

3. HERRAMIENTAS COMPUTACIONALES

3.1 Internal Combustion Engine Simulator

ICESym es un simulador de motores de combustión interna que utiliza modelos 0D para la cámara de combustión y 1D para el flujo a través del sistema de intercambio de gases. Esta combinación permite evaluar la *performance* de un motor a un costo computacional relativamente bajo; además la implementación de entrada y salida de datos facilita utilizar el simulador como una *caja negra*. Esta característica permite incluir al simulador en un *script* como una función, a la cual se le otorga un conjunto de parámetros de entrada y devuelve los resultados de la simulación en un formato que permite la lectura y evaluación de los mismos.

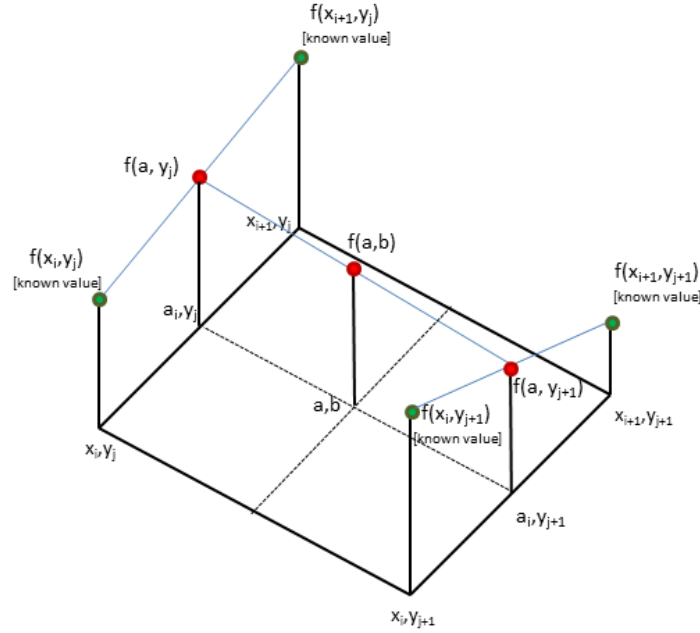
ICESym contiene en su código las rutinas necesarias para simular el ciclo operativo y la geometría del MRCVC. Se realizaron modificaciones menores para facilitar la ejecución en conjunto con el optimizador, algunas de estas modificaciones fueron:

1. Modificar el formato de los archivos de salida, con el fin de reducir el tamaño de los mismos, facilitar la lectura y el procesamiento de datos.
2. Incluir una opción para elegir entre un modelo de C_D de una o dos variables.
3. Modificar el área de referencia, ver Ecuación 3.2.
4. Agregar un esquema de interpolación bilineal que permita trabajar con el modelo de C_D de dos variables.

3.2 Modificaciones a ICESym

3.2.1 Flujo a Través de los Puertos

Se introdujo una opción para poder ejecutar ICESym con un modelo del coeficiente de descarga que dependa de dos variables: diferencia de presión y *alzada* o apertura del puerto, $C_D = f(lv; \Delta P)$. Esto significó agregar un *switch* en el código que permita seleccionar entre un modelo de una o dos variables, con el agregado

Figura 3.1. Interpolación bilineal¹

de las instrucciones de lectura de datos y armado de un arreglo bidimensional que contiene los valores del mapa de C_D en un orden dado. Con esto se construye un mapa del coeficiente de descarga de la forma $C_D = f(lv, \Delta P)$, que se utiliza para calcular el área efectiva del puerto.

Independientemente de la cantidad de variables que formen parte del coeficiente de descarga, a ICESym se introduce un vector para el caso 1D y matriz para el caso 2D. El esquema de interpolación bilineal implementado requiere de una malla rectangular. Se reutilizó el código existente para el caso 1D y se realiza una interpolación lineal entre dos valores en planos con datos conocidos, como se ve en la Figura 3.1. Si bien hay otros métodos de interpolación para estimar el valor de C_D a partir de una nube de puntos, este método es sencillo y da resultados satisfactorios. En la Figura 3.2 se muestra a modo de ejemplo el error obtenido con este método para interpolar una función de prueba $f = \sin(\sqrt{x^2 + y^2})$.

La malla rectangular requerida para la interpolación bilineal del mapa de C_D se realizó a partir de los valores resultantes de las flujometrías con *OpenFOAM* (The

¹<https://stackoverflow.com/questions/8808996/bilinear-interpolation-to-enlarge-bitmap-images>

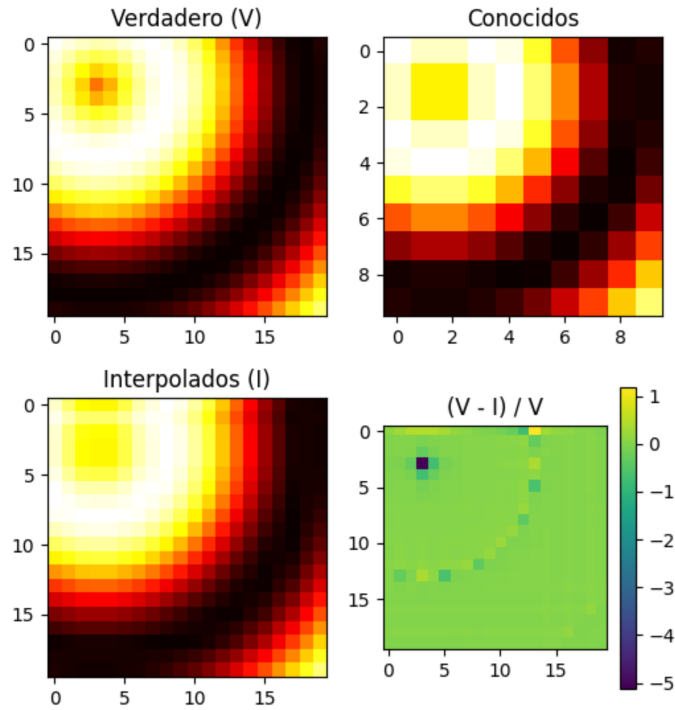


Figura 3.2. Interpolación bilineal de $\sin(\sqrt{x^2 + y^2})$

[OpenFOAM Foundation, 2011–2024](#)). Debido al costo computacional que requieren las flujometrías, solo una cantidad reducida de puntos se obtendrá con este método. Se tiene como punto de partida una malla no rectangular, por lo que se utiliza un método intermedio para obtener una matriz de puntos que pueda ser leída por la interpolación bilineal.

