

Manual de instalación, uso y actualización



Anexo al Trabajo Especial de Grado:

«Desarrollo de una herramienta de software para el
diseño de redes de adaptación de impedancias tipo
MSMN de banda ancha en líneas de transmisión basada
en algoritmos heurísticos de optimización comparados»

Autor: Abraham Hidalgo

Febrero, 2022

Índice general

1. Requerimientos	2
2. Instalación	3
3. Uso	4
3.1. Ingresar los parámetros del sistema	5
3.1.1. Configurar el generador	6
3.1.2. Configurar la línea de transmisión	8
3.1.3. Configurar la MSMN	9
3.1.4. Configurar la impedancia de carga	10
3.2. Modo: Diseño MSMN	12
3.2.1. Establecer requerimientos de adaptación de impedancia .	12
3.2.2. Resultados de la búsqueda	13
3.2.3. Rendimiento computacional	13
3.2.4. Graficas	14
3.3. Modo: Análisis de MSMN	17
4. Actualización	18
4.1. Modificar ventanaPrincipal.ui	19
4.2. Modificar ControladorDeObjetivo.py	20
4.3. Modificar AlgoritmosDeOptimizacion.py	21

Requerimientos

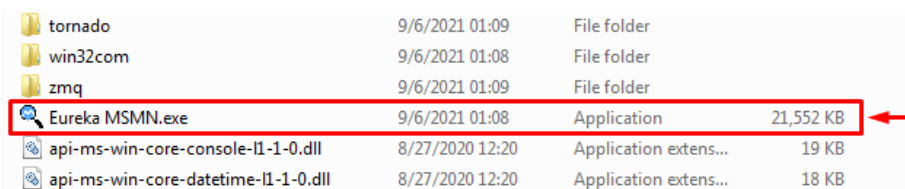
A continuación se listan los requerimientos de software y hardware para el correcto funcionamiento de Eureka MSMN:

- Sistema operativo Microsoft Windows de 32 o 64 bits.
- Resolución mínima de pantalla de 800x600 píxeles.
- Memoria RAM mínima de 2 GB (recomendado 4 GB).
- Procesador igual o superior al Intel Celeron de 1,10 GHz (o equivalente).

Instalación

Eureka MSMN es un software portable, diseñado para ser ejecutado en el sistema operativo Microsoft Windows, de modo que todos los archivos necesarios para su funcionamiento se encuentran en una carpeta principal. Para la distribución de este software, se provee el archivo **Eureka MSMN.rar**, el cual contiene la carpeta principal. Lleve a cabo el siguiente procedimiento:

- Copie el archivo **Eureka MSMN.rar** y péguelo en la carpeta que desee.
- Extraiga el contenido del archivo comprimido **Eureka MSMN.rar**.
- Después de realizar el paso anterior, aparece una nueva carpeta llamada **Eureka MSMN**, esta es la carpeta principal del programa.
- Para iniciar el programa, ingrese a la carpeta principal **Eureka MSMN** y ejecute el archivo **Eureka MSMN.exe**.



tornado	9/6/2021 01:09	File folder	
win32com	9/6/2021 01:08	File folder	
zmq	9/6/2021 01:09	File folder	
Eureka MSMN.exe	9/6/2021 01:08	Application	21,552 KB
api-ms-win-core-console-l1-1-0.dll	8/27/2020 12:20	Application extens...	19 KB
api-ms-win-core-datetime-l1-1-0.dll	8/27/2020 12:20	Application extens...	18 KB

Figura 2.1: Archivo ejecutable Eureka MSMN.exe.

Uso

Al ejecutar el archivo **Eureka MSMN.exe**, se muestra el logo de la aplicación y el proceso de carga de las librerías necesarias, seguidamente se muestra la ventana principal. En esta ventana debe ingresar los parámetros que caracterizan al sistema de línea de transmisión.

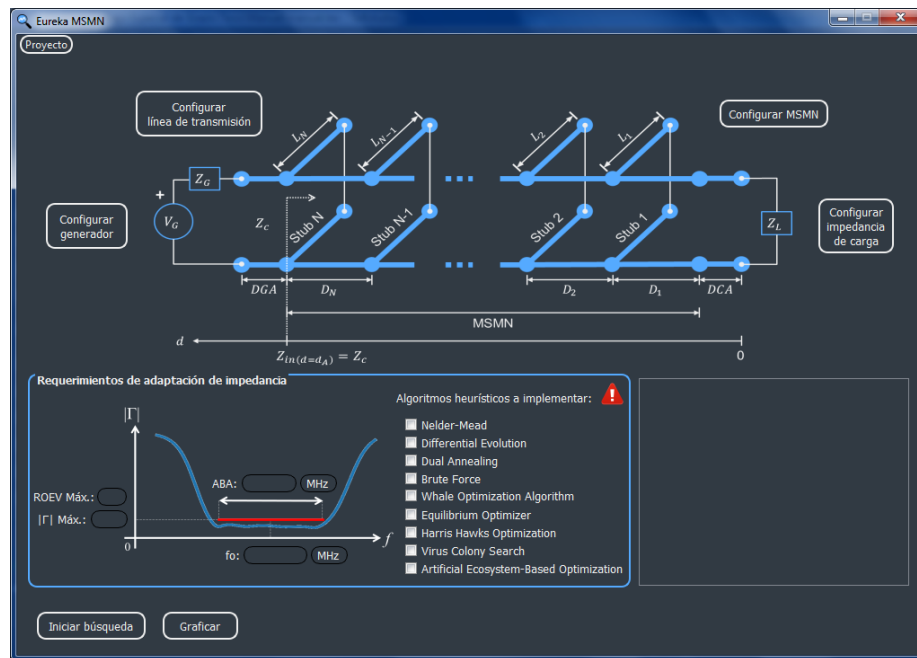


Figura 3.1: Ventana principal.

Eureka MSMN tiene básicamente dos modos de operación:

- **Análisis de MSMN:** Se ingresan todos los parámetros que definen al sistema, incluyendo la configuración de la red MSMN y el programa le permite explorar el funcionamiento de dicha red de adaptación a lo largo de esta y en un rango de frecuencia establecido.
- **Diseño de MSMN:** Se ingresan los parámetros del sistema, menos la configuración de la red MSMN. Además se ingresa los requerimientos de adaptación de impedancia. La herramienta se encarga de diseñar las redes MSMN y, una vez finalizado el proceso de diseño, se puede explorar el desempeño de las redes de adaptación obtenidas, a lo largo de estas y en un rango de frecuencia establecido.

