL As Double

On Error GoTo ERROR_LABEL

PIKAIA_STDREP_FUNC = False

NNEW_INT = 0
'compute offspring fitness (with caller's fitness function)
For j = 1 To 2
```

```
'if fit enough, insert in population
    For k = 1 To NSIZE
        XTEMP_ARR(k) = PH_MAT(k, j)
    Next k
    FIT_VAL = PIKAIA_OBJ_FUNC(NSIZE, XTEMP_ARR())
    For i = NPOP_INT To 1 Step -1
        If (FIT_VAL > FITNS_ARR(IFIT_ARR(i))) Then
        'make sure the phenotype is not already in the population

            If (i < NPOP_INT) Then
                For k = 1 To NSIZE
                    If (OLDPH_MAT(k, IFIT_ARR(i + 1)) <> PH_MAT(k, j)) Then GoTo 6
                Next k
                GoTo 1
6:
            End If

'offspring is fit enough for insertion, and is unique
'(i) insert phenotype at appropriate place in population

            If (IREP_INT = 3) Then
                h = 1
            ElseIf (IELITE_INT = 0 Or i = NPOP_INT) Then
                h = Int(NEXT_PSEUDO_RND_FUNC() * NPOP_INT) + 1
            Else
                h = Int(NEXT_PSEUDO_RND_FUNC() * (NPOP_INT - 1)) + 1
            End If

            l = IFIT_ARR(h)
            FITNS_ARR(l) = FIT_VAL
            For k = 1 To NSIZE
                OLDPH_MAT(k, l) = PH_MAT(k, j)
```

```
            Next k
            'shift and update ranking arrays
            If (i < h) Then
               JFIT_ARR(l) = NPOP_INT - i
               For k = h - 1 To i + 1 Step -1
                  JFIT_ARR(IFIT_ARR(k)) = JFIT_ARR(IFIT_ARR(k)) - 1
                  IFIT_ARR(k + 1) = IFIT_ARR(k)
               Next k
               IFIT_ARR(i + 1) = l
            Else
               'shift down
               JFIT_ARR(l) = NPOP_INT - i + 1
               For k = h + 1 To i
                  JFIT_ARR(IFIT_ARR(k)) = JFIT_ARR(IFIT_ARR(k)) + 1
                  IFIT_ARR(k - 1) = IFIT_ARR(k)
               Next k
               IFIT_ARR(i) = l
            End If
            NNEW_INT = NNEW_INT + 1
GoTo 1
         End If
      Next i
1:
Next j

PIKAIA_STDREP_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_STDREP_FUNC = False
End Function
```

```
'*******************************************************************************************

'*******************************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca

'*******************************************************************************************

'*******************************************************************************************

'FUNCTION      : PIKAIA_NEWPOP_FUNC

'DESCRIPTION   : Replaces old population by new; recomputes fitnesses & ranks

'LIBRARY       : OPTIMIZATION

'GROUP         : GENETIC_PIKAIA

'ID            : 014

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'*******************************************************************************************

'*******************************************************************************************


Private Function PIKAIA_NEWPOP_FUNC(ByRef IELITE_INT As Integer, _

ByRef NO_PARAM As Integer, _

ByRef NSIZE As Integer, _

ByRef NPOP_INT As Integer, _

ByRef OLDPH_MAT As Variant, _

ByRef NEWPH_MAT As Variant, _

ByRef IFIT_ARR As Variant, _

ByRef JFIT_ARR As Variant, _

ByRef FITNS_ARR As Variant, _

ByRef NNEW_INT As Integer)


Dim i As Integer

Dim k As Integer


On Error GoTo ERROR_LABEL
```

```
PIKAIA_NEWPOP_FUNC = False

'if using elitism, introduce in new population fittest of old
'population (if greater than fitness of the individual it is
'to Replace)

NNEW_INT = NPOP_INT
For k = 1 To NSIZE
   XTEMP_ARR(k) = NEWPH_MAT(k, 1)
Next k

If (IELITE_INT = 1 And _
   PIKAIA_OBJ_FUNC(NSIZE, XTEMP_ARR()) < FITNS_ARR(IFIT_ARR(NPOP_INT))) Then
   For k = 1 To NSIZE
      NEWPH_MAT(k, 1) = OLDPH_MAT(k, IFIT_ARR(NPOP_INT))
   Next k
   NNEW_INT = NNEW_INT - 1
End If

'Replace population
For i = 1 To NPOP_INT
   For k = 1 To NSIZE
      OLDPH_MAT(k, i) = NEWPH_MAT(k, i)
   Next k
   For k = 1 To NSIZE
      XTEMP_ARR(k) = OLDPH_MAT(k, i)
   Next k
   'get fitness using caller's fitness function
   FITNS_ARR(i) = PIKAIA_OBJ_FUNC(NSIZE, XTEMP_ARR())
Next i
'compute new population fitness rank order
```

```
Call PIKAIA_RANK_FUNC(NPOP_INT, FITNS_ARR, IFIT_ARR, JFIT_ARR)

PIKAIA_NEWPOP_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_NEWPOP_FUNC = False
End Function

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'FUNCTION      : PIKAIA_SORT_FUNC
'DESCRIPTION   : Return integer array p which indexes array a in increasing order.
'Array A is not disturbed.  The Quicksort algorithm is used.
'B.G.Knapp, 86 / 12 / 23
'Reference: N. Wirth, Algorithms and Data Structures,
'Prentice - Hall, 1986

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 015
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Private Function PIKAIA_SORT_FUNC(ByRef NROWS As Integer, _
```

```
ByRef FITNS_ARR As Variant, _
ByRef IFIT_ARR As Variant)

Const Q_VAL As Integer = 11
' Q_VAL = smallest subfile to use quicksort on
Const LGN_VAL As Integer = 32
' LGN_VAL = log base 2 of maximum n;

Dim i As Integer
Dim j As Integer
Dim l As Integer
Dim m As Integer
Dim r As Integer
Dim s As Integer
Dim t As Integer

Dim XDATA_VAL As Double

On Error GoTo ERROR_LABEL

Dim LTEMP_ARR(1 To LGN_VAL) As Integer
Dim RTEMP_ARR(1 To LGN_VAL) As Integer

PIKAIA_SORT_FUNC = False

'Initialize the stack
LTEMP_ARR(1) = 1
RTEMP_ARR(1) = NROWS
s = 1
'Initialize the pointer array
For i = 1 To NROWS
    IFIT_ARR(i) = i
```

```
Next i
2:

If s > 0 Then
   l = LTEMP_ARR(s)
   r = RTEMP_ARR(s)
   s = s - 1
3:
   If (r - l) < Q_VAL Then 'Use straight insertion
      For i = l + 1 To r
         t = IFIT_ARR(i)
         XDATA_VAL = FITNS_ARR(t)
         For j = i - 1 To l Step -1
            If FITNS_ARR(IFIT_ARR(j)) >= XDATA_VAL Then GoTo 5
            IFIT_ARR(j + 1) = IFIT_ARR(j)
         Next j
         j = l - 1
5:          IFIT_ARR(j + 1) = t
      Next i
   Else
      'Use quicksort, with pivot as median of FITNS_ARR(l), FITNS_ARR(m), FITNS_ARR(r)
      m = (l + r) / 2
      t = IFIT_ARR(m)
      If FITNS_ARR(t) > FITNS_ARR(IFIT_ARR(l)) Then
         IFIT_ARR(m) = IFIT_ARR(l)
         IFIT_ARR(l) = t
         t = IFIT_ARR(m)
      End If
      If FITNS_ARR(t) < FITNS_ARR(IFIT_ARR(r)) Then
         IFIT_ARR(m) = IFIT_ARR(r)
         IFIT_ARR(r) = t
         t = IFIT_ARR(m)
```

```
            If FITNS_ARR(t) > FITNS_ARR(IFIT_ARR(l)) Then
               IFIT_ARR(m) = IFIT_ARR(l)
               IFIT_ARR(l) = t
               t = IFIT_ARR(m)
            End If
         End If

         'Partition
         XDATA_VAL = FITNS_ARR(t)
         i = l + 1
         j = r - 1

7:       If i <= j Then
8:          If FITNS_ARR(IFIT_ARR(i)) > XDATA_VAL Then
               i = i + 1
               GoTo 8
            End If
9:          If XDATA_VAL > FITNS_ARR(IFIT_ARR(j)) Then
               j = j - 1
               GoTo 9
            End If

            If i <= j Then
               t = IFIT_ARR(i)
               IFIT_ARR(i) = IFIT_ARR(j)
               IFIT_ARR(j) = t
               i = i + 1
               j = j - 1
            End If
            GoTo 7
         End If
```

```
'Stack the larger subfile
        s = s + 1
        If (j - l) > (r - i) Then
           LTEMP_ARR(s) = l
           RTEMP_ARR(s) = j
           l = i
        Else
           LTEMP_ARR(s) = i
           RTEMP_ARR(s) = r
           r = j
        End If
        GoTo 3
     End If
     GoTo 2
End If

PIKAIA_SORT_FUNC = True

Exit Function
ERROR_LABEL:
PIKAIA_SORT_FUNC = False
End Function

'*******************************************************************************
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION      : NEXT_PSEUDO_RND_FUNC
```

```
'DESCRIPTION   : This routine does not take any arguments.  If the user wishes
'to be able to initialize NEXT_PSEUDO_RND_FUNC, so that the same sequence of
'random numbers can be repeated, this capability could be imple-
'mented with a separate subroutine, and called from the user's
'driver program.  An example NEXT_PSEUDO_RND_FUNC function (and initialization
'subroutine) which uses the function PSEUDO_RND_FUNC (the "minimal standard"
'random number generator of Park and Miller [Comm. ACM 31, 1192-
'1201, Oct 1988; Comm. ACM 36 No. 7, 105-110, July 1993]) is
'provided.

