Resumen

En esta memoria se desarrolla la modelización y simulación del comportamiento termodinámico de motores Otto y Diesel, así como la optimización de sus principales parámetros termodinámicos. El software utilizado es Matlab.

Se han introducido determinados parámetros característicos en Matlab con la finalidad de aproximarse lo máximo posible a la realidad. Estas variables son características de cada tipo de motor.

Con el objetivo de modelar y obtener los parámetros óptimos de nuestro motor, en segundo lugar se ha simulado gráficamente y teóricamente los ciclos tanto Otto como Diesel y se han hallado ejemplos de puntos óptimos de interés.

Una correcta optimización de un motor mediante un modelado previo implica un incremento de sus prestaciones y su economía, así como una menor emisión de elementos contaminantes.

Sumario

1. Glossario

$PCI_{gasolina}$	Poder Calorífico Inferior de la gasolina [$J \cdot g^{-1}$]
PCI_{diesel}	Poder Calorífico Inferior del diesel [$J \cdot g^{-1}$]
w_{giro}	Régimen de giro del motor [rpm]
m_molar_{aire}	Masa molar del aire seco [$g \cdot mol^{-1}$]
$masa\_molar_{diesel}$	Masa molar del diesel [$g \cdot mol^{-1}$]
R	Constante universal de los gases [0.082 $atm \cdot L \cdot K^{-1} \cdot mol^{-1}$]
p_{otto}	Presión en el estado 1 en el ciclo Otto [atm]
p_{diesel}	Presión en el estado 1 en el ciclo Diesel [atm]
c_v	Capacidad calorífica a volumen constante [$J \cdot g^{-1} \cdot K^{-1}$]
c_p	Capacidad calorífica a presión constante [$J \cdot g^{-1} \cdot K^{-1}$]
ε_{Otto}	Relación de compresión para el ciclo Otto
ε_{diesel}	Relación de compresión para el ciclo Diesel
n	Número de moles [moles]
AFR	Relación Aire / Combustible ciclo Otto
AFD	Relación Aire / Combustible ciclo Diesel
λ	Pobreza de la mezcla
Fe	Dosado estequiométrico
V_n	Volumen en el estado n [L]
η_{VOtto}	Rendimiento volumétrico ciclo Otto
$\eta_{vDiesel}$	Rendimiento volumétrico ciclo Diesel
η_{cOtto}	Rendimiento de la combustión en ciclo Otto
$\eta_{cDiesel}$	Rendimiento de la combustión en ciclo Diesel
η_t	Rendimiento termodinámico

$Cte_{compresión}$	Constante de compresión
$Cte_{expansión}$	Constante de expansión
K	Constante de evolución politrópica
x	Incremento de volumen en las politrópicas [L]
z	Incremento de cualquier variable en funciones
T_n	Temperatura en el estado n [K]
m_aire	Masa de aire [g]
m_combustible	Masa de combustible en el ciclo Otto [g]
m_diesel	Masa de combusbile en el ciclo Diesel [g]
Q	Calor aportada al ciclo [J]
MFB	Porcentaje de masa quemada en la combustión
$W_{expansión}$	Trabajo de expansión [atm·L]
$W_{compresión}$	Trabajo de compresión [atm·L]
WJ	Trabajo útil del ciclo [J]
P	Potencia por ciclo [W]
ge	Consumo específico $[g\cdot(KW\cdot h)^{-1}]$
Γ	Par [N·m]
pme	Presión media específica [atm]
otto	Variable booleana que es 1 si el ciclo es Otto
diesel	Variable booleana que es 1 si el ciclo es Diesel

2. Prefacio

2.1. Origen del proyecto

En 1862 fue enunciado el ciclo de volumen constante por Beau de Rochar con el título "ciclo de cuatro tiempos". Posteriormente el alemán Otto lo aplicó a un motor térmico denominándolo como ciclo Otto.

Si históricamente Carl Benz ha sido considerado como el padre del automóvil, ya que en 1885 fue el primer constructor de un motor de cuatro tiempos de encendido por bujías, hay que remontarse al año 1860 para encontrar los primeros experimentos sobre motores de combustión interna. El primer antecedente al motor de Carl Benz, fue ideado por un belga de fértil imaginación llamado Etienne Lenoir, que construyó su primer modelo práctico veinticinco años antes que Benz, y que en aquel momento abrió la puerta de la evolución y estableció una serie de principios técnicos que han permanecido inmutables hasta hace pocos años. La primera vez que el ciclo de cuatro tiempos se empleó con éxito fue en 1876, en un motor construido por un ingeniero alemán, el conde Nicholas Otto.

En 1895, Rudolf Diesel presentó por primera vez su invento al público. Un motor con encendido por compresión. En comparación con el ya acreditado motor de explosión Otto, este motor tenía las ventajas de consumir mucho menos y de poder funcionar con un combustible relativamente barato, siendo posible además alcanzar potencias muy superiores.

Desde entonces hasta hoy en día la evolución de estos motores ha sido sorprendente. Aspectos como la modelización de parámetros termodinámicos y su optimización han ayudado claramente a que estos motores puedan y sigan evolucionando no solo tecnológicamente sino también en temas medioambientales.

2.2. Motivación

Después de haber cursado y superado la asignatura *Máquinas Térmicas* impartida en cuarto curso de Ingeniería Industrial, la motivación por realizar un proyecto sobre motores térmicos ciclo Otto y Diesel aumentó considerablemente.

3. Introducción

3.1. Objetivos del proyecto

3.2. Abaste del proyecto

El estudio se centra en la modelización y simulación de los ciclos teóricos Otto y Diesel ya dejándose para futuros estudios el desarrollo del diagrama indicado y su correspondiente simulación. Por otra parte se toman los resultados de la simulación como ciertos y se pretende que en los futuros estudios se corroboren los resultados con hechos experimentales.

3.3. Conceptos previos

3.3.1. El motor térmico

Un motor térmico o máquina térmica es un artefacto que convierte energía térmica en trabajo mecánico por medio del aprovechamiento del gradiente de temperatura entre una "fuente" caliente y un "sumidero" frío. El calor se transfiere de la fuente al sumidero y, durante este proceso, algo del calor se convierte en trabajo por medio del aprovechamiento de las propiedades de un fluido de trabajo, usualmente un gas o un líquido.

Un motor de combustión interna es un tipo de máquina que obtiene energía mecánica directamente de la energía química producida por un combustible que arde dentro de una cámara de combustión, la parte principal de un motor.

3.3.2. Ciclos termodinámicos

Representado en un diagrama P-V, un ciclo termodinámico adopta la forma de una curva cerrada. En este diagrama el volumen de un sistema es representado en abscisas y la presión en ordenadas de forma que el trabajo por cambio de volumen es igual al área descrita entre la línea que representa el proceso y el eje de abscisas.

$$W = \int P \, dV$$

El sentido de avance de la curva, indicado por las puntas de flecha, nos indica si el incremento de volumen es positivo (hacia la derecha) o negativo (hacia la izquierda) y, como consecuencia, si el trabajo es positivo o negativo, respectivamente.

Por lo tanto, se puede concluir que el área encerrada por la curva que representa un ciclo termodinámico en este diagrama, indica el trabajo total realizado (en un ciclo completo) por el sistema, si éste avanza en sentido horario o, por el contrario, el trabajo total ejercido sobre el sistema si lo hace en sentido anti horario.

Ciclo de Carnot

El ciclo de Carnot es un ciclo termodinámico ideal reversible entre dos fuentes de temperatura, en el cual el rendimiento es máximo. Actualmente no se ha conseguido implementar este ciclo en ninguna máquina de manera eficiente.

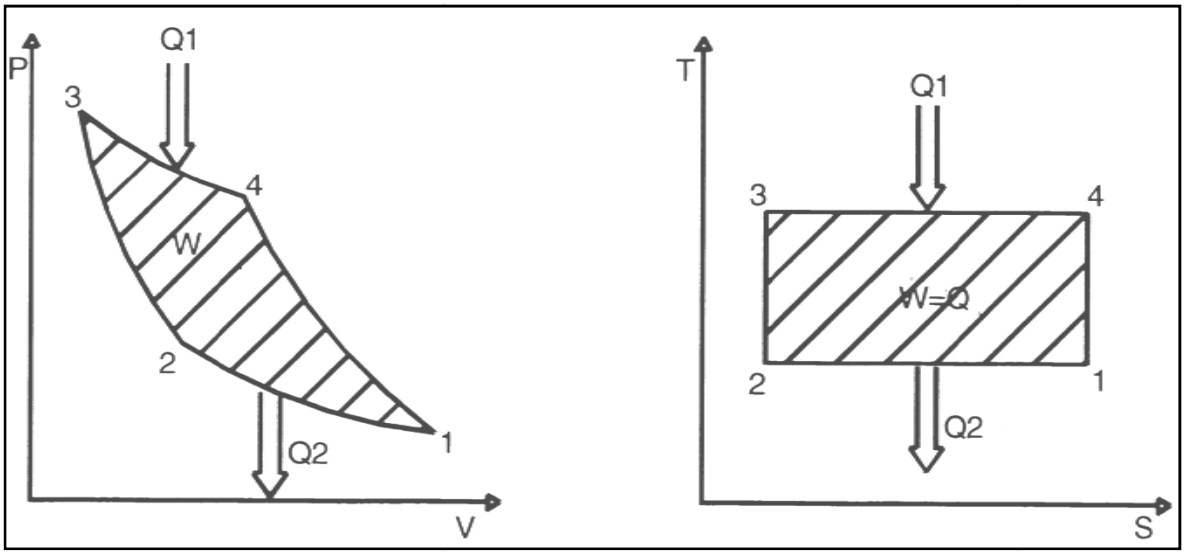


Figura 3.1: Ciclo de Carnot [1]

Ciclo Otto

El ciclo Otto es el ciclo termodinámico ideal que se aplica en los motores de combustión interna. Se caracteriza porque todo el calor se aporta a volumen constante. En los motores de 4 tiempos, la extracción de calor se realiza en la fase de escape y la admisión de la nueva carga se realiza en la fase de admisión. Los procesos termodinámicos que se producen están representados en la figura 3.2 y son los siguientes:

a) 1-2 *Compresión adiabática*: compresión del fluido de trabajo, el pistón tiene que realizar el trabajo de compresión W1.
b) 2-3 *Aportación de calor a volumen constante*: introducción instantánea del calor aportado Q1.
c) 3-4 *Expansión adiabática*: expansión, correspondiente al trabajo W2, realizado por el fluido de trabajo.
d) 4-1 *Extracción de calor a volumen constante*: extracción instantánea del calor Q2.

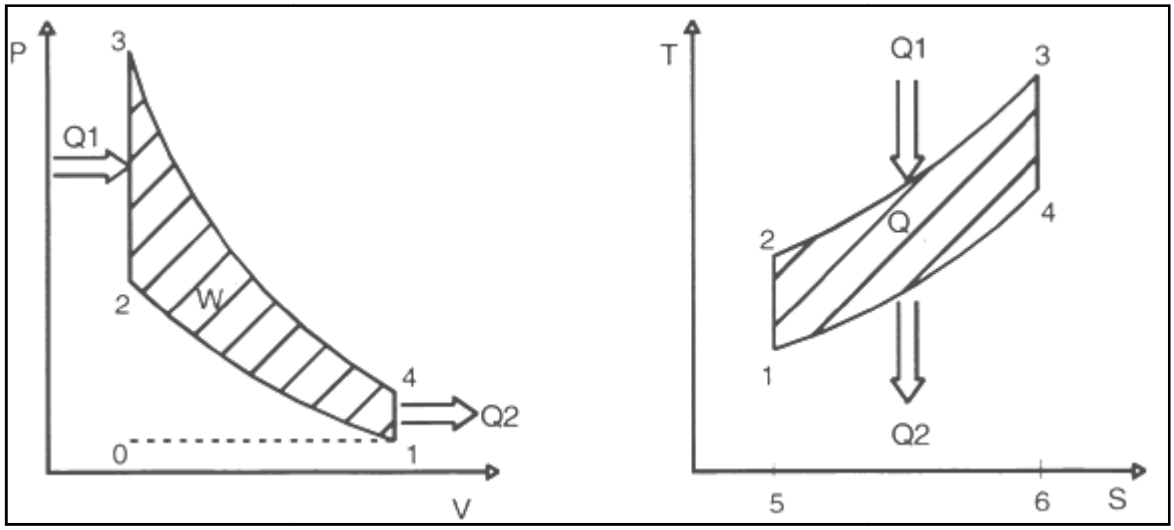


Figura 3.2: Ciclo Otto [1]

Los procesos 1-0 y 0-1 de la figura 3.2 quedan representados por una línea horizontal discontínua corresponden a la admisión y escape. En el diagrama teórico estos dos procesos se anulan dado que tienen una pérdida o ganancia de calor nulos. [1]

Ciclo Diesel

La diferencia fundamental entre el ciclo Diesel y el ciclo Otto está en la fase de aportación de calor. En el ciclo Otto el calor era introducido a volumen constante, y en el ciclo Diesel es introducido a presión constante.

Los procesos termodinámicos que se producen están representados en la figura 3.3 y son los siguientes:

a) 1-2 *Compresión adiabática*
b) 2-3 *Aportación de calor a presión constante*
c) 3-4 *Expansión adiabática*
d) 4-1 *Extracción de calor a volumen constante*

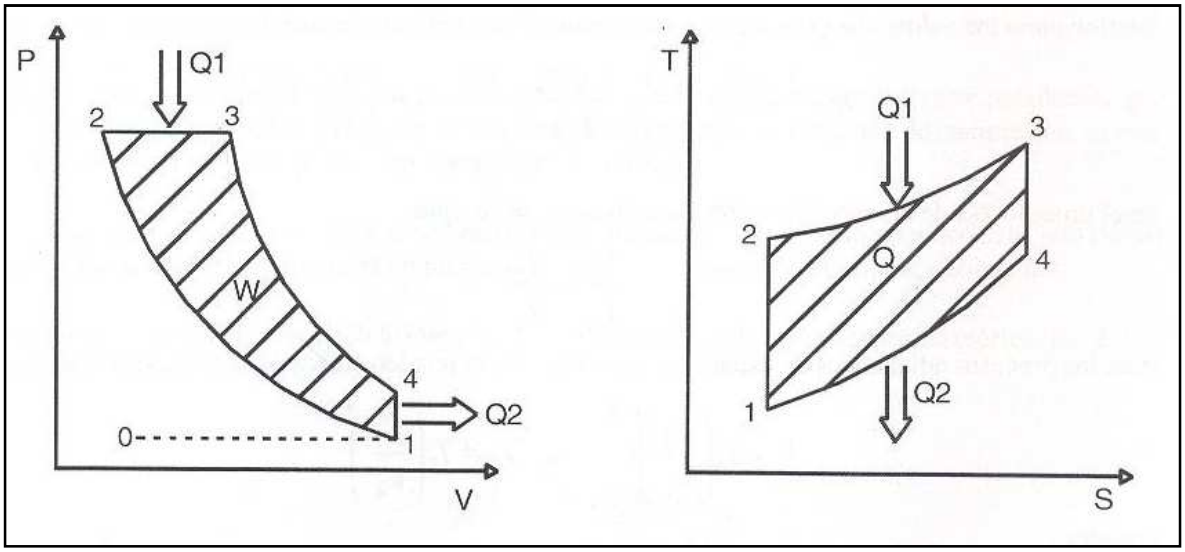


Figura 3.3: Ciclo Diesel [1]

4. Modelización

4.1. Ciclo Otto

4.1.1. Compresión

Para modelizar el comportamiento termodinámico de un ciclo Otto primero se introducen las características iniciales para poder representar la primera politrópica en Matlab.

Introducimos valores a las variables c_p, c_v, V_c, T1, p_{otto} y ε_{Otto}.

Una vez introducidas se dispone a realizar la primera curva politrópica que corresponde a la compresión. Para la realización de esta curva se decide una resolución de 20 puntos.

Partiendo del estado inicial se define la constante de compresión:

$$Cte_{\text{compresión}} = P_{0\text{tto}} \times {V_{\text{cc}}}^{K}$$

Definiendo previamente $K = \frac{c_p}{c_v}$

A través de la relación de compresión se conoce el volumen al finalizar la compresión.

$$V_{20} = \frac{V_{\text{cc}}}{\varepsilon_{0\text{tto}}}$$

Una vez sabido el volumen al final de la compresión se puede empezar a realizar la politrópica partiendo del volumen inicial (V_c) y llegando hasta el volumen al final de la compresión (V_{20}) en 20 pasos, así obteniendo en cada paso la presión y el volumen correspondiente con las siguientes fórmulas:

$$x = \frac{V_1 - V_{20}}{20}$$

$$V_{n+1} = V_n - x$$

$$P_{n+1} = P_n \times \frac{V_n}{V_{n+1}}^{K}$$

De esta forma obtenemos en el estado 20 las variables termodinámicas de presión y volumen al final de la compresión.

4.1.2. Aportación de calor

Una vez realizada la compresión, se desarrolla la aportación de calor a volumen constante. Para realizar este paso, se precisan definir las siguientes variables:

Rendimiento volumétrico: Masa de mezcla fresca que entra en el cilindro dividido por la masa de la mezcla que entraría si esta llenase todo el volumen desplazado por el pistón y estuviera a la densidad de la admisión antes de la mariposa (a nivel del filtro del aire). En el ciclo Otto el rendimiento volumétrico es siempre inferior al 100 % porque la mariposa limita la entrada de aire y se ha supuesto que no es un motor sobrealimentado. [2]

El rendimiento volumétrico depende del régimen de giro del motor [3]. Se ha diseccionado la curva de la gráfica de la figura 4.1 en 20 puntos para poder introducirla al Matlab.

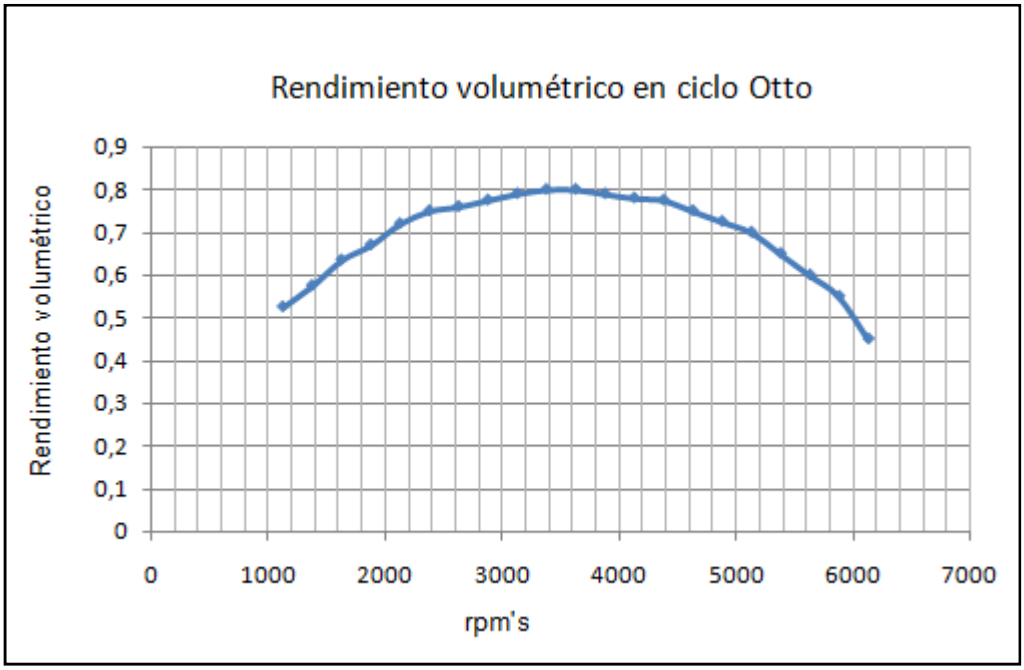


Figura 4.1: Rendimiento volumétrico en función del régimen de giro del motor.

Rendimiento de la combustión: El calor que se puede obtener en una combustión es el correspondiente al Poder Calorífico Inferior del combustible (PCI). Al realizar la combustión, una parte del calor producido se pierde, asociado a los productos de la combustión.

Este parámetro depende esencialmente del régimen de giro del motor y de la velocidad de combustión [4]. Para realizar una distinción entre las velocidades de combustión se diferencia entre una combustión laminar, una combustión turbulenta o una combustión casi instantánea. En nuestro diseño se puede escoger entre uno de estos tres tipos de combustión.

La figura 4.2 muestra el rendimiento de la combustión en función del régimen de giro y de la velocidad de combustión.

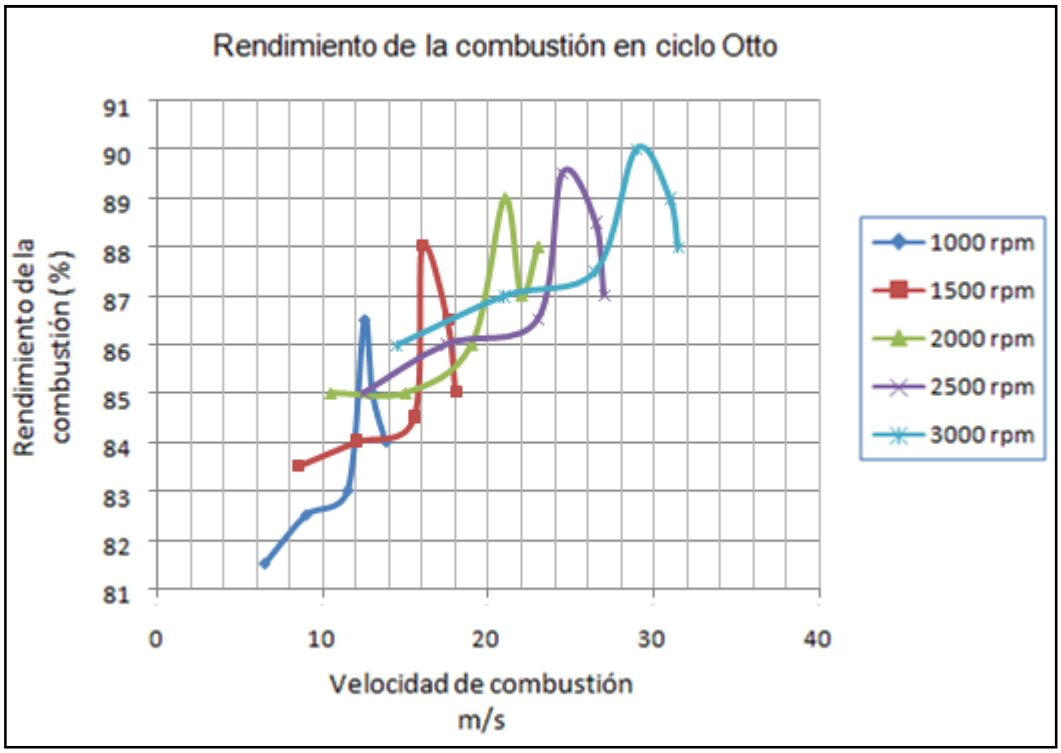


Figura 4.2: Rendimiento de la combustión en ciclo Otto

Una vez se han introducido estos dos parámetros en el programa, se procede a la aportación de calor primero utilizando la ecuación de los gases ideales:

$$n = \frac{P_{0tto} \times V_c \times \eta_{v0tto}}{R \times T1}$$

$$m\_aire = m\_molar_{aire} \times n$$

$$m\_combustible = \frac{m\_aire}{AFR}$$

Cuando se tiene la masa de combustible, se puede obtener el calor aportado:

$$Q = m\_combustible \times PCI_{gasolina} \times \eta_{c0tto}$$

Una vez se tiene el calor aportado, se puede encontrar la temperatura al final de la aportación del calor (denominado estado 30).

$$\Delta T = \frac{Q}{(m\_aire + m\_combustible) \times c_v}$$

$$T_{20}=T_1\times\left(\frac{V_c}{V_{20}}\right)^{k-1}$$

$$T_{30}=T_{20}+\Delta T$$

La aportación de calor es a volumen constante, por lo tanto, el volumen antes y después de la aportación de calor es el mismo.

La presión al final de la aportación de calor se obtiene mediante la aplicación de la ecuación de los gases ideales:

$$p_{30}=\frac{n\times R\times T_{30}}{V_{30}}$$

4.1.3. Expansión

Para realizar la expansión, primero se determina la constate de evolución politrópica (K) a partir de las condiciones finales de la aportación de calor.

Muchos autores relatan métodos diferentes para encontrar esta constante de evolución politrópica. Cada uno de ellos depende de unas variables diferentes. A continuación se presentan algunos de ellos y luego se decide cuál se utilizará. [5]

-Función de Gatowski:

$$K = K_o\text{-}\frac{K_1 x(T\text{-}T_{ref})}{1000}$$

Dónde K_o es un valor de referencia (1.38), K_1 es una constante (0.88) y T_{ref} es una temperatura de referencia (300K).

-Función de Brunt et al.:

$$K=1.338\text{-}6x10^{-5}xT+10^{-8}xT^{-2}$$

-Función de Egnell:

$$K=K_o\text{-}k_1\frac{\text{-}k_2}{T}$$

Dónde K_o es un valor de referencia (1.38), k_1 y k_2 son constantes (0.2 y 900).

Se presenta una última función denominada “función de las mezclas quemadas y sin quemar” que se obtiene experimentalmente. Esta última función se aproxima casi a la perfección a los modelos reales con un error que no llega al 1%.

En la gráfica 4.3 se presentan las diferentes funciones para ver de una manera más visual la diferencia entre ellas.

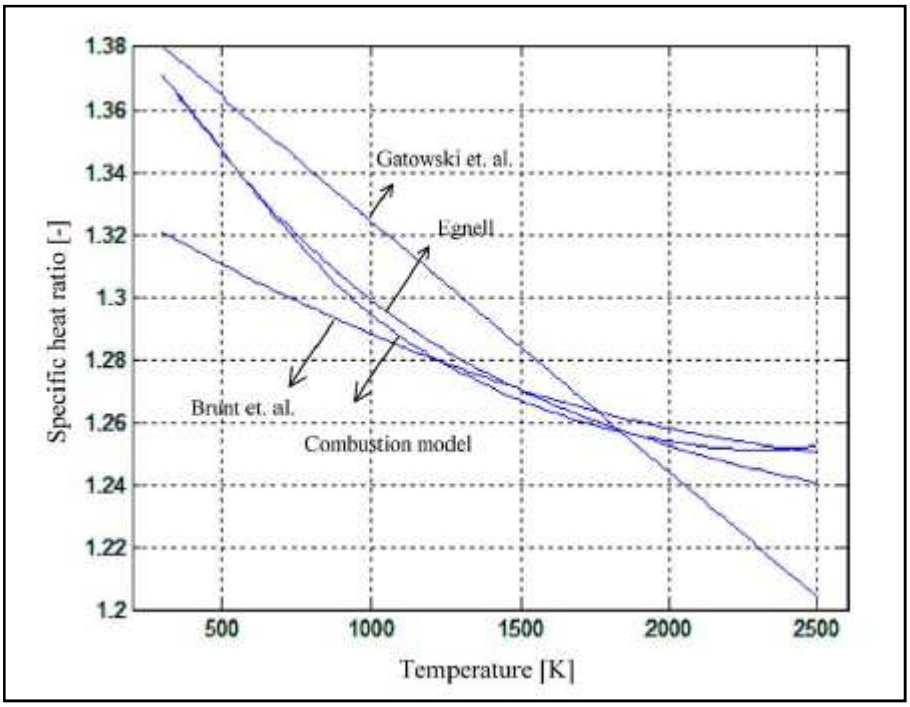


Figura 4.3: Funciones para hallar K en función de la temperatura caso mezcla quemada [5]

Como se puede observar en la figura 4.3, las funciones de Gatowski, Brunt et al difieren bastante de la función obtenida experimentalmente mientras que la de Egnell se le parece bastante.

Como hemos podido observar, la K depende de la fracción de masa quemada en el proceso de combustión y de la temperatura al final de la aportación de calor. Se supone que en la combustión de nuestro modelo se ha quemado todo el combustible, por lo tanto se considera MFB=1.

Como se ha comentado, la función obtenida experimentalmente apenas llega al 1% de error respecto la real y la función de Egnell no discrepa excesivamente. Se precisa observar el comportamiento de las funciones también en el caso de que la mezcla no se queme, aún no pasando en nuestro modelo. La figura 4.4 lo muestra.

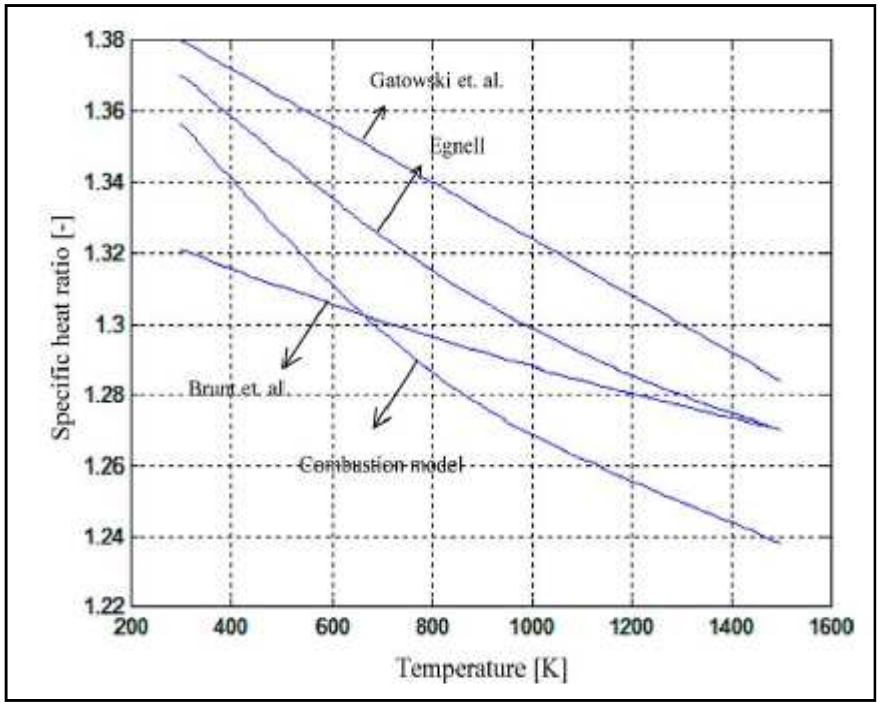


Figura 4.4: Funciones para hallar K en función de la temperatura caso mezcla sin quemar [5]

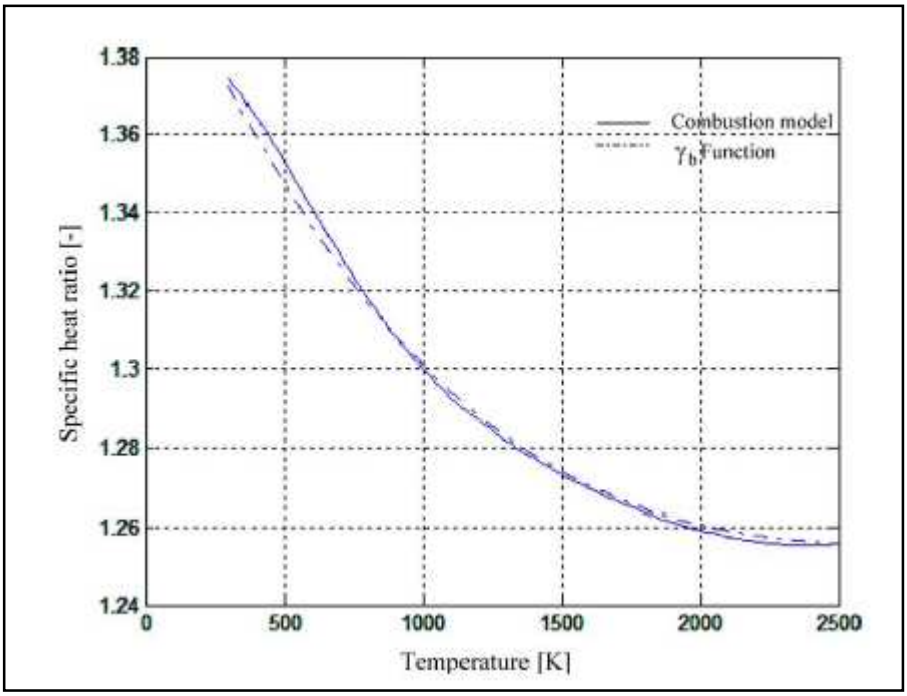


Figura 4.5: Constante de expansión K en función de la temperatura [5]

En la figura 4.4, la función de Egnell y la función experimental difieren bastante, por lo tanto, nos quedaremos con la función experimental.