3.1. Ingresar los parámetros del sistema

En la parte superior de la ventana principal se encuentran 4 botones que sirven para configurar el generador, la línea de transmisión, la red MSMN y la impedancia de carga, respectivamente (ver Figura 3.1). Puede configurar estas partes del sistema en el orden que desee. Al hacer click en cada uno de estos botones de configuración, se abre una nueva ventana correspondiente.

Otra manera de ingresar los parámetros del sistema es importando un archivo de extensión **.search**. Este tipo de archivo contiene toda la información del generador, la línea de transmisión, requerimientos de diseño de la red MSMN, impedancia de carga y requerimientos de adaptación de impedancia. Es decir, un archivo **.search** contiene toda la información de un proyecto. Para cargar un archivo **.search**, desde la ventana principal, siga la ruta: Proyecto → Cargar (ver Figura 3.2).

Una vez que haya ingresado todos los parámetros de un proyecto, este se puede guardar en un archivo **.search**. Para esto, desde la ventana principal, siga la ruta: Proyecto → Guardar.

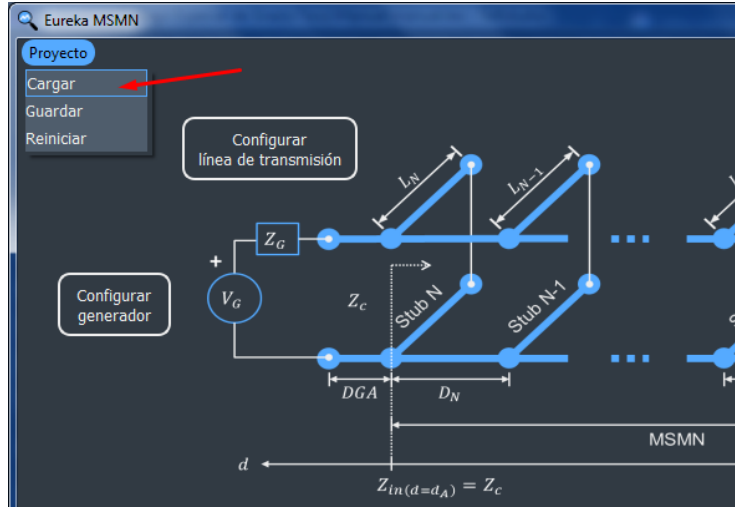


Figura 3.2: Cargar archivo **.search**.

3.1.1. Configurar el generador

En la ventana de configuración del generador, ingrese la información de la fuente y de la impedancia del mismo (ver Figura 3.3).

Configurar la fuente del generador

En la ventana de configuración del generador, puede ingresar la potencia Pin versus la frecuencia o el voltaje $|Vin|$ versus la frecuencia. Se puede obtener más información acerca del significado de Pin y $|Vin|$ al situar el cursor sobre estos nombres en la ventana. Otra manera de ingresar la información es cargando un archivo **.source** que contiene toda la información necesaria de la fuente. Para cargar un archivo **.source**, desde la ventana de configuración del generador, siga la ruta: Archivo → Cargar → Fuente (ver Figura 3.4). Para guardar un archivo **.source** siga la ruta: Archivo → Guardar → Fuente.

Configurar la impedancia del generador

En la ventana de configuración del generador, puede ingresar la impedancia del mismo seleccionando un modelo circuital y especificando los valores de resistencia, inductancia y capacitancia según corresponda.

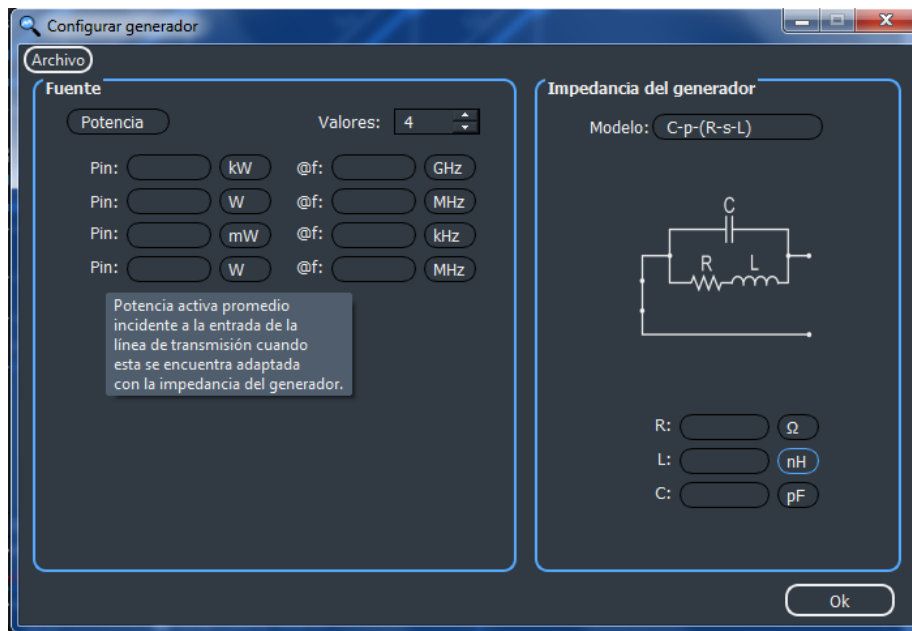


Figura 3.3: Ventana de configuración del generador.

Se puede cargar un archivo **.modelimp**, el cual contiene toda la información de un modelo circuital de impedancia. Para cargar un archivo **.modelimp**, desde la ventana de configuración del generador, siga la siguiente ruta: Archivo → Cargar → Impedancia → Modelo. Para guardar un archivo **.modelimp**, siga la siguiente ruta: Archivo → Guardar → Impedancia → Modelo.

Otra forma de especificar la impedancia del generador es importando un archivo **.imp** el cual es un archivo de texto que consiste de datos tabulados correspondientes a la resistencia y la inductancia obtenida para un conjunto de puntos de frecuencia (ver Figura 3.5). Para cargar un archivo **.imp**, desde la ventana de configuración del generador, siga la siguiente ruta: Archivo → Cargar → Impedancia → Archivo.imp.