Se probaron dos métodos para realizar esta interpolación, el método del punto más cercano (MC) y la interpolación por la suma de la inversa de la distancia o IDW por sus siglas en inglés (*Inverse Distance Weighting*). Estos se combinan con métodos de suavizado de promedio móvil con los S valores más cercanos. Con este método cada valor original de la matriz se reemplaza por el promedio aritmético de los valores a una distancia S de cada celda evaluada. En la Figura 3.3 se muestra este proceso para una matriz de 5×5 .

El método del punto más cercano consiste en asignar para cada par (x, y) el valor conocido más cercano, ver Algoritmo 1.

La interpolación por IDW consiste en asignar a cada punto el resultado de un promedio de los valores cercanos, ponderado por la distancia elevado a un exponente

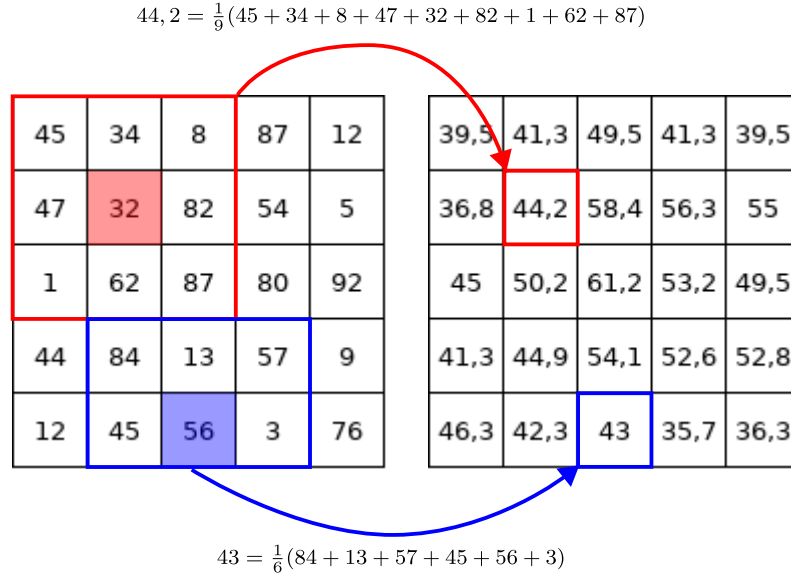


Figura 3.3. Suavizado por promedio con celdas vecinas, S=1

arbitrario p . Cuanto más grande el valor de p , más sensible es el método a los valores cercanos. La ecuación del promedio es la (3.1) y en el Algoritmo 2 se presenta el esquema utilizado.

$$f_p = \frac{\sum_{i=1}^n \frac{z_i}{d_i^p}}{\sum_{i=1}^n \frac{1}{d_i^p}} \quad (3.1)$$

En la Figura 3.4 se muestra una comparación de ambos métodos, para una malla de $C_D = f(\Delta P, l_v)$ generada al azar.

3.2.2 Área de Referencia

El área de referencia utilizada por ICESym es el área de cortina (ver Ec. 2.28) y se expresa en el código del programa como el área efectiva $F_V = A_R \cdot C_D$. Como se indicó en el apartado 2.7, para el MRCVC el área de referencia es el área frontal del puerto expuesta a la cámara, calculada como la altura de la ranura h_p multiplicada por la distancia entre el borde del puerto y la paleta que delimita la cámara, denominada como l_v .

Este valor se afecta por el coeficiente de descarga intermedio $C_{D,int}$, que puede ser un valor fijo o el resultado de interpolar de un mapa de C_D para un valor de cuerda y ΔP dado, como se indica en la ecuación (3.2).

Algoritmo 1: Interpolación por punto más cercano**Entrada:**

V_x, V_y : valores de x, y en los que se conoce el valor en z .

V_z : valores conocidos de z .

I_x : n puntos de x donde se quiere interpolar

I_y : m puntos de y donde se quiere interpolar

Resultado: Devuelve una matriz $I_{[n,m]}$ con los valores interpolados, donde a cada punto $I(x, y)$ se le asigna al valor de v_z más cercano conocido. Da como resultado superficies escalonadas.

```

1  $I = \text{zeros}_{[n,m]}$ ;
2 para  $i \leftarrow 0$  a  $n$  hacer
3     para  $j \leftarrow 0$  a  $m$  hacer
4          $d = \sqrt{(V_x - I_{xi})^2 + (V_y - I_{yj})^2}$ ;
5          $I[i, j] = v_z[\min(d)]$ ;