La figura 4.5 muestra la función experimental para obtener la K en función de la temperatura al final de la aportación de calor. Se introduce este gráfico con una resolución de 20 puntos al modelo.

Una vez introducido estos parámetros y calculado la nueva K, se dispone a realizar la curva politrópica que corresponde a la expansión. Para la realización de esta curva se decide una resolución de 20 puntos.

Partiendo del estado final de la aportación de calor se define la constante de expansión:

$$Cte_{\text{expansión}} = P_{30} \times V_{30}{}^{K}$$

Una vez sabido el volumen al final de la expansión (V_c) se puede empezar a realizar la politrópica partiendo del volumen inicial (V_{30}) y llegando hasta el volumen al final de la compresión (V_1) en 20 pasos, así obteniendo en cada paso la presión y el volumen correspondiente con las siguientes fórmulas:

$$x = \frac{V_1\text{-}V_{30}}{20}$$

$$V_{n+1}=V_n+x$$

$$P_{n+1}=P_n\times\frac{V_n}{V_{n+1}}^{K}$$

El ciclo termodinámico se cierra pasando del final de la expansión al inicio de la compresión. Se supone que el volumen al final de la expansión y al principio de la compresión es el mismo.

4.2. Ciclo Diesel

El ciclo Diesel es diferente al ciclo Otto, la principal diferencia reside en la aportación de calor, en el ciclo Otto se realiza a volumen constante y en el ciclo Diesel se desarrolla a presión constante. Algunas diferencias menos significativas como relaciones de compresión más elevadas o el tipo de combustible también hacen que el ciclo Diesel difiera del ciclo Otto.

4.2.1. Compresión

La compresión en un ciclo Diesel es similar a la de un ciclo Otto. Las diferencias radican en el valor de las variables, por ejemplo, la relación de compresión de un motor Diesel es más elevada que la de un motor Otto. Se ha supuesto que el ciclo Diesel es sobrealimentado, por lo tanto la presión inicial del ciclo Diesel es mayor que la del ciclo Otto.

Las únicas variables que difieren de la modelización de esta primera etapa del ciclo Otto son la relación de compresión (ε_{diesel}) y la presión inicial (p_{diesel}).

Una vez introducidas se dispone a realizar la primera curva politrópica que corresponde a la compresión. Para la realización de esta curva se decide una resolución de 20 puntos.

Partiendo del estado inicial se define la constante de compresión:

$$Cte_{\text{compresión}} = P_{\text{diesel}} \times {V_{\text{cc}}}^{K}$$

Definiendo previamente $K = \frac{c_p}{c_v}$

A través de la relación de compresión se conoce el volumen al finalizar la compresión.

$$V_{20} = \frac{V_{\text{cc}}}{\varepsilon_{\text{diesel}}}$$

Una vez sabido el volumen al final de la compresión se puede empezar a realizar la politrópica partiendo del volumen inicial (V_1) y llegando hasta el volumen al final de la compresión (V_{20}) en 20 pasos, así obteniendo en cada paso la presión y el volumen correspondiente con las siguiente fórmulas:

$$x = \frac{V_1 - V_{20}}{20}$$

$$V_{n+1} = V_n - x$$

$$P_{n+1} = P_n \times \frac{V_n}{V_{n+1}}^{K}$$

De esta forma obtenemos en el estado 20 las variables termodinámicas de presión y volumen al final de la compresión.

4.2.2. Aportación de calor

Una vez realizada la compresión, se desarrolla la aportación de calor a presión constante.

El rendimiento volumétrico depende del régimen de giro del motor [3][3]. Se ha diseccionado la curva de la gráfica de la figura 4.6 en 18 partes para poder introducirla al Matlab.

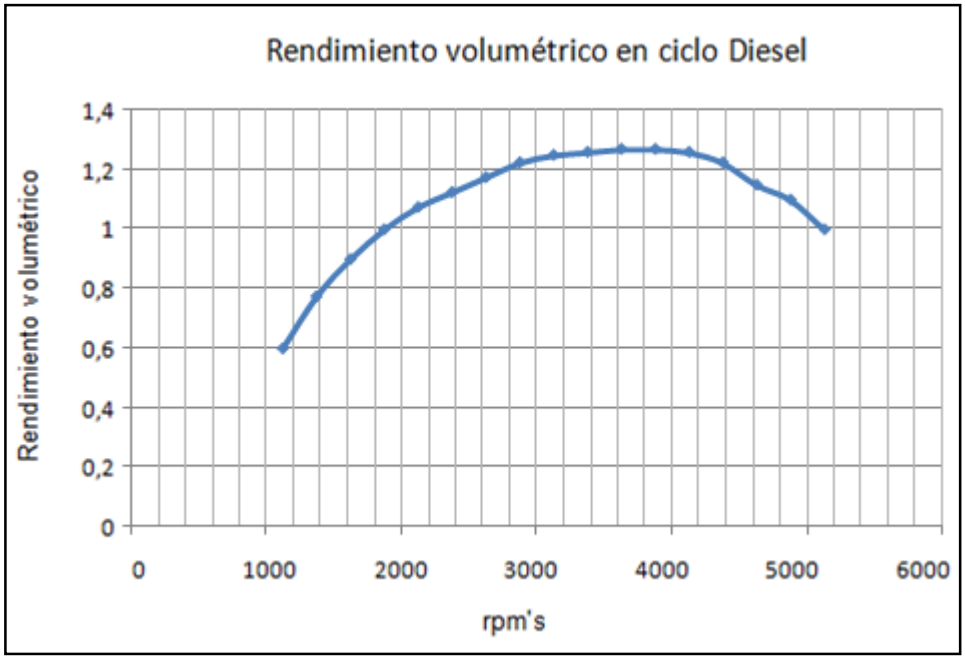


Figura 4.6: Rendimiento volumétrico ciclo Diesel

A diferencia del ciclo Otto, el ciclo Diesel se ha supuesto sobrealimentado, por lo tanto es normal que el rendimiento volumétrico supere el 100% en algunos casos ya que la presión de entrada suele ser mayor a la atmosférica.

El rendimiento de la combustión en el ciclo Diesel depende de varios factores [6]:

-Las forma de interacción entre el dardo con las paredes de la cámara de combustión.

-La temperatura al final de la compresión.

-El retardo de inyección.

Se distinguen tres tipos de interacción entre el dardo con las paredes de la cámara de combustión (A, B y C). Se pueden observar en la figura 4.7.

Para una buena combustión, es mejor que el contacto entre el jet y la pared sea inclinada y contorneada. En los tres tipos de variantes se han considerado que la expansión del chorro es de 15º y la pendiente de las paredes es de 20º respecto la vertical.

La variante A se trata de una pared llana sin ningún tipo de curvatura. La variante B es una pared recta pero con un poco de curvatura al final del tramo. La variante C se trata de una pared en la cual hay más tramos curvados que rectos.

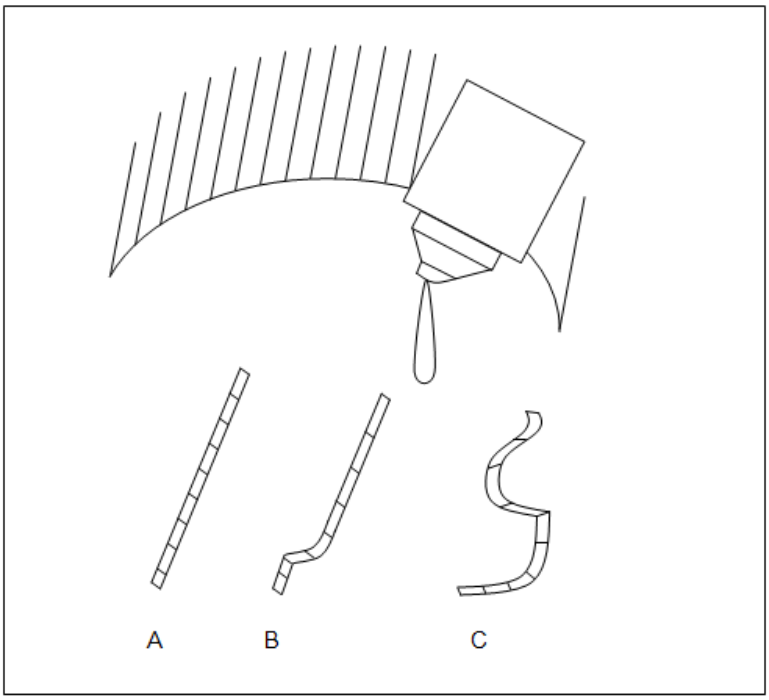


Figura 4.7: Formas de interacción del dardo con las paredes de la cámara de combustión

Se puede observar en la figura 4.8 el rendimiento de la combustión en función del retardo de inyección.

El retardo de la inyección depende también de la temperatura al final de la compresión como se puede observar en la figura 4.9.

El rendimiento de la combustión depende del retardo de inyección, y a su vez este depende de la temperatura al final de la compresión. Se puede relacionar el rendimiento de la combustión en función de la temperatura al final de la compresión tal y como se observa en la figura 4.10.

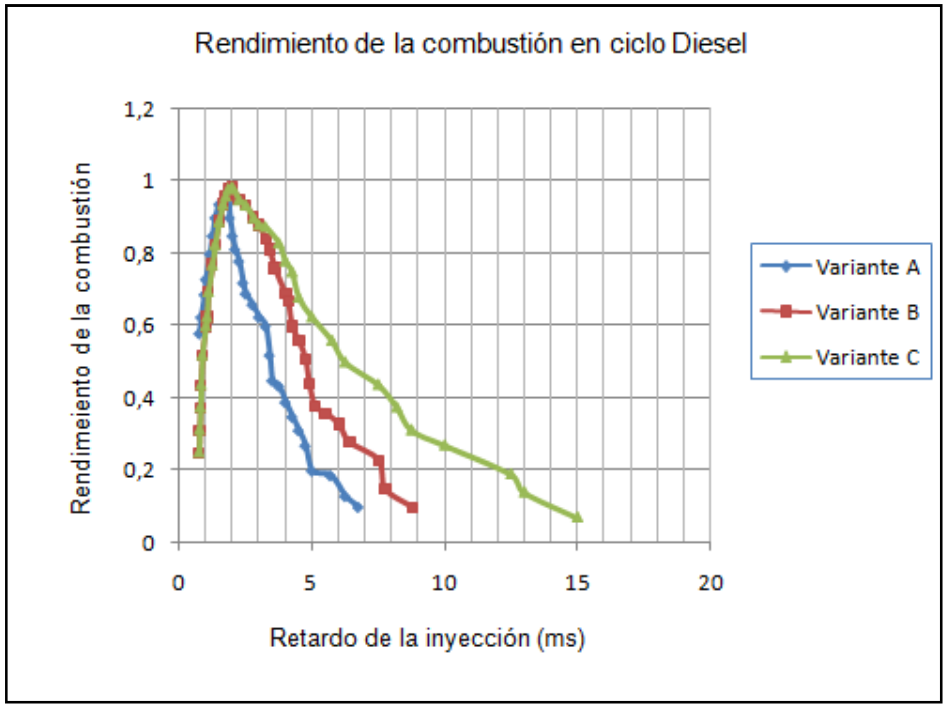


Figura 4.8: Rendimiento de la combustión en función del retardo de la inyección

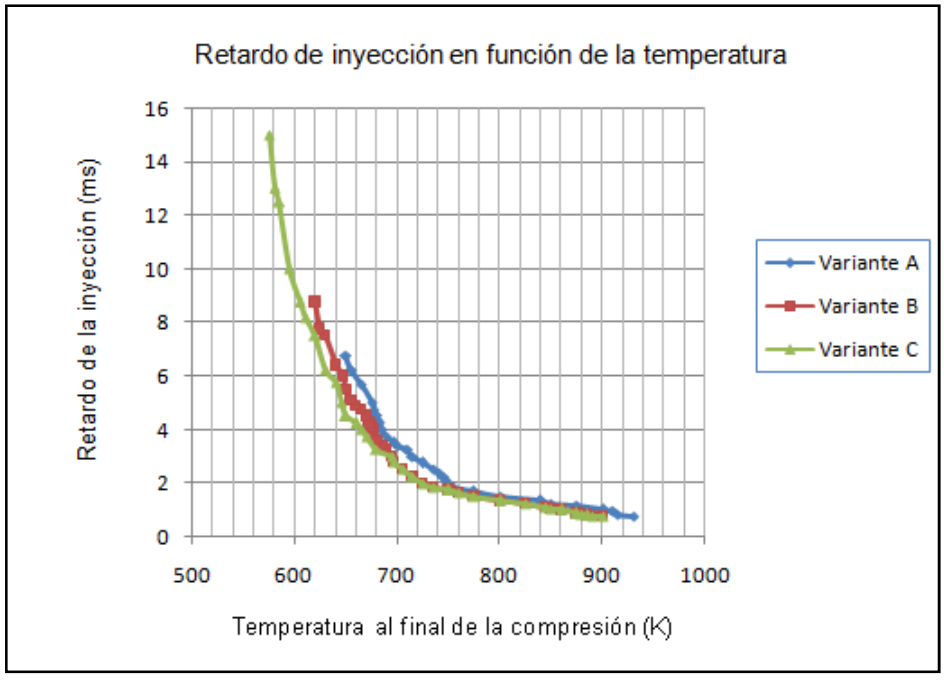


Figura 4.9: Retardo de la inyección en función de la temperatura

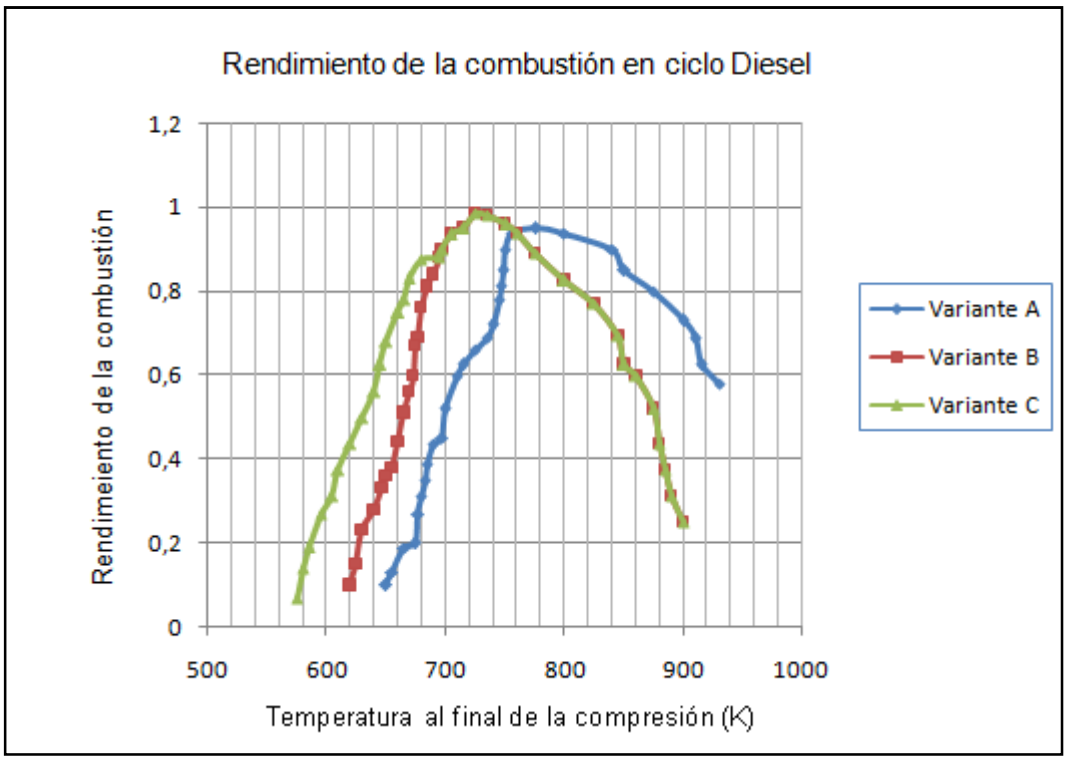


Figura 4.10: Rendimiento de la combustión en función de la temperatura

Se introducen al Matlab los valores de la figura 4.7 y de esta forma se procede a la aportación de calor primero utilizando la ecuación de los gases ideales:

$$n = \frac{P_{diesel} \times V_1 \times \eta_{vDiesel}}{R \times T1}$$

$$m\_aire = m\_molar_{aire} \times n$$

$$m\_diesel = \frac{m\_aire}{AFD}$$

Una vez tenemos la masa de combustible, podemos obtener el calor aportado:

$$Q = m\_diesel \times PCI_{diesel} \times \eta_{cDiesel}$$

Una vez se tiene el calor aportado, se puede encontrar la temperatura al final de la aportación del calor (denominado estado 30).

$$\Delta T = \frac{Q}{(m\_aire + m\_diesel) \times c_p}$$

$$T_{20}=T_1\times\left(\frac{V_1}{V_{20}}\right)^{k-1}$$

$$T_{30}=T_{20}+\Delta T$$

La aportación de calor es a presión constante, por lo tanto, la presión antes y después de la aportación de calor es la misma.

El volumen al final de la aportación de calor se obtiene mediante la aplicación de la ecuación de los gases ideales:

$$V_{30}=\frac{\left(n+\frac{m\_diesel}{masa\_molar_{diesel}}\right)\times R\times T_{30}}{p_{30}}$$

4.2.3. Expansión

Al tratarse de un ciclo Diesel, se asume una K (constante de expansión politrópica) entre los valores de 1,30 y 1,38 [7].

Se dispone a realizar la curva politrópica que corresponde a la expansión. Para la realización de esta curva se decide una resolución de 20 puntos igual que en el ciclo Otto.

Partiendo del estado final de la aportación de calor se define la constante de expansión:

$$Cte_{\text{expansión}} = P_{30}\times {V_{30}}^{K}$$

Una vez sabido el volumen al final de la expansión (V_1) se puede empezar a realizar la politrópica partiendo del volumen inicial (V_{30}) y llegando hasta el volumen al final de la compresión (V_1) en 20 pasos, así obteniendo en cada paso la presión y el volumen correspondiente con las siguiente fórmulas:

$$x=\frac{V_1\text{-}V_{30}}{20}$$

$$V_{n+1}=V_n+x$$

$$P_{n+1}=P_n\times\frac{{V_n}^{K}}{V_{n+1}}$$

El ciclo termodinámico se cierra pasando del final de la expansión al inicio de la compresión. Se supone que el volumen al final de la expansión y al principio de la compresión es el mismo.

4.3. Cálculo de variables de interés

En Matlab se han introducido una serie de variables de conveniente estudio para obtener valores cuantitativos de nuestros ciclos, tanto Otto como Diesel. Son las siguientes:

-Cálculo del trabajo del ciclo (J):

$$W_{compresión} = \int_{V_1}^{V_{20}} pdV$$

$$W_{expansión} = \int_{V_{30}}^{V_{50}} pdV$$

$$WJ = (W_{expansión} - \|W_{compresión}\|) \times 101.32$$

El cambio de unidades de [atm·l] a [J] lleva el factor 101.32.

-Cálculo de la potencia por ciclo (W):

$$P = \left(\frac{W_{expansión} \times 101.32 \times w_{giro} \times 2 \times \pi}{60}\right) - \| \left(\frac{W_{compresión} \times 101.32 \times w_{giro} \times 2 \times \pi}{60}\right) \|$$

-Cálculo del par (J) (mismo concepto que trabajo del ciclo):

$$\Gamma = \frac{P}{\frac{w_{giro} \times 2 \times \pi}{60}}$$

-Cálculo del consumo específico (g/KWh):

$$ge = \frac{m\_combustible}{\frac{P}{1000}} x \frac{60}{2}$$

-Cálculo del rendimiento termodinámico:

$$\eta_t = \frac{WJ}{Q}$$

-Cálculo de la presión media específica (atm):

$$pme = \frac{WJ}{(V_1 - V_{20}) \times 101.32}$$

Cabe comentar el hecho de que a algunas variables no se les ha asignado unidades del SI ya que cualitativamente a simple vista no se puede observar su magnitud. Se ha preferido realizarlo de esta forma para ver mejor la envergadura de estas. En los casos numéricos que se ven más adelante, las variables que están en unidades que no son las del SI, al lado se pone su valor en unidades del SI.

5. Simulación y optimización

Para introducir todas las modelizaciones correspondientes comentadas en el apartado 4 se ha utilizado el programa Matlab. Con este programa se ha creado una interface a través de un subprograma que tiene el Matlab que crea entornos gráficos llamados “Guide” para facilitar al usuario la introducción de variables y el estudio de los resultados. A continuación se detalla un apartado explicando el contenido y funcionamiento del Guide.

5.1. Funcionamiento del Guide

Guide es una aplicación del Matlab que permite crear entornos gráficos interactivos. El objetivo de la creación de un Guide es facilitar al usuario la introducción de las variables iniciales careciendo de conocimientos previos de Matlab, así pudiendo observar los resultados y cambiar las variables inciales de una forma fácil e interactiva.

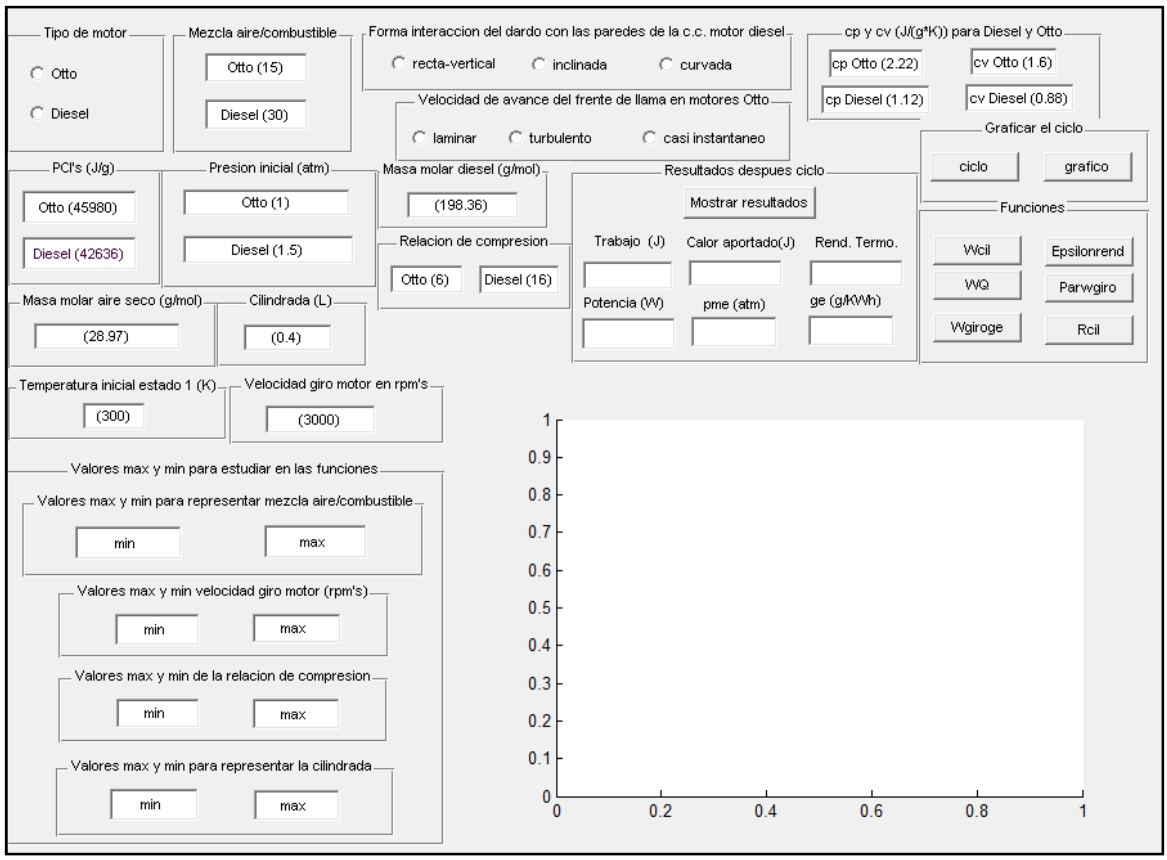


Figura 5.1: Visualización del Guide

El aspecto que presenta el Guide creado es el de la figura 5.1, dónde seguidamente se verá mejor cada variable que se ha de introducir así como su funcionamiento.

5.1.1. Variables de inicialización del Guide

Las variables que se han de introducir para realizar el estudio son las siguientes:

-Tipo de motor:

Se diferencia entre motor Otto o motor Diesel. Estas dos variables son variables booleanas. Cuando se activa Otto, a una variable booleana se le asigna el valor 1 y cuando se desactiva se le asigna un 0. El caso del Diesel es exactamente igual que el de Otto.

Figura 5.2: Motores

Cuando se empiecen a introducir variables, esta debe de ser la primera de ellas.

La figura 5.2 muestra cómo se visualiza esta variable en el Guide.

Variables en la modelización: “otto” y “diesel”.

-Poder calorífico inferior (PCI):

Se define el PCI (Poder Calorífico Inferior) como la calor generada por la combustión completa de una unidad de masa de dicho combustible suponiendo que todo el agua del combustible o generada por la combustión se encuentra como vapor en los productos de la combustión. Desde el punto de vista de la combustión, es el que da una idea más real de este proceso.

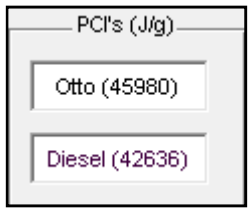


Figura 5.3: PCI’s

La figura 5.3 muestra los PCI’s. Se ha puesto entre paréntesis los valores orientativos (en unidades de J/g) en los que están los PCI’s en el caso del ciclo Otto y Diesel. En el caso de estudiar el ciclo Otto, solamente haría falta rellenar la casilla correspondiente al ciclo Otto e igual para el ciclo Diesel.

Variables en la modelización: “$PCI_{gasolina}$” y “PCI_{diesel}”.

-Masa molar aire seco:

Este parámetro es necesario para la modelización y simulación del ciclo comentado en el apartado 4.1.2 y 4.2.2.

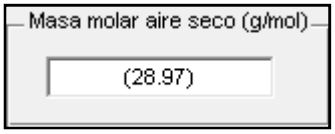


Fig.5.4: Masa molar aire

La figura 5.4 muestra este parámetro y el valor orientativo recomendado para la simulación entre paréntesis.

Variable en la modelización: ” m_molar_{aire}”.

-Temperatura inicial:

Esta variable representa la temperatura al inicio del ciclo termodinámico, ya sea ciclo Otto o ciclo Diesel.

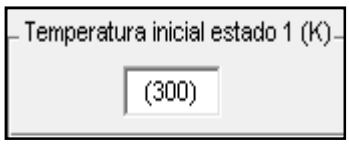


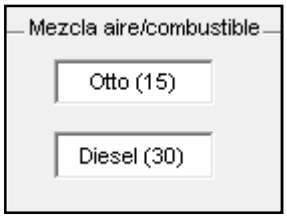
Fig.5.5:Temperatura inicial

La figura 5.5 muestra este parámetro y su valor orientativo entre paréntesis ya siendo una aproximación a la temperatura ambiente.

Variable en la modelización: "T_1".

-Relación aire / combustible:

Este parámetro es una relación de aire / combustible de la mezcla. Se diferencia caso Otto o caso Diesel ya que su valor difiere considerablemente.



En la figura 5.6 se pueden observar estos dos parámetros y su valor orientativo correspondiente en cada caso.

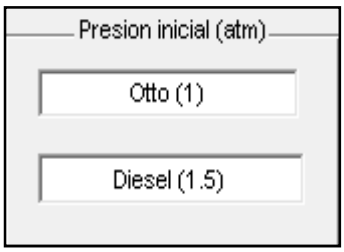
Variables en la modelización: "AFR" y "AFD".

Fig.5.6: Aire/combust.

.

-Presión inicial:

Esta variable representa la presión al inicio del ciclo. Se diferencia entre ciclo Otto o ciclo Diesel ya que en el apartado 4 se comentó que el ciclo Diesel es un ciclo sobrealimentado, por lo tanto, la presión a su estado inicial es superior a la atmosférica.



La figura 5.7 muestra los valores orientativos de la presión en el estado inicial.

Fig.5.7: Presión inicial

Las unidades de esta variable son atmósferas y no las del sistema internacional ya que de esta forma se da una visión más representativa y cualitativa de la presión en cada momento del ciclo. Se ha supuesto como orientativo el valor de 1.5 atm para el caso Diesel basándose en la presión del compresor de los vehículos actuales.

La presión introducida es la presión absoluta (presión relativa + presión atmosférica).

Variables en la modelización: "p_{otto}" y "p_{diesel}".

-Cilindrada (L):

El recorrido que efectúa el embolo entre el PMS y el PMI se denomina carrera, que multiplicada por la superfície del pistón, determina la cilindrada.

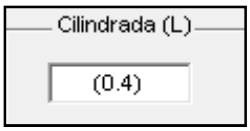


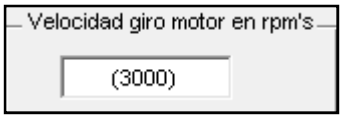
Fig.5.8: Cilindrada

Este volumen es el de un cilindro y las unidades para la introducción de esta variable son litros tal y como se muestra en la figura 5.8.

Variable en la modelización: "V_1"

-Velocidad de giro del motor (rmp's):

Esta variable hace referéncia a la velocidad de giro del cigüeñal. La figura 5.9 muestra esta variable y entre paréntesis se muestra un valor orientativo de la velocidad de giro de este.



Variable en la modelización: "w_{giro}"

Fig.5.9: Vel. de giro del motor

-Forma de interacción de del dardo con las paredes de la c.c. en el ciclo Diesel:

Tal y como se ha comentado en el apartado 4.2.2, el rendimento de la combustión en el ciclo Diesel depende de varios factores. Uno de ellos es la forma de interacción del dardo con las paredes de la c.c..

Como muestra la figura 5.10, hay tres tipos de interacción del dardo con las paredes de la c.c.. Cada una hace referencia a una variable booleana de tal manera que si se aprieta encima de una de ellas, esta variable se activa, si se desmarca y se marca otra, la desmarcada se desactiva y la marcada se activa.