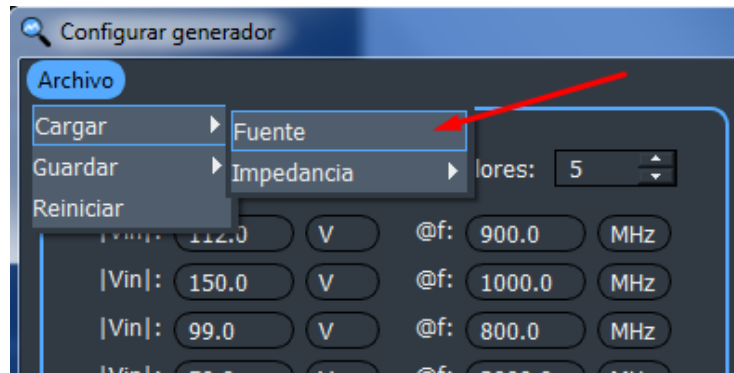
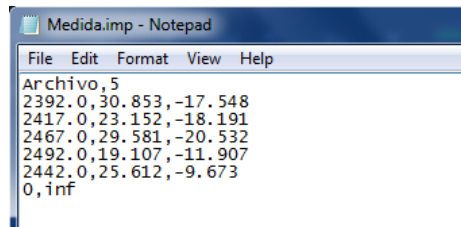
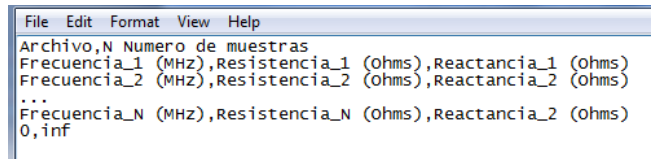


Figura 3.4: Cargar archivo **.source**.



(a) Ejemplo de un archivo **.imp**



(b) Estructura de un archivo **.imp**

Figura 3.5: Ejemplo y estructura de un archivo **.imp**.

3.1.2. Configurar la línea de transmisión

En la ventana de configuración de la línea de transmisión (ver Figura 3.6), ingrese los parámetros que caracterizan la línea. Se puede especificar el tipo de pérdidas, parámetros distribuidos o modelos de línea basados en la geometría y propiedades eléctricas de la misma. Además debe especificar la potencia activa promedio máxima que puede soportar la línea de transmisión y las distancias entre el generador, la impedancia de carga y el adaptador MSMN.

Otra manera de ingresar la información es cargando un archivo **.line**, el cual contiene toda la información de la línea de transmisión. Para cargar un archivo **.line**, desde la ventana de configuración de la línea de transmisión, siga la siguiente ruta: Archivo → Cargar. Para guardar un archivo **.line**, siga la siguiente ruta: Archivo → Guardar.

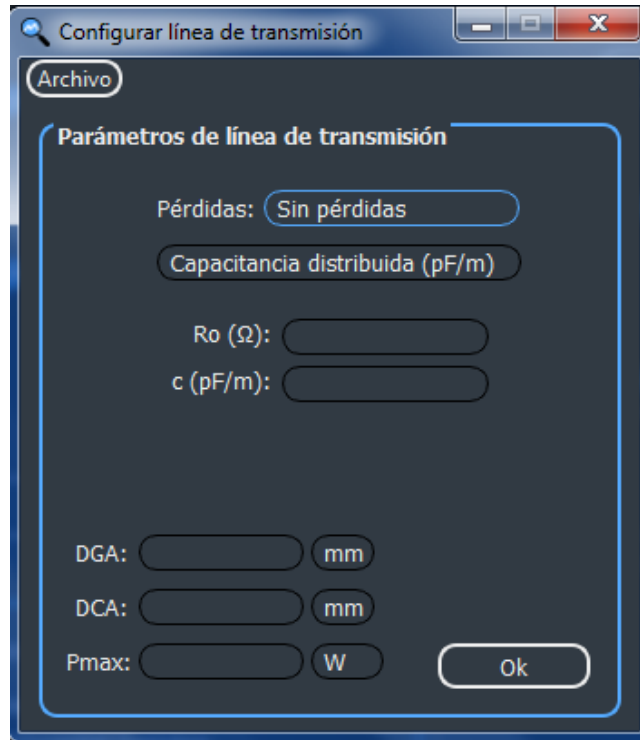


Figura 3.6: Ventana de configuración de la línea de transmisión.

3.1.3. Configurar la MSMN

En la ventana de configuración de la MSMN (ver Figura 3.7), especifique las características que deben tener las redes de adaptación de impedancia que serán diseñadas mediante los algoritmos heurísticos de optimización. Puede establecer el número máximo de stubs que conformarán la red o puede especificar un número fijo de stubs. También debe especificar la longitud y distancia mínima que debe tener cada stub y el tipo de carga de estos, ya sea corto circuito o circuito abierto.

Otra manera de ingresar la información es cargando un archivo **.reqstubs**, el cual contiene toda la información de los requerimientos establecidos para las

redes MSMN. Para cargar un archivo **.reqstubs**, desde la ventana de configuración de la MSMN, siga la siguiente ruta: Archivo → Cargar. Para guardar un archivo **.reqstubs**, siga la siguiente ruta: Archivo → Guardar.



Figura 3.7: Ventana de configuración de la MSMN.

3.1.4. Configurar la impedancia de carga

En la ventana de configuración de la impedancia de carga (ver Figura 3.8), puede ingresar la impedancia de carga del sistema seleccionando un modelo circuital y especificando los valores de resistencia, inductancia y capacitancia según corresponda.

Se puede cargar un archivo **.modelimp**, el cual contiene toda la información de un modelo circuital de impedancia. Para cargar un archivo **.modelimp**, desde la ventana de configuración de la impedancia de carga, siga la siguiente ruta: Archivo → Cargar → Modelo. Para guardar un archivo **.modelimp**, siga

la siguiente ruta: Archivo → Guardar → Modelo.

Otra forma de especificar la impedancia de carga es importando un archivo **.imp** (ver Figura 3.5). Para cargar un archivo **.imp**, desde la ventana de configuración de la impedancia de carga, siga la siguiente ruta: Archivo → Cargar → Archivo.imp.

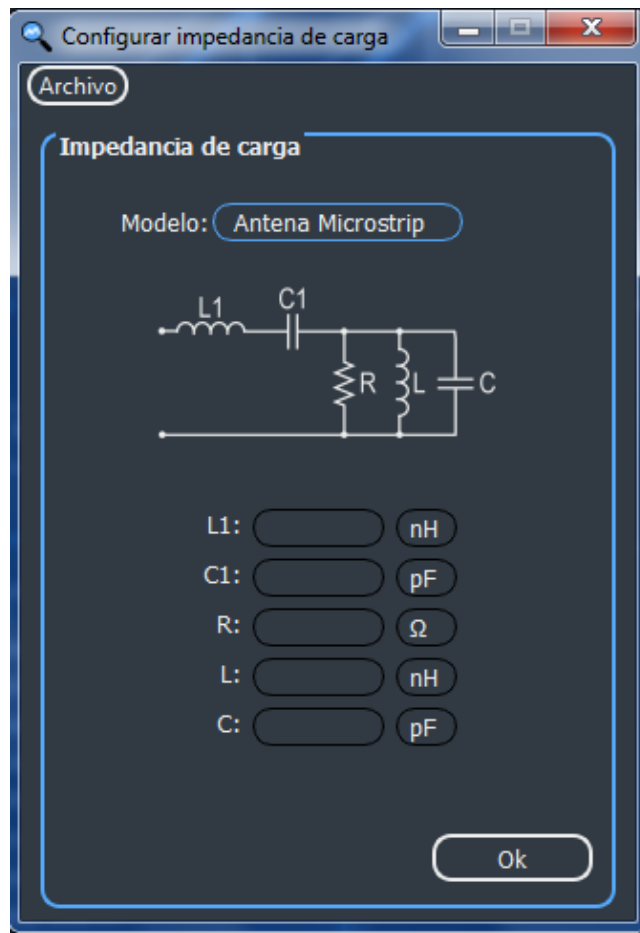


Figura 3.8: Ventana de configuración de la impedancia de carga.