```

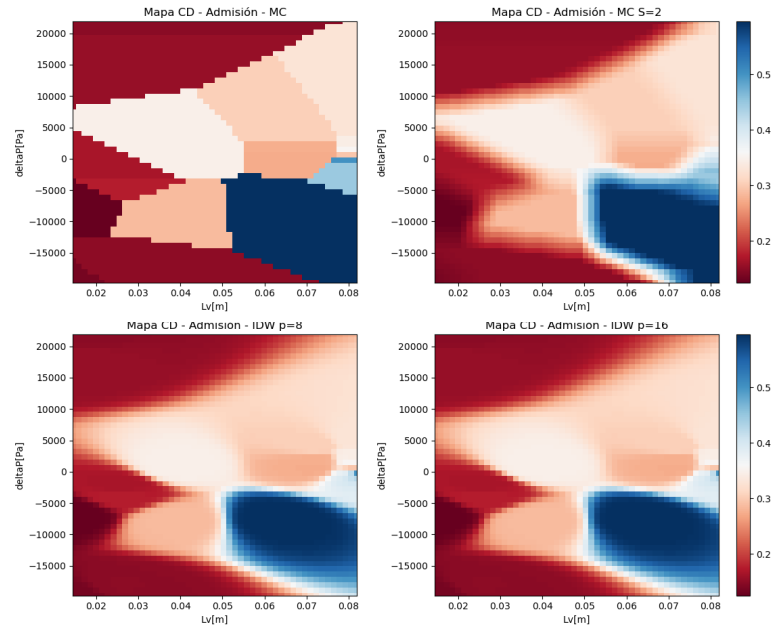


Figura 3.4. Comparación de métodos de interpolación

Algoritmo 2: Interpolación IDW

Entrada:

V_x, V_y : valores de x, y en los que se conoce el valor en z .

V_z : valores conocidos de z .

I_x : n puntos de x donde se quiere interpolar

I_y : m puntos de y donde se quiere interpolar

p : potencia a la que se eleva cada peso

Resultado: Interpolación ponderada por inverso de la distancia.

Dependiendo del valor de p , se obtienen valores más o menos suavizados.

```

1  $I = \text{zeros}_{[n,m]}$ ;
2 para  $i \leftarrow 0$  a  $n$  hacer
3   para  $j \leftarrow 0$  a  $m$  hacer
4      $d = [(V_x - I_{xi})^2 + (V_y - I_{yj})^2]^{\frac{p}{2}}$ ;
5     si  $\exists i : d[i] = 0$  entonces
6        $I[i, j] = V_z[i]$ ;
7     en otro caso
8        $I[i, j] = \frac{\sum V_{zi}/d_i}{\sum \frac{1}{d}}$ ;

```

$$F_v = C_{D,int} \cdot h_p \cdot l_v \quad (3.2)$$

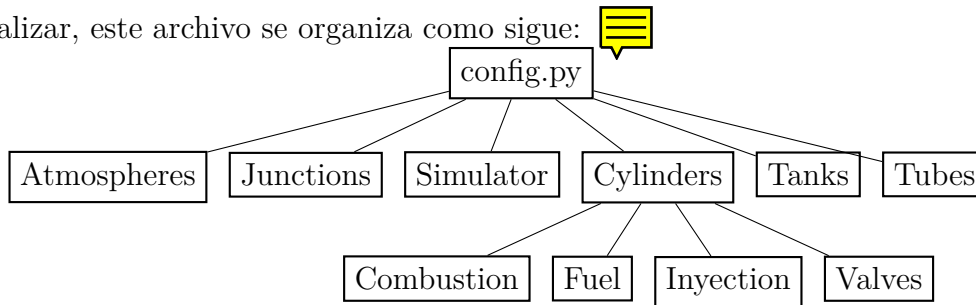
Tanto al inicio como al cierre del puerto ocurre solape de cámaras, por lo que en estos intervalos angulares hay un valor de C_D para cada cámara. Cada valor se calcula con el flujo másico que atraviesa las secciones de entrada correspondientes y el área de puerto expuesta por cada cámara.

3.2.3 Interfaz con Optimizador

Para lograr ejecutar el simulador automáticamente, se creó una librería de funciones capaz de tomar como dato de entrada un archivo de configuración que incluye geometría, velocidades a ejecutar y cantidad de ciclos de simulación, entre otros.

Para ejecutar una instancia de ICESym se puede utilizar la interfaz gráfica de usuario (GUI) ó ejecutarlo por línea de comando desde una consola. El simulador de motores se ejecuta como un archivo de Python >> `python main.py`, el cual contiene las instrucciones que lanzan la simulación del motor con la configuración requerida.

ICESym requiere de un archivo de configuración con los datos de la simulación a realizar, este archivo se organiza como sigue:



- **Atmospheres:** contiene el estado de la atmósfera, que es condición de contorno de la simulación: presión, densidad y velocidad inicial.
- **Cylinders:** geometría y condiciones de contorno, estado inicial, tipo de motor, como así también de las válvulas.
- **Valves:** geometría, tipo de válvula, modelo de C_D , perfil de alzada y datos de C_D y tubo conexionado.
- **Junctions:** contiene información de las uniones entre tubos.

- **Simulator:** configuración de la simulación, velocidades a simular, propiedades de gas, tipo de motor, directorios, entre otros.
- **Tanks:** volumen, masa y temperatura de pared de tanques.
- **Tubes:** geometría, cantidad de nodos y conexiones de los tubos.

Los elementos de configuración intervenidos por el optimizador son **Cylinders**, **Valves**, **Simulator** y **Tubes**; donde se modifican los siguientes valores:

- **Simulator:**
 - **RPMS:** Velocidades a simular (por ejemplo una lista de [1000, 2000, ..., 9000]).
 - **NCYCLES:** cantidad de ciclos por velocidad (un entero mayor o igual a 1).
 - **FOLDER NAME:** nombre de la carpeta donde se guardan los resultados de la simulación.
 - **SHOW INFO:** selector para mostrar o no información de la simulación.
 - **CONFIG DATA:** archivo donde se guarda la configuración utilizada.
- **Cylinders** \longrightarrow **Valves**
 - **LvI:** perfil de alzada del puerto de admisión.
 - **LvE:** perfil de alzada del puerto de escape.
 - **IPO:** ángulo de apertura del puerto de admisión.
 - **IPC:** ángulo de cierre del puerto de admisión.
 - **EPO:** ángulo de apertura del puerto de escape.
 - **EPC:** ángulo de cierre del puerto de escape.
 - **cd_model:** selector de modelo de C_D .
 - * $C_D l_v$ valores de alzada para el mapa de C_D (para modelo de 2 variables).
 - * $C_D d_p$ valores de Δ_P para el mapa de C_D (para modelo de 2 variables).