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 016
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************
'*****************************************************************************

Private Function NEXT_PSEUDO_RND_FUNC()

'Return the next pseudo-random deviate from a sequence which is
'uniformly distributed in the interval [0,1]
'Uses the function PSEUDO_RND_FUNC, the "minimal standard" random number
'generator of Park and Miller (Comm. ACM 31, 1192-1201, Oct 1988;
'Comm. ACM 36 No. 7, 105-110, July 1993).

'Common block to make PUB_SEED_VAL visible to INIT_RND_GENER_FUNC (and to save
'it between calls) common /rnseed/ PUB_SEED_VAL

On Error GoTo ERROR_LABEL
If PUB_SEED_VAL <= 0 Then PUB_SEED_VAL = 123456
NEXT_PSEUDO_RND_FUNC = PSEUDO_RND_FUNC(PUB_SEED_VAL)
```

```
Exit Function
ERROR_LABEL:
NEXT_PSEUDO_RND_FUNC = Err.Number
End Function

'*********************************************************************************
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION      : INIT_RND_GENER_FUNC
'DESCRIPTION   : Initialize random number generator
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 017
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*********************************************************************************
'*********************************************************************************

Private Function INIT_RND_GENER_FUNC(ByRef SEED_VAL As Long)

On Error GoTo ERROR_LABEL
PUB_SEED_VAL = SEED_VAL
If PUB_SEED_VAL <= 0 Then PUB_SEED_VAL = 123456
Exit Function
ERROR_LABEL:
End Function

'*********************************************************************************
```

```
'*****************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*****************************************************************************************
'*****************************************************************************************
'FUNCTION      : PSEUDO_RND_FUNC
'DESCRIPTION   :

'"Minimal standard" pseudo-random number generator of Park and
'Miller.  Returns a uniform random deviate r s.t. 0 < r < 1.0.
'Set SEED_VAL to any non-zero integer value to initialize a sequence,
'then do not change SEED_VAL between calls for successive deviates
'in the sequence.

'References:
'Park, S. and Miller, K., "Random Number Generators: Good Ones
' are Hard to Find", Comm. ACM 31, 1192-1201 (Oct. 1988)
'Park, S. and Miller, K., in "Remarks on Choosing and Imple-
' menting Random Number Generators", Comm. ACM 36 No. 7,
' 105-110 (July 1993)

'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 018
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************************
'*****************************************************************************************

Private Function PSEUDO_RND_FUNC(ByRef SEED_VAL As Long)
```

```
Const A_VAL As Long = 48271
Const M_VAL As Long = 2147483647
Const Q_VAL As Long = 44488
Const R_VAL As Long = 3399

Dim j As Long

Dim SCALE_VAL As Double
Dim epsilon As Double
Dim tolerance As Double

On Error GoTo ERROR_LABEL

SCALE_VAL = 1# / M_VAL
epsilon = 0.00000012
tolerance = 1# - epsilon

'Executable section
j = SEED_VAL / Q_VAL
SEED_VAL = A_VAL * (SEED_VAL - j * Q_VAL) - R_VAL * j
If SEED_VAL < 0 Then SEED_VAL = SEED_VAL + M_VAL
PSEUDO_RND_FUNC = MINIMUM_FUNC(SEED_VAL * SCALE_VAL, tolerance)

Exit Function
ERROR_LABEL:
PSEUDO_RND_FUNC = Err.Number
End Function

'************************************************************************************
'************************************************************************************
```

```
'nfermincota.hba2005@ivey.ca
'*****************************************************************************
'*****************************************************************************
'FUNCTION      : PIKAIA_OBJ_FUNC
'DESCRIPTION   : This is the fitness function that is called from sub pikaia
'LIBRARY       : OPTIMIZATION
'GROUP         : GENETIC_PIKAIA
'ID            : 019
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************
'*****************************************************************************

Private Function PIKAIA_OBJ_FUNC(ByVal NROWS As Integer, _
ByRef XDATA_ARR As Variant)

Dim i As Integer
Dim YTEMP_VAL As Double
Dim PARAM_VECTOR() As Variant

ReDim PARAM_VECTOR(1 To NROWS, 1 To 1)

On Error GoTo ERROR_LABEL
For i = 1 To NROWS
   PARAM_ARR(i) = PIKAI_PARAM_SCALE_FUNC(i, XDATA_ARR(i))
   PARAM_VECTOR(i, 1) = PARAM_ARR(i)
Next i
YTEMP_VAL = Excel.Application.Run(PUB_FUNC_STR_NAME, PARAM_VECTOR)
PIKAIA_OBJ_FUNC = YTEMP_VAL

Exit Function
ERROR_LABEL:
```

```
PIKAIA_OBJ_FUNC = Err.Number

End Function

'*********************************************************************************

'*********************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca

'*********************************************************************************

'*********************************************************************************

'FUNCTION      : PIKAI_PARAM_SCALE_FUNC

'DESCRIPTION   : This function scales the parameter values from the 0-1 fraction (frac)

'LIBRARY       : OPTIMIZATION

'GROUP         : GENETIC_PIKAIA

'ID            : 020

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'*********************************************************************************

'*********************************************************************************


Private Function PIKAI_PARAM_SCALE_FUNC(ByRef i As Integer, _

ByVal SCALE_FACTOR As Double)


'paramID = parameter ID number (1-256)

'frac = scaling fraction for the rate (0-1)

'paramScale = scaled paramter value in the original units for the parameter


On Error GoTo ERROR_LABEL


PIKAI_PARAM_SCALE_FUNC = LOWER_ARR(i) + SCALE_FACTOR * (UPPER_ARR(i) - LOWER_ARR(i))


Exit Function
```

```
ERROR_LABEL:
PIKAI_PARAM_SCALE_FUNC = Err.Number
End Function

Sub progress(pctCompl As Single)

UserForm1.Text.Caption = pctCompl & "% Completed"
UserForm1.Bar.Width = pctCompl * 2

DoEvents

End Sub



'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
'SELECTED NOTES ON GENETIC ALGORITHMS
'-------------------------------------------------------------------------------
'-------------------------------------------------------------------------------
'Genetic algorithms (hereafter GAs) are a class of search techniques inspired
'from the biological process of evolution by means of natural selection. They
'can be used to construct numerical optimization techniques that perform robustly
'on problem characterized by ill-behaved search spaces.

'Consider the following generic modeling task: a model that depends on a set of
'adjustable parameters is used to fit a given dataset; the task consists in finding
'the single parameter set that minimizes the difference between the model's
'predictions and the data. A top-level view of a canonical genetic algorithm for
'this task could be as follows: start by generating a set (``population'') of trial
'solutions, usually by choosing random values for all model
'parameters; then:
```

'Evaluate the goodness of fit (``fitness'') of each member of the current population
'(through a chi square measure with the data, for example).

'Select pairs of solutions (``parents'') from the current population, with the
'probability of a given solution being selected made proportional to that solution's
'fitness.

'Breed the two solutions selected in (2) and produce two new solutions (``offspring'').
'Repeat steps (2)-(3) until the number of offspring produced equals the number of
'individuals in the current population.

'Use the new population of offspring to Replace the old population.

'Repeat steps (1) through (5) until some termination criterion is satisfied (e.g.,
'the best solution of the current population reaches a goodness of fit exceeding
'some preset value).

'Superficially, this may look like some peculiar variation on the Monte Carlo theme.
'There are two crucial differences: First, the probability of a given solution being
'selected to participate in a breeding event is made proportional to that solution's
'fitness (step 2); better trial solutions breed more often, the computational
'equivalent of natural selection.

'Second, the production of new trial solutions from existing ones occurs through
'breeding. This involves encoding the parameters defining each solution as a
'string-like structure (``chromosome''), and performing genetically inspired
'operations of crossover and mutation to the pair of chromosomes encoding the two
'parents, the end result of these operations being two new chromosomes defining
'the two offspring. Applying the reverse process of decoding those strings into
'solution parameters completes the breeding process and yields
'two new offspring solutions that incorporate information from both parents.

'Technical Details:

'A genetic-algorithm based approach to a given optimization task, as defined above,
'amounts to a form of forward modeling. Generally speaking, adopting a forward
'modeling approach has both advantages and drawbacks;

'Advantages:

'1) No derivatives of the goodness of fit function with respect to model parameters
'need be computed; it matters little whether the relationship between the model
'and its parameters is linear or nonlinear.

'2) Nothing in the procedure outlined above depends critically on using a least-squares
'statistical estimator; any other robust estimator can be substituted, with little or
'no changes to the overall procedure.

'Drawbacks:

'In most real applications, the model will need to be evaluated (i.e., given a
'parameter set, compute a synthetic dataset and its associated goodness of fit)
'a great many times; if this evaluation is computationally expensive, the forward
'modeling approach can become impractical.

'GA-based optimization retains the advantageous features of forward modeling, while
'reducing the number of required function evaluations to a level that is often
'much more computationally manageable.

'---

'---

Módulo: OPTIM_MULTIVAR_OBJ_LIBR

```
'-------------------------------------------------------------------------------

'-------------------------------------------------------------------------------

Option Explicit     'Requires that all variables to be declared explicitly.

'-------------------------------------------------------------------------------

'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------

'-------------------------------------------------------------------------------

Option Base 1       'The "Option Base" statement allows to specify 0 or 1 as the

                    'default first index of arrays.