Si el ciclo a estudiar es un ciclo Otto, no hace falta apretar en ninguna de las casillas ya que no afectará para nada en su estudio.

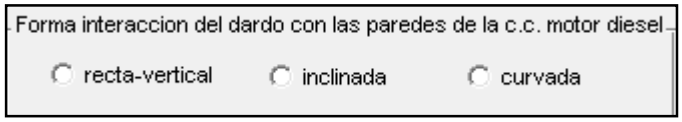


Figura 5.10: Interacción dardo-paredes ciclo Diesel

-Velocidad de avance del frente de llama en motores Otto:

Tal y como se ha comentado en el apartado 4.1.2., el rendimiento de la combustión en el ciclo Otto depende de diferentes parámetros, uno de ellos es la velocidad del frente de llama.

La figura 5.11 muestra las diferentes opciones de avance de frente de llama. Cada opción va referenciada a una variable booleana, que se activa si esta opción está activada, se desactiva si se desactiva esta variable y se activa otra.

Si el ciclo a estudiar es un ciclo Diesel, no hace falta apretar en ninguna de las casillas ya que no afectará para nada en su estudio.

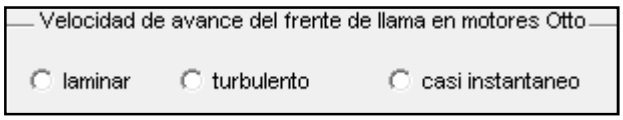


Figura 5.11: Velocidad frente llama ciclo Otto

-Masa molar diesel (g/mol):

Esta casilla hace referéncia a la masa molar del combustible diesel cuando se explicó la modelización del ciclo Diesel. El valor entre paréntesis que muestra la figura 5.12 es un valor orientativo ya que la composición del combustible puede variar sustancialmente.

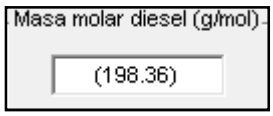


Fig.5.12: Masa molar

Cabe comentar que si lo que se está estudiando es un ciclo Otto, esta variable se puede dejar en blanco (es decir, sin introducir ningún valor).

Variable en la modelización: "masa_molar$_{diesel}$ ".

-Relación de compresión:

Relación entre el volumen máximo del cilindro (cilindro en punto muerto inferior) y el volumen mínimo (cilindro en el punto muerto superior). Para ciclos Otto este valor es más bajo que en el caso de ciclos Diesel tal y como se puede observar en la figura 5.13 los valores orientativos entre paréntesis.

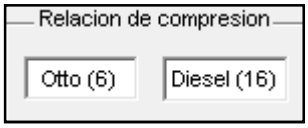


Fig.5.13:Rel.compresión

Variables en la modelización: "ε_{Otto}" y "ε_{diesel} ".

-Cp y Cv para ciclo Otto y Diesel:

Estas variables son muy importantes en el estudio de nuestro motor ya que una mínima variación de una de ellas puede hacer que los resultados se alteren notablemente, por eso la elección de estas tiene que ser muy concreta. [8]

Los valores que se muestran en la figura 5.14 son orientativos como todos los que se muestran en paréntesis de las otras variables, pero se recomienda que se pongan justamente estos ya que son valores tabulados de una gran importancia.

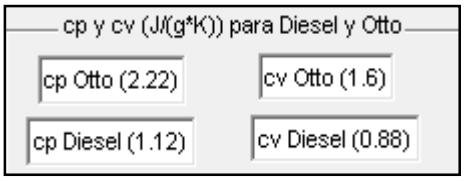


Figura 5.14: Cp y Cv Otto y Diesel

Variables en la modelización: "c_p" y "c_v".

-Variables máximas y mínimas para estudiar las funciones:

Para el estudio de las funciones que se explicarán más adelante, es necesario introducir unos valores máximos y mínimos que comprenderán algunas variables de estudio.

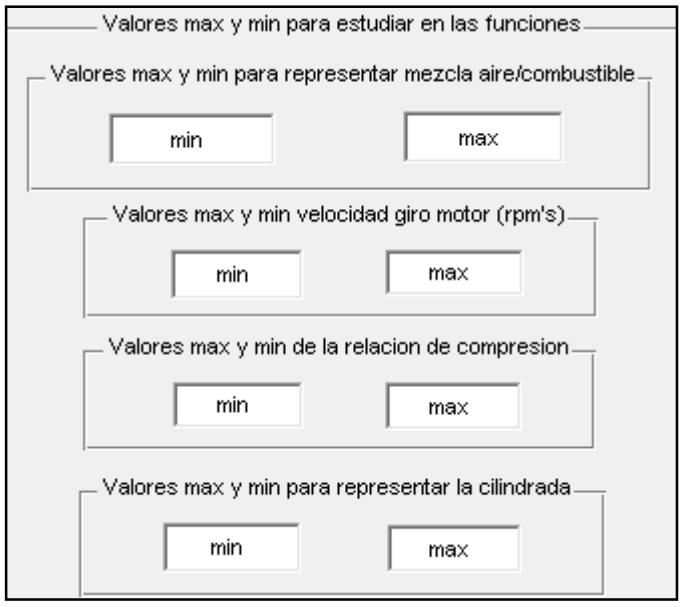


Figura 5.15: Valores máximos y mínimos de variables

Tal y como muestra la figura 5.15, las variables que se tienen que introducir estos valores máximos y mínimos son: la relación aire/combustible, la velocidad de giro del motor, la relación de compresión y la cilindrada.

5.1.2. Funciones de estudio y resultados obtenidos

En este apartado se explica el estudio que se hace a partir de las variables iniciales y la manera de mostrar los resultados obtenidos.

Una vez introducidas todas las variables correspondientes se produce al estudio de estas. Primero de todo se clica en el botón "ciclo" (como se puede observar en la figura 5.16). Esta función lo que realiza es un ciclo termodinámico completo. Lo realiza internamente sin mostrar ningún resultado por pantalla, todas las variables creadas y modificadas quedan internamente guardadas a la espera que alguna función las muestre (se verá más adelante).

El botón que pone "grafico" que está al lado del botón "ciclo" (figura 5.16) realiza la representación gráfica del ciclo termodinámico realizado anteriormente por la función ciclo. Así representando de color rojo el ciclo termodinámico y de color azul la presión media efectiva.

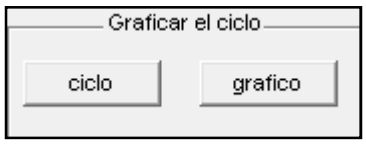


Fig.5.16: Funciones

Apretando el botón "Mostrar resultados" de la figura 5.17 muestra por pantalla las siguientes variables de salida:

-Trabajo obtenido (WJ) [J].

-Calor aportado (Q) [J].

-Rendimiento termodinámico (η_t).

-Potencia obtenida (P) [W].

-Presión media efectiva (pme) [atm].

-Consumo específico (ge) [$g\cdot(KW\cdot h)^{-1}$].

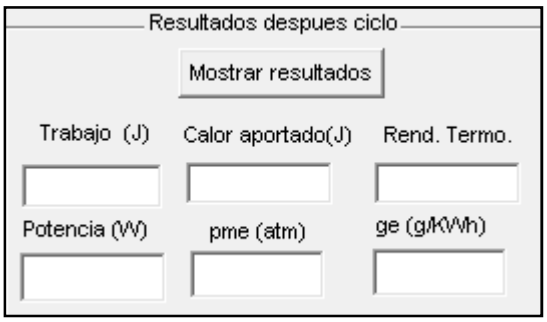


Figura 5.17: Resultados

Cabe comentar que las unidades de algunas variables no son del sistema internacional porque cualitativamente son muy significativas.

Si se quiere cambiar de valor cualquier variable inicial y luego ver los resultados, primero se cambia el valor, luego se deberá ejecutar la función “ciclo” y luego “mostrar resultados” para ver los cambios.

A continuación se presentan las funciones de estudio tal y como se puede observar en la figura 5.18.

Fig.5.18: Funciones

-Función Wcil:

Esta función estudia cómo afecta la variación del volumen de la cilindrada al trabajo obtenido por ciclo. Se ha creado una función interna en el Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de cilindrada se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de cilindrada. Hay que tener en cuenta que se estudia solo cilindro.

Entre el valor máximo y el valor mínimo de la cilindrada se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de volumen de la cilindrada a estudiar, para cada uno de ellos se ejecuta la función “ciclo” manteniendo constante todas las demás variables y se le asigna una variable interna al trabajo obtenido en cada ciclo.

Una vez tenemos todas las variables de trabajo en función de todas las variables de volumen, se representa gráficamente los resultados. Cuando se aprieta el botón “Wcil” se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la cilindrada, aumente el trabajo obtenido por ciclo.

-Función “Espsiolonrend”:

Esta función estudia cómo afecta la variación de la relación de compresión al rendimeinto termodiámico. Se ha creado una función interna en el Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de relación de compresión se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de relación de compresión. Hay que tener en cuenta si lo que se está estudiando es un ciclo

Otto o ciclo Diesel, ya que el rango de valores de relación de compresión son diferentes. Para ciclo Otto, un rango aceptable de relaciones de compresión es de 6 a 10, mientras que de un motor diesel es de 14 a 19.

Entre el valor máximo y el valor mínimo de la relación de compresión se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de volumen de la relación de compresión a estudiar, para cada uno de ellos se ejecuta la función “ciclo” manteniendo constante todas las demás variables y se le asigna una variable interna al rendimiento termodinámico obtenido en cada ciclo.

Una vez tenemos todas las variables de rendimiento termodinámico en función de todas las variables de relación de compresión, se representa gráficamente los resultados. Cuando se aprieta el botón “Epsilonrend” se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la relación de compresión, aumente el rendimiento termodinámico obtenido.

-Función “WQ”:

Esta función estudia cómo afecta la variación de la relación aire/combustible (AFR o AFD) al trabajo obtenido por ciclo. Se ha creado una función interna del Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de relación aire combustible se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de relación aire/combusbitle. Hay que tener en cuenta si lo que se está estudiando es un ciclo Otto o ciclo Diesel, ya que el rango de valores de relación aire/combustible es diferente para cada caso. Para ciclo Otto, un rango aceptable de relaciones aire/combustible es de 13 a 17 (AFR), mientras que de un motor diesel es de 17 a 50 (AFD).

Entre el valor máximo y el valor mínimo de la relación aire/combustible se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de relación aire/combustible a estudiar, para cada uno de ellos se ejecuta la función “ciclo” manteniendo constante todas las demás variables y se le asigna una variable interna al trabajo obtenido en cada ciclo.

Una vez tenemos todas las variables de trabajo en función de todas las variables de relación aire/combustible, se representa gráficamente los resultados. Cuando se aprieta el botón

"WQ" se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la relación aire/combustible (disminuye el combustible aportado al ciclo), el trabajo obtenido es menor.

-Función "Parwgiro":

Esta función estudia cómo afecta la variación de la velocidad de giro del motor al par motor obtenido por ciclo. Se ha creado una función interna del Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de velocidad de giro del motor se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de velocidad de giro del motor. En este caso es indiferente si el ciclo es Otto o Diesel.

Entre el valor máximo y el valor mínimo de la velocidad de giro del motor se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de la velocidad de giro a estudiar, para cada uno de ellos se ejecuta la función "ciclo" manteniendo constante todas las demás variables y se le asigna una variable interna al par motor obtenido en cada ciclo.

Una vez tenemos todas las variables de par en función de todas las variables de velocidad y giro del motor, se representa gráficamente los resultados. Cuando se aprieta el botón "Parwgiro" se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la velocidad de giro del motor, el par motor aumenta hasta llegar a un punto máximo, y luego va disminuyendo. Esto se supone porque es la forma que tiene el rendimiento volumétrico ya que depende de la velocidad de giro tanto en ciclo Otto como en ciclo Diesel. También es lógico pensar que el ciclo Diesel dé más par que el ciclo Otto ya que está realimentado y hay casos en que el rendimiento volumétrico sobrepasa del 100%.

-Función “Wgiroge”:

Esta función estudia cómo afecta la variación de la velocidad de giro del motor al consumo específico obtenido por ciclo. Se ha creado una función interna del Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de velocidad de giro del motor se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de velocidad de giro del motor. En este caso es indiferente si el ciclo es Otto o Diesel.

Entre el valor máximo y el valor mínimo de la velocidad de giro del motor se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de la velocidad de giro a estudiar, para cada uno de ellos se ejecuta la función “ciclo” manteniendo constante todas las demás variables y se le asigna una variable interna al consumo específico obtenido en cada ciclo.

Una vez tenemos todas las variables de par en función de todas las variables de velocidad e giro del motor, se representa s gráficamente los resultados. Cuando se aprieta el botón “Wigorge” se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la velocidad de giro del motor, el consumo específico disminuya hasta un mínimo y luego vaya aumentado, es decir, que tenga una forma parabólica, esto es debido a un hecho parecido al que pasa en la función “Parwgiro”. Hay un momento en que la potencia es máxima, por lo tanto, cuando esto pasa, el consumo específico debería será mínimo.

-Función “Rcil”:

Esta función estudia cómo afecta la variación del volumen de la cilindrada al rendimiento termodinámico por ciclo. Se ha creado una función interna en el Matlab que lo realiza.

Esta función lo primero que pide es entre qué valores de cilindrada se quiere estudiar. Por eso antes se pide que se introduzcan un valor máximo y un valor mínimo de cilindrada. Hay que tener en cuenta que se estudia un cilindro.

Entre el valor máximo y el valor mínimo de la cilindrada se definen 20 partes intermedias para obtener una resolución aceptable de la gráfica. Una vez definidas las 20 partes, podemos obtener 21 puntos de volumen de la cilindrada a estudiar, para cada uno de ellos

se ejecuta la función “ciclo” manteniendo constante todas las demás variables y se le asigna una variable interna al rendimiento termodinámico obtenido en cada ciclo.

Una vez tenemos todas las variables de trabajo en función de todas las variables de volumen, se representa gráficamente los resultados. Cuando se aprieta el botón “Rcil” se ejecuta internamente todo esto y lo que vemos en pantalla es la representación gráfica comentada.

Los resultados lógicos esperados de ésta gráfica sin conocer previamente su forma es de que a medida que aumentamos la cilindrada, disminuya el rendimiento termodinámico, ya que cuanto más grande es el cilindro, más pérdidas tiene.

5.2. Ejemplo simulación y optimización Ciclo Otto

A continuación se presenta un ejemplo de estudio de un motor ciclo Otto. Para realizar este ejemplo se toman las variables iniciales de la tabla 5.1. y los valores máximos y mínimos de la talba 5.2.

Variable	Valor	Unidades	Valor SI	Unidades SI
$PCI_{gasolina}$	-	-	45980	$J \cdot g^{-1}$
m_molar_{aire}	-	-	28.97	$g \cdot mol^{-1}$
T_1	-	-	300	K
AFR	-	-	14.00	-
p_{otto}	1	atm	1E+05	Pa
V_1	0,40	l	4E-04	m^3
w_{giro}	2500	rpm	261.80	$rad \cdot s^{-1}$
ε_{Otto}	6	-	-	-
c_p	2.22	$J \cdot (g \cdot K)^{-1}$	-	-
c_v	1.6	$J \cdot (g \cdot K)^{-1}$	-	-

Tabla 5.1: Valores de las variables ciclo Otto

Variables	Mínimo	Máximo	Unidades	Mínimo SI	Máximo SI	Unidades SI
AFR	13	17	-	-	-	-
w_{giro}	1000	6000	rpm	104.72	628.32	rad/s
ε_{Otto}	6	10	-	-	-	-
V_1	0.30	0.80	l	3E-04	8E-04	m^3

Tabla 5.2: Valores máximos y mínimos de variables ciclo Otto

Se ha procurado que los valores de las variables introducidas en el Guide en este ejemplo de ciclo Otto, sean valores razonables y comunes en un motor ciclo Otto.

La figura 5.19 muestra cómo quedaría el Guide después de la introducción de variables.

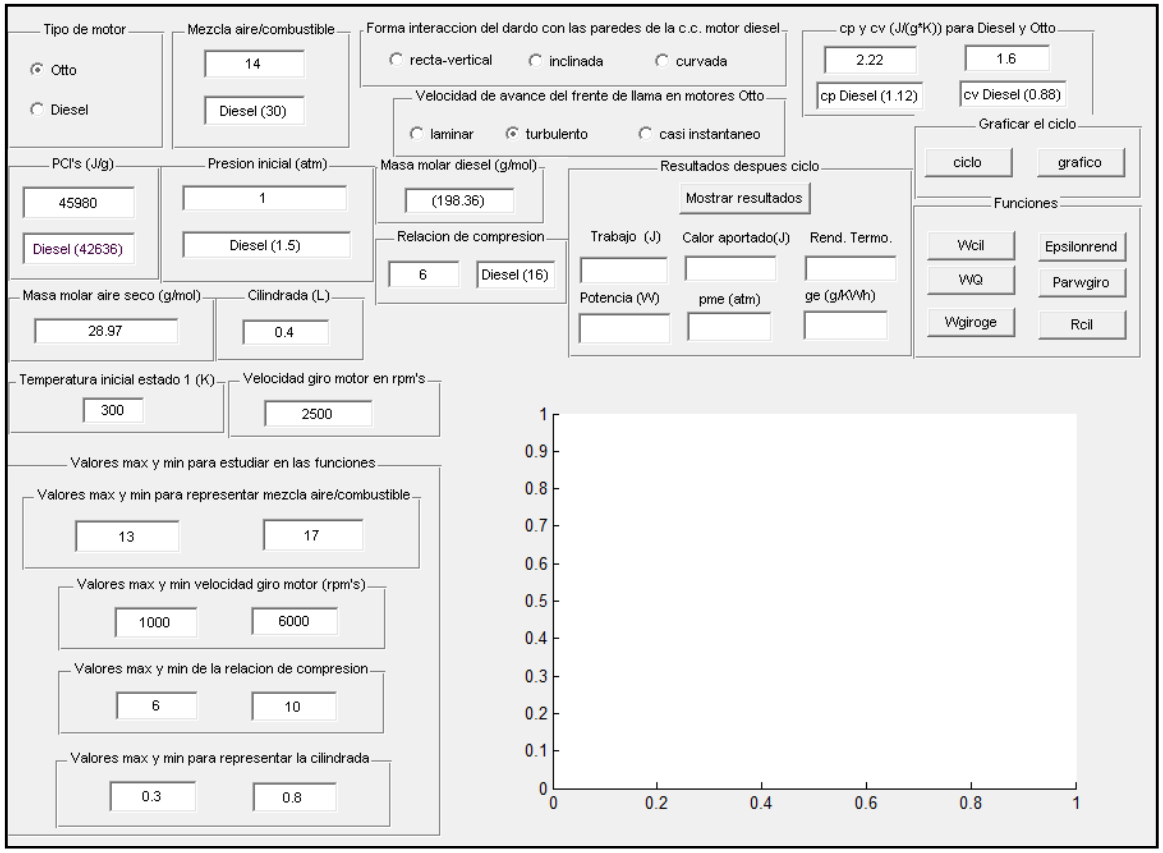


Figura 5.19: Introducción de variables en el Guide

Como podemos observar en la figura 5.19, los valores que hacen referencia al ciclo Diesel no se han tocado, se han dejado tal y como estaban.

Como velocidad de avance de frente de llama se ha asignado régimen turbulento.

A continuación se ejecuta la función ciclo apretando sobre el botón “ciclo”, después se presiona el botón “gráfico” para ver la representación termodinámica del ciclo y después se presiona el botón “mostrar resultados” para ver los resultados numéricos más destacados del ciclo.

La figura 5.20 muestra el gráfico obtenido del ciclo y la figura 5.21 muestra los resultados obtenidos.

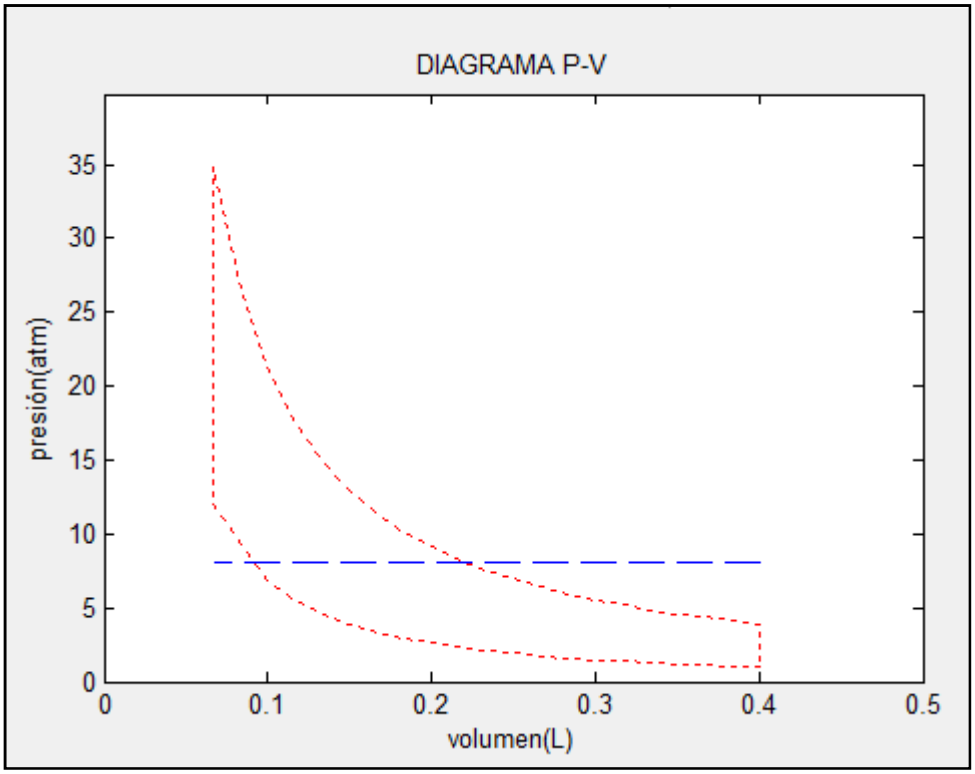


Figura 5.20: Gráfico del ciclo termodinámico

Los resultados obtenidos una vez presionado el botón “mostrar resultados” son los siguientes:

-Trabajo obtenido = 271.30 J

-Calor aportado = 1033.96 J

-Rendimiento termodinámico = 0.26

-Potencia obtenida = 71025.70 W

-Presión media efectiva = 8.03 atm

-Consumo específico = 26.98 $g \cdot (KW \cdot h)^{-1}$

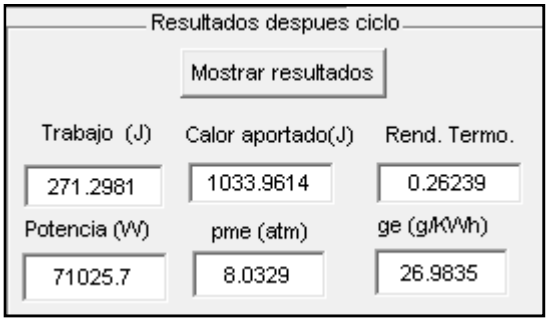


Figura 5.21: Resultados después ciclo

De esta forma se puede saber las características puntuales de un motor. A continuación se describe el mismo ejemplo pero estudiando las funciones que relacionan los valores máximos y mínimos.

-Función “Wcil”:

Se aprieta sobre el botón “Wcil” del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la cilindrada como (figura 5.22).

Figura 5.22: Variación del trabajo en función de la cilindrada.

A medida que se va aumentando la cilindrada el trabajo obtenido por ciclo aumenta. Es un resultado lógico y esperado ya que se tiene más volumen, se tiene más área para integrar y eso implica más trabajo.

Cabe comentar que los valores obtenidos de trabajo son bastante altos, siempre se tiene que el diagrama representado no es el indicado (real) y a pesar de intentar aproximar muchos parámetros hacia la realidad quedan algunos otros como por ejemplo las pérdidas mecánicas que no se han tenido en cuenta.

-Función “Epsilonrend”:

Se aprieta sobre el botón “Epsilonrend” del apartado de funciones y nos muestra gráficamente la variación del rendimiento termodinámico en función de la relación de compresión tal y como se puede observar en la figura 5.23.

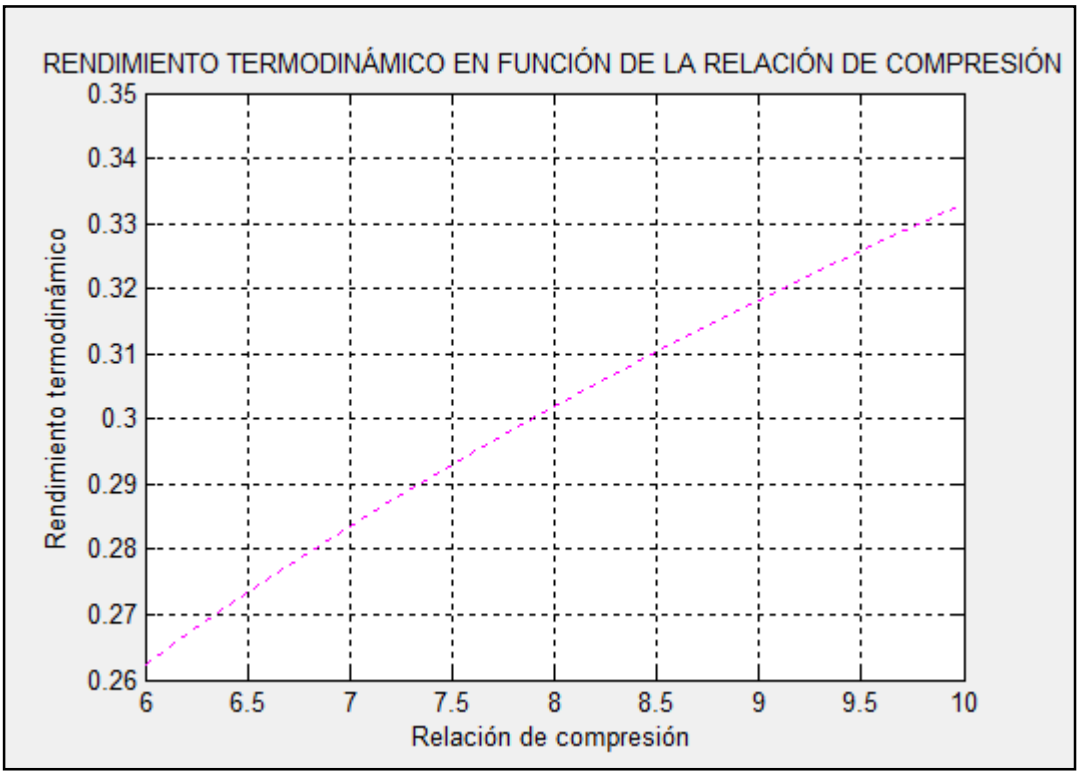


Figura 5.23: Rendimiento termodinámico en función de la relación de compresión

Como se puede observar en la figura 5.23, el rendimiento termodinámico aumenta si se aumenta la relación de compresión. Es un resultado lógico y coherente ya que se aumenta el área de la integral del diagrama termodinámico p-v (se expande más el volumen), por lo tanto, el trabajo obtenido se va incrementando y con éste, el rendimiento termodinámico.

-Función “WQ”:

Se aprieta sobre el botón “WQ” del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la relación de aire/combustible tal y como se puede observar en la figura 5.24.

A medida que se aumenta la relación aire/combustible, el trabajo obtenido por ciclo disminuye. Este hecho es normal, ya que aumentar la relación aire/combustible implica que el combustible que se aportando al ciclo disminuye, y con menos combustible, se obtiene menos trabajo.

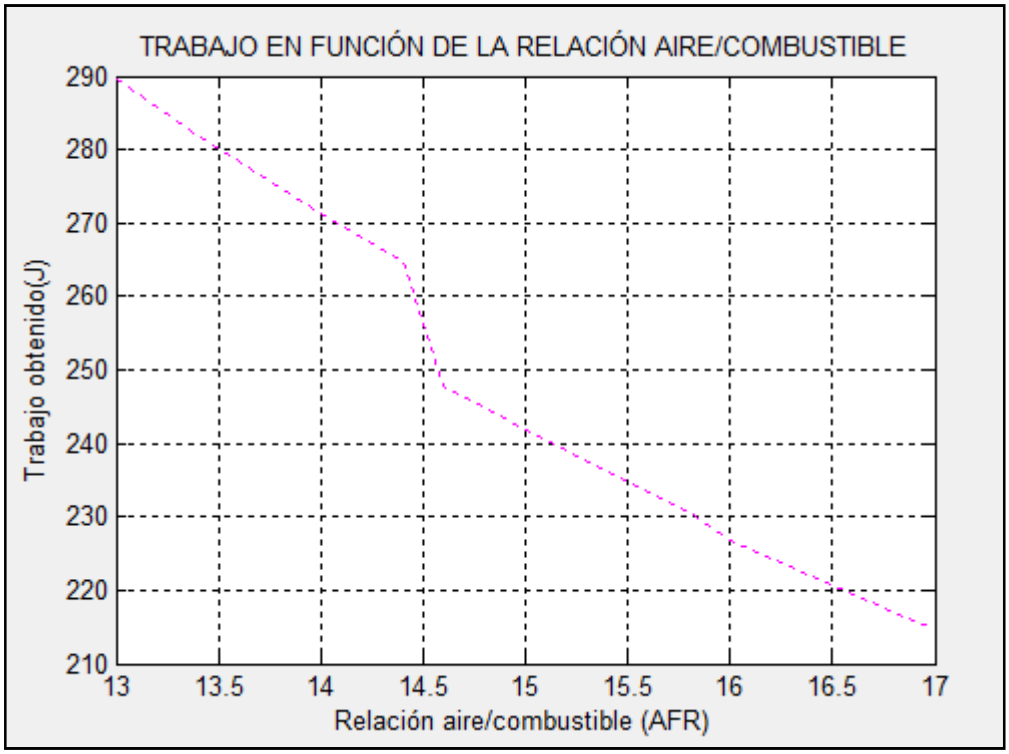


Figura 5.24: Variación del trabajo en función de la relación aire/combustible

Cabe comentar que en la figura 5.24 se puede observar un cambio de pendiente sobre el valor 14.4 de la relación aire/combustible. Después de hacer estudios con variables internas del Matlab, se ha llegado a la conclusión de que este cambio de pendiente lo provoca un cambio de la constante politrópica de expansión (K).

La constante K de expansión en el ciclo Otto depende de la temperatura al final de la aportación de calor. Si se varia la relación aire/combustible, el calor aportado al ciclo hace variar la temperatura al final de la aportación de calor y esto hace cambiar la K de expansión. Si se cambia la K de expansión, se modifica el área generada por la integral cuando se calcula el trabajo.

-Función “Parwgiro”:

Se aprieta sobre el botón “Parwgiro” del apartado de funciones y nos muestra gráficamente la variación del par motor en función de la velocidad de giro del motor tal y como se puede observar en la figura 5.25.