- * C_D valores de C_D relacionados con alzada (para modelo de 1 variable).
- D_v : diámetro de la cabeza de la válvula.
- Tubes
 - longitud: longitud total del tubo de admisión o escape.

3.3 Optimizador y Algoritmo Genético

Se seleccionó un algoritmo genético (AG) para realizar la optimización de la geometría del MRCVC por la simplicidad y facilidad de implementación del mismo. Si bien este tipo de métodos no garantiza que se alcance un resultado óptimo, en la práctica se ha observado que alcanzan soluciones muy cercanas a las óptimas tras pocas iteraciones del método (Goldberg, 1989)(Yu Shi et al., 2011).

Una de las ventajas de este método es que no requiere información del gradiente de la función que se está evaluando, lo cual es útil cuando no se puede asegurar la existencia de la derivada de la función en todo el dominio ó cuando se tiene una función con más de un máximo o mínimo local. Además, el punto de partida de la optimización es una población generada al azar, se tiene un muestreo aleatorio del dominio que se está evaluando. Esto hace que el método sea poco susceptible a dar como resultado óptimos locales.

Se puede decir que un algoritmo genético es un método de búsqueda aleatoria guiada. ¿Cómo difieren los AG de los métodos tradicionales de búsqueda?

1. Los AG pueden operar sobre una representación de las variables estudiadas y no necesariamente sobre las variables de estudio.
2. Cada iteración utiliza un conjunto de datos con cierto grado de aleatoriedad.
3. Utilizan una función objetivo para evaluar cada punto sin necesidad de conocer la derivada de la función que se está evaluando.
4. Los AG usan reglas probabilísticas de decisión.

Los mecanismos básicos que hacen a un algoritmo genético son: 1) *selección*, 2) *cruza* y 3) *mutación*. El funcionamiento básico se sintetiza en el Algoritmo 3.

La *selección* consiste en crear individuos a partir del puntaje que devuelve una función objetivo, la cual es la encargada de guiar el proceso de optimización dando mayor o menor puntaje a un candidato según el resultado que se quiere obtener. Este paso significa que, aquellos individuos a los cuales se les asignó un puntaje más elevado tienen más probabilidades de ser copiados o de “transmitir” sus parámetros a la iteración siguiente. Este proceso imita en cierta forma la selección natural o evolución Darwiniana y de aquí viene el nombre de algoritmo genético o evolutivo.

El segundo operador es la *cruza*, que consiste en combinar los parámetros de dos individuos para obtener uno nuevo, esto se asemeja a la reproducción.

Finalmente la *mutación* es la encargada de modificar aleatoriamente uno o más parámetros de cada nuevo individuo. Este operador juega un rol secundario pero muy importante en la simulación. Es secundario porque se pueden alcanzar soluciones satisfactorias sin que aplique este operador en la población. Es importante porque utilizando probabilidades pequeñas de ocurrencia (de la mutación), permite evitar la pérdida temprana de información relevante por convergencia temprana de la simulación. Por otro lado, en caso de que la probabilidad de mutación sea muy alta, el AG se convierte en un método de búsqueda aleatoria.

Algoritmo 3: Algoritmo de optimización

```

1 Inicializar población, al azar o a partir de una población “semilla”;
2 mientras no se cumpla condición de parada hacer
3   | Seleccionar a los individuos más aptos, evaluándolos según la función
   |   objetivo.;
4   | Cruzar los candidatos seleccionados para crear la nueva población (la
   |   próxima iteración del método);
5   | Mutar algunos individuos de la nueva población;
6   | si se cumple la condición de parada entonces
7   |   | Parar;
8 Guardar resultados;
```

Gran parte de este trabajo consistió en adaptar el uso de ICESym y emplearlo como base para generar una función objetivo, aprovechando la cualidad de “caja negra” con la que se puede implementar el simulador. Para lograr esto se modificó parte del código de ICESym con el objetivo de facilitar la configuración, ejecución y lectura de los resultados que arroja el simulador y así poder ejecutar de manera automática una simulación con una configuración particular del motor. Otro aspecto del optimizador que se desarrolló, es el de poder ejecutar múltiples instancias de ICESym en paralelo con el fin de reducir el tiempo de ejecución de cada generación, pudiendo evaluar varios motores (o individuos) al mismo tiempo.

Para la primera iteración se programaron desde cero los algoritmos y funciones necesarias para llevar a cabo la optimización con el AG. Posteriormente se tomó la librería DEAP ([Fortin et al., 2012](#)) y se modificaron los operadores a medida, para poder utilizarlos con ICESym.

En los apartados siguientes se describe la implementación de cada uno de los operadores en el optimizador.

3.3.1 Población

Se decidió representar cada motor como un vector con las dimensiones y reglaje que definen la geometría del sistema de intercambio de gases, los cuales se listan en la Tabla 3.1. Se limitaron los valores que puede tomar cada parámetro para que la geometría resultante se asemeje a la geometría del motor utilizado en trabajos anteriores, aprovechando así los resultados obtenidos en el primer barrido paramétrico.