'-------------------------------------------------------------------------------

'*******************************************************************************

'*******************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca

'*******************************************************************************

'*******************************************************************************

'FUNCTION      : MULTVAR_CALL_OBJ_FUNC

'DESCRIPTION   : Load Objective Function

'LIBRARY       : OPTIMIZATION

'GROUP         : MULTVAR_OBJ

'ID            : 001

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'*******************************************************************************

'*******************************************************************************
```

```
Function MULTVAR_CALL_OBJ_FUNC(ByVal FUNC_STR_NAME As Variant, _
ByRef PARAM_RNG As Variant, _
Optional ByRef SCALE_RNG As Variant, _
Optional ByVal MIN_FLAG As Boolean = True)

Dim i As Long
Dim j As Long
Dim NROWS As Long
Dim NCOLUMNS As Long

Dim YTEMP_VAL As Double
Dim TEMP_FACTOR As Double

Dim YTEMP_VECTOR As Variant
Dim XTEMP_VECTOR As Variant

Dim PARAM_VECTOR As Variant
Dim SCALE_VECTOR As Variant

On Error GoTo ERROR_LABEL

If MIN_FLAG Then
   TEMP_FACTOR = 1
Else
   TEMP_FACTOR = -1
End If

PARAM_VECTOR = PARAM_RNG
If UBound(PARAM_VECTOR, 1) = 1 Then: PARAM_VECTOR = MATRIX_TRANSPOSE_FUNC(PARAM_VECTOR)
NROWS = UBound(PARAM_VECTOR, 1)  'number of points
NCOLUMNS = UBound(PARAM_VECTOR, 2)  'number of  variables
```

```
If IsArray(SCALE_RNG) = True Then
   SCALE_VECTOR = SCALE_RNG
   If UBound(SCALE_VECTOR, 1) = 1 Then: SCALE_VECTOR = MATRIX_TRANSPOSE_FUNC(SCALE_VECTOR)
Else
   ReDim SCALE_VECTOR(1 To NROWS, 1 To 1)
   For i = 1 To NROWS
      SCALE_VECTOR(i, 1) = 1
   Next i
End If

'------------------------------------------------------------------------
If NCOLUMNS > 1 Then
'------------------------------------------------------------------------
   ReDim YTEMP_VECTOR(1 To NROWS, 1 To 1)
   ReDim XTEMP_VECTOR(1 To NCOLUMNS, 1 To 1)

   For i = 1 To NROWS
      For j = 1 To NCOLUMNS
         XTEMP_VECTOR(j, 1) = SCALE_VECTOR(j, 1) * PARAM_VECTOR(i, j)
      Next j
      YTEMP_VAL = Excel.Application.Run(FUNC_STR_NAME, XTEMP_VECTOR)
      'YTEMP_VAL = YTEMP_VAL ' * TEMP_FACTOR

      YTEMP_VECTOR(i, 1) = YTEMP_VAL
   Next i

   MULTVAR_CALL_OBJ_FUNC = YTEMP_VECTOR

'------------------------------------------------------------------------
Else
'------------------------------------------------------------------------
```

```
    For i = 1 To NROWS
        PARAM_VECTOR(i, 1) = SCALE_VECTOR(i, 1) * PARAM_VECTOR(i, 1)
    Next i

    YTEMP_VAL = Excel.Application.Run(FUNC_STR_NAME, PARAM_VECTOR)
    'YTEMP_VAL = YTEMP_VAL ' * TEMP_FACTOR

    MULTVAR_CALL_OBJ_FUNC = YTEMP_VAL

'-------------------------------------------------------------------------
End If
'-------------------------------------------------------------------------

Exit Function
ERROR_LABEL:
MULTVAR_CALL_OBJ_FUNC = Err.Number
End Function
```

Módulo: EXPORT_CATIA_EXTRUDED

```
Option Explicit     'Requires that all variables to be declared explicitly.
Option Base 1       'The "Option Base" statement allows to specify 0 or 1 as the
                    'default first index of arrays.

Private PUB_FUNC_STR_NAME As String

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'FUNCTION      : CALL_TEST_MULTVAR_FRAME_FUNC
'DESCRIPTION   :
'LIBRARY       : OPTIMIZATION
'GROUP         : MULTVAR_TEST
'ID            : 001
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function CALL_TEST_MULTVAR_FRAME_FUNC(ByVal FUNC_STR_NAME As String, _
ByRef PARAM_RNG As Variant, _
ByRef CONST_RNG As Variant, _
Optional ByVal GRAD_STR_NAME As String = "", _
Optional ByVal CONST_STR_NAME As String = "", _
Optional ByVal MIN_FLAG As Boolean = True, _
```

```
Optional ByVal nLOOPS As Long = 600, _
Optional ByVal epsilon As Double = 0.000000000000001)

Dim i As Long
Dim NSIZE As Long

Dim CONST_BOX As Variant
Dim TEMP_MATRIX As Variant
Dim XTEMP_VECTOR As Variant
Dim PARAM_VECTOR As Variant

On Error GoTo ERROR_LABEL



PUB_FUNC_STR_NAME = FUNC_STR_NAME
CONST_BOX = CONST_RNG
PARAM_VECTOR = PARAM_RNG
If UBound(PARAM_VECTOR, 1) = 1 Then: PARAM_VECTOR = MATRIX_TRANSPOSE_FUNC(PARAM_VECTOR)
NSIZE = UBound(PARAM_VECTOR, 1)
'------------------------------------------------------------------------------
ReDim TEMP_MATRIX(0 To 8, 0 To NSIZE + 1)
'------------------------------------------------------------------------------
TEMP_MATRIX(0, 0) = "--"
TEMP_MATRIX(0, NSIZE + 1) = "Y_VAR"
For i = 1 To NSIZE
    TEMP_MATRIX(0, i) = "X_VAR_" & i
Next i

1983:

'------------------------------------------------------------------------------
'------------------------------------------------------------------------------
```

```
TEMP_MATRIX(1, 0) = "PIKAIA"
XTEMP_VECTOR = PIKAIA_OPTIMIZATION_FUNC(FUNC_STR_NAME, _
                        CONST_BOX, False)
For i = 1 To NSIZE
   XTEMP_VECTOR(i, 1) = XTEMP_VECTOR(i, 1) '* IIf(MIN_FLAG = True, -1, 1)
   TEMP_MATRIX(1, i) = XTEMP_VECTOR(i, 1)
Next i
'--------------------------------------------------------------------------------
TEMP_MATRIX(1, NSIZE + 1) = MULTVAR_CALL_OBJ_FUNC(FUNC_STR_NAME, _
                        XTEMP_VECTOR, "", MIN_FLAG)
'--------------------------------------------------------------------------------



CALL_TEST_MULTVAR_FRAME_FUNC = TEMP_MATRIX


Exit Function
ERROR_LABEL:
CALL_TEST_MULTVAR_FRAME_FUNC = Err.Number
End Function


'*******************************************************************************
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION     : CALL_MULTVAR_JACOB_FUNC
'DESCRIPTION  :
'LIBRARY      : OPTIMIZATION
'GROUP        : MULTVAR_TEST
'ID           : 002
```

```
'AUTHOR       : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'***************************************************************************************

'***************************************************************************************


Function CALL_MULTVAR_JACOB_FUNC(ByRef PARAM_RNG As Variant)


Dim PARAM_VECTOR As Variant


On Error GoTo ERROR_LABEL


PUB_FUNC_STR_NAME = "CALL_MULTVAR_OBJ_1_FUNC"


PARAM_VECTOR = PARAM_RNG

If UBound(PARAM_VECTOR, 1) = 1 Then: PARAM_VECTOR = MATRIX_TRANSPOSE_FUNC(PARAM_VECTOR)


'CALL_MULTVAR_JACOB_FUNC                                                                 =
MATRIX_TRANSPOSE_FUNC(JACOB_CENTRAL_FUNC(PUB_FUNC_STR_NAME, _

                PARAM_VECTOR, 0.00001))


CALL_MULTVAR_JACOB_FUNC                                                                 =
MATRIX_TRANSPOSE_FUNC(JACOB_FORWARD_FUNC(PUB_FUNC_STR_NAME, _

                PARAM_VECTOR, 0.00001))

Exit Function

ERROR_LABEL:

CALL_MULTVAR_JACOB_FUNC = Err.Number

End Function


'***************************************************************************************

'***************************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca
```

```
'*******************************************************************************

'*******************************************************************************

'FUNCTION      : CALL_MULTVAR_OBJ_1_FUNC

'DESCRIPTION   : Peak and Pit minimum function

'LIBRARY       : OPTIMIZATION

'GROUP         : MULTVAR_TEST

'ID            : 003

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'*******************************************************************************

'*******************************************************************************

Function CALL_MULTVAR_OBJ_1_FUNC(ByRef PARAM_RNG As Variant)

Dim X_VAL As Variant

Dim Y_VAL As Variant

Dim PARAM_VECTOR As Variant

On Error GoTo ERROR_LABEL

PARAM_VECTOR = PARAM_RNG

X_VAL = PARAM_VECTOR(1, 1)

Y_VAL = PARAM_VECTOR(2, 1)

CALL_MULTVAR_OBJ_1_FUNC = (X_VAL * Y_VAL)

Exit Function

ERROR_LABEL:

CALL_MULTVAR_OBJ_1_FUNC = Err.Number

End Function

'*******************************************************************************
```

```
'*******************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*******************************************************************************
'*******************************************************************************
'FUNCTION      : CALL_MULTVAR_CONST_1_FUNC
'DESCRIPTION   :
'LIBRARY       : OPTIMIZATION
'GROUP         : MULTVAR_TEST
'ID            : 004
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*******************************************************************************
'*******************************************************************************

Function CALL_MULTVAR_CONST_1_FUNC(ByRef PARAM_RNG As Variant)

Dim X_VAL As Variant
Dim Y_VAL As Variant
Dim TEMP_FLAG As Boolean
Dim PARAM_VECTOR As Variant

On Error GoTo ERROR_LABEL

TEMP_FLAG = True
PARAM_VECTOR = PARAM_RNG
X_VAL = PARAM_VECTOR(1, 1)
Y_VAL = PARAM_VECTOR(2, 1)

If X_VAL < -2.5 Then: TEMP_FLAG = False
If X_VAL > 2.5 Then: TEMP_FLAG = False
```

```
If Y_VAL < -2.5 Then: TEMP_FLAG = False

If Y_VAL > 2.5 Then: TEMP_FLAG = False


'In the plot we clearly observe the presence of a maximum and a minimum in

'the domain -2.5 < x < 2.5 , -2.5 < y < 2.5


'CONST_BOX(1, 1) = -2.5 'X-Min

'CONST_BOX(1, 2) = -2.5 'Y-Min


'CONST_BOX(2, 1) = 2.5 'X-Max

'CONST_BOX(2, 2) = 2.5 'Y-Max


'The maximum is located in the area { x, y | x>0 , y>0 } and the minimum is located in

'the area { x, y | x<0 , y<0 }. The point (0, 0) is at the middle of the maximum and

'minimum points so we can use it as starting point for both searches.