3.2. Modo: Diseño MSMN

En este modo se tiene un sistema de línea de transmisión inicialmente desadaptado en impedancia, se establecen los requerimientos de adaptación, y se emplean algoritmos heurísticos de optimización para diseñar redes MSMN.

3.2.1. Establecer requerimientos de adaptación de impedancia

Una vez que ha ingresado toda la información que caracteriza al sistema de línea de transmisión (generador, línea, requerimientos de diseño de MSMN, impedancia de carga), se debe establecer los requerimientos de adaptación de impedancia, lo cual consiste en: frecuencia central (f_0), ancho de banda de adaptación (ABA), valor máximo permisible del módulo del coeficiente de reflexión de voltaje ($|\Gamma|$) en todo el ancho de banda, o en su lugar el valor máximo de la relación de onda estacionaria de voltaje (ROEV) (ver la Figura 3.9).

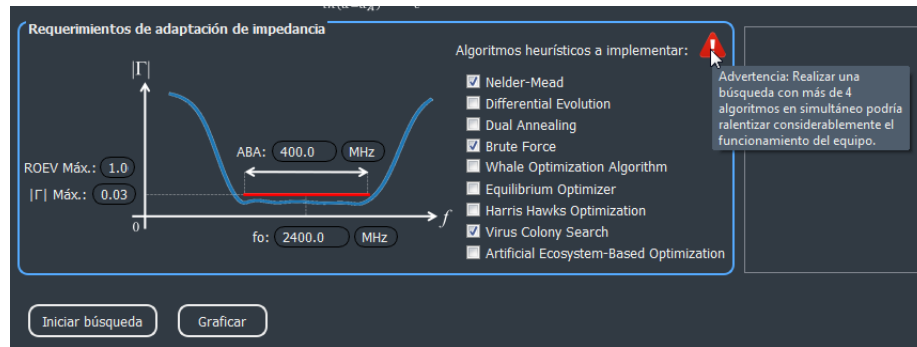


Figura 3.9: Requerimientos de adaptación de impedancia.

Puede escoger cuales algoritmos de optimización quiere emplear en el diseño de las redes MSMN. Tenga en cuenta que a mayor número de algoritmos a implementar mayor será el consumo de recursos computacionales (porcentaje de uso de CPU y memoria RAM). Una vez que se ha establecido todos los parámetros, haga click en el botón **Iniciar búsqueda**.

3.2.2. Resultados de la búsqueda

Una vez que cada algoritmo haya terminado de ejecutarse, aparece un aviso indicando que la búsqueda ha finalizado. Además aparecen dos nuevos botones: **Ver graficas** y **Ver rendimiento computacional**.

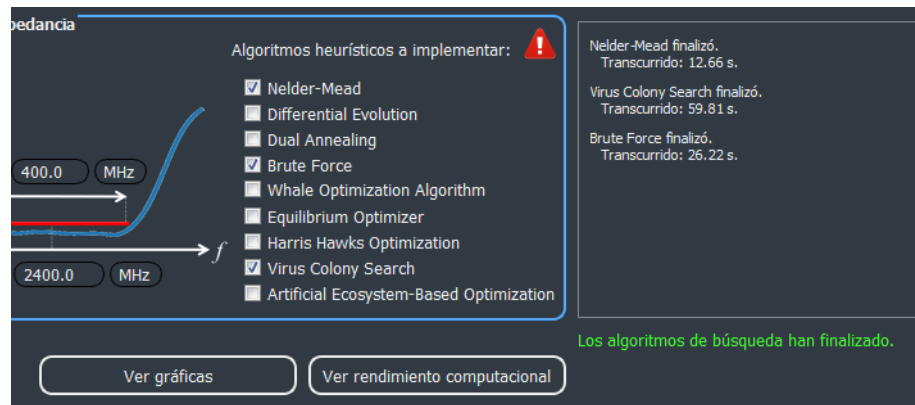


Figura 3.10: Búsqueda finalizada.

3.2.3. Rendimiento computacional

Al hacer click en el botón **Ver rendimiento computacional**, se abre una nueva ventana que muestra los algoritmos seleccionados y su consumo en porcentaje de CPU, memoria RAM y tiempo de búsqueda.

The screenshot shows a window titled 'Rendimiento computacional'. It has a dropdown menu set to 'Uso de CPU' and a button 'Mayor a menor ↓'. Below is a table with four columns: 'Algoritmo', 'Uso de CPU (%)', 'Memoria máxima utilizada (MiB)', and 'Tiempo de búsqueda (s)'. The table lists three algorithms: Virus Colony Search, Nelder-Mead, and Brute Force, with their respective performance metrics.

Algoritmo	Uso de CPU (%)	Memoria máxima utilizada (MiB)	Tiempo de búsqueda (s)
Virus Colony Search	8.36	709.98	59.81
Nelder-Mead	7.79	709.96	12.66
Brute Force	6.75	709.99	26.22

Figura 3.11: Rendimiento computacional.

3.2.4. Graficas

Al hacer click en el botón **Ver gráficas**, se abre una nueva ventana que muestra los resultados arrojados por cada algoritmo de búsqueda. Inicialmente se muestra el módulo del coeficiente de reflexión obtenido en el rango de frecuencia de interés para todos los algoritmos. Puede modificar el rango de frecuencia para visualizar las gráficas según lo desee. También se puede elegir ver el resultado de un algoritmo en particular o todos en simultáneo.

Haciendo click en el selector de **Tipo de gráfica** se puede elegir la gráfica de interés, las opciones disponibles son: Coeficiente de reflexión, impedancia vista, ROEV, carta de Smith, potencia activa promedio y respuesta en frecuencia. Además, el selector de **Presentación** permite escoger entre cuatro opciones a mostrar: Magnitud, fase, parte real y parte imaginaria.