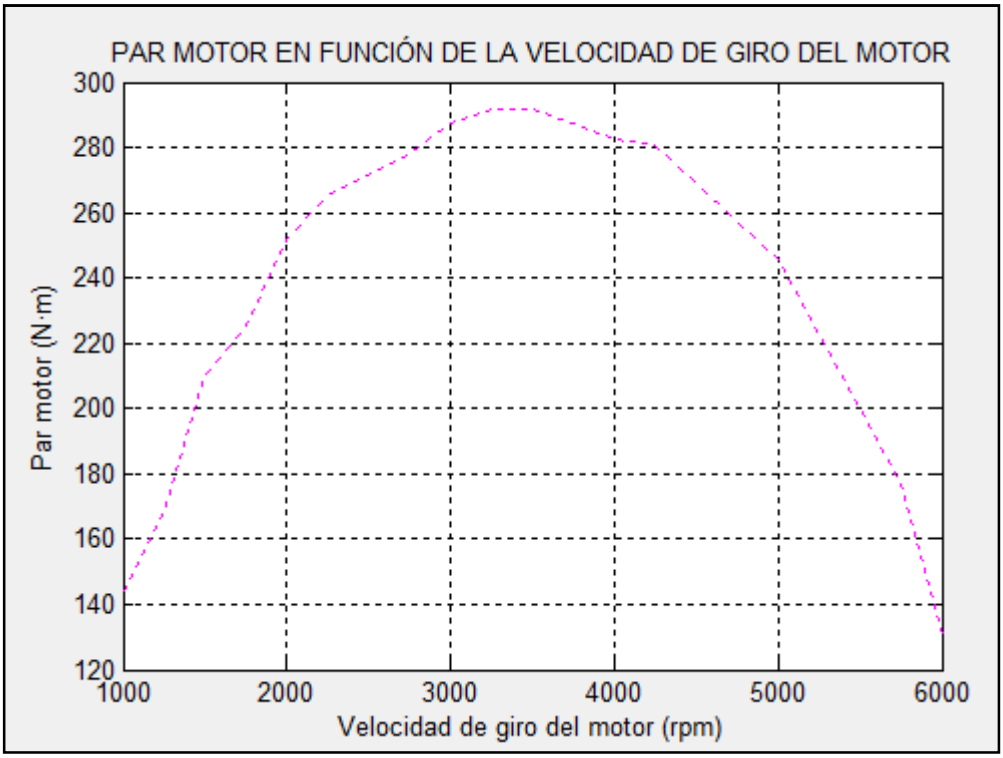


Figura 5.25: Par motor en función de la velocidad de giro del motor

A medida que se aumenta la velocidad de giro del motor, el par obtenido aumenta hasta alcanzar un máximo y seguidamente disminuye. Esto pasa sobre las 3500 rpm's, valor razonable y similar en automoción. Esto sucede gracias al rendimiento volumétrico, que depende de la velocidad de giro del motor y el valor más alto lo tiene sobre las 3500 rpm's.

-Función "Wgiroge":

Se aprieta sobre el botón "Wgiroge" del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la relación de aire/combustible como (figura 5.26).

A medida que se va aumentando la velocidad de giro del motor, el consumo específico disminuye hasta un mínimo y luego va aumentando. Esto es debido a un hecho parecido a lo que pasa con en la función "Parwgiro". Hay un momento en que la potencia es máxima, por lo tanto, cuando esto pasa, el consumo específico es mínimo.

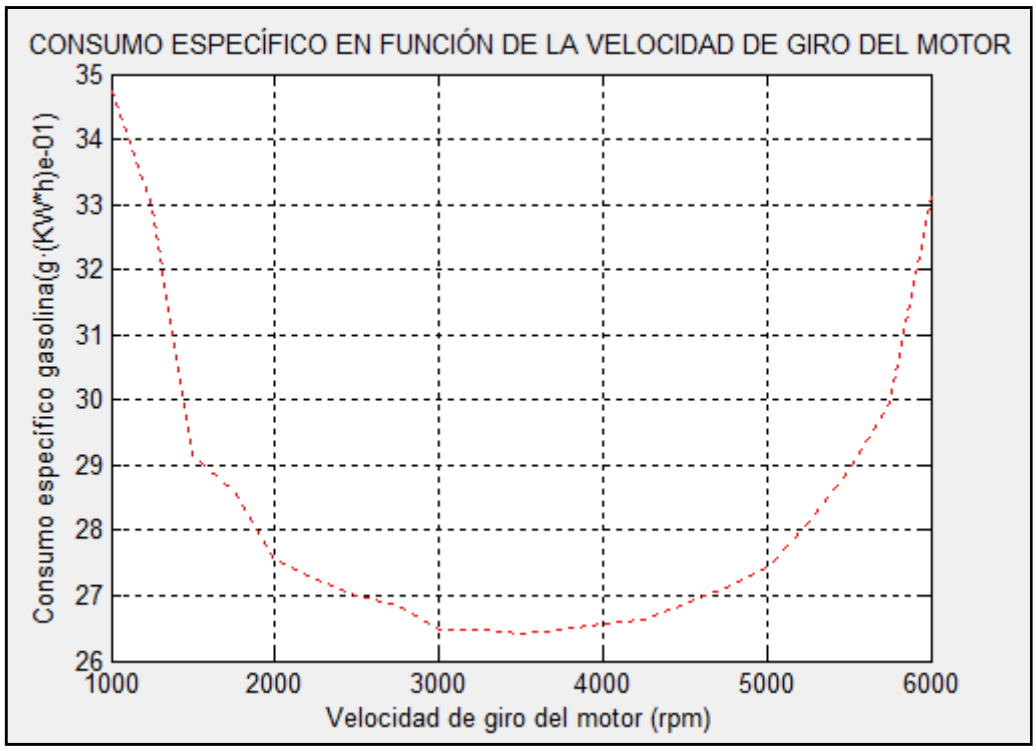


Figura 5.26: Consumo específico en función de la velocidad de giro del motor

-Función "Rcil":

Se aprieta sobre el botón "Rcil" del apartado de funciones y nos muestra gráficamente la variación del rendimiento termodinámico en función de la cilindrada (figura 5.27).

Cuando se aumenta la cilindrada, el rendimiento termodinámico disminuye, este hecho es lógico, ya que cuanto más grande es el cilindro, más pérdidas tiene.

Cabe comentar que los valores del rendimiento termodinámico son valores razonables y que se recuerda que la relación de compresión con la que estamos estudiando esta función es de 6. Si nos fijamos en la figura 5.23, los valores del rendimiento termodinámico asociados a una relación de compresión de 6 son de 0.26 y en la figura 5.27, los valores del rendimiento termodinámico están sobre 0.26.

Una vez visto todas las funciones se procede a estudiar la optimización del motor dependiendo del objetivo que se tenga.

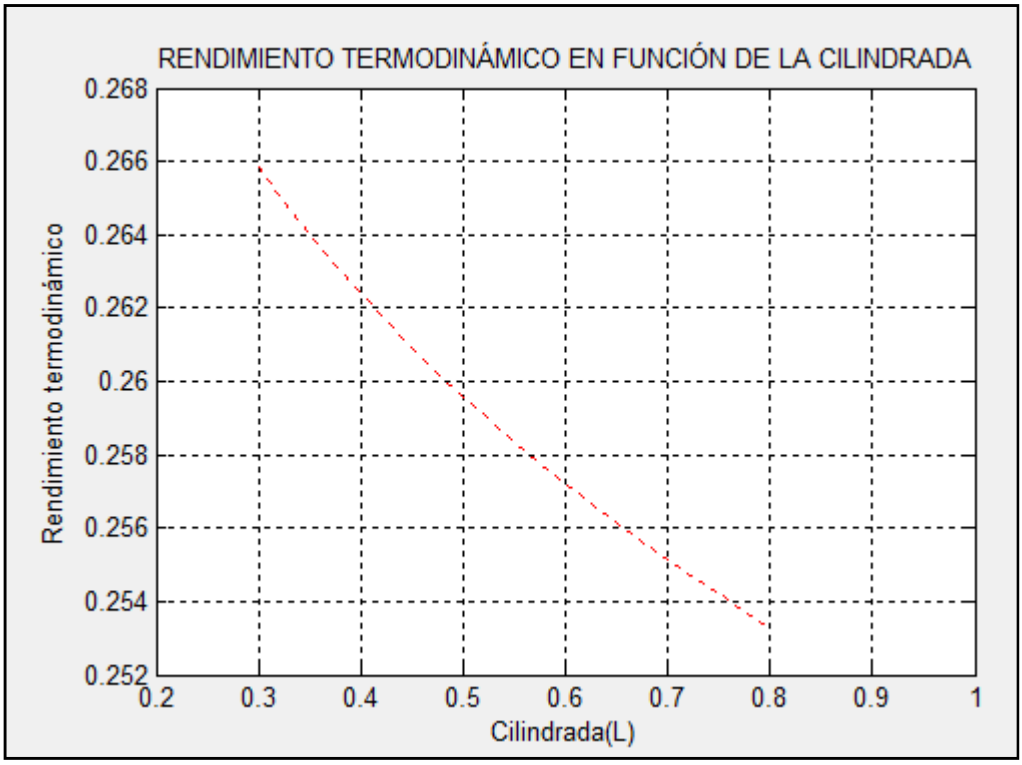


Figura 5.27: Rendimiento termodinámico en función de la cilindrada

Se propone optimizar el motor de tres maneras posibles que son las siguientes: obtención del máximo trabajo por ciclo, obtención del rendimiento termodinámico más elevado para aprovechar al máximo el calor aportado y obtención del mínimo consumo específico para gastar lo mínimo en combustible.

-Obtención del máximo trabajo por ciclo:

Para obtener el máximo trabajo por ciclo se observan las figuras 5.22, 5.24 y 5.25 en las cuales el trabajo (par) aparece. Si se quiere que éste sea máximo, la cilindrada tiene que ser lo más grande posible (figura 5.22), la relación aire/combustible debe de ser mínimo (figura 5.24) y la velocidad de giro del motor tiene que ser de 3500 rpm (figura 5.25).

El rango de valores en el que nos moveremos será el estipulado por las variables máximas y mínimas que se han definido anteriormente.

Partiendo de las variables iniciales del modelo, se cambian las siguientes variables por los siguientes valores:

-La cilindrada de 0.80 litros.

-Relación aire/combustible pasa a ser de 13.

-La velocidad de giro del motor de 3500 rpm.

Se realizan estos cambios en el modelo y se aprieta en la función “ciclo” y luego en “mostrar resultados” y se obtienen los resultados siguientes tal y como muestra la figura 5.28:

-Trabajo obtenido = 603.86 J

-Calor aportado = 2364.18 J

-Rendimiento termodinámico = 0.255

-Potencia obtenida = 221328.00 W

-Presión media efectiva = 8.94 atm

-Consumo específico = 27.49 $g\cdot(KW\cdot h)^{-1}$

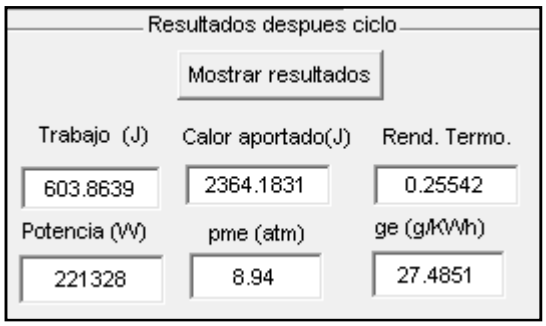


Figura 5.28: Resultados

El trabajo obtenido ha aumentado considerablemente en relación a los resultados obtenidos por el modelo inicial y sin alterar excesivamente los demás resultados, por lo tanto, se da el modelo de obtención de máximo trabajo por bueno.

-Obtención del rendimiento termodinámico más elevado:

Para la obtención del rendimiento termodinámico más elevado se observan las figuras 5.23 y 5.27 en las cuales aparece el rendimiento termodinámico. Si se desea que el rendimiento termodinámico sea máximo, la relación de compresión tiene que ser lo más grande posible (figura 5.23) y la cilindrada lo más pequeña posible (figura 5.27) y la velocidad de giro el motor debe de ser de 3500 rpm, ya que de esta forma aumentamos el trabajo obtenido y aumentaremos también el rendimiento termodinámico.

Partiendo de las variables iniciales del modelo, se cambiarán las siguientes variables por los siguientes valores:

-La relación de compresión del motor de 10.

-La cilindrada de 0.30 litros.

-La velocidad de giro del motor de 3500 rpm.

Se realizan estos cambios en el modelo y se aprieta en la función “ciclo” y luego en “mostrar resultados” y se obtienen los resultados siguientes tal y como muestra la figura 5.29:

-Trabajo obtenido = 281.71 J

-Calor aportado = 823.24 J

-Rendimiento termodinámico = 0.34

-Potencia obtenida = 103250.00 W

-Presión media efectiva = 10.30 atm

-Consumo específico = 20.52 $g \cdot (KW \cdot h)^{-1}$

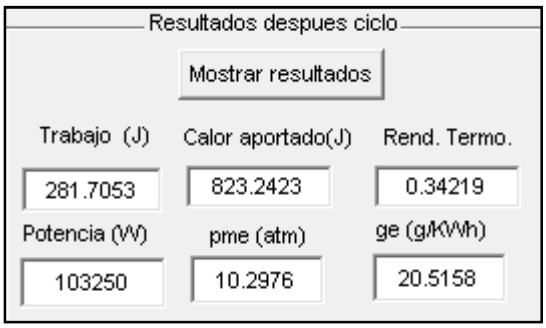


Figura 5.29: Resultados

El rendimiento termodinámico ha aumentado considerablemente respecto el modelo incial. Las otras variables no se han alterado excesivamente, por lo tanto, se da el modelo de obtención de máximo rendimiento termodinámico por bueno.

-Obtener el mínimo consumo específico para así gastar lo mínimo en combustible.

Para obtener el mínimo consumo específico debemos observar la función “Wgiroge” (figura 5.26). El mínimo consumo específico está en las 3500 rpm. En el modelo de obtención de máximo trabajo y el de máximo rendimiento termodinámico, se hacen a 3500 rpm, por lo tanto, no se estudia este caso y se engloba en los dos modelos anteriores ya que estos dos garantizan el mínimo consumo específico en cada caso.

5.3. Ejemplo simulación y optimización ciclo Diesel

A continuación se presenta un ejemplo de estudio de un motor ciclo Diesel. Para realizar este ejemplo se toman las variables iniciales de la tabla 5.3. y los valores máximos y mínimos de la tabla 5.4.

Se ha procurado que los valores de las variables introducidos en el Guide en este ejemplo de ciclo Otto, sean valores razonables y comunes en un motor ciclo Diesel.

Variable	Valor	Unidades	Valor SI	Unidades SI
PCI_{diesel}	-	-	42636	$J \cdot g^{-1}$
m_molar_{aire}	-	-	28.97	$g \cdot mol^{-1}$
T_1	-	-	300	K
AFD	-	-	40.00	-
p_{diesel}	1.50	atm	2E+05	Pa
V_1	0.40	l	4E-04	m^3
w_{giro}	2500	rpm	261.80	$rad \cdot s^{-1}$
ε_{diesel}	16.00	-	-	-
c_p	1.12	$J \cdot (g \cdot K)^{-1}$	-	-
c_v	0.88	$J \cdot (g \cdot K)^{-1}$	-	-

Tabla 5.1: Valores de las variables ciclo Diesel

Variables	Mínimo	Máximo	Unidades	Mínimo SI	Máximo SI	Unidades SI
AFD	20	50	-	-	-	-
w_{giro}	1500	5000	rpm	157.01	523.6	rad/s
ε_{diesel}	14	19	-	-	-	-
V_1	0.3	0.8	l	3E-04	8E-04	m^3

Tabla 5.2: Valores máximos y mínimos de variables ciclo Diesel

La figura 5.30 muestra cómo queda el Guide después de la introducción de variables. Como se puede observar, los valores que hacen referencia al ciclo Diesel no se han tocado, se han dejado tal y como estaban.

Como forma de interacción del dardo con las paredes de la c.c. se ha puesto la opción inclinada.

A continuación se ejecuta la función ciclo apretando sobre el botón “ciclo”, después se presiona el botón “gráfico” para ver la representación termodinámica del ciclo y después se presiona el botón “mostrar resultados” para ver los resultados numéricos más destacados del ciclo.

La figura 5.32 muestra el gráfico obtenido del ciclo y la figura 5.33 muestra los resultados obtenidos.

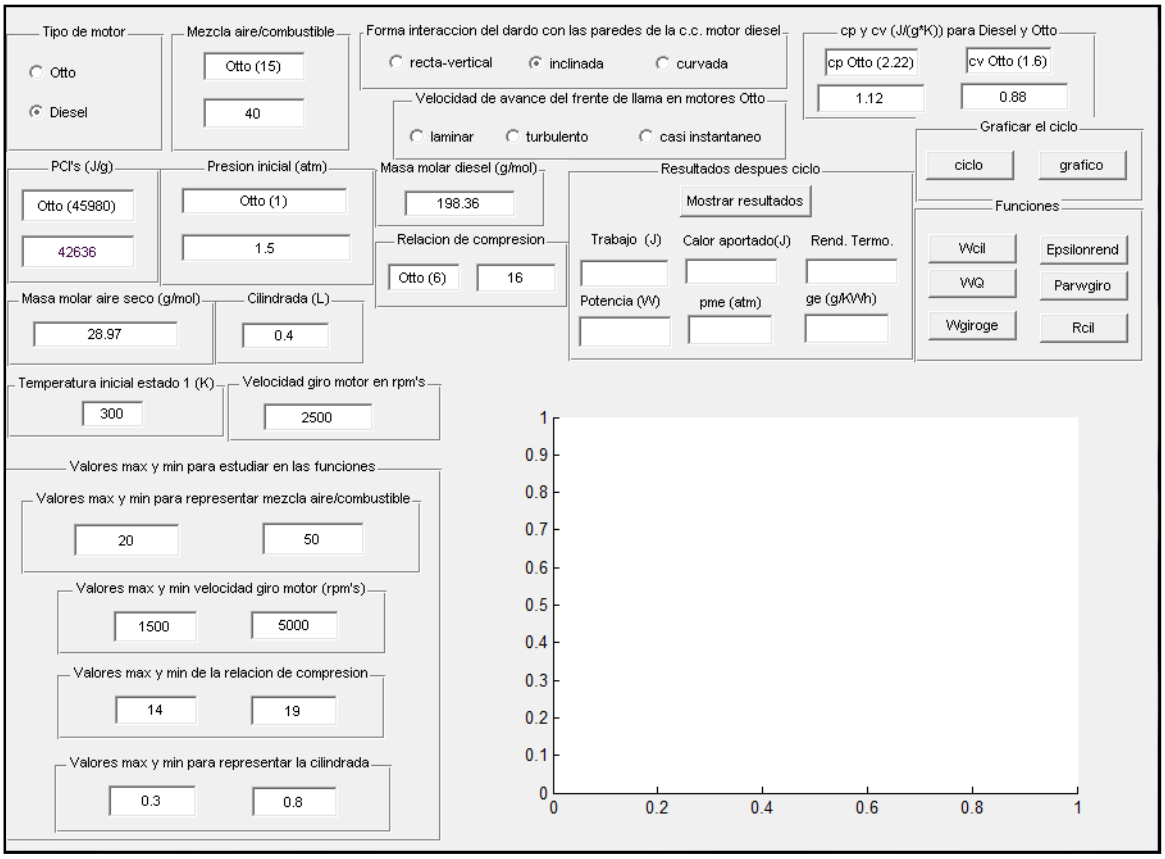


Figura 5.31: Introducción de variables en el Guide

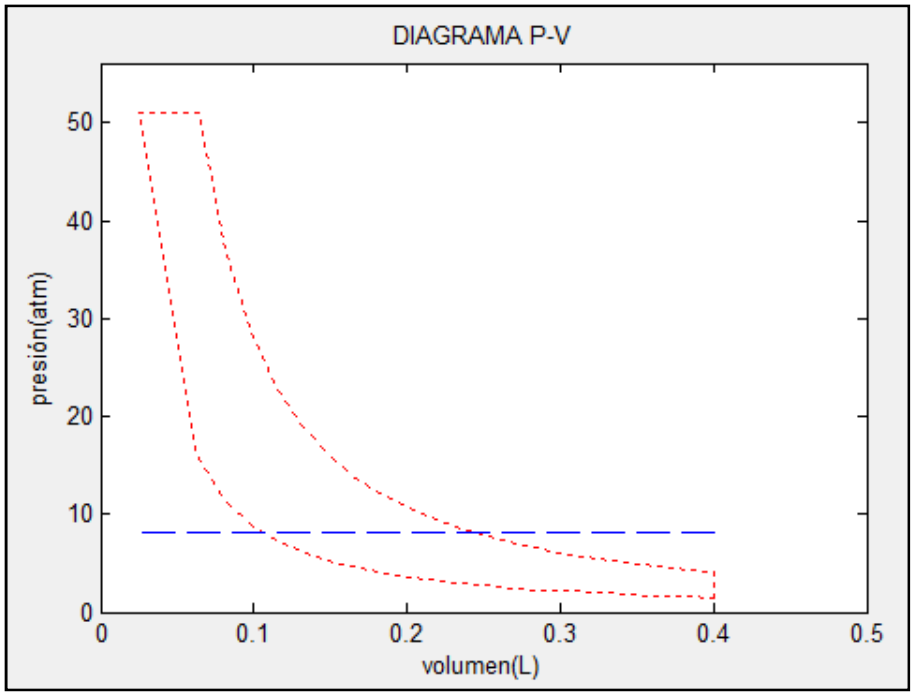


Figura 5.32: Gráfico del ciclo termodinámico

Los resultados obtenidos una vez presionado el botón “mostrar resultados” son los siguientes:

-Trabajo obtenido = 312.42 J

-Calor aportado = 718.07 J

-Rendimiento termodinámico = 0.43

-Potencia obtenida = 81789.8 W

-Presión media efectiva = 8.22 atm

-Consumo específico = 19.02 $g\cdot(KW\cdot h)^{-1}$

Figura 5.33: Resultados después ciclo

De esta forma se puede saber las características puntuales de un motor. A continuación se describe el mismo ejemplo pero estudiando las funciones que relacionan los valores máximos y mínimos.

<u>-Función “Wcil”:</u>

Se aprieta sobre el botón “Wcil” del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la cilindrada como se puede observar en la figura 5.34.

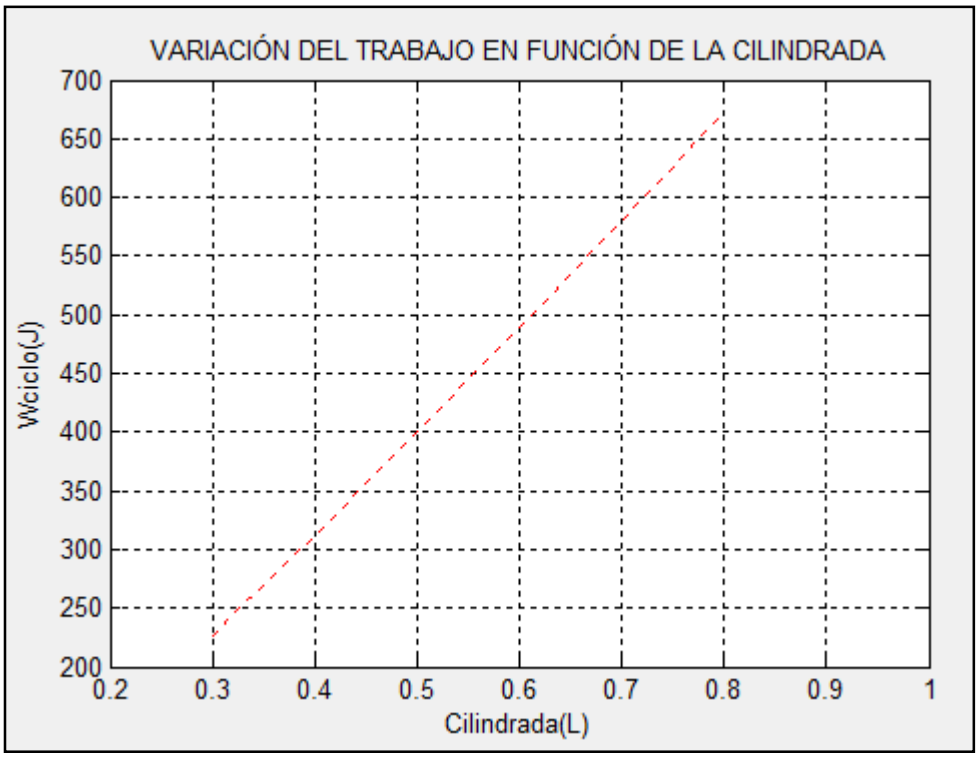


Figura 5.34: Variación del trabajo en función de la relación aire/combustible

A medida que se va aumentando la cilindrada el trabajo obtenido por ciclo aumenta. Es un resultado lógico y esperado ya que si se tiene más volumen, se tiene más área para integrar y eso implica más trabajo. En el caso de ciclo Otto pasa exactamente lo mismo.

Cabe comentar que los valores obtenidos de trabajo son bastante altos, aún en este caso son más altos que en el ciclo Otto, esto es debido a la sobrealimentación a la que está expuesto el ciclo Diesel, que hace que tenga más rendimiento volumétrico y esto aumenta el trabajo obtenido en el ciclo.

Siempre se tiene que el diagrama representado no es el indicado (real) y a pesar de intentar aproximar muchos parámetros hacia la realidad quedan algunos otros como por ejemplo las pérdidas mecánicas que no se han tenido en cuenta.

-Función “Epsilonrend”:

Se aprieta sobre el botón “Epsilonrend” del apartado de funciones y muestra gráficamente la variación del rendimiento termodinámico en función de la relación de compresión como se puede observar en la figura 5.35.

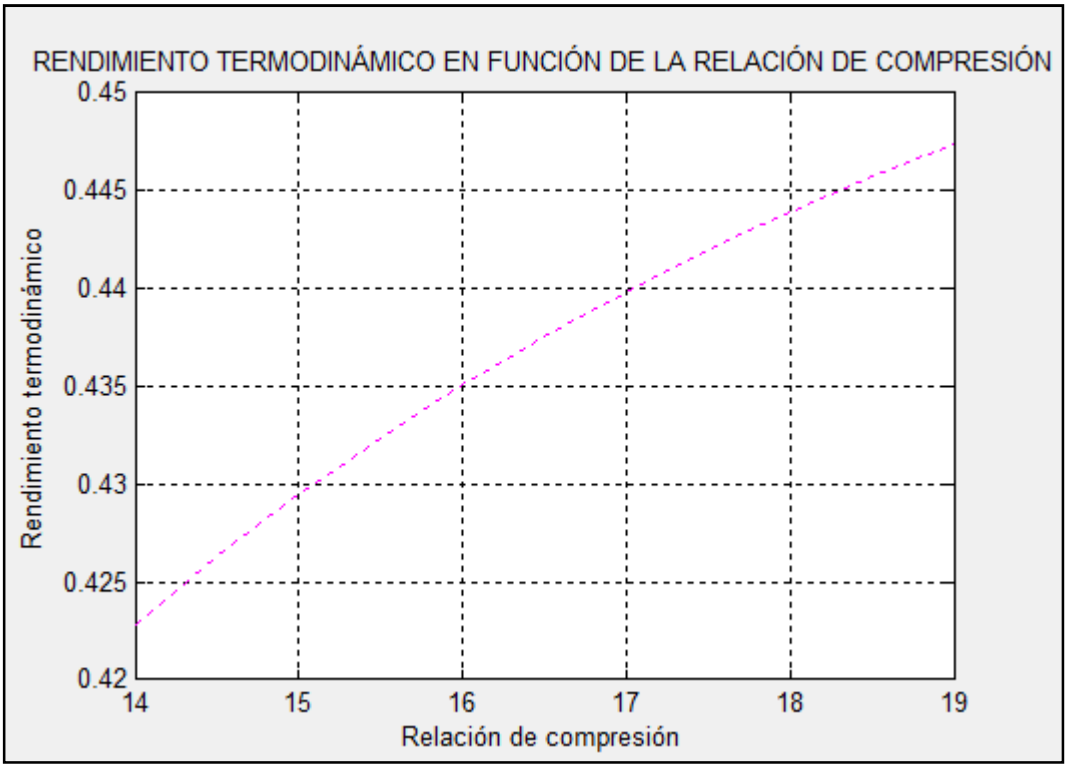


Figura 5.35: Rendimiento termodinámico en función de la relación de compresión

Como se puede observar en la figura 5.35, el rendimiento termodinámico aumenta si aumentamos la relación de compresión. Es un resultado lógico y coherente ya que se aumenta el área de la integral del diagrama termodinámico p-v (se expande más el volumen), por lo tanto, el trabajo obtenido se va incrementando y con este, el rendimiento termodinámico.

El rendimiento termodinámico del ciclo Diesel es más alto que el rendimiento termodinámico obtenido en el ciclo Otto, esto es normal ya que trabaja con relaciones de compresión mucho más elevadas. Cabe comentar que a mismas condiciones de motor (misma relación de compresión y misma sobrealimentación entre otras cosas), el rendimiento termodinámico sería mayor en el ciclo Otto.

-Función “WQ”:

Se aprieta sobre el botón “WQ” del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la relación de aire/combustible como se puede observar en la figura 5.36.

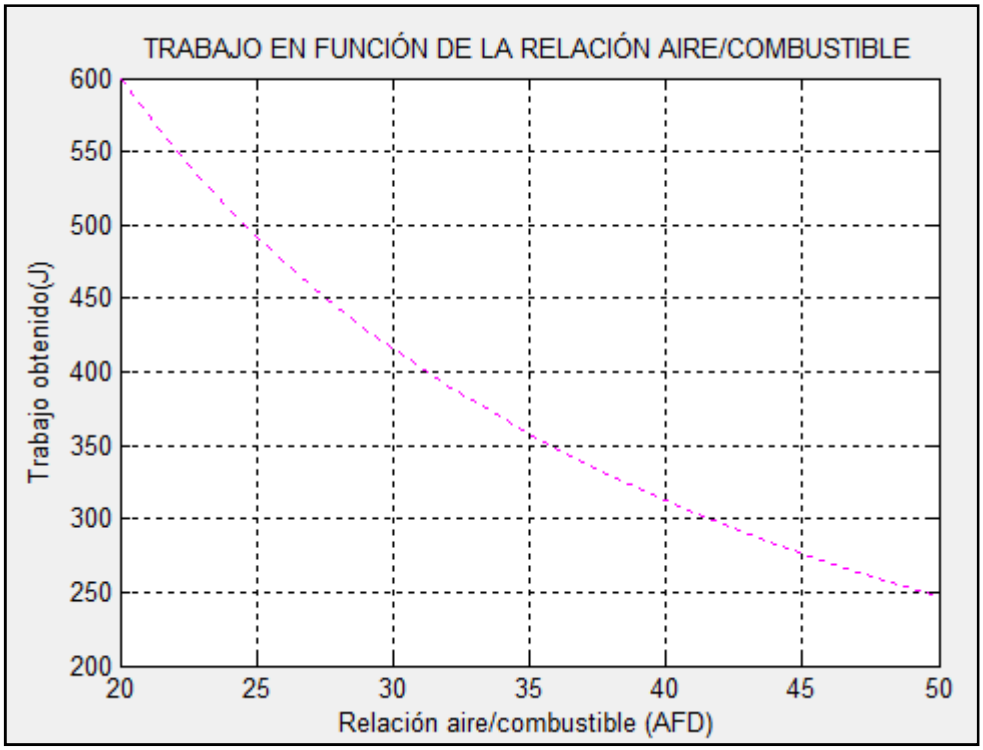


Figura 5.36: Variación del trabajo en función de la relación aire/combustible

A medida que se aumenta la relación aire/combustible, el trabajo obtenido por ciclo disminuye. Este hecho es normal, ya que aumentar la relación aire/combustible implica que el combustible que le estamos aportando al ciclo disminuye, y con menos combustible, obtenemos menos trabajo.