CALL_MULTVAR_CONST_1_FUNC = TEMP_FLAG


Exit Function

ERROR_LABEL:

CALL_MULTVAR_CONST_1_FUNC = TEMP_FLAG

End Function



'LIBRARY      : OPTIMIZATION

'GROUP        : MULTVAR_TEST

'ID           : 006

'AUTHOR       : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 02/13/2009

'*****************************************************************************************

'*****************************************************************************************
```

```
Function CALL_MULTVAR_GRAD_2_FUNC(ByRef PARAM_RNG As Variant)

Dim X_VAL As Variant
Dim Y_VAL As Variant
Dim TEMP_ARR As Variant
Dim PARAM_VECTOR As Variant

On Error GoTo ERROR_LABEL

PARAM_VECTOR = PARAM_RNG

X_VAL = PARAM_VECTOR(1, 1)
Y_VAL = PARAM_VECTOR(2, 1)

ReDim TEMP_ARR(1 To 2, 1 To 1)

TEMP_ARR(1, 1) = 4 * X_VAL + 4 * Y_VAL - 12
TEMP_ARR(2, 1) = 4 * X_VAL + 16 * Y_VAL - 36

CALL_MULTVAR_GRAD_2_FUNC = TEMP_ARR

Exit Function
ERROR_LABEL:
CALL_MULTVAR_GRAD_2_FUNC = Err.Number
End Function



'*****************************************************************************************
'*****************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
```

```
'nfermincota.hba2005@ivey.ca
'*****************************************************************************
'*****************************************************************************
'FUNCTION      : CALL_MULTVAR_OBJ_3_FUNC

'DESCRIPTION   : Super parabolic surface minimum function
'This example shows another case in which the optimization algorithms that do not
'require external derivative equations are sometimes superior to those that
'require external derivative equations.

'LIBRARY       : OPTIMIZATION
'GROUP         : MULTVAR_TEST
'ID            : 008
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 02/13/2009
'*****************************************************************************
'*****************************************************************************

Function CALL_MULTVAR_OBJ_3_FUNC(ByRef PARAM_RNG As Variant)

'reading the variables from the worksheet to be
    Dim i, j, k, l, m, n As Integer
    Dim PARAM_VECTOR As Variant
    Dim Base(29), Height(29), thickness(29) As Double

    On Error GoTo ERROR_LABEL

    PARAM_VECTOR = PARAM_RNG

    For i = 1 To 29

       Height(i) = PARAM_VECTOR((3 * i) - 2, 1)
```

```
        Base(i) = PARAM_VECTOR((3 * i) - 1, 1)
        thickness(i) = PARAM_VECTOR((3 * i), 1)

    Next i

'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
'Stiffness matrix of each element
Dim E_Steel, G_Steel As Double
E_Steel = 210000000000#       'Pa for steel
G_Steel = 81000000000#        'Pa for steel


    Dim Area(29), Iy(29), Iz(29), I_zero(29), Mass(29) As Double
    Dim Area1, Area2, Izz1, Izz2, dcuadz1, dcuadz2, Areadcuadz1, Areadcuadz2, Iyy1, Iyy2, dcuady1, _
    dcuady2, Areadcuady1, Areadcuady2 As Double
    Dim AE, PE As Double

       i = 1
    'Define area
       For i = 1 To 29

          Area1 = Height(i) * thickness(i)
          Area2 = (Base(i) - 2 * thickness(i)) * thickness(i)
          Izz1 = (thickness(i) * Height(i) ^ 3) / 12
          Izz2 = ((Base(i) - 2 * thickness(i)) * thickness(i) ^ 3) / 12
          dcuadz1 = 0
          dcuadz2 = (Height(i) / 2 - thickness(i) / 2) ^ 2
          Areadcuadz1 = Area1 * dcuadz1
          Areadcuadz2 = Area2 * dcuadz2

          Iz(i) = 2 * (Izz1 + Izz2) + 2 * (Areadcuadz1 + Areadcuadz2)
```

```
        Iyy1 = (Height(i) * thickness(i) ^ 3) / 12
        Iyy2 = (thickness(i) * (Base(i) - 2 * thickness(i)) ^ 3) / 12
        dcuady1 = (Base(i) / 2 - thickness(i) / 2) ^ 2
        dcuady2 = 0
        Areadcuady1 = Area1 * dcuady1
        Areadcuady2 = Area2 * dcuady2

        Iy(i) = 2 * (Iyy1 + Iyy2) + 2 * (Areadcuady1 + Areadcuady2)


        Area(i) = 2 * Area1 + 2 * Area2

        Mass(i) = Area(i) * length(i - 1) * Steel_Density



        AE = (Height(i) - thickness(i)) * (Base(i) - thickness(i))
        PE = 2 * (Height(i) - thickness(i)) + 2 * (Base(i) - thickness(i))

        I_zero(i) = (4 * (AE ^ 2 * thickness(i))) / PE

    Next i

'//////////////CREATING MATRIX K OF EACH ELEMENT INTO ONE ARRAY

    Dim Kelementsmatrixlocal(12, 348) As Double


    '//// initialize with 0 all the matrix

    Erase Kelementsmatrixlocal
```

```
Dim elem1, elem2, elem3, elem4, elem5, elem6, elem7, elem8, elem9, elem10 As Double

For i = 1 To 29

   elem1 = (E_Steel * Area(i)) / length(i - 1)
   elem2 = (12 * E_Steel * Iz(i)) / (length(i - 1)) ^ 3
   elem3 = (12 * E_Steel * Iy(i)) / (length(i - 1)) ^ 3
   elem4 = (G_Steel * I_zero(i)) / length(i - 1)
   elem5 = (4 * E_Steel * Iy(i)) / (length(i - 1))
   elem6 = (4 * E_Steel * Iz(i)) / (length(i - 1))
   elem7 = (6 * E_Steel * Iz(i)) / (length(i - 1)) ^ 2
   elem8 = (6 * E_Steel * Iy(i)) / (length(i - 1)) ^ 2
   elem9 = (2 * E_Steel * Iy(i)) / (length(i - 1))
   elem10 = (2 * E_Steel * Iz(i)) / (length(i - 1))

   Kelementsmatrixlocal(1, 12 * i - 11) = elem1
   Kelementsmatrixlocal(1, 12 * i - 5) = -elem1
   Kelementsmatrixlocal(7, 12 * i - 11) = -elem1
   Kelementsmatrixlocal(7, 12 * i - 5) = elem1

   Kelementsmatrixlocal(2, 12 * i - 10) = elem2
   Kelementsmatrixlocal(2, 12 * i - 4) = -elem2
   Kelementsmatrixlocal(8, 12 * i - 10) = -elem2
   Kelementsmatrixlocal(8, 12 * i - 4) = elem2

   Kelementsmatrixlocal(3, 12 * i - 9) = elem3
   Kelementsmatrixlocal(3, 12 * i - 3) = -elem3
   Kelementsmatrixlocal(9, 12 * i - 9) = -elem3
   Kelementsmatrixlocal(9, 12 * i - 3) = elem3

   Kelementsmatrixlocal(4, 12 * i - 8) = elem4
   Kelementsmatrixlocal(4, 12 * i - 2) = -elem4
```

```
Kelementsmatrixlocal(10, 12 * i - 8) = -elem4
Kelementsmatrixlocal(10, 12 * i - 2) = elem4

Kelementsmatrixlocal(5, 12 * i - 7) = elem5
Kelementsmatrixlocal(11, 12 * i - 1) = elem5

Kelementsmatrixlocal(6, 12 * i - 6) = elem6
Kelementsmatrixlocal(12, 12 * i) = elem6

Kelementsmatrixlocal(6, 12 * i - 10) = elem7
Kelementsmatrixlocal(6, 12 * i - 4) = -elem7
Kelementsmatrixlocal(12, 12 * i - 10) = elem7
Kelementsmatrixlocal(12, 12 * i - 4) = -elem7
Kelementsmatrixlocal(2, 12 * i - 6) = elem7
Kelementsmatrixlocal(2, 12 * i) = elem7
Kelementsmatrixlocal(8, 12 * i - 6) = -elem7
Kelementsmatrixlocal(8, 12 * i) = -elem7

Kelementsmatrixlocal(5, 12 * i - 9) = -elem8
Kelementsmatrixlocal(5, 12 * i - 3) = elem8
Kelementsmatrixlocal(11, 12 * i - 9) = -elem8
Kelementsmatrixlocal(11, 12 * i - 3) = elem8
Kelementsmatrixlocal(3, 12 * i - 7) = -elem8
Kelementsmatrixlocal(3, 12 * i - 1) = -elem8
Kelementsmatrixlocal(9, 12 * i - 7) = elem8
Kelementsmatrixlocal(9, 12 * i - 1) = elem8

Kelementsmatrixlocal(11, 12 * i - 7) = elem9
Kelementsmatrixlocal(5, 12 * i - 1) = elem9

Kelementsmatrixlocal(12, 12 * i - 6) = elem10
Kelementsmatrixlocal(6, 12 * i) = elem10
```

```
Next i

Dim KLocal1(12, 12), KLocal2(12, 12), KLocal3(12, 12), KLocal4(12, 12), KLocal5(12, 12), KLocal6(12, 12), KLocal7(12, 12) _

, KLocal8(12, 12), KLocal9(12, 12), KLocal10(12, 12), KLocal11(12, 12), KLocal12(12, 12), KLocal13(12, 12), KLocal14(12, 12) _

, KLocal15(12, 12), KLocal16(12, 12), KLocal17(12, 12), KLocal18(12, 12), KLocal19(12, 12), KLocal20(12, 12), KLocal21(12, 12) _

, KLocal22(12, 12), KLocal23(12, 12), KLocal24(12, 12), KLocal25(12, 12), KLocal26(12, 12), KLocal27(12, 12), KLocal28(12, 12) _