Los vectores que hacen a cada motor se representan como un número binario de 40 dígitos, ocupando 5 dígitos para representar cada uno de los 8 parámetros que hacen a cada motor. Esto facilita la implementación de los operadores de selección, cruza y mutación, pudiendo aprovechar implementaciones de operadores existentes en librerías como DEAP. Estos 8 números binarios luego se convierten en una lista de enteros mediante una transformación lineal $f(x) = a \cdot x + b$, en la que se ingresa con un entero entre 0 y $2^n - 1$ para ir del número binario a un decimal, siendo n la cantidad de dígitos del número binario (en este caso 5). Los coeficientes a y b son tales que $f(0) = x_0$ y $f(2^n - 1) = x_1$, donde x_0 y x_1 son los extremos del rango para

Nº	Parámetro	Descripción	Sistema	Límites
1	DTA	Diámetro de tubo	Admisión	[60, 100] mm
2	DTE	Diámetro de tubo	Escape	[60, 100] mm
3	LIT	Largo de tubo	Admisión	[300, 2000] mm
4	LET	Largo de tubo	Escape	[300, 2000] mm
5	IIA	Ángulo geométrico de apertura	Admisión	[0,90]°
6	IFA	Ángulo geométrico de cierre	Admisión	[IIA, 90]°
7	IIE	Ángulo geométrico de apertura	Escape	[0, 90]°
8	IFE	Ángulo geométrico de cierre	Escape	[IIE, 90]°

Tabla 3.1. Parámetros que representan al motor

el que se quiere aplicar la transformación. Estos coeficientes (a y b) son particulares a cada parámetro, porque se determinan de acuerdo a los valores que puede tomar cada uno.

De este modo se obtiene el valor de cada uno de los parámetros que hacen a la configuración particular de cada motor en ICESym. El orden de los mismos se mantiene constante, por lo que cada sección del número representa una característica en particular del motor.

A modo de ejemplo, en la Figura 3.5 se muestra un número generado aleatoriamente, la transformada para los primeros 5 dígitos que corresponden al diámetro del tubo de admisión, 001111. Si se desea que el diámetro del tubo de admisión varíe entre 60 y 100 mm, se deben obtener los coeficientes a y b para la transformación lineal a partir del largo del número binario que se va a utilizar, `binLen`, y los valores mínimos y máximos del rango sobre el que se quiere transformar el valor de entrada: $vMin = 60$ y $vMax = 100$.

Los coeficientes a y b se obtienen a partir de:

$$a = \frac{vMax - vMin}{2^{binLen} - 1} = \frac{100 - 60}{2^5 - 1} = 1,2903$$

$$b = vMin = 60$$

Luego, el número binario transformado a entero vale:

$$00111 \longrightarrow 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 7$$

Finalmente, con los coeficientes (a, b) y el binario transformado en entero, se tiene que DTA vale

$$DTA = a \cdot x + b = (1,2903 \cdot 7 + 60) = 69mm = 0,069m$$

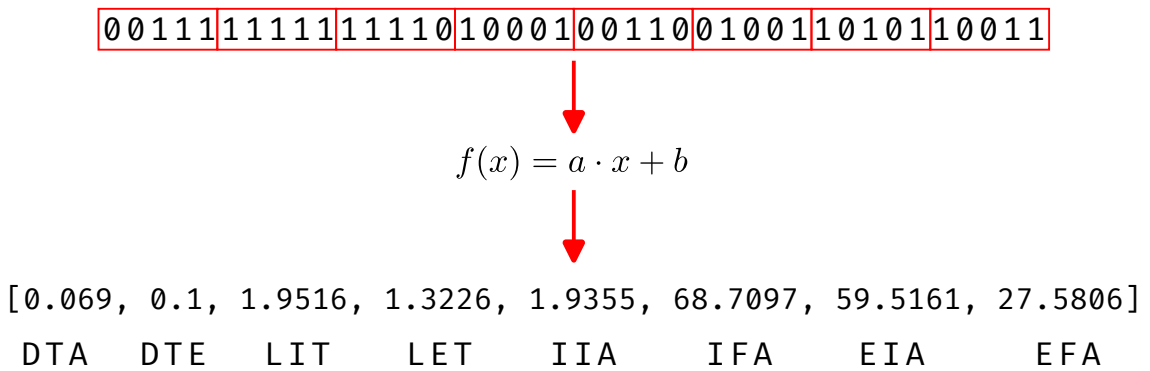


Figura 3.5. Representación del individuo

3.3.2 Selección

Para crear la nueva población se debe elegir a los nuevos candidatos basándose en los puntajes de la población actual. Hay varios métodos diferentes de selección, como lo son de ruleta, aleatoria, por puntaje y de tipo torneo. Para este trabajo se seleccionó el método de torneo, es uno de los métodos más populares para los procesos de selección de AG.

El método consiste en seleccionar k individuos de la población al azar, se comparan los puntajes de estos individuos y resulta “ganador” aquel que tenga más puntaje. El proceso se repite N veces hasta generar la nueva población.

El parámetro k es el tamaño de torneo y comúnmente se utiliza 2 (Oladele y Sadiku, 2013). A mayores valores para k se tiene una mayor pérdida de diversidad en los resultados (Blickle y Thiele, 1995) porque reduce la posibilidad de que candidatos con menor puntaje sean seleccionados para la nueva generación (convergencia temprana).

Para este torneo se utilizó además un *salón de la fama* (Wirsansky, 2020) de 1 individuo. Esto significa que el mejor individuo de la población actual es automáticamente seleccionado para la iteración siguiente y la selección por torneo se realiza $N - 1$ veces.