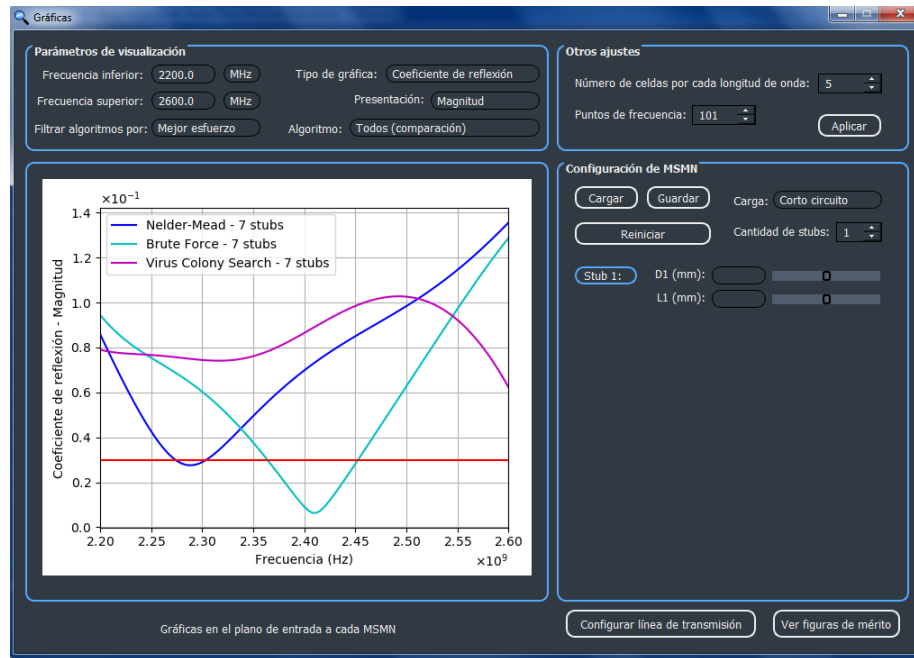


Figura 3.12: Ventana de gráficas.

En la gráfica de la magnitud del coeficiente de reflexión se muestra una línea roja horizontal que representa el valor $|\Gamma|_{Max}$, el cual fue especificado en los requerimientos de adaptación de impedancia.

Los resultados pueden ser filtrados por «Mejor esfuerzo» o «Modo estricto». En el «Modo estricto» sólo se muestran los algoritmos cuya gráfica de la magnitud del coeficiente de reflexión esté por debajo de la línea roja de $|\Gamma|_{Max}$ en todo el ancho de banda especificado en los requerimientos de adaptación de impedancia.

Movimiento en el eje d

En la ventana de gráficas, al seleccionar un algoritmo en específico, se muestra una barra azul horizontal debajo de la gráfica (ver Figura 3.13). Esta barra permite cambiar el punto de vista a lo largo de la línea de transmisión principal. De acuerdo a la ilustración de un sistema de línea de transmisión, este eje es denotado con la letra d (ver Figura 3.14).

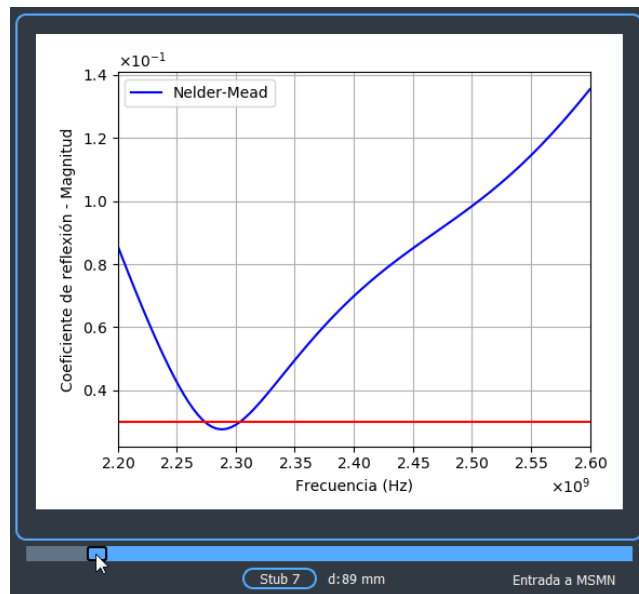


Figura 3.13: Barra horizontal, movimiento en el eje d.

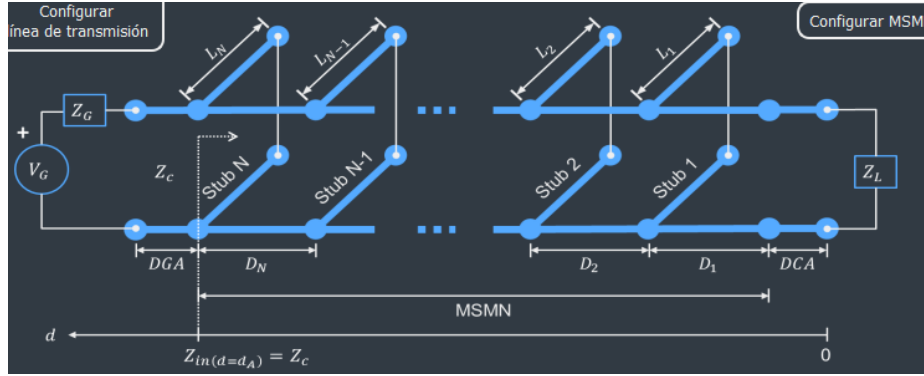


Figura 3.14: Sistema de línea de transmisión.

Configuración de MSMN

En la ventana de gráficas, al seleccionar un algoritmo en específico, se activa la sección de configuración de MSMN en la que se muestra la posición y longitud de cada stub de la red MSMN. Varíe los valores de posición (D) y longitud (L) de los stubs editando el texto directamente o moviendo la respectiva barra azul que está a la derecha. Los cambios se reflejan automáticamente en la gráfica.