, KLocal29(12, 12) As Variant

Dim p As Integer

For k = 1 To 12

    For p = 1 To 12

        KLocal1(k, p) = Kelementsmatrixlocal(k, p)

        KLocal2(k, p) = Kelementsmatrixlocal(k, p + 12)

        KLocal3(k, p) = Kelementsmatrixlocal(k, p + 24)

        KLocal4(k, p) = Kelementsmatrixlocal(k, p + 36)

        KLocal5(k, p) = Kelementsmatrixlocal(k, p + 48)

        KLocal6(k, p) = Kelementsmatrixlocal(k, p + 60)

        KLocal7(k, p) = Kelementsmatrixlocal(k, p + 72)

        KLocal8(k, p) = Kelementsmatrixlocal(k, p + 84)

        KLocal9(k, p) = Kelementsmatrixlocal(k, p + 96)

        KLocal10(k, p) = Kelementsmatrixlocal(k, p + 108)

        KLocal11(k, p) = Kelementsmatrixlocal(k, p + 120)

        KLocal12(k, p) = Kelementsmatrixlocal(k, p + 132)

        KLocal13(k, p) = Kelementsmatrixlocal(k, p + 144)

        KLocal14(k, p) = Kelementsmatrixlocal(k, p + 156)

        KLocal15(k, p) = Kelementsmatrixlocal(k, p + 168)

        KLocal16(k, p) = Kelementsmatrixlocal(k, p + 180)

        KLocal17(k, p) = Kelementsmatrixlocal(k, p + 192)
```

```
            KLocal18(k, p) = Kelementsmatrixlocal(k, p + 204)

            KLocal19(k, p) = Kelementsmatrixlocal(k, p + 216)

            KLocal20(k, p) = Kelementsmatrixlocal(k, p + 228)

            KLocal21(k, p) = Kelementsmatrixlocal(k, p + 240)

            KLocal22(k, p) = Kelementsmatrixlocal(k, p + 252)

            KLocal23(k, p) = Kelementsmatrixlocal(k, p + 264)

            KLocal24(k, p) = Kelementsmatrixlocal(k, p + 276)

            KLocal25(k, p) = Kelementsmatrixlocal(k, p + 288)

            KLocal26(k, p) = Kelementsmatrixlocal(k, p + 300)

            KLocal27(k, p) = Kelementsmatrixlocal(k, p + 312)

            KLocal28(k, p) = Kelementsmatrixlocal(k, p + 324)

            KLocal29(k, p) = Kelementsmatrixlocal(k, p + 336)

        Next p

    Next k


'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\


'Transform each matrix K into global coordinates


    Dim Kelemglob1, Kelemglob2, Kelemglob3, Kelemglob4, Kelemglob5, Kelemglob6, Kelemglob7, Kelemglob8, Kelemglob9 _

    , Kelemglob10, Kelemglob11, Kelemglob12, Kelemglob13, Kelemglob14, Kelemglob15, Kelemglob16, Kelemglob17, Kelemglob18 _

    , Kelemglob19, Kelemglob20, Kelemglob21, Kelemglob22, Kelemglob23, Kelemglob24, Kelemglob25, Kelemglob26, Kelemglob27, _

    Kelemglob28, Kelemglob29


        Kelemglob1 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix1, KLocal1), WorksheetFunction.MInverse(Tmatrix1))

        Kelemglob2 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix2, KLocal2), WorksheetFunction.MInverse(Tmatrix2))

        Kelemglob3 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix3, KLocal3), WorksheetFunction.MInverse(Tmatrix3))

        Kelemglob4 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix4, KLocal4), WorksheetFunction.MInverse(Tmatrix4))
```

```
    Kelemglob5 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix5, KLocal5),
WorksheetFunction.MInverse(Tmatrix5))

    Kelemglob6 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix6, KLocal6),
WorksheetFunction.MInverse(Tmatrix6))

    Kelemglob7 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix7, KLocal7),
WorksheetFunction.MInverse(Tmatrix7))

    Kelemglob8 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix8, KLocal8),
WorksheetFunction.MInverse(Tmatrix8))

    Kelemglob9 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix9, KLocal9),
WorksheetFunction.MInverse(Tmatrix9))

    Kelemglob10 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix10, KLocal10),
WorksheetFunction.MInverse(Tmatrix10))

    Kelemglob11 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix11, KLocal11),
WorksheetFunction.MInverse(Tmatrix11))

    Kelemglob12 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix12, KLocal12),
WorksheetFunction.MInverse(Tmatrix12))

    Kelemglob13 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix13, KLocal13),
WorksheetFunction.MInverse(Tmatrix13))

    Kelemglob14 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix14, KLocal14),
WorksheetFunction.MInverse(Tmatrix14))

    Kelemglob15 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix15, KLocal15),
WorksheetFunction.MInverse(Tmatrix15))

    Kelemglob16 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix16, KLocal16),
WorksheetFunction.MInverse(Tmatrix16))

    Kelemglob17 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix17, KLocal17),
WorksheetFunction.MInverse(Tmatrix17))

    Kelemglob18 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix18, KLocal18),
WorksheetFunction.MInverse(Tmatrix18))

    Kelemglob19 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix19, KLocal19),
WorksheetFunction.MInverse(Tmatrix19))

    Kelemglob20 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix20, KLocal20),
WorksheetFunction.MInverse(Tmatrix20))

    Kelemglob21 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix21, KLocal21),
WorksheetFunction.MInverse(Tmatrix21))

    Kelemglob22 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix22, KLocal22),
WorksheetFunction.MInverse(Tmatrix22))

    Kelemglob23 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix23, KLocal23),
WorksheetFunction.MInverse(Tmatrix23))

    Kelemglob24 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix24, KLocal24),
WorksheetFunction.MInverse(Tmatrix24))

    Kelemglob25 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix25, KLocal25),
WorksheetFunction.MInverse(Tmatrix25))
```

```
    Kelemglob26 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix26, KLocal26), WorksheetFunction.MInverse(Tmatrix26))

    Kelemglob27 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix27, KLocal27), WorksheetFunction.MInverse(Tmatrix27))

    Kelemglob28 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix28, KLocal28), WorksheetFunction.MInverse(Tmatrix28))

    Kelemglob29 = WorksheetFunction.MMult(WorksheetFunction.MMult(Tmatrix29, KLocal29), WorksheetFunction.MInverse(Tmatrix29))


'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\


' to separate the matrix into 4 sub matrix for the K matrix formation


  Dim A11(6, 6), A12(6, 6), A21(6, 6), A22(6, 6), B11(6, 6), B12(6, 6), B21(6, 6), B22(6, 6), C11(6, 6), C12(6, 6), C21(6, 6), C22(6, 6) _

  , DR11(6, 6), DR12(6, 6), DR21(6, 6), DR22(6, 6), DL11(6, 6), DL12(6, 6), DL21(6, 6), DL22(6, 6), E11(6, 6), E12(6, 6), E21(6, 6), E22(6, 6) _

  , F11(6, 6), F12(6, 6), F21(6, 6), F22(6, 6), G11(6, 6), G12(6, 6), G21(6, 6), G22(6, 6), H11(6, 6), H12(6, 6), H21(6, 6), H22(6, 6) _

  , I11(6, 6), I12(6, 6), I21(6, 6), I22(6, 6), J11(6, 6), J12(6, 6), J21(6, 6), J22(6, 6), K11(6, 6), K12(6, 6), K21(6, 6), K22(6, 6) _

  , L11(6, 6), L12(6, 6), L21(6, 6), L22(6, 6), M11(6, 6), M12(6, 6), M21(6, 6), M22(6, 6), N11(6, 6), N12(6, 6), N21(6, 6), N22(6, 6) _

  , OA11(6, 6), OA12(6, 6), OA21(6, 6), OA22(6, 6), PA11(6, 6), PA12(6, 6), PA21(6, 6), PA22(6, 6), QA11(6, 6), QA12(6, 6), QA21(6, 6), QA22(6, 6) _

  , RA11(6, 6), RA12(6, 6), RA21(6, 6), RA22(6, 6), OB11(6, 6), OB12(6, 6), OB21(6, 6), OB22(6, 6), PB11(6, 6), PB12(6, 6), PB21(6, 6), PB22(6, 6) _

  , QB11(6, 6), QB12(6, 6), QB21(6, 6), QB22(6, 6), RB11(6, 6), RB12(6, 6), RB21(6, 6), RB22(6, 6), S11(6, 6), S12(6, 6), S21(6, 6), S22(6, 6) _

  , T11(6, 6), T12(6, 6), T21(6, 6), T22(6, 6), U11(6, 6), U12(6, 6), U21(6, 6), U22(6, 6), VD11(6, 6), VD12(6, 6), VD21(6, 6), VD22(6, 6), VE11(6, 6) _

  , VE12(6, 6), VE21(6, 6), VE22(6, 6), VF11(6, 6), VF12(6, 6), VF21(6, 6), VF22(6, 6) As Variant