En este caso no hay ningún salto de pendiente como pasaba en el ciclo Otto ya que la constante de expansión politrópica en el ciclo Diesel se supone constante (visto en el apartado 4.2.3).

-Función “Parwgiro”:

Se aprieta sobre el botón “Parwgiro” del apartado de funciones y nos muestra gráficamente la variación del par motor en función de la velocidad de giro del motor como se puede observar en la figura 5.37.

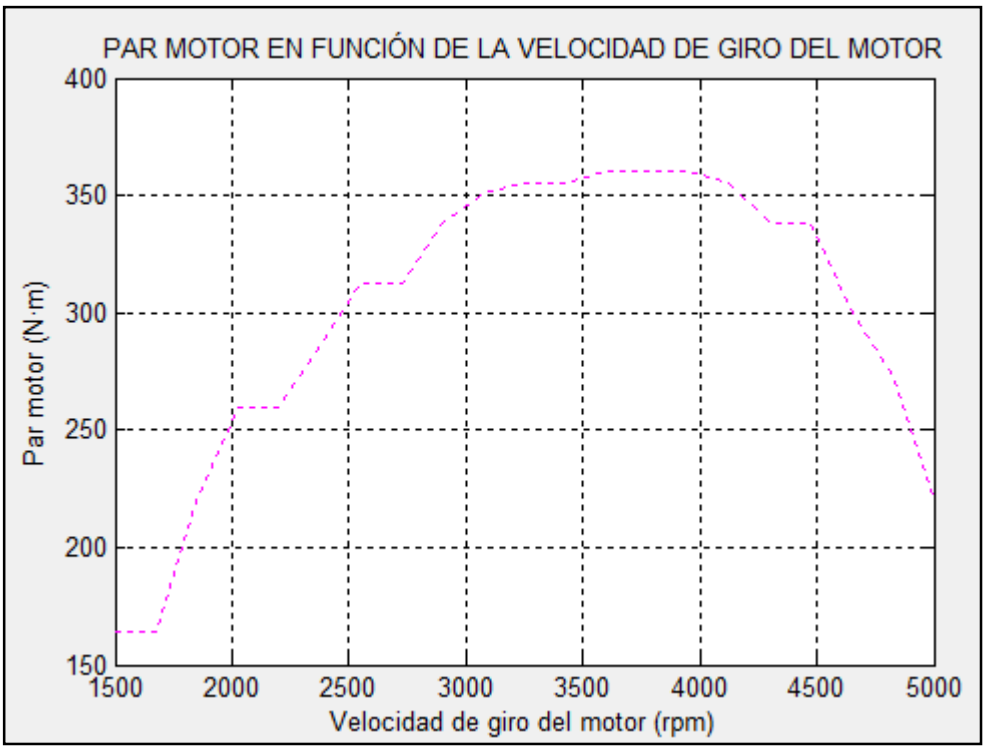


Figura 5.37: Par motor en función de la velocidad de giro del motor

A medida que se aumenta la velocidad de giro del motor, el par obtenido aumenta hasta alcanzar un máximo y seguidamente disminuye. Esto pasa sobre las 3750 rpm’s, valor razonable y similar en automoción. Esto sucede gracias al rendimiento volumétrico, que

depende de la velocidad de giro del motor y el valor más alto lo tiene sobre las 3750 rpm's en el ciclo Diesel.

Cabe comentar los saltos que realiza la función. Esto es debido a la resolución de la representación gráfica y a la resolución de la extracción de datos para introducirlos al modelo.

-Función "Wgiroge":

Se aprieta sobre el botón "Wgiroge" del apartado de funciones y nos muestra gráficamente la variación del trabajo en función de la relación de aire/combustible como se puede observar en la figura 5.37.

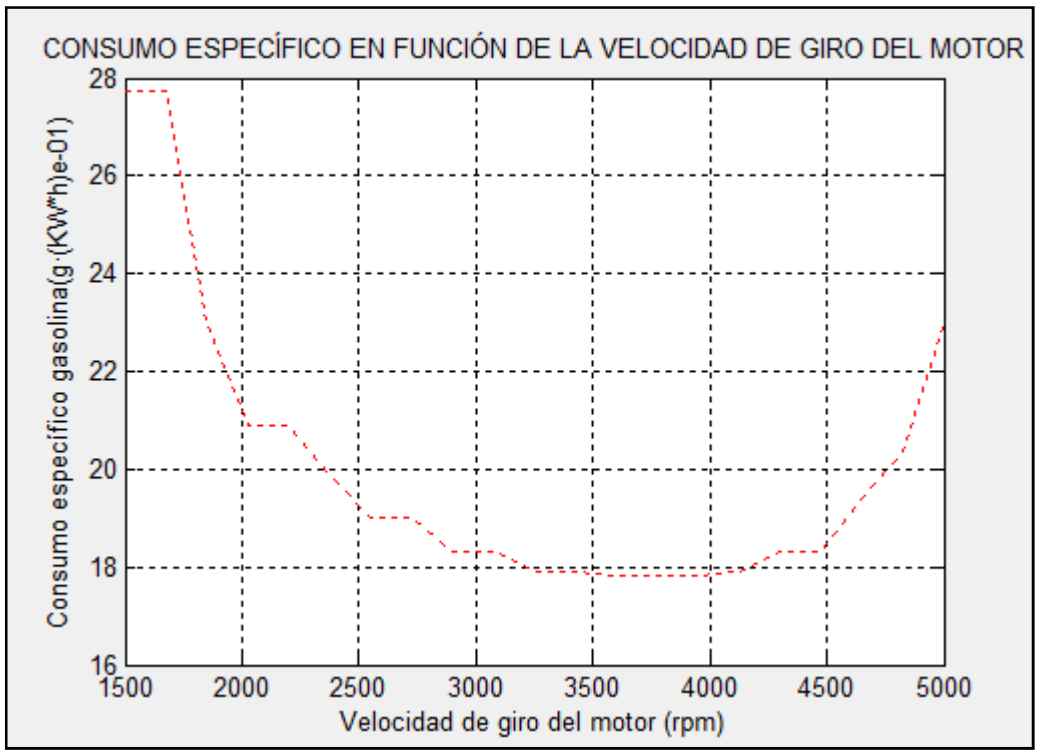


Figura 5.37: Consumo específico en función de la velocidad de giro del motor

A medida que se va aumentado la velocidad de giro del motor, el consumo específico disminuye hasta un mínimo y luego va aumentando. Esto es debido a un hecho parecido a lo que pasa con en la función "Parwgiro". Hay un momento en que la potencia es máxima, por lo tanto, cuando esto pasa, el consumo específico es mínimo.

-Función “Rcil”:

Se aprieta sobre el botón “Rcil” del apartado de funciones y nos muestra gráficamente la variación del rendimiento termodinámico en función de la cilindrada como se puede observar en la figura 5.38.

Cuando se aumenta la cilindrada, el rendimiento termodinámico aumenta, este hecho no es lógico, ya que cuanto más grande es el cilindro, más pérdidas debería tener. Cabe comentar que en el modelo, en ningún momento se han introducido pérdidas mecánicas y el programa no sabe realmente si las hay o no. Por lo tanto, que la gráfica salga coherente o que no salga coherente no relaciona las pérdidas mecánicas.

Si se investiga un poco la razón de la gráfica, se llega a la conclusión de que variando la K de expansión, la gráfica toma una curvatura diferente, es decir, si se estudia el modelo introduciendo valores de c_p por encima de 1.215, se obtiene que el rendimiento termodinámico disminuye en función de la cilindrada. La c_p del diesel está tabulada y no se puede cambiar. Por lo tanto, para que dé un resultado lógico, sería necesario introducir parámetros que hicieran penalizar mecánicamente el aumento de cilindrada. Esto se deja para futuros estudios.

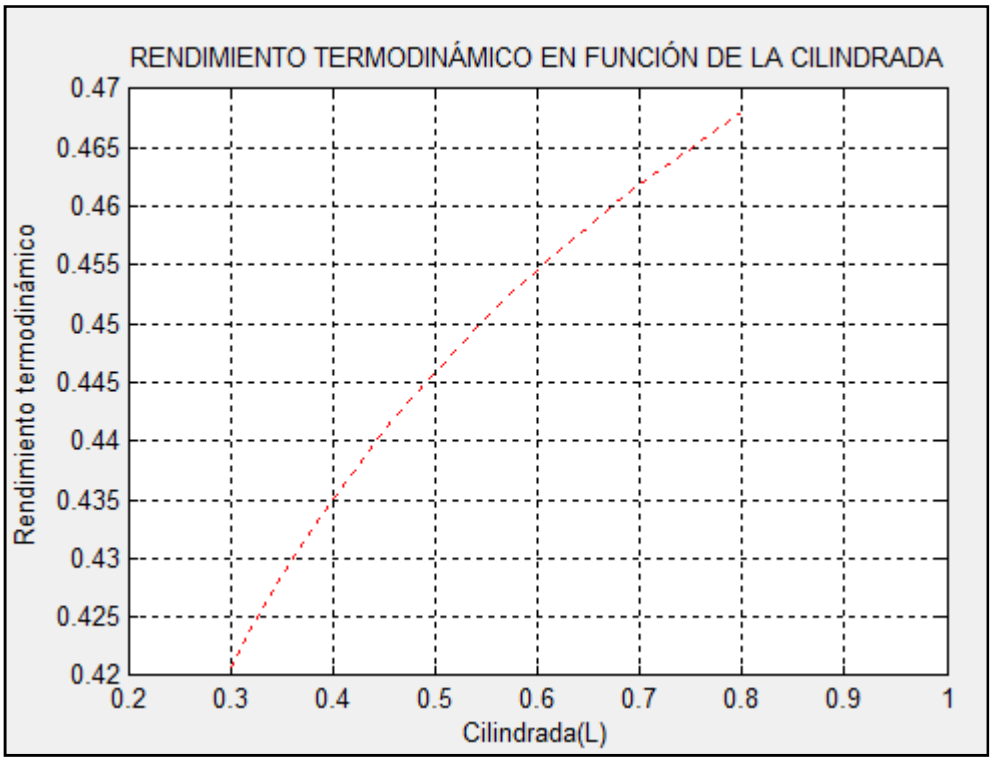


Figura 5.38: Rendimiento termodinámico en función de la cilindrada

También se tiene que tener presente que la K de expansión (directamente relacionada con la c_p) es función del volumen y en este modelo se ha supuesto constante. La K de expansión también depende de si el motor es supercuadrado, cuadrado o alargado. Esto a su vez depende de las prestaciones que deseemos obtener. Debido a la cantidad de dependencias que tiene la K de expansión y que al modelo se le ha introducido una K de expansión constante en el ciclo Diesel, no se califica de error del programa que la gráfica de la figura 5.38 sea así. Esto da pie a pensar en una posible mejora del modelo introduciendo todas las dependencias de la K de expansión comentadas.

Una vez visto todas las funciones se procede a estudiar la optimización del motor dependiendo del objetivo que se tenga. Se propone optimizar el motor de dos maneras posibles que son las siguientes: obtención del máximo trabajo por ciclo y obtención del rendimiento termodinámico más elevado para aprovechar al máximo el calor aportado.

Ya no se plantea estudiar la obtención del mínimo consumo específico para gastar lo mínimo en combustible, ya que los resultados son bastante similares en el ciclo Otto y esta posibilidad ya estaba incluida en los dos modelos al trabajar sobre la velocidad de giro óptima.

-Obtención del máximo trabajo por ciclo:

Para obtener el máximo trabajo por ciclo se observan las figuras 5.34, 5.36 y 5.37 en las cuales el trabajo (par) aparece. Si se quiere que éste sea máximo, la cilindrada tiene que ser lo más grande posible (figura 5.34), la relación aire/combustible debe de ser mínimo (figura 5.36) y la velocidad de giro del motor tiene que ser de 3750 rpm (figura 5.37).

El rango de valores en el que nos moveremos será el estipulado por las variables máximas y mínimas que se han definido anteriormente.

Partiendo de las variables iniciales del modelo, se cambiarán las siguientes variables por los siguientes valores:

-La cilindrada de 0.80 litros.

-Relación aire/combustible de 20.

-La velocidad de giro del motor de 3750 rpm.

Se realizan estos cambios en el modelo y se aprieta en la función “ciclo” y luego en “mostrar resultados” y se obtienen los resultados siguientes tal y como muestra la figura 5.28:

-Trabajo obtenido = 1366.91 J

-Calor aportado = 3104.52 J

-Rendimiento termodinámico = 0.44

-Potencia obtenida = 536781 W

-Presión media efectiva = 17.99 atm

-Consumo específico = 18.79 $g \cdot (KW \cdot h)^{-1}$

Figura 5.28: Resultados

El trabajo obtenido ha aumentado considerablemente en relación a los resultados obtenidos por el modelo inicial y sin alterar excesivamente los demás resultados, por lo tanto, se da el modelo de obtención de máximo trabajo por bueno.

-Obtención del rendimiento termodinámico más elevado:

Para la obtención del rendimiento termodinámico más elevado se observan las figuras 5.35 y 5.38 en las cuales aparece el rendimiento termodinámico. Si se desea que el rendimiento termodinámico sea máximo, la relación de compresión tiene que ser lo más grande posible (figura 5.35) y la cilindrada lo más pequeña posible a pesar de que en el estudio el rendimiento termodinámico es máximo con máxima cilindrada (ya se comentó el razonamiento de esto en la figura 5.38) y la velocidad de giro el motor debe de ser de 3750 rpm, ya que de esta forma aumentamos el trabajo obtenido y aumentaremos también el rendimiento termodinámico.

Partiendo de las variables iniciales del modelo, se cambiarán las siguientes variables por los siguientes valores:

-La relación de compresión del motor de 19.

-La cilindrada de 0.3 litros.

-La velocidad de giro del motor de 3500 rpm.

Se realizan estos cambios en el modelo y se aprieta en la función “ciclo” y luego en “mostrar resultados” y se obtienen los resultados siguientes tal y como muestra la figura 5.29:

-Trabajo obtenido = 517.58 J

-Calor aportado = 1164.20 J

-Rendimiento termodinámico = 0.44

-Potencia obtenida = 203253 W

-Presión media efectiva = 17.97 atm

-Consumo específico = 18.61 $g \cdot (KW \cdot h)^{-1}$

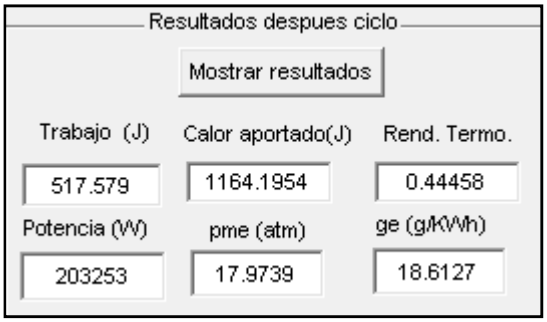


Figura 5.29: Resultados

El rendimiento termodinámico no ha aumentado considerablemente respecto el modelo incial debido a que no se ha cogido la cilindrada más grande ya que no se encuentra lógico coger la cilindrada más grande a pesar de que los resultados lo digan (razonamiento referente a figura 5.38). Las otras variables no se han alterado excesivamente, por lo tanto, se da el modelo de obtención de máximo rendimiento termodinámico por bueno.

6. Impacto medioambiental

6.1. Cálculo del impacto medioambiental ciclo Otto

Al haber diseñado un programa en el cual se pueden estudiar muchos tipos de motores, en este apartado se ejemplifica el cálculo de un motor gasolina de características iguales a las descritas en la tabla 5.1, el apartado 5.2.

La combustión se define como una reacción química producida entre un combustible (gasolina) y un comburente (aire) con desprendimiento de calor. El desprendimiento de calor se realiza de forma tan rápida e intensamente que básicamente se considera como una explosión.

El aire está compuesto básicamente por dos gases: nitrógeno (N_2) y oxígeno (0_2). En un volumen determinado de aire se encuentra una proporción de nitrógeno (N_2) del 79 % mientras que el contenido de oxígeno es aproximadamente de un 21 %. El nitrógeno durante la combustión, en principio, no se combina con nada y tal como entra en el cilindro es expulsado al exterior sin modificación alguna, excepto en pequeñas cantidades, para formar óxidos de nitrógeno (NO_x). El oxígeno es el elemento indispensable para producir la combustión de la mezcla.

El aire, aspirado, comprimido y combinado con el combustible forma una mezcla capaz de arder con gran rapidez. El salto de una chispa eléctrica provocará la ignición (explosión) y la consiguiente presión sobre el pistón. La mezcla, una vez quemada da origen a la emisión de gases como: hidrocarburos (HC) que son restos de gasolina sin quemar, el monóxido de carbono (CO) y el oxígeno (0_2) gases que aparecen debido a la combustión defectuosa, los óxidos de nitrógeno (NO_x) que surgen con temperaturas altas, el anhídrido carbónico ($C0_2$) y vapor de agua como residuos de la combustión.

La pobreza de la mezcla tiene una influencia decisiva sobre la emisión de los gases contaminantes. El motor a estudiar tiene una pobreza de 1,05.

$$\lambda = \frac{\mathrm{Fe}}{\mathrm{AFR}} = \frac{14.7}{14} = 1{,}05$$

La figura 6.1 muestra la concentración de monóxido de carbono y dióxido de carbono en porcentaje y la concentración de hidrocarburos en partes por millón en función de la pobreza de la mezcla.

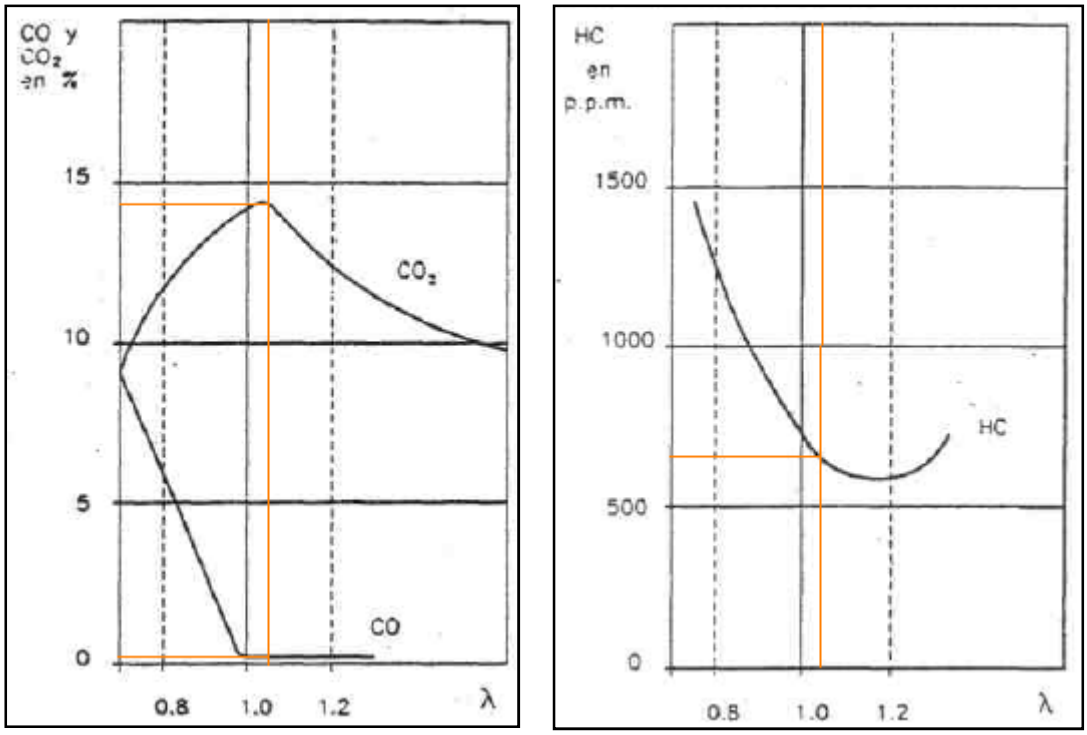


Figura 6.1: Concentración de monóxido de carbono, dióxido de carbono y hidrocarburos [9]

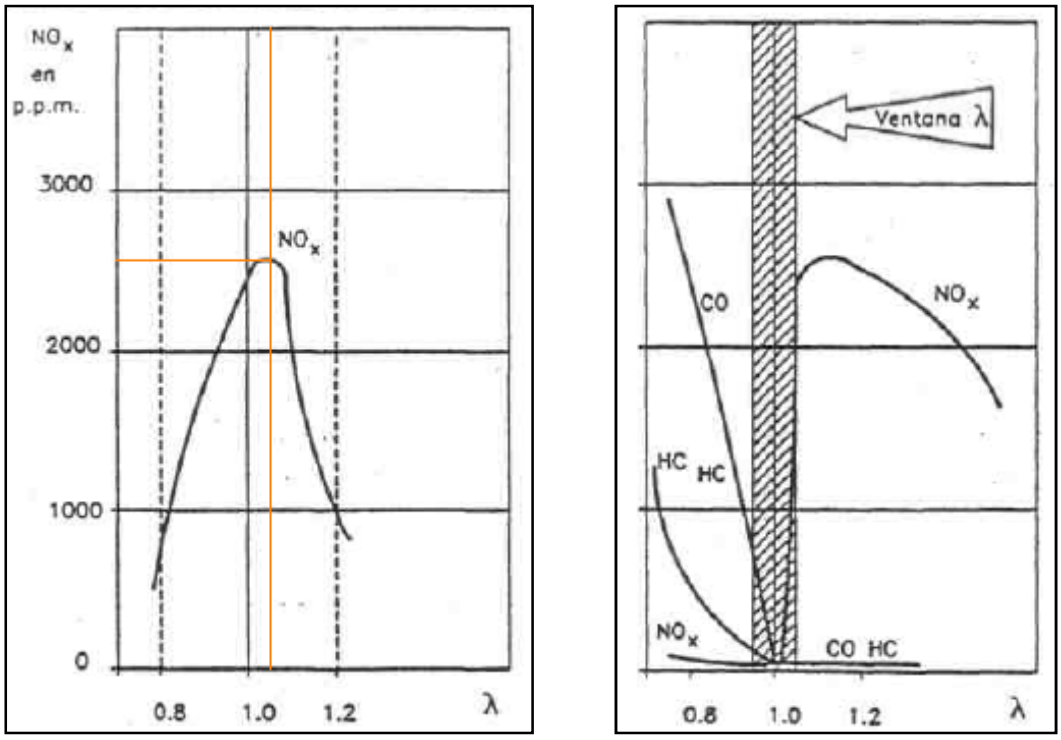


Figura 6.2: Óxidos de nitrógeno a la izquierda y a la derecha la zona ventana λ [9]

Entonces, la combustión genera:

- 14,5 % de CO_2
- 0.1 % de CO
- 650 PPM de HC
- 2550 PPM de NO_x

Inyectamos m_combustible [g] de gasolina al ciclo. Si se mira internamente este valor es de 26,56 mg. Por lo tanto:

$$\frac{26{,}56\text{mg gasolina}}{\text{cilindro y ciclo}} \times \frac{14\text{mg aire}}{1\text{ mg gasolina}} = \frac{371{,}84\text{ mg aire}}{\text{cilindro y ciclo}}$$

La masa de aire introducida a la cámara de combustión es de 371,84 mg, sumando los 26,56 mg de gasolina que se inyecta, se tiene un total de 398,4 mg de mezcla.

Tomando en consideración la masa molecular de la mezcla 0 y considerando que el motor a estudiar tiene 4 cilindros:

$$\frac{398{,}4\text{ mg mezcla}}{\text{cilindro y ciclo}} \times \frac{1\text{g}}{1000\text{mg}} \text{x} \frac{1\text{mol}}{30{,}3\text{g}} \times 4\text{ cilindros} = 1{,}31\text{E-}02\ \frac{\text{mols}}{\text{ciclo}}$$

Por lo tanto, el volumen de los gases quemados lo obtenemos aplicando la ecuación de los gases ideales:

$$P \times V = n \times R \times T$$

$$V = 1{,}31\text{E-}02 \times 0.082 \times (273.15 + 25) = 0{,}32\ \text{litros} = 0{,}32\text{E-}0.3\ \text{m}^3$$

Con el volumen de los gases quemados, podemos obtener el volumen total y la masa total de las emisiones:

Gas	Concentración (%vol o PPM)	Volumen en condiciones estándar (m3)	Massa (g)
CO_2	14,5%	4,64E-05	9,2E-02
CO	0,10%	3,20E-07	6,28E-04
HC	650	2,08E-07	2,79E-04
No_x	2550	8,16E-07	1,67E-03

Tabla 6.1: Emisiones ciclo Otto

Para pasar de m^3 a masa (g) se multiplica por el peso molecular y se divide por $22.4 m^3$/kmol.

Se han de tener en cuenta las siguientes suposiciones efectuadas en el cálculo de la masa:

- El peso molecular del HC se ha considerado igual al peso atómico del C_2H_3 (30.06 g/mol).
- El peso molecular del NO_x se ha considerado igual al peso atómico del NO_2 (46.01 g/mol).

Cabe comentar que los valores de la tabla 6.1 son valores por ciclo termodinámico.

Una vez calculado las emisiones de este motor, se plantea la reducción de la emisión de contaminantes. Resulta muy difícil limitar al mismo tiempo los cuatro gases contaminantes principales: CO, HC, CO_2 y NO_x. La zona que a los valores mínimos de las emisiones de CO y HC corresponde al valor máximo de NO_x.

Para conseguir al mismo tiempo una reducción drástica de CO y de NOx y obtener así un buen comportamiento de los HC, sería preciso garantizar una combustión completa con un factor λ superior a 1.05.

Existe una zona llamada “ventana lambda” (Figura 6.2 izquierda) dónde la proporción de gases es mínima y si puede conseguirse que el motor trabaje en esa zona, se garantiza una reducción de los gases contaminantes.

Esta condición impone en la práctica buscar soluciones técnicas que garanticen el funcionamiento correcto en todas las condiciones de servicio del motor. A continuación se describen algunas:

- La implementación de sistemas de inyección de gasolina con mando electrónico.
- La regulación de la mezcla para que trabaje cercana a la "ventana lambda". y el uso del catalizador.
- El uso del catalizador.

De este modo se consigue reducir la emisión de gases contaminantes mediante la optimización de la combustión y la posterior depuración de los gases.

Se podría diseñar un motor para que la emisión de contaminantes fuera lo más pequeña posible, basándose en la “ventana lambda” e introduciendo como principal parámetro de diseño la pobreza de la mezcla (el parámetro sería AFR).

Si lo hiciéramos obtendríamos un motor optimizado para la mejora del medio ambiente (menos emisión de contaminantes), en cambio, otros parámetros como la potencia máxima o rendimientos ya sean volumétricos o de la combustión podrían disminuir.

6.2. Cálculo del impacto medioambiental ciclo Diesel

En este apartado se realiza un cálculo similar al apartado 6.1 pero en ciclo Diesel. Para realizarlo se ha supuesto un motor ciclo diesel con un AFD de 25.

Primero se calcula la pobreza de la mezcla:

$$\lambda = \frac{\text{Fe}}{\text{AFD}} = \frac{14.5}{25} = 0{,}58$$

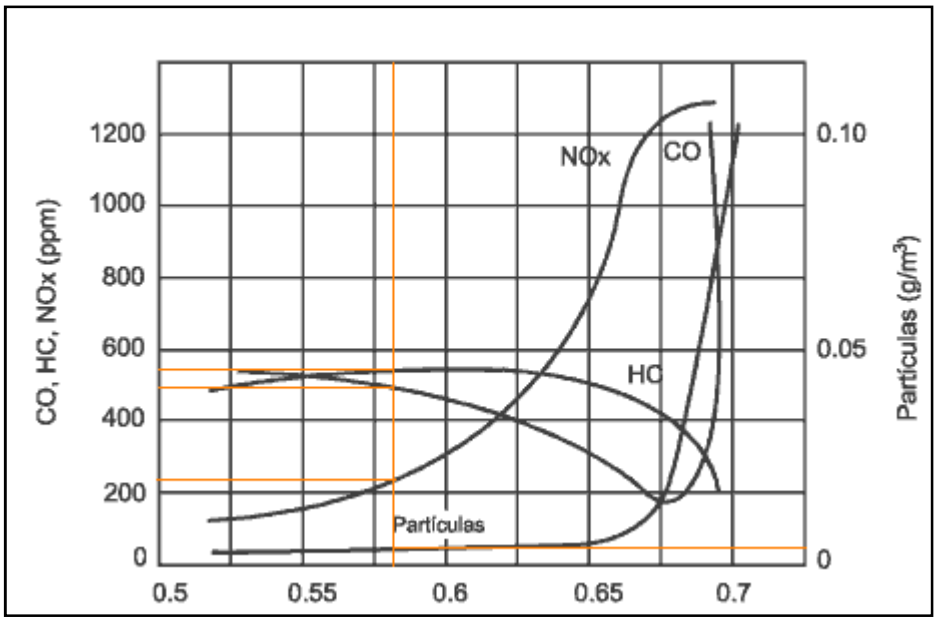


Figura 6.3: Contaminantes ciclo Diesel [1]

La figra 6.3 muestra los contaminantes generados por la combustión, a continuación se cuantifican:

- 4.17 g/m^3 de partículas
- 500 PPM de CO
- 550 PPM de HC

- 240 PPM de NO_x

Inyectamos m_diesel [g] de diesel al ciclo. Si se mira internamente este valor es de 22,06 mg. Por lo tanto:

$$\frac{22{,}06\text{mg diesel}}{\text{cilindro y ciclo}} \times \frac{25\text{mg aire}}{1\text{ mg gasolina}} = \frac{551{,}5\text{ mg aire}}{\text{cilindro y ciclo}}$$

La masa de aire introducida a la cámara de combustión es de 882,4 mg, sumando los 22,06 mg de diesel que se inyecta, se tiene un total de 904,46 mg de mezcla.

Para considerar la masa molecular de la mezcla haremos un promedio de la masa molecular del aire y de la masa molecular del diesel:

$$\frac{25}{26}\text{xm\_molar}_{\text{aire}} + (\text{m\_diesel})\text{x}\frac{1}{26} = \frac{25\text{x}28.97}{26} + \frac{1\text{x}198{,}36}{26} = 35{,}48\frac{\text{g}}{\text{mol}}$$

Considerando que el motor a estudiar tiene 4 cilindros:

$$\frac{551{,}5\text{ mg mezcla}}{\text{cilindro y ciclo}} \times \frac{1\text{g}}{1000\text{mg}}\text{x}\frac{1\text{mol}}{35{,}48\text{g}} \times 4\text{ cilindros} = 1{,}55\text{E-}02\ \frac{\text{mols}}{\text{ciclo}}$$

Por lo tanto, el volumen de los gases quemados lo obtenemos aplicando la ecuación de los gases ideales:

$$\text{P}\times\text{V} = \text{n}\times\text{R}\times\text{T}$$

$$\text{V} = 1{,}55\text{E-}02 \times 0.082 \times (273.15+25) = 0{,}38\ \text{litros} = 0{,}38\text{E-}0.3\ \text{m}^3$$

Con el volumen de los gases quemados, podemos obtener el volumen total y la masa total de las emisiones:

Componente	Valor y unidades	Volumen en condiciones estándar (m3)	Massa (g)
Partículas	4,17 g/m^3	-	1,52E-03
CO	500 PPM	1,9E-07	3,73E-04
HC	550 PPM	2,09E-07	2,85E-04
No_x	240 PPM	9,12E-08	1,87E-04

Tabla 6.2: Emisiones ciclo Diesel

Para pasar de m^3 a masa (g) se multiplica por el peso molecular y se divide por 22.4m^3/kmol.