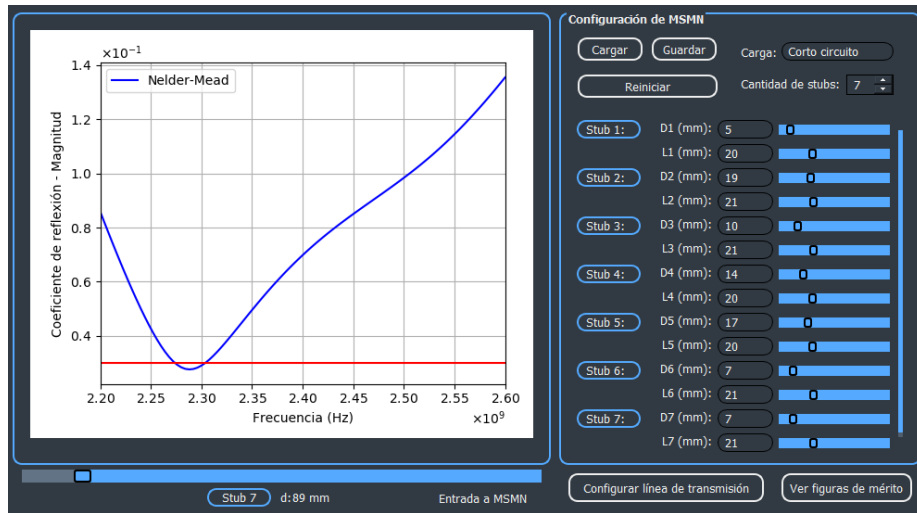
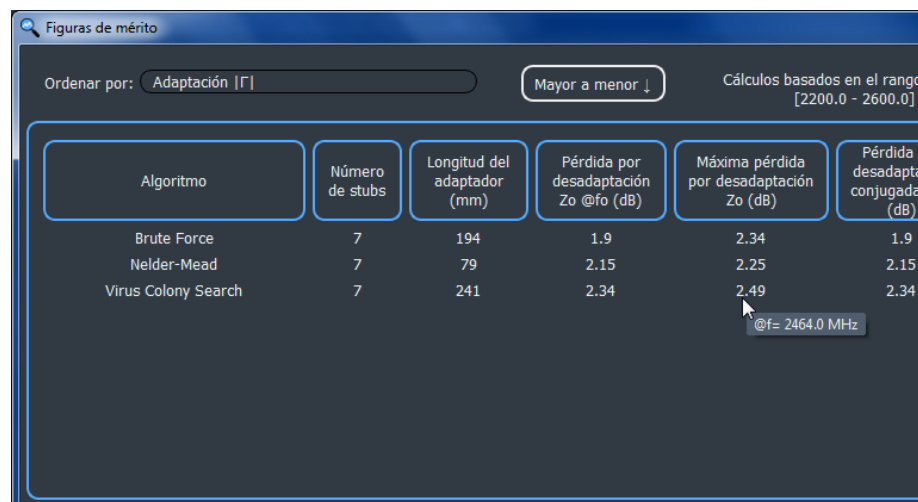


Figura 3.15: Sección de configuración de MSMN.

Figuras de mérito

Al hacer click en el botón de **Ver figuras de mérito** se abre una nueva ventana que muestra los algoritmos organizados a modo de ranking de acuerdo a criterios seleccionables. Esta ventana ofrece mayor información al posicionar el cursor sobre las cantidades mostradas.



The screenshot shows a window titled 'Figuras de mérito' with a search icon. It contains a table with columns: Algoritmo, Número de stubs, Longitud del adaptador (mm), Pérdida por desadaptación Zo @fo (dB), Máxima pérdida por desadaptación Zo (dB), and Pérdida por desadaptación conjugada (dB). The table lists three algorithms: Brute Force, Nelder-Mead, and Virus Colony Search. A tooltip is visible over the 'Máxima pérdida por desadaptación Zo (dB)' value for Virus Colony Search, showing '@f= 2464.0 MHz'.

Algoritmo	Número de stubs	Longitud del adaptador (mm)	Pérdida por desadaptación Zo @fo (dB)	Máxima pérdida por desadaptación Zo (dB)	Pérdida por desadaptación conjugada (dB)
Brute Force	7	194	1.9	2.34	1.9
Nelder-Mead	7	79	2.15	2.25	2.15
Virus Colony Search	7	241	2.34	2.49	2.34

Figura 3.16: Ventana de figuras de mérito.

3.3. Modo: Análisis de MSMN

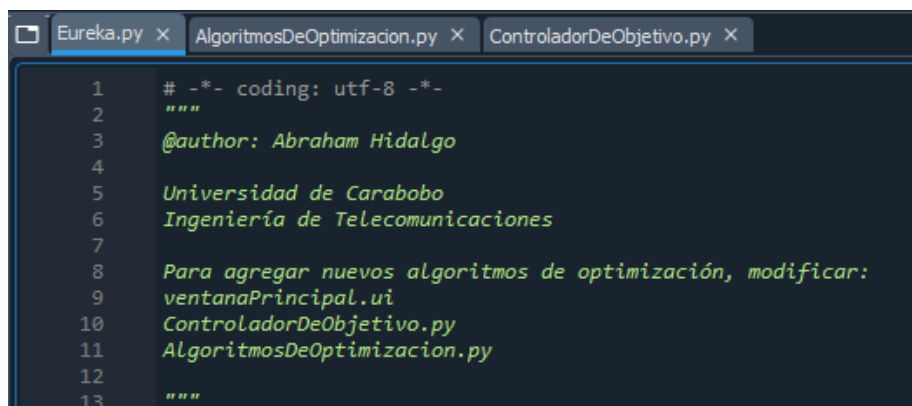
Para utilizar el modo Análisis de MSMN, en la ventana principal del programa, ingrese los parámetros del sistema de acuerdo a lo explicado anteriormente. Posteriormente haga click en el botón **Graficar**, esto lo llevará a la ventana de gráfica. Una vez en la ventana de gráfica, especifique el rango de frecuencia y la configuración de los stubs que definen la red MSMN.

En ambos modos (análisis y diseño), al modificar el rango de frecuencia, la configuración de la red MSMN, el punto de vista sobre el eje d o las propiedades de la línea de transmisión; se aprecia los efectos automáticamente en la grafica.

Actualización

El proceso de actualización al que se hace referencia en esta sección consiste en la inserción o modificación de módulos computacionales, en concreto, se trata de algoritmos de optimización a implementar. Para ello, se debe contar con la carpeta del proyecto Eureka MSMN, donde se encuentran los archivos que conforman el código fuente y demás archivos necesarios para su ejecución.

El archivo principal se encuentra nombrado como **Eureka.py**, al abrir este archivo, en sus primeras líneas se encuentra un comentario que indica cuáles archivos deben ser modificados con el fin de actualizar la aplicación (ver Figura 4.1).



```
1  # -*- coding: utf-8 -*-
2  """
3  @author: Abraham Hidalgo
4
5  Universidad de Carabobo
6  Ingeniería de Telecomunicaciones
7
8  Para agregar nuevos algoritmos de optimización, modificar:
9  ventanaPrincipal.ui
10 ControladorDeObjetivo.py
11 AlgoritmosDeOptimizacion.py
12 """
13
```

Figura 4.1: Archivo principal Eureka.py.

4.1. Modificar ventanaPrincipal.ui

El archivo **ventanaPrincipal.ui** corresponde a la configuración de la interfaz gráfica de la ventana principal del programa. En este paso se modificará la interfaz gráfica para agregar los widgets que representen a los nuevos algoritmos. Abra este archivo con la aplicación Qt Designer (ver Figura 4.2).

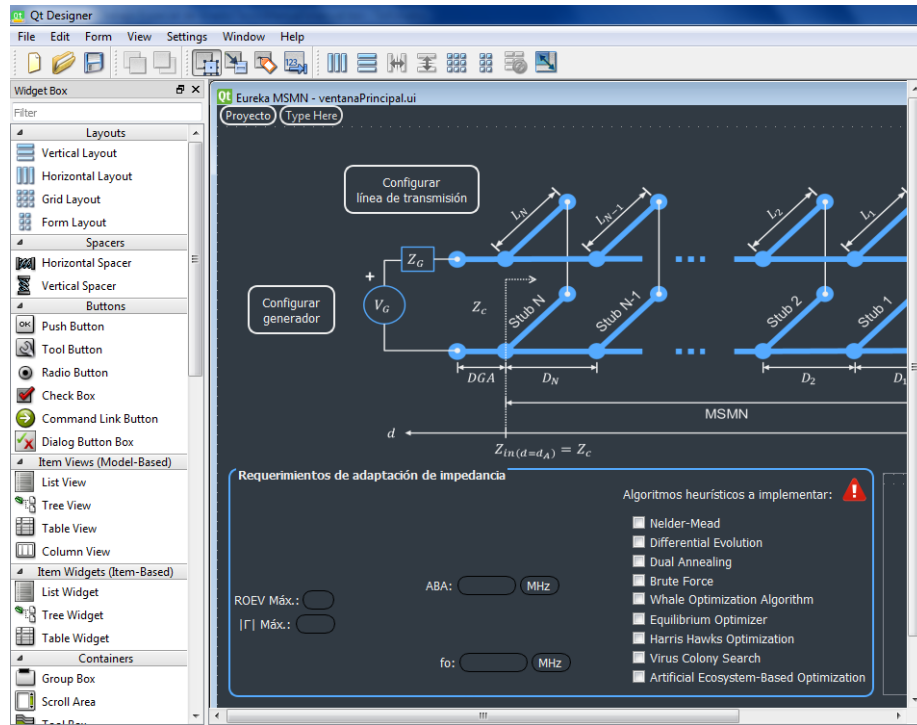
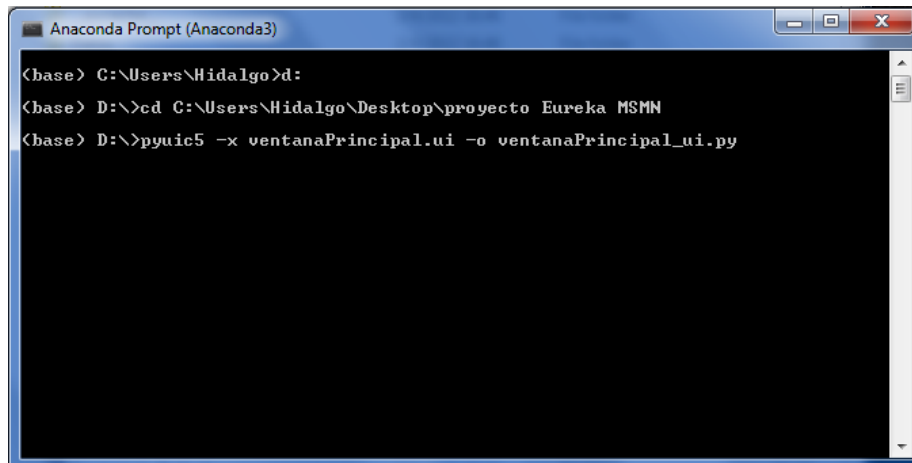


Figura 4.2: Ventana principal en Qt Designer.

En la sección de **Requerimientos de adaptación de impedancia** se encuentra una lista con los nombres de los algoritmos y su CheckBox correspondiente. Agregue y/o elimine los CheckBox de los algoritmos que desee. Recuerde colocarle nombres representativos a los CheckBox, por ejemplo, en el caso del algoritmo Nelder-Mead, su CheckBox está nombrado como `chkbx_NM`.

Una vez que haya terminado de modificar la interfaz de la ventana principal

pal, con la ventana de comandos ubicada en la carpeta del proyecto, ejecute el siguiente comando: `pyuic5 -x ventanaPrincipal.ui -o ventanaPrincipal_ui.py` (ver Figura 4.3). Este comando actualiza los widgets de la interfaz grafica y los incluye como variables que posteriormente vamos a manejar vía código.



```
Anaconda Prompt (Anaconda3)
(base) C:\Users\Hidalgo>cd
(base) D:\>cd C:\Users\Hidalgo\Desktop\proyecto Eureka MSMN
(base) D:\>pyuic5 -x ventanaPrincipal.ui -o ventanaPrincipal_ui.py
```

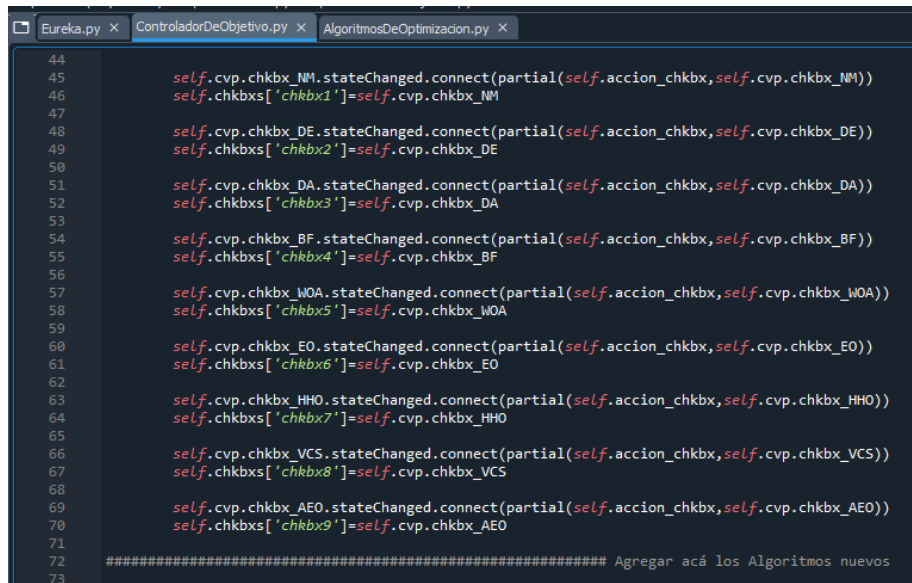
Figura 4.3: Comando de actualización de interfaz gráfica.

4.2. Modificar ControladorDeObjetivo.py

En el archivo **ControladorDeObjetivo.py** hay un comentario que indica el lugar donde se debe llevar a cabo la modificación (ver Figura 4.4).