  Dim r As Integer


    For r = 1 To 6
```

```
For n = 1 To 6

    A11(r, n) = Kelemglob1(r, n)
    A12(r, n) = Kelemglob1(r, n + 6)
    A21(r, n) = Kelemglob1(r + 6, n)
    A22(r, n) = Kelemglob1(r + 6, n + 6)
    B11(r, n) = Kelemglob2(r, n)
    B12(r, n) = Kelemglob2(r, n + 6)
    B21(r, n) = Kelemglob2(r + 6, n)
    B22(r, n) = Kelemglob2(r + 6, n + 6)
    C11(r, n) = Kelemglob3(r, n)
    C12(r, n) = Kelemglob3(r, n + 6)
    C21(r, n) = Kelemglob3(r + 6, n)
    C22(r, n) = Kelemglob3(r + 6, n + 6)
    DR11(r, n) = Kelemglob4(r, n)
    DR12(r, n) = Kelemglob4(r, n + 6)
    DR21(r, n) = Kelemglob4(r + 6, n)
    DR22(r, n) = Kelemglob4(r + 6, n + 6)
    DL11(r, n) = Kelemglob5(r, n)
    DL12(r, n) = Kelemglob5(r, n + 6)
    DL21(r, n) = Kelemglob5(r + 6, n)
    DL22(r, n) = Kelemglob5(r + 6, n + 6)
    E11(r, n) = Kelemglob6(r, n)
    E12(r, n) = Kelemglob6(r, n + 6)
    E21(r, n) = Kelemglob6(r + 6, n)
    E22(r, n) = Kelemglob6(r + 6, n + 6)
    F11(r, n) = Kelemglob7(r, n)
    F12(r, n) = Kelemglob7(r, n + 6)
    F21(r, n) = Kelemglob7(r + 6, n)
    F22(r, n) = Kelemglob7(r + 6, n + 6)
    G11(r, n) = Kelemglob8(r, n)
    G12(r, n) = Kelemglob8(r, n + 6)
```

G21(r, n) = Kelemglob8(r + 6, n)

G22(r, n) = Kelemglob8(r + 6, n + 6)

H11(r, n) = Kelemglob9(r, n)

H12(r, n) = Kelemglob9(r, n + 6)

H21(r, n) = Kelemglob9(r + 6, n)

H22(r, n) = Kelemglob9(r + 6, n + 6)

I11(r, n) = Kelemglob10(r, n)

I12(r, n) = Kelemglob10(r, n + 6)

I21(r, n) = Kelemglob10(r + 6, n)

I22(r, n) = Kelemglob10(r + 6, n + 6)

J11(r, n) = Kelemglob11(r, n)

J12(r, n) = Kelemglob11(r, n + 6)

J21(r, n) = Kelemglob11(r + 6, n)

J22(r, n) = Kelemglob11(r + 6, n + 6)

K11(r, n) = Kelemglob12(r, n)

K12(r, n) = Kelemglob12(r, n + 6)

K21(r, n) = Kelemglob12(r + 6, n)

K22(r, n) = Kelemglob12(r + 6, n + 6)

L11(r, n) = Kelemglob13(r, n)

L12(r, n) = Kelemglob13(r, n + 6)

L21(r, n) = Kelemglob13(r + 6, n)

L22(r, n) = Kelemglob13(r + 6, n + 6)

M11(r, n) = Kelemglob14(r, n)

M12(r, n) = Kelemglob14(r, n + 6)

M21(r, n) = Kelemglob14(r + 6, n)

M22(r, n) = Kelemglob14(r + 6, n + 6)

N11(r, n) = Kelemglob15(r, n)

N12(r, n) = Kelemglob15(r, n + 6)

N21(r, n) = Kelemglob15(r + 6, n)

N22(r, n) = Kelemglob15(r + 6, n + 6)

OA11(r, n) = Kelemglob16(r, n)

OA12(r, n) = Kelemglob16(r, n + 6)

OA21(r, n) = Kelemglob16(r + 6, n)

OA22(r, n) = Kelemglob16(r + 6, n + 6)

PA11(r, n) = Kelemglob17(r, n)

PA12(r, n) = Kelemglob17(r, n + 6)

PA21(r, n) = Kelemglob17(r + 6, n)

PA22(r, n) = Kelemglob17(r + 6, n + 6)

QA11(r, n) = Kelemglob18(r, n)

QA12(r, n) = Kelemglob18(r, n + 6)

QA21(r, n) = Kelemglob18(r + 6, n)

QA22(r, n) = Kelemglob18(r + 6, n + 6)

RA11(r, n) = Kelemglob19(r, n)

RA12(r, n) = Kelemglob19(r, n + 6)

RA21(r, n) = Kelemglob19(r + 6, n)

RA22(r, n) = Kelemglob19(r + 6, n + 6)

OB11(r, n) = Kelemglob20(r, n)

OB12(r, n) = Kelemglob20(r, n + 6)

OB21(r, n) = Kelemglob20(r + 6, n)

OB22(r, n) = Kelemglob20(r + 6, n + 6)

PB11(r, n) = Kelemglob21(r, n)

PB12(r, n) = Kelemglob21(r, n + 6)

PB21(r, n) = Kelemglob21(r + 6, n)

PB22(r, n) = Kelemglob21(r + 6, n + 6)

QB11(r, n) = Kelemglob22(r, n)

QB12(r, n) = Kelemglob22(r, n + 6)

QB21(r, n) = Kelemglob22(r + 6, n)

QB22(r, n) = Kelemglob22(r + 6, n + 6)

RB11(r, n) = Kelemglob23(r, n)

RB12(r, n) = Kelemglob23(r, n + 6)

RB21(r, n) = Kelemglob23(r + 6, n)

RB22(r, n) = Kelemglob23(r + 6, n + 6)

S11(r, n) = Kelemglob24(r, n)

S12(r, n) = Kelemglob24(r, n + 6)

```
            S21(r, n) = Kelemglob24(r + 6, n)
            S22(r, n) = Kelemglob24(r + 6, n + 6)
            T11(r, n) = Kelemglob25(r, n)
            T12(r, n) = Kelemglob25(r, n + 6)
            T21(r, n) = Kelemglob25(r + 6, n)
            T22(r, n) = Kelemglob25(r + 6, n + 6)
            U11(r, n) = Kelemglob26(r, n)
            U12(r, n) = Kelemglob26(r, n + 6)
            U21(r, n) = Kelemglob26(r + 6, n)
            U22(r, n) = Kelemglob26(r + 6, n + 6)
            VD11(r, n) = Kelemglob27(r, n)
            VD12(r, n) = Kelemglob27(r, n + 6)
            VD21(r, n) = Kelemglob27(r + 6, n)
            VD22(r, n) = Kelemglob27(r + 6, n + 6)
            VE11(r, n) = Kelemglob28(r, n)
            VE12(r, n) = Kelemglob28(r, n + 6)
            VE21(r, n) = Kelemglob28(r + 6, n)
            VE22(r, n) = Kelemglob28(r + 6, n + 6)
            VF11(r, n) = Kelemglob29(r, n)
            VF12(r, n) = Kelemglob29(r, n + 6)
            VF21(r, n) = Kelemglob29(r + 6, n)
            VF22(r, n) = Kelemglob29(r + 6, n + 6)

        Next n
    Next r

'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

'Stiffness Matrix of the structure

    Dim Ksum1(6, 6), Ksum2(6, 6), Ksum3(6, 6), Ksum4(6, 6), Ksum5(6, 6), Ksum6(6, 6), Ksum7(6, 6), Ksum8(6, 6), Ksum9(6, 6), _
```

```
    Ksum10(6, 6), Ksum11(6, 6), Ksum12(6, 6), Ksum13(6, 6), Ksum14(6, 6), Ksum15(6, 6), Ksum16(6, 6), Ksum17(6, 6), Ksum18(6, 6), _

    Ksum19(6, 6), Ksum20(6, 6), Ksum21(6, 6), Ksum22(6, 6), Ksum23(6, 6), Ksum24(6, 6), Ksum25(6, 6), Ksum26(6, 6), Ksum27(6, 6), _

    Ksum28(6, 6), Ksum29(6, 6), Ksum30(6, 6), Ksum31(6, 6), Ksum32(6, 6), Ksum33(6, 6), Ksum34(6, 6) As Variant


    Dim q As Integer


    For q = 1 To 6

      For r = 1 To 6


        Ksum1(q, r) = A22(q, r) + B11(q, r) + OA11(q, r)

        Ksum2(q, r) = B22(q, r) + C11(q, r)

        Ksum3(q, r) = C22(q, r) + DR11(q, r) + OB11(q, r)

        Ksum4(q, r) = DR22(q, r) + E11(q, r)

        Ksum5(q, r) = E22(q, r) + F11(q, r) + S11(q, r)

        Ksum6(q, r) = F22(q, r) + G11(q, r)

        Ksum7(q, r) = G22(q, r) + H11(q, r) + VD11(q, r)

        Ksum8(q, r) = H22(q, r) + I11(q, r)

        Ksum9(q, r) = I22(q, r) + J11(q, r) + VE11(q, r)

        Ksum10(q, r) = J22(q, r) + K11(q, r)

        Ksum11(q, r) = K22(q, r) + L11(q, r)

        Ksum12(q, r) = L22(q, r) + M11(q, r)

        Ksum13(q, r) = M22(q, r) + N11(q, r) + VF11(q, r)

        Ksum14(q, r) = A22(q, r) + B11(q, r) + RA22(q, r)

        Ksum15(q, r) = B22(q, r) + C11(q, r)

        Ksum16(q, r) = C22(q, r) + DL11(q, r) + RB22(q, r)

        Ksum17(q, r) = DL22(q, r) + E11(q, r)

        Ksum18(q, r) = E22(q, r) + F11(q, r) + U22(q, r)

        Ksum19(q, r) = F22(q, r) + G11(q, r)

        Ksum20(q, r) = G22(q, r) + H11(q, r) + VD22(q, r)

        Ksum21(q, r) = H22(q, r) + I11(q, r)

        Ksum22(q, r) = I22(q, r) + J11(q, r) + VE22(q, r)
```

```
            Ksum23(q, r) = J22(q, r) + K11(q, r)
            Ksum24(q, r) = K22(q, r) + L11(q, r)
            Ksum25(q, r) = L22(q, r) + M11(q, r)
            Ksum26(q, r) = M22(q, r) + N11(q, r) + VF22(q, r)
            Ksum27(q, r) = OA22(q, r) + PA11(q, r)
            Ksum28(q, r) = PA22(q, r) + QA11(q, r)
            Ksum29(q, r) = QA22(q, r) + RA11(q, r)
            Ksum30(q, r) = OB22(q, r) + PB11(q, r)
            Ksum31(q, r) = PB22(q, r) + QB11(q, r)
            Ksum32(q, r) = QB22(q, r) + RB11(q, r)
            Ksum33(q, r) = S22(q, r) + T11(q, r)
            Ksum34(q, r) = T22(q, r) + U11(q, r)