Se han de tener en cuenta las siguientes suposiciones efectuadas en el cálculo de la masa:

- El peso molecular del HC se ha considerado igual al peso atómico del C_2H_3 (30.06 g/mol).
- El peso molecular del NO_x se ha considerado igual al peso atómico del NO_2 (46.01 g/mol).

Cabe comentar que los valores de la tabla 6.2 son valores por ciclo termodinámico.

Para reducir los contaminantes en un motor diesel existen sistemas de anitpolución como los siguientes: [10]

-Para reducir CO i Hc:

Calentamiento rápido del motor.

Pre-calentamiento del aire admitido

Optimización de la homogeneidad de la mezcla

Inyección de aire en los gases de escape

Catalizador de oxidación

Absorción de los vapores de depósito

-Para reducir NO_x:

Bajar la relación de compresión → Bajar la presión y la temperatura al final de la combustión

Disminuir los puntos calientes en la c.c.

Disminuir el ángulo de encendida para bajar la T de combustión

Reciclar los gases de de escape (válvula ERG)

Catalizador de reducción

-Para reducir las partículas:

Mejorar la polvorización del gasoil

Filtro de partículas

7. Presupuesto

El siguiente presupuesto se basa en la faena realizada en el proyecto, las licencias de software y el hardware necesario para que se pueda utilizar el software. Se deja para un posible comprador el estudio de la amortización.

	Importe aspecto laboral	
		Observaciones
Fecha de inicio		01/02/2008
Fecha de finalización		05/12/2008
Total dias	309	Dedicación al proyecto
Factor de trabajo	4/7	Se trabajan 4 dias de cada 7 en proyecto
Dias trabajados	177	Total de dias x factor de trabajo
Factor horario	3/24	Se trabajan 3 horas cada dia
Horas trabajadas	531	Dias trabajados x factor horario
Precio/hora	10 €	
Subtotal	5.310 €	

Importe software y hardware		
Concepto	Precio	Observaciones
Matlab 7,0 Ordenador	2.200,00 € 800,00€	Licencia Matlab 7,0
Subtotal	3.000,00 €	

Importe global		
Concepto	Precio	Observaciones
Subtotal IVA	8.310,00 € 1.329,6€	Laboral + software-hardware
Total presupuesto	9.639,6 €	

Conclusiones

Gracias a la evolución de las nuevas tecnologías como la informática, se han podido realizar cálculos que hace unos años eran impensables. Por esta razón cada vez se utilizan más herramientas informáticas para dar soporte a la ingeniería en fase de desarrollo del producto.

La herramienta presentada anteriormente permite simular un ciclo termodinámico completo y junto con las condiciones de diseño, permite encontrar las características que ha de tener para funcionar de manera óptima y deseada.

La utilización de este programa permite el conocimiento de muchas variables termodinámicas de un motor como temperaturas, volumen y presiones en cada estado. También permite el conocimiento de variables como el trabajo obtenido, potencia, presión media efectiva, rendimiento termodinámico y consumo específico entre otras sin necesidad de un motor y las correspondientes sondas.

Nunca se ha de olvidar que un motor real se ve afectado por muchos otros parámetros que no han estado considerados en la simulación, como pueden ser el rendimiento mecánico o la existencia de fase abierta real. También se ha de tener en cuenta que se puede mejorar el ajuste de los parámetros introducidos en la simulación.

Agradecimientos

Quiero mostrar públicamente mi agradecimiento hacia mi tutor Ernesto por el interés y esfuerzo mostrado durante la realización del proyecto. Quiero agradecer a mi familia más cercana; padre, madre y hermanos el gran interés que han mostrado durante el transcurso del aprendizaje en la E.T.S.E.I.B y la gran paciencia que han tenido conmigo. Finalmente agradecer a Sandra y a los amigos más íntimos por estar siempre a mi lado en los momentos fáciles y difíciles.

Pido disculpas a todo aquel que no haya estado anunciado anteriormente y también haya tenido alguna parte relevante en la ejecución de mis estudios y del proyecto.

Bibliografía

Referencias bibliográficas

[1] ÁLVAREZ, J., CALLEJÓN, I. *Máquinas térmicas motoras – 1,* Barcelona: EDICIONS UPC. 2002.

[2] DEPARTAMENTO DE MÁQUINAS TÉRMICAS. *Transparencias.* Publicaciones Abast, Setiembre 2006.

[3] MIGUEL ÁNGEL PÉREZ BELLÓ, *Tecnología de los motores*, Barcelona: ETSEIB - CPDA . 1985.

[4] J.A. YAMIN, O.O. BADRAN. *Analytical styudy to minimise the heat losses from a propane powered 4-stroke spark ignition engine. Journal of Renewable Energy*. Vol. 27, 2002, p.463-478.

[5] M.A. CEVIZ, I. KAYMAZ. *Temperature and air-fuel ratio dependent specific heat ratio functions for lean burned and unburned mixture. Journal of Energy Conversion and Management Vol.46, 2005, p.2387-2404.*

[6] A.A. BUZUKOV, B.P. TIMOSHENKO. *Effect of secondary mixing on fuel ignition and combustion in a diesel engine. Journal of Combustion, Explosion, and Shock Waves Vol. 33, 1997, p.26-33.*

[7] TAYLOR, C. *The Internal Combustion Engine in Theory and Practice*, Massachusetts:
M.I.T. PRESS. 1985.

[8] UNIVERSITAT POLITÈCNICA DE CATALUNYA. *Taules i gràfiques de propietats termodinàmiques*. Setembre, 2002.

[9] INSTITUTO LEONARDO DA VINCI. *La combustión y los gases de escape*.

[10] ÁLVAREZ, J., CALLEJÓN, I., LÓPEZ, J. *Transparències Ampliació de Motors Tèrmics,* Barcelona: CPDA. 2003.

Bibliografía complementaria

- WIKIMEDIA PROJECT, *Wikipedia, the free* [http://www.wikipedia.org; 15 d'abril de 2008]

- TODOMOTORES, Arreglo de Motor [http://www.todomotores.cl; 9 de marzo de 2008]

- UNIVERSIDAD POLITÉCNICA DE MADRID. *Aprenda Matlab 7.0 como si estuviera en primero*. Madrid, Diciembre 2005.

ANEXOS

A. Funciones

A continuación se presentan todas las líneas de código del Matlab asociadas a sus correspondientes funciones.

A.1 Función “glo”

Ejecutando esta función permite ver a través del Matlab, el estado de las variables del guide escribiendo en el Matlab la variable que se quiere visualizar.

```
global wgiro1
global wgiro21
global rho1
global rho21
global v01
global v021
global AF1 T20 T30
global AF21 m_diesel m_combustible renddiesel GE rendgasolina
renddiesel
global p v Wcompresion Wutil Wexpansion WutilJ WJ Pc Pexpansion
Putil PutilW par ghorario ges gespecifico rendtermoint
global otto wgiro rendvolum diesel PAR Wgiro po V W rhod rhog pat
par1 par2
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17
p18 p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43
p44 p45 p46 p47 p48 p49 p50
global v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17
v18 v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43
v44 v45 v46 v47 v48 v49 v50
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F v30
```

A.2 Función “ciclo”

Permite ejecutar un ciclo termodinámico internamente en el Matlab sin mostrar ningún resultado

```
%CALCULO DE LA PRIMERA ADIABATICA
rendvol;

if otto==1
    rho=rhog;
    k=cpotto/cvotto;
    p1=potto;
```

```
elseif diesel==1
    rho=rhod;
    k=cpdiesel/cvdiesel;
    p1=pdiesel;
end

v20=v1/rho;


%cálculo de la constante de compresión
ctecompresion=p1*v1^k;
%para saber cuanto resto de cada volumen, es como la resolución
x=(v1-v20)/20;
%empezamos a calcular v i p en cada estado restando x a v
v2=v1-x;
p2=p1*(v1/v2)^k;
v3=v2-x;
p3=p2*(v2/v3)^k;
v4=v3-x;
p4=p3*(v3/v4)^k;
v5=v4-x;
p5=p4*(v4/v5)^k;
v6=v5-x;
p6=p5*(v5/v6)^k;
v7=v6-x;
p7=p6*(v6/v7)^k;
v8=v7-x;
p8=p7*(v7/v8)^k;
v9=v8-x;
p9=p8*(v8/v9)^k;
v10=v9-x;
p10=p9*(v9/v10)^k;
v11=v10-x;
p11=p10*(v10/v11)^k;
v12=v11-x;
p12=p11*(v11/v12)^k;
v13=v12-x;
p13=p12*(v12/v13)^k;
v14=v13-x;
p14=p13*(v13/v14)^k;
v15=v14-x;
p15=p14*(v14/v15)^k;
v16=v15-x;
p16=p15*(v15/v16)^k;
v17=v16-x;
p17=p16*(v16/v17)^k;
v18=v17-x;
p18=p17*(v17/v18)^k;
v19=v18-x;
p19=p18*(v18/v19)^k;
%no calculo v20 ya que es un dato
p20=p19*(v19/v20)^k;

%APORTACIÓN DE CALOR, SE DIFERENCIA ENTRE SI EL CICLO ES OTTO O
DIESEL

if      otto==1
        rendgasoline;
```

```
        n=p1*v1*rendvolum/(R*T1);
        m_aire=m_mol_aire*n;
        m_combustible=m_aire/AFR;
        Q=m_combustible*PCIgasolina*rendgasolina;
        incrementoT=Q/((m_aire+m_combustible)*cvotto);
        T20=T1*(v1/v20)^(k-1);
        T30=T20+incrementoT;
        v30=v20;
        p30=n*R*T30/v30;
        landa=AFR/14.7;
        newkk;

elseif  diesel==1
        T20=T1*(v1/v20)^(k-1);
        rendiesel;
        n=p1*v1*rendvolum/(R*T1);
        m_aire=m_mol_aire*n;
        m_diesel=m_aire/AFD;
        Q=m_diesel*PCIdiesel*renddiesel;
        incrementoT=Q/((m_diesel+m_aire)*cpdiesel);
        p30=p20;
        T30=T20+incrementoT;
        v30=(n+m_diesel/masamolardiesel)*R*T30/p30;
        k=1.38;

end

%CALCULO DE LA SEGUNA ADIABÁTICA
%para saber cuanto resto de cada volumen, es como la resolución
x=(v1-v30)/20;
%cálculo de la constante de expansión
ctexpansion=p30*v30^k;
%empezamos a calcular v i p en cada estado restando x a v
v31=v30+x;
p31=p30*(v30/v31)^k;
v32=v31+x;
p32=p31*(v31/v32)^k;
v33=v32+x;
p33=p32*(v32/v33)^k;
v34=v33+x;
p34=p33*(v33/v34)^k;
v35=v34+x;
p35=p34*(v34/v35)^k;
v36=v35+x;
p36=p35*(v35/v36)^k;
v37=v36+x;
p37=p36*(v36/v37)^k;
v38=v37+x;
p38=p37*(v37/v38)^k;
v39=v38+x;
p39=p38*(v38/v39)^k;
v40=v39+x;
p40=p39*(v39/v40)^k;
v41=v40+x;
p41=p40*(v40/v41)^k;
v42=v41+x;
p42=p41*(v41/v42)^k;
v43=v42+x;
```

```
p43=p42*(v42/v43)^k;
v44=v43+x;
p44=p43*(v43/v44)^k;
v45=v44+x;
p45=p44*(v44/v45)^k;
v46=v45+x;
p46=p45*(v45/v46)^k;
v47=v46+x;
p47=p46*(v46/v47)^k;
v48=v47+x;
p48=p47*(v47/v48)^k;
v49=v48+x;
p49=p48*(v48/v49)^k;
v50=v1;
p50=p49*(v49/v50)^k;


%CÁLCULO DEL W I LA POTENCIA PRODUCIDA EN EL CICLO

if otto==1
    syms p;
    syms v;
    p='ctecompresion/v^k';
    Wcompresion=int(p,v1,v20);
    p='ctexpansion/v^k';
    Wexpansion=int(p,v30,v50);
    Wutil=Wexpansion-abs(Wcompresion);
    %multiplico por 101.32 ya que 1atm*l=101.32J
    WutilJ=Wutil*101.32;
    %Normalmente mostraremos este resultado
    WJ=subs(WutilJ);
    %al pasar de rpm a rad/s tengo que multiplicar por 2*pi/60 ya la
    %potencia la tengo en W
    Pc=abs(Wcompresion*101.32)*wgiro*2*pi/60;
    Pexpansion=(Wexpansion*101.32)*(wgiro*2*pi/60);
    Putil=Pexpansion-abs(Pc);
    %Normalmente mostraremos este resultado
    PutilW=subs(Putil);
    %par en N*m es WJ
    par=PutilW/(wgiro*2*pi/60);
    %consumo horario (g/hora)
    ghorario=m_combustible*wgiro*60/2;
    %consumo específico (g/(KW*hora))
    ges=ghorario/(PutilW/1000);
    gespecifico=subs(ges);


elseif diesel==1
    syms p;
    syms v;
    p='ctecompresion/v^k';
    Wcompresion=int(p,v1,v20);
    Wexpansion1=p20*(v30-v20);
    p='ctexpansion/v^k';
    Wexpansion2=int(p,v30,v50);
    Wutil=Wexpansion1+Wexpansion2-abs(Wcompresion);
    %multiplico por 101.32 ya que 1atm*l=101.32J
    WutilJ=Wutil*101.32;
```

```
    %Normalmente mostraremos este resultado
    WJ=subs(WutilJ);
    %al pasar de rpm a rad/s tengo que multiplicar por 2*pi/60 ya la
    %potencia la tengo en W
    Pc=abs(Wcompresion*101.32)*wgiro*2*pi/60;
    Pexpansion=(Wexpansion1+Wexpansion2)*101.32*(wgiro*2*pi/60);
    Putil=Pexpansion-abs(Pc);
    %Normalmente mostraremos este resultado
    PutilW=subs(Putil);
    %par en N*m
    par=PutilW/(wgiro*2*pi/60);
     %consumo horario (g/hora)
    ghorario=m_diesel*wgiro*60/2;
    %consumo específico (g/(KW*hora))
    ges=ghorario/(PutilW/1000);
    gespecifico=subs(ges);
end

%CALCULO DEL RENDIMIENTO TERMODINÁMICO INTEGRANDO Y PME
rendtermoint=(WJ/Q);
pme=WJ/((v1-v20)*101.32);
```

A.3 Función “grafico”

Ejecutando esta función a través del Matlab permite la representación gráfica del ciclo termodinámico a estudiar.

```
%REPRESENTACIÓN DEL GRÁFICO CON TODOS LOS PUNTOS ENCONTRADOS
%calculo de la presión media efectiva
pme=WJ/((v1-v20)*101.32);
p=[p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17 p18
p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43 p44
p45 p46 p47 p48 p49 p50 p1];
v=[v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18
v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43 v44
v45 v46 v47 v48 v49 v50 v1];
plot(v,p,'r:')
z=[pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,p
me,pme,pme,pme];
s=[v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18
v19 v20];
hold on
plot(s,z,'b--')
hold off
%para configurar la dimensión de los ejes
axis([0,v1+0.1,0,p30+5])
xlabel('volumen(L)')
ylabel('presión(atm)')
title('DIAGRAMA P-V')
```

A.4 Función “newkk”

Esta función calcula la nueva K de expansión en ciclo Otto.

```
if otto==1
    if (T30<=250)
            k=1.4;
        elseif (T30>250)&(T30<=350)
            k=1.37;
        elseif (T30>350)&(T30<=450)
            k=1.365;
        elseif (T30>450)&(T30<=550)
            k=1.35;
        elseif (T30>550)&(T30<=700)
            k=1.34;
        elseif (T30>700)&(T30<=825)
            k=1.325;
        elseif (T30>825)&(T30<=975)
            k=1.31;
        elseif (T30>975)&(T30<=1100)
            k=1.3;
        elseif (T30>1100)&(T30<=1225)
            k=1.29;
        elseif (T30>1225)&(T30<=1350)
            k=1.285;
        elseif (T30>1350)&(T30<=1475)
            k=1.275;
        elseif (T30>1475)&(T30<=1600)
            k=1.27;
        elseif (T30>1600)&(T30<=1725)
            k=1.263;
        elseif (T30>1725)&(T30<=1850)
            k=1.26;
        elseif (T30>1850)&(T30<=1975)
            k=1.258;
        elseif (T30>1975)&(T30<=2100)
            k=1.255;
        elseif (T30>2100)&(T30<=2225)
            k=1.252;
        elseif (T30>2225)
            k=1.22;
    end
end
```

A.5 Función “rendgasoline”

Esta función calcula el rendimiento de la combustión ciclo Otto.

```
if otto==1
      if D==1
        if (wgiro<=1250)
            rendgasolina=0.82;
        elseif (wgiro>1250)&(wgiro<=1750)
```

```
            rendgasolina=0.8375;
          elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.85;
          elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.855;
          elseif (wgiro>2750)
            rendgasolina=0.865;
          end

        elseif E==1
           if (wgiro<=1250)
            rendgasolina=0.8475;
          elseif (wgiro>1250)&(wgiro<=1750)
            rendgasolina=0.8625;
          elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.875;
          elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.88;
          elseif (wgiro>2750)
            rendgasolina=0.8875;
           end

        elseif F==1
           if (wgiro<=1250)
            rendgasolina=0.845;
          elseif (wgiro>1250)&(wgiro<=1750)
            rendgasolina=0.8575;
          elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.875;
          elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.8775;
          elseif (wgiro>2750)
            rendgasolina=0.885;
           end
        end
    end
```

A.6 Función “rendiesel”

Esta función calcula el rendimiento de la combustión ciclo Diesel.

```
    if diesel==1
          if A==1
           if (T20<=749)
            renddiesel=0.85;
          elseif (T20>749)&(T20<=750)
            renddiesel=0.9;
          elseif (T20>750)&(T20<=755)
            renddiesel=0.9375;
          elseif (T20>755)&(T20<=775)
            renddiesel=0.95;
          elseif (T20>775)&(T20<=800)
            renddiesel=0.9375;
          elseif (T20>800)&(T20<=850)
```

```
            renddiesel=0.85;
        elseif (T20>850)&(T20<=875)
            renddiesel=0.8;
        elseif (T20>875)&(T20<=900)
            renddiesel=0.73;
        elseif (T20>900)&(T20<=910)
            renddiesel=0.6875;
        elseif (T20>910)&(T20<=915)
            renddiesel=0.625;
        elseif (T20>915)
            renddiesel=0.58;
        end

    elseif B==1
        if (T20<=685)
            renddiesel=0.812;
        elseif (T20>685)&(T20<=690)
            renddiesel=0.84;
        elseif (T20>690)&(T20<=695)
            renddiesel=0.88;
        elseif (T20>695)&(T20<=697)
            renddiesel=0.9;
        elseif (T20>697)&(T20<=705)
            renddiesel=0.935;
        elseif (T20>705)&(T20<=715)
            renddiesel=0.95;
        elseif (T20>715)&(T20<=725)
            renddiesel=0.985;
        elseif (T20>725)&(T20<=735)
            renddiesel=0.98;
        elseif (T20>735)&(T20<=750)
            renddiesel=0.96;
        elseif (T20>750)&(T20<=760)
            renddiesel=0.9375;
        elseif (T20>760)&(T20<=775)
            renddiesel=0.89;
        elseif (T20>775)&(T20<=800)
            renddiesel=0.825;
        elseif (T20>800)&(T20<=825)
            renddiesel=0.77;
        elseif (T20>825)&(T20<=845)
            renddiesel=0.695;
        elseif (T20>845)&(T20<=850)
            renddiesel=0.625;
        elseif (T20>850)&(T20<=860)
            renddiesel=0.6;
        elseif (T20>860)&(T20<=875)
            renddiesel=0.52;
        elseif (T20>875)&(T20<=880)
            renddiesel=0.4375;
        elseif (T20>880)&(T20<=885)
            renddiesel=0.375;
        elseif (T20>885)&(T20<=890)
            renddiesel=0.3125;
        elseif (T20>890)
            renddiesel=0.25;
         end
```

```
        elseif C==1
            if (T20<=670)
                renddiesel=0.83;
            elseif (T20>670)&(T20<=690)
                renddiesel=0.875;
            elseif (T20>690)&(T20<=695)
                renddiesel=0.88;
            elseif (T20>695)&(T20<=697)
                renddiesel=0.9;
            elseif (T20>697)&(T20<=705)
                renddiesel=0.935;
            elseif (T20>705)&(T20<=715)
                renddiesel=0.95;
            elseif (T20>715)&(T20<=725)
                renddiesel=0.985;
            elseif (T20>725)&(T20<=735)
                renddiesel=0.98;
            elseif (T20>735)&(T20<=750)
                renddiesel=0.96;
            elseif (T20>750)&(T20<=760)
                renddiesel=0.9375;
            elseif (T20>760)&(T20<=775)
                renddiesel=0.89;
            elseif (T20>775)&(T20<=800)
                renddiesel=0.825;
            elseif (T20>800)&(T20<=825)
                renddiesel=0.77;
            elseif (T20>825)&(T20<=845)
                renddiesel=0.695;
            elseif (T20>845)&(T20<=850)
                renddiesel=0.625;
            elseif (T20>850)&(T20<=860)
                renddiesel=0.6;
            elseif (T20>860)&(T20<=875)
                renddiesel=0.52;
            elseif (T20>875)&(T20<=880)
                renddiesel=0.4375;
            elseif (T20>880)&(T20<=885)
                renddiesel=0.375;
            elseif (T20>885)&(T20<=890)
                renddiesel=0.3125;
            elseif (T20>890)
                renddiesel=0.25;
            end

        end

    end
```

A.7 Función “rendvol”

Esta función calcula el rendimiento de la volumétrico ciclo Otto y Diesel.

```
if otto==1
    if (wgiro>=1000)&(wgiro<1250)
        rendvolum=0.52;
    elseif (wgiro>=1250)&(wgiro<1500)
        rendvolum=0.575;
    elseif (wgiro>=1500)&(wgiro<1750)
        rendvolum=0.635;
    elseif (wgiro>=1750)&(wgiro<2000)
        rendvolum=0.67;
    elseif (wgiro>=2000)&(wgiro<2250)
        rendvolum=0.72;
    elseif (wgiro>=2250)&(wgiro<2500)
        rendvolum=0.75;
    elseif (wgiro>=2500)&(wgiro<2750)
        rendvolum=0.76;
    elseif (wgiro>=2750)&(wgiro<3000)
        rendvolum=0.775;
    elseif (wgiro>=3000)&(wgiro<3250)
        rendvolum=0.79;
    elseif (wgiro>=3250)&(wgiro<3500)
        rendvolum=0.8;
    elseif (wgiro>=3500)&(wgiro<3750)
        rendvolum=0.8;
    elseif (wgiro>=3750)&(wgiro<4000)
        rendvolum=0.79;
    elseif (wgiro>=4000)&(wgiro<4250)
        rendvolum=0.78;
    elseif (wgiro>=4250)&(wgiro<4500)
        rendvolum=0.775;
    elseif (wgiro>=4500)&(wgiro<4750)
        rendvolum=0.75;
    elseif (wgiro>=4750)&(wgiro<5000)
        rendvolum=0.725;
    elseif (wgiro>=5000)&(wgiro<5250)
        rendvolum=0.7;
    elseif (wgiro>=5250)&(wgiro<5500)
        rendvolum=0.65;
    elseif (wgiro>=5500)&(wgiro<5750)
        rendvolum=0.6;
    elseif (wgiro>=5750)&(wgiro<6000)
        rendvolum=0.55;
    elseif (wgiro>=6000)&(wgiro<6250)
        rendvolum=0.45;
    end

elseif diesel==1
    if (wgiro>=1000)&(wgiro<1250)
        rendvolum=0.6;
    elseif (wgiro>=1250)&(wgiro<1500)
        rendvolum=0.775;
    elseif (wgiro>=1500)&(wgiro<1750)
        rendvolum=0.9;
    elseif (wgiro>=1750)&(wgiro<2000)
        rendvolum=1;
    elseif (wgiro>=2000)&(wgiro<2250)
        rendvolum=1.075;
    elseif (wgiro>=2250)&(wgiro<2500)
        rendvolum=1.125;
```

```
    elseif (wgiro>=2500)&(wgiro<2750)
        rendvolum=1.175;
    elseif (wgiro>=2750)&(wgiro<3000)
        rendvolum=1.225;
    elseif (wgiro>=3000)&(wgiro<3250)
        rendvolum=1.25;
    elseif (wgiro>=3250)&(wgiro<3500)
        rendvolum=1.26;
    elseif (wgiro>=3500)&(wgiro<3750)
        rendvolum=1.27;
    elseif (wgiro>=3750)&(wgiro<4000)
        rendvolum=1.27;
    elseif (wgiro>=4000)&(wgiro<4250)
        rendvolum=1.26;
    elseif (wgiro>=4250)&(wgiro<4500)
        rendvolum=1.225;
    elseif (wgiro>=4500)&(wgiro<4750)
        rendvolum=1.15;
    elseif (wgiro>=4750)&(wgiro<5000)
        rendvolum=1.1;
    elseif (wgiro>=5000)&(wgiro<5250)
        rendvolum=1;
    end

end
```

A.8 Función “ciclo_guide”