En esta sección del código se guardan los widgets `CheckBox` en un diccionario denominado `self.chkboxs` que facilita el manejo de las interacciones del usuario con la interfaz gráfica. Acá se usan los nombres representativos de los `CheckBox` que se asignaron en la interfaz gráfica, por ejemplo, `self.cvp.chkbox.VCS` es el `CheckBox` del algoritmo `Virus Colony Search`.

Además, en esta sección se conecta las señales de cambio de estado de cada `CheckBox` con la función correspondiente. Modifique el código siguiendo el formato establecido.



```

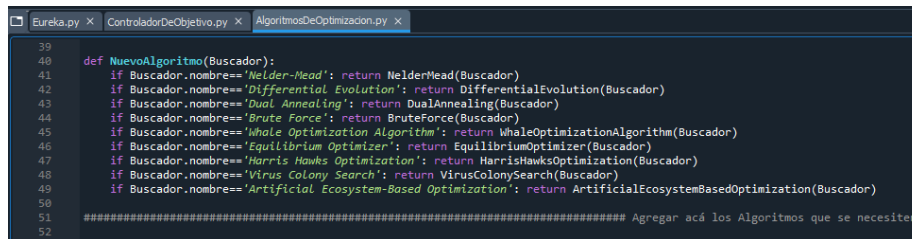
44
45     self.cvp.chkbox_NM.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_NM))
46     self.chkboxs['chkbox1']=self.cvp.chkbox_NM
47
48     self.cvp.chkbox_DE.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_DE))
49     self.chkboxs['chkbox2']=self.cvp.chkbox_DE
50
51     self.cvp.chkbox_DA.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_DA))
52     self.chkboxs['chkbox3']=self.cvp.chkbox_DA
53
54     self.cvp.chkbox_BF.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_BF))
55     self.chkboxs['chkbox4']=self.cvp.chkbox_BF
56
57     self.cvp.chkbox_WOA.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_WOA))
58     self.chkboxs['chkbox5']=self.cvp.chkbox_WOA
59
60     self.cvp.chkbox_EO.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_EO))
61     self.chkboxs['chkbox6']=self.cvp.chkbox_EO
62
63     self.cvp.chkbox_HHO.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_HHO))
64     self.chkboxs['chkbox7']=self.cvp.chkbox_HHO
65
66     self.cvp.chkbox_VCS.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_VCS))
67     self.chkboxs['chkbox8']=self.cvp.chkbox_VCS
68
69     self.cvp.chkbox_AEO.stateChanged.connect(partial(self.accion_chkbx,self.cvp.chkbox_AEO))
70     self.chkboxs['chkbox9']=self.cvp.chkbox_AEO
71
72     ##### Agregar acá los Algoritmos nuevos
73

```

Figura 4.4: Archivo ControladorDeObjetivo.py.

4.3. Modificar AlgoritmosDeOptimizacion.py

En el archivo **AlgoritmosDeOptimizacion.py** hay comentarios que indican el lugar donde se debe llevar a cabo las modificaciones (ver Figuras 4.5 y 4.6).



```

39
40 def NuevoAlgoritmo(Buscador):
41     if Buscador.nombre=='Welder-Head': return WelderHead(Buscador)
42     if Buscador.nombre=='Differential Evolution': return DifferentialEvolution(Buscador)
43     if Buscador.nombre=='Dual Annealing': return DualAnnealing(Buscador)
44     if Buscador.nombre=='Brute Force': return BruteForce(Buscador)
45     if Buscador.nombre=='Whale Optimization Algorithm': return WhaleOptimizationAlgorithm(Buscador)
46     if Buscador.nombre=='Equilibrium Optimizer': return EquilibriumOptimizer(Buscador)
47     if Buscador.nombre=='Harris Hawks Optimization': return HarrisHawksOptimization(Buscador)
48     if Buscador.nombre=='Virus Colony Search': return VirusColonySearch(Buscador)
49     if Buscador.nombre=='Artificial Ecosystem-Based Optimization': return ArtificialEcosystemBasedOptimization(Buscador)
50
51     ##### Agregar acá los Algoritmos que se necesiten
52

```

Figura 4.5: Archivo AlgoritmosDeOptimizacion.py.

Además, los nuevos algoritmos de optimización deben ser agregados como clases de python, vea ejemplos de otros algoritmos en el archivo **AlgoritmosDeOptimizacion.py** para basarse en ese formato (por ejemplo, ver la Figura 4.7).

```

915 class ArtificialEcosystemBasedOptimization():
916     def __init__(self, Buscador):
917         from mealpy.system_based.AEO import BaseAEO
918         self.BaseAEO = BaseAEO
919         self.Buscador = Buscador
920         self.Buscador.color = 'C4'
921
922     def ejecutarFijo(self):
923
924     def ejecutarMax(self):
925
926
927
928
929 #
930 ##### Agregar acá los Algoritmos que se necesiten, en formato de clases
931 #
932

```

Figura 4.6: Archivo AlgoritmosDeOptimizacion.py.

```

465 class NelderMead():
466     def __init__(self, Buscador):
467         from scipy.optimize import minimize
468         self.minimize = minimize
469         self.Buscador = Buscador
470         self.Buscador.color = 'b'
471
472     def ejecutarFijo(self):
473         Xi = self.Buscador.XoSeed(self.Buscador.dim)
474         self.respuesta = self.minimize(self.Buscador.ControladorDeFuncionAminimizar.FAM,
475                                       Xi,
476                                       method='Nelder-Mead',
477                                       options={'maxiter': None, 'maxfev': None,
478                                               'disp': False, 'return_all': False,
479                                               'initial_simplex': None, 'xatol': 0.2,
480                                               'fatol': 0.01, 'adaptive': True})
481
482         self.Xout = self.respuesta.x
483         aux = []
484         for elemento in self.Xout: aux.append(round(float(elemento)))
485         self.Xout = aux
486         self.PIRA = self.Buscador.ControladorDeFuncionAminimizar.PIRA
487
488     def ejecutarMax(self):
489         Xi = self.Buscador.XoSeed(self.Buscador.dim)
490         self.respuesta = self.minimize(self.Buscador.ControladorDeFuncionAminimizar.FAM,
491                                       Xi,
492                                       method='Nelder-Mead',
493                                       options={'maxiter': None, 'maxfev': None,
494                                               'disp': False, 'return_all': False,
495                                               'initial_simplex': None, 'xatol': 0.2,
496                                               'fatol': 0.01, 'adaptive': True})
497
498         self.Xout = self.respuesta.x
499         aux = []
500         for elemento in self.Xout: aux.append(round(float(elemento)))
501         self.Xout = aux
502         self.PIRA = self.Buscador.ControladorDeFuncionAminimizar.PIRA
503

```

Figura 4.7: Clase para el algoritmo Nelder-Mead.