        Next r
    Next q


    Dim Ktotal(228, 228) As Double

' initialize the matrix with 0

    Erase Ktotal

' Generation of the Stiffness matrix of the STRUCTURE
    Dim a, b As Integer
    For a = 1 To 6
        For b = 1 To 6

            Ktotal(a, b) = A11(a, b)
            Ktotal(a, b + 6) = A12(a, b)
            Ktotal(a + 6, b) = A21(a, b)
            Ktotal(a + 12, b + 6) = B21(a, b)
```

Ktotal(a + 6, b + 12) = B12(a, b)

Ktotal(a + 18, b + 12) = C21(a, b)

Ktotal(a + 12, b + 18) = C12(a, b)

Ktotal(a + 24, b + 18) = DR21(a, b)

Ktotal(a + 18, b + 24) = DR12(a, b)

Ktotal(a + 30, b + 24) = E21(a, b)

Ktotal(a + 24, b + 30) = E12(a, b)

Ktotal(a + 36, b + 30) = F21(a, b)

Ktotal(a + 30, b + 36) = F12(a, b)

Ktotal(a + 42, b + 36) = G21(a, b)

Ktotal(a + 36, b + 42) = G12(a, b)

Ktotal(a + 48, b + 42) = H21(a, b)

Ktotal(a + 42, b + 48) = H12(a, b)

Ktotal(a + 54, b + 48) = I21(a, b)

Ktotal(a + 48, b + 54) = I12(a, b)

Ktotal(a + 60, b + 54) = J21(a, b)

Ktotal(a + 54, b + 60) = J12(a, b)

Ktotal(a + 66, b + 60) = K21(a, b)

Ktotal(a + 60, b + 66) = K12(a, b)

Ktotal(a + 72, b + 66) = L21(a, b)

Ktotal(a + 66, b + 72) = L12(a, b)

Ktotal(a + 78, b + 72) = M21(a, b)

Ktotal(a + 72, b + 78) = M12(a, b)

Ktotal(a + 84, b + 78) = N21(a, b)

Ktotal(a + 78, b + 84) = N12(a, b)

Ktotal(a + 84, b + 84) = N22(a, b)

Ktotal(a + 90, b + 90) = A11(a, b)

Ktotal(a + 96, b + 90) = A21(a, b)

Ktotal(a + 90, b + 96) = A12(a, b)

Ktotal(a + 102, b + 96) = B21(a, b)

Ktotal(a + 96, b + 102) = B12(a, b)

Ktotal(a + 108, b + 102) = C21(a, b)

Ktotal(a + 102, b + 108) = C12(a, b)

Ktotal(a + 114, b + 108) = DL21(a, b)

Ktotal(a + 108, b + 114) = DL12(a, b)

Ktotal(a + 120, b + 114) = E21(a, b)

Ktotal(a + 114, b + 120) = E12(a, b)

Ktotal(a + 126, b + 120) = F21(a, b)

Ktotal(a + 120, b + 126) = F12(a, b)

Ktotal(a + 132, b + 126) = G21(a, b)

Ktotal(a + 126, b + 132) = G12(a, b)

Ktotal(a + 138, b + 132) = H21(a, b)

Ktotal(a + 132, b + 138) = H12(a, b)

Ktotal(a + 144, b + 138) = I21(a, b)

Ktotal(a + 138, b + 144) = I12(a, b)

Ktotal(a + 150, b + 144) = J21(a, b)

Ktotal(a + 144, b + 150) = J12(a, b)

Ktotal(a + 156, b + 150) = K21(a, b)

Ktotal(a + 150, b + 156) = K12(a, b)

Ktotal(a + 162, b + 156) = L21(a, b)

Ktotal(a + 156, b + 162) = L12(a, b)

Ktotal(a + 168, b + 162) = M21(a, b)

Ktotal(a + 162, b + 168) = M12(a, b)

Ktotal(a + 174, b + 168) = N21(a, b)

Ktotal(a + 168, b + 174) = N12(a, b)

Ktotal(a + 174, b + 174) = N22(a, b)

Ktotal(a + 186, b + 180) = PA21(a, b)

Ktotal(a + 180, b + 186) = PA12(a, b)

Ktotal(a + 192, b + 186) = QA21(a, b)

Ktotal(a + 186, b + 192) = QA12(a, b)

Ktotal(a + 204, b + 198) = PB21(a, b)

Ktotal(a + 198, b + 204) = PB12(a, b)

Ktotal(a + 210, b + 204) = QB21(a, b)

Ktotal(a + 204, b + 210) = QB12(a, b)

Ktotal(a + 222, b + 216) = T21(a, b)

Ktotal(a + 216, b + 222) = T12(a, b)

Ktotal(a + 180, b + 6) = OA21(a, b)

Ktotal(a + 198, b + 18) = OB21(a, b)

Ktotal(a + 216, b + 30) = S21(a, b)

Ktotal(a + 132, b + 42) = VD21(a, b)

Ktotal(a + 144, b + 54) = VE21(a, b)

Ktotal(a + 168, b + 78) = VF21(a, b)

Ktotal(a + 192, b + 96) = RA12(a, b)

Ktotal(a + 210, b + 108) = RB12(a, b)

Ktotal(a + 222, b + 120) = U12(a, b)

Ktotal(a + 42, b + 132) = VD12(a, b)

Ktotal(a + 54, b + 144) = VE12(a, b)

Ktotal(a + 78, b + 168) = VF12(a, b)

Ktotal(a + 6, b + 180) = OA12(a, b)

Ktotal(a + 96, b + 192) = RA21(a, b)

Ktotal(a + 18, b + 198) = OB12(a, b)

Ktotal(a + 108, b + 210) = RB21(a, b)

Ktotal(a + 30, b + 216) = S12(a, b)

Ktotal(a + 120, b + 222) = U21(a, b)

Ktotal(a + 6, b + 6) = Ksum1(a, b)

Ktotal(a + 12, b + 12) = Ksum2(a, b)

Ktotal(a + 18, b + 18) = Ksum3(a, b)

Ktotal(a + 24, b + 24) = Ksum4(a, b)

Ktotal(a + 30, b + 30) = Ksum5(a, b)

Ktotal(a + 36, b + 36) = Ksum6(a, b)

Ktotal(a + 42, b + 42) = Ksum7(a, b)

Ktotal(a + 48, b + 48) = Ksum8(a, b)

Ktotal(a + 54, b + 54) = Ksum9(a, b)

Ktotal(a + 60, b + 60) = Ksum10(a, b)

```
            Ktotal(a + 66, b + 66) = Ksum11(a, b)
            Ktotal(a + 72, b + 72) = Ksum12(a, b)
            Ktotal(a + 78, b + 78) = Ksum13(a, b)
            Ktotal(a + 96, b + 96) = Ksum14(a, b)
            Ktotal(a + 102, b + 102) = Ksum15(a, b)
            Ktotal(a + 108, b + 108) = Ksum16(a, b)
            Ktotal(a + 114, b + 114) = Ksum17(a, b)
            Ktotal(a + 120, b + 120) = Ksum18(a, b)
            Ktotal(a + 126, b + 126) = Ksum19(a, b)
            Ktotal(a + 132, b + 132) = Ksum20(a, b)
            Ktotal(a + 138, b + 138) = Ksum21(a, b)
            Ktotal(a + 144, b + 144) = Ksum22(a, b)
            Ktotal(a + 150, b + 150) = Ksum23(a, b)
            Ktotal(a + 156, b + 156) = Ksum24(a, b)
            Ktotal(a + 162, b + 162) = Ksum25(a, b)
            Ktotal(a + 168, b + 168) = Ksum26(a, b)
            Ktotal(a + 180, b + 180) = Ksum27(a, b)
            Ktotal(a + 186, b + 186) = Ksum28(a, b)
            Ktotal(a + 192, b + 192) = Ksum29(a, b)
            Ktotal(a + 198, b + 198) = Ksum30(a, b)
            Ktotal(a + 204, b + 204) = Ksum31(a, b)
            Ktotal(a + 210, b + 210) = Ksum32(a, b)
            Ktotal(a + 216, b + 216) = Ksum33(a, b)
            Ktotal(a + 222, b + 222) = Ksum34(a, b)


        Next b
    Next a

'\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

'Print the stiffness matrix into excel worksheet to re-arrange it
```

```
Dim KtotalVertical(218, 218) As Double
Dim KtotalTorsional(221, 221) As Double

'rearranging for vertical
    For i = 1 To 12
        For j = 1 To 12
            KtotalVertical(i, j) = Ktotal(i, j)
        Next j

        For k = 13 To 58
            KtotalVertical(i, k) = Ktotal(i, k + 3)
        Next k

        For l = 59 To 97
            KtotalVertical(i, l) = Ktotal(i, l + 5)
        Next l

        For m = 98 To 143
            KtotalVertical(i, m) = Ktotal(i, m + 8)
        Next m

        For n = 144 To 218
            KtotalVertical(i, n) = Ktotal(i, n + 10)
        Next n

    Next i


    For i = 13 To 58
        For j = 1 To 12
            KtotalVertical(i, j) = Ktotal(i + 3, j)
```

```
    Next j

    For k = 13 To 58
        KtotalVertical(i, k) = Ktotal(i + 3, k + 3)
    Next k

    For l = 59 To 97
        KtotalVertical(i, l) = Ktotal(i + 3, l + 5)
    Next l

    For m = 98 To 143
        KtotalVertical(i, m) = Ktotal(i + 3, m + 8)
    Next m

    For n = 144 To 218
        KtotalVertical(i, n) = Ktotal(i + 3, n + 10)
    Next n

Next i

For i = 59 To 97
    For j = 1 To 12
        KtotalVertical(i, j) = Ktotal(i + 5, j)
    Next j

    For k = 13 To 58
        KtotalVertical(i, k) = Ktotal(i + 5, k + 3)
    Next k

    For l = 59 To 97
        KtotalVertical(i, l) = Ktotal(i + 5, l + 5)
    Next l
```

```
    For m = 98 To 143
        KtotalVertical(i, m) = Ktotal(i + 5, m + 8)
    Next m

    For n = 144 To 218
        KtotalVertical(i, n) = Ktotal(i + 5, n + 10)
    Next n

Next i

For i = 98 To 143
    For j = 1 To 12
        KtotalVertical(i, j) = Ktotal(i + 8, j)
    Next j

    For k = 13 To 58
        KtotalVertical(i, k) = Ktotal(i + 8, k + 3)
    Next k

    For l = 59 To 97
        KtotalVertical(i, l) = Ktotal(i + 8, l + 5)
    Next l

    For m = 98 To 143
        KtotalVertical(i, m) = Ktotal(i + 8, m + 8)
    Next m

    For n = 144 To 218
        KtotalVertical(i, n) = Ktotal(i + 8, n + 10)
    Next n
```

```
    Next i

    For i = 144 To 218
        For j = 1 To 12
            KtotalVertical(i, j) = Ktotal(i + 10, j)
        Next j

        For k = 13 To 58
            KtotalVertical(i, k) = Ktotal(i + 10, k + 3)
        Next k

        For l = 59 To 97
            KtotalVertical(i, l) = Ktotal(i + 10, l + 5)
        Next l

        For m = 98 To 143
            KtotalVertical(i, m) = Ktotal(i + 10, m + 8)
        Next m

        For n = 144 To 218
            KtotalVertical(i, n) = Ktotal(i + 10, n + 10)
        Next n

    Next i


'rearranging for torsional

    For i = 1 To 60
        For j = 1 To 60
            KtotalTorsional(i, j) = Ktotal(i, j)
        Next j
```

```
    For k = 61 To 147
        KtotalTorsional(i, k) = Ktotal(i, k + 3)
    Next k

    For l = 148 To 200
        KtotalTorsional(i, l) = Ktotal(i, l + 6)
    Next l

    For m = 201 To 221
        KtotalTorsional(i, m) = Ktotal(i, m + 7)
    Next m

Next i

For i = 61 To 147
    For j = 1 To 60
        KtotalTorsional(i, j) = Ktotal(i + 3, j)
    Next j

    For k = 61 To 147
        KtotalTorsional(i, k) = Ktotal(i + 3, k + 3)
    Next k

    For l = 148 To 200
        KtotalTorsional(i, l) = Ktotal(i + 3, l + 6)
    Next l

    For m = 201 To 221
        KtotalTorsional(i, m) = Ktotal(i + 3, m + 7)
    Next m
```

```
 Next i


For i = 148 To 200
    For j = 1 To 60
       KtotalTorsional(i, j) = Ktotal(i + 6, j)
    Next j

    For k = 61 To 147
       KtotalTorsional(i, k) = Ktotal(i + 6, k + 3)
    Next k

    For l = 148 To 200
       KtotalTorsional(i, l) = Ktotal(i + 6, l + 6)
    Next l

    For m = 201 To 221
       KtotalTorsional(i, m) = Ktotal(i + 6, m + 7)
    Next m


 Next i


  For i = 201 To 221
    For j = 1 To 60
       KtotalTorsional(i, j) = Ktotal(i + 7, j)
    Next j

    For k = 61 To 147
       KtotalTorsional(i, k) = Ktotal(i + 7, k + 3)
    Next k
```

```
            For l = 148 To 200
                KtotalTorsional(i, l) = Ktotal(i + 7, l + 6)
            Next l

            For m = 201 To 221
                KtotalTorsional(i, m) = Ktotal(i + 7, m + 7)
            Next m


        Next i

'Read the re-arranged matrix


    Dim Vertical_Stiffness_Matrix() As Variant
    Dim Torsional_Stiffness_Matrix() As Variant
    Dim Vertical_Forces_vector(218, 1) As Double
    Dim Torsional_Forces_vector(221, 1) As Double
    Dim Vertical_displacement(), Torsional_displacement() As Variant

    Erase Vertical_Forces_vector
    Erase Torsional_Forces_vector

    Vertical_Forces_vector(36, 1) = -1000
    Vertical_Forces_vector(121, 1) = -1000

    Torsional_Forces_vector(15, 1) = -1000
    Torsional_Forces_vector(102, 1) = 1000


'Invert the matrix to resolve with setup
```

```
    Vertical_displacement = WorksheetFunction.MMult(WorksheetFunction.MInverse(KtotalVertical), Vertical_Forces_vector)

    Torsional_displacement = WorksheetFunction.MMult(WorksheetFunction.MInverse(KtotalTorsional), Torsional_Forces_vector)



'Get the mass of the structure

    Mass_total = 2 * (Mass(1) + Mass(2) + Mass(3) + (Mass(4) + Mass(5)) / 2 + Mass(6) + Mass(7) + Mass(8) + Mass(9) + Mass(10) + Mass(11) + Mass(12) + _

                Mass(13) + Mass(14) + Mass(15)) + Mass(16) + Mass(17) + Mass(18) + Mass(19) + Mass(20) + Mass(21) + Mass(22) + Mass(23) + Mass(24) + _

                Mass(25) + Mass(26) + Mass(27) + Mass(28) + Mass(29)



'Get the Stiffness of the structure



    Vertical_Stiffness = ((2000 * (WB * 1000) ^ 3) / (48 * Abs((Vertical_displacement(36, 1) + Vertical_displacement(121, 1)) / 2))) * 0.000000000001

    Torsional_Stiffness = (((1000 * (W1 * 1000) ^ 2) / (Abs(Torsional_displacement(15, 1)) + Abs(Torsional_displacement(102, 1)))) * (WB * 1000)) * 0.000000000001



    'Range("MyArray9") = Vertical_displacement

    'Range("MyArray10") = Torsional_displacement



'Print the results into the worksheet



   ' Range("TORSIONAL") = Torsional_Stiffness

   ' Range("VERTICAL") = Vertical_Stiffness

   ' Range("MASS") = Mass_total



'\\\\\\\\\\\\\\\\\\\\

'Objective function



    Dim SOLUTION As Variant
```

```
        If Limit_Vert_Stiff > Vertical_Stiffness Or Limit_Tors_Stiff > Torsional_Stiffness Then

            SOLUTION = Mass_total + Torsional_Stiffness * 10000 + Vertical_Stiffness * 10000

        Else

            SOLUTION = Mass_total + Torsional_Stiffness + Vertical_Stiffness

        End If

    CALL_MULTVAR_OBJ_3_FUNC = SOLUTION

Exit Function

ERROR_LABEL:

CALL_MULTVAR_OBJ_3_FUNC = Err.Number

End Function
```

Módulo: STAT_MIN_MAX_LIBR

```
Option Explicit     'Requires that all variables to be declared explicitly.
Option Base 1       'The "Option Base" statement allows to specify 0 or 1 as the
                    'default first index of arrays.

'*********************************************************************************
'*********************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*********************************************************************************
'*********************************************************************************
'FUNCTION      : MAXIMUM_FUNC
'DESCRIPTION   : COMPARE TWO VALUES, AND RETURNS THE MAX VALUE
'LIBRARY       : STATISTICS
'GROUP         : MAX-MIN
'ID            : 001
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 14/02/2008
'*********************************************************************************
'*********************************************************************************


Function MAXIMUM_FUNC(ByVal FIRST_VAL As Variant, _
ByVal SECOND_VAL As Variant)

On Error GoTo ERROR_LABEL

  MAXIMUM_FUNC = FIRST_VAL
  If SECOND_VAL > FIRST_VAL Then MAXIMUM_FUNC = SECOND_VAL
```

```
Exit Function
ERROR_LABEL:
MAXIMUM_FUNC = Err.Number
End Function

'*****************************************************************************************
'*****************************************************************************************
'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,
'San Francisco, CA. USA  www.rnfc.org
'nfermincota.hba2005@ivey.ca
'*****************************************************************************************
'*****************************************************************************************
'FUNCTION      : MINIMUM_FUNC
'DESCRIPTION   : COMPARE TWO VALUES, AND RETURNS THE MIN VALUE
'LIBRARY       : STATISTICS
'GROUP         : MAX-MIN
'ID            : 002
'AUTHOR        : RAFAEL NICOLAS FERMIN COTA
'LAST UPDATE   : 14/02/2008
'*****************************************************************************************
'*****************************************************************************************

Function MINIMUM_FUNC(ByVal FIRST_VAL As Variant, _
ByVal SECOND_VAL As Variant)

On Error GoTo ERROR_LABEL

  MINIMUM_FUNC = FIRST_VAL
  If SECOND_VAL < FIRST_VAL Then MINIMUM_FUNC = SECOND_VAL

Exit Function
ERROR_LABEL:
```

```
MINIMUM_FUNC = Err.Number

End Function

'*****************************************************************************************

'*****************************************************************************************

'© Copyright NicoSystem 2009. All rights reserved by Rafael Nicolas Fermin Cota,

'San Francisco, CA. USA  www.rnfc.org

'nfermincota.hba2005@ivey.ca

'*****************************************************************************************

'*****************************************************************************************

'FUNCTION      : COLLAR_FUNC

'DESCRIPTION   : Collar(a; b; c) = max(a; min(b; c)).

'LIBRARY       : STATISTICS

'GROUP         : MAX-MIN

'ID            : 003

'AUTHOR        : RAFAEL NICOLAS FERMIN COTA

'LAST UPDATE   : 14/02/2008

'*****************************************************************************************

'*****************************************************************************************


Function COLLAR_FUNC(ByVal MIN_VAL As Variant, _

ByVal VAR_VAL As Variant, _

ByVal MAX_VAL As Variant) 'As Double


On Error GoTo ERROR_LABEL


If VAR_VAL < MIN_VAL Then

   COLLAR_FUNC = MIN_VAL

ElseIf VAR_VAL > MAX_VAL Then

   COLLAR_FUNC = MAX_VAL

Else

   COLLAR_FUNC = VAR_VAL
```

```
End If


Exit Function

ERROR_LABEL:

COLLAR_FUNC = Err.Number

End Function
```

Módulo: WINDOWS_INIT

```
Sub Button2_Click()

UserForm1.Show



End Sub
```