Esta función es el Guide.

```
function varargout = ciclo_guide(varargin)
% CICLO_GUIDE M-file for ciclo_guide.fig
%      CICLO_GUIDE, by itself, creates a new CICLO_GUIDE or raises
the existing
%      singleton*.
%
%      H = CICLO_GUIDE returns the handle to a new CICLO_GUIDE or
the handle to
%      the existing singleton*.
%
%      CICLO_GUIDE('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in CICLO_GUIDE.M with the given input
arguments.
%
%      CICLO_GUIDE('Property','Value',...) creates a new CICLO_GUIDE
or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before ciclo_guide_OpeningFunction gets
called.  An
%      unrecognized property name or invalid value makes property
application
```

```
%      stop.  All inputs are passed to ciclo_guide_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES


% Edit the above text to modify the response to help ciclo_guide

% Last Modified by GUIDE v2.5 19-Nov-2008 19:04:50

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ciclo_guide_OpeningFcn, ...
                   'gui_OutputFcn',  @ciclo_guide_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT



% --- Executes just before ciclo_guide is made visible.
function ciclo_guide_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ciclo_guide (see VARARGIN)

% Choose default command line output for ciclo_guide
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ciclo_guide wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ciclo_guide_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
```

```
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in Otto.
function Otto_Callback(hObject, eventdata, handles)
% hObject    handle to Otto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of Otto

global otto
global diesel
otto=1;
diesel=0;



% --- Executes on button press in Diesel.
function Diesel_Callback(hObject, eventdata, handles)
% hObject    handle to Diesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of Diesel

global otto
global diesel
diesel=1;
otto=0;



function velgiro_Callback(hObject, eventdata, handles)
% hObject    handle to velgiro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of velgiro as text
%        str2double(get(hObject,'String')) returns contents of
velgiro as a double

global wgiro
wgiro=str2double(get(hObject,'String'));


% --- Executes during object creation, after setting all properties.
function velgiro_CreateFcn(hObject, eventdata, handles)
% hObject    handle to velgiro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function PCIOTTO_Callback(hObject, eventdata, handles)
% hObject    handle to PCIOTTO (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of PCIOTTO as text
%        str2double(get(hObject,'String')) returns contents of
PCIOTTO as a double

global PCIgasolina
PCIgasolina=str2double(get(hObject,'String'));


% --- Executes during object creation, after setting all properties.
function PCIOTTO_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PCIOTTO (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end


function PCIDIESEL_Callback(hObject, eventdata, handles)
% hObject    handle to PCIDIESEL (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of PCIDIESEL as text
%        str2double(get(hObject,'String')) returns contents of
PCIDIESEL as a double

global PCIdiesel
PCIdiesel=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
```

```
function PCIDIESEL_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PCIDIESEL (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function masamolaraire_Callback(hObject, eventdata, handles)
% hObject    handle to masamolaraire (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of masamolaraire as
text
%        str2double(get(hObject,'String')) returns contents of
masamolaraire as a double

global m_mol_aire
m_mol_aire=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function masamolaraire_CreateFcn(hObject, eventdata, handles)
% hObject    handle to masamolaraire (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function Masamolardiesel_Callback(hObject, eventdata, handles)
% hObject    handle to Masamolardiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of Masamolardiesel
as text
%        str2double(get(hObject,'String')) returns contents of
Masamolardiesel as a double

global masamolardiesel
masamolardiesel=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function Masamolardiesel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Masamolardiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function Runiversal_Callback(hObject, eventdata, handles)
% hObject    handle to Runiversal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Runiversal as
text
%        str2double(get(hObject,'String')) returns contents of
Runiversal as a double

global R
R=str2double(get(hObject,'String'));


% --- Executes during object creation, after setting all properties.
function Runiversal_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Runiversal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function AFR_Callback(hObject, eventdata, handles)
% hObject    handle to AFR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AFR as text
%        str2double(get(hObject,'String')) returns contents of AFR
as a double

global AFR
AFR=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function AFR_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AFR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function AFD_Callback(hObject, eventdata, handles)
% hObject    handle to AFD (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AFD as text
%        str2double(get(hObject,'String')) returns contents of AFD
as a double

global AFD
AFD=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function AFD_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AFD (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function p1otto_Callback(hObject, eventdata, handles)
% hObject    handle to p1otto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of p1otto as text
%        str2double(get(hObject,'String')) returns contents of
p1otto as a double

global potto
potto=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function p1otto_CreateFcn(hObject, eventdata, handles)
% hObject    handle to p1otto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function p1diesel_Callback(hObject, eventdata, handles)
% hObject    handle to p1diesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of p1diesel as text
%        str2double(get(hObject,'String')) returns contents of
p1diesel as a double

global pdiesel
pdiesel=str2double(get(hObject,'String'));
```

```
% --- Executes during object creation, after setting all properties.
function p1diesel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to p1diesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function T1_Callback(hObject, eventdata, handles)
% hObject    handle to T1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of T1 as text
%        str2double(get(hObject,'String')) returns contents of T1 as
a double

global T1
T1=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function T1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function cpotto_Callback(hObject, eventdata, handles)
% hObject    handle to cpotto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of cpotto as text
%        str2double(get(hObject,'String')) returns contents of
cpotto as a double

global cpotto
cpotto=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function cpotto_CreateFcn(hObject, eventdata, handles)
% hObject    handle to cpotto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function cvdiesel_Callback(hObject, eventdata, handles)
% hObject    handle to cvdiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of cvdiesel as text
%        str2double(get(hObject,'String')) returns contents of
cvdiesel as a double

global cvdiesel
cvdiesel=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function cvdiesel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to cvdiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end
```

```
function cvotto_Callback(hObject, eventdata, handles)
% hObject    handle to cvotto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of cvotto as text
%        str2double(get(hObject,'String')) returns contents of
cvotto as a double

global cvotto
cvotto=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function cvotto_CreateFcn(hObject, eventdata, handles)
% hObject    handle to cvotto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function cpdiesel_Callback(hObject, eventdata, handles)
% hObject    handle to cpdiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of cpdiesel as text
%        str2double(get(hObject,'String')) returns contents of
cpdiesel as a double

global cpdiesel
cpdiesel=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function cpdiesel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to cpdiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
```

```
end



function rhootto_Callback(hObject, eventdata, handles)
% hObject    handle to rhootto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rhootto as text
%        str2double(get(hObject,'String')) returns contents of
rhootto as a double

global rhog
rhog=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function rhootto_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rhootto (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function rhodiesel_Callback(hObject, eventdata, handles)
% hObject    handle to rhodiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rhodiesel as text
%        str2double(get(hObject,'String')) returns contents of
rhodiesel as a double

global rhod
rhod=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function rhodiesel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rhodiesel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function cilindrada_Callback(hObject, eventdata, handles)
% hObject    handle to cilindrada (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of cilindrada as
text
%        str2double(get(hObject,'String')) returns contents of
cilindrada as a double

global v1
v1=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function cilindrada_CreateFcn(hObject, eventdata, handles)
% hObject    handle to cilindrada (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function t_Callback(hObject, eventdata, handles)
% hObject    handle to t (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of t as text
%        str2double(get(hObject,'String')) returns contents of t as
a double


% --- Executes during object creation, after setting all properties.
function t_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to t (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end


% --- Executes on selection change in Tipo.
function Tipo_Callback(hObject, eventdata, handles)
% hObject    handle to Tipo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns Tipo contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from
Tipo


% --- Executes during object creation, after setting all properties.
function Tipo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Tipo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




% --- Executes on button press in D.
function D_Callback(hObject, eventdata, handles)
% hObject    handle to D (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of D

global D
```

```
global E
global F
D=1;
E=0;
F=0;



% --- Executes on button press in F.
function F_Callback(hObject, eventdata, handles)
% hObject    handle to F (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of F

global D
global E
global F
D=0;
E=0;
F=1;



% --- Executes on button press in E.
function E_Callback(hObject, eventdata, handles)
% hObject    handle to E (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of E

global D
global E
global F
D=0;
E=1;
F=0;




function wgiro1_Callback(hObject, eventdata, handles)
% hObject    handle to wgiro1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of wgiro1 as text
%        str2double(get(hObject,'String')) returns contents of
wgiro1 as a double

global wgiro1
wgiro1=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function wgiro1_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to wgiro1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function wgiro21_Callback(hObject, eventdata, handles)
% hObject    handle to wgiro21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of wgiro21 as text
%        str2double(get(hObject,'String')) returns contents of
wgiro21 as a double

global wgiro21
wgiro21=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function wgiro21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wgiro21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function rho1_Callback(hObject, eventdata, handles)
% hObject    handle to rho1 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rho1 as text
%        str2double(get(hObject,'String')) returns contents of rho1
as a double

global rho1
rho1=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function rho1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rho1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function rho21_Callback(hObject, eventdata, handles)
% hObject    handle to rho21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rho21 as text
%        str2double(get(hObject,'String')) returns contents of rho21
as a double

global rho21
rho21=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function rho21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rho21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end
```

```
function AF1_Callback(hObject, eventdata, handles)
% hObject    handle to AF1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AF1 as text
%        str2double(get(hObject,'String')) returns contents of AF1
as a double

global AF1
AF1=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function AF1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AF1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end
```

```
function AF21_Callback(hObject, eventdata, handles)
% hObject    handle to AF21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of AF21 as text
%        str2double(get(hObject,'String')) returns contents of AF21
as a double

global AF21
AF21=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function AF21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to AF21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function v01_Callback(hObject, eventdata, handles)
% hObject    handle to v01 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of v01 as text
%        str2double(get(hObject,'String')) returns contents of v01
as a double

global v01
v01=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function v01_CreateFcn(hObject, eventdata, handles)
% hObject    handle to v01 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function v021_Callback(hObject, eventdata, handles)
% hObject    handle to v021 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of v021 as text
%        str2double(get(hObject,'String')) returns contents of v021
as a double

global v021
v021=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function v021_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to v021 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



% --- Executes on button press in ciclo.
function ciclo_Callback(hObject, eventdata, handles)
% hObject    handle to ciclo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global otto wgiro rendvolum diesel R m_diesel T30 T20 renddiesel
rendgasolina pme Q gespecifico
R=0.08205784;

if otto==1
    if (wgiro>=1000)&(wgiro<1250)
        rendvolum=0.52;
    elseif (wgiro>=1250)&(wgiro<1500)
        rendvolum=0.575;
    elseif (wgiro>=1500)&(wgiro<1750)
        rendvolum=0.635;
    elseif (wgiro>=1750)&(wgiro<2000)
        rendvolum=0.67;
    elseif (wgiro>=2000)&(wgiro<2250)
        rendvolum=0.72;
    elseif (wgiro>=2250)&(wgiro<2500)
        rendvolum=0.75;
    elseif (wgiro>=2500)&(wgiro<2750)
        rendvolum=0.76;
    elseif (wgiro>=2750)&(wgiro<3000)
        rendvolum=0.775;
    elseif (wgiro>=3000)&(wgiro<3250)
        rendvolum=0.79;
    elseif (wgiro>=3250)&(wgiro<3500)
        rendvolum=0.8;
    elseif (wgiro>=3500)&(wgiro<3750)
        rendvolum=0.8;
    elseif (wgiro>=3750)&(wgiro<4000)
        rendvolum=0.79;
    elseif (wgiro>=4000)&(wgiro<4250)
        rendvolum=0.78;
    elseif (wgiro>=4250)&(wgiro<4500)
        rendvolum=0.775;
    elseif (wgiro>=4500)&(wgiro<4750)
        rendvolum=0.75;
```

```
        elseif (wgiro>=4750)&(wgiro<5000)
            rendvolum=0.725;
        elseif (wgiro>=5000)&(wgiro<5250)
            rendvolum=0.7;
        elseif (wgiro>=5250)&(wgiro<5500)
            rendvolum=0.65;
        elseif (wgiro>=5500)&(wgiro<5750)
            rendvolum=0.6;
        elseif (wgiro>=5750)&(wgiro<6000)
            rendvolum=0.55;
        elseif (wgiro>=6000)
            rendvolum=0.45;
        end

    elseif diesel==1
        if (wgiro>=1000)&(wgiro<1250)
            rendvolum=0.6;
        elseif (wgiro>=1250)&(wgiro<1500)
            rendvolum=0.775;
        elseif (wgiro>=1500)&(wgiro<1750)
            rendvolum=0.9;
        elseif (wgiro>=1750)&(wgiro<2000)
            rendvolum=1;
        elseif (wgiro>=2000)&(wgiro<2250)
            rendvolum=1.075;
        elseif (wgiro>=2250)&(wgiro<2500)
            rendvolum=1.125;
        elseif (wgiro>=2500)&(wgiro<2750)
            rendvolum=1.175;
        elseif (wgiro>=2750)&(wgiro<3000)
            rendvolum=1.225;
        elseif (wgiro>=3000)&(wgiro<3250)
            rendvolum=1.25;
        elseif (wgiro>=3250)&(wgiro<3500)
            rendvolum=1.26;
        elseif (wgiro>=3500)&(wgiro<3750)
            rendvolum=1.27;
        elseif (wgiro>=3750)&(wgiro<4000)
            rendvolum=1.27;
        elseif (wgiro>=4000)&(wgiro<4250)
            rendvolum=1.26;
        elseif (wgiro>=4250)&(wgiro<4500)
            rendvolum=1.225;
        elseif (wgiro>=4500)&(wgiro<4750)
            rendvolum=1.15;
        elseif (wgiro>=4750)&(wgiro<5000)
            rendvolum=1.1;
        elseif (wgiro>=5000)
            rendvolum=1;
        end

    end

    global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto rhod
    global p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17
    p18 p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43
    p44 p45 p46 p47 p48 p49 p50
```

```
global v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17
v18 v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43
v44 v45 v46 v47 v48 v49 v50
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F pat


if otto==1
    rho=rhog;
    k=cpotto/cvotto;
    p1=potto;
elseif diesel==1
    rho=rhod;
    k=cpdiesel/cvdiesel;
    p1=pdiesel;
end

v20=v1/rho;


%cálculo de la constante de compresión
ctecompresion=p1*v1^k;
%para saber cuanto resto de cada volumen, es como la resolución
x=(v1-v20)/20;
%empezamos a calcular v i p en cada estado restando x a v
v2=v1-x;
p2=p1*(v1/v2)^k;
v3=v2-x;
p3=p2*(v2/v3)^k;
v4=v3-x;
p4=p3*(v3/v4)^k;
v5=v4-x;
p5=p4*(v4/v5)^k;
v6=v5-x;
p6=p5*(v5/v6)^k;
v7=v6-x;
p7=p6*(v6/v7)^k;
v8=v7-x;
p8=p7*(v7/v8)^k;
v9=v8-x;
p9=p8*(v8/v9)^k;
v10=v9-x;
p10=p9*(v9/v10)^k;
v11=v10-x;
p11=p10*(v10/v11)^k;
v12=v11-x;
p12=p11*(v11/v12)^k;
v13=v12-x;
p13=p12*(v12/v13)^k;
v14=v13-x;
p14=p13*(v13/v14)^k;
v15=v14-x;
p15=p14*(v14/v15)^k;
v16=v15-x;
p16=p15*(v15/v16)^k;
v17=v16-x;
p17=p16*(v16/v17)^k;
```

```
v18=v17-x;
p18=p17*(v17/v18)^k;
v19=v18-x;
p19=p18*(v18/v19)^k;
%no calculo v20 ya que es un dato
p20=p19*(v19/v20)^k;

%APORTACIÓN DE CALOR, SE DIFERENCIA ENTRE SI EL CICLO ES OTTO O
DIESEL

if      otto==1
        if D==1
        if (wgiro<=1250)
            rendgasolina=0.82;
        elseif (wgiro>1250)&(wgiro<=1750)
            rendgasolina=0.8375;
        elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.85;
        elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.855;
        elseif (wgiro>2750)
            rendgasolina=0.865;
        end

        elseif E==1
         if (wgiro<=1250)
            rendgasolina=0.8475;
        elseif (wgiro>1250)&(wgiro<=1750)
            rendgasolina=0.8625;
        elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.875;
        elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.88;
        elseif (wgiro>2750)
            rendgasolina=0.8875;
         end

        elseif F==1
         if (wgiro<=1250)
            rendgasolina=0.845;
        elseif (wgiro>1250)&(wgiro<=1750)
            rendgasolina=0.8575;
        elseif (wgiro>1750)&(wgiro<=2250)
            rendgasolina=0.875;
        elseif (wgiro>2250)&(wgiro<=2750)
            rendgasolina=0.8775;
        elseif (wgiro>2750)
            rendgasolina=0.885;
         end
      end
        n=p1*v1*rendvolum/(R*T1);
        m_aire=m_mol_aire*n;
        m_combustible=m_aire/AFR;
        Q=m_combustible*PCIgasolina*rendgasolina;
        incrementoT=Q/((m_aire+m_combustible)*cvotto);
        T20=T1*(v1/v20)^(k-1);
        T30=T20+incrementoT;
        v30=v20;
```

```
            p30=n*R*T30/v30;
            landa=AFR/14.7;
            if (T30<=250)
                k=1.4;
            elseif (T30>250)&(T30<=350)
                k=1.37;
            elseif (T30>350)&(T30<=450)
                k=1.365;
            elseif (T30>450)&(T30<=550)
                k=1.35;
            elseif (T30>550)&(T30<=700)
                k=1.34;
            elseif (T30>700)&(T30<=825)
                k=1.325;
            elseif (T30>825)&(T30<=975)
                k=1.31;
            elseif (T30>975)&(T30<=1100)
                k=1.3;
            elseif (T30>1100)&(T30<=1225)
                k=1.29;
            elseif (T30>1225)&(T30<=1350)
                k=1.285;
            elseif (T30>1350)&(T30<=1475)
                k=1.275;
            elseif (T30>1475)&(T30<=1600)
                k=1.27;
            elseif (T30>1600)&(T30<=1725)
                k=1.263;
            elseif (T30>1725)&(T30<=1850)
                k=1.26;
            elseif (T30>1850)&(T30<=1975)
                k=1.258;
            elseif (T30>1975)&(T30<=2100)
                k=1.255;
            elseif (T30>2100)&(T30<=2225)
                k=1.252;
            elseif (T30>2225)
                k=1.22;
        end

    elseif  diesel==1
            T20=T1*(v1/v20)^(k-1);
            if A==1
            if (T20<=749)
                renddiesel=0.85;
            elseif (T20>749)&(T20<=750)
                renddiesel=0.9;
            elseif (T20>750)&(T20<=755)
                renddiesel=0.9375;
            elseif (T20>755)&(T20<=775)
                renddiesel=0.95;
            elseif (T20>775)&(T20<=800)
                renddiesel=0.9375;
            elseif (T20>800)&(T20<=850)
                renddiesel=0.85;
            elseif (T20>850)&(T20<=875)
                renddiesel=0.8;
            elseif (T20>875)&(T20<=900)
```

```
            renddiesel=0.73;
        elseif (T20>900)&(T20<=910)
            renddiesel=0.6875;
        elseif (T20>910)&(T20<=915)
            renddiesel=0.625;
        elseif (T20>915)
            renddiesel=0.58;
        end

    elseif B==1
        if (T20<=685)
            renddiesel=0.812;
        elseif (T20>685)&(T20<=690)
            renddiesel=0.84;
        elseif (T20>690)&(T20<=695)
            renddiesel=0.88;
        elseif (T20>695)&(T20<=697)
            renddiesel=0.9;
        elseif (T20>697)&(T20<=705)
            renddiesel=0.935;
        elseif (T20>705)&(T20<=715)
            renddiesel=0.95;
        elseif (T20>715)&(T20<=725)
            renddiesel=0.985;
        elseif (T20>725)&(T20<=735)
            renddiesel=0.98;
        elseif (T20>735)&(T20<=750)
            renddiesel=0.96;
        elseif (T20>750)&(T20<=760)
            renddiesel=0.9375;
        elseif (T20>760)&(T20<=775)
            renddiesel=0.89;
        elseif (T20>775)&(T20<=800)
            renddiesel=0.825;
        elseif (T20>800)&(T20<=825)
            renddiesel=0.77;
        elseif (T20>825)&(T20<=845)
            renddiesel=0.695;
        elseif (T20>845)&(T20<=850)
            renddiesel=0.625;
        elseif (T20>850)&(T20<=860)
            renddiesel=0.6;
        elseif (T20>860)&(T20<=875)
            renddiesel=0.52;
        elseif (T20>875)&(T20<=880)
            renddiesel=0.4375;
        elseif (T20>880)&(T20<=885)
            renddiesel=0.375;
        elseif (T20>885)&(T20<=890)
            renddiesel=0.3125;
        elseif (T20>890)
            renddiesel=0.25;
         end


    elseif C==1
        if (T20<=670)
            renddiesel=0.83;
```

```
            elseif (T20>670)&(T20<=690)
                renddiesel=0.875;
            elseif (T20>690)&(T20<=695)
                renddiesel=0.88;
            elseif (T20>695)&(T20<=697)
                renddiesel=0.9;
            elseif (T20>697)&(T20<=705)
                renddiesel=0.935;
            elseif (T20>705)&(T20<=715)
                renddiesel=0.95;
            elseif (T20>715)&(T20<=725)
                renddiesel=0.985;
            elseif (T20>725)&(T20<=735)
                renddiesel=0.98;
            elseif (T20>735)&(T20<=750)
                renddiesel=0.96;
            elseif (T20>750)&(T20<=760)
                renddiesel=0.9375;
            elseif (T20>760)&(T20<=775)
                renddiesel=0.89;
            elseif (T20>775)&(T20<=800)
                renddiesel=0.825;
            elseif (T20>800)&(T20<=825)
                renddiesel=0.77;
            elseif (T20>825)&(T20<=845)
                renddiesel=0.695;
            elseif (T20>845)&(T20<=850)
                renddiesel=0.625;
            elseif (T20>850)&(T20<=860)
                renddiesel=0.6;
            elseif (T20>860)&(T20<=875)
                renddiesel=0.52;
            elseif (T20>875)&(T20<=880)
                renddiesel=0.4375;
            elseif (T20>880)&(T20<=885)
                renddiesel=0.375;
            elseif (T20>885)&(T20<=890)
                renddiesel=0.3125;
            elseif (T20>890)
                renddiesel=0.25;
            end

        end
            n=p1*v1*rendvolum/(R*T1);
            m_aire=m_mol_aire*n;
            m_diesel=m_aire/AFD;
            Q=m_diesel*PCIdiesel*renddiesel;
            incrementoT=Q/((m_diesel+m_aire)*cpdiesel);
            p30=p20;
            T30=T20+incrementoT;
            v30=(n+m_diesel/masamolardiesel)*R*T30/p30;
            k=1.38;

    end

    %CALCULO DE LA SEGUNA ADIABÁTICA
    %para saber cuanto resto de cada volumen, es como la resolución
    x=(v1-v30)/20;
```

```
%cálculo de la constante de expansión
ctexpansion=p30*v30^k;
%empezamos a calcular v i p en cada estado restando x a v
v31=v30+x;
p31=p30*(v30/v31)^k;
v32=v31+x;
p32=p31*(v31/v32)^k;
v33=v32+x;
p33=p32*(v32/v33)^k;
v34=v33+x;
p34=p33*(v33/v34)^k;
v35=v34+x;
p35=p34*(v34/v35)^k;
v36=v35+x;
p36=p35*(v35/v36)^k;
v37=v36+x;
p37=p36*(v36/v37)^k;
v38=v37+x;
p38=p37*(v37/v38)^k;
v39=v38+x;
p39=p38*(v38/v39)^k;
v40=v39+x;
p40=p39*(v39/v40)^k;
v41=v40+x;
p41=p40*(v40/v41)^k;
v42=v41+x;
p42=p41*(v41/v42)^k;
v43=v42+x;
p43=p42*(v42/v43)^k;
v44=v43+x;
p44=p43*(v43/v44)^k;
v45=v44+x;
p45=p44*(v44/v45)^k;
v46=v45+x;
p46=p45*(v45/v46)^k;
v47=v46+x;
p47=p46*(v46/v47)^k;
v48=v47+x;
p48=p47*(v47/v48)^k;
v49=v48+x;
p49=p48*(v48/v49)^k;
v50=v1;
p50=p49*(v49/v50)^k;


%CÁLCULO DEL W I LA POTENCIA PRODUCIDA EN EL CICLO
global p v e j Wcompresion Wutil Wexpansion Wexpansion1 Wexpansion2
WutilJ WJ Pc Pexpansion Putil PutilW par ghorario ges gespecifico
rendtermoint

if otto==1
    syms p;
    syms v;
    p='ctecompresion/v^k';
    Wcompresion=int(p,v1,v20);
    p='ctexpansion/v^k';
    Wexpansion=int(p,v30,v50);
    Wutil=Wexpansion-abs(Wcompresion);
```

```
        %multiplico por 101.32 ya que 1atm*l=101.32J
        WutilJ=Wutil*101.32;
        %Normalmente mostraremos este resultado
        WJ=subs(WutilJ);
        %al pasar de rpm a rad/s tengo que multiplicar por 2*pi/60 ya la
        %potencia la tengo en W
        Pc=abs(Wcompresion*101.32)*wgiro*2*pi/60;
        Pexpansion=(Wexpansion*101.32)*(wgiro*2*pi/60);
        Putil=Pexpansion-abs(Pc);
        PutilW=subs(Putil);
        %par en N*m es WJ;
        %consumo horario (g/hora)
        ghorario=m_combustible*wgiro*60/2;
        %consumo específico (g/(KW*hora))
        ges=ghorario/(PutilW/1000);
        gespecifico=subs(ges);


    elseif diesel==1
        syms p;
        syms v;
        p='ctecompresion/p^k';
        Wcompresion=int(p,v1,v20);
        Wexpansion1=p20*(v30-v20);
        p='ctexpansion/v^k';
        Wexpansion2=int(p,v30,v50);
        Wutil=Wexpansion1+Wexpansion2-abs(Wcompresion);
        %multiplico por 101.32 ya que 1atm*l=101.32J
        WutilJ=Wutil*101.32;
        %Normalmente mostraremos este resultado
        WJ=subs(WutilJ);
        %al pasar de rpm a rad/s tengo que multiplicar por 2*pi/60 ya la
        %potencia la tengo en W
        Pc=abs(Wcompresion*101.32)*wgiro*2*pi/60;
        Pexpansion=(Wexpansion1+Wexpansion2)*101.32*(wgiro*2*pi/60);
        Putil=Pexpansion-abs(Pc);
        %Normalmente mostraremos este resultado
        PutilW=subs(Putil);
        %par en N*m es WJ
        %consumo horario (g/hora)
        ghorario=m_diesel*wgiro*60/2;
        %consumo específico (g/(KW*hora))
        ges=ghorario/(PutilW/1000);
        gespecifico=subs(ges);
    end

    %CALCULO DEL RENDIMIENTO TERMODINÁMICO INTEGRANDO Y PME
    rendtermoint=(WJ/Q);
    pme=WJ/((v1-v20)*101.32);



    % --- Executes on button press in grafico.
    function grafico_Callback(hObject, eventdata, handles)
    % hObject    handle to grafico (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
```

```
global p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17
p18 p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43
p44 p45 p46 p47 p48 p49 p50
global v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17
v18 v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43
v44 v45 v46 v47 v48 v49 v50
global pme WJ z s p v pme

%REPRESENTACIÓN DEL GRÁFICO CON TODOS LOS PUNTOS ENCONTRADOS
%calculo de la presión media efectiva
pme=WJ/((v1-v20)*101.32);
p=[p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17 p18