3.3.3 Cruza

El operador de cruza se encarga de combinar los genes de dos individuos para producir uno nuevo. Se encarga de combinar/intercambiar los parámetros de los individuos “cruzados” para generar uno nuevo. Para individuos representados por un vector se suelen usar operadores de tipo cruza de uno o múltiples puntos, también se utilizan mecanismos de cruza uniforme. El método seleccionado es *cruza de dos puntos*. En este método se corta el vector que forma al individuo en dos puntos, la posición de estos puntos se selecciona al azar, manteniendo el largo original de los vectores. Los individuos “cruzados” se combinan de forma complementaria como se indica en la Figura 3.6, el algoritmo 4 esquematiza el proceso.

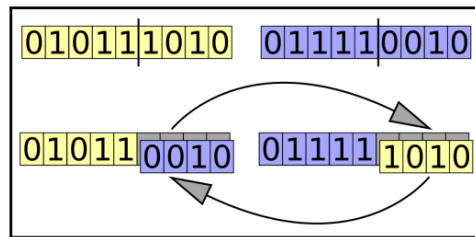


Figura 3.6. Cruza de dos puntos (Wirsansky, 2020)

3.3.4 Mutación

La mutación juega un rol secundario pero importante en los AG, consiste en modificar aleatoriamente alguno de los parámetros que definen a un individuo. Este mecanismo contribuye a la diversidad de soluciones y por ende reduce la posibilidad de convergencia temprana. Se utilizan probabilidades bajas de mutación, en el caso extremo si la probabilidad de mutación es del 100% el AG se convierte en un método de búsqueda aleatoria. Algunos de los métodos de mutación utilizados son:

1. Flip Bit

Algoritmo 4: Cruza de dos puntos

Entrada:

ind_1, ind_2 : dos individuos de entrada, por ej. [101...011], [110...100].

$EA(a, b)$: devuelve un entero al azar entre los enteros a y b .

$L(a)$: devuelve la cantidad de elementos en a .

Salida:

ind_1, ind_2 : individuos de entrada modificados

```

1  $s = \min(L(ind_1), L(ind_2));$ 
2  $CX_1 = EA(1, s);$ 
3  $CX_2 = EA(1, s-1);$ 
4 si  $CX_1 \geq CX_2$  entonces
5    $CX_2 = CX_2 + 1;$ 
6 en otro caso
7    $aux = CX_1;$ 
8    $CX_1 = CX_2;$ 
9    $CX_2 = aux;$ 
10  $aux = ind_1;$ 
11  $ind_1[CX_1 : CX_2] = ind_2[CX_1 : CX_2];$ 
12  $ind_2[CX_1 : CX_2] = aux[CX_1 : CX_2];$ 
13 devolver  $ind_1, ind_2;$ 

```

2. Intercambio

3. Inversión

4. Reordenado Aleatorio

En este trabajo se utiliza el método de reordenado aleatorio en el cual se modifica al azar el orden de los números que hacen al individuo, modificando los índices de la lista que define el arreglo, por ejemplo: 11100 \rightarrow 10011. El pseudocódigo de este proceso se presenta en el algoritmo 5.

Algoritmo 5: Flip Bit

Entrada: $A = (a_1, a_2, \dots, a_n)$ es un vector compuesto de unos y ceros.

R , es una función aleatoria que devuelve un número real entre 0 y 1.

p , es un número real entre 0 y 1 que representa la probabilidad de mutación.

```

1 para  $i=1$  a  $n$  hacer
2   si  $R < p$  entonces
3     si  $A_i = 1$  entonces  $A_i = 0$ 
4     en otro caso  $A_i = 1$ 
5 devolver  $A$ 

```

3.3.5 Función Objetivo

La función objetivo es la encargada de dar puntaje a los individuos, en la analogía con la selección natural esta función es el ambiente. Determina la aptitud de un motor con respecto a otro en lo que respecta a *performance* del sistema de intercambio de gases. Inicialmente se propuso que la función objetivo sea la suma de los rendimientos volumétricos a todas las velocidades simuladas $s = \sum \eta_v$. Este tipo de funciones dió como resultado una curva de η_v aserrada como se muestra en la Figura 3.7.

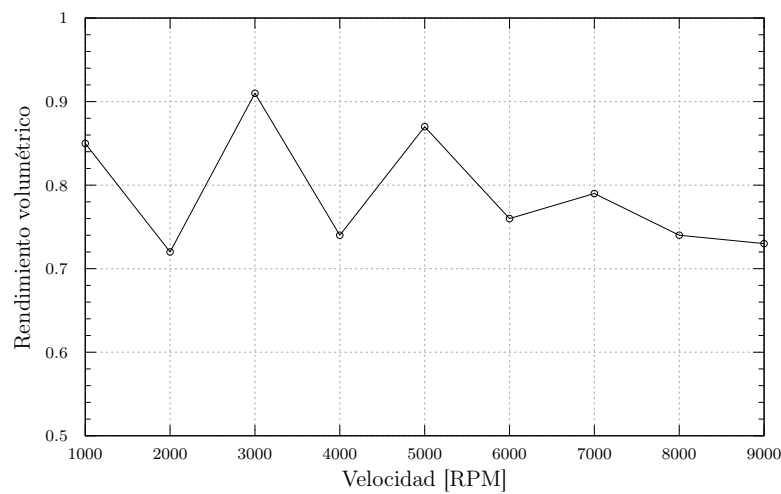


Figura 3.7. Curvas de rendimiento volumétrico aserradas

Esta curva aserrada es poco deseable porque significa una entrega de torque y

potencia dispar, por este motivo se modificó la función objetivo para favorecer curvas suaves y preferentemente con un solo punto de inflexión. Se implementó una suma ponderada para obtener un rendimiento volumétrico máximo en un valor cercano a 6000 RPM de modo de aprovechar las características de balanceo de fuerzas y mayores velocidades de giro de los motores rotativos. La aptitud resulta de la suma del rendimiento volumétrico y el inverso de la fracción de gases residuales, lo cual probó ser la función objetivo que mejores resultados dió. La metodología utilizada se resume a continuación.

1. Se evalúa cada motor, calculando el rendimiento volumétrico η_v y la fracción de gases residuales x_r para cada velocidad de giro simulada.
2. Con $\eta_v = (\eta_{v,1}, \dots, \eta_{v,n})$ y $x_r = (x_{r,1}, \dots, x_{r,n})$ se realiza la siguiente suma para cada velocidad $S_i = \eta_{v,i} + x_{r,i}^{-1}$.
3. Cada motor tiene un vector o lista de valores $S = (S_1, \dots, S_n)$ para cada velocidad evaluada, con la cual se calcula el puntaje del motor como:

$$f = \sum_{i=1}^n S_i + S_k^2 \quad (3.3)$$

El valor S_k es el puntaje para la k -ésima velocidad de giro (6000 RPM en este caso) y se eleva al cuadrado para favorecer altos rendimientos en esta velocidad.

Durante las primeras iteraciones del método hay una gran cantidad de geometrías inválidas que devuelven puntaje muy bajo o nulo. En caso de que alguna de las soluciones tenga un puntaje relativamente alto, existe la posibilidad de una dominancia temprana de la población, provocando una convergencia temprana de la optimización. Estos candidatos tienen una mayor probabilidad de “pasar” sus características geométricas a las iteraciones siguientes y es algo especialmente problemático en optimizaciones con poblaciones de alrededor de 100 individuos.


Para reducir la posibilidad de una convergencia temprana se utiliza un método de escalado de puntajes, que consiste en una transformación lineal en la que se define el puntaje bruto de un individuo como f y el puntaje escalado como f' , la

relación entre ambos es $f' = a \cdot f + b$. Los coeficientes a y b se determinan de modo que $f'_{media} = f_{media}$, de este modo un motor con puntaje promedio tiene la misma influencia sobre la población ya sea con la aptitud original o escalada. Para controlar la influencia del mejor individuo de una generación sobre la próxima, los puntajes se transforman de tal modo que $f'_{max} = C_{mult} f_{media}$. El valor de C_{mult} es la cantidad de copias que se espera obtener del mejor de los candidatos en la generación siguiente y se usa en 1, 2 a 2 para poblaciones de entre 50 y 100 individuos (Goldberg, 1989).

Hacia el final donde la diferencia entre puntajes de los individuos de la población tiende a achicarse, el parámetro C_{mult} cumple la función de acrecentar las diferencias entre individuos.

En caso de existir individuos con puntaje muy bajo o nulo se hace un pre-escalado del puntaje que fija el mínimo en $f'_{min} = 0$. El procedimiento se lista en los algoritmos 6 y 7.

3.4 OpenFOAM

Las flujometrías se realizaron con *OpenFOAM*, un software de Fluidodinámica Computacional, o CFD por sus siglas en inglés, de código libre y abierto e  escrito en “C++”. Junto con este programa se utilizaron otras herramientas libres para generar la geometría a modelar y post-procesar los resultados.. El esquema de trabajo para realizar las simulaciones consistió en:

1. Pre-procesado
 - (a) Definir la geometría a analizar.
 - (b) Generar una malla con un tamaño de elemento adecuado (la solución a problemas de CFD depende fuertemente de la cantidad y tamaño de celdas utilizadas).
 - (c) Seleccionar los modelos adecuados.
 - (d) Definir las propiedades del fluido.
 - (e) Definir las condiciones de borde.
2. Solver

Algoritmo 6: Algoritmo de pre-escalado

Entrada:

F , es un vector que contiene los puntajes de todos los individuos;

C_{mult} , es un multiplicador para el escalado, se suele usar $C_{mult} \in [1.2, 2]$;

Salida:

a, b , son los coeficientes para la transformación lineal $f(x) = a \cdot x + b$;

```

1   $u_{max} = \max(F)$ ;
2   $u_{min} = \min(F)$ ;
3   $u_{medio} = \text{media}(F)$ ;
4  si  $u_{min} > aux = (C_{mult} \cdot u_{medio} - u_{max}) / (C_{mult} - 1)$  entonces
5       $\Delta_u = u_{max} - u_{avg}$ ;
6       $a = (C_{mult} - 1) \cdot u_{avg} / \Delta_u$ ;
7       $b = u_{avg} \cdot (u_{max} - C_{mult} \cdot u_{avg}) \Delta_u$ ;
8  en otro caso
9      si  $\Delta \neq 0$  entonces
10          $a = u_{avg} / \Delta_u$ ;
11          $b = -u_{min} \cdot u_{avg} / \Delta_u$ ;
12     en otro caso
13          $a = 1$ ;
14          $b = 0$ ;
15 devolver  $a, b$ 

```

Algoritmo 7: Escalado de población

Entrada:

f , es la aptitud.

a, b , son los parámetros de la función de pre-escalado.

Salida:

f^* , los puntajes escalados.

```

1  $a, b = \text{PreEscalado}(f, \mathcal{Z})$ ;
2  $f^* = ()$  ;
3  $n = \text{Largo}(f)$ ;
4 para  $i = 1$  a  $n$  hacer
5    $f_i^* = a \cdot f_i + b$ ;
6 devolver  $f^*$ ;

```

(a) Seleccionar el solver a utilizar.

(b) Ejecutar la simulación.

3. Post-procesado

(a) Visualizar los resultados de las distintas variables de la simulación.

(b) Extraer la información necesaria.

3.5 Esquemas de Discretización

Se utilizan para resolver ecuaciones de variables continuas con funciones discretas en tiempo y espacio. Se deben seleccionar esquemas para resolver:

- Primera derivada temporal
- Interpolación
- Gradiente
- Divergencia
- Gradientes normales a superficies
- Laplacianos

3.5.1 Derivadas temporales, $\delta/\delta t$

Estas derivadas se discretizan con el método de Euler (Burden y Faires, 2012), que aproxima la integración de un paso n a $n + 1$ con $y_{n+1} - y_n \simeq hf_n$ donde $h = t_{n+1} - t_n$ es el paso temporal y $f_n = f(t_n, y_n)$. Para el esquema de Euler hacia atrás la aproximación es $y_{n+1} - y_n \simeq hf_{n+1}$. A este esquema se le agrega un coeficiente $\gamma \in [0, 1]$ de modo que:

$$y_{n+1} - y_n \simeq \gamma hf_{n+1} + (1 - \gamma)hf_n \quad (3.4)$$

Con $\gamma = 1/2$ el esquema es equivalente a Crank-Nicolson estándar. Se puede convertir al esquema de Euler hacia adelante con $\gamma = 0$.

3.5.2 Gradientes

Se discretiza utilizando integración Gaussiana con interpolación lineal entre valores de celdas. El método define al gradiente medio en un elemento de volumen finito con centroide \mathbf{C} y volumen V_c en términos de los flujos a través de sus caras, como lo indica la ecuación (3.5). Para esto se requiere conocer los valores de la variable ϕ_f en las caras vecinas e información del área de la celda y su normal (\vec{S}_f).

$$\nabla \phi_P = \frac{1}{V_c} \sum_f \vec{S}_f \phi_f \quad (3.5)$$

El método de volúmenes finitos utiliza valores en las caras de las celdas, por lo que se debe aproximar el valor de la variable en una cara dada para obtener el valor del gradiente en dicha celda. Los valores de ϕ_f se obtienen de una interpolación lineal entre valores conocidos de celdas adyacentes. Un método de interpolación entre celdas puede ser:

$$\alpha = \frac{|\vec{r}_N - \vec{r}_f|}{|\vec{r}_N - \vec{r}_P|} \quad (3.6)$$

$$\phi_f = \alpha \phi_P + (1 - \alpha) \phi_N \quad (3.7)$$

Donde α es un factor de ponderación geométrico entre las celdas \mathbf{P} , \mathbf{N} y \vec{r} es el vector posición del centroide de las celdas.

La interpolación se puede limitar para que los valores obtenidos se encuentren entre el mínimo y máximo de las celdas vecinas, este método se denomina “limitado”.

3.5.3 Gradiente normal a una superficie

Este gradiente es evaluado en la cara de la celda. Es la componente (normal a la cara) del gradiente entre los valores de los centroides de 2 celdas conectadas por la cara evaluada. En general las mallas utilizadas para modelar geometrías reales no son ortogonales. Esto implica que el vector \vec{CF} que une el centroide de dos celdas contiguas (\mathbf{C} y \mathbf{F}) no necesariamente es colineal con el vector \vec{S}_f normal a la superficie. El gradiente evaluado en la cara de una celda $(\nabla \cdot \vec{e})_f$, en la dirección del vector unitario que une los centroides de \mathbf{C} y \mathbf{F} (\vec{e}), se puede expresar como se indica en la Ecuación 3.8

$$\vec{e} = \frac{\vec{r}_F - \vec{r}_C}{\|\vec{r}_F - \vec{r}_C\|} = \frac{\vec{d}_{CF}}{d_{CF}} \quad (3.8)$$

$$(\nabla \phi \cdot \vec{e})_f = \frac{\partial \phi}{\partial n} = \frac{\phi_F - \phi_C}{\|\vec{r}_C - \vec{r}_F\|} = \frac{\phi_F - \phi_C}{d_{CF}} \quad (3.9)$$

Donde el subíndice f indica que se evalúa en la cara de una celda. El vector de superficie \vec{S}_f se puede escribir en términos de sus componentes normal y tangente a la cara f en la que es evaluado:

$$\vec{S}_f = \vec{E}_f + \vec{T}_f \quad (3.10)$$

De esta forma, el gradiente de la variable ϕ en mallas no ortogonales se puede expresar en términos de las componentes normal y tangente a la cara de la celda (Moukalled et al., 2016). El término \mathbf{E} indica la componente del gradiente normal a la cara y \mathbf{T} la componente tangente.

$$(\nabla \phi)_f \cdot \vec{S}_f = (\nabla \phi)_f \cdot \vec{E}_f + (\nabla \phi)_f \cdot \vec{T}_f = \vec{E}_f \frac{\phi_F - \phi_C}{d_{CF}} + (\nabla \phi)_f \cdot \vec{T}_f \quad (3.11)$$

Algunos esquemas de discretización de este tipo de gradientes son:

- No corregido

- Ortogonal
- Corregido y limitado

La corrección ortogonal ajusta el vector d_{CF}^{\rightarrow} para que sea colineal con el vector \vec{S}_f , asegurando que el cálculo del gradiente se haga en la dirección normal a la superficie. La corrección limitada aplica la corrección ortogonal, sumando un término adicional para tener en cuenta la desviación entre d_{CF}^{\rightarrow} y \vec{S}_f debido a la no ortogonalidad, generalmente en términos de $\cos^{-1} \theta$, donde θ es el ángulo entre el vector normal a la cara y el vector d_{CF}^{\rightarrow} .

3.5.4 Divergencia

Se utiliza un esquema de integración Gaussiana con interpolación lineal para la discretización de la divergencia.

Dependiendo de los tipos de variable, se utilizan diferentes esquemas de interpolación disponibles son:

- centrada
- hacia adelante
- hacia atrás
- limitada



3.5.5 Laplacianos

Los términos Laplacianos se discretizan utilizando integración Gaussiana con interpolación lineal.