p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43 p44
p45 p46 p47 p48 p49 p50 p1];
v=[v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18
v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43 v44
v45 v46 v47 v48 v49 v50 v1];
plot(v,p,'r:')
z=[pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,pme,p
me,pme,pme,pme];
s=[v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18
v19 v20];
hold on
plot(s,z,'b--')
hold off
%para configurar la dimensión de los ejes
axis([0,v1+0.1,0,p30+5])
xlabel('volumen(L)')
ylabel('presión(atm)')
title('DIAGRAMA P-V')



% --- Executes on button press in WQ.
function WQ_Callback(hObject, eventdata, handles)
% hObject    handle to WQ (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14
W15 W16 W17 W18 W19 W20 W21 W
global Qc Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17
Q18 Q19 Q20 Q21
global AF AF21 AF1 AFR1 AFR2 AFR3 AFR4 AFR5 AFR6 AFR7 AFR8 AFR9
AFR10 AFR11 AFR12 AFR13 AFR14 AFR15 AFR16 AFR17 AFR18 AFR19 AFR20
AFR21
global AFD1 AFD2 AFD3 AFD4 AFD5 AFD6 AFD7 AFD8 AFD9 AFD10 AFD11
AFD12 AFD13 AFD14 AFD15 AFD16 AFD17 AFD18 AFD19 AFD20 AFD21
global rendvolum diesel wgiro pafr afd
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17
p18 p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43
p44 p45 p46 p47 p48 p49 p50
global v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17
v18 v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43
v44 v45 v46 v47 v48 v49 v50
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
```

```
global incrementoT landa AFR AFD Q A B C D E F rhod


z=(AF21-AF1)/20;
pafr=AFR;
pafd=AFD;

if diesel==1
    AFD1=AF1;
    AFD=AFD1;
    ciclo;
    Q1=Q;
    W1=WJ;
    AFD2=AFD1+z;
    AFD=AFD2;
    ciclo;
    Q2=Q;
    W2=WJ;
    AFD3=AFD2+z;
    AFD=AFD3;
    ciclo;
    Q3=Q;
    W3=WJ;
    AFD4=AFD3+z;
    AFD=AFD4;
    ciclo;
    Q4=Q;
    W4=WJ;
    AFD5=AFD4+z;
    AFD=AFD5;
    ciclo;
    Q5=Q;
    W5=WJ;
    AFD6=AFD5+z;
    AFD=AFD6;
    ciclo;
    Q6=Q;
    W6=WJ;
    AFD7=AFD6+z;
    AFD=AFD7;
    ciclo;
    Q7=Q;
    W7=WJ;
    AFD8=AFD7+z;
    AFD=AFD8;
    ciclo;
    Q8=Q;
    W8=WJ;
    AFD9=AFD8+z;
    AFD=AFD9;
    ciclo;
    Q9=Q;
    W9=WJ;
    AFD10=AFD9+z;
    AFD=AFD10;
    ciclo;
    Q10=Q;
    W10=WJ;
```

```
AFD11=AFD10+z;
AFD=AFD11;
ciclo;
Q11=Q;
W11=WJ;
AFD12=AFD11+z;
AFD=AFD12;
ciclo;
Q12=Q;
W12=WJ;
AFD13=AFD12+z;
AFD=AFD13;
ciclo;
Q13=Q;
W13=WJ;
AFD14=AFD13+z;
AFD=AFD14;
ciclo;
Q14=Q;
W14=WJ;
AFD15=AFD14+z;
AFD=AFD15;
ciclo;
Q15=Q;
W15=WJ;
AFD16=AFD15+z;
AFD=AFD16;
ciclo;
Q16=Q;
W16=WJ;
AFD17=AFD16+z;
AFD=AFD17;
ciclo;
Q17=Q;
W17=WJ;
AFD18=AFD17+z;
AFD=AFD18;
ciclo;
Q18=Q;
W18=WJ;
AFD19=AFD18+z;
AFD=AFD19;
ciclo;
Q19=Q;
W19=WJ;
AFD20=AFD19+z;
AFD=AFD20;
ciclo;
Q20=Q;
W20=WJ;
AFD21=AFD20+z;
AFD=AFD21;
ciclo;
Q21=Q;
W21=WJ;
AFD=pafd;
%%representación del gráfico
```

```
W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
9,W20,W21];

Qc=[Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11,Q12,Q13,Q14,Q15,Q16,Q17,Q18,Q
19,Q20,Q21];

AF=[AFD1,AFD2,AFD3,AFD4,AFD5,AFD6,AFD7,AFD8,AFD9,AFD10,AFD11,AFD12,A
FD13,AFD14,AFD15,AFD16,AFD17,AFD18,AFD19,AFD20,AFD21];
    plot(AF,W,'m:')
    xlabel('Relación aire/combustible (AFD)')
    ylabel('Trabajo obtenido(J)')
    grid
    title('TRABAJO EN FUNCIÓN DE LA RELACIÓN AIRE/COMBUSTIBLE')

elseif otto==1
    AFR1=AF1;
    AFR=AFR1;
    ciclo;
    Q1=Q;
    W1=WJ;
    AFR2=AFR1+z;
    AFR=AFR2;
    ciclo;
    Q2=Q;
    W2=WJ;
    AFR3=AFR2+z;
    AFR=AFR3;
    ciclo;
    Q3=Q;
    W3=WJ;
    AFR4=AFR3+z;
    AFR=AFR4;
    ciclo;
    Q4=Q;
    W4=WJ;
    AFR5=AFR4+z;
    AFR=AFR5;
    ciclo;
    Q5=Q;
    W5=WJ;
    AFR6=AFR5+z;
    AFR=AFR6;
    ciclo;
    Q6=Q;
    W6=WJ;
    AFR7=AFR6+z;
    AFR=AFR7;
    ciclo;
    Q7=Q;
    W7=WJ;
    AFR8=AFR7+z;
    AFR=AFR8;
    ciclo;
    Q8=Q;
    W8=WJ;
    AFR9=AFR8+z;
    AFR=AFR9;
```

```
ciclo;
Q9=Q;
W9=WJ;
AFR10=AFR9+z;
AFR=AFR10;
ciclo;
Q10=Q;
W10=WJ;
AFR11=AFR10+z;
AFR=AFR11;
ciclo;
Q11=Q;
W11=WJ;
AFR12=AFR11+z;
AFR=AFR12;
ciclo;
Q12=Q;
W12=WJ;
AFR13=AFR12+z;
AFR=AFR13;
ciclo;
Q13=Q;
W13=WJ;
AFR14=AFR13+z;
AFR=AFR14;
ciclo;
Q14=Q;
W14=WJ;
AFR15=AFR14+z;
AFR=AFR15;
ciclo;
Q15=Q;
W15=WJ;
AFR16=AFR15+z;
AFR=AFR16;
ciclo;
Q16=Q;
W16=WJ;
AFR17=AFR16+z;
AFR=AFR17;
ciclo;
Q17=Q;
W17=WJ;
AFR18=AFR17+z;
AFR=AFR18;
ciclo;
Q18=Q;
W18=WJ;
AFR19=AFR18+z;
AFR=AFR19;
ciclo;
Q19=Q;
W19=WJ;
AFR20=AFR19+z;
AFR=AFR20;
ciclo;
Q20=Q;
W20=WJ;
```

```
    AFR21=AFR20+z;
    AFR=AFR21;
    ciclo;
    Q21=Q;
    W21=WJ;
    AFR=pafr;
    %representación del gráfico

W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
9,W20,W21];

Qc=[Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11,Q12,Q13,Q14,Q15,Q16,Q17,Q18,Q
19,Q20,Q21];

AF=[AFR1,AFR2,AFR3,AFR4,AFR5,AFR6,AFR7,AFR8,AFR9,AFR10,AFR11,AFR12,A
FR13,AFR14,AFR15,AFR16,AFR17,AFR18,AFR19,AFR20,AFR21];
    plot(AF,W,'m:')
    xlabel('Relación aire/combustible (AFR)')
    ylabel('Trabajo obtenido(J)')
    grid
    title('TRABAJO EN FUNCIÓN DE LA RELACIÓN AIRE/COMBUSTIBLE')

end



% --- Executes on button press in wgiroge.
function wgiroge_Callback(hObject, eventdata, handles)
% hObject    handle to wgiroge (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto wwgiro
global WGIRO wgiro1 wgiro2 wgiro3 wgiro4 wgiro5 wgiro6 wgiro7 wgiro8
wgiro9 wgiro10 wgiro11 wgiro12 wgiro13 wgiro14 wgiro15 wgiro16
wgiro17 wgiro18 wgiro19 wgiro20 wgiro21
global GE ge1 ge2 ge3 ge4 ge5 ge6 ge7 ge8 ge9 ge10 ge11 ge12 ge13
ge14 ge15 ge16 ge17 ge18 ge19 ge20 ge21
global rendvolum diesel wgiro pz rhod gespecifico
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p13 p14 p15 p16 p17
p18 p19 p20 p30 p31 p32 p33 p34 p35 p36 p37 p38 p39 p40 p41 p42 p43
p44 p45 p46 p47 p48 p49 p50
global v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17
v18 v19 v20 v30 v31 v32 v33 v34 v35 v36 v37 v38 v39 v40 v41 v42 v43
v44 v45 v46 v47 v48 v49 v50
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F AF1 AFR21 t


z=(wgiro21-wgiro1)/20;
wwgiro=wgiro;


    wgiro=wgiro1;
    ciclo;
```

```
ge1=gespecifico;
wgiro2=wgiro1+z;
wgiro=wgiro2;
ciclo;
ge2=gespecifico;
wgiro3=wgiro2+z;
wgiro=wgiro3;
ciclo;
ge3=gespecifico;
wgiro4=wgiro3+z;
wgiro=wgiro4;
ciclo;
ge4=gespecifico;
wgiro5=wgiro4+z;
wgiro=wgiro5;
ciclo;
ge5=gespecifico;
wgiro6=wgiro5+z;
wgiro=wgiro6;
ciclo;
ge6=gespecifico;
wgiro7=wgiro6+z;
wgiro=wgiro7;
ciclo;
ge7=gespecifico;
wgiro8=wgiro7+z;
wgiro=wgiro8;
ciclo;
ge8=gespecifico;
wgiro9=wgiro8+z;
wgiro=wgiro9;
ciclo
ge9=gespecifico;
wgiro10=wgiro9+z;
ciclo;
ge10=gespecifico;
wgiro11=wgiro10+z;
wgiro=wgiro11;
ciclo;
ge11=gespecifico;
wgiro12=wgiro11+z;
wgiro=wgiro12;
ciclo;
ge12=gespecifico;
wgiro13=wgiro12+z;
wgiro=wgiro13;
ciclo;
ge13=gespecifico;
wgiro14=wgiro13+z;
wgiro=wgiro14;
ciclo;
ge14=gespecifico;
wgiro15=wgiro14+z;
wgiro=wgiro15;
ciclo;
ge15=gespecifico;
wgiro16=wgiro15+z;
wgiro=wgiro16;
```

```
    ciclo;
    ge16=gespecifico;
    wgiro17=wgiro16+z;
    wgiro=wgiro17;
    ciclo;
    ge17=gespecifico;
    wgiro18=wgiro17+z;
    wgiro=wgiro18;
    ciclo
    ge18=gespecifico;
    wgiro19=wgiro18+z;
    wgiro=wgiro19;
    ciclo;
    ge19=gespecifico;
    wgiro20=wgiro19+z;
    wgiro=wgiro20;
    ciclo;
    ge20=gespecifico;
    wgiro=wgiro21;
    ciclo;
    ge21=gespecifico;

  %representación del gráfico

WGIRO=[wgiro1,wgiro2,wgiro3,wgiro4,wgiro5,wgiro6,wgiro7,wgiro8,wgiro
9,wgiro10,wgiro11,wgiro12,wgiro13,wgiro14,wgiro15,wgiro16,wgiro17,wg
iro18,wgiro19,wgiro20,wgiro21];

GE=[ge1,ge2,ge3,ge4,ge5,ge6,ge7,ge8,ge9,ge10,ge11,ge12,ge13,ge14,ge1
5,ge16,ge17,ge18,ge19,ge20,ge21];
    %FALTA ESTO "subplot(1,2,1)"
    plot(WGIRO,GE,'r:')
    %para configurar la dimensión de los ejes
    xlabel('Velocidad de giro del motor (rpm)')
    ylabel('Consumo específico gasolina(g·(KW*h)e-01)')
    grid
    title('CONSUMO ESPECÍFICO EN FUNCIÓN DE LA VELOCIDAD DE GIRO DEL
MOTOR')


wgiro=wwgiro;

if otto==1
AFR1=14.7;
AFR21=19.6;
end

if otto==1
    pz=AFR;
    t=(AFR21-AFR1)/20;
    AFR=AFR1;
    ciclo;
    ge1=gespecifico;
    AFR2=AFR1+t;
    AFR=AFR2;
    ciclo;
    ge2=gespecifico;
    AFR3=AFR2+t;
```

```
AFR=AFR3;
ciclo;
ge3=gespecifico;
AFR4=AFR3+t;
AFR=AFR4;
ciclo;
ge4=gespecifico;
AFR5=AFR4+t;
AFR=AFR5;
ciclo;
ge5=gespecifico;
AFR6=AFR5+t;
AFR=AFR6;
ciclo;
ge6=gespecifico;
AFR7=AFR6+t;
AFR=AFR7;
ciclo;
ge7=gespecifico;
AFR8=AFR7+t;
AFR=AFR8;
ciclo;
ge8=gespecifico;
AFR9=AFR8+t;
AFR=AFR9;
ciclo
ge9=gespecifico;
AFR10=AFR9+t;
AFR=AFR10;
ciclo;
ge10=gespecifico;
AFR11=AFR10+t;
AFR=AFR11;
ciclo;
ge11=gespecifico;
AFR12=AFR11+t;
AFR=AFR12;
ciclo;
ge12=gespecifico;
AFR13=AFR12+t;
AFR=AFR13;
ciclo;
ge13=gespecifico;
AFR14=AFR13+t;
AFR=AFR14;
ciclo;
ge14=gespecifico;
AFR15=AFR14+t;
AFR=AFR15;
ciclo;
ge15=gespecifico;
AFR16=AFR15+t;
AFR=AFR16;
ciclo;
ge16=gespecifico;
AFR17=AFR16+t;
AFR=AFR17;
ciclo;
```

```
    ge17=gespecifico;
    AFR18=AFR17+t;
    AFR=AFR18;
    ciclo
    ge18=gespecifico;
    AFR19=AFR18+t;
    AFR=AFR19;
    ciclo;
    ge19=gespecifico;
    AFR20=AFR19+t;
    AFR=AFR20;
    ciclo;
    ge20=gespecifico;
    AFR=AFR21;
    ciclo;
    ge21=gespecifico;
    AFR=pz;


GE=[ge1,ge2,ge3,ge4,ge5,ge6,ge7,ge8,ge9,ge10,ge11,ge12,ge13,ge14,ge1
5,ge16,ge17,ge18,ge19,ge20,ge21];

FF=[AFR1,AFR2,AFR3,AFR4,AFR5,AFR6,AFR7,AFR8,AFR9,AFR10,AFR11,AFR12,A
FR13,AFR14,AFR15,AFR16,AFR17,AFR18,AFR19,AFR20,AFR21];
    FFF=FF/14.7;

    %Falta esto"subplot(1,2,2)"
    %FALTA ESTO"plot(FFF,GE,'r:')"
    %FALTA "xlabel('landa')"
    %FALTA "ylabel('Consumo específico gasolina(g·(KW*h)e-01)')"
    %FALTA"grid"
    %FALTA"title('consumo específico en funcion de landa')"
end



% --- Executes on button press in Wcil.
function Wcil_Callback(hObject, eventdata, handles)
% hObject    handle to Wcil (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto rhod
global rendvolum diesel wgiro hj
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global W W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14 W15 W16 W17
W18 W19 W20 W21
global V v01 v02 v03 v04 v05 v06 v07 v08 v09 v010 v011 v012 v013
v014 v015 v016 v017 v018 v019 v020 v021
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F AF1 AFR21

z=(v021-v01)/20;
v1=hj;

if diesel==1
    v1=v01;
```

```
ciclo;
W1=WJ;
v02=v01+z;
v1=v02;
ciclo;
W2=WJ;
v03=v02+z;
v1=v03;
ciclo;
W3=WJ;
v04=v03+z;
v1=v04;
ciclo;
W4=WJ;
v05=v04+z;
v1=v05;
ciclo;
W5=WJ;
v06=v05+z;
v1=v06;
ciclo;
W6=WJ;
v07=v06+z;
v1=v07;
ciclo;
W7=WJ;
v08=v07+z;
v1=v08;
ciclo;
W8=WJ;
v09=v08+z;
v1=v09;
ciclo;
W9=WJ;
v010=v09+z;
v1=v010;
ciclo;
W10=WJ;
v011=v010+z;
v1=v011;
ciclo;
W11=WJ;
v012=v011+z;
v1=v012;
ciclo;
W12=WJ;
v013=v012+z;
v1=v013;
ciclo;
W13=WJ;
v014=v013+z;
v1=v014;
ciclo;
W14=WJ;
v015=v014+z;
v1=v015;
ciclo;
W15=WJ;
```

```
        v016=v015+z;
        v1=v016;
        ciclo;
        W16=WJ;
        v017=v016+z;
        v1=v017;
        ciclo;
        W17=WJ;
        v018=v017+z;
        v1=v018;
        ciclo;
        W18=WJ;
        v019=v018+z;
        v1=v019;
        ciclo;
        W19=WJ;
        v020=v019+z;
        v1=v020;
        ciclo;
        W20=WJ;
        v021=v020+z;
        v1=v021;
        ciclo
        W21=WJ;
        v1=hj;
        %representación del gráfico

    W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
    9,W20,W21];

    V=[v01,v02,v03,v04,v05,v06,v07,v08,v09,v010,v011,v012,v013,v014,v015
    ,v016,v017,v018,v019,v020,v021];
        plot(V,W,'r:')
        %para configurar los ejes
        title('VARIACIÓN DEL TRABAJO EN FUNCIÓN DE LA CILINDRADA')
        xlabel('Cilindrada(L)')
        ylabel('Wciclo(J)')
        grid


    elseif otto==1
        v1=v01;
        ciclo;
        W1=WJ;
        v02=v01+z;
        v1=v02;
        ciclo;
        W2=WJ;
        v03=v02+z;
        v1=v03;
        ciclo;
        W3=WJ;
        v04=v03+z;
        v1=v04;
        ciclo;
        W4=WJ;
        v05=v04+z;
        v1=v05;
```

```
ciclo;
W5=WJ;
v06=v05+z;
v1=v06;
ciclo;
W6=WJ;
v07=v06+z;
v1=v07;
ciclo;
W7=WJ;
v08=v07+z;
v1=v08;
ciclo;
W8=WJ;
v09=v08+z;
v1=v09;
ciclo;
W9=WJ;
v010=v09+z;
v1=v010;
ciclo;
W10=WJ;
v011=v010+z;
v1=v011;
ciclo;
W11=WJ;
v012=v011+z;
v1=v012;
ciclo;
W12=WJ;
v013=v012+z;
v1=v013;
ciclo;
W13=WJ;
v014=v013+z;
v1=v014;
ciclo;
W14=WJ;
v015=v014+z;
v1=v015;
ciclo;
W15=WJ;
v016=v015+z;
v1=v016;
ciclo;
W16=WJ;
v017=v016+z;
v1=v017;
ciclo;
W17=WJ;
v018=v017+z;
v1=v018;
ciclo;
W18=WJ;
v019=v018+z;
v1=v019;
ciclo;
W19=WJ;
```

```
    v020=v019+z;
    v1=v020;
    ciclo;
    W20=WJ;
    v021=v020+z;
    v1=v021;
    ciclo
    W21=WJ;
    v1=hj;
    %representación del gráfico

W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
9,W20,W21];

V=[v01,v02,v03,v04,v05,v06,v07,v08,v09,v010,v011,v012,v013,v014,v015
,v016,v017,v018,v019,v020,v021];
    plot(V,W,'r:')
    %para configurar los ejes
    title('VARIACIÓN DEL TRABAJO EN FUNCIÓN DE LA CILINDRADA')
    xlabel('Cilindrada(L)')
    ylabel('Wciclo(J)')
    grid


end


% --- Executes on button press in rhorend.
function rhorend_Callback(hObject, eventdata, handles)
% hObject    handle to rhorend (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto
global rendvolum diesel wgiro eg ed
global ren rendtermo1 rendtermo2 rendtermo3 rendtermo4 rendtermo5
rendtermo6 rendtermo7 rendtermo8 rendtermo9 rendtermo10 rendtermo11
rendtermo12 rendtermo13 rendtermo14 rendtermo15 rendtermo16
rendtermo17 rendtermo18 rendtermo19 rendtermo20 rendtermo21
global AF rho1 rho2 rho3 rho4 rho5 rho6 rho7 rho8 rho9 rho10 rho11
rho12 rho13 rho14 rho15 rho16 rho17 rho18 rho19 rho20 rho21
global V v01 v02 v03 v04 v05 v06 v07 v08 v09 v010 v011 v012 v013
v014 v015 v016 v017 v018 v019 v020 v021
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F AF1 AFR21 rho21 rho1
k cpotto cpdiesel cvotto cvdiesel potto pdiesel v1

z=(rho21-rho1)/20;
rhod=ed;
rhog=eg;

if diesel==1
    rhod=rho1;
    ciclo;
    rendtermo1=rendtermoint;
    rho2=rho1+z;
    rhod=rho2;
```

```
ciclo;
rendtermo2=rendtermoint;
rho3=rho2+z;
rhod=rho3;
ciclo;
rendtermo3=rendtermoint;
rho4=rho3+z;
rhod=rho4;
ciclo;
rendtermo4=rendtermoint;
rho5=rho4+z;
rhod=rho5;
ciclo;
rendtermo5=rendtermoint;
rho6=rho5+z;
rhod=rho6;
ciclo;
rendtermo6=rendtermoint;
rho7=rho6+z;
rhod=rho7;
ciclo;
rendtermo7=rendtermoint;
rho8=rho7+z;
rhod=rho8;
ciclo;
rendtermo8=rendtermoint;
rho9=rho8+z;
rhod=rho9;
ciclo;
rendtermo9=rendtermoint;
rho10=rho9+z;
rhod=rho10;
ciclo;
rendtermo10=rendtermoint;
rho11=rho10+z;
rhod=rho11;
ciclo;
rendtermo11=rendtermoint;
rho12=rho11+z;
rhod=rho12;
ciclo;
rendtermo12=rendtermoint;
rho13=rho12+z;
rhod=rho13;
ciclo;
rendtermo13=rendtermoint;
rho14=rho13+z;
rhod=rho14;
ciclo;
rendtermo14=rendtermoint;
rho15=rho14+z;
rhod=rho15;
ciclo;
rendtermo15=rendtermoint;
rho16=rho15+z;
rhod=rho16;
ciclo;
rendtermo16=rendtermoint;
```

```
        rho17=rho16+z;
        rhod=rho17;
        ciclo;
        rendtermo17=rendtermoint;
        rho18=rho17+z;
        rhod=rho18;
        ciclo;
        rendtermo18=rendtermoint;
        rho19=rho18+z;
        rhod=rho19;
        ciclo;
        rendtermo19=rendtermoint;
        rho20=rho19+z;
        rhod=rho20;
        ciclo;
        rendtermo20=rendtermoint;
        rho21=rho20+z;
        rhod=rho21;
        ciclo;
        rendtermo21=rendtermoint;
        rhod=ed;
        %representación del gráfico

    ren=[rendtermo1,rendtermo2,rendtermo3,rendtermo4,rendtermo5,rendterm
    o6,rendtermo7,rendtermo8,rendtermo9,rendtermo10,rendtermo11,rendterm
    o12,rendtermo13,rendtermo14,rendtermo15,rendtermo16,rendtermo17,rend
    termo18,rendtermo19,rendtermo20,rendtermo21];

    AF=[rho1,rho2,rho3,rho4,rho5,rho6,rho7,rho8,rho9,rho10,rho11,rho12,r
    ho13,rho14,rho15,rho16,rho17,rho18,rho19,rho20,rho21];
        plot(AF,ren,'m:')
        xlabel('Relación de compresión')
        ylabel('Rendimiento termodinámico')
        grid
        title('RENDIMIENTO TERMODINÁMICO EN FUNCIÓN DE LA RELACIÓN DE
    COMPRESIÓN')

    elseif otto==1
        rhog=rho1;
        ciclo;
        rendtermo1=rendtermoint;
        rho2=rho1+z;
        rhog=rho2;
        ciclo;
        rendtermo2=rendtermoint;
        rho3=rho2+z;
        rhog=rho3;
        ciclo;
        rendtermo3=rendtermoint;
        rho4=rho3+z;
        rhog=rho4;
        ciclo;
        rendtermo4=rendtermoint;
        rho5=rho4+z;
        rhog=rho5;
        ciclo;
        rendtermo5=rendtermoint;
        rho6=rho5+z;
```

```
rhog=rho6;
ciclo;
rendtermo6=rendtermoint;
rho7=rho6+z;
rhog=rho7;
ciclo;
rendtermo7=rendtermoint;
rho8=rho7+z;
rhog=rho8;
ciclo;
rendtermo8=rendtermoint;
rho9=rho8+z;
rhog=rho9;
ciclo;
rendtermo9=rendtermoint;
rho10=rho9+z;
rhog=rho10;
ciclo;
rendtermo10=rendtermoint;
rho11=rho10+z;
rhog=rho11;
ciclo;
rendtermo11=rendtermoint;
rho12=rho11+z;
rhog=rho12;
ciclo;
rendtermo12=rendtermoint;
rho13=rho12+z;
rhog=rho13;
ciclo;
rendtermo13=rendtermoint;
rho14=rho13+z;
rhog=rho14;
ciclo;
rendtermo14=rendtermoint;
rho15=rho14+z;
rhog=rho15;
ciclo;
rendtermo15=rendtermoint;
rho16=rho15+z;
rhog=rho16;
ciclo;
rendtermo16=rendtermoint;
rho17=rho16+z;
rhog=rho17;
ciclo;
rendtermo17=rendtermoint;
rho18=rho17+z;
rhog=rho18;
ciclo;
rendtermo18=rendtermoint;
rho19=rho18+z;
rhog=rho19;
ciclo;
rendtermo19=rendtermoint;
rho20=rho19+z;
rhog=rho20;
ciclo;
```

```
    rendtermo20=rendtermoint;
    rho21=rho20+z;
    rhog=rho21;
    ciclo;
    rendtermo21=rendtermoint;
    rhog=eg;
    %representación del gráfico

ren=[rendtermo1,rendtermo2,rendtermo3,rendtermo4,rendtermo5,rendterm
o6,rendtermo7,rendtermo8,rendtermo9,rendtermo10,rendtermo11,rendterm
o12,rendtermo13,rendtermo14,rendtermo15,rendtermo16,rendtermo17,rend
termo18,rendtermo19,rendtermo20,rendtermo21];

AF=[rho1,rho2,rho3,rho4,rho5,rho6,rho7,rho8,rho9,rho10,rho11,rho12,r
ho13,rho14,rho15,rho16,rho17,rho18,rho19,rho20,rho21];
    plot(AF,ren,'m:')
    xlabel('Relación de compresión')
    ylabel('Rendimiento termodinámico')
    grid
    title('RENDIMIENTO TERMODINÁMICO EN FUNCIÓN DE LA RELACIÓN DE
COMPRESIÓN')
end



% --- Executes on button press in parwgiro.
function parwgiro_Callback(hObject, eventdata, handles)
% hObject    handle to parwgiro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto rhod
global rendvolum diesel wgiro par pw
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global Wgiro wgiro1 wgiro2 wgiro3 wgiro4 wgiro5 wgiro6 wgiro7 wgiro8
wgiro9 wgiro10 wgiro11 wgiro12 wgiro13 wgiro14 wgiro15 wgiro16
wgiro17 wgiro18 wgiro19 wgiro20 wgiro21
global ren rendtermo1 rendtermo2 rendtermo3 rendtermo4 rendtermo5
rendtermo6 rendtermo7 rendtermo8 rendtermo9 rendtermo10 rendtermo11
rendtermo12 rendtermo13 rendtermo14 rendtermo15 rendtermo16
rendtermo17 rendtermo18 rendtermo19 rendtermo20 rendtermo21
global PAR par1 par2 par3 par4 par5 par6 par7 par8 par9 par10 par11
par12 par13 par14 par15 par16 par17 par18 par19 par20 par21
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F AF1 AFR21 v1


z=(wgiro21-wgiro1)/20;
pw=wgiro;

    wgiro=wgiro1;
    ciclo;
    par1=WJ;
    rendtermo1=rendtermoint;
    wgiro2=wgiro1+z;
    wgiro=wgiro2;
    ciclo;
```

```
par2=WJ;
rendtermo2=rendtermoint;
wgiro3=wgiro2+z;
wgiro=wgiro3;
ciclo;
par3=WJ;
rendtermo3=rendtermoint;
wgiro4=wgiro3+z;
wgiro=wgiro4;
ciclo;
par4=WJ;
rendtermo4=rendtermoint;
wgiro5=wgiro4+z;
wgiro=wgiro5;
ciclo;
par5=WJ;
rendtermo5=rendtermoint;
wgiro6=wgiro5+z;
wgiro=wgiro6;
ciclo;
par6=WJ;
rendtermo6=rendtermoint;
wgiro7=wgiro6+z;
wgiro=wgiro7;
ciclo;
par7=WJ;
rendtermo7=rendtermoint;
wgiro8=wgiro7+z;
wgiro=wgiro8;
ciclo;
par8=WJ;
rendtermo8=rendtermoint;
wgiro9=wgiro8+z;
wgiro=wgiro9;
ciclo;
par9=WJ;
rendtermo9=rendtermoint;
wgiro10=wgiro9+z;
wgiro=wgiro10;
ciclo;
par10=WJ;
rendtermo10=rendtermoint;
wgiro11=wgiro10+z;
wgiro=wgiro11;
ciclo;
par11=WJ;
rendtermo11=rendtermoint;
wgiro12=wgiro11+z;
wgiro=wgiro12;
ciclo;
par12=WJ;
rendtermo12=rendtermoint;
wgiro13=wgiro12+z;
wgiro=wgiro13;
ciclo;
par13=WJ;
rendtermo13=rendtermoint;
wgiro14=wgiro13+z;
```

```
    wgiro=wgiro14;
    ciclo;
    par14=WJ;
    rendtermo14=rendtermoint;
    wgiro15=wgiro14+z;
    wgiro=wgiro15;
    ciclo;
    par15=WJ;
    rendtermo15=rendtermoint;
    wgiro16=wgiro15+z;
    wgiro=wgiro16;
    ciclo;
    par16=WJ;
    rendtermo16=rendtermoint;
    wgiro17=wgiro16+z;
    wgiro=wgiro17;
    ciclo;
    par17=WJ;
    rendtermo17=rendtermoint;
    wgiro18=wgiro17+z;
    wgiro=wgiro18;
    ciclo;
    par18=WJ;
    rendtermo18=rendtermoint;
    wgiro19=wgiro18+z;
    wgiro=wgiro19;
    ciclo;
    par19=WJ;
    rendtermo19=rendtermoint;
    wgiro20=wgiro19+z;
    wgiro=wgiro20;
    ciclo;
    par20=WJ;
    rendtermo20=rendtermoint;
    wgiro21=wgiro20+z;
    wgiro=wgiro21;
    ciclo;
    par21=WJ;
    rendtermo21=rendtermoint;
    wgiro=pw;
    %representación del gráfico

Wgiro=[wgiro1,wgiro2,wgiro3,wgiro4,wgiro5,wgiro6,wgiro7,wgiro8,wgiro
9,wgiro10,wgiro11,wgiro12,wgiro13,wgiro14,wgiro15,wgiro16,wgiro17,wg
iro18,wgiro19,wgiro20,wgiro21];

ren=[rendtermo1,rendtermo2,rendtermo3,rendtermo4,rendtermo5,rendterm
o6,rendtermo7,rendtermo8,rendtermo9,rendtermo10,rendtermo11,rendterm
o12,rendtermo13,rendtermo14,rendtermo15,rendtermo16,rendtermo17,rend
termo18,rendtermo19,rendtermo20,rendtermo21];

PAR=[par1,par2,par3,par4,par5,par6,par7,par8,par9,par10,par11,par12,
par13,par14,par15,par16,par17,par18,par19,par20,par21];
    plot(Wgiro,PAR,'m:')
    xlabel('Velocidad de giro del motor (rpm)')
    ylabel('Par motor (N·m)')
    grid
    title('PAR MOTOR EN FUNCIÓN DE LA VELOCIDAD DE GIRO DEL MOTOR')
```

```
% --- Executes on button press in Rcil.
function Rcil_Callback(hObject, eventdata, handles)
% hObject    handle to Rcil (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global z diesel otto rhod po
global rendvolum diesel wgiro
global rho rhog k cpotto cpdiesel cvotto cvdiesel pdiesel potto
global W W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14 W15 W16 W17
W18 W19 W20 W21
global V v01 v02 v03 v04 v05 v06 v07 v08 v09 v010 v011 v012 v013
v014 v015 v016 v017 v018 v019 v020 v021
global ctecompresion x rendgasolina n R T1 m_aire m_mol_aire
m_combustible PCIgasolina PCIdiesel masamolardiesel
global incrementoT landa AFR AFD Q A B C D E F AF1 AFR21 v1


z=(v021-v01)/20;
po=v1;

if diesel==1
    v1=v01;
    ciclo;
    W1=rendtermoint;
    v02=v01+z;
    v1=v02;
    ciclo;
    W2=rendtermoint;
    v03=v02+z;
    v1=v03;
    ciclo;
    W3=rendtermoint;
    v04=v03+z;
    v1=v04;
    ciclo;
    W4=rendtermoint;
    v05=v04+z;
    v1=v05;
    ciclo;
    W5=rendtermoint;
    v06=v05+z;
    v1=v06;
    ciclo;
    W6=rendtermoint;
    v07=v06+z;
    v1=v07;
    ciclo;
    W7=rendtermoint;
    v08=v07+z;
    v1=v08;
    ciclo;
    W8=rendtermoint;
    v09=v08+z;
    v1=v09;
```

```
    ciclo;
    W9=rendtermoint;
    v010=v09+z;
    v1=v010;
    ciclo;
    W10=rendtermoint;
    v011=v010+z;
    v1=v011;
    ciclo;
    W11=rendtermoint;
    v012=v011+z;
    v1=v012;
    ciclo;
    W12=rendtermoint;
    v013=v012+z;
    v1=v013;
    ciclo;
    W13=rendtermoint;
    v014=v013+z;
    v1=v014;
    ciclo;
    W14=rendtermoint;
    v015=v014+z;
    v1=v015;
    ciclo;
    W15=rendtermoint;
    v016=v015+z;
    v1=v016;
    ciclo;
    W16=rendtermoint;
    v017=v016+z;
    v1=v017;
    ciclo;
    W17=rendtermoint;
    v018=v017+z;
    v1=v018;
    ciclo;
    W18=rendtermoint;
    v019=v018+z;
    v1=v019;
    ciclo;
    W19=rendtermoint;
    v020=v019+z;
    v1=v020;
    ciclo;
    W20=rendtermoint;
    v021=v020+z;
    v1=v021;
    ciclo
    W21=rendtermoint;
    v1=po;
    %representación del gráfico

W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
9,W20,W21];

V=[v01,v02,v03,v04,v05,v06,v07,v08,v09,v010,v011,v012,v013,v014,v015
,v016,v017,v018,v019,v020,v021];
```

```
        plot(V,W,'r:')
        %para configurar los ejes
        title('RENDIMIENTO TERMODINÁMICO EN FUNCIÓN DE LA CILINDRADA')
        xlabel('Cilindrada(L)')
        ylabel('Rendimiento termodinámico')
        grid


    elseif otto==1
        v1=v01;
        ciclo;
        W1=rendtermoint;
        v02=v01+z;
        v1=v02;
        ciclo;
        W2=rendtermoint;
        v03=v02+z;
        v1=v03;
        ciclo;
        W3=rendtermoint;
        v04=v03+z;
        v1=v04;
        ciclo;
        W4=rendtermoint;
        v05=v04+z;
        v1=v05;
        ciclo;
        W5=rendtermoint;
        v06=v05+z;
        v1=v06;
        ciclo;
        W6=rendtermoint;
        v07=v06+z;
        v1=v07;
        ciclo;
        W7=rendtermoint;
        v08=v07+z;
        v1=v08;
        ciclo;
        W8=rendtermoint;
        v09=v08+z;
        v1=v09;
        ciclo;
        W9=rendtermoint;
        v010=v09+z;
        v1=v010;
        ciclo;
        W10=rendtermoint;
        v011=v010+z;
        v1=v011;
        ciclo;
        W11=rendtermoint;
        v012=v011+z;
        v1=v012;
        ciclo;
        W12=rendtermoint;
        v013=v012+z;
        v1=v013;
```

```
        ciclo;
        W13=rendtermoint;
        v014=v013+z;
        v1=v014;
        ciclo;
        W14=rendtermoint;
        v015=v014+z;
        v1=v015;
        ciclo;
        W15=rendtermoint;
        v016=v015+z;
        v1=v016;
        ciclo;
        W16=rendtermoint;
        v017=v016+z;
        v1=v017;
        ciclo;
        W17=rendtermoint;
        v018=v017+z;
        v1=v018;
        ciclo;
        W18=rendtermoint;
        v019=v018+z;
        v1=v019;
        ciclo;
        W19=rendtermoint;
        v020=v019+z;
        v1=v020;
        ciclo;
        W20=rendtermoint;
        v021=v020+z;
        v1=v021;
        ciclo
        W21=rendtermoint;
        v1=po;
        %representación del gráfico

    W=[W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,W11,W12,W13,W14,W15,W16,W17,W18,W1
    9,W20,W21];

    V=[v01,v02,v03,v04,v05,v06,v07,v08,v09,v010,v011,v012,v013,v014,v015
    ,v016,v017,v018,v019,v020,v021];
        plot(V,W,'r:')
        %para configurar los ejes
        title('RENDIMIENTO TERMODINÁMICO EN FUNCIÓN DE LA CILINDRADA')
        xlabel('Cilindrada(L)')
        ylabel('Rendimiento termodinámico')
        grid


    end




    % --- Executes on button press in mostrar.
```

```
function mostrar_Callback(hObject, eventdata, handles)
% hObject    handle to mostrar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global WJ rendtermoint PutilW pme Q gespecifico
set(handles.edit42,'string',num2str(WJ));
set(handles.edit43,'string',num2str(rendtermoint));
set(handles.edit44,'string',subs(PutilW));
set(handles.edit47,'string',num2str(Q));
set(handles.edit45,'string',num2str(pme));
set(handles.edit46,'string',num2str(gespecifico));


function edit42_Callback(hObject, eventdata, handles)
% hObject    handle to edit42 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit42 as text
%        str2double(get(hObject,'String')) returns contents of
edit42 as a double


% --- Executes during object creation, after setting all properties.
function edit42_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit42 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end




function edit43_Callback(hObject, eventdata, handles)
% hObject    handle to edit43 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit43 as text
%        str2double(get(hObject,'String')) returns contents of
edit43 as a double


% --- Executes during object creation, after setting all properties.
function edit43_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit43 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function edit44_Callback(hObject, eventdata, handles)
% hObject    handle to edit44 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit44 as text
%        str2double(get(hObject,'String')) returns contents of
edit44 as a double


% --- Executes during object creation, after setting all properties.
function edit44_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit44 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end







% --- Executes on button press in radiobutton26.
function radiobutton26_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton26 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton26


% --- Executes on button press in radiobutton25.
```

```
function radiobutton25_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton25 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton25


% --- Executes on button press in radiobutton24.
function radiobutton24_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton24


% --- Executes on button press in A.
function A_Callback(hObject, eventdata, handles)
% hObject    handle to A (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of A

global A B C
A=1;
B=0;
C=0;

% --- Executes on button press in B1.
function B1_Callback(hObject, eventdata, handles)
% hObject    handle to B1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of B1

global A B C
A=0;
B=1;
C=0;


% --- Executes on button press in C.
function C_Callback(hObject, eventdata, handles)
% hObject    handle to C (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of C

global A B C
A=0;
B=0;
C=1;
```

```
function edit45_Callback(hObject, eventdata, handles)
% hObject    handle to edit45 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit45 as text
%        str2double(get(hObject,'String')) returns contents of
edit45 as a double


% --- Executes during object creation, after setting all properties.
function edit45_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit45 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end



function edit46_Callback(hObject, eventdata, handles)
% hObject    handle to edit46 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit46 as text
%        str2double(get(hObject,'String')) returns contents of
edit46 as a double


% --- Executes during object creation, after setting all properties.
function edit46_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit46 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));
end
```

```
function edit47_Callback(hObject, eventdata, handles)
% hObject    handle to edit47 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit47 as text
%        str2double(get(hObject,'String')) returns contents of
edit47 as a double


% --- Executes during object creation, after setting all properties.
function edit47_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit47 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor
'));

